UNIVERSITATEA TEHNICĂ A MOLDOVEI

CONFERINȚA TEHNICO-ȘTIINȚIFICĂ A STUDENȚILOR, MASTERANZILOR ȘI DOCTORANZILOR

Chișinău, 5-7 aprilie 2023

TECHNICAL SCIENTIFIC CONFERENCE OF UNDERGRADUATE, MASTER AND PHD STUDENTS

Chișinău, April 5-7, 2023

VOL. I

Chișinău 2023

CZU 082=135.1=111=161.1
C 65

DESCRIEREA CIP A CAMEREI NAȚIONALE A CĂRȚII DIN RM

Conferinţa tehnico-ştiinţifică a studenţilor, masteranzilor şi doctoranzilor = Technical Scientific Conference of Undergraduate, Master and PhD Students, 5-7 aprilie 2023, Chişinău / comitetul științific: Bostan Viorel [et al.]. – Chişinău: [Tehnica-UTM], 2023 – .

– ISBN 978-9975-45-956-3.

Cerinţe de sistem: PDF Reader.

Vol. 1. – 2023. – 505 p.: fig., tab. în parte color – Antetit.: Universitatea Tehnică a Moldovei. – Texte, rez.: lb. rom., engl., rusă. – Referinţe bibliogr. la sfârşitul art.

– ISBN 978-9975-45-957-0 (PDF).

082=135.1=111=161.1

C 65

Bun de tipar 29.08.2023

Comanda nr. 80

Forma electronica

MD-2004, Chișinău, bd. Ștefan cel Mare și Sfânt, 168, UTM
MD-2045, Chișinău, str. Studenților, 9/9, Editura „Tehnica-UTM”

ISBN 978-9975-45-956-3

ISBN 978-9975-45-957-0 (Vol. 1)

© Universitatea Tehnică a Moldovei, 2023

COMITETUL ȘTIINȚIFIC:

BOSTAN Viorel, Dr. hab., prof. univ., Rector al UTM, Rep. Moldova
TRONCIU Vasile, Dr.hab., prof. univ., Prorector pentru cercetare şi doctorat, UTM, Rep. Moldova;
SIMINIUC Rodica, Dr., conf.univ., Director Școala Doctorală UTM, Rep. Moldova;
BUZEA Carmen, Dr., prof.univ., Prorector pentru activitatea de cercetare științifică și informatizare, Universitatea Transilvania Brașov, România;
CEPOI Liliana, Dr., conf. univ, director IMB, UTM, Rep. Moldova;
CILOCI Rafael, Dr., conf.univ., decan FIEB, UTM, Rep. Moldova;
CIORBĂ Dumitru, Dr., conf.univ., decan FCIM, UTM, Rep. Moldova;
COTEAȚĂ Margareta, dr., conf. univ., Universitatea Tehnică „Gh. Asachi" din Iași, România;
DÎNTU Sergiu, Dr., conf.univ., decan FIMIT, UTM, Rep. Moldova;
GHIMPU Lidia, Dr., director IIEN „Dumitru Ghițu", UTM, Rep. Moldova;
GROPA Victor, Dr., conf.univ., decan FEIE, UTM, Rep. Moldova;
HAREA Olga, Dr., conf. univ., decană FUA, UTM, Rep. Moldova;
MUNTEANU Radu Adrian, Dr.ing. hab., prof.univ., Universitatea Tehnică din Cluj-Napoca, România;
SAVA Lilia, Dr., conf. univ., decană FET, UTM, Rep. Moldova;
NISTOR-LOPATENCO Livia, Dr., conf.univ., decan FCGC, UTM, Rep. Moldova;
POPA Sergiu, Dr., conf. univ., decan FȘASM, UTM, Rep. Moldova;
POPOVICI Mihail, Dr., conf. univ., decan FMV, UTM, Rep. Moldova;
RADZIUNAS Mindaugas, Dr., cerc. coord., Institutul Weierstrass, Berlin, Germania;
SUBOTIN Iurie, Dr., conf. univ., decan FTA, UTM, Rep. Moldova;
TÎRȘU Mihai, Dr., director IE, UTM, Rep. Moldova;
TRONCIU Sergiu, Dr., conf.univ., decan FD, UTM, Rep. Moldova.

SCIENTIFIC COMMITTEE:

BOSTAN Viorel, Dr.hab., Prof., Rector of TUM, Rep. of Moldova.
TRONCIU Vasile, Dr.hab., Prof., Vice-rector for research and doctorate, TUM, Rep. of Moldova;
SIMINIUC Rodica, Dr., Assoc. prof., Director of the PhD School, TUM, Rep. of Moldova;
BUZEA Carmen, Dr., Prof., Vice-rector for scientific research activity and informatization, Transilvania University of Brașov, Romania;
CEPOI Liliana, Dr., Assoc. Prof., Director IMB, TUM, Rep. of Moldova;
CILOCI Rafael, Dr., Assoc. prof., Dean of FEEB, TUM, Rep. of Moldova;
CIORBĂ Dumitru, Dr., Assoc. prof., Dean of FCIM, TUM, Rep. of Moldova;
COTEAȚĂ Margarita, Dr., Gheorghe Asachi Technical University of Iasi, Romania;
DÎNTU Sergiu, Dr., Assoc. prof., Dean of FMET, TUM, Rep. of Moldova;
GHIMPU Lidia, Dr., director, Ghitu IEIN, TUM, Rep. of Moldova;
GROPA Victor, Dr., Assoc. prof., Dean of FPEE, TUM, Rep. of Moldova;
HAREA Olga, Dr., Assoc. prof., Dean of FAUP, TUM, Rep. of Moldova;
MUNTEANU Radu Adrian, Dr. ing. hab., Prof., Technical University of Cluj-Napoca, Romania;
SAVA Lilia, Dr., Assoc. prof., Dean of FET, TUM, Rep. of Moldova;
NISTOR-LOPATENCO Livia, Dr., Assoc. prof., Dean of FCGC, TUM, Rep. of Moldova;
POPA Sergiu, Dr., Assoc. prof., Dean of FAFAS, TUM, Rep. Moldova;
POPOVICI Mihail, Dr., Assoc. prof., Dean of FVM, TUM, Rep. Moldova;
RADZIUNAS Mindaugas, Dr., Senior scientist, Weierstrass Institute Berlin, Germany;
SUBOTIN Iurie, Dr., Assoc. prof., Dean of FFT, TUM, Rep. of Moldova;
TÎRȘU MIHAI, Dr., Director, IE, TUM, Rep. of Moldova;
TRONCIU Sergiu, Dr., Assoc. prof., Dean FD, TUM, Rep. of Moldova

COMITETUL ORGANIZATORIC:

CUJBA Rodica, Dr., Șef Direcția Management Cercetări Științifice, UTM, Rep. Moldova;

BATÎRU Grigorii, Dr., conf. univ., FȘASM, UTM, Rep. Moldova;

BRAGUȚA Eugeniu, Dr., conf. univ., Dep. Ingineria Infrastructurii Transporturilor, FUA, UTM, Rep. Moldova;

COCIU Valeriu, Dr., conf. univ., FMV, UTM, Rep. Moldova;

CREȚU Vasilii, Dr., conf. univ., Dep. Microelectronică și Inginerie Biomedicală, FCIM , UTM, Rep. Moldova;

GUȚU-CHETRUȘCA Corina, Dr., lect. univ., Dep. Energetică, FEIE; UTM, Rep. Moldova;

MALCOCI Iulian, Dr., conf. Dep. Bazele Proiectării Mașinilor, FIMIT , UTM, Rep. Moldova;

PALADI Daniela, Dr., conf. Dep. Alimentație și Nutriție, FTA;

SANDULEAC Ionel, Dr., conf., Dep. Fizică, FET , UTM, Rep. Moldova;

SCRIPCENCO Angela, Dr., conf., Dep. Design și Tehnologii în Textile, FD, UTM, Rep. Moldova;

ȘONTEA Victor, Dr., prof. univ., Dep. Microelectronică și Inginerie Biomedicală, FCIM , UTM, Rep. Moldova;

ȚIBICHI Viorica, Dr., conf. univ., Dep. Inginerie Civilă și Geodezie, FCGC , UTM, Rep. Moldova;

ȚURCAN Iuliu, Dr., conf. univ., Dep. Economie și Management, FIEB , UTM, Rep. Moldova.

ORGANISING COMMITTEE:

CUJBA Rodica, Dr., Head, Dep. of Management of Scientific Research, TUM, Rep. of Moldova;

BATÎRU Grigorii, Dr., Assoc. prof., Dep. AM, FAFAS, TUM, Rep. of Moldova;

BRAGUȚA Eugeniu, Dr., Assoc. prof., Dep. TIE, FAUP TUM, Rep. of Moldova;

COCIU Valeriu, Dr., Assoc. prof., Dep. FCS, FVM, TUM, Rep. of Moldova;

CREȚU Vasilii, Dr., Assoc. prof., Dep. MIB, FCIM, TUM, Rep. of Moldova;

GUȚU-CHETRUȘCA Corina, Dr., Assoc. prof., Dep. PE, FEEE, TUM, Rep. of Moldova;

MALCOCI Iulian, Dr., Assoc. prof., Dep. BPM, FMET, TUM, Rep. of Moldova;

PALADI Daniela, Dr., Assoc. prof., Dep. AN, FFT, TUM, Rep. of Moldova;

SANDULEAC Ionel, Dr., Assoc. prof., Dep. Phys, FET, TUM, Rep. of Moldova;

SCRIPCENCO Angela, Dr., Assoc. prof., Dep. DTT, FD, TUM, Rep. of Moldova;

ȘONTEA Victor, Dr., Univ. prof., Dep. MIB, FCIM, TUM, Rep. of Moldova;

ȚIBICHI Viorica, Dr., Assoc. prof., Dep. ICG, FCGC, TUM, Rep. of Moldova;

ȚURCAN Iuliu, Dr., Assoc. prof., Dep. EM, FEEB, TUM, Rep. of Moldova.

CUPRINSUL / CONTENTS

SECȚIA ELECTRONICĂ ŞI TELECOMUNICAȚII
SECTION OF ELECTRONICS AND TELECOMMUNICATIONS

PROIECTAREA REȚELEI DE COMUNICAȚII MOBILE CELULARE ÎN DOMENIUL FRECVENȚĂ
Ilie AXENTI, Ana BURLAC 13
INTERACȚIUNEA DISPOZITIVELOR DIN ECOSISTEMA COMPANIEI APPLE
Igor FORTUNA 16
CONTROLUL DINAMICII FOTONILOR EMIȘI ÎN MEDIILE BIOLOGICE
Natalia GUBCEAC 19
ANALIZA CAPITALULUI FIX ȘI PROBLEMELE GENERALE DE UTILIZARE LA ÎNTREPRINDEREA USSS
Mihaela MĂMĂLIGĂ 24
CERCETAREA PARTICULARITĂȚILOR DE ELABORARE ȘI IMPLEMENTARE A SISTEMELOR SATELITAR DE TELEVIZIUNE DIGITALĂ
Mihail MEREUȚĂ 28
ANALIZA COSTURILOR DE PRODUCȚIE LA ÎNTREPRINDEREA DE TELECOMUNICAȚII
Iulita POPESCU 33
DEZVOLTAREA SISTEMULUI EFICIENT DE CONTROL AL ECHIPAMENTELOR TEHNOLOGICE ÎN PRODUCERE A CIMENTULUI
Ion RUSNAC 37
WI – FI 7 – NOUA INOVAȚIE ÎN SFERA COMUNICAȚIILOR WIRELESS
Victor SCLIFOS 42
METODOLOGIA DETERMINARII NIVELULUI DE RADIAȚIE EMIS DE TELEFOANELE MOBILE ŞI ECHIPAMENTELE RADIO. SAR (SAR - SPECIFIC ABSORPTION RATE)
Ariana ȚENU, Andrei APOSTOL 45
INTELIGENȚA ARTIFICIALĂ ȘI IMPACTUL ASUPRA PROCESULUI EDUCAȚIONAL
Valeria ȚURCAN 49
TEHNOLOGII DE ANULARE ACTIVE A ZGOMOTULUI
Victor ZMUNCILĂ 53

SECȚIA ENERGETICĂ ȘI INGINERIE ELECTRICĂ
SECTION OF ENERGETICS AND ELECTRICAL ENGINEERING

AUTOMATIZAREA PROCESULUI TEHNOLOGIC DE SORTARE A UNUI SET DE OBIECTE DUPĂ CULORI
Florin TESLARI, Corneliu RĂILEANU, Valentin BÂNZARU, Vladislav ENACHI 57
PROVOCĂRI ȘI OPORTUNITĂȚI DE DEZVOLTARE SUSTENABILĂ A PV-URILOR PE ACOPERIȘUL CLĂDIRILOR
Ion CARAI 61
RESULTS OF EXPERIMENTAL AND COMPUTATIONAL STUDIES FOR STEAM TURBINE EXHAUST HOODS APPLYING A WIDE-MODE DEFLECTOR
Alexander LAPUZIN, Valery SUBOTOVICH, Ivan MALYMON 67
IMPACT OF QUALITY INDICATORS OF ELECTRICAL ENERGY ON METROPOLITAN ROLLING STOCK
Roman DEMIANENKO 74

ANALYSING HOUSEHOLD CONSUMERS' DEMAND RESPONSE THROUGH BEHAVIOURAL LOAD PROFILES
Vladyslav KIIANCHUK 80
АНАЛИЗ РАБОТЫ КОНДЕНСАЦИОННО-ОХЛАДИТЕЛЬНОГО УЗЛА В УСЛОВИЯХ ПОВЫШЕННОЙ ТЕМПЕРАТУРЫ НАРУЖНОГО ВОЗДУХА
Андрей ГЕЛДЫШЕВ 84
PORTOFOLIU STANDARDELOR ENERGIILOR REGENERABILE. PRINCIPII GENERALE
Ruslan BALACCI 89
SUSTENABILITATEA UTILIZĂRII LITIULUI PENTRU BATERII CU ACUMULATOARE DE ENERGIE ELECTRICĂ
Marcel ROTARU 95
MODELING ELEMENTS of A BASIC CLOUD SUBSTATION
Vladyslav TSIUPA, Andrii POTRYVAI 99
IMPLEMENTAREA SOFTULUI PENTRU OPTIMIZAREA PARAMETRILOR TEHNICI A SISTEMELOR ELECTROMECANICE DISTRIBUITE ÎN CADRUL STAȚIEI ORĂȘENEȘTI DE EPURARE A APELOR UZATE
Artiom MOLDOVAN 105
ABOUT UNDERGROUND OVERHEAD POWER LINE
Roman GANUS, Andrii POTRYVAI 112

SECŢIA CALCULATOARE, INFORMATICĂ ŞI MICROELECTRONICĂ
SECTION OF COMPUTERS, INFORMATICS AND MICROELECTRONICS

SUBSECŢIA INGINERIA ŞI ŞTIINŢA CALCULATOARELOR
ENGINEERING AND COMPUTER SCIENCE SUBSECTION

DOMAIN-SPECIFIC LANGUAGE FOR ANALYZING MEDICAL RESULTS
Maia ZAICA, Felicia LUPAȘCU, Cristian BRÂNZA, Nichifor POPESCU 118
SMART CITY – O SIMBIOZĂ DINTRE MEDIU, TEHNOLOGII, DATE, CUNOȘTINȚE ȘI CETĂȚENI
Ana ȚURCAN 123
DOMAIN SPECIFIC LANGUAGE FOR PLC
Inga PALADI, Vladislav FRUNZE, Vlad URSU, Roman TRANDAFIR, Irina RACOVCENA 126
THE DEVELOPMENT OF A DOMAIN SPECIFIC MARKUP LANGUAGE FOR INTERACTIVE STORYTELLING
Denis PRODAN , Rodica PRODAN, Sandu GAZEA, Daniel BUCĂTARU, Nicolae CHIPERI 130
SINTEZA ALGORITMULUI DE REGLARE LA MODELUL OBIECTULUI CUANTICIPAȚIE-ÎNTÂRZIERE DE ORDINUL TREI CU OSCILAȚII
Dumitru MORARU 134
SISTEM DE CALCUL COLABORATIV PENTRU COMANDA ROBOȚILOR MOBILI
Olesea BOROZAN, Constantin ABABII, Victor LAȘCO, Ianuș FORTUNA, Vasile IONIȚA 138
MARKUP LANGUAGE FOR LABEL DESIGN
Călin RADU, Maxim COJOCARI-GONCEAR, Sorin IAȚCO, Nichita CHIHAI, Andrei BARDIER 144

CREATING A GENERAL PURPOSE LANGUAGE: DESIGN AND IMPLEMENTATION OF PRISM
Valeria COZLOV, Tudor SÎRGHI, Beatricia GOLBAN, Ion VASLUIAN 148
DOMAIL SPECIFIC LANGUAGE FOR COMPUTING TAXES
Diana BELAIA, Corneliu CATLABUGA, Ana BZOVII, Igor MIHAILUIC, Constantin BAXANEAN 151
FRACLANG - A DOMAIN SPECIFIC LANGUAGE FOR BUILDING FRACTALS
Ciprian BOTNARI, Grigore GUZUN, Emanuil CUCOȘ, Artiom MIROVSKI 156
DOMAIN SPECIFIC LANGUAGE FOR UNIT TESTING
Andreea CHIPER, Denis SMOCVIN, Cezar GUZUN, Vladimir LUCHIAOV, Anatolie TELUG 160
DOMAIN SPECIFIC LANGUAGE FOR PETRI NETS
Maxim CERNETCHI, Corneliu NASTAS, Alexandr CARA, Vladislav CRUCERESCU 166
GENERAL PURPOSE LANGUAGE
Maria PROCOPII, Alexei CIUMAC, Dorin OTGON, Andreea MANOLE, Alexandru DOBROJAN 171
JADE - A GENERAL PURPOSE LANGUAGE
Dinu GUȚU, Adelia BRĂGUȚA, Andrei SĂRĂTEANU, Alexandra KONJEVIC, Milena COJUHAR 175
MARKUP LANGUAGE FOR PSYCHOLOGICAL ARCHETYPES
Alexandru BUZU, Bogdan ZLATOVCEN, Dumitru MORARU, Maria LEȘENCO, Marius BOTEZATU 181
ANALIZA METODELOR DE ACORDARE A REGULATORULUI LA MODELUL OBIECTULUI CU ANTICIPAŢIE-ÎNTÂRZIERE DE ORDINUL DOI CU ASTATISM
Dumitru MORARU 184
DOMAIN SPECIFIC LANGUAGE FOR SOUND PROCESSING
Mihail ECHIM, Anastasia CUNEV, Iulian BERCU, Stefan NISTOR 188
IMPLEMENTAREA ALGORITMILOR DIN INTELIGENȚA ARTIFICIALĂ PENTRU DEPĂȘIREA DECALAJULUI DE COMUNICARE ÎNTRE PERSOANELE CU DIZABILITĂȚI (SURDO-MUȚI) ȘI OAMENII FĂRĂ DIZABILITĂȚI
Dragoș COJOCARI 192
MARKUP LANGUAGE FOR DESIGNING DIAGRAMS
Maria AFTENI, Sevastian BAJENOV, Mihaela CUȘNIR, Alexandru FURDUI, Gabriel GÂTLAN 198
ANALIZA CATEGORICĂ A ALGORITMILOR DE SORTARE
Petru CERVAC 203
VINSENT: AN IOT SOLUTION FOR OPTIMAL WINE BARREL MANAGEMENT IN MOLDAVIAN WINERIES
Vlada MAGAL, Irina TIORA, Viorel NOROC, Marcel VLASENCO, Maria-Mădălina UNGUREANU 210
DOMAIN SPECIFIC LANGUAGE FOR GEOMETRIC CALCULATIONS
Ion ROTARU, Alexandru GRAMA, Ahmed AL HAJ, Mihai CORETCHI, Maxim ZADAROJNII 213

NAVIGAREA LA INTERSECȚIA DINTRE TEHNOLOGIILE EMERGENTE, ANALIZA AVANSATĂ A DATELOR ȘI SECURITATEA CIBERNETICĂ: STRATEGII PENTRU ERA DIGITALĂ
Alexandr DIMOGLO 218
ANALIZA EMPIRICĂ A REȚELELOR SERIE-PARALEL VERSUS PARALEL-SERIE CU METODA MONTE CARLO PENTRU DISTRIBUȚIA EXPONENȚIALĂ A ELEMENTELOR REȚELELOR
Veronica ANDRIEVSCHI-BAGRIN 222
MODELING AND SIMULATION OF A K-MEANS CLUSTERING ALGORITHM IN MATLAB AND PYTHON
Ștefana DUȚĂ, Laura-Elena DOROBANȚU, Dan-Gabriel RADU 228
DESIGN OF DOMAIN SPECIFIC LANGUAGE FOR ASTROLOGICAL CHARTS GENERATION - ALAKIRQL
Alexandru ANDRIEȘ, Georgeana GLOBA, Arteom KALAMAGHIN, Dionisie SPATARU 233
THE STRUCTURE OF THE ISMS DOCUMENTATION IN ACCORDANCE WITH UPDATES TO ISO 27001:2022, 27002:2022
Dmitrii OBUH 237
ПРИМЕНЕНИЕ 3D-МОДЕЛИРОВАНИЕ В НЕЙРОХИРУРГИИ ДЛЯ УЛУЧШЕНИЯ ПЛАНИРОВАНИЯ И ПРОВЕДЕНИЯ ОПЕРАЦИЙ НА ГОЛОВНОМ МОЗГЕ И СПИННОМ МОЗГЕ
Анатолий ГУМЕНЮК, Ростислав КЭЛИН 242
UNLOCKING THE POWER OF QUERY LANGUAGE
Cristina ȚĂRNĂ, Tudor SCLIFOS, Dan HARITON, Leonid CAMINSCHI, Andrei CEBAN 246
SUBCONȘTIINȚA ARTIFICIALĂ
Vadim STRUNĂ 250
WHAT IS SOCIAL ENGINEERING AND HOW TO PREVENT IT?
Alexandru TODOS 256

SUBSECŢIA MICROELECTRONICĂ ŞI NANOTEHNOLOGII
MICROELECTRONICS AND NANOTECHNOLOGIES SUBSECTION

REALIZATION OF H2S GAS SENSOR USING SOL-GEL PROCESSED ZnFe2O4 THIN FILM
Rajat, Amit Kumar SHRINGI, Mahesh KUMAR 261
PROPRIETĂȚILE NANOBAGHETELOR DE ZnO:Sn
Tudor ZADOROJNEAC, Dinu LITRA, Philipp SCHADTE, Cristian LUPAN 265
EFECTUL DE PROXIMITATE ÎN MICROELECTRONICA SUPRACONDUCTOARE
Vladimir BOIAN 270
CONTROLUL INTELIGENT A TEMPERATURII DE LA DISTANȚĂ ELABORAT PENTRU TRATAMENTUL TERMIC AL FOLIILOR DE ZINC
Cătălin CRECIUNEL 274
OBȚINEREA ȘI STUDIUL STRATURILOR DE OXID DE ZINC PRIN CORODAREA ELECTROCHIMICĂ A FOLIILOR DE ZINC
Cătălin CRECIUNEL 279
IMPLEMENTATION OF A SOURCE METER CONTROL SYSTEM WITH PYTHON USING SCPI AND VISA
Alexandr SEREACOV, Adrian BIRNAZ 284

LUMINISCENȚA SOLUȚIEI COLOIDALE ALE PUNCTELOR CUANTICE DE SnS_2
Ecaterina CRISTEA 290
РЕАЛИЗАЦИЯ ДОЗАТОРА ДЛЯ ПРОХЛАДИТЕЛЬНЫХ НАПИТКОВ НА МИКРОКОНТРОЛЛЕРЕ STM32 С ПОМОЩЬЮ КОНЕЧНОГО АВТОМАТА
Татьяна МАСЛОВА 294
SISTEM DEPARTAMENTAL DE MANAGEMENT A COMPONENTELOR ELECTRONICE
Rodica POPA, Maxim CHIRIAC 298
SISTEM HOLOGRAFIC DE AFIȘARE A INFORMAȚIEI
Maxim CHIRIAC 303
EXFOLIEREA MATERIALELOR 2D SnS, SnS_2 ȘI SnSe PRIN METODE ȘI TEHNICI COST-EFECTIVE
Mihail BATIRI 307
ELECTROLYTE MATTERS: UNDERSTANDING ITS IMPACT ON SUPERCAPACITOR PERFORMANCE
Barnika CHAKRABORTY, Mangesh DESAI, Kirandeep SINGH 311
STUDIEREA SUPRACONDUCTIBILITĂȚII ÎN NANOSTRUCTURI DE NIOBIU-COBALT
Cezar-Casian MALCOCI 315
TESTER DE RESPIRAȚIE CU HIDROGEN
Mihai BRÎNZĂ 319
PROCESOR DIGITAL DE SUNET PE BAZA MICROCONTROLERULUI STM32F446RE
Andrei SPATARI, Adrian BÎRNAZ 331

SUBSECȚIA BAZE DE DATE
DATABASE SUBSECTION

MEMCACHED – РАСПРЕДЕЛЕННАЯ СИСТЕМА КЭШИРОВАНИЯ ПАМЯТИ
Николай СКУРТУ 337
ТЕХНОЛОГИЯ ENTITY FRAMEWORK CORE
Алексей КОЗЬМА 341
ОПТИМИЗАЦИЯ ЗАПРОСОВ SQL (MySQL)
Эдуард ЦУРКАН 345
РАСПРЕДЕЛЕННАЯ СУБД APACHE CASSANDRA
Яна ЗАБОЛОТНЫЙ 348
INTERACȚIUNEA DINTRE ENTITY FRAMEWORK ȘI BAZE DE DATE
Victor COVRIG 352
ТРАДИЦИОННАЯ И ОБЛАЧНАЯ АРХИТЕКТУРА ХРАНИЛИЩ ДАННЫХ
Лучия НИГАЙ 356
POSTGRESQL
Анатолий ГАРБУЗОВ 360
СУБД «DB2» IBM
Дмитрий РОМАНЕНКО 364
GRAPHQL – ЯЗЫК ЗАПРОСОВ И МАНИПУЛИРОВАНИЯ ДАННЫМИ С ОТКРЫТЫМ ИСХОДНЫМ КОДОМ
Роман ПОГОНЕЦ 368
SISTEME DE STOCARE A DATELOR: INNODB VS MYISAM
Valentina ASTAFI 371

ВЗАИМОДЕЙСТВИЕ R И SQL SERVER
Рувим БОЗАЖИ 375
НОВОВВЕДЕНИЯ MICROSOFT SQL SERVER
Владимир ВЕРЕТКО 377
КЛАССИФИКАЦИЯ БАЗ ДАННЫХ
Олег БОГАЧ 380
SQL SERVER BUSINESS INTELLIGENCE
Максим ГОДУНОВ 383
INTERBASE — РЕЛЯЦИОННАЯ СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ
Вадим ПОТРЫМБА 388
INFLUENȚA VOLUMULUI DE DATE ASUPRA TIMPULUI DE EXECUȚIE A DIFERITOR TIPURI DE INSTRUCȚIUNI ÎN SQL SERVER
Irina GRIGORAȘ 392
БЛОКЧЕЙН
Николай ШАРАФУДИНОВ 396
FIREBASE
Руслан ЦЫПУШТАНОВ 400
SQLite – КОМПАКТНАЯ ВСТРАИВАЕМАЯ СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ
Светлана СОЛТЫК 403
OLAP СИСТЕМА АНАЛИЗА
Никита МАКЕЕВ 407
POWER BI: ВИЗУАЛИЗАЦИЯ И ИСТОЧНИКИ ДАННЫХ
Елисавета ДМИТРИЕВА 412
MICROSOFT SQL SERVER DATA TOOLS - BUSINESS INTELLIGENCE (SSDT-BI)
Светлана КРАЧУН 416
EXECUTAREA UNEI CERERI (QUERY) în MS SQL SERVER
Ion MORARU 422
RISCURILE ASOCIATE CU UTILIZAREA SQL. SQL INJECTION
Cristian DICOL 426
ИСПОЛЬЗОВАНИЕ МАШИННОГО ОБУЧЕНИЯ В УПРАВЛЕНИИ БАЗАМИ ДАННЫХ
Максим СТРОПША 430
SQL SERVER INTEGRATION SERVICES VS AZURE DATA FACTORY
Xenia DAMANCIUC 435
TRANZACȚIILE ÎN SQL
Mariana TROFIN 439
ПОСТРОЕНИЕ И ОПТИМИЗАЦИЯ ЗАПРОСОВ SQL
Денис ЛУПАШКО 442
ANALIZAREA MODELELELOR DE DATE OBIECT-RELAȚIONALE DJANGO ȘI DOCTRINE
Mihail ULINICI 444
AMAZON OPENSEARCH SERVICE
Vadim VICOL 447

АВТОМАТИЗАЦИЯ ПРОЦЕССОВ НА ПРЕДПРИЯТИИ С ПОМОЩЬЮ 1С
Илья ЯМБОГЛО 451
TEHNOLOGIA DBLINK ÎN POSTGRESQL
Dumitru BEȘELEA 455
GOOGLE CLOUD SQL
Serghei BUGA 458
СРАВНЕНИЕ БЛОКЧЕЙН И БАЗЫ ДАННЫХ
Григорий ЧАГЛЕЙ 461
TEHNICI DE RECUPERARE A DATELOR ÎN SQL SERVER
Eugeniu ROMANESCU 465
OPTIMIZAREA OPERAȚIUNILOR SQL CU AJUTORUL LIMBAJULUI PYTHON
Daria NEDELCO 468
СЕРИАЛИЗАЦИЯ ТРАНЗАКЦИЙ В БАЗАХ ДАННЫХ
Александр ЛУПУ 472
PARTIȚIONAREA BAZELOR DE DATE
Victor ȘAPTEFRAȚI 476
LIMBAJE PENTRU BAZE DE DATE
Ana CASTRAVEȚ 480
АДМИНИСТРИРОВАНИЕ БАЗ ДАННЫХ
Александр ОСИНСКИЙ 482
ПОДХОДЫ CODE FIRST, DATABASE FIRST И MODEL FIRST В СОЗДАНИИ БАЗЫ ДАННЫХ ИСПОЛЬЗУЯ ENTITY FRAMEWORK
Роман КИОРОГЛО 486
UTILIZAREA DIVERSELOR TEHNOLOGII BD ÎN DEPENDENȚĂ DE SCOP
Filip FLORIN 491
INIȚIERE ÎN TEORIA BAZELOR DE DATE
Ariadna ONISIM 495
СУБД MS ACCESS
Адриана БУШУЕВ 499
APACHE CASSANDRA
Яна ТОПАЛ 502

SECȚIA ELECTRONICĂ ŞI TELECOMUNICAŢII

SECTION OF ELECTRONICS AND TELECOMMUNICATIONS

PROIECTAREA REȚELEI DE COMUNICAȚII MOBILE CELULARE ÎN DOMENIUL FRECVENȚĂ

Ilie AXENTI[1*], Ana BURLAC[2]

[1,2]*Departamentul Telecomunicații și Sisteme Electronice, grupa RST- 201, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova.*

*Autorul corespondent: Ilie Axenti, ilie.axenti@tse.utm.md

Coordonator științific: Ion AVRAM, dr., conf.univ, *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova.*

Rezumat. *În lucrare se efectuează proiectarea rețelei de comunicații mobile celulare in domeniul frecvență. Pentru aceasta se îndeplinesc următoarele proceduri: control egalitate module frecvențe emisie-recepție, calcul număr total de canale, calcul număr canale de trafic, formarea seturilor de canale pentru fiecare celulă a clusterului , construirea canalului duplex dat.*

Cuvinte cheie*: Rețea, comunjicații, mobile, canal, duplex, proiectare, frecvență.*

Introducere

Scopul lucrării constă în descrierea procedurii de proiectare a rețelei mobile celulare în domeniul frecvență.

Date inițiale:

Să se proiecteze o rețea celulară de comunicații mobile conform următoarelor date inițiale:
i=4,
j=1,
Sistem GSM 900,
ΔF1=890,55...912 [MHz]
ΔF2=935,55...957 [MHz]
Δfc=200 kHz = 0,2 [MHz]

Să se efectueze: construirea seturilor de canale pentru fiecare celulă a clusterului, să se construiască canalul duplex 3. Procesul de proiectare constă din următoarele etape:

1.Controlul egalității modulelor benzilor de frecvență $|\Delta F_1|$ și $|\Delta F_2|$;

$|\Delta F_1|$=912 - 890,55[MHz] =21,45 [MHz],
$|\Delta F_2|$=957 - 935,55[MHz] =21,45 [MHz].
Egalitatea se adevereşte, adica $|\Delta F_1| = |\Delta F_2|$ și se efectuează calculele propriu-zise.

2. Se calculează numărul total N al canalelor în sistem după formula:

$$N = \frac{|\Delta F1|}{\Delta fc} = \frac{|\Delta F2|}{\Delta fc} \quad (1)$$

și se obține

$$N = \frac{21{,}45\text{MHz}}{0{,}2\text{MHz}} = 107{,}25 \text{ canale;}$$

În calcule se va utiliza doar partea intreagă a rezultatului obținut N=107 canale, iar partea fracționară nu se utilizează și va fi indicată doar pe planul de frecvențe.

3. Se calculează numărul canalelor de trafic în sistem din formula:

$$N_{tr} = N - 1 \quad (2)$$

$$N_{tr} = 107 - 1 = 106 \text{ canale de trafic.}$$

4. Se calculează numărul k de celule în cluster după formula:

$$\mathrm{k} = i^2 + \mathrm{i} \cdot \mathrm{j} + j^2 \quad (3)$$

înlocuind se obține $k = 4^2 + 4 * 1 + 1^2 = 21$ celule.

5. Se calculează numărul canalelor de trafic în fiecare celulă a clusterului după formula:

$$N_{tr/celulă} = \frac{N_{tr}}{k} \quad (4)$$

Substituind, obținem $N_{tr/celulă} = \frac{106}{21} = 5{,}04$ număr canale de trafic pe celulă.
Din rezultatul obținut putem face următoarele concluzii:

a) În unele celule ale clusterului se va utiliza un număr de canale de trafic egal cu partea întreagă a rezultatului obținut,

adica $N_{tr/cel} = 5$,

b) În celelalte celule ale clusterului se va utiliza un număr de canale de trafic egal cu partea întreagă a rezultatului obținut plus 1,

adica $N_{tr/cel} = 5 + 1 = 6$.

6. Se formează seturi de canale pentru fiecare celulă a clusterului după metoda tabelară.

Tabelul 1

Seturi canale pentru fiecare celulă a clusterului

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
									52											
									73											
									94	95	96	97	98	99	100	101	102	103	104	105
106																				

Din tabel se fac concluziile:

1) Setul de canale numărul 10 conține următoarele canale:10,31,52,73,94.
2) În celula 1 se conțin 6 canale de trafic.
3) În celulele cu numerele de la 2 până la 21 se conțin câte 5 canale de trafic.

7. Se construiește planul de frecvențe al RCM proiectate și canalul duplex nr=3.

Canalul duplex reprezintă canalul de trafic prin care informația utilă poate fi transmisă concomitent în ambele direcții, adică atât de la ME→BS, cât și de la BS→MS. Acel 1 canal care s-a scăzut la etapa 3, s-a împărțit îndouă părți egale, care au format două benzi de protecție. În Fig. 1 este indicat planul de frecvențe și canalul duplex nr.n 3.

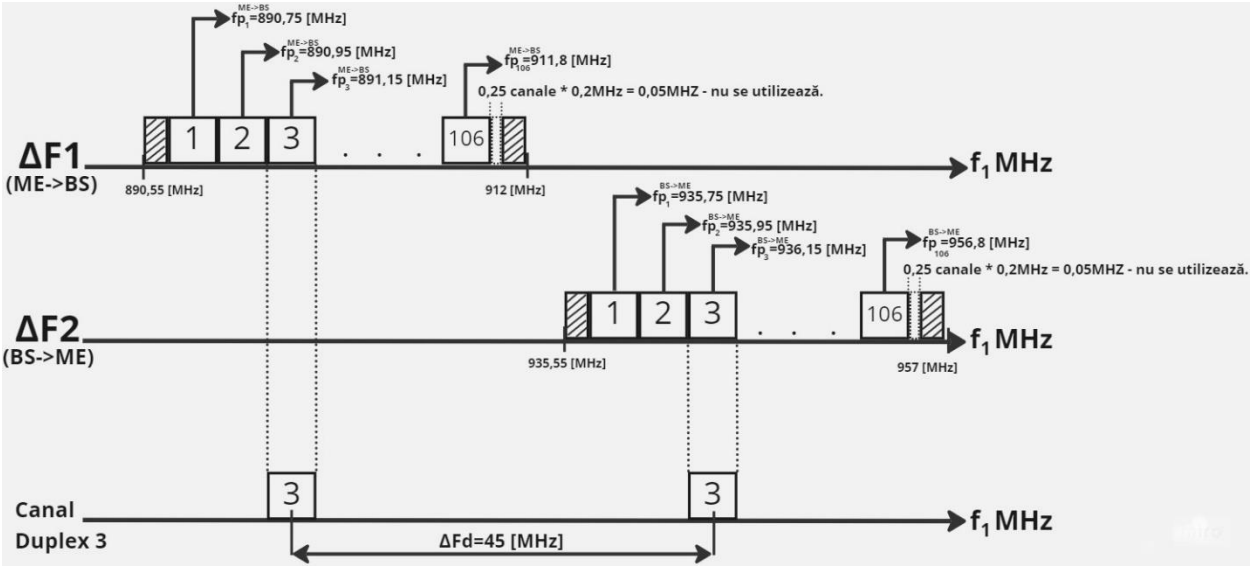


Figura 1. Construirea planului de frecvențe și a canalului duplex nr = 3

ΔFd – departajare în frecvență a purtătoarelor canalelor simplex al unui canal duplex.

Concluzii

În lucrare a fost efectuat :

a) Contorlul egalității modulelor $|\Delta F_1|, |\Delta F_2|$. Modulele sunt egale și au valoare de 21,45 [MHz],

b) Calculul numărului de celule in cluster și s-a obținut valoare k=21,

c) S-au construit seturi de canale pentru fiecare celulă a clusterului.

d) S-a construit planul de frecvențe și canalul duplex nr.3.

Referințe

1. Ion Avram. Comunicaţii mobile. Ciclu de prelegeri. Chișinău: Secţia Redactare şi Editare a UTM, 2011. - 103 p.
2. Ion Avram. Îndrumar metodic Lucrarea individuală nr.1. Proiectarea în frecvență a rețelelor de comunicații mobile. Chișinău: Secția editare și redactare a UTM, 2022 - 14 p.

INTERACȚIUNEA DISPOZITIVELOR DIN ECOSISTEMA COMPANIEI APPLE

Igor FORTUNA

Departamentul Telecomunicații și Sisteme Electronice, grupa , Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova.

*Autorul corespondent Igor Fortuna, igor.fortuna@tse.utm.md

Coordonator științific: Roman GRIȚCO, *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

Rezumat. *Crearea unui ecosistem este esențial pentru o companie. Cea mai mare problema este interacțiunea între toate dispozitivele create..Un progress eneorm în această direcție a fost efectuat de compania Apple. Când vă conectați la ID-ul Apple pe toate dispozitivele dvs., aveți deschis liberul acces la toate oportunitățile oferite de companie pentru interacțiunea ușoară, liberă și accesibilă a dispozitivelor. În această prezentare o să aflați mai multe posibilităti de interacțiune între toate dispozitivele companiei, cum ar fi cum să vă deblocați automat Mac-ul atunci când purtați Apple Watch-ul sau cum să folosiți iPhone-ul ca cameră web pentru Mac.*

Cuvinte cheie*: Ecosistem, interacțiune, Apple.*

Introducere

Apple este una dintre companiile care depun un efort pentru a orefei calitate, confort si disponibilitate clienților săi. Pentru a menține clienții în cadrul companiei, apple a creat un ecosistem de invidiat. Multe companii încearcă să recreeze acest ecosistem pentru a ușura utilizarea dispozitivelor create de ei insă fără success, deoarece compania apple a făcut nu doar interacțiunea între dispozitivele ei foarte simplă si usoară, dar și a optimizat toate aplicațiile create de companie pentru o utilizare ușoară în timp accesării acestora . Printre produsele și serviciile cele mai cunoscute și de succes, se enumeră: iPhone, iPad, iPod, Mac și iTunes.

Ecosistema Apple. Caracteristici generale.

Un ecosistem Apple este utilizarea colectivă a hardware-ului, software-ului și a serviciilor Apple care pot funcționa și interacționa între ele. Produsele sale au avut un asalt de vînzări mari chiar și după pandemia globală. În ciuda faptului că sunt scumpe, oamenilor le place să cumpere cu bucurie produse Apple în fiecare an, argumentând alegrea datorită securizării datelor și a serviciilor. Unul dintre principalele motive care stă în spatele acestei popularităţi și achiziționării într-un număr valuros a produselor Apple este datorită ecosistemului său. Acesta include diverse dispozitive Apple, cum ar fi iPhone- un smartphone care rulează pe sistemul de operare iOS, dezvoltat de Apple; MacBook- un computer desktop sau laptop care rulează pe sistemul de operare macOS; iPad-o tabletă care rulează pe sistemul de operare iOS; AirPods-căști wireless care se conectează la dispozitivele Apple prin Bluetooth; Apple Watch-un ceas inteligent care se conectează la iPhone și permite utilizatorilor să primească notificări, să monitorizeze activitatea fizică și să utilizeze diverse aplicații. , Homepod și multe altele. Companiile concurente au încercat să imite acest ecosistem, dar nimic nu a putut funcționa la fel de bine ca acel realizat de Apple.

Utilizarea unui singur produs Apple nu se referă la ecosistemul Apple; este vorba despre utilizarea întregului grup de tehnologie împreună. Și dacă utilizați produsele Apple sincron unul cu celălalt, înseamnă că vă aflați într-un ecosistem Apple. De exemplu, un MacBook poate fi mult mai util dacă toate notele, mesajele, fotografiile etc. se sincronizează automat cu iPhone-ul folosit. Primirea unui apel telefonic pe ceasul Apple atunci când sunteți prea ocupat pentru a utiliza telefonul și utilizarea Siri pentru o experiență hands-free oferă experiența supremă.

Cele mai utilizate servicii din cadrul ecosistemului.

AirDrop. Este una din principalele tehnologii create de companie. La început telefoanele companiei erau discriminate din cauza lipsei posibilității de a transmite date prin Bluetooth, dar pe parcursul timpului a fost creat unul din cele mai rapide, sigure si ușoare în folosință posibilitate de a trsansmite date de toate timpurile, ba chiar mai mult o dată cu trimiterea datelor,fotografiilor etc se menține și calitatea acestora.. Cu ajutorul airdropului aveți posibilitatea de a transmite date între oride dispositive apple, indifferent de id-ul apple introdus pe dispozitiv.

Airplay reprezintă o tehnologie dezvoltată de Apple, care permite streaming-ul de conținut multimedia între dispozitive, utilizând o conexiune Wi-Fi pentru a transmite datele între dispozitive. Pe lângă streaming-ul de conținut de pe dispozitive Apple, AirPlay poate fi utilizat și pentru a afișa conținut de pe site-uri web sau aplicații de divertisment compatibile cu AirPlay pe un dispozitiv mai mare, cum ar fi un televizor.

ApplePay. Este una din cele mai utilizate oportunități oferită de companie. Ea presupune înregistrarea unuia sau a mai multor carduri pe un id Apple și la necesitatea efectuării unei achitări online sau pentru a achita în mod fizic doar că cu lipsa cardului, dar cu apropierea telefonului. Plata se poate efectua de pe orice dispozitiv care este inclus în ecosistem.

Odata cu introducerea carantinei și petrecerea timpului mai mult timp în domeniul online, a aparut și necesitatea de a petrece conferințele în mod online la un nivel înnalt, daca ai un laptop de pe care să organizezi conferința ai posibilitatea de a utiliza iphonul cu camera principală pentru o calitate mai bună a imaginii, viteza mai rapidă a datelor și cu această ocazie e posibil de a petrece conferințele la un alt nivel atît pentru cei de după camere, cît și pentru cei di-n fața camerei.

O altă oportunitate excelentă este inițiată pentru designeri, arhitectori dar si profesori, care ar putea utiliza iphonul sau ipadul ca tabletă grafică. Pentru designeri și arhitectori în general tableta a devenit un lucru de neînlocuit, deoarece sunt foarte multe aplicații care îi ajută în efectuarea sarcinilor mult mai rapide și mai eficiente. Dar și pentru profesori dispozitivul a devenit foarte practic în special, în timpul studiilor online unde era nevoie să arate exemple directe, grafice, tabele și tot o dată să dubleze ecranul, deoarece era inposibil de a reprezenta astea pe o tablă sau pe o foaie apropiată elevilor sau chiar la unii dispozitivul nu corespundea așteptărilor, de asta predarea era doar în mod verbal.

Încă o posibilitate apreciată de companie este selectarea unui text sau a unei imagini de pe un dispozitiv și de a-l pune pe altul fară nici o problemă, ușor utilizat în unele oficii unde mai multe personae lucreaza în grup asupra unui proiect, dar și în viața de zi cu zi când avem de lucru cu niște acte sau prezentări și cautarea informației de pe telefon, dupa care selectarea și amplasarea directă pe calculator în documentul necesar fară a pierde timp. De asemenea odată cu apariția acestei oportunități a mai apărut posibilitatea de a continua crearea documentelor de pe toate dispozitivele, spre exemplu editarea unei prezentări de pe calculator și descărcarea acestuia, dar prezentarea trebuie terminată de urgență, atunci în ajutor vine telefonul din care ușor deschidem prezentarea și continuarea editării a documentului fără nici o pierdere sau risc.

Încă o facilitate care este foarte des folosită și foarte comod în utilizare este hotspot-ul instant, dacă dorești să lucrezi de pe un dispozitiv care nu are conexiune la internet, el automat îți va propune să te folosești de hotspot-ul telefonului la care te poți conecta automat fără a pierde mult timp. Dese ori se întâmplă să lăsăm telefonul la încărcat, și nu avem posibilitatea de a răspunde la un apel sau putem să nu observăm apelul, dar în ecosistema propusă de Apple avem posibilitatea de a răspunde de pe orice dispozitiv conectat la rețea. De asemenea această posibilitate nu se oprește la apeluri, ea dezvoltă toate modalitățile de comunicare.

Compania este cunoscută pentru un număr mic de porturi și necesitatea de a avea mereu un adaptor la îndemână, din acest motiv a fost inventat “sidecar”. Aceasta reprezintă posibilitatea de a utiliza orice sis pozitiv al companiei, cu ecran, sau mai bine zis ca un ecran adăugător pentru lucru, pentru al utiliza trebuie simplu de a da permisiunea utilizării acestei opțiuni și simplu prin apropiere, alăturare s dispozitivelor care automat se sincronizează.

Și nu în ultimul rând, una din cele mai folosite funcții este controlul universal, unde poți lucra separat pe mai multe dispozitive dar și cu aceași tastatură și mouse, simplu trecând de pe un dispozitiv pe altul fără a fi nevoie să le sincronizăm din nou.

Auto-deblocarea dispozitivului dacă prin apropiere este ceasul configurat cu ecosistema personală. În caz că avem un AppleWatch pe mână, care se consideră un dispozitiv ce mereu este purtat pe mână și nu este așa ușor de pierdut, în timpul când deschidem telefonul sau orice alt dispozitiv al companiei vom fi scutiți de necesitatea de a pune o parolă. La prima vedere pare o aberație, dar în practică sa demonstrat a fi foarte utilă posibilitatea [1-6].

Concluzii În urma analizei și expunerea acestui articol am sesizat importanța utilizării unui ecosistem și ușurința cu care se poate munci în cadrul unui ecosistem. Interacțiunea dispozitivelor din ecosistemul companiei. Compania Apple este una care preferă unul din cele mai bine dezvoltate ecosisteme și cel mai ușor de folosit în viața de zi cu zi. Desigur nu este posibil de creat ceva bun fără a avea unele minusuri, însă cel mai mare minos în opinia mea reprezintă prețul ridicat al produselor. Cu toate acestea, compania a reușit să creeze niște produse de neînlocuit și un șir de posibilități de interacțiune între ele. Prin aceste acțiuni au reușit nu doar să atragă clienții care caută niște produse de o calitate ușor de folosit și pus în practică, dar și să mențină clienții fideli, deoarece este foarte greu să revii la dispositive simple după ce simți ușurința utilizării unui dispozitiv din companie, facilitând de toate aspectele propus.

Referințe

1 Apple company, [online],[accesat 28.02.2023] https://www.apple.com
2 Apple company, [online],[accesat 01.03.2023] https://geekflare.com/apple-ecosystem/
3 Tech journeyman, [online], [accesat 01.03.2023] https://techjourneyman.com/blog/appleecosystem-explained/
4 Lucus Newman, [online], [accesat 01.03.2023] https://www.makeuseof.com/features-thatmake-apple-ecosystem-amazing/
5 Apple company, [online], [accesat 01.03.2023] https://support.apple.com/en-us/HT204681
6 Apple company, [online], [accesat 28.02.2023] https://developer.apple.com/nearby-interaction/

CONTROLUL DINAMICII FOTONILOR EMIȘI ÎN MEDIILE BIOLOGICE

Natalia GUBCEAC[1,2]

[1]*Departamentul Fizică, Doctorandă, Facultatea de Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*
[2]*Catedra de Fiziologie a Omului și Biofizica, Facultatea de Medicină, Universitatea de Stat de Medicină și Farmacie "Nicolae Testemițanu", Chișinău, R. Moldova*

*Autorul corespondent: Natalia Gubceac, natalia.gubceac@scee.utm.md

Coordonatori științifici:
Nellu CIOBANU, dr., conf.univ., *Catedra de Fiziologie a Omului și Biofizica, Facultatea de Medicină, Universitatea de Stat de Medicină și Farmacie "Nicolae Testemițanu", Chișinău, R. Moldova.*
Vasile TRONCIU, dr.hab., prof.univ., *Departamentul Fizică, Facultatea de Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

Rezumat. *În această lucrare discutăm o posibilitate de control a dinamicii fotonilor Fröhlich emise de către un mediu biologic care este radiat cu o sursă externă de unde milimetrice. Pentru anumiți parametri ai sistemului considerat, dinamica fotonilor și fononilor emiși pot avea un comportament de unde continue periodice, aperiodice precum și haotice. Un rol aparte îl constituie influența pompajului modulat asupra evoluției sistemului.*

Cuvinte cheie*: unde milimetrice, fotoni Fröhlich, fononi, pompaj modulat, controlul dinamicii.*

Introducere

În ultimele decenii, studierea influenței câmpurilor electromagnetice asupra materiei vii reprezintă un obiectiv destul de atractiv pentru biomedicina modernă [1, 2]. Organismul uman este expus zilnic iradierii diverselor tipuri de unde electromagne prin intermediul tehnologiilor moderne, care sunt extrem de răspândite în viața de zi cu zi. Totalitatea efectelor produse de aceste surse pot distruge sistemele de apărare și funcționare al organismului [3].

Un interes sporit în medicina modernă îl prezintă intervalul undelor milimetrice (MM) și teraherți (THZ) [4-6]. Astfel, o multitudine de companii din întreaga lume care produc dispozitive medicale încearcă să înlocuiască metodele clasice (lezarea tisulară în timpul intervenției chirurgicale) cu cele neinvazive bazate pe terapia undelor MM [7-9]. În timp ce rezultatele practice sunt destul de vizibile și impresionante, mecanismul fizic complet care poate explica procesul de interacțiune a undelor MM sau THZ cu țesutul viu, încă nu există. Experimental a fost demonstrat că la intensități mici frecvențele undelor de 42,7 GHz, 53,5 GHz și 60,7 GHz intră în rezonanță cu undele electromagnetice generate de țesuturile vii. În rezultat, aceste interacțiuni pot juca un rol important în starea funcțională la nivel de celulă, țesut, sau chiar întreg organismul. Din alt punct de vedere, tot experimental este demonstrat că undele MM au efect terapeutic benefic asupra organismelor vii [10,11].

Conform teoriei Fröhlich [12], celulele vii sunt capabile să genereze unde MM. Alte studii au arătat că undele coerente emise de către celulele vii reprezintă un proces sistematic care implică membranele celulare, canalul proteic, precum și pompele de transport celular [13, 14]. O altă teorie explică că în urma acestei interacțiuni are loc un mecanism de disipare neomogenă a energiei undelor MM [15]. Această disipare contribuie la apariția fononilor Bose care este importantă pentru mediile biologice, din punct de vedere al mecanismului de control privind interacțiunea undelor MM cu organismele vii [16].

În această lucrare, este raportată o metodă de control a dinamicii sistemului de fotoni Fröhlich și fononi cu ajutorul pompajului modulat al sursei de unde milimetrice. Pentru anumite valori ai parametrilor sistemului precum și a indicelui de modulare, dinamica fononilor dipol-activi Bose condensați și a fotonilor interni Fröhlich pot prezenta regimuri de oscilații continuee, dublări de perioadă precum și regimuri continue.

Modelul teoretic

Modelul teoretic ce descrie evoluția fotonilor și fononilor generați de către mediul biologic a fost discutat în lucrările anterioare [15,16]. În Figura 1 este reprezentată schema de iradiere a mediului biologic cu ajutorul unei surse de unde MM cu pompaj modulat de energie: $P = P_0(1 + m(cos2\pi\omega t))$, unde $m -$ reprezintă parametrul de modulare. În urma acestei iradieri mediul biologic generează fotoni Frohlich și fononi, care ulterior sunt folosiți pentru iradierea aceluiași mediu biologic. Procesul de iradiere are loc în baza unui feedback optic exterior cu faza φ și amplitudinea Γ.

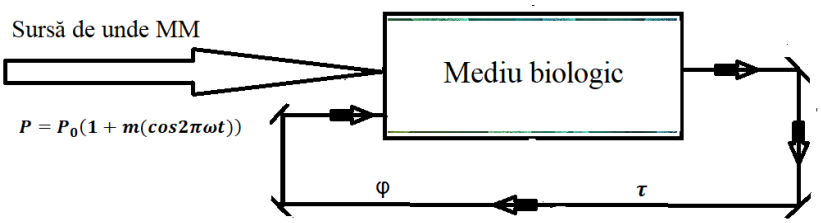


Figura 1. Schema de generare a fotonilor și fononilor mediului biologic

Pentru descrierea modelului dat vom considera același sistem de ecuații diferențiale [16]:

$$\frac{dE_1}{dT} = -\sigma E_1 - (\tilde{\delta} - \tilde{\omega}_0)E_2 + 2\alpha\tilde{\gamma}A_1 + \alpha\{2\tilde{\gamma}[\delta - (A_1^2 + A_2^2)] - 1\}A_2 + P + \\ +\Gamma[cos(\varphi)\, E_1(t-\tau) + sin(\varphi)\, E_2(t-\tau)], \quad (1)$$

$$\frac{dE_2}{dT} = (\tilde{\delta} - \tilde{\omega}_0)E_1 - \sigma E_2 + 2\alpha\tilde{\gamma}A_2 + -\alpha\{2\tilde{\gamma}[\delta - (A_1^2 + A_2^2)] - 1\}A_1 + \\ +\Gamma[cos(\varphi)\, E_2(t-\tau) - sin(\varphi)\, E_1(t-\tau)], \quad (2)$$

$$\frac{dA_1}{dT} = -\alpha E_2 - A_1 - [\delta - \nu(A_1^2 + A_2^2)]A_2, \quad (3)$$

$$\frac{dA_2}{dT} = -\alpha E_1 + [\delta - \nu(A_1^2 + A_2^2)]A_1 - A_2. \quad (4)$$

În acest sistem δ, $\tilde{\delta}\tilde{\delta} = \frac{\omega^2\text{-}c^2k^2}{2\omega\gamma}$, α, Ω_0, $\tilde{\omega}_0$, $\tilde{\gamma}$, σ, și T sunt parametrii adimensionali$\tilde{\delta} = \frac{\omega^2\text{-}c^2k^2}{2\omega\gamma}$.

Rezultate

În continuare vom discuta evoluția dinamicii sistemului descris de ecuațiile (1)-(4) pentru diferite valori ale parametrilor sistemului. Puterea fotonilor emiși precum și concentrația fononilor generați de mediul biologic pot fi definiți astfel: $P = E_1^2 + E_2^2$, $n = A_1^2 + A_2^2$. În calculele numerice au fost fixaţi următorii parametri adimensionali: $\tilde{\omega}_0 = 0.1$, $\nu = 1$, $\tilde{\gamma} = 1.1$, $\tilde{\delta} = 0.1$, $\tau = 0.15$, $\delta = 3$, $\sigma = 2$, $P_0 = 120$, $f = 2$.

Figura 2 descrie evoluţia în timp a puterii fotonilor emiși și portretele de fază pentru anumite valori ale parametrilor sistemului. Influența pompajului modulat asupra dinamicii este studiată cu ajutorul parametrului de modulare m. Pentru valori nule ale indicelui de modulare ($m = 0$), evoluția sistemului tinde către o stare staționară, iar traiectoriile de fază descriu un ciclu limită. Pentru $m = 0.1$, dependența în timp a puterii fotonilor emiși descrie un comportament de oscilații continue, iar traiectoriile de fază bifurcații de tip Hopes. Cu creșterea parametrului de modulare, pentru *m=0.4* în sistem pot apărea pulsații periodice, iar pentru *m=0.9* se poate observa și apariția regimului haotic. Așadar pentru anumite valori fixe ale sistemului considerat, putem controla cu dinamica sistemului prin intermediul parametrului de modulare al pompajului.

În Fig. 3 este studiată dinamica sistemului pentru $\alpha = 8.5$. După cum se observă, pentru *m=0,* evoluția în timp a fotonilor emiși descrie un regim de oscilații continue, iar traiectoriile de fază

bifurcații de tip Hopes. Cu creșterea indicelui de modulare în sistem pot apărea bifurcații cu dublare de perioadă ($m = 0.4$) precum și multiplicare de perioadă ($m = 0.9$). Astfel, cu ajutorul parametru α este posibil de trecut doar de la regim continuu la regim cu dublare de perioadă.

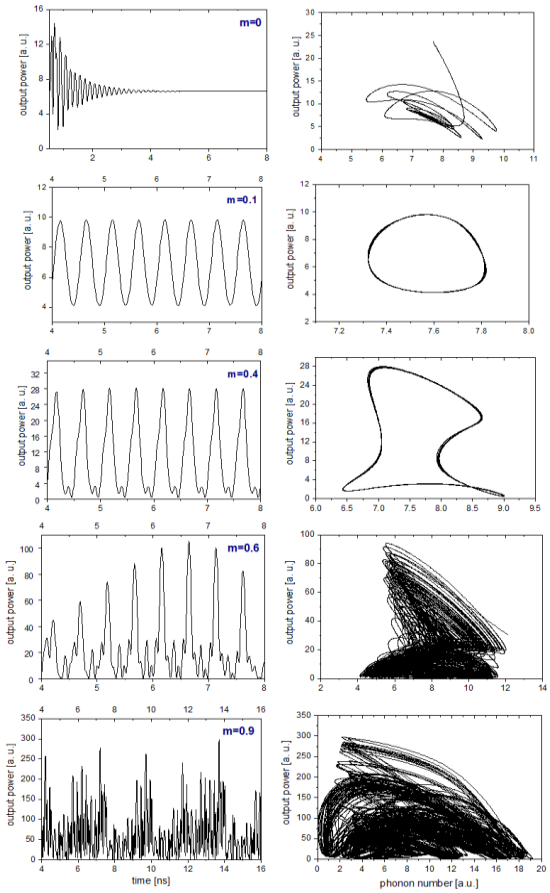


Figura 2. Evoluția în timp a intensității fotonilor emiși (stânga) și portretele de fază (dreapta) pentru $\alpha = 5, \Gamma = 10, \varphi = \pi/3$.

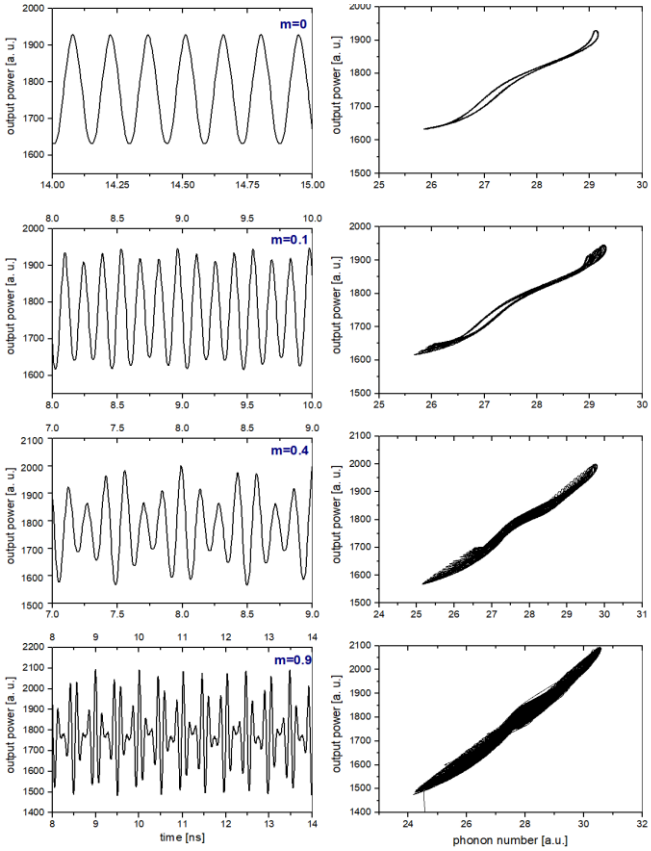


Figura 3. Evoluția în timp a intensității fotonilor emiși (stânga) și portretele de fază (dreapta) pentru pentru $\alpha = 8.5, \Gamma = 10, \varphi = \pi/3$.

În Fig. 4 este studiată dinamica sistemului la schimbarea fazei fedback-lui optic ($\varphi = 3\pi/2$). Pentru m =0, pot fi observate oscilații periodice în dinamica sistemului, iar cu creșterea acustuia pot apărea și oscilații haotice. Pentru $m = 0.1$, traiectoriile de fază descriu un atractor straniu.

Influența amplitudinii fedback-lui optic este reprezentată în Figura 5. După cum se observă, indicele de modulare nu influențează drastic dinamica sistemului.

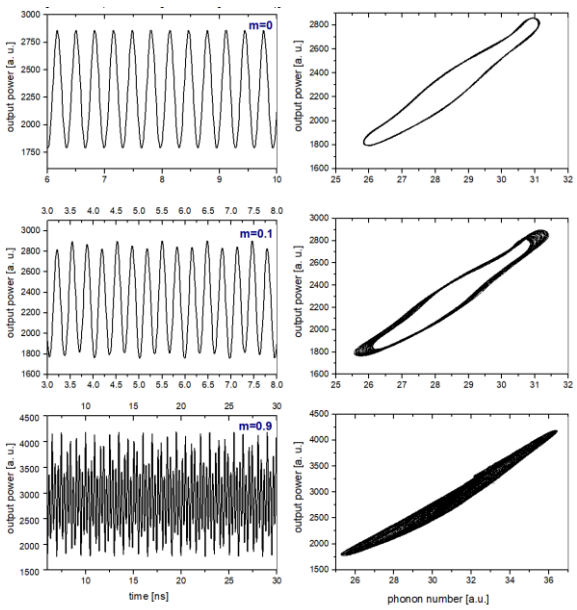


Figura 4. Evoluția în timp a intensității fotonilor emiși (stânga) și portretele de fază (dreapta) pentru pentru $\alpha = 5$, $\Gamma = 10$, $\varphi = 3\pi/2$.

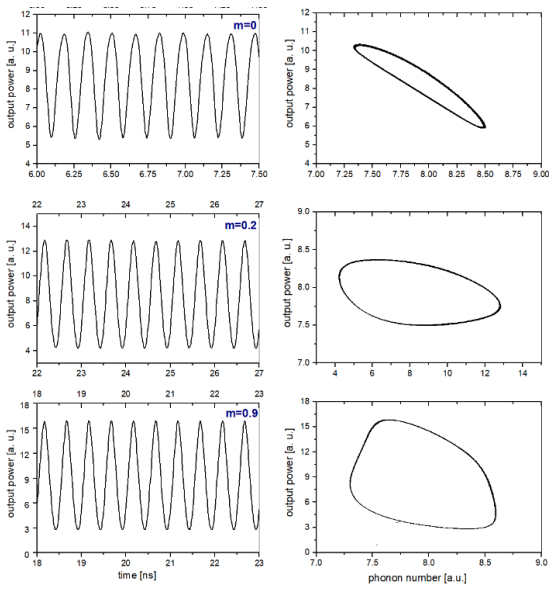


Figura 5. Evoluția în timp a intensității fotonilor (stânga) și portretele de fază (dreapta) pentru pentru $\alpha = 5$, $\Gamma = 14$, $\varphi = \pi/3$.

Concluzii

În această lucre este studiată o posibilitate de control a dinamicii sistemului de fotoni și fononi cu ajutorului pompajului modulat. După cum a fost demonstrat pentru anumite valori fixe ale sistemului, cu ajutorul parametrului de modulare se poate de trecut de la un regim de unde continue la regimuri cu dublare de perioadă, multiplicare precum și la regim haotic. Parametrii fedback-lui optic influențează drastic dinamica sistemului, la care sa observat și apariția atractorilor stranii. Considerăm ca cu ajutorul acestor structuri putem prezice evoluția în timp a proceselor fiziologice la nivel de celule care decurg în sistemele biologice.

Mulţumiri. Această lucrare a fost realizată cu suportul ANCD în cadrul proiectului 20.80009.5007.08 „Studiul structurilor optoelectronice și a dispozitivelor termoelectrice cu eficiență înaltă”.

Referinţe

1. A. D’Arco, M. Di Fabrizio, V. Dolci, M. Petrarca, S. Lupi, „THz pulsed imaging in biomedical applications”, Condensed Matter 5(2):25, 2020. https://doi.org/10.3390/condmat5020025
2. I. Akinori, L. Stefano, M. Augusto, „Terahertz as a frontier area for science and technology”, Condensed Matter, **6**, 23, 2021. https://doi.org/10.3390/condmat6030023
3. A. Ramundo-Orlando, “Effects of Millimeter Waves Radiation on Cell Membrane,” J Infrared Milli Terahz Waves, **31**, pp. 1400–1411, 2010. https://doi.org/10.1007/s10517-018-4243-2
4. Sarika S., Neeru K..Health Implications of Electromagnetic Fields, Mechanisms of Action, and Research Needs, In: Advances in Biology, 2014, 24 pages.
5. M. Kojima, C.-Y. Tsai, Y. Suzuki, K. Sasaki, T. Tasaki, M. Taki, S. Watanabe, H. Sasaki „Ocular Response to Millimeter Wave Exposure Under Different Levels of Humidity”, Journal of Infrared, Millimeter, and Terahertz Waves, Vol. **40**, 2019, pp. 574–584. https://doi.org/10.1007/s10762-019-00586-0
6. P. Le Pogam, Y. Le Page, D. Habauzit, M. Doué, M. Zhadobov, R. Sauleau, Y. Le Dréan, D. Rondeau, “Untargeted metabolomics unveil alterations of biomembranes permeability in human HaCaT keratinocytes upon 60 GHz millimeter-wave exposure”, 2019. https://doi.org/10.1038/s41598-019-45662-6

7. H. Torgomyan, A. Trchounian, “Bactericidal effects of low-intensity extremely high frequency electromagnetic field: An overview with phenomenon, mechanisms, targets and consequences”, Critical Reviews in Microbiology **39**(1), 2013, pp.102-111. http://dx.doi.org/10.3109/1040841X.2012.691461
8. A.G. Davies, E.H. Linfield, M.B. Johnston, “The development of terahertz sources and their applications”, Phys. Med. Biol., **47**, 2002, pp. 3679–3689.
9. T. Globus, D.L. Woolard, T. Khromova, T.W. Crowe, M. Bykhovskaya, B.L. Gelmont, et al., „THz spectroscopy of biological molecules”, *J.* Biol. Phys. **29**, 2003**,** pp. 89–100.
10. X.-H. Li, J.-T. Tang, Y.-P. Liao, H.-K. Jin, J.-M. Zhou, G.-H. Wang, H. Wang, “Millimeter wave in the treatment of acute radiation-induced cervical skin ulcers,” J. Clin. Rehab. Tissue Eng. Res. **12**, 2008, pp. 663-666.
11. H. Frohlich, „Bose condensation of strongly excited longitudinal electric modes”, Phys. Lett, A **26**, 1968, pp. 402-403. https://doi.org/10.1016/0375-9601(68)90242-9
12. V.I. Geletyuk, V.N. Kazachenko et al., „Dual effects of microwaves on single Ca2+-activated K+ channels in cultured kidney cells”, FEBS Letters, **359**, 1995, pp. 85-88.
13. M. Zhadobov, R. Saileau, V. Viè, „Interactions between 60-GHz millimeter waves and artificial biological membranes: dependence on radiation parameters”, IEEE Tras. MW Theory and Tec., **54**, 2006, pp. 2534-2542.
14. H. Bolterauer, J.A. Tuszyński, „Fröhlich and Davydov regimes in the dynamics of dipolar oscillations of biological membranes”, Phys. Rev. A, **44**, 1991, p. 1366.
15. N. Ciobanu, S. Rusu, V. Tronciu, „Dynamical behavior of Bose-condensed dipole-active phonons and internal Fröhlich photons in biological media”, Proceedings of the 5th IEEE International Conference on E-Health and Bioengineering-EHB 2015, Iasi, 19-21 November, 2015, pp 978. http://dx.doi.org/10.1109/EHB.2015.7391369
16. N. Gubccac, N. Ciobanu and V. Tronciu, "Control Dynamics of the Generated Frohlich Photons by Biological System," 2022 E-Health and Bioengineering Conference (EHB), Iasi, Romania, 2022, pp. 01-04, doi: 10.1109/EHB55594.2022.9991356.

ANALIZA CAPITALULUI FIX ȘI PROBLEMELE GENERALE DE UTILIZARE LA ÎNTREPRINDEREA USSS

Mihaela MĂMĂLIGĂ

Departamentul Telecomunicații și Sisteme Electronice, grupa MMRT-221M , Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova.

*Autorul corespondent: Mihaela Mămăligă, mihaela.mamaliga@tlc.utm.md

Coordonator științific: Lilia SAVA, dr., conf.univ., *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

Rezumat. *Capitalul fix poate fi definit ca unul dintre factorii de producţie, situaţie în care el este acceptat ca o categorie de bunuri produse şi utilizate în scopul producerii altor bunuri economice şi se constituie ca obiect de analiză a rezultatelor combinării cu alţi factori de producţie, rezultate ce depind de abilitatea întreprinzătorului. Noţiunea de „eficienţă" este o concepţie modernă de evaluare a activităţii şi serveşte la fundamentarea deciziilor, astfel încât resursele disponibile să fie consumate cât mai eficient şi inofensiv pentru societate. Principiul eficienţei economice, ca şi alt principiu are două părţi componente: cantitativă şi calitativă .*

Cuvinte cheie*: Mijloace fixe, randament, capacitate, eficiență.*

Introducere

Indiferent de dimensiune şi spaţiul socio-economic în care activează, orice întreprindere trebuie să ţină cont de componenţa, structura, eficienţa şi reînnoirea mijloacelor fixe.

Astfel, utilizarea mai eficientă a mijloacelor fixe dă posibilitatea companiei de a efectua un volum mai mare de prestări servicii, aceasta reflectîndu-se pozitiv asupra rezultatelor finale ale activităţii economico–financiare şi anume, asupra profitului.

Din acest punct de vedere, analiza economică serveşte drept instrument efectiv de evidenţiere a căilor de îmbunătăţire a calităţii mijloacelor fixe din cadrul companiei.

1. Definirea și tipologia capitalului fix

Prin noțiunea de **capital fix** se definește acea parte a capitalului productiv (real, tehnic) format din bunuri de lungă durată, acestea servind ca instrument al muncii oamenilor în mai multe cicluri de producţie, care apoi se consumă treptat şi se înlocuiesc după mai mulţi ani de folosire [1].

Caracteristicile ce definesc capitalul fix sunt:[5]

- participarea la mai multe cicluri de producţie;
- nu intră în componenţa produsului creat;
- nu îşi transmit integral valoarea, ci parţial prin amortizare.

Cele mai importante caracteristici de grupare utilizate în analiza statistică sunt definite în funcţie de:

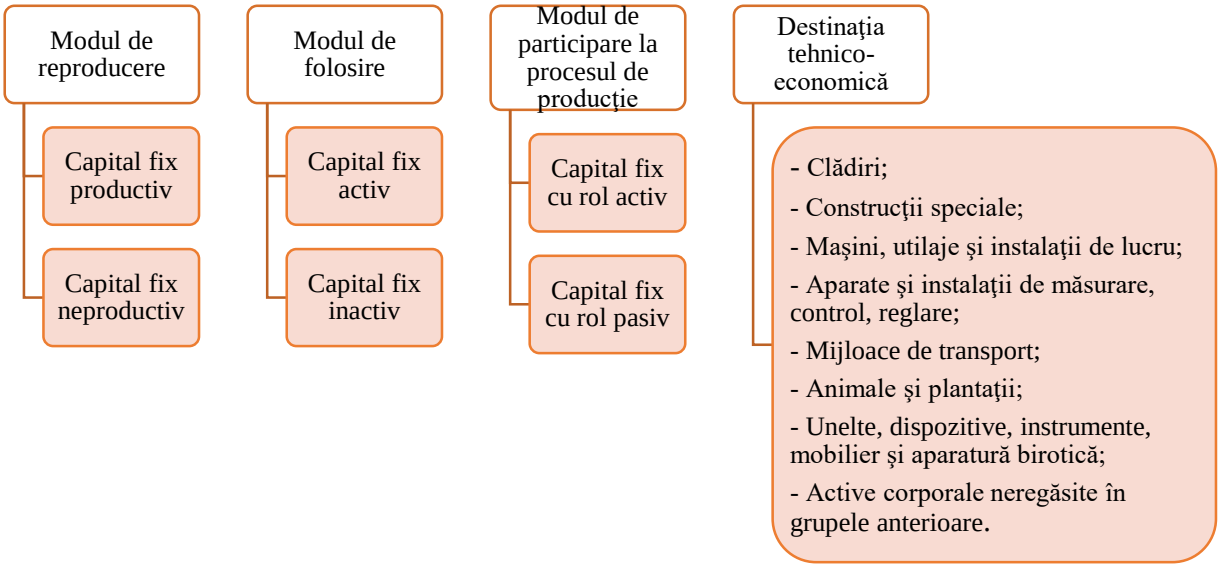


Figura 1. Tipologia capitalului fix [2]

2. Analiza eficienţei utilizării mijloacelor fixe

Analiza eficienţii utilizării mijloacelor fixe se va efectua în baza determinării randamentului mijloacelor fixe. Pentru a scoate în evidenţă cauzele abaterii randamentului mijloacelor fixe productive se efectuează analiza factorială a acestui indicator. În acest scop datele iniţiale se întroduc în tabel (Tab.1, sursa: Bilanţul economic al întreprinderii US Software Solutions) cu ajutorul cărora se vor efectua calculele indicatorilor respectivi pentru analiza factorială.

Tabelul 1

Date iniţiale şi de calcul privind analiza factorială a randamentului MF de producţie

Indicatorii	**Semn convţ**	**Anul precedent**	**Anul de gestiune**	**Abaterea (+/-)**
A	***B***	***1***	***2***	***3***
1. Volumul producţiei fabricate în preţuri comparabile, mii lei.	***VPF***	15580,7	27707,3	+12126,6
2. Valoarea medie a MF , mii lei.	***MF***	1341,0	2181,9	+840,9
3. Valoarea medie a mijloacelor fixe active, mii lei.	***MFa***	1017,5	1157,8	+140,3
A	***B***	1	2	3
4. Numărul mediu de utilaj instalat, unităţi.	***U***	448	324	-124
5. Timpul efectiv lucrat în total, utilaj-ore.	T_{ut}	2568	2104	-464
6. Valoarea medie a unităţii de utilaj, mii lei (3/4)	V_U	2,271	3,573	+1,302
7.Ponderea mijloacelor fixe active în suma totală a MF, % (3/2*100%)	***P%***	75,876	53,064	-22,812
8. Durata medie de lucru a unui utilaj, utilaj-ore. (5/4)	***D***	5,732	6,494	+0,762
9. Productivitatea medie a unui utilaj pe oră, lei (1/5)	***Wh***	6,067	13,169	+7,102
10. Randamentul MF, lei (1/2)	R_{MF}	11,619	12,699	+1,08
11. Randamentul MF active, lei (1/3)	R_{MFa}	15,313	23,931	+8,618

În prima etapă de analiză, prin metoda substituţiei în lanţ, se calculează influenţa factorilor generali asupra modificării randamentului mijloacelor fixe, în tabelul 2.

Tabelul 2

Calculul influenţei factorilor generali asupra modificării randamentului MF

Nr de cal	Nr de subs	Indictorii factorilor corelaţi		Rand MF, lei	Calculul influienţei	Rezultatul influenţei (+/-), lei	Denumirea factorilor
		P%	R_{MFa}				
1	*2*	*3*	*4*	*5*	*6*	*7*	*8*
1	0	75,876	15,313	11,619	x	X	x
2	1	53,064	15,313	8,126	8,126-11,619	-3,493	ΔP%
3	2	53,064	23,931	12,699	12,699-8,126	+4,573	ΔR_{MFa}
Total		***x***	***x***	***x***	***x***	***1,08***	***x***

BIF: 1,08=(-3,493)+4,573

Observaţie. În baza rezultatelor obţinute din tabelul 2.2 se poate constata că randamentul mijloacelor fixe la întreprinderea analizată s-a majorat faţă de anul precedent cu 1,08 lei.

Această majorare a fost influenţată în mare parte de creşterea randamentului mijloacelor fixe active în anul de gestiune cu 8,618 lei ce a dus la majorarea randamentului mijloacelor fixe totale cu 4,573.

Concomitent, o influenţă negativă în valoare de 3,493 lei a fost marcată din cauza scăderii ponderii mijloacelor active în volumul total a mijloacelor fixe cu 22,812 lei

Calculul şi aprecierea factorilor ce ţin de asigurarea şi utilizarea mijloacelor fixe la devierea venitului din vânzări

În prima etapă de analiză, prin metoda diferenţelor absolute cu doi factori, calculăm influenţa factorilor generali conform tabelului 4 [3]

Tabelul 4

Calculul influenţei factorilor generali ai MF asupra modificării venitului din vânzări

Indicatorii	Semn con venţin	Anul prece dent	Anul de gestiune	Abaterea absolută (+/-)	Inclusiv sub influienţa	
					MF	R_{MF}
A	***B***	***1***	***2***	***3***	***4***	***5***
1. Volumul producţiei fabricate în preţuri comparabile, mii lei.	VPF	15580,7	27707,3	+12126,6	9770,417	2356,452
2. Valoarea medie a MF, mii lei.	MF	1341,0	2181,9	+840,9	x	X
3. Randamentul MF, lei. (1/2)	R_{MF}	11,619	12,699	+1,08	x	X

Observaţie: În baza rezultatelor obţinute s-a constatat că volumul producţiei fabricate a crescut cu 12126,6 mii lei ca rezultat al influenţei pozitive atât a factorului cantitativ cât şi a factorului calitativ şi anume: creşterea valorii medii a mijloacelor fixe cu 840,9 mii lei a dus la majorarea volumului producţiei fabricate cu 9770,417 mii lei, iar sporirea cu 1,08 lei a randamentului a dus la creşterea indicatorului rezultativ cu 2356,452 mii lei.

La **etapa următoare de analiză**, prin metoda substituţiei în lanţ, calculăm influienţa factorilor detaliaţi la modificarea randamentului mijloacelor fixe.

Observaţie. În baza rezultatelor obţinute, se poate constata că volumul producţiei fabricate s-a majorat faţă de anul precedent cu 12130,904 mii lei. Această abatere a fost asigurată de creşterea valorii medii a mijloacelor fixe cu 9770,388 mii lei, creşterii duratei medie de lucru a unui utilaj cu 1498,066 mii lei şi a sporirii productivităţii unui utilaj pe oră cu 14944,96 mii lei. Concomitent, scăderea ponderii mijloacelor fixe active și creşterea preţului mediu al unui utilaj a dus la micşorarea fabricarii producţiei cu 7621,865 mii lei și respectiv 6460,645 mii lei.

Tabelul 5

Calculul influienţei factorilor detaliaţi asupra modificării venitului din vânzări

Nr de calcul	Nr de substituţ	Indicatorii (factorii) corelaţi					VV, mii lei	Calculul influienţei factorilor	Rezultatul Influienţei	Denumirea factorilor
		MF	P%	$\frac{1}{Vu}$	D	W_h				
1	0	1341,0	75,876	1/2,271	5,732	6,067	15581,033	x	x	x
2	1	2181,9	75,876	1/2,271	5,732	6,067	25351,421	25351,421-15581,033	+9770,388	*ΔMF*
3	2	2181,9	53,064	1/2,271	5,732	6,067	17729,556	17729,556-25351,421	-7621,865	*ΔP%*
4	3	2181,9	53,064	1/3,573	5,732	6,067	11268,911	11268,911-17729,556	-6460,645	*Δ1/Vu*
5	4	2181,9	53,064	1/3,573	6,494	6,067	12766,977	12766,977-11268,911	+1498,066	*ΔD*
6	5	2181,9	53,064	1/3,573	6,494	13,16	27711,937	27711,937-12766,977	+14944,96	*ΔWh*
Total		***x***	***x***	***x***	***x***	***x***	***x***	***x***	***12130,904***	***x***

Concluzii:

În baza analizei efectuate putem menționa că orice întreprindere, de orice dimensiune şi spaţiul socio-economic în care aceasta activează, trebuie să ţină cont mereu de structura, componenţa, eficienţa şi reînnoirea mijloacelor fixe. În așa mod, utilizarea mai eficientă a mijloacelor fixe dă posibilitatea întreprinderii de a realiza un volum mai mare de prestări servicii, aceasta reflectîndu-se pozitiv asupra rezultatelor finale ale activităţii economico- financiare şi anume, asupra profitului. Rezultatele analizei factoriale fac posibilă să concluzionăm că întreprinderea USSS dispune de rezerve interne de majorare a volumului producţiei care va fi fabricată în viitor, şi anume:

- trebuie luate măsuri privind majorarea duratei de funcţionare a unui utilaj;
- creșterea volumului producției poate fi efectuată prin majorarea ponderii mijloacelor fixe active în suma totală a mijloacelor fixe.

Astfel, eficienţa economică reprezintă cea mai importantă cerinţă care se impune în toate activităţile economice şi în acţiunea oricărei întreprinderi. În acest context, scopul companiei este nu numai de a obţine rezultate maximale, dar şi de a determina cu ce preţ vor fi obţinute acestea şi care este eficienţa economică de producere a lor.

Referinţe

1. Lilia SAVA, Maria GRIŢCO. Organizarea și analiza activității economice în domeniul telecomunicațiilor. Ghid metodic. Partea II. Chișinău, Editura ”Tehnica –UTM”, 2018.
2. Lilia SAVA, Daniela Vortolomei. Organizarea și analiza activității economice în domeniul telecomunicațiilor. Note de curs. Chișinău, Editura ”Tehnica –UTM”, 2022. ISBN 979-9975-45-805-4., pag. 25-32.
3. Lilia SAVA, Blanaru Vasile. Statistica ramurii. Note de curs. Chișinău, Editura ”Tehnica –UTM”, 2022. ISBN 978-9975-45-808-5., pag. 4-15.
4. Analiza mijloacelor fixe. https://www.scrigroup.com/afaceri/economie/Analiza-mijloacelor-fixe62389.php
5. Analiza Statistica a Capitalului Firmei. https://ro.scribd.com/doc/57197937/04-Analiza-Statistic-A-a-Capitalului-Firmei?language_settings_changed=Rom%C3%A2n%C4%83#

CERCETAREA PARTICULARITĂȚILOR DE ELABORARE ȘI IMPLEMENTARE A SISTEMELOR SATELITAR DE TELEVIZIUNE DIGITALĂ

Mihail MEREUȚĂ

Departamentul Telecomunicații și Sisteme Electronice, grupa SCE-211M, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova.

*Autorul corespondent: Mereuță Mihail, mihail.mereuta@tse.utm.md

Coordonator științific: Tatiana ȘESTACOVA, dr., conf.univ., *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

Rezumat. *În lucrarea dată a fost efectuat cercetarea particularităților de elaborare și implementare a sistemelor satelitar de televiziune digital. Au fost luate în considerare principiile de bază ale construirii sistemelor de comunicații prin satelit, au fost studiate caracteristicile și tehnologiile de construire și implementare a comunicațiilor prin satelit VSAT. De asemenea, a fost efectuată o analiză a topologiei construirii unei rețele VSAT, a echipamentelor și a fost efectuat un calcul energetic al comunicațiilor prin satelit.*

Cuvinte cheie*: televiziunea digitală, sisteme satelitar, rețele VSAT*

Introducere

Comunicațiile prin satelit sunt unul dintre tipurile de comunicații radio bazate pe utilizarea sateliților artificiali ca repetitoare. Comunicarea prin satelit se realizează între stațiile terestre, care pot fi atât staționare, cât și mobile. Industria de comunicații prin satelit și de radiodifuziune este un element esențial al pieței globale de telecomunicații Noile aplicații prin satelit permit crearea rapidă de noi servicii de difuzare și rețele private, ceea ce oferă un impul spre dezvoltare cît în comunicații , atît și sectorului IT.

În cadrul acestei lucrări se realizează atât crearea și lansarea navelor spațiale, cât și introducerea noilor tehnologii în domeniul dispozitivelor cu microunde și microprocesor. În acest sens, orice problemă de pe piața telecomunicațiilor afectează într-un fel sau altul dezvoltarea comunicațiilor prin satelit. În această lucrare va fi luat în considerare un sistem de transmisie a semnalului de televiziune. Televiziunea prin satelit este o metodă de transmitere a datelor video prin transmiterea unui semnal printr-un satelit artificial de pe Pământ (AES).

1. Principii de construire a sistemelor de comunicare prin satelit

Principiul comunicației prin satelit este de a transmite semnalul de la stațiile terestre de transmisie către receptori prin echipamentul satelitului. Satelitul este un dispozitiv de comunicație care primește semnale de la o stație terestră (ES), care ulterior le amplifică și le transmite într-un mod de difuzare simultan către toate ES care se află în raza de vizibilitate a satelitului. Principiul comunicării prin sateliți constă în transmiterea semnalelor de la una sau mai multe stații terestre către sateliți, cu retransmiterea lor ulterioară către toate ES ale sistemului. În prezent, se utilizează mai multe tipuri de orbite pentru a găzdui sateliții. Atragem atenția la sistemele de orbită joasă, geostaționare și eliptice. Principiul de funcționare a sistemelor de comunicații prin satelit (SCS) se bazează pe utilizarea unui releu intermediar prin satelit (SR), prin care se asigură comunicația între ES (Fig. 1).

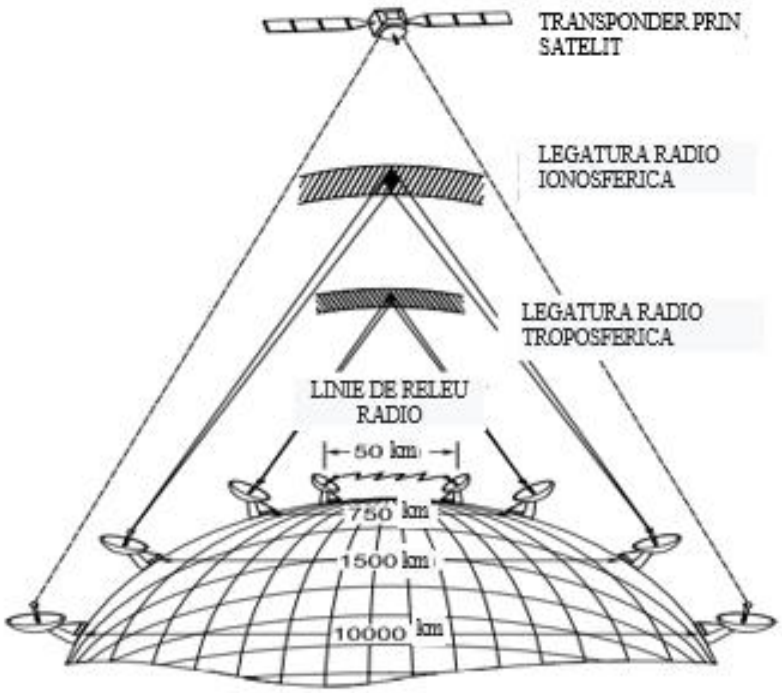


Figura 1. Principiul de funcționare a sistemelor de comunicații prin satelit

În funcție de scopul SSN, punctele care trebuie conectate pot fi localizate pe suprafața Pământului, în atmosferă sau în spațiu. În fiecare dintre aceste puncte, este instalată de obicei o stație radio de comunicații de transmisie și recepție (un singur canal sau multicanal), iar pe sateliți - SR-uri care primesc semnale radio de la unii abonați și transmit aceste semnale către alți abonați. În cel mai simplu caz, retransmiterea se reduce la amplificarea puterii semnalelor de intrare și transferul spectrelor acestora către alte frecvențe purtătoare. Cu toate acestea, într-un număr de SCS -uri din SR, se realizează o procesare mai complexă a semnalului pentru a reduce diafonia între semnalele de la diferite SCS -uri și pentru a crește imunitatea la zgomot a sistemului. În cazul general, pentru a asigura o comunicare de înaltă calitate între toate punctele (abonați), SR-ul trebuie plasat pe mai mulți sateliți care se rotesc pe orbite diferite.

Prin afiliere, SCS -urile sunt împărțite în internaționale, naționale, corporative. În funcție de zona de serviciu, SCS -urile sunt împărțite în globale, regionale, zonale .

Alocarea benzilor de frecvență între diferitele servicii de radiocomunicații se ocupă de una dintre organizațiile specializate ale ONU - Uniunea Internațională de Telecomunicații (ITU), care, pe baza studiilor efectuate în țările membre ale UIT și înaintate Comitetului Consultativ Internațional Radio (CCIR).), la conferințele sale administrative elaborează reguli și proceduri adecvate de guvernare. Principalul document internațional care reglementează utilizarea frecvențelor este Regulamentul Radio.

Acesta conține Tabelul de alocări ale benzilor de frecvență între servicii, anumite restricții tehnice impuse cu privire la partajarea între diferite servicii, proceduri pentru sistemele de coordonare și reguli de înregistrare a asignărilor de frecvență în Consiliul Internațional de Înregistrare a Frecvenței . Au fost alocate benzi de frecvență pentru sistemele de difuzare prin satelit (Tabelul 1).

Benzile Ka și K nu sunt aproape niciodată folosite și în prezent sunt considerate experimentale. Cu toate acestea, difuzarea programelor TV prin satelit în aceste benzi va reduce semnificativ diametrul antenelor de recepție.

Tabelul 1

Benzile de frecvență ale sistemelor de difuzare prin satelit

Numele intervalului	Banda de frecventa, GHz
L - diapazon	1,452 – 1,550 și 1,61 – 1,71
S - diapazon	1,93 – 2,70
C - diapazon	3,40 – 5,25 și 5,725 – 7,075
X - diapazon	7,25 – 8,40
KU - diapazon	10,70 – 12,75 și 12,75 – 14,80
Ka - diapazon	15,4 – 26,5 și 27,0 – 50,2
K - diapazon	84 - 86

2. Cercetarea caracteristicilor de construcție și aplicare a sistemului de televiziune digital prin satelit

O caracteristică distinctivă a transmisiunii de televiziune prin satelit (STV) este capacitatea abonatului de a primi programul necesar de la orice satelit.

Avantajele televiziunii prin satelit sunt:

- în primul rând, este o imagine de înaltă calitate, fără interferențe, datorită tehnologiei utilizate pentru comprimarea semnalului transmis;
- posibilitatea de a viziona TV prin satelit oriunde
- în oraș sau la țară;
- calitate ridicată a sunetului;
- sute de canale interesante, atât cu plată, cât și gratuite;

Acest lucru se datorează faptului că atât de multe canale sunt în difuzare deschisă; STV se efectuează în două moduri. Prima metodă este următoarea: semnalul primit de la satelit este trimis către centrul local de televiziune, care asigură retransmisia lui ulterioară

A doua modalitate este următoarea: recepția semnalului de televiziune prin satelit se realizează pe o unitate de recepție separată cu o antenă mică. Pentru a asigura o astfel de recepție, este necesar să se folosească un satelit cu o putere de emițător relativ mare și o bună precizie de păstrare a orbitei pentru a exclude utilizarea receptorilor de urmărire.

3. Calcul energetic al liniei de comunicație

Pentru a efectua un calcul energetic al unei linii de comunicație prin satelit, este necesar să se cunoască coordonatele satelitului selectat și locația stației de recepție. Considerăm că modulul de abonat receptor se află în Chișinău. Vom determina locația pe hartă .

Indicăm coordonatele geografice:

- latitudine - 47°0′18,0″ latitudine nordică, 47,0056 în grade zecimale;
- longitudine - 28°51′27,7″ longitudine estică, 28,8575 în grade zecimale.

Fasciculele puternice și concentrate în bandă Ku oferă servicii de bandă largă pentru Europa, Rusia, Africa, Orientul Mijlociu și Asia .

Pentru a rezolva o serie de probleme care apar la proiectarea unei linii de comunicație prin satelit, este necesar să se cunoască relațiile geometrice care determină poziția relativă a stației terestre și a satelitului. Deci, pentru orientarea corectă a antenei ES, este necesar să se cunoască unghiul de elevație β și azimutul α al acesteia. Azimutul este unghiul măsurat în plan orizontal în sensul acelor de ceasornic de la direcția către Polul Nord până la direcția de la stația terestra la satelit. Unghiul de elevație este unghiul măsurat în plan vertical de la tangenta la orizont la direcția de la stația terestra la navă spațială. Intervalul de înclinare este distanța de la stația terestra la navă spațială. Intervalul de înclinare este determinat de formula:

$$d = R_p \cdot \frac{\sqrt{1+\gamma_o^2 - 2\gamma_o \cdot cos\psi}}{\gamma_o}, C = W \log_2(1 + \frac{P_c}{P_ш}), \quad (1)$$

γ_0 și cosψ prezenți în formulă sunt coeficienți intermediari.

Găsim coeficientul intermediar folosind formula:

$$\gamma_o = \frac{R_p}{R_p + H}, \quad (2)$$

unde H este înălțimea orbitei geostaționare.
Al doilea coeficient intermediar:

$$\cos\psi = \cos\varphi ct \cdot \cos\Delta\lambda \quad (3)$$

a) Pierderi în spațiu.

Un factor important care afectează nivelul semnalului la intrarea receptorului este pierderea de energie în timpul propagării semnalului de-a lungul legăturii radio prin satelit. Atenuarea energiei semnalului în spațiul liber, cauzată de o scădere a densității fluxului de putere cu distanța de la emițător, este determinată de următoarea formulă:

$$L_{sl} = 20lg\left(\frac{4\pi d}{\lambda}\right) = 20lg\left(\frac{4\cdot 3{,}14\cdot 38267520}{0{,}025641}\right) = 205{,}4\ dB, \quad (4)$$

unde d - este intervalul de înclinare, în metri; λ- este lungimea de undă.
Lungimea de undă este determinată de formula:

$$\lambda = \frac{c}{f} = \frac{3\cdot 10^8}{11{,}7\cdot 10^9} = 0{,}02564\ \text{м}, \quad (5)$$

unde c - este viteza luminii (m/s); f - este frecvența de funcționare (Hz).

b) Pierderea energiei semnalului în atmosferă.

Absorbția undelor radio în atmosferă este cuantificată prin coeficientul La. S-a stabilit că în intervalele de frecvență de peste 500 MHz, absorbția principală este determinată de troposferă și anume gazele troposferei: oxigen și vapori de apă, precum și ploaia și alți hidrometeori. În acest caz, ionosfera și alte gaze ale troposferei, cum ar fi dioxidul de carbon sau azotul, joacă un rol nesemnificativ.
Absorbția undelor radio în atmosferă este determinată de formula:

$$\boldsymbol{L_{atm} = L_o \cdot l_1 + L_{H2O} \cdot l_2}, \quad (6)$$

unde - $\boldsymbol{L_O}$ este coeficientul de absorbție pe unitate (dB/km) în oxigen:

$$L_O = \rho_O \cdot l_O; \qquad l_O = \frac{h_0 - h_t}{sin\beta}$$

$\boldsymbol{L_{H2O}}$ - coeficient liniar de absorbție (dB/km) în vapori de apă:

$$L_{H2O} = \rho_{H2O} \cdot l_{H2O};\ l_{H_2O} = \frac{h_{H_2O} - h_3}{sin\beta},$$

h_0 = 6km - este grosimea echivalentă a stratului de oxigen din atmosfera standard; h_t = 0,176 km - este înălțimea stației terestre deasupra nivelului mării; h_{H2O} - este grosimea echivalentă a stratului de vapori de apă din atmosfera standard. $\boldsymbol{L_1}$ - este lungimea traseului echivalent în oxigen; $\boldsymbol{L_2}$ - este lungimea echivalentă a căii semnalului în vapori de apă.

c) Calculul pierderilor de energie în precipitaţii

Pierderea semnalului de ploaie este calculată folosind formula empirică:

$$L_{pl} = a \cdot I^b \cdot l, \text{ (dB)}, \quad (7)$$

unde I - este intensitatea precipitațiilor (mm/h); l - este lungimea traseului semnalului în stratul de ploaie (km); a și b sunt coeficienți auxiliari.
Coeficienții a și b depind puternic de frecvență, influența altor factori (dimensiunea și temperatura picăturilor de ploaie, polarizarea semnalului) este mult mai slabă

$$a = 4{,}21\cdot 10^{-5} \cdot F^{2{,}49}, \text{ cînd } 2{,}9 \le F \le 54 \text{ GHz}; \quad b = 1{,}41 \cdot F^{-0{,}0779}, \text{ cînd} F \le 25 \text{ GHz}$$

Concluzii

În această lucrare a fost realizat un studiu al caracteristicilor construcției și utilizării sistemelor de televiziune digitală prin satelit. În urma muncii depuse, se pot trage următoarele concluzii:

1. Televiziunea prin satelit este în prezent una dintre cele mai fiabile și economice modalități de a transmite semnale de televiziune de înaltă calitate în orice parte a țării noastre .
2. La construirea unui sistem de comunicații prin satelit, este necesar să se țină cont de caracteristicile tranziției de la o formă analogică a unui semnal de televiziune la una digitală (potriviți spectrul unui semnal de televiziune digitală cu lățimea de bandă a unui canal de comunicație prin satelit), caracteristicile trecerii unui semnal de televiziune printr-un canal de comunicație prin satelit.
3. Comunicații prin satelit VSAT nu are restricții în legătură cu terenul și permite organizarea canalelor de comunicații de telecomunicații unde construirea altor sisteme de comunicații este neprofitabilă sau imposibilă: regiuni slab populate sau nepopulate, teritorii fără infrastructură terestră, rute de transport maritim. Astfel, prin instalarea VSAT, utilizatorul are acces la toate serviciile de telecomunicații practic oriunde în lume: difuzare TV, acces la Internet, telefonie IP cu furnizarea unui număr sau mai multe numere, canale pentru conferințe audio și video.
4. Caracteristicile diferitelor topologii de comunicații prin satelit (VSAT SCPC, VSAT - STAR, VSAT - STAR cu tehnologie TDM/TDMA, VSAT cu tehnologie MESH HYBRID) vă permit să selectați topologia corespunzătoare pe baza cerințelor tehnice specificate pentru sistemul prin satelit (cost, numărul de abonați etc.).
5. În partea practica au fost efectuate calculele în baza echipamentul ales, rezultatul obținut ne arată ca acest sistem va asigura o calitate înaltă a semnalului recepționat. S-a ales un satelit de pământ artificial care îndeplinește următorii parametri: asigură o comunicație stabilă și neîntreruptă la Chișinău în banda Ku, respectiv satelitul Intelsat 33e, situat pe o orbită geostaționară la 60° Est. Au fost selectate echipamente de constelație terestră: un set de echipamente, care include: o antenă cu diametrul de 0,6 m; receptor Castpal DS701; convertor Universal Single LNB. Acest echipament a fost selectat în conformitate cu calculele.
6. Calculul energetic al canalului satelit „nava spațială – stație terestră” a arătat că echipamentul selectat va asigura recepția de înaltă calitate a semnalului satelitului de la satelitul Intelsat 33e către stația de recepție situată în Chișinău.

Referinţe

1. Jerry Whitetaker. Digital television transmission systems. McGraw-Hill, 2000. ISBN: 007-139-14-57.
2. Drury G., Markarian G., Pickavance K. Coding and modulation for digital television. Kluwer Academic Pubishers, 2002. ISBN: 079-237-969-1.
3. Constantin I., Marghescu I., Transmisiuni analogice şi digitale. – Bucureşti: Ed. Tehnică, 1995. ISBN/COD:973-31-0676-3.
4. Arhitectura și principiile de bază de funcționare a sistemelor de comunicații prin satelit [Sursă web]. Disponibil: https://docplayer.ru/75447592-4-arhitekturai-osnovnye-principy-raboty-sputnikovyh-sistem-svyazi.html.

ANALIZA COSTURILOR DE PRODUCȚIE LA ÎNTREPRINDEREA DE TELECOMUNICAȚII

Iulita POPESCU

Departamentul Telecomunicații și Sisteme Electronice, grupa MMRT-221M, Universitatea Tehnică a Moldovei, Facultatea Electronică și Telecomunicații, Chișinău, Republica Moldova

*Autorul corespondent: Iulita Popescu, iulita.popescu@tlc.utm.md

Coordonator științific: Lilia SAVA, dr., conf.univ., *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

Rezumat. *În raport cu Standardul Național de Contabilitate (S.N.C.) "Consumurile" reflectă resursele folosite pentru producția produselor şi efectuarea serviciilor în scopul dobîndirii venitului. Consumurile sunt strîns legate de procesul de producţie şi în final sunt regăsite ca stocuri de producţie finită şi a mărfii în curs de execuţie. Clasificarea costurilor pe categorii de activităţi ale entităţii are un caracter convenţional şi nu este asemănător pentru toate entitatățile economice. De felul cum se utilizează şi cum se foloseşte factorii de producţie depinde în mare parte de performanța produselor şi productivitatea activităţii derulate. Micșorarea costurilor de producţie constituie un obiectiv privilegiat al oricărei entități economice.*

Cuvinte cheie*: consumuri, cheltuieli, capital propriu, entități economice, produs.*

Costurile: clasificarea și analiza lor

Analiza prin costuri reprezintă un cadru foarte important în activitatea entității, deoarece în aceast sector se realizează un ansamblu de aspecte economice legate de consumul și utilizarea parametrilor de producţie. Felul cum se consumă şi cum servesc parametrii de producţie depinde întro mare parte de performanța produselor şi eficienţa activităţii rulate. Micșorarea costurilor de producţie se stabilește, acolo unde este posibil, un obiectiv important al oricărei entități. Pentru a micșora costurile este necesar analiza detaliată a principalelor categorii de cheltuieli care participă la formarea costurilor. Problemele privilegiate ale stabilirii cheltuielilor vizează, întro oarecare măsură, cheltuielile aferente veniturilor entității, în special cheltuielile de exploatare, pe de altă parte, eficienţa diferitelor categorii de cheltuieli.

Cheltuielile, potrivit prevederilor art. 95 din Regulamentul privind aplicarea Legii Contabilităţii nr. 82/1991, reprezintă sumele sau valorile plătite sau de plătit pentru consumurile, lucrările executate şi serviciile prestate de terţi, remunerarea personalului, executarea unor obligaţii legale sau contractuale de către unitatea patrimonială, constituirea amortizărilor şi provizioanelor, consumuri excepţionale[7]. Mulţimea cheltuielilor determină necesitatea unei clasificări în funcţie de diferite criterii reprezentată în Fig. 1.

Figura 1. Criterii de clasificare a cheltuielilor

Cheltuielile ocazionate de fabricarea şi realizarea producţiei se grupează în două grupe mari reprezentate în Fig. 2:

Figura 2. Cheltuielile ocazionate de fabricarea şi realizarea producţiei

Clasificarea pe elemente economice poate fi efectuată conform criteriilor de clasificare reflectate în Fig. 3.

Cheltuieli materiale	• materii prime şi materiale achiziţionate • combustibilul achiziţionat pentru scopuri tehnologice, producerea energiei electrice
Cheltuieli pentru remunerarea muncii	• plata pentru lucrul efectuat • primele
Defalcări pentru necesităţi sociale	• reflectă defalcările în fondurile sociale extrabugetare
Amortizarea fondurilor fixe	• Reflectă suma cotei de amortizare pentru restabilirea fondurilor fixe, calculate pornind de la costul de balanţă al fondurilor fixe, luând în consideraţie indexarea lor
Alte **cheltuieli**	• impozite • plata pentru arenda fondurilor fixe

Figura 3. Criterii de clasificare pe elemente economice

Gruparea cheltuielilor pe **articole de calculaţie,** include următoarele articole prezentate în Fig. 4:

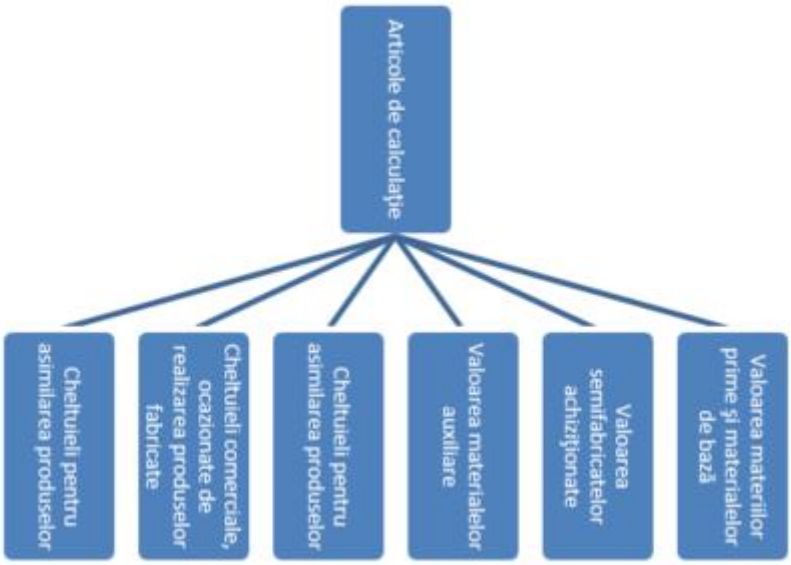


Figura 4. Gruparea cheltuielilor pe articole de calculaţie

Analiza structurii consumurilor la întreprindere

În scopul analizei consumirilor în cadrul companiilor din domeniul telecomunicaţiilor au fost extrase datele din bilanţul contabil prezentat de S.A. Moldtelecom[5]:

În baza datelor din Tab. 1 putem constata că costul producţiei fabricate s-a majorat din anul precedent cu 26050 mii lei. Această abatere a fost condiţionaţă de majorarea consumurilor directe producţie respectiv cu 15156 şi 22268 mii lei. Concomitent, consumurile indirecte de producţie s-au redus în perioada de gestiune cu 11374 mii lei.

Tabelul 1

Analiza consumurilor pe elemente economice

Nr. d/o	Indicatori	Anul precedent		Anul de gestiunea		Abaterea(+,-)	
		suma, mii lei	pondere a, %	suma, mii lei	ponderea, %	Mii lei	%
	1	2	3	4	5	6	7
1	Consumuri de materiale	436127	28.26	451283	28.75	+15156	+0.49
2	Consumuri privind retribuirea muncii	359986	23.32	382254	24.35	+22268	+1.03
3	Consumuri indirecte de producție	747371	48.42	735997	46.89	-11374	-1.53
4	Costul producției fabricate (1+2+3)	1543484	100	1569534	100	+26050	x

Dacă examinăm structura costului de producţie fabricată [6], apoi observăm ca la consumurile materiale ponderea lor s-a majorat cu 0.49. La consumurile privind retribuirea muncii ponderea s-a majorat cu 1.03. Iar ponderea consumurilor indirecte de producţie pe parcursul perioadei de gestiune s-au redus cu 1.53.

Tabelul 2

Analiza structurii cheltuielilor întreprinderii după tipul de activitate în dinamică

Nr. d/o	Indicatorii	Anul precedent		Anul de gestiune		Abaterea(+,-)	
		suma mii lei	pond. %	suma, mii lei	pond. %	Mii lei	%
	1	2	3	4	5	6	7
1	Cheltuielile activ. operaţionale total, inclusiv:	1543484	96.85	1569534	98.1	+26050	+1.25
1.1	costul vînzărilor	1266209	79.45	1274674	79.67	+8465	+0.22
1.2	cheltuieli comerciale	67734	4.25	63953	3.99	-3781	-0.26
1.3	cheltuieli generale şi administrative	52809	3.31	55745	3.48	+2936	+0.17
1.4	alte cheltuieli operaţionale	436127	27.37	451283	28.21	+15156	+0.84
1.5	cheltuieli activ. Investiționale	3930	0.25	2804	0.18	-1126	-0,07
1.6	cheltuieli activ.financiare	46286	2.9	27617	1.73	-18669	-1.17
2	Cheltuieli excepţionale	-	-	-	-	-	-
3	Total cheltuieli (rd.1+2+3+4)	1593700	100	1599955	100	+29031	x

În baza datelor din Tab. 2 putem afirma că cheltuielile totale s-au majorat faţă de anul precedent cu 29031 mii lei. Această creştere a fost condiţionată de creşterea cheltuielilor în urma activității operaţionale.

Din aceste cheltuieli, numai compartimentul "cheltuieli comerciale" și-a redus cheltuielile cu 0.84 puncte. Cît priveşte ponderea cheltuielilor activităţii financiare în totalul cheltuielilor, atunci aici se observă o micșorare faţă de anul precedent cu 1.17 puncte sau cu 18669 mii lei. De asemenea, dacă examinăm structura cheltuielilor activităţii operaţionale în totalul cheltuielilor perioadei, observăm că ponderea lor s-a majorat cu 1.25 puncte.

Calcularea amortizării dispozitivelor utilizate

Pentru calcularea amortizării vom utiliza metoda liniară [3, 4]. În conformitate cu această metodă, amortizarea dispozitivelor va fi decontată (repartizată) egal pe toată durata de funcţionare utilă. Suma amortizării nu depinde de intensitatea folosirii activului şi se stabilește prin raportul valorii uzurabile către numărul anilor (perioadelor gestionare) duratei de funcţionare utilă. Norma de amortizare va fi egală cu raportul 100 la numărul de ani ai duratei de funcţionare utilă.

Cităm un exemplu numeric. Să se calculeze amortizarea calculatorului dacă sînt date: valoarea de intrare – 12460 mii lei, valoarea probabilă rămasă – 5105 mii lei; durata de funcţionare – 6 ani; norma uzurii anuale 20% **Amortizarea anuală constituie:** A_{an} =(12460-5105)/6=1225,83 mii lei.

Tabelul 3

Calculul amortizării unui calculator

Perioada de gestiune	**Valoarea de intrare, mii lei**	**Suma uzurii anuale, mii lei**	**Uzura acumulată, mii lei**	**Valoarea de bilanţ, mii lei**
La data punerii în funcţiune	12460	-	-	12460
La sfîrşitul primului an	12460	1225.83	1225.83	11234.17
La sfîrşitul anului doi	12460	1225.83	2451.66	10008.34
La sfîrşitul anului trei	12460	1225.83	3677.49	8782.51
La sfîrşitul anului patru	12460	1225.83	4903.32	7556.68
La sfîrşitul anului cinci	12460	1225.83	6129.15	6330.85
La sfîrşitul anului șase	12460	1225.83	7354.98	5105.02

Conform datelor din Tab. 3 putem afirma, că pe parcursul duratei de funcţionare a calculatorului observăm că valoare sa inițială a fost de 12460 mii lei, dar cînd sa ajuns în anul 6 de funcționare valoare sa a ajuns la suma de 5105.02 mii lei. Suma uzurii timp de 6 ani este de 7354.98 mii lei, dar suma uzurii anuale este de 1225.83 mii lei.

Concluzii: Analiza costurilor se efectuiază cu scopul de a eficientiza consumurilor de resurse în acțiunea interne ale companiilor și de creare a valorii. Prin urmare reducerea consumurilor de resurse în condiţiile menţinerii valorii existente, sau, în condițiile creşterii acesteia. În scopul efiecintizării activității companiei se propun următoarele **cai de reducere a costurilor de productie:**

- micșorarea consumurilor specifice/tehnologice;
- utilizarea intensiva a capacitatilor de productie a mijloacelor fixe ce va contribui la at scăderea partii de amortizare si implicit reducerea costului de productie;
- cresterea productivitatii muncii;
- reducerea cheltuielilor de desfacere si de publicitate comerciala

Una dintre cele mai importante condiţii pe care trebuie să le îndeplinească măsurile de reducere a costurilor este să nu afecteze în mod negativ calitatea produselor și serviciilor prestate [6]. Concomitent, soluţiile nu trebuie să aibă rezultat negative asupra indicatorilor entității de natura volumului de activitate. De aici rezultă că soluţiile de optimizare a nivelului costurilor trebuie să fie de lungă durată.

Referinţe

1. Lilia Sava, Popescu Iulita. Contabilitate. Partea I. Note de curs. Chișinău, Editura ”Tehnica – UTM”, 2022. ISBN 978-9975-45-804-7., pag. 72-74.
2. Lilia SAVA, Blanaru Vasile. Statistica ramurii. Note de curs. Chișinău, Editura ”Tehnica – UTM”, 2022, pag. 4-15. ISBN 978-9975-45-808-5.
3. Lilia SAVA, Maria GRIŢCO. Organizarea și analiza activităţii economice în domeniul telecomunicațiilor. Ghid metodic. Partea II. Chișinău, Editura ”Tehnica –UTM”, 2018.
4. Lilia SAVA, Daniela Vortolomei. Organizarea și analiza activităţii economice în domeniul telecomunicațiilor. Note de curs. Chișinău, Editura ”Tehnica –UTM”, 2022, pag. 45-51. ISBN 979-9975-45-805-4.
5. Bilanțului contabil prezentat de SA Moldtelecom. [online] Disponibil: https://doi.org/10.1016/j.catcom.2008.08.027
6. Metode de reducere a costurilor. [online] Disponibil: https://cuttingcost.wordpress.com/
7. Legea contabilității și raportării financiare Nr. 287 din 15.12.2017, intrată în vigoare la 01.01.2019.// Monitorul Oficial al Republicii Moldova Nr. 1-6. [online]. Disponibil: https://www.legis.md/cautare/getResults?doc_id=120938&lang=ro

DEZVOLTAREA SISTEMULUI EFICIENT DE CONTROL AL ECHIPAMENTELOR TEHNOLOGICE ÎN PRODUCERE A CIMENTULUI

Ion RUSNAC

Departamentul Telecomunicații și Sisteme Electronice, grupa SISRC-211M, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova.

*Autorul corespondent: Ion Rusnac ion.rusnac1@mail.ru

Coordonator științific: Vladimir CICLICCI, conf. univ., dr., *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

Rezumat. *A fost efectuată o analiză complexă a sistemului existent de tunuri pneumatice în producere a cimentului și a modului lor de operare și a identificat modalități de îmbunătățire a eficienței acesteia. Au fost elaborat și implementat sistemul de control al echipamentului primar, în baza căruia eficiența procesului a crescut semnificativ. Implementarea acestui sistem a redus la zero toate opririle de echipament, astfel fiind reduse și costurile adiționale.*

Cuvinte cheie*: sisteme Simatic S7, WinCC, WinCC Run Time*

Introducere

Evoluția constantă pe care o trăim astăzi a adus în fiecare segment, domeniu de activitate, industrie, noi necesități, cerințe și, printre acestea, nevoia constantă de a automatiza procesele care anterior erau efectuate de oameni și sisteme electromecanice și acum incorporează chiar și sisteme de inteligența artificială.

În acest sens, uzina de ciment "LAFARGE CIMENT MOLDOVA, S.A", lider pe piața materialelor de construcții, se focusează pe o modernizare impunătoare a procesului tehnologic de producere a cimentului și a utilajelor de automatizare.

Acest proiect a fost ales ca subiect al prezentei lucrări deoarece implementarea lui a adus un aport semnificativ în eficiența și modernizarea echipamentului primar de producere a cimentului, totodată complexitatea acestuia fiind o provocare (în sensul pozitiv al cuvântului) în activitatea mea în cadrul uzinei. Proiectul descris mai jos constă în instalarea unui sistem de tunuri pneumatice care permit evitarea blocajelor de material în echipamentul de producere, mai exact în Cuptorul rotativ, unde se arde mixtul de materie primă și în Turnul de cicloane care joacă rolul unui schimbător de căldură pentru gazele fierbinți din Cuptor și materia primă.

Inițial, echipamentul care prezintă risc de lipituri și respectiv blocare de material a fost dotat cu 30 de tunuri pneumatice care operau în mod haotic și necontrolat de către operatori. Astfel, eficiența lor era evaluată ca una scăzută deoarece problema blocării materialului pe flux era periodic prezentă. În anul 2020, la nivel de uzină a fost setat un obiectiv prioritar și urgent de reducere la minim blocările pe Turnul de Cicloane. A fost efectuată analiza completă a întregului sistem și a fost propusă proiectarea și instalarea a 50 de tunuri pneumatice adiționale, cu un sistem automatizat de comandă la distanță de către operatori din camera de comandă.

1. Analiza eficienței sistemului existent de control a echipamentului

Industria de producere a cimentului reprezintă un proces complex care presupune respectarea unei balanțe chimice în ceea ce ține toate tipurile de materii utilizate, fie materii prime din carieră sau combustibilii utilizați la ardere. Una din constrângerile primare în această industrie sunt alcaliile, sulful și clorura: Aporturile excesive ale acestor compuși pot duce la depuneri și blocaje în sistemul cuptorului. Acolo unde acestea nu pot fi captate în clincherul de ciment, sunt necesare sisteme de prevenire a lipiturilor sau sisteme de *by-pass* pentru a elimina excesul de compuși din cuptoarele de preîncălzire/precalcinare (în special pentru uzinele care utilizează combustibil alternativ).

Din acest motiv a fost efectuat o analiză înainte de implementarea a sistemului, sa analizat că tunurile pneumatice care există nu au o eficientă semnificativă, nu pot fi monitorizate și controlate ceia ce a demonstrat necesitatea de implementarea unui nou sistem de comandă pentru a exclude blocările de cicloane și de menține echipamentul primar întru continue funcționare .

După analiză și stabilirea designului, capacității și amplasării, următoarea etapă a fost proiectarea sistemului de comandă. Etapa de proiectare include următoarele:

- alegerea controlului necesar cu toate modulele de semnale pentru a efectua legătură cu procesor central Simatic S7-400 al sistemului de control;
- proiectarea schemelor electronice și montajul electric al fiecărui tun pneumatic separat;
- elaborarea softului în Simatic PCS 7 unde s-a creat și imaginea în timp real al sistemului de comanda.

O analiză comparativă de pierderi de energie termică și electrică până implementarea sistemului de comandă elaborat este afișată în Tab. 1.

Tabelul 1

Analiză comparativă de pierderi de energie termică

Data de blocare	Unde	Durata blocări	Funcționare echipament primar	Gas Natural consumat	Consumul de amestec de combustibil	Energia electrică Apx.5 MW/h
Anul 2019 deplin	Blocare ciclon x7 (x6-1A, x1-1B)	21.2	3235	46379 m3	4138 GJ	105850 kWh
Anul 2020 deplin	Blocare ciclon x11 (x6-1A, x5-1B)	51.9	3677	57267m3	3538 GJ	259550 kW/h
Anul 2021 deplin	Blocare ciclon x6 (x3-1A, x2-1B, x1-2A)	42	2899	51387 m3	1513 GJ	209900 kW/h

Pe parcursul anului 2020 echipamentul primar a funcționat 3677 ore cu blocare de cicloane, iar în anul 2021 echipamentul a funcționat 2899 ore cu o stopare, respectiv în acești doi ani au fost suportate pierderilor de energie termică și electrică.

2. Dezvoltarea sistemului de control a echipamentului primar de producere a cimentului

Pentru dezvoltarea sistemului de comandă a fost necesar de proiectat schema funcțională și algoritmul de funcționare a întregului sistem, unde a fost descris cum decurge transmiterea de semnale de la partea fizică la partea logică. După care a fost necesar proiectarea schemelor electronice a sistemului de comandă, unde a fost specificat componentele folosite și principiu de funcționare a schemelor electrice. În Fig. 1 este prezentat tabloul de comandă și schema electronice în baza controlerului ET 200M .

Au fost elaborate schemele electronice și efectuarea conexiunelor fizic, după care a fost necesar de elaborat programul de logică a tunurilor penumatice, programul este creat în PCS S7 unde sa elaborat logica de funcționare a fiecărui turn sepărat sau în grup în dependență cum au fost stabilite grupele de funcționare. Au fost descriși toți pași pentru a crea un soft de la punctul zero și pînă la verificare a semnalelor din camera de comandă către tunurile pneumatice, la fel au fost descris principiu de funcționare a blocurile logice folosite. În Fig. 2 este indicate un exemplu de consecutivitatea a tunurilor pneumatice.

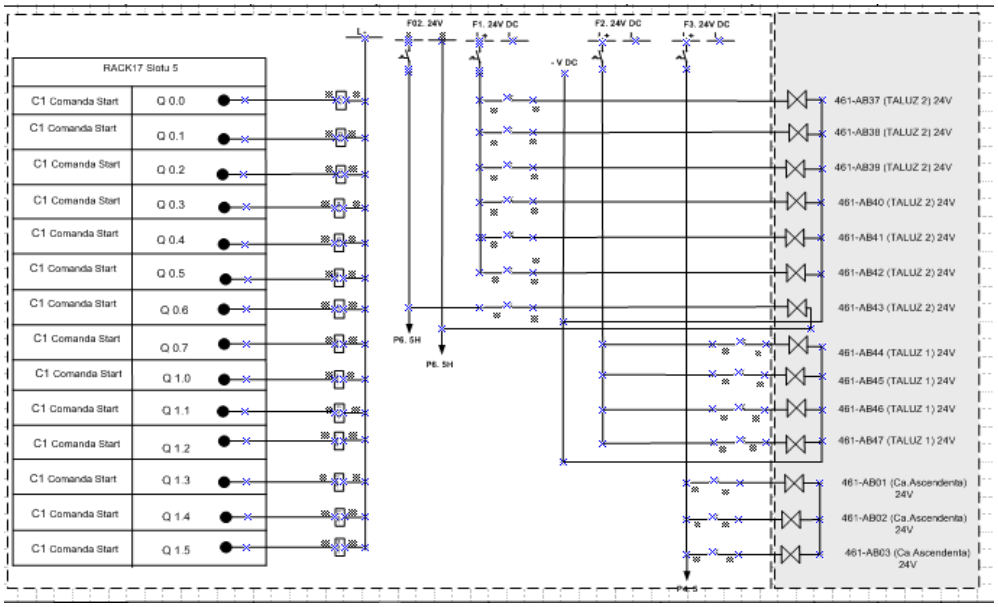


Figura 1. Schema electronică

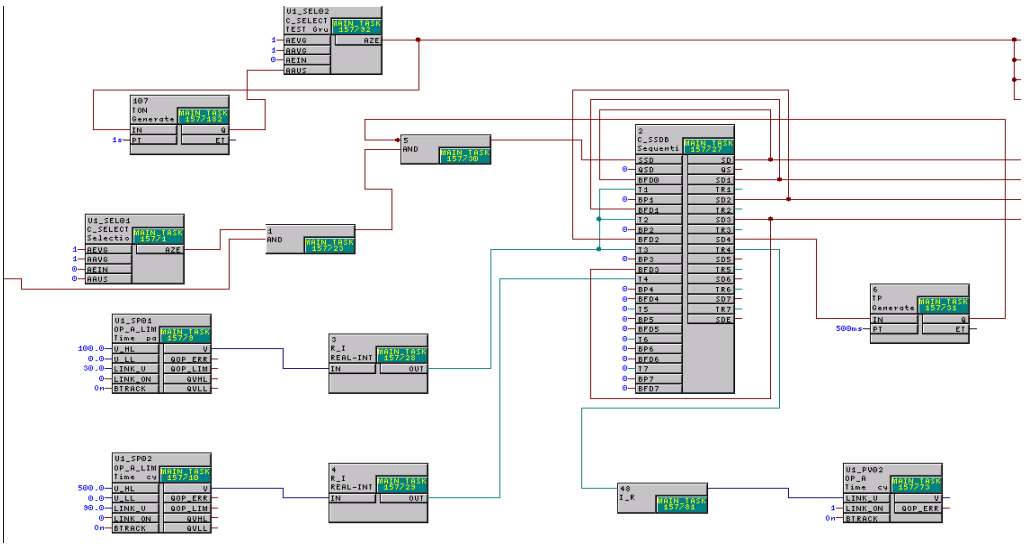


Figura 2. Programarea consecutivității tunurilor pneumatice

După elaborarea softul a fost proiectarea imaginii în WinCC pentru sistemul de comandă, respective a fost descriși pași de crearea unei imaginei noi sau a unui nou proiect la fel și etapele de conectarea a semnalelor din baza logică CFC către imaginea de operare.

Fig. 3 reprezintă aspectele reale de funcționare a sistemului în WinCC RunTime, în care se vede că toate grupele sunt selectate, avem disponibilitatea și sunt puse în distanta toate tunurile pneumatice. La fel putem monitoriza timpurile, prezența mesajelor. Aceasta este baza după care se conduc operatorii pentru a urmări funcționarea sistemului de comandă al tunurilor pneumatice.

S-a efectuat calculul fiabilității care ne-a demonstrat îndeplinirea condițiilor stabilite **Ps(t)=0,961> $P_{s.dat}$(t)=0,95**, ceea ce reprezintă un indicator foarte important deoarece ia în considerare toate condițiile căror fa fi supus sistemul în perioada de exploatare. În urma calculelor a fost dedus timpul mediu de bună funcționare a sistemului - 3294 ore. La fel a fost calculat coeficientul de garanție, 0.4 ani, care caracterizează perioada de funcționare a sistemului cu menținerea valorii de fiabilitate.

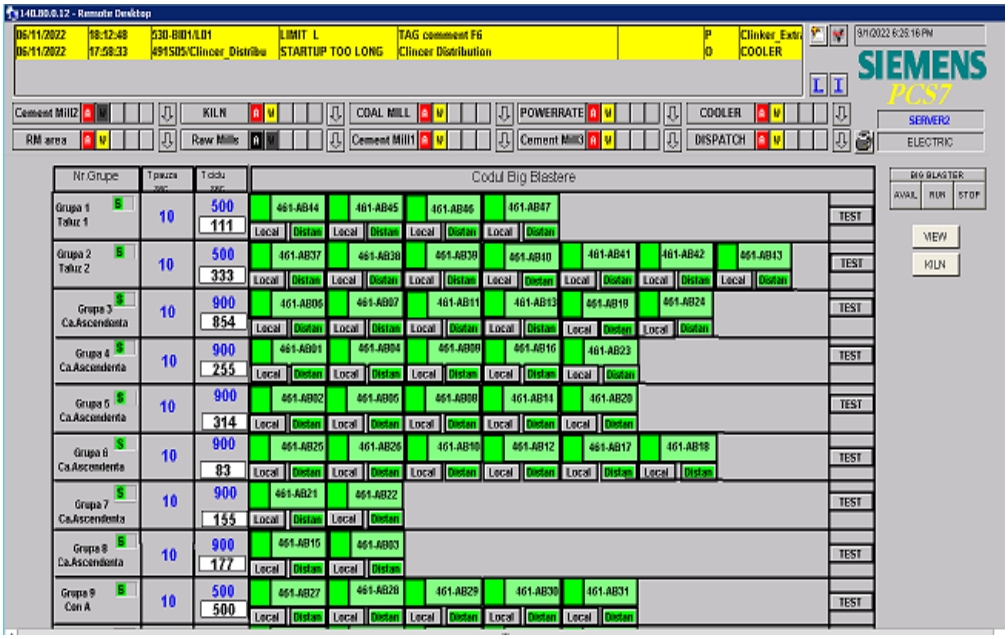


Figura 3. WinCC RunTime

3. Studiu de fezabilitate al implementării sistemului de control dezvoltat

Pentru determinarea efectului economic al sistemului de comandă până la implementarea sistemului, au fost colectate valorile cheie pe partea de costuri, efectuată o analiză de date și parametri. Toate acestea sunt grupate care cuprinde toate blocările a cicloanelor care au dus la oprirea accidentală a echipamentului de producere a cimentului. În dependență de durata și periodicitatea opririlor accidentale, au fost estimate pierderile financiare legate de consumul de gas natural, consum de combustibil solid și consum de energie electrică. Informația pe partea de volume a fost extrasă din sistemul de raportare a procesul tehnologic.

Ca referință au fost utilizate și analizate datele din anul 2019, perioada în care au fost înregistrate 6 blocări de cicloane. Suma totală pentru 2019, cheltuită din cauza consumului cauzat de oprirea accidentală a echipamentului.

$$\boldsymbol{CT = CDg + CDGj + CME = 1548495 + 505299 + 596018 = 2\,649\,812\ lei\,,} \qquad (1)$$

unde: ***CDg*** - costurile directe pentru consumul de gaz natural; ***CDGj*** - costurile directe de combustibil solid; ***CME*** - costurile directe pentru energia electrică.

Astfel, mai mult de 2 milioane de lei a fost pierderea financiară cauzată de blocările de material. Pentru a justifica investiția într-un proiect nou care ar soluționa problema blocărilor, au fost estimate costurile aferente a materialelor și echipamentelor necesare pentru instalarea sistemului de deblocare și sistemului de control de la distanță. Toate datele sunt prezentate în Tab.2.

În Tab.3 este specificat cheltuielile necesare pentru realizarea sistemului astfel sistemul de comandă a tunurilor pneumatice se estimează la **739 764 lei.**

Tabelul 2

Costul sistemului de control

Cheltuieli	Suma, lei
Costuri material	607158
Cost operational (10%)	60715
Costuri pentru retribuirea munci	58 212
Contribuţii sociale	11 060
Asigurări medicale	2619
Total	739764

După implementarea și punerea în funcțiune a sistemului de comandă, s-a efectuat o analiză a eficienței sistemului nou. Astfel din Tab. 3 putem observa eficiența înaltă care se definește prin zero opriri de echipament din cauza blocării de material în Turnul de Cicloane. Ca urmare, pierderile financiare la fel au fost reduse la 0.

Tabelul 3

Pierderi economice înainte și după implementarea sistemului dezvoltat

Data de blocare	Unde	Durata blocări	Funcționare echipament primar	Gas natural consumat	Consumul de amestec de combustibil	Energia electrică Apx.5MW/h	Total Cost energie electrică și termică (pierderi)
Anul 2019 deplin	Blocare ciclon x11	51.9	3677	57267m3	3538 GJ	259550 kW/h	**3185783 lei**
Anul 2020 deplin	Blocare ciclon x6	42	2899	51387 m3	1513 GJ	209900 kW/h	**2411392 lei**
Anul 2021 deplin	Blocare ciclon x0	0	3555	0 m3	0 GJ	0 kW/h	**0**
Anul 2022 deplin	Blocare ciclon x0	0	534	0 m3	0 GJ	0 kW/h	**0**

Concluzii

Au fost realizate schemele electronice sistemului de comandă, descrise principiile de funcționare a schemelor și conectarea comenzilor efectuate cu controlerul ET 200M. Adițional au fost specificate componentele folosite în realizarea schemelor electronice și rolul fiecăruia.

A fost elaborat softul PCS S7, care coordonează cu sistemul de comandă automatizat care se bazează pe funcționarea după timp pauză și timp ciclu a tunurilor pneumatice. Au fost descrise toate etapele necesare pentru crearea și elaborarea proiectului, toate etapele de configurare în Step 7 și o scurtă descriere a elementelor logice folosite în sistemul de comandă.

Referinţe

1. PLC program SIMATIC STEP7 [citat 10.10.2022].
2. Disponibil:https://new.siemens.com/global/en/products/automation/industry-software/automation-software/tia-portal/software/step7-tia-portal.html
3. Controlerele Simatic S7-400. [citat 10.10.2022]
4. Disponibil: https://new.siemens.com/global/en/products/automation/systems/industrial/plc/simatic-s7-400.htm
5. Simatic PCS 7. [citat 15.10.2022]
6. Disponibil: https://new.siemens.com/global/en/products/automation/process-control/simatic-pcs-7.html
7. ET 200M și modulele folosite. [citat 20.10.2022]
8. Disponibil: https://mall.industry.siemens.com/mall/en/WW/Catalog/Products/5000114
9. Simatic WinCC. [citat 25.10.2022]
10. Disponibil: https://support.industry.siemens.com/cs/document/11841504/how-do-you-integrate-an-existing-wincc-project-into-a-step-7-project-?dti=0&lc=en-WW

WI – FI 7 – NOUA INOVAȚIE ÎN SFERA COMUNICAȚIILOR WIRELESS

Victor SCLIFOS

Departamentul Telecomunicații și Sisteme Electronice, grupa IMTC-201, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova.

*Autorul corespondent: Victor SCLIFOS, victor.sclifos@tse.utm.md

Coordonator științific: Roman GRIȚCO, *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

Rezumat:

Lucrarea dată reflectă soluționarea necesității suprasolicitării conectării dispozitivelor la rețelele wireless prin implementarea unui standard nou 802.11be, cunoscut sub denumirea de Wi-Fi 7. Standardul dat include în sine aplicarea unor noi facilități precum modularea QAM, operațiuni multi-link MLO și preamble puncturing, care au drept scop remedierea și îmbunătățirea conexiunilor wireless. Prin urmare, în lucrare sunt analizate principiile de funcționare și impactul acestor facilități asupra noului standard și modul de lucru a Wi – Fi 7.

***Cuvinte cheie**: IEEE, WLAN, QAM, OFDMA, MLO, preamble puncturing*

Introducere

Actualmente, se atestă o majorare exponențială în necesitatea conectării dispozitivelor la tehnologia WLAN, din cauza suprasolicitării și convenabilității utilizării acesteia. În ultima perioadă, se remarcă creșterea cerințelor aplicațiilor în ceea ce privește debitul și latența la conectare. Printre exemplele tipice ale acestor aplicații se includ videoclipuri 4K și 8K, care implică o rată de transmisie de până la 20 Gbps, realitatea virtuală (VR)/realitate augmentată (AR), jocuri online, care necesită o latență mai mică de 5 ms, lucrul de la distanță, conferințe video online și cloud computing. Confruntându-se cu cerințe atât de înalte, Wi-Fi 6 - în prezent cel mai recent standard Wi-Fi - este și el insuficient pentru îndeplinirea volumului suprasolicitat de viteză, în pofida scopului său menit să îmbunătățească experiența utilizatorului în scenarii de înaltă densitate. Drept urmare, IEEE a decis să lanseze un nou amendament intitulat IEEE 802.11be EHT, acesta fiind recunoscut și sub numele de Wi-Fi 7.

Modul de funcționalitate a Wi – Fi 7

Wi-Fi 7 are drept scop să mărească debitul WLAN până la 30 – 46 Gbps și să asigurare o conectare și acces cu latență scăzută. Pentru a atingerea acestor obiective, standardul definește modificări atât la stratul fizic (PHY) cât și la nivelul MAC. În cadrul standardului 802.11be, benzile de frecvență de 2,4 GHz și 5 GHz sunt spectre fără limite și congestionate. Atunci când rulează aplicații emergente (cum ar fi VR/AR), rețelele Wi-Fi existente se confruntă inevitabil cu o calitate scăzută a serviciului (QoS). Pentru obținerea un debit de 30 Gbps, Wi-Fi 7 va implementa banda de frecvență de 6 GHz și va extinde noi moduri de lățime de bandă, inclusiv 240 MHz învecinat, 160+80 MHz, 320 MHz învecinat și 160+ MHz. În tabelul 1 sunt prezentate diferențele majore între standardele existente și Wi – Fi [1].

Tabelul.1.

Compararea generațiilor de Wi - Fi

	Wi-Fi 5	***Wi-Fi 6***	***Wi-Fi 6E***	***Wi-Fi 7***
IEEE Standard	802.11ac	802.11ax	802.11ax	802.11be
Frecvență	5 GHz	Bandă – dublă (2.4 GHz, 5 GHz)	Bandă – triplă (2.4 GHz, 5 GHz, 6 GHz)	
Lungimea de bandă	20, 40 , 80, 80+80, 160MHz			20, 40 , 80, 80+80, 160, 320MHz
Multiplexarea	OFDM	OFDMA		

	Wi-Fi 5	Wi-Fi 6	Wi-Fi 6E	Wi-Fi 7
Modulația	254QAM	1024QAM		4096 (4K) QAM
Securitatea	WPA2	WPA3		
Rata maximă teoretică de date transmise	3.5 Gbps	9.6 Gbps		30 – 46 Gbps

În comparație cu Wi-Fi 6, Wi-Fi 7 implementează un șir de inovații tehnice:

- ***QAM (Quadrature Amplide Modulation)*** este o metodată prin care pachetele de date sunt traduse în semnale analogice transmise fără fir. Wi-Fi 6E acceptă 1024 QAM, în timp ce Wi-Fi 7 îl crește la 4K QAM. Modulația de cel mai înalt nivel acceptată de Wi-Fi 6 este 1024-QAM, care permite fiecărui simbol de modulație să transporte până la 10 biți. Pentru a îmbunătăți și mai mult rata, Wi-Fi 7 introduce 4096-QAM, astfel încât fiecare simbol de modulație să poată transporta 12 biți. Creșterea de la 1024 QAM la 4K QAM are ca rezultat o creștere a debitului cu 20%. Rezultatul este o eficiență, capacitate și rate mai mari de transmisie a datelor în comparație cu Wi-Fi 6/6E.

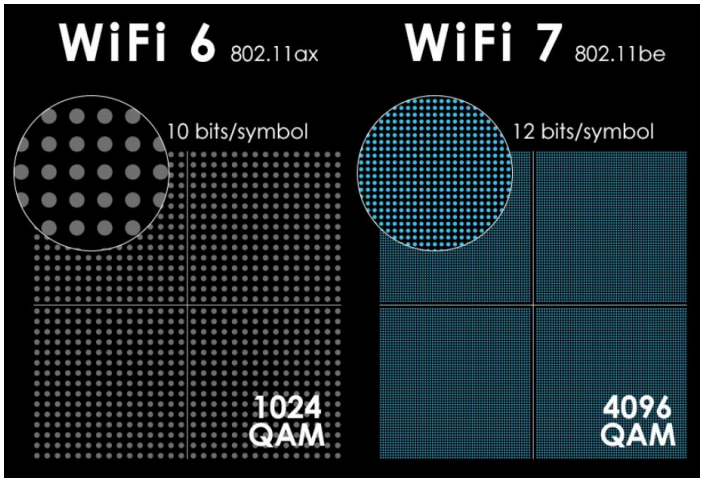


Figura 1. Structura QAM în cadrul standardelor Wi – Fi 6 și Wi – Fi 7

- ***Multi-Link Operation (MLO)*** – Wi-Fi 7 dublează fluxurile MU-MIMO (Multi-User, Multiple Input, Multiple Output) de la 8 la 16 și adaugă Multi-Link Operation (MLO) la mix. Cu standardele wireless actuale, clienții aleg o bandă pentru a transmite date. Dacă se schimbă condițiile care vor face operarea pe o bandă diferită mai eficientă, clientul va comuta automat. De exemplu, un router Wi-Fi 6E ar alege un singur canal pe o singură bandă (2,4GHz, 5GHz sau 6GHz) atunci când se conectează la un client Wi-Fi 6E. Cu toate acestea, MLO permite routerelor Wi-Fi 7 să se conecteze la un client Wi-Fi 7 prin mai multe benzi și canale wireless simultan, ca o singură conexiune agregată. MLO poate combina mai multe frecvențe pe benzi într-o singură conexiune. Un router Wi-Fi 7 se poate conecta la un dispozitiv Wi-Fi 7 pe două sau mai multe canale în benzi diferite simultan. MLO poate permite canale mai largi, capabile să transmită mai multe date.

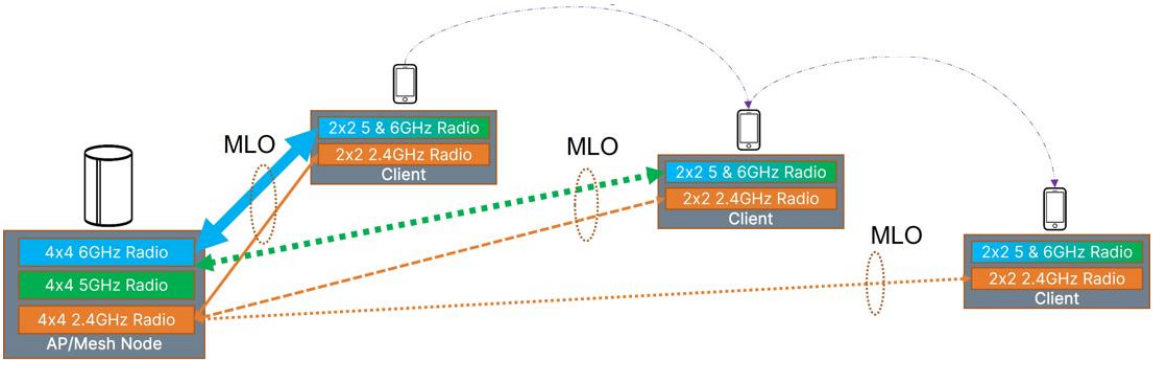


Figura 2. Implementarea MLO în cadrul Wi – Fi 7

- ***Orthogonal Frequency Division Multiple Access (OFDMA)*** permite rate de date și mai mari și o utilizare mai eficientă a spectrului disponibil. OFDMA permite mai multor dispozitive să transmită date pe același canal, ceea ce îmbunătățește debitul general al rețelei. În timp ce OFDMA permite rate de date mai rapide, contribuie, de asemenea, la stabilitatea conexiunilor la o rețea fără fir. Dispozitivele multiple, capabile să transmită date simultan, reduce conflictele într-o rețea – îmbunătățind stabilitatea generală a conexiunilor.

- ***Preamble Puncturing*** permite dispozitivelor fără fir să creeze o porțiune de canale „ocupate" atunci când sunt disponibile. Să presupunem că un dispozitiv comunică continuu pe un canal de 160 MHz, dar utilizează doar 20 MHz. În generațiile anterioare de Wi-Fi, acest lucru ar împiedica punctele de acces să utilizeze acest spectru particular. Preamble Puncturing va permite unui punct de acces să creeze lățimea de bandă rămasă din acel canal pentru a fi utilizată fără interferențe [2-6].

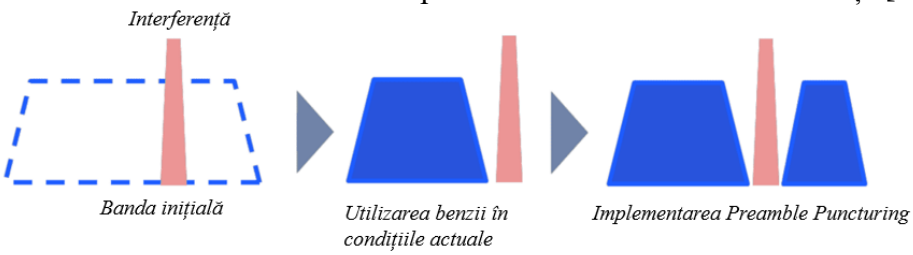


Figura 3. Implementarea Preamble Puncturing

Concluzii

Wi-Fi 7 reprezintă un pas înainte foarte semnificativ în experiența utilizatorului, în special în aplicațiile solicitante, precum cloud gaming, AR/VR imersiv, streaming video 8K, etc. Standardul dat va fi pe larg aplicat și în cadrul industriei 4.0. Ca consecință, utilizatorii finali vor beneficia de o viteză mai mare, latență mai mică, putere sporită, arie de acoperire înaltă și un număr mai sporit de dispozitive conectate simultan în comparație cu standardele deja existente, precum Wi-Fi 6/6E (deja bun).

Referințe

1. INTEL, *What Is Wi-Fi 7?* [online], [accesat 18.02.2023]. Disponibil: What Is Wi-Fi 7? (intel.com)
2. HILL B., *Wi-Fi 7 is Coming: Here's What You Need to Know* [online], [accesat 19.02.2023]. Disponibil: https://www.tomshardware.com/news/wi-fi-7-explained
3. HUAWEI TECHNOLOGIES CO. LTD., *What Is WiFi 7?* [online], [accesat 18.02.2023]. Disponibil: What Is WiFi 7? WiFi 7 vs. WiFi 6 - Huawei
4. MICRO-STAR INTERNATIONAL CO. LTD., *Introducing Wi-Fi 7: The Next Wi-Fi Revolution?* [online], [accesat 20.02.2023]. Disponibil: https://www.msi.com/blog/what-is-wifi-7
5. ELECTRONIC SPECIFIER, *Wi-Fi 7 – A huge step forward for networking* [online], [accesat 20.02.2023]. Disponibil: https://www.electronicspecifier.com/industries/wireless/wi-fi-7-a-huge-step-forward-for-networking
6. SBEGLIA NIN C., *What is preamble puncturing and what does it have to do with Wi-Fi 7?* [online], [accesat 18.02.2023]. Disponibil: https://www.rcrwireless.com/20220216/network-infrastructure/wi-fi/what-is-preamble-puncturing-and-what-does-it-have-to-do-with-wi-fi-7

METODOLOGIA DETERMINARII NIVELULUI DE RADIAŢIE EMIS DE TELEFOANELE MOBILE ŞI ECHIPAMENTELE RADIO. SAR (SAR - SPECIFIC ABSORPTION RATE)

Ariana ȚENU[1], Andrei APOSTOL[2]

[1]*Departamentul Telecomunicații și Sisteme Electronice, grupa IMTC-221, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*
[2]*Departamentul Telecomunicații și Sisteme Electronice, grupa RST-221, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

*Autorul corespondent: Ariana Țenu, tenuariana77@gmail.com

Coordonator științific: Serafima SOROCHIN, asistent universitar, *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

Rezumat. *Metodologia de determinare a nivelului Specific Absorption Rate (SAR) este utilizată pentru a evalua expunerea la radiații electromagnetice emise de telefoane mobile și alte echipamente radio. Pentru a măsura SAR-ul, sunt necesare echipamente specializate, cum ar fi sonde și fantome, și este important să se urmeze norme de siguranță și standarde internaționale. Măsurarea SAR este importantă pentru a asigura siguranța utilizatorilor de telefoane mobile și a altor dispozitive electronice și pentru a evalua nivelurile de radiații din mediul înconjurător. Este important ca utilizatorii să fie conștienți de nivelul SAR al dispozitivelor lor și să ia măsuri pentru a reduce expunerea la radiații electromagnetice.*

Cuvinte cheie*: radiații electromagnetice, metodologie de măsurare, unde radio, expunere la radiație*

Introducere

Cu siguranță deducem faptul că efectele radiațiilor emise de telefoanele mobile au un impact negativ asupra corpului uman. Aceste radiații au aceeași lungime de undă ca microundele pe baza cărora funcționează cuptoarele cu microunde. În mod reprezentativ telefoanele mobile operează în benzile de frecvenţă de 850MHz, 900MHz, 1800 MHz şi 1900MHz, acestea incluzându-se în limitele benzii 915MHz-2450MHz folosită de cuptorul cu microunde. Acesta este exact diapazonul în care moleculele de apă rezonează, pătrunzând într-o vibraţie rapidă care se soldează cu încălzire. Pentru a demonstra influența negativă a radiațiilor emise, ne putem imagina creierul uman care conține 75% apă și este predispus la distrugerea celulelor nervoase, care nu au capacitatea de a se regenera comparativ cu celelalte celule ale corpului uman. Desigur sângele nostru nu va începe să fiarbă însă molecule de ADN şi alţi constituenţi celulari vor începe să se deterioreze.

Câmpul magnetic este mediul prin care se propagă toate aceste unde, iar faptul că există mai multe surse, aceste unde se pot suprapune, astfel amplificând impactul asupra corpului.

Orice dispozitiv electronic, nou procurat, are capacitatea de a emana unde radio, astfel fiind considerat drept sursă de radiație pentru orice corp viu aflat în apropierea lui.

Caracteristicile esențiale SAR

SAR (Specific Absorption Rate)-rata specifică de absorbţie care reprezintă unitatea de măsură pentru radiaţia emisă de telefoanele mobile şi absorbită de corpul uman. Cu cât SAR-ul unui telefon este mai mare, cu atât telefonul va emite mai multă radiaţie către noi. Este important să fim conştienţi de faptul că unele telefoane emit de două sau trei ori mai multă radiaţie decât alte dispozitive, deci utilizarea unui telefon cu un SAR mai mare este echivalentă cu folosirea simultană a două sau trei telefoane cu un SAR mai mic.

SAR se masoară în Waţi per Kilogram (W/Kg) şi este estimată pe o mică mostră de ţesut (1g sau 10g). Radiaţia absorbită de corp este măsurată de mai multe ori, la diferite frecvenţe şi valoarea

maximă este reţinută. Sunt unele diferenţe uşoare între standardele americane şi europene de măsurare:

Agenţia americană de regularizare este FCC (Federal Communications Commission) şi măsoară SAR pe 1g de ţesut. SAR-ul maxim admis pentru SUA este de 1.6 W/Kg.

Agenţia Uniunii Europene de regularizare este CENELEC (Comité Européen de Normalisation Électrotechnique) şi măsoară SAR pe 10g de ţesut. SAR-ul maxim admis pentru UE este 2.0 W/Kg.

Telefoanele mobile pot avea una sau mai multe valori SAR pentru diverse regiuni ale corpului şi pentru diverse frecvenţe radio. Este important să conștientizăm faptul că odată ce sunt pornite, telefoanele mobile emit radiții. Deși telefonul nu este folosit pentru convorbiri telefonice, el în continuu păstrează legătura cu rețelele de telefonie mobilă şi permanent face schimb de pachete și date prin intermediul undelor radio.

Una dintre obligațiunile producătorilor de telefoane mobile este să publice valorile SAR pentru dispozitivele produse. Aceste valori ar trebuie să fie publicate atât online cât și etichetate în manualul dispozitivului.

Autenticitatea rezultatelor conform celor 2 standarte pot fi verificate de autorităţile de reglementare dar şi de consumatori în laboratoare special.

În România există un astfel de laborator în cadrul ICMET Craiova (laboratorul de evaluare şi certificare SAR, ICMET, Craiova).

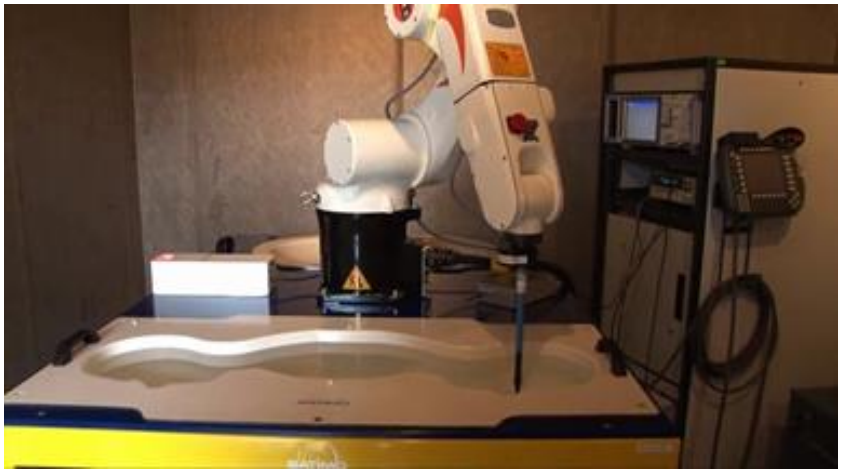


Figura1. Metodologia măsurării nivelului SAR cu ajutorul utilajului special înzestrat cu sondă, echipament electronic, lichid specific şi formă/fantomă

Măsurarea nivelului SAR (Specific Absorption Rate) este o metodă importantă pentru evaluarea expunerii la radiațiile electromagnetice produse de dispozitivele electronice, cum ar fi telefoanele mobile și tabletele. Pentru a măsura nivelul SAR, se utilizează un utilaj special echipat cu o sondă, un echipament electronic, un lichid specific și o formă sau fantomă.

Iată pașii generali pentru a efectua o măsurare a nivelului SAR:

1. Se pregătește fantoma sau forma - aceasta poate fi o formă umană sau animală sau o fantomă, o simulare a corpului uman care este proiectată să reproducă proprietățile electrice și fizice ale țesutului uman.
2. Se introduce sonda în interiorul fantomei sau forme, poziționându-l în locul unde se dorește să se facă măsurarea SAR.
3. Se utilizează un echipament electronic specializat pentru a genera radiații electromagnetice, care sunt dirijate către fantomă sau forma prin intermediul unei antene.

4. Se măsoară nivelul de radiații electromagnetice absorbit de către fantomă sau formă prin intermediul sondei.
5. Se măsoară temperatura lichidului specific, care este utilizat pentru a simula efectele pe care le-ar avea țesuturile biologice în contact cu radiațiile electromagnetice.
6. Se calculează nivelul SAR pentru zona specifică din interiorul fantomei sau forme în care a fost efectuată măsurarea.
7. Este important să se asigure că măsurările sunt efectuate în condiții standardizate pentru a obține rezultate precise și comparabile. De exemplu, trebuie să se utilizeze aceleași tipuri de fantome sau forme, aceleași metode de poziționare a sondei și aceleași setări ale echipamentului electronic pentru a asigura consistența măsurătorilor.
8. În general, măsurarea SAR este realizată de către specialiști în domeniul fizicii sau ingineriei electromagnetice, deoarece este nevoie de cunoștințe specializate pentru a interpreta rezultatele măsurătorilor și a face recomandări privind siguranța dispozitivelor electronice.

La finalizarea măsurării, softul face calculul valorii SAR-ului şi în zona mediană vedem pentru canalul de mijloc calculată valoarea SAR-ului raportat la 1g sau 10g. Standartul European impune standartul sau norma maximă de 2W/kg pentru 10g de ţesut uman raportată la 10g.

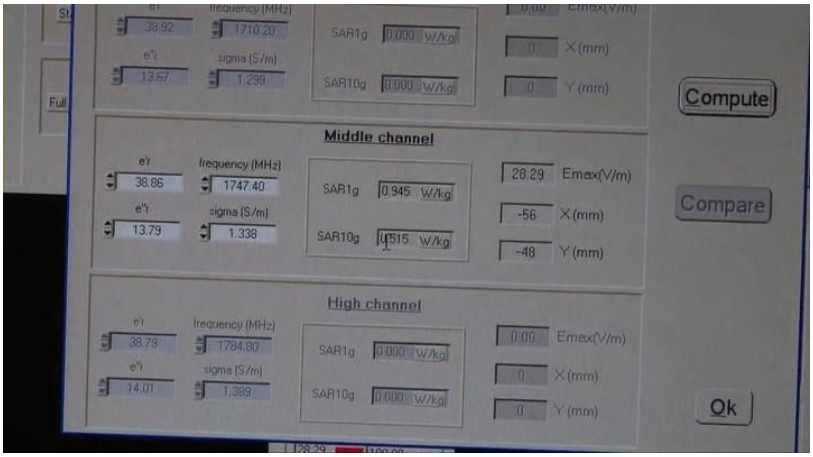


Figura 2. Calculul nivelului SAR prin intermediul softului în zona de median pentru canalul de mijloc

Pentru a finisa cu success măsurările asupra telefonului mobil , el trebuie testat şi în camera semialihoică.În această încăpere se fac încercările de imunitate la cîmp electromagnetic, se verifică dacă telefonul îşi păstrează starea de funcţionare, de conexiune cu un alt telefon mobil în timpul încercării de imunitate la cîmp electromagnetic, se produce un cîmp cu o intensitate specificată în domeniul de frecvenţă de la 80Mhz-1Ghz sau mai mult în funcţie de standart şi se urmăreşte cu ajutorul unei camera video amplasate în faţa telefonului mobil, fapt ce demonstrează că telefonul mobil îşi păstrează starea de funcţionare.

Altă metodă de măsurare a SAR-ului unor echipamente radio mai mari sau unor staţii de bază, antene etc.,se poate de folosit o antenă specific care are proprietatea de a măsura nivelul cîmpului electromagnetic din zonele populate, adică spre exemplu în zonele imediat apropiate în reţelele telefoniei mobile, în staţiile de bază, au fost solicitaţi că nivelul de cîmp electromagnetic este mai mare decît limita admisă şi atunci deasemeni s-au făcut măsurări în zonele locuite (în special a intensităţii cîmpului electromagnetic).

Concluzii

În concluzie, metodologia de determinare a nivelului de radiație emis de telefoanele mobile și echipamentele radio prin intermediul SAR este un proces complex care necesită un echipament specializat și cunoștințe specializate în domeniul fizicii sau ingineriei electromagnetice. Măsurarea SAR este importantă pentru a evalua expunerea la radiații electromagnetice și pentru a se asigura că dispozitivele electronice sunt sigure pentru utilizare. Standardizarea măsurătorilor este crucială pentru a obține rezultate precise și comparabile, iar interpretarea acestor rezultate poate fi realizată numai de specialiști calificați. În general, este important ca utilizatorii să fie conștienți de nivelul SAR al dispozitivelor lor și să ia măsuri pentru a reduce expunerea la radiații electromagnetice, cum ar fi utilizarea căștilor cu fir sau a dispozitivelor cu reducere de radiații.

Referinţe

1. https://ibn.idsi.md/vizualizare_articol/55340
2. http://repository.utm.md/bitstream/handle/5014/3300/Conf_McCS_2017_p181-183.pdf.pdf?sequence=1&isAllowed=y
3. https://www.mobilissimo.ro/stiri-telefoane/cele-mai-bune-si-cele-mai-proaste-10-telefoane-la-capitolul-sar
4. https://asf.alaska.edu/information/sar-information/what-is-sar/
5. https://www.emfexplained.info/?ID=25584

INTELIGENȚA ARTIFICIALĂ ȘI IMPACTUL ASUPRA PROCESULUI EDUCAȚIONAL

Valeria ȚURCAN

Departamentul Telecomunicații și Sisteme Electronice, grupa IMTC-211, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova.

*Autorul corespondent: Valeria Țurcan, valeria.turcan@tse.utm.md

Coordonator științific: Andrei DOROGAN, conf. univ., dr., *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

Rezumat. *În lucrare sunt prezentate noile oportunități și posibilități ale aplicației software bazată pe inteligență artificială ChatGPT și estimate aspectele pozitive, negative și eventualul impact asupra activităților din diverse domenii și a procesului educațional.*

Cuvinte cheie*: inteligență artificială, învățare automatizată, sisteme inteligente, digitalizare.*

Introducere

Ideea despre implementarea inteligenței artificiale preocupă omenirea din cele mai vechi timpuri. Termenul de "inteligență artificială" a apărut în 1956, dar popularitatea reală a tehnologiei AI (Inteligență Artificială) a ajuns abia astăzi, pe fondul creșterii volumului de date, al îmbunătățirii algoritmilor, al optimizării puterii de calcul și stocării datelor.

Domeniile de aplicare a inteligenței artificiale sunt foarte diverse, printre care:

1. Utilizarea AI în domeniul finanțelor. AI analizează acțiunile pe conturile bancare și permite detectarea acțiunilor suspicioase, ceea ce sporește securitatea contribuțiilor financiare.
2. Inteligența artificială este utilizată în cadrul complexului militar-industrial pentru implementarea diverselor simulatoare cu realitate augmentată.
3. Robotica efectuează parțial lucrări periculoase și de asamblare în fabricile de automobile utilizând AI.
4. Utilizarea AI în medicină pentru realizarea tomografiei computerizate prin diagnosticarea tumorilor și altor boli în stadiu incipient.
5. Utilizare în diverse domenii precum vânzări, divertisment, transport, securitate și siguranță, etc.

Caracteristicile, utilizarea și funcționarea ChatGPT

ChatGPT este un chatbot creat de OpenAI, mai exact, a fost creat de Sam Altman în coordonare cu Elon Musk în 2015. Deoarece ChatGPT este recent lansat, sunt multe detalii care încă trebuie lucrate. Inițial, comunicarea cu AI putea fi realizată doar în limba engleză, acum însă, chat-ul recunoaște mai multe limbi, iar răspunsurile pot fi traduse în limba necesară. Dacă cineva cere să creeze un cod în oricare dintre limbajele informatice, atunci ChatGPT îl poate crea cu ușurință. Trebuie remarcat faptul că informațiile oferite de ChatGPT nu sunt 100% corecte [1].

Una dintre cele mai importante caracteristici ale ChatGPT este răspunsul detaliat la interogare. În comparație cu predecesorul său InstructGPT, ChatGPT se concentrează pe reducerea răspunsurilor înșelătoare și dăunătoare. Datele de instruire cuprind informații referitoare la fenomene colectate din internet, pagini de manual și limbaje de programare precum un limbaj de programare python, sisteme de buletin etc.

ChatGPT reprezintă un soft de tip statutar și își amintește interogările anterioare care i-au fost adresate în cadrul aceleiași conversații. Pentru a preveni prezentarea de către program a rezultatelor abuzive, întrebările adresate de utilizatori sunt filtrate printr-un API (Interfață de Programare a

Aplicației) de moderare. Aspectul negativ constă în aceea, că modelul nu este antrenat pe conversații de lungă durată.

Dezvoltatorii au folosit date publice cu acces liber pentru antrenarea soft-ului. Datele salvate în motorul de căutare sunt prezentate utilizatorilor sub forma unei soluții specifice la interogare. Soluția este afișată pe ecran atâta timp cât durează sesiunea sau pînă la următoarea postare de interogare.

Modelul arhitectural ChatGPT

Modelul ChatGPT se bazează pe arhitectura GPT-3.5 fiind alcătuit din trei modele: „code-davinci-002” - modelul de bază pentru sarcinile de completare a codurilor software, „text-davinci-002” - antrenat prin perfecționarea modelului InstructGPT pe o rețea specială cu instrucțiuni complexe și validat cu ajutorul experților, si cea mai recentă versiune „text-davinci-003” - reprezintă o dezvoltare a modelului „text-davinci-002” pe un set de instrucțiuni mai complexe ale utilizatorului. Fiecare dintre cele trei modele este o versiune îmbunătățită a celei precedente, iar cel mai puternic modernizat este modelul „text-davinci-003”, care stă la baza ChatGPT. Faza de instruire a ChatGPT utilizează date de tip text suplimentare și coduri de program colectate la sfârșitul anului 2021.

Modelul de învățare RLHF (Reinforcement Learning with Human Feedback) se bazează pe un model lingvistic pre instruit. O caracteristică distinctivă esențială este încorporarea unui model de recompensă (numit și model de preferințe), calibrat în funcție de părerea experților. Obiectivul principal este de a obține un sistem care ia o secvență de propoziții și returnează o valoare scalară de recompensă, care ar trebui să reflecte numeric judecata experților.

O metodă de succes pentru clasificarea texului este aceea de a cere experților să compare textul generat de două modele lingvistice. Prin compararea de perechi a rezultatelor generate de modele pot fi clasificate modelele și rezultatele. Metodele de clasificare sunt normalizate la valoarea scalară a recompensei de învățare.

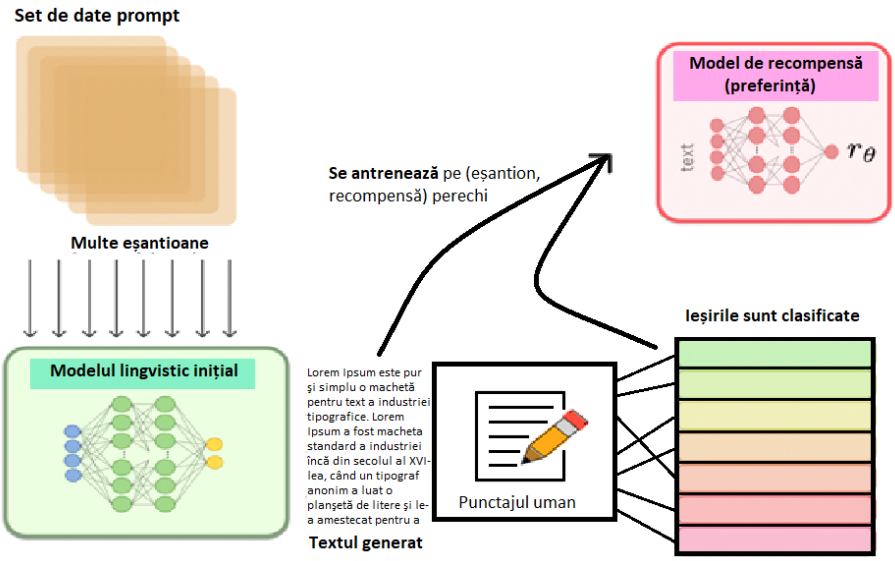


Figura 1. Implicarea evaluării „inter-perechi” ca parte a procesului de RL (Reinforcement Learning) [2]

Posibilitățile oferite de ChatGPT

Aplicația ChatGPT poate genera fragmente de cod coerente pentru sarcini tipice, cu explicații și identifica erori în codul de programare (Fig. 2). Cu excepția scrierii de coduri, ChatGPT poate, de asemenea, să ajute inginerii electronești și dezvoltatorii să corecteze codurile de program și să servească drept un asistent puternic pentru programele open-source.

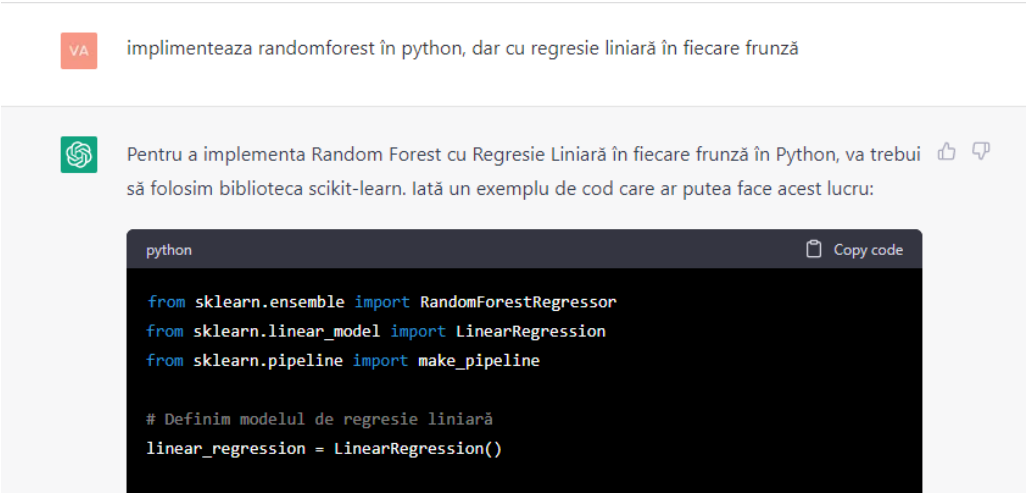


Figura 2. Generarea unui cod de program utilizând aplicația ChatGPT

O opțiune interesantă a ChatGPT este parafrazarea texului, care permite ocolirea sistemelor anti-plagiat și generarea unui conținut textual original de calitate sau un eseu pe diverse teme.

Impactul inteligenței artificiale asupra procesului educațional

Facilitățile AI sunt utilizate în viața de zi cu zi, prin intermediul aplicațiilor care furnizează informații în funcție de preferințe. Din majoritatea domeniilor, sectorul educației beneficiază enorm de pe urma AI. Cu toate acestea, multe școli și profesori au fost reticenți în a include AI în metodele lor de predare. Motivele sunt lipsa de conștientizare a AI și a modului în care aceasta poate ajuta sistemul educațional.

AI este înaintată frecvent ca fiind o soluție pentru diverse probleme de bază ale educației (de exemplu, lipsa profesorilor calificați, performanța slabă a elevilor. Acest fapt relevă necesitatea de a lua în considerare mai multe aspecte: obiectivele utilizării AI în educație, unde și de către cine este utilizată, (persoane, instituții sau industrie), cum și la ce niveluri este operaționalizată (de la un singur elev la clase întregi, rețele de colaborare, etc).

Învățarea cu ajutorul AI presupune utilizarea instrumentelor bazate pe inteligență artificială în predare și învățare, care include:

• utilizarea AI pentru suportul elevilor, studenților, implicând instrumente precum: sisteme de exemple inteligente, sisteme de exemple bazate pe dialog, sisteme de explorare, evaluarea automată a scrisului, aplicații AI pentru studenți cu dizabilități;

• utilizarea inteligenței artificiale destinat pentru sistemele administrative (recrutarea, stabilirea orarului și gestionarea învățământului);

• utilizarea AI pentru suportul profesorilor (corecția și prelucrarea materialelor de studiu [3].

Avantajele inteligenței artificiale

Organizarea informației. Actual, datorită motorului de căutare Google, putem accesa orice informație necesară, fie pentru accesarea resurselor pentru o oarecare temă sau un serviciu de redactare a eseurilor. Platformele AI pot ajuta studenții, oferindu-le exact informația necesară. Acest aspect oferă materiale de studiu relevante, eliminând presupunerile necesare în elaborarea planurilor de studiu individuale. Aplicațiile AI pot colecta date despre abilitățile elevilor cu permisiunea tutorelui lor și îi poate provoca bazându-se pe punctele lor forte și slabe.

Suportul elevilor cu dizabilități. AI joacă un rol crucial în îmbunătățirea vieții persoanelor cu dizabilități. Softurile de recunoaștere a vorbirii, cum ar fi „Nuance”, pot ajuta la transcrierea cuvintelor pentru elevii cu dificultăți de scriere sau cu mobilitate limitată. Aceste soluții îi pot ajuta pe profesori să ofere studenților materiale de studiu mai calitative, conform cerințelor lor unice.

Învățarea imersivă. Sistemul de învățământ actual s-a concentrat mult pe reținerea memoriei elevilor, mai degrabă decât pe înțelegerea esenței materiei studiate. AI ar putea schimba acest lucru. Combinată cu realitatea virtuală și augmentată, AI poate contribui la un mediu de învățare dinamic.

Sisteme inteligente de tutorat. Moderarea inteligentă a informației permite profesorilor și tutorilor să analizeze mai calitativ informațiile, ceea ce va contribui la îmbunătățirea sistemele de notare și eficientizarea procesului didactic. Utilizând aplicațiile AI, profesorii pot afla mai multe despre obiceiurile elevilor și pot crea un program de studiu care să funcționeze mai eficient în funcție de cerințele acestora.

Dezavantajele inteligenței artificiale

Pe cât de eficientă este inteligența artificială, pe atât de ridicată este și costul ei. Nu doar prețul produsului este înalt, ci și costul de întreținere și reparații este la nivel. Pe măsură ce AI necesită mai multe dispozitive digitale, cantitatea de energie necesară pentru a asigura funcționarea lui crește exponențial. La moment, doar școlile bine finanțate pot beneficia de oportunitățile aplicațiilor AI.

În contextul experienței educaționale, aplicațiile AI pot înlocui profesorii în multe aspecte. Procesul didactic este reprezentat de relația dintre profesori și elevi și de modul în care conexiunile personale pot modela comportamentele. Dependența de tehnologii poate avea un aspect negativ asupra eficientizării procesului educațional.

În timp ce industria de software ar putea fi în plină expansiune, AI ar putea înlocui în curând membrii personalului didactic și ar putea contribui la un șomaj considerabil în sectorul educațional. Digitalizarea informațiilor și a procesului educațional ar putea contribui la predispunerea datelor elevilor la acțiuni de hacking cu consecințe de abuz de date cu caracter personal [4].

Concluzii

Progresul informațional își continuă dezvoltarea. Domeniului IT nu poate neglija impactul inteligenței artificiale asupra societății, fiind el negativ sau pozitiv. Este foarte important de identificat și stabilit modul de utilizare a acestor resurse. Oportunitățile pot fi folosite după destinație, oferind un proces educațional confortabil sau din contra, pot agrava situația.

Aplicația ChatGPT oferă diverse posibilități noi de suport și asistență a procesului educațional, a activităților din diverse domenii ale industriei, medicinii, economiei, care pot avea un impact preponderent pozitiv.

Perspectiva inteligenței artificiale este incontestabil de incitantă și întotdeauna există argumentul că avantajele depășesc dezavantajele. Este timpul ca sistemul nostru educațional să aibă nevoie de o revizuire, iar AI ne poate îndruma în direcția cea bună. În același timp, este extrem de important să estimăm cât de mult control acordăm mașinilor și inteligenței lor.

Referinţe

1. What is ChatGPT? History, Features, Uses, Benefits, Drawbacks 2023 [online] [accesat 08.03.2023]. Disponibil: https://medium.com/illumination/what-is-chatgpt-history-features-uses-benefits-drawbacks-2023-ec970bc38f84.
2. Illustrating Reinforcement Learning from Human Feedback (RLHF) [online] [accesat 27.03.2023] Disponibil: https://huggingface.co/blog/rlhf
3. W. Holmes, J. Persson, I.A. Chounta, B. Wasson and V. Dimitrova *Artificial intelligence and education* [online]. p. 19-24 [online] [accesat 08.03.2023]. Disponibil https://rm.coe.int/artificial-intelligence-and-education-a-critical-view-through-the-lens/1680a886bd.
4. Using Artificial Intelligence in Education: Pros and Cons [online] [accesat 08.03.2023] https://theknowledgereview.com/using-artificial-intelligence-in-education-pros-and-cons/.

TEHNOLOGII DE ANULARE ACTIVE A ZGOMOTULUI

Victor ZMUNCILĂ

Departamentul Telecomunicații și Sisteme Electronice, grupa IMTC-201, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova.

*Autorul corespondent: Victor Zmuncilă, victor.zmuncila@tse.utm.md

Coordonator științific: Roman GRIȚCO, *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, Chișinău, R. Moldova*

Rezumat. *Această lucrare prezintă tehnologiile de anulare activă a zgomotului folosite în căstile de redare, sau în cele de izolare a sunetului și anume este explicat modul de functionare a tehnologiilor utilizate în ziua de azi precum „feedforward”care utilizeaza un microfon intern pentru preluarea sunetului nedorit, „feedback ”care folosește un microfon la exteriorul căștii și o combinație a celor două denumită „ hibrid”care folosește deja microfon intern și extern pentru a analiza cu mai multă prezie.*

Cuvinte cheie: *ANC, PNC, Feedforward, Feedback, Hybrid*

Introducere

Există două tipuri de anulare a zgomotului în căștile din ziua de azi și anume pasiv și activ, unde ambele pot fi utilizate separat sau împreună. Anularea pasivă a zgomotului (cunoscută și sub numele de izolare a zgomotului PNC) filtrează zgomotul ambiental cu componente căștilor, cum ar fi vârfurile de cauciuc sau prin intermediul designului fizic (dimensiune, diametru, formă, material) care este eficientă la zgomot de înaltă frecvență (lățime de bandă audio peste 1 kHz), dar nu la zgomot scăzut la care este frecvența zgomotului ambiental. Anularea activă a zgomotului, pe de altă parte, este mai eficientă în a face față zgomotului constant și repetitiv de joasă frecvență, cum ar fi zgomotul format de automobile sau zgomotul unui frigider, etc . În această lucrare, ne vom concentra în principal pe tehnologiile de anulare activă a zgomotului folosit în căștile contemporane.

Modul de funcționare a ANC

Anularea activă a zgomotului (ANC, **Active Noise Cancellation**) a fost dezvoltată inițial pentru a reduce zgomotul static al motoarelor în timpul zborurilor. În zilele noastre, algoritmi mai puternici și microfoane mai bune au împins ANC dincolo de lumea călătoriilor, permițând utilizatorilor să elimine zgomotul de fundal și vocile la birou și acasă. Principiul anulării zgomotului nedoriit este că microfonul colectează zgomotul extern din mediu, iar apoi sistemul îl transformă într-o undă sonoră inversă și îl adaugă la capătul difuzorului, reducând astfel zgomotul senzorial.

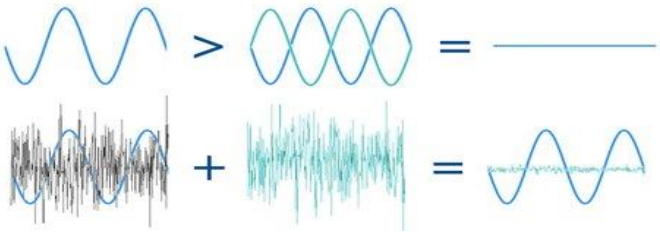


Figura 1. Principiul anularii active a zgomotului (ANC) [2]

Feedforward ANC

Tehnologia Feedforward ANC folosește un microfon extern, care se află de obicei pe carcasa exterioară a căștilor ce se observă în fig.1, sau uneori, în cazul modelelor cu fir, microfonul este poziționat pe cablu. La așa căști zgomotul nedorit este captat extern, procesat în cipul ANC unde se formează semnalul antizgomot și în final, este transmis împreună cu cu semnalul inițial la difuzor unde se emite sunetul corectat. Această este cea mai simplă formă de anulare activă a zgomotului.

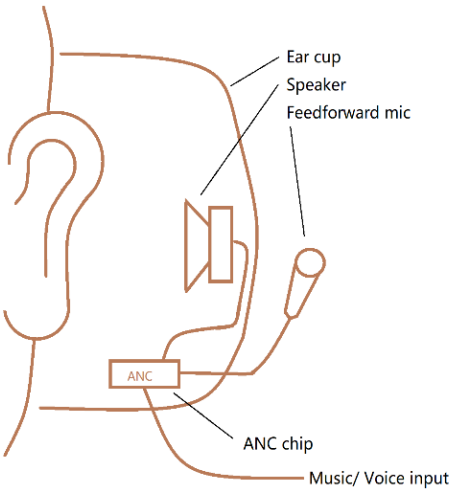


Figura 2. Structuara căștii la tehnolohia Feedforward [4]

Tehnologia feedforward are avantajul că preia zgomotul devreme, astfel având timp pentru a genera antizgomot, dar are și dezavantaj că nu are nicio modalitate de a se auto-corectă, deoarece nu aude niciodată antizgomotul pe care îl produce. Feedforward ANC funcționează eficient de la aproximativ 80 Hz la 2 kHz din lățimea de bandă audio.

Feedback ANC

Anularea activă a zgomotului cu feedback plasează microfonul în fața difuzorului, ceea ce schimbă modul în care este procesat sunetul nedorit. Dacă microfonul intern preia orice zgomot care nu iese din speaker, cipul ANC îl identifică că zgomot nedorit și creează o undă opusă, care este trimisă înapoi prin difuzor la urechi. Această tehnică necesită mai multă putere de procesare, deoarece cipul se adaptează constant la schimbările de sunet.

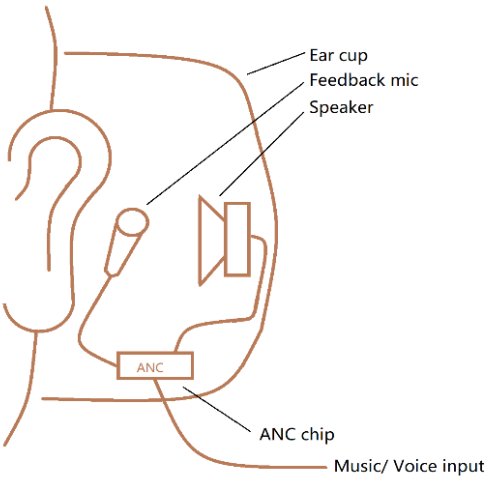


Figura 3.Structuara căștii la tehnolohia Feedback [4]

Beneficiul major este că zgomotul captat de microfon reflectă cu mai multă acuratețe zgomotul pe care îl aude purtătorul , feedback-ul ANC se poate adapta mai bine la orice variații de intrare și poate corectă semnalul, indiferent de poziționarea și potrivirea exactă a căștilor. Există un risc mic că sistemul să-și preia propriul semnal anti-zgomot și să crească nivelul de amplificare în încercarea de a-l anula. Un alt beneficiu este că există mai puțîn timp de procesare, deoarece funcționează pe sunet deja foarte aproape de ureche. Tehnologia feedback funcționează eficient de la aproximativ 50 Hz până la 800 Hz din lățimea de bandă audio și nu poate gestiona frecvențele mai mari că Feedforward ANC.

Hybrid ANC

Tehnologia Hybrid ANC este o combinație dintre Feedforward și Feedback, astfel are și microfoan intern și extern. În consecință, hibrid ANC obține cea mai bună acoperire a frecvenței de atenuare a zgomotului și cea mai mică șansă de probleme de feedback. Dezavantajul este că hibridul ANC este mai scump. Nu numai că există două microfoane, dar aceste microfoane trebuie să fie de înaltă calitate pentru a evita introducerea de zgomot suplimentar. În plus, căștile necesită un hardware de procesare dedicat mai puternic pentru a gestiona procesarea complexă a datelor, ceea ce va crește costurile de dezvoltare a software-ului ANC sau a algoritmului. ANC hibrid se poate adapta și corectă erorile și nu este la fel de sensibil la modul în care persoană poartă setul cu cască.

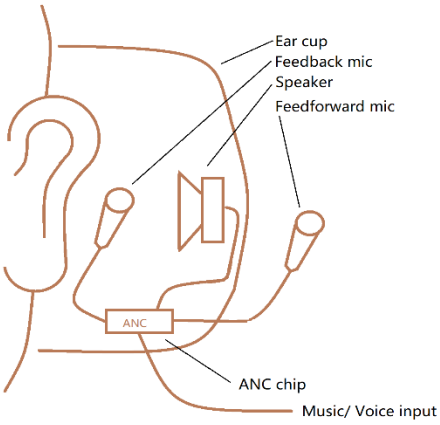


Figura 4. Structuara căștii la tehnolohia Hibrid [4]

Concluzii

Atenuare activă a zgomotului pot fi folosită în căștile din majoritatea domeniilor de activitate unde este careva zgomot. Astfel fiecare tehnologie este eficientă în dependent de necesități, căci feedforward este mai bun la anularea zgomotelor de frecvență mai înaltă, dar nu are mijloace de auto-corectare, și în cele mai rele cazuri, poate chiar amplifică anumite sunete, feedback permite microfonului să audă anti-zgomotul pe care îl produce, dar deoarece microfonul este mai aproape de ureche, are mai puțin timp să reacționeze și, prin urmare, nu este la fel de eficient în anularea zgomotelor noastre de frecvență mai înaltă, iar hybrid ANC folosește ambele metode menționate anterior, cu microfoane în interiorul și în exteriorul căștii, acest lucru îi permite să culeagă toate beneficiile fără a suferi niciunul dintre dezavantajele. De asemenea, costă la fel de mult că și celelalte două subtipuri combinate.

Referințe:

1 *Jabra GN*, [online], [accesat 10.03.2023] https://www.jabra.com/supportpages/jabra-elite-7-pro/100-99172700-98/faq/what-is-passive-noise-cancellation
2 *Design & Reuse*, [online], [accesat 10.03.2023] https://www.design-reuse.com/articles/49714/audio-codec-for-tws-optimizing-latency-key-factor.html
3 *Jabra GN*, [online], [accesat 10.03.2023] https://www.jabra.com/blog/anc-headsets-arent-all-the-same-three-types-of-anc/
4 *Ole Wolff Elektronik A/S*, [online], [accesat 10.03.2023] https://www.owolff.com/acoustics/design-support/active-noise-cancellation
5 *Audio-Technica U.S, Inc.* , [online], [accesat 10.03.2023] https://www.audio-technica.com/en-us/support/audio-solutions-question-of-the-week-how-do-different-active-noise-cancelling-headphones-work/
6 *EarbudsZone*, [online], [accesat 10.03.2023] https://www.earbudszone.com/active-vs-passive-noise-cancelling/

SECŢIA ENERGETICĂ ȘI INGINERIE ELECTRICĂ

SECTION OF ENERGETICS AND ELECTRICAL ENGINEERING

AUTOMATIZAREA PROCESULUI TEHNOLOGIC DE SORTARE A UNUI SET DE OBIECTE DUPĂ CULORI

Florin TESLARI, Corneliu RĂILEANU,
Valentin BÂNZARU, Vladislav ENACHI

Inginerie electrică, ISEM-201, FEIE, UTM, Chișinău, Republica Moldova

*Autorul corespondent: Florin Teslari, florin.teslari@ie.utm.md

Îndrumător/coordonator științific: Vadim CAZAC, dr. conf., FEIE, UTM, Moldova

Rezumat. *Acest articol descrie un dispozitiv cu bandă rulantă care utilizează un senzor de culoare RGB pentru a diferenția între 4 culori și a direcționa obiectele către locul prestabilit Fig. 3, folosind microcontrolerul Arduino Mega, 3 servo motoare și un motor de curent continuu. Acesta poate fi folosit în automatizarea domeniilor precum, procesarea produselor alimentare sau sortarea obiectelor în fabrici. Acest articol va explora sistemul de dozare a obiectelor, sistemul de direcționare, sistemul de control, hardware, software și prezintă o metodă simplă și eficientă de construire a dispozitivului și funcționarea sa.*

Cuvinte cheie*: microcontroler, bandă rulantă, senzor RGB, arduino.*

Introducere

În ultimele decenii, tehnologia a evoluat considerabil, iar acest lucru a dus la dezvoltarea unor dispozitive, care ne pot ajuta să automatizăm anumite procese tehnologice. Un exemplu în acest sens este dispozitivul cu bandă rulantă, care poate repartiza obiecte în funcție de culoarea acestora. Această tehnologie este posibilă datorită utilizării unui senzor de culoare RGB, care poate diferenția 4 culori.

Instalația este controlată de un microcontroler, care primește semnale de la senzorul de culoare și analizează datele pentru a identifica culoarea obiectelor. În funcție de aceste date, microcontrolerul emite o serie de comenzi care dirijează obiectele spre locul prestabilit Fig. 3.

Procesul de construcție al acestui dispozitiv a implicat o serie de etape tehnologice complexe, precum proiectarea, asamblarea și programarea. De asemenea, a fost necesară utilizarea de materiale de calitate și instrumente speciale, precum și o bună cunoaștere a programării și proiectare 3D. În final, acest dispozitiv reprezintă un exemplu concret al modului în care tehnologia poate fi folosită pentru a automatiza procese și pentru a crește eficiența și precizia acestora.

Sistemul de dozare

Sistemul de dozare [1] din Fig. 1 este proiectat pentru a temporiza procesul de încărcare a benzii rulante și a detecta culoarea obiectelor. Senzorul de culoare RGB poate detecta culorile roșu, galben, albastru și roz, precum și combinațiile acestora pentru a identifica alte culori. După ce senzorul a detectat culoarea obiectului care se deplasează pe bandă, datele sunt transmise către microcontrolerul Arduino Mega, care apoi determină unde trebuie direcționat obiectul. În funcție de culorile detectate, microcontrolerul direcționează obiectele către locul prestabilit.

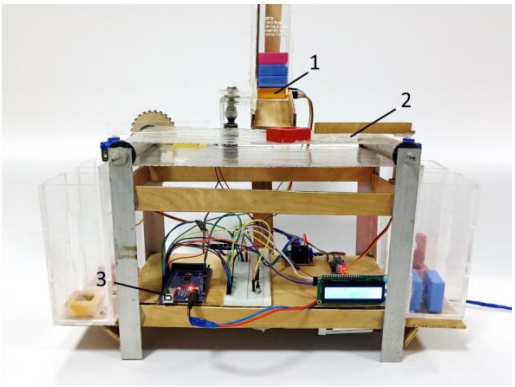


Figura 1. Instalație de sortare cu bandă rulantă

Figura 2. Obiectele pentru sortare

Sistemul de direcționare

Sistemul de direcționare [2] din Fig. 1 folosește o bandă rulantă pentru a transporta obiectele spre locul lor prestabilit Fig. 3. Pentru a asigura că fiecare obiect ajunge la locul său, a fost implementat un sistem de poziționare care se bazează pe feedback-ul senzorului de culoare. Acest sistem este compus din 3 servo-motoare si un motor de curent continuu. Acestea sunt poziționate astfel încât să poată muta obiectul în direcția dorită, în funcție de culoarea detectată. Servo-motoarele controlează mișcarea brațelor care îndreaptă obiectele către locul prestabilit. Motorul de curent continuu controlează viteza benzii rulante și asigura că obiectele sunt transportate cu precizie prin sistem.

Fiecare servo-motor are un unghi de rotație maxim și minim, care poate fi programat în funcție de necesități. Microcontrolerul Arduino Mega poate controla direcția și unghiul de rotație al fiecărui servo-motor, în funcție de culoarea detectată și de locul unde trebuie direcționat obiectul.

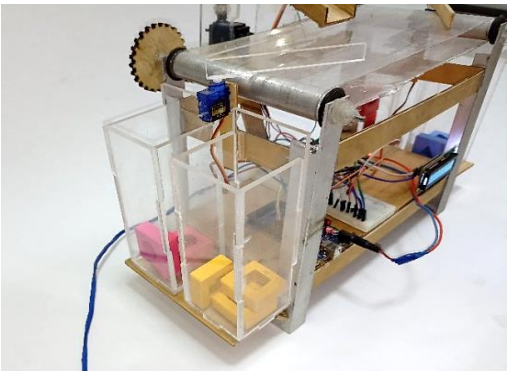


Figura 3. Coșurile pentru obiectele sortate

Sistemul de control software/hardware

Sistemul de control [3] din Fig. 1 este bazat pe un microcontroler Arduino Mega [3] din Fig. 4. Microcontrolerul preia informații de la senzorul RGB și controlează servo-motoarele și driver-ul [5] pentru motorul de curent continuu [7] pentru a asigura un flux continuu și precis de obiecte către locul prestabilit. Dispozitivul cu bandă rulantă este construit folosind componente hardware precum benzi rulante, servo-motoare [6, 4] din Fig. 4 și senzori de culoare [2] din
Fig. 4. Software-ul personalizat este dezvoltat
în limbajul de programare Arduino și controlează fluxul de date între componentele hardware.

Microcontrolerul Arduino Mega este utilizat pentru a controla întregul dispozitiv și afișarea informațiilor despre culoarea detectată pe display [1]. Acesta este programat cu un cod care permite identificarea culorilor, dirijarea obiectelor către locul lor prestabilit și controlul poziției benzii rulante.

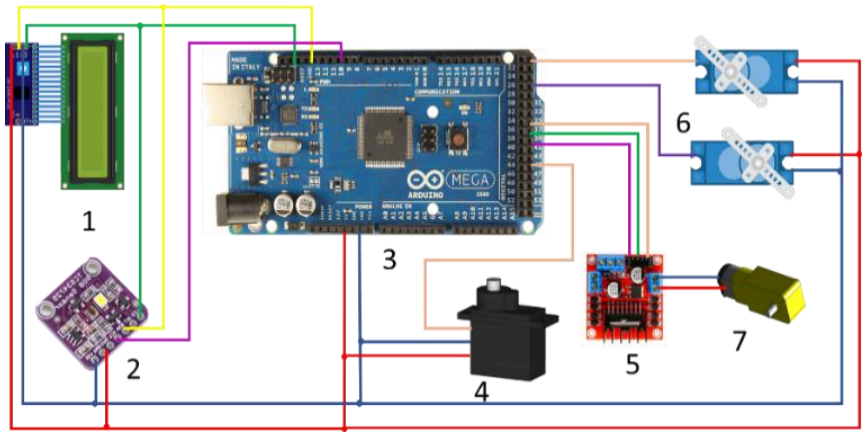


Figura 4. Schema electrică a dispozitivului

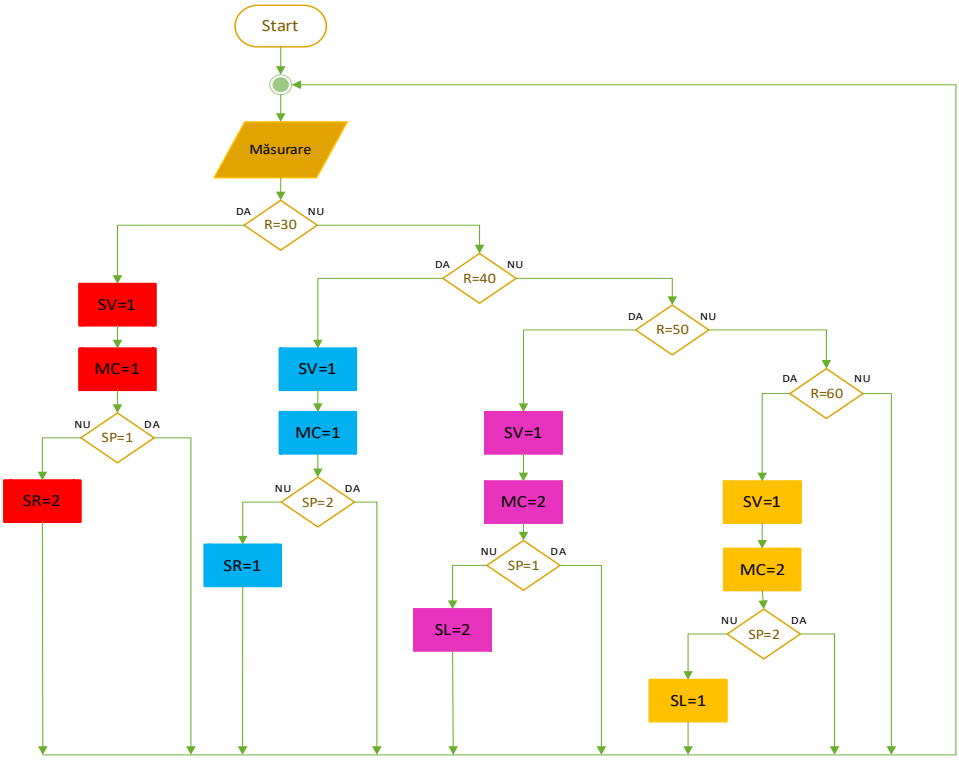


Figura 5. Schema logică după care funcționează instalația

Perspective de utilizare a instalației de sortare după culoare

Dispozitivul cu bandă rulantă poate fi utilizat în diferite domenii, cum ar fi producția de alimente, fabricarea de componente electronice sau în industria farmaceutică.
În producția de alimente, acesta poate fi utilizat pentru sortarea fructelor și legumelor în funcție de mărime și culoare.

Dispozitivele cu bandă rulantă sunt utilizate într-o varietate de aplicații, de la transportul de marfă și bagaje în aeroporturi sau depozite, la aparatele de casă din supermarketuri.

În industria alimentară, dispozitivele cu bandă rulantă sunt utilizate pentru transportul produselor alimentare, cum ar fi produsele de patiserie, carnea și legumele. Acest lucru ajută la optimizarea procesului de producție și la minimizarea timpului necesar pentru transportarea produselor.

Concluzii

Construirea unui dispozitiv cu bandă rulantă care poate diferenția culorile și dirija obiectele spre locul prestabilit este o soluție eficientă pentru sortarea obiectelor în diverse domenii.

1. Utilizarea senzorului de culoare RGB, microcontrolerului Arduino Mega și a motoarelor servo și de curent continuu poate fi personalizată și adaptată în funcție de nevoile utilizatorului.
2. Instalațiile cu bandă rulantă sunt universale și pot fi utilizate într-o varietate de aplicații, de la producția industrială la depozitări.
3. Utilizarea instalațiilor automate de sortare contribuie la eficientizarea proceselor tehnologice din diferite domenii cum ar fi agricultură sortarea deșeuri etc.
4. Pentru instalații industriale utilizarea echipamentelor care au fost prezentate în această lucrare necesită a fi înlocuite cu niște echipamente analogice dar de uz industrial.

Mulţumiri. Aducem sincere mulțumiri domnului dr. conf. univ.Vadim Cazac pentru ajutorul, sprijinul și ghidajul oferit în procesul realizării acestui articol. Prin sfaturile sale și vasta experiență în domeniul automatizării proceselor tehnologice.

Referinţe

1. TODOS P., GOLOVANOV C., *Senzori şi traductoare*, Editura: Tehnică, Chișinău, 2017.
2. TODOS P., AMBROS T., ARION V.,. GUȚU A,. SOBOR I, UNGUREANU D., *Surse regenerabile de energie*, Editura: Tehnica-info Chișinău, 1999.
3. AMBROS T., *Maşini electrice*, Vol.1, Chișinău, Universitas, 1992.
4. NUCĂ I., ȚURCAN A., *Implementarea procesului de tasare reglabil automat în prepararea automată a cafelei*, Conferința internațională ICATE, 2016.
5. CAZAC V., BURDUNIUC M., NUCĂ I., AMBROS T., MANGOS O,, *Magnetic field analysis in asynchronous motors with six-phase windings*, Annals of the Faculty of Engineering Hunedoara, 20 (3), 55-60, 2022.
6. Arduino Compounds, [online]. [accesat 01.03.2023]. Disponibil: https://docs.arduino.cc/library-examples/
7. Guide for TCS230/TCS3200 Color Sensor with Arduino, [online]. [accesat 02.03.2023]. Disponibil: https://randomnerdtutorials.com/arduino-color-sensor-tcs230-tcs3200/
8. Design the gears your project needs with our free gear generator, [online]. [accesat 05.03.2023]. Disponibil: https://evolventdesign.com/pages/spur-gear-generator

PROVOCĂRI ȘI OPORTUNITĂȚI DE DEZVOLTARE SUSTENABILĂ A PV-URILOR PE ACOPERIȘUL CLĂDIRILOR

Ion CARAI

FEIE, EE-191,Electroenergetica, UTM, Chișinău, Moldova

Coordonator științific: Cristina EFREMOV, dr., lect. univ. FEIE, UTM, Moldova

Rezumat. *În prezenta lucrare este abordată problema panourilor solare fotovoltaice (PV-uri) care au devenit o soluție importantă pentru producerea energiei electrice curată și sustenabilă. În ultimii ani, tot mai multe din clădirile comerciale și rezidențiale au fost echipate cu sisteme PV montate pe acoperișurile lor. Această creștere a instalării de PV-uri a venit la pachet atât cu o serie de provocări, cât și cu oportunități de dezvoltare sustenabilă. Printre provocările majore se numără costul inițial ridicat și durata de amortizare, care poate fi destul de lungă, în funcție de mărimea sistemului și costul energiei electrice în regiunea respectivă. De asemenea, trebuie să fie luat în considerare și designul clădirii pentru a maximiza eficiența PV-urilor și pentru a preveni orice daune la acoperiș în timpul instalării cât și al exploatării. Creșterea utilizării PV-urilor pe acoperișurile clădirilor poate contribui semnificativ la dezvoltarea sustenabilă a energiei electrice și la reducerea impactului asupra mediului.*

Cuvinte cheie*:*
energie din surse regenerabile, dezvoltare sustenabilă, energie curată.

Introducere

Realimentarea Europei cu energie solară pentru a pune capăt dependenței sale de combustibilii fosili din Rusia [1] va contribui, în prima fază, la reducerea dependenței în toate sectoarele economiei. Accelerarea planului REPowerEU va promova mai mult decât dublu față de 2020 energie solară fotovoltaică până în 2025 (320 GW) și până la aproape 600 GW până în 2030. Aceste valori privind capacitatea de producere de energie electrică se referă la curentul alternativ (CA).

În contextul schimbărilor climatice, trecerea la surse de energie curate și sustenabile devine din ce în ce mai importantă. Printre aceste surse se numără și energia solară, care poate fi convertită în energie electrică prin intermediul panourilor solare fotovoltaice (PV-uri). Energia solară va fi elementul central al acestui efort. Cu fiecare panou instalat, energia infinită a soarelui va reduce dependența de combustibilii fosili, de la încălzirea locuințelor, până la procesele industriale.

Utilizarea PV-urilor pe acoperișurile clădirilor poate fi o soluție eficientă pentru generarea de energie electrică curată și sustenabilă [2]. Această creștere a instalării de PV-uri a venit la pachet atât cu provocări, cât și cu oportunități de dezvoltare sustenabilă.

În ciuda acestor provocări, există o serie de oportunități pentru dezvoltarea sustenabilă prin utilizarea PV-urilor pe acoperișurile clădirilor. În primul rând, PV-urile pot reduce dependența de combustibilii fosili și contribuie la reducerea emisiilor de gaze cu efect de seră. De asemenea, PV-urile pot aduce economii la facturile de energie electrică și pot oferi o sursă de venit pentru proprietarii de clădiri, cu astfel de instalații, prin vânzarea excedentului de energie produsă în rețeaua națională.

O altă oportunitate este creșterea accesibilității la energie electrică în zonele izolate sau cu rețele electrice slabe. Instalarea PV-urilor pe acoperișurile clădirilor poate oferi o sursă de energie curată și sustenabilă pentru aceste comunități.

1. Inițiativa de promovare a acoperișurilor solare în Uniunea Europeană

Uniunea Europeană, prin consensul statelor membre, s-a angajat să profite, cât mai repede posibil, unindu-și forțele pentru a concretiza beneficiile multiple pe care le are pentru consumatori imensul potențial neexploatat al acoperișurilor solare.

Potrivit unor estimări realizate la nivel european, energia fotovoltaică, produsă pe acoperișuri, ar putea asigura aproape 25 % din necesarul de energie electrică al acesteia, ceea ce reprezintă mai mult decât ponderea actuală a gazelor naturale. Acest avantaj ar permite protejarea consumatorilor în raport cu prețurile ridicate la energie, contribuind la acceptarea de către public a energiei din surse regenerabile. Inițiativa europeană privind acoperișurile solare anunțată în Comunicarea REPowerEU a Comisiei presupune acțiuni rapide de implementare. Astfel, Uniunea Europeană [1] s-a angajat la:

- majorarea până la 45 % a obiectivului pentru 2030 privind ponderea energiei din surse regenerabile;
- adoptarea dispozițiilor pentru a se asigura că toate clădirile noi sunt „proiectate pentru producerea de energie solară";
- executarea obligatorie de instalații pe acoperișuri, pentru producerea de energie solară, la toate clădirile publice și comerciale noi (până în 2026, 2027 în dependență de suprafață) și la toate clădirile rezidențiale noi până în 2029;
- asigurarea că legislația UE este pusă în aplicare pe deplin în toate statele membre, permițând consumatorilor din clădirile cu mai multe apartamente să își exercite efectiv dreptul la autoconsum colectiv, fără costuri nejustificate (conform Directivei (UE) 2018/2001 a Parlamentului European și a Consiliului din 11 decembrie 2018 privind promovarea utilizării energiei din surse regenerabile și Directiva (UE) 2019/944 a Parlamentului European și a Consiliului din 5 iunie 2019 privind normele comune pentru piața internă de energie electrică conțin dispoziții privind autoconsumul colectiv).

Utilizarea fondurilor UE disponibile pentru redresare și reziliență în planul REPowerEU va facilita aplicarea cu prioritate a măsurilor din cadrul inițiativei de promovare a acoperișurilor solare.

2. Dezvoltarea panourilor fotovoltaice pe acoperiș în Republica Moldova

Tranziția energetică declanșată atât la nivel mondial și european, cât și la nivelul Republicii Moldova [3] poate fi ajustată și prin valorificarea potențialului vast și insuficient utilizat al acoperișurilor în ceea ce privește producerea de energie solară, pentru ca energia să devină mai curată, mai sigură și mai accesibilă ca preț. Pentru a valorifica acest potențial, este stringent necesar de a acționa imediat. Dezvoltarea panourilor fotovoltaice pe acoperiș, în Republica Moldova, a cunoscut o creștere semnificativă în ultimii ani, deoarece acestea sunt o soluție eficientă și ecologică pentru producerea de energie electrică. Potrivit datelor Agenției Naționale pentru Reglementare în Energie din Republica Moldova, capacitatea totală instalată de energie solară fotovoltaică în țară a crescut [4], iar o mare parte din această capacitate este instalată pe acoperișurile clădirilor.

Există câțiva factori care au contribuit la creșterea utilizării panourilor solare pe acoperiș în Republica Moldova. Primul factor este creșterea costurilor energiei electrice, ceea ce face ca energia solară să devină o opțiune mai accesibilă pentru consumatorii casnici și industriali. De asemenea, statul a implementat un program de sprijin pentru instalarea de sisteme solare, ceea ce a stimulat creșterea investițiilor în această tehnologie [5 -8].

În plus, guvernul Republicii Moldova a implementat politici și regulamente pentru a încuraja dezvoltarea energiei solare, cum ar fi legislația în vigoare, care stabilește obiective pentru producția de energie din surse regenerabile și mecanisme de sprijin pentru dezvoltarea energiei din surse regenerabile. Aceste politici au creat un mediu favorabil pentru investițiile în energia solară și au încurajat proprietarii de clădiri să instaleze panouri solare pe acoperișurile lor.

În plus, există un potențial mare pentru dezvoltarea energiei solare în Republica Moldova [4], având în vedere că țara beneficiază de un număr mare de ore însorite pe an și are o suprafață relativ mică. Aceasta face ca instalarea de panouri solare pe acoperișurile clădirilor să fie o opțiune atractivă pentru a produce energie electrică fără a ocupa terenuri sau a afecta mediul înconjurător.

3. Parametrii tehnici ai panourilor fotovoltaice și a invertoarelor

Panourile fotovoltaice (sau modulele fotovoltaice) sunt dispozitive care transformă energia solară în energie electrică. Principalii parametri tehnici ai panourilor fotovoltaice includ:

- Puterea nominală (P_{max}) - este puterea electrică maximă pe care un panou o poate produce în condiții standard de testare (STC). Acest parametru este exprimat în watt (W).
- Tensiunea la puterea maximă (V_{mpp}) - este tensiunea electrică la care panoul fotovoltaic produce puterea nominală. Acest parametru este exprimat în volți (V).
- Curentul la puterea maximă (I_{mpp}) - este curentul electric la care panoul fotovoltaic produce puterea nominală. Acest parametru este exprimat în amperi (A).
- Tensiunea circuitului deschis (V_{oc}) - este tensiunea electrică pe care panoul o produce atunci când nu este conectat la nici o încărcare. Acest parametru este exprimat în volți (V).
- Curentul de scurtcircuit (I_{sc}) - este curentul electric maxim pe care panoul îl poate furniza atunci când este conectat la o încărcare de scurtcircuit. Acest parametru este exprimat în amperi (A).

Este important de menționat ca PV-urile își pierd din eficiență pe parcursul anilor de funcționare. În Fig. 1 este prezentată garanția de putere a panoului de tip Longi LR5-72HBD cu o eficiență mai ridicată pe parcursul a 30 de ani.

Invertoarele sunt dispozitive care transformă curentul continuu (DC) produs de panourile fotovoltaice în curent alternativ (AC) utilizabil în rețelele de distribuție a energiei electrice. Principalii parametri tehnici ai invertoarelor includ:

- Puterea nominală (P_{nom}) - este puterea electrică maximă pe care un invertor o poate furniza la ieșirea sa. Acest parametru este exprimat în watt (W).
- Tensiunea maximă de intrare (V_{max}) - este tensiunea maximă de intrare la care invertoarele pot funcționa. Acest parametru este exprimat în volt (V).
- Curentul maxim de intrare (I_{max}) - este curentul maxim de intrare la care invertoarele pot funcționa. Acest parametru este exprimat amper (A).
- Eficiența - este raportul dintre puterea electrică furnizată la ieșirea invertoarelor și puterea electrică absorbită de la intrare. Acest parametru este exprimat sub formă de procent (%).

Pentru eficiența invertorului trebuie de luat în considerare și parametrii curbei de eficiență care sunt, pentru fiecare tip de invertor, diferiți. În Fig.2 este prezentată curba de eficiență pentru invertorul de tip Huawei SUN2000-20KTL-M2 care va avea o eficiență maximă la 600V.

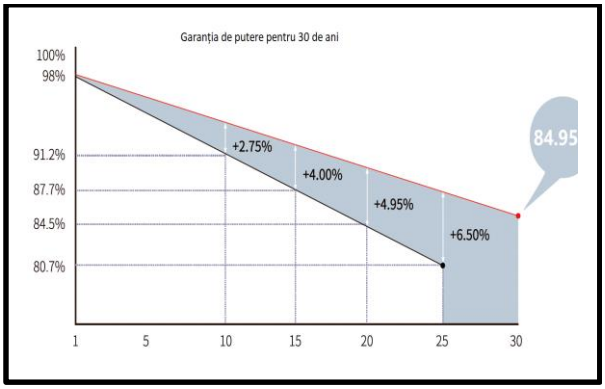


Figura 1. Eficiența Panoului fotovoltaic de tip Longi LR5-75 HBD pe parcursul a 30 ani

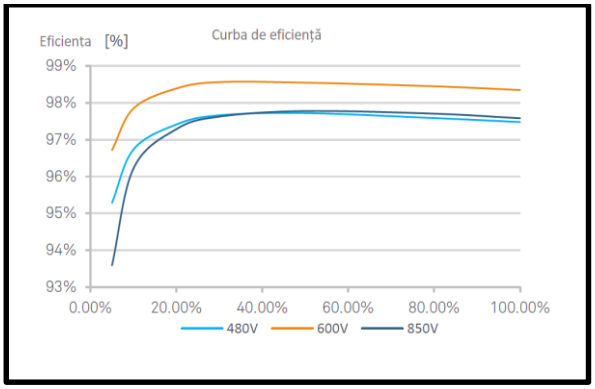


Figura 2. Curba de eficiență a invertorului de tip Huawei SUN2000-20KTL-M2

4. Modalități de amplasare a panourilor fotovoltaice și unghiul de înclinare

Panourile fotovoltaice pot fi montate pe acoperișul clădirilor la diferite unghiuri de înclinare, iar acest lucru poate afecta eficiența și producția lor de energie electrică. Iată cum influențează unghiul de înclinare amplasarea panourilor fotovoltaice:

- Unghiul optim de înclinare: În general, unghiul optim de înclinare al panourilor fotovoltaice este egal cu latitudinea locației. De exemplu, dacă locația dvs. este la 30 grade nord, atunci unghiul optim de înclinare este de 30 grade. Aceasta maximizează producția de energie electrică pe parcursul întregului an.

- Înclinare mai mică decât optimul: Dacă panourile fotovoltaice sunt montate la un unghi mai mic decât cel optim, vor produce mai puțină energie electrică. Acest lucru se datorează faptului că mai multă lumină a soarelui va fi reflectată în atmosferă, mai degrabă, decât să ajungă pe panouri.
- Înclinare mai mare decât optimul: Dacă panourile fotovoltaice sunt montate la un unghi mai mare decât cel optim, vor produce mai puțină energie electrică decât la un unghi optim. Acest lucru se datorează faptului că mai multă lumină a soarelui va fi absorbită de acoperiș, și, ca atare, mai puțină va ajunge la panouri.
- Înclinare orizontală: În general, montarea panourilor fotovoltaice pe un acoperiș orizontal nu este recomandată, deoarece acestea vor produce mai puțină energie electrică decât la un unghi optim. Acest lucru se datorează faptului că panourile nu vor beneficia de lumina directă a soarelui pe parcursul întregii zile.

Să luăm un exemplu pentru amplasarea panourilor fotovoltaice pe acoperișul unei case de cultură din mun. Ceadâr-Lunga cu puterea instalată de 20 kW (studiu de caz real). În primul caz se va amplasa PV-urile la înclinare de 30 grade și azimut 40 grade Fig.3. În al doilea caz se va amplasa PV-urile la înclinare de 30 grade și azimut -50 grade Fig.4.

Tabelul 1

Rezultatul obținut în urma calculelor realizate cu softul „PVsyst”

	Varianta 1 **înclinare de 30° și azimut 40°**	**Varianta 2** **înclinare de 30° și azimut -50°**
Energia efectiva la ieșire (anual) kWh	27219	26565
Energia injectată în rețea (anual) kWh	26637	25990
Performanță raport %	0.878	0.869

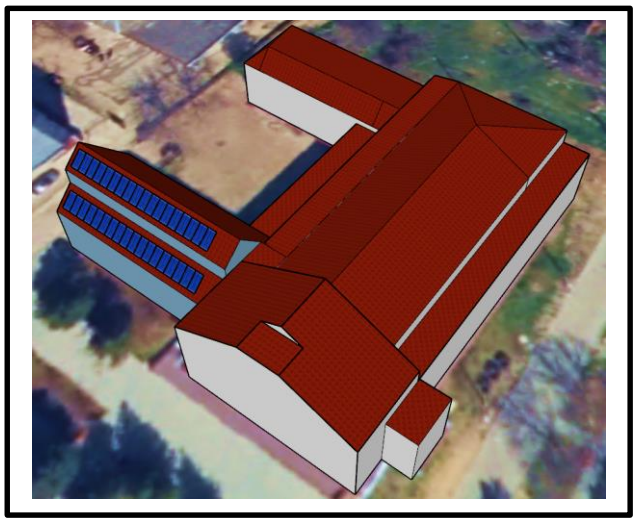


Figura 3. Imagine cu amplasarea PV-urilor la înclinare de 30° și azimut 40°

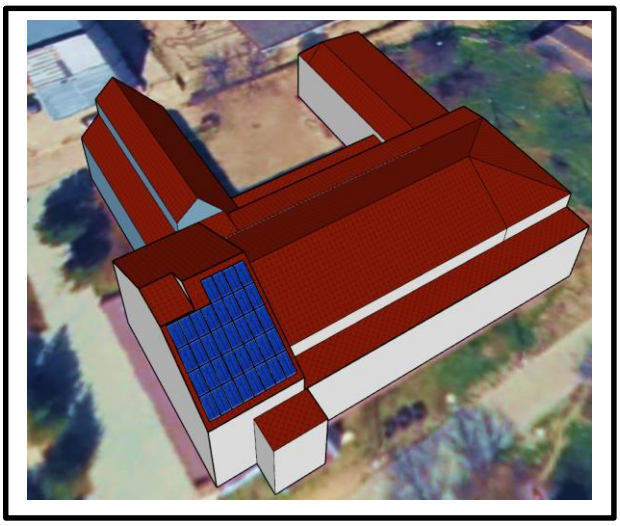


Figura 4. Imagine cu amplasarea PV-urilor la înclinare de 30° și azimut -50°

Concluzii

În concluzie, există numeroase provocări și oportunități pentru dezvoltarea sustenabilă a panourilor fotovoltaice pe acoperișul clădirilor. Pe de o parte, există provocări legate de costuri, tehnologie, reglementări și infrastructură, care trebuie abordate pentru a permite o creștere semnificativă a utilizării energiei solare în construcții. Pe de altă parte, există oportunități importante de a spori eficiența energetică a clădirilor și de a reduce dependența de sursele de energie nesustenabile prin intermediul panourilor fotovoltaice montate pe acoperiș.

Pentru a răspunde tuturor provocărilor legate de valorificarea potențialului de energie solară pe acoperișurile clădirilor trebuie să se încurajeze, prin legislație, sprijinul față de sursele de energie fotovoltaică integrate în clădiri, atât pentru clădirile noi, cât și pentru clădirile renovate. Totodată, cu

aplicarea legislației trebuie să se permită utilizatorilor din clădirile cu mai multe apartamente să își exercite efectiv dreptul la autoconsum colectiv, fără costuri nejustificate.

Cu toate acestea, pentru a profita de aceste oportunități, este necesară o abordare cuprinzătoare care să abordeze aspecte tehnice, economice și sociale ale implementării panourilor fotovoltaice. Acest lucru poate include încurajarea inovației și a dezvoltării tehnologice, îmbunătățirea infrastructurii și a reglementărilor, și educarea și implicarea comunității.

În final, desprindem ca o concluzie directă că panourile fotovoltaice montate pe acoperișul clădirilor pot juca un rol important în dezvoltarea sustenabilă a societății noastre, iar abordarea provocărilor și a oportunităților acestui domeniu poate aduce beneficii semnificative pentru mediul înconjurător și pentru economie.

Mulţumiri. Aduc mulțumiri d-nei dr., lect. univ. Cristina Efremov pentru ajutor și sprijin la scrierea acestei cercetări.

Bibliografie

1. Comunicare a comisiei către Parlamentul European, Consiliu, Comitetul Economic și Social European și Comitetul Regiunilor. *Strategia UE pentru energia solară*. Bruxelles, 18.5.2022 COM(2022) 221. 27 p.
2. BÓDIS, K., KOUGIAS, I., JÄGER-WALDAU, A., TAYLOR, N., SZABÓ, S.: „A high-resolution geospatial assessment of the rooftop solar photovoltaic potential in the European Union" (O evaluare geospațială de înaltă rezoluție a potențialului fotovoltaic solar al acoperișurilor în Uniunea Europeană) (2019), Renewable and Sustainable Energy Reviews, 114, articolul nr. 109309.
3. EFREMOV C.: Tranziția energetica si principalele provocări pentru Republica Moldova https://ibn.idsi.md/sites/default/files/imag_file/79-82_13.pdf
4. EFREMOV C. „Contribuții la majorarea flexibilității sistemului energetic în vederea integrării surselor de energie regenerabilă", Teza de doctorat, Chișinău, 2022. C.Z.U.: 620.9:621.3(4+478) (043).
5. EFREMOV C. „Challenges of the energy transition in maintaining the energy system security and continuity". Quarterly publication of Romanian National Committee of World Energy Council (WEC/RNC) and The General Association of Engineers in Romania (AGIR) – EMERG 10 (Energy, Environment, Efficiency, Resources, Globalization, Volume 10) – ISSN 2457-5011; An V / 2019 – p.p. 43-55. https://cnr-cme.ro/wp-content/uploads/2019/11/EMERG-10-corectat-13.11_compressed.pdf
6. EFREMOV C., CERNEI M., Leu V., „Sustainable energy transition roadmap to 2050 for Republic of Moldova". Quarterly publication of Romanian National Committee of World Energy Council (WEC/RNC) and The General Association of Engineers in Romania (AGIR) – EMERG 10 (Energy, Environment, Efficiency, Resources, Globalization, Volume VIII) – ISSN 2668-7003, ISSN-L2457-5011; Volume VIII, Issue 3/2022, pp 11 – 25, https://emerg.ro/wp-content/uploads/2022/10/1-SUSTAINABLE-ENERGY-TRANSITION-ROADMAP-TO-2050-FOR-REPUBLIC-OF-MOLDOVA.pdf
7. EFREMOV C., ARION V., SĂNDULEAC M. „Short term daily storage need assessment for a large PV deployment scenario - preliminary case study for Republic of Moldova". 2021 10th International Conference on Energy and Environment (CIEM), 2021, pp. 1-5, doi: 10.1109/CIEM52821.2021.9614898. https://ieeexplore.ieee.org/document/9614898
8. EFREMOV C.. Photovoltaics solutions and energy communities in a clean energy roadmap. In: Journal of Engineering Sciences. 2022, nr. 3, pp. 110-125. ISSN 2587-3474.10.52326/jes.utm.2022.29(3).10 https://ibn.idsi.md/sites/default/files/imag_file/110-125_0.pdfhttps://doi.org/10.52326/jes.utm.2022.29(3).10.

9. ARMAROLI N., BALZANI V., Energy for a sustainable world. From the oil age to a sun-powerd future, Editura Wiley-VCH, Weinheim, 2011. LEBEDEV, A.A., CHERNOBROVKIN, L.S.
10. Agenției Naționale pentru Reglementare în Energie din Republica Moldova https://anre.md/garantiile-de-participare-si-de-buna-executie-3-338
11. Parlamentul Republicii Moldova LEGE Nr. 10 din 26-02-2016 privind promovarea utilizării energiei din surse regenerabile https://www.legis.md/cautare/getResults?doc_id=132691&lang=ro# https://www.legis.md/cautare/getResults?doc_id=132578&lang=ro
12. Photovoltaics in the built environment: A critical review https://www.sciencedirect.com/science/article/pii/S0378778821007635

RESULTS OF EXPERIMENTAL AND COMPUTATIONAL STUDIES FOR STEAM TURBINE EXHAUST HOODS APPLYING A WIDE-MODE DEFLECTOR

Alexander LAPUZIN[1], Valery SUBOTOVICH[1], Ivan MALYMON[2*]

[1]*Department of Turbine Construction, ScD, Institute of Education and Science in Power Engineering, Electronics and Electromechanics, National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine*
[2]*Department of Turbine Construction, PhD Student, Institute of Education and Science in Power Engineering, Electronics and Electromechanics, National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine*

*Corresponding author: Ivan Malymon, sevcrimea13@gmail.com

Tutor/coordinator: Yudin YURIY, Department of Turbine Construction, PhD, Institute of Education and Science in Power Engineering, Electronics and Electromechanics, National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine,

Abstract. *Research of exhaust hoods operation combined with the turbines' last stages at reduced volume flow rates is of high scientific importance. With a decrease of load in the diffusers, flow passage the bushing circulation zone of the separated flow increases, which affects the efficiency of the turbine outlet part. During the first phase, an aerodynamic experiment was executed, that studied an outlet hood model of the powerful steam turbine, as well as the turbine's last stage. Based on the results of the experiment carried out in wide operating modes range of the turbine last stage, it was determined that the installation of a wide-mode deflector at the inlet to the diffuser allows to reduce the separated circulation flows and the nonuniform flow in the outlet hood, as well as to reduce the total loss coefficient. During the second phase, computational studies were performed that used CFD four design forms of an axial-radial diffuser, distinguished by presence or absence of deflectors. It was determined that in the studied range of operating modes, all design forms of the diffuser with a wide-mode deflector have a positive effect on the flow – the total loss coefficient decreases and the efficiency of the turbine outlet part increases respectively, while the dimensions of the bushing circulation zone significantly decrease. The research results can be used both in the development of new exhaust hoods designs and in the modernization of the operated turbines flow path.*

Keywords: *circulation zone, diffuser, loss coefficient, operating mode.*

Introduction

Nowadays, many powerful steam turbines are operating at reduced volume flow rates for a long time [1]. The flow pattern in these operating modes contains flow separation from the diffuser walls of the outlet hood and developed circulation zones. This pattern significantly affects the turbine efficiency and the blade row reliability of the steam turbine's last stage. Therefore, the study of the turbine exhaust hoods operation within a wide operating modes range and the improvement of their aerodynamic characteristics is contemporary and is a promising method to increase the efficiency of turbomachines. This research contains comparative results of experimental and computational studies of steam turbine exhaust hoods using a wide-mode deflector. The aim of the given computational study is to confirm the effectiveness of the wide-mode deflector installation, and to verify performed calculations.

Numerous studies about the operation of the "last stage with exhaust hood" steam turbine compartment are aimed at investigating the three-dimensional flow fields in the diffuser and finding ways to improve their characteristics. Various options are proposed in order to reduce the exhaust hood resistance by optimizing its geometry parameters [2-5]. Several works [6-9] are dedicated to the investigating and finding possible ways to decrease the flow parameters circumferential nonuniformity downstream of the turbine's last stage by changing the flow path of the stage and improving characteristics of an axial-radial diffuser, and to minimize the separated flows [10-12].

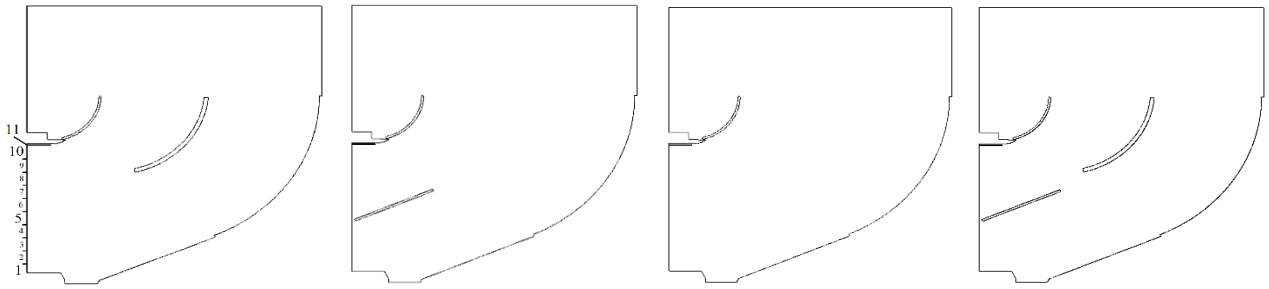
Efficiency increase is achieved by optimizing the design of the turbine's last stage and by modifying the geometry of the turbine's outlet diffuser [13].

On the other hand, the available literary sources in this research area are very limited. According to experimental studies of the low pressure cylinder's last stages of a steam turbine [14, 15], the low load operating mode of the last stages is characterized by streamlines contraction to the flow path periphery and the formation of a wide root zone of the separated flow, the size of which increases as the load decreases. Concurrently, the speed of the reverse steam flow in the circulation zone is quite high, which, due to the removal of moisture from the condenser, leads to the erosion of the rotor blades trailing edges of the turbine's last stage [16-18]. Recent studies of the flow in the exhaust hoods under low load operating modes allowed developing several designs with movable and fixed elements for more efficient flow control in a wide range of operating modes [3, 12, 19-20].

Research methodology

The research was executed in two steps. At first, aerodynamic experimental studies were performed to investigate the axial-radial diffuser models of the outlet hood of a powerful steam turbine, as well as its last stage. The "last stage and diffuser" compartment of a powerful steam turbine, made on a scale of 1:10, was studied at the air turbine stand. Three diffusers designs were tested, that had the same external and internal contours, but different elements inside the flow part: type 1 - with a toroidal deflector, type 2 - without any deflectors, type 3 - with a wide-mode deflector. The flow parameters were measured in four sections I, II, III, IV (Fig. 2, a) along the diffuser (I - inlet, IV - outlet). The experiments were executed in three operating modes of the last stage with *U/C* = 0.6; 0.74; 1.22-1.28 (*U/C* is operating mode characteristic parameter, circumferential to fictitious velocities ratio), that corresponded to the relative volume flow rate of the working fluid *GV2* ~ 1, 0.8 and 0.55. The influence of *U/C* on the compartment efficiency η_{com} and on the total loss coefficient ς_t with its components, internal and leaving losses, was determined.

Secondly, computational studies were performed using the ANSYS Fluent software for similar diffuser models under the same operating modes. The boundary conditions at the diffuser inlet corresponded to the experimental distribution downstream the last stage. Four designs of axial-radial diffusers were studied, where a rectangular mesh is built with clustering near the wall surfaces. The inlet section of the diffusers up to the shroud is divided into 11 segments to simulate the distribution of parameters behind the last stage (Fig. 1). The flow pattern in the flow part, as well as loss coefficients were determined.



Design 1 **Design 2** **Design 3** **Design 4**

Figure 1. Models of diffusers

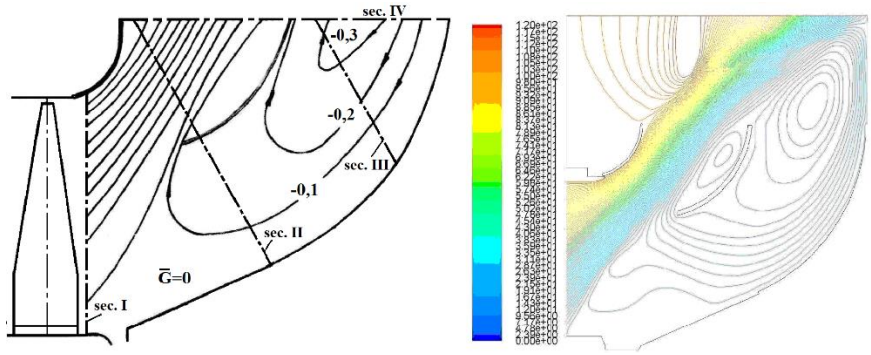
Results

A) Experimental studies

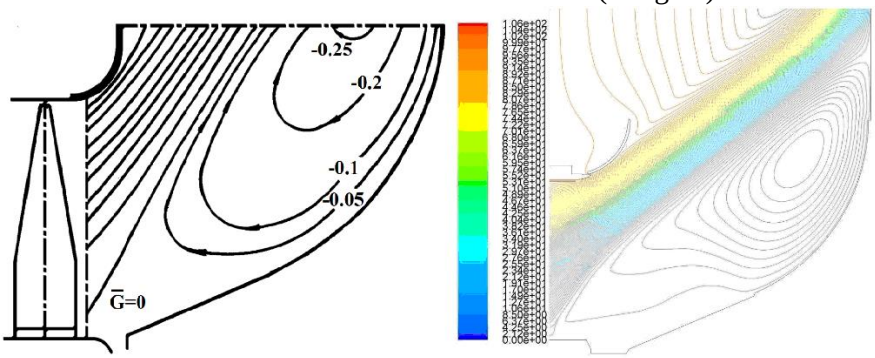
The influence of the last stage operating modes *U/C* on the flow in the diffuser flow path, the compartment efficiency η_{com} and losses was determined. In design mode *U/C* = 0.6, the flow at the outlet of the last stage is close to axial, the flow is contracted to the diffuser peripheral contour due to the leakage flow through the radial clearance of the rotor blade. Total loss coefficients are the same for both designs, but the values of their components are different: with a deflector installed, the leaving losses are reduced, which is compensated by the increase in internal losses. In the outlet

section of design 3, the streamlines are more rarefied near the internal contour. This flow is associated with an additional diffuser effect that occurs in the channel under the deflector.

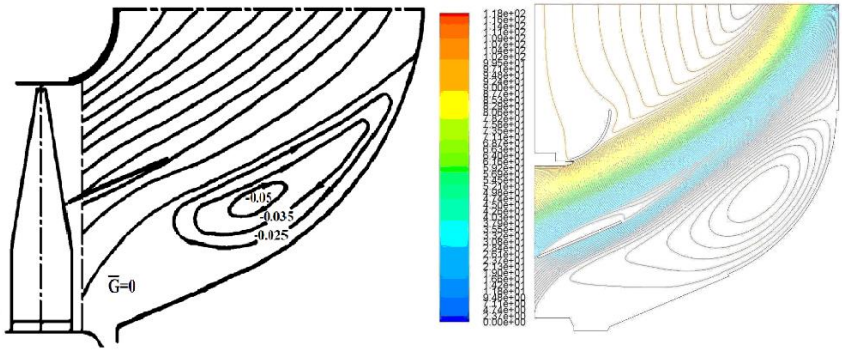
In the mode *U/C* = 0.74 in a diffuser 2, a flow separation from the internal contour appears, which is followed by a circulating flow from section II to section IV. More than 1.5% of the total flow rate of working fluid through the stage is involved in the circulating motion. With a deflector installed, no significant changes in the flow pattern are observed, however, the size of the circulation zone and the circulating flow rate decrease by 1.6 times.



a – diffuser with a toroidal deflector (design 1)



b – diffuser without deflector (design 2)



c – diffuser with a wide-mode deflector (design 3)

Figure 2. Flow in diffuser designs 1, 2 and 3 at operating modes *U/C* = 1.22-1.28

In the operating mode *U/C* = 1.22, the circulation zone in design 2 takes more than half of the diffuser area and extends beyond the outlet section IV. Up to 25% of the total flow rate of the working fluid through the stage is involved in the circulation motion (Fig. 2, b). The streamlines are contracted to the diffuser peripheral contour. In design 3, the resistance of the diffuser with a deflector decreases and the last turbine stage operates in the mode *U/C* = 1.28 (Fig. 3). The circulating zone does not expand beyond the outlet section, and the flow rate of the circulating working fluid is only 5% of the total flow rate of active flow.

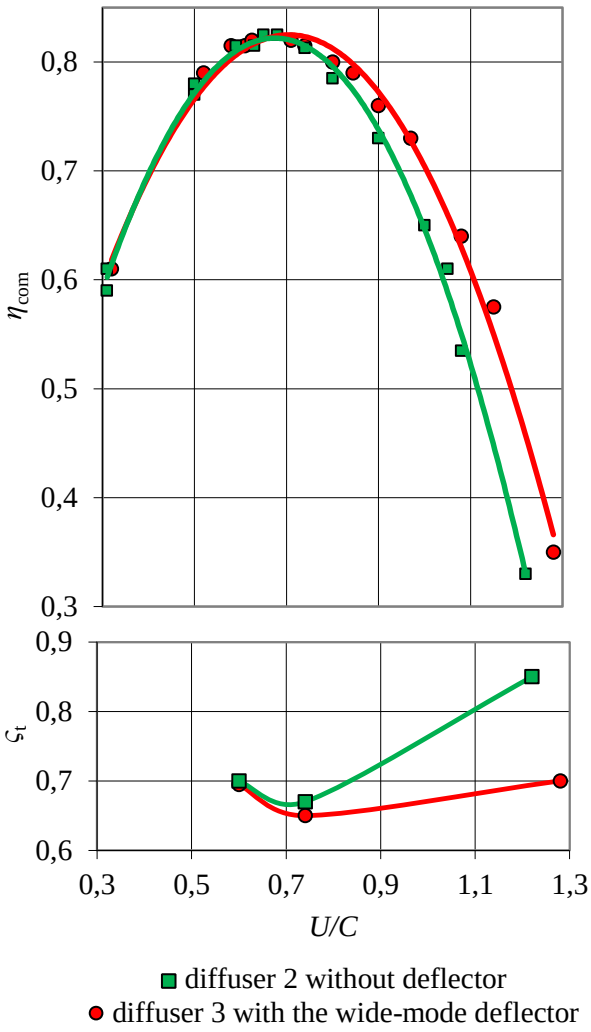


Figure 3. Dependence of total loss coefficient and efficiency on *U/C* ratio

The installation of the deflector leads to a change in the flow pattern in the diffuser. At the periphery of section I, the speed decreases, and it increases in the central zone. In sections II and IV, the flow becomes more uniform, and the circulation zone decreases due to the powerful flow jet formed under the deflector. In section IV of the diffuser with a deflector, the distribution of the flow rate component of velocity is almost uniform. The interaction of the flow with the deflector leads to a twofold increase in internal losses, however, at the same time, the leaving losses decrease by more than 2 times, and leading to a relatively low value of the ς_t. The use of a fixed wide-mode deflector in the diffuser makes it possible to keep the efficiency η_{com} at an unchanged level in the operating modes U/C = 0.3...0.74, and to increase it by reducing the total loss coefficient in the operating modes U/C > 0.74.

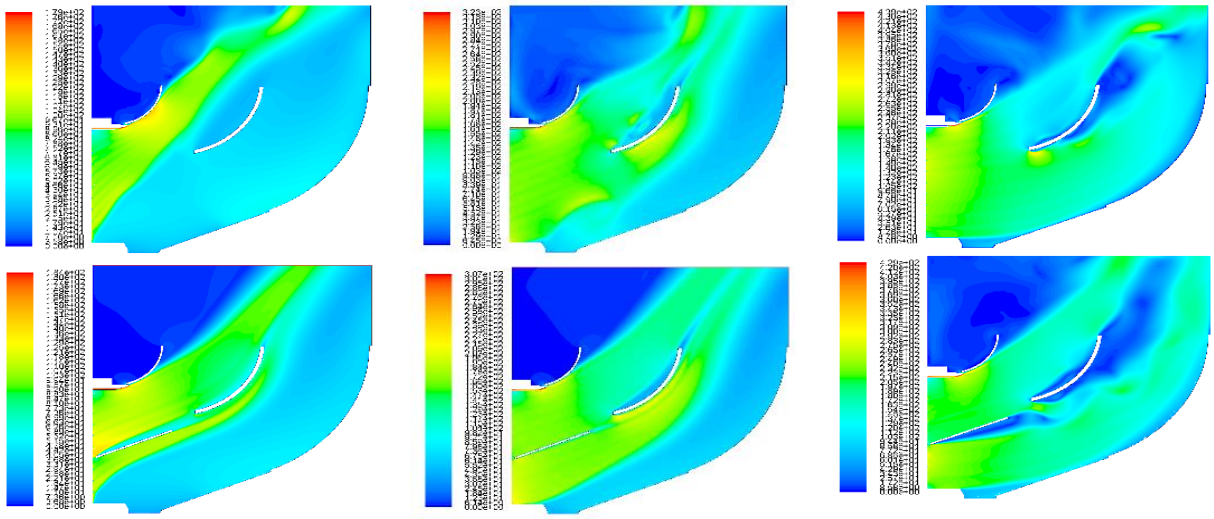
B) Computational studies

Four models of axial-radial diffusers were studied. The first three models, 1st, 2nd and 3rd designs, are similar to the diffusers that were studied at the air turbine stand. The 4th design of the diffuser - with toroidal and wide-mode deflectors Fig. 1 was studied for the first time. The idea was to test the performance of the wide-mode deflector under the conditions of an already existing standard toroidal deflector installed in the exhaust turbine hoods.

The Fig. 2 shows the comparison of the calculated equal flow rates lines distribution with experimental ones in the operating mode U/C = 1.22-1.28 of 1, 2 and 3 diffuser designs. The flow performance of the 1st diffuser design almost does not differ from the one of the 2nd design. The

flow velocity near the surface of the external diffuser contour is 1.5 times higher than in other zones due to the flow contraction and curvature changing. The circulation zone takes most part of the diffuser flow channel. Within the 3rd diffuser design the dimensions of the circulation zone decreased by ~ 2.5 times, and the flow rate of the working fluid involved in the circulation motion decreased by ~ 5 times.

The comparative computational study of the modernized diffuser model 4 and the original model 1 was executed Fig. 4. Installation of a wide-mode deflector has little effect on total loss coefficient, but it causes some improvement of the flow in the diffuser when simulating the modes *U/C* = 0.6 and *U/C* = 0.74. In off-design mode *U/C* > 0.74, the wide-mode deflector starts producing a more positive effect on the flow. When simulating the mode *U/C* = 1.28 in the modernized design 4 of the diffuser, the main flow is deflected by a wide-mode deflector to the diffuser bushing, the size of the circulating flow decreases. Concurrently, the total loss coefficient decreases by almost 1.5 times.



a – *U/C* = 1.22-1.28 b – *U/C* = 0.74 c – *U/C* = 0.6

Figure 4. Design distribution of velocity (m/s) in diffuser's design 1 (top row) and design 4 (bottom row) in transient operating modes

Conclusions

A computational model of the diffuser was tested, which makes it possible to reliably and efficiently simulate complex distributions of boundary conditions at the inlet to the diffuser of the exhaust hood, especially when simulating variable operating modes of the turbine last stage. Comparison with experiment showed that the flow pattern in various diffusers designs is adequately simulated even in off-design operating modes with flow separations and developed circulation zones.

The results of aerodynamic experimental and computational studies of axial-radial diffusers of the steam turbine exhaust hood show that the installation of a wide-mode deflector in the diffuser has a positive effect on the flow in the diffuser at variable operating modes. At the off-design modes, such a modernization can significantly reduce the size of the circulating zones and the total loss coefficient.

References

1. ZAYCEV, M., SLABCHENKO, O.: Modernizatsiya TsND turbin K-300-240 elektrostantsiy Ukrainy [Modernization of the K-300-240 steam turbine LPC at Ukraine's electric power plants]. In: Power Engineering and Electrification, 1996, 4, pp. 6-9.
2. GRIBIN, V., MITROKHOVA, O., PARAMONOV, A., REVENKO, A. Numerical investigations the effect of the overall dimensions on the efficiency of steam turbine exhaust

hood. In: *AIP Conference Proceedings 2047 (1)*. Pilsen: American Institute of Physics, 2018.

3. GRIGORYEV, E., ZARYANKIN, A., ROGALEV, A., GARANIN, I. Issledovanie i aerodinamicheskoe sovershenstvovanie vykhlopnogo patrubka nizkogo davleniya parovoi turbiny [Study and aerodynamic improvement of steam turbine low pressure cylinder exhaust nozzle]. In: *Vestnik IGEU*, 2017, 2, pp. 18-26.
4. GALAEV, S., RIS, V., SMIRNOV, E., BABIEV, A. Experience gained from designing exhaust hoods of large steam turbines using computational fluid dynamics techniques. In: *Thermal Engineering*, 2018, 65, pp. 352–361.
5. MUNYOKI, D., SCHATZ, M., VOGT, D. Numerical investigation of the influence of hood height variation on performance of low pressure steam turbine exhaust hood. In: *Proceedings of the ASME Turbo Expo 2018*. Oslo: ASME Turbo Expo, 2018, 8, pp. 11.
6. DING, B., XU, L., YANG, J., YANG, R., DAI, Y. The effect of stage-diffuser interaction on the aerodynamic performance and design of LP steam turbine exhaust systems. In: *Proceedings of the ASME Turbo Expo 2018*. Oslo: ASME Turbo Expo, 2018, 8, pp. 11.
7. SIDOROV, A., GOLIKOV, A., POLNIKOVA, T. Vliyaniye diffuzora na okruzhnuyu neravnomernost davleniya gaza v vykhodnom patrubke turbiny [Diffuser effect on circumferential gas pressure non-uniformity in the exhaust manifold of a steam turbine]. In: *Izvestiya vysshikh uchebnykh zavedeniy. Mashinostroyeniye*, 2015, 6 (663), pp. 20-25.
8. MITROKHOVA O., GRIBIN V., PARAMONOV A., GURYANOVA A., REVENKO A. Numerical simulation of 3D flow in the diffuser exhaust hood of the high-power steam turbine. In: *AIP Conference Proceedings 2189 (1)*. Pilsen: American Institute of Physics, 2019.
9. HOZNEDL, M., ŽIVNÝ, A., MACÁLKA, A., KALISTA, R., SEDLÁK, K., BEDNÁŘ, L., TAJČ, L. The pressure field at the output from a low pressure exhaust hood and condenser neck of the 1090 MW steam turbine: experimental and numerical research. In: *Proceedings of the ASME Turbo Expo 2018*. Oslo: ASME Turbo Expo, 2018, 8, pp. 11.
10. YIN, M., YANG, C., MENG, L., YAN, W., ZHUHAI, Z., LI, J., FENG, Z. Numerical analysis on the swirl flows in the exhaust hood of steam turbine and optimization design. In: *Proceedings of the ASME Turbo Expo 2016*. Seoul: ASME Turbo Expo, 2016, 8, pp. 9.
11. TABATA, S., FUKUSHIMA, H., SEGAWA, K., ISHIBASHI, K., KUWAMURA, Y., SUGISHITA, H. Experimental and numerical investigations of steam turbine exhaust hood flow field with two types of diffusers. In: *Proceedings of the ASME Turbo Expo 2019*. Phoenix: ASME Turbo Expo, 2019, 8, pp. 11.
12. FU, J., LIU J. Investigations on the improving aerodynamic performances of last stage steam turbine and exhaust hood under design and off design conditions. In: *Journal of Thermal Science*, 2015, 24, pp. 468-477.
13. YUDIN, Y., LAPUZIN, A. Povysheniye effektivnosti vykhlopnykh patrubkov TsND parovykh turbin s pomoshchyu shirokorezhimnogo deflektora [Increasing the efficiency of the exhaust hoods of the steam turbines LPC using a wide-mode deflector]. In: *NTU "KhPI" Bulletin: Power and Heat Engineering Processes and Equipment*, 2005, 6, pp. 60-64.
14. ALYOKHINA, S., ISHCHENKO, M., SLASTON, L., SHERFEDINOV, R. Experimental study of the model compartment of the low-pressure cylinder of K-320-240 turbines of JSC “Turboatom”. In: *Journal of Mechanical Engineering – Problemy Mashynobuduvannia*, 2019, 22 (4), pp. 6-10.
15. SHUBENKO, A., HOLOSHCHAPOV, V., STRELNIKOV, I., RESHYTKO, I. Vliyaniye krupnodispersnoy vlagi na rabochiye protsessy vlazhnoparovykh stupeney turbin [Effect of coarse particle moisture on work process of the wet steam stages of turbines]. In: *Energosberezheniye, Energetika, Energoaudit*, 2014, 11 (130), pp. 28-39.
16. SHUBENKO, A., KOVALSKIY, A., VOROBYEV, Y., KARTMAZOV, G., ROMANENKO, V. Vliyaniye erozii na osnovnyye ekspluatatsionnyye kharakteristiki

rabochey lopatki posledney stupeni tsilindra nizkogo davleniya moshchnoy parovoy turbiny. Chast 1. Prognozirovaniye erozionnoy opasnosti v poslednikh stupenyakh energeticheskikh turbin [Erosion influence on the main operational characteristics of the powerful steam turbine low-pressure cylinder last stage rotor blade. Part 1. Erosion hazard prediction in the power turbines last stages]. In: *Problemy Mashinostroyeniya*, 2009, 12 (4), pp. 7-16.

17. SENOO, S., WHITE, A. Analysis and design of wet-steam stages. In: *Advances in Steam Turbines for Modern Power Plants*, 2017, pp. 165–218.
18. LIN, A., CHANG, X., CAO, L., ZHANG, H., SUN, L. Effect of wet steam on aerodynamic performance of low-pressure exhaust passage with last stage blade. In: *Journal of Applied Fluid Mechanics*, 2019, 12 (6), pp. 1837-1845.
19. YUDIN, Y., SUBOTOVICH, V., LAPUZIN, A., MALYMON, I. Rozrakhunkove aerodynamichne doslidzhennia vykhlopnoho dyfuzora potuzhnoi parovoi turbiny v shyrokomu diapazoni rezhymiv roboty [Numerical aerodynamic investigation of exhaust diffuser for powerful steam turbine in a wide operating modes range]. In: *NTU "KhPI" Bulletin: Power and Heat Engineering Processes and Equipment*, 2020, 1 (3), pp. 5-9.
20. EPIPHANOV, V., GAEV, V., LISYANSKII, A., KIRILLOV, A., NIKOLAEV, M., SMIRNOV, E., ZAJTSEV, D. Effect of deflector vane geometry on performance of large-scale turbine exhaust hood at transonic flow conditions: air-test experiments and 3D numerical simulation. In: *Proceedings of the 5th European Conference Turbomachinery*. Praha: Environmental Science, 2003, pp. 803-812.

IMPACT OF QUALITY INDICATORS OF ELECTRICAL ENERGY ON METROPOLITAN ROLLING STOCK

Roman DEMIANENKO

Department of Automation and Cybersecurity of Power Systems,
National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine

Corresponding author: Roman Demianenko, Roman.Demianenko@ieee.khpi.edu.ua

Coordinator: Oleg GRIB, Professor, *National Technical University "Kharkiv Polytechnic Institute"*

Abstract. *The quality of electrical energy is becoming an increasingly sought-after factor that affects various parameters of electrical equipment, including reliability. In the article, the authors conducted a study of the influence of the quality of electric energy on the performance indicators of the metro rolling stock.*

Keywords: *quality of electric energy, electric discharge, wagon, metro, bearing, rolling stock.*

Introduction

Monitoring the electrical energy quality in the metro power system is uncommon, despite the importance of maintaining the system's performance. The electric motor of the rolling stock is the main consumer, which can deteriorate the quality of electrical energy. While passive filters can suppress harmonic surges, improvement systems are not typically utilized. Diagnosing quality of electrical energy through direct measurements at electrical substations is a promising method for the metro's rolling stock.

Quality parameters of electric energy in the electric transport network

One of the components of low quality electrical energy isthe presence of higher harmonics in the electrical network [1,2]. Supplies s of higher harmonics are traction substations ofelectric transport and other non-linear consumers.

A series of measurements showed that the deterioration ofthe quality of electrical energy in electric transport is a consequence of the operation of the electrical equipment of thesubway and various electrical devices contribute to the deterioration of the quality parameters of electricity. At the same time, it was found that the harmonic components have an influence on the technical elements and nodes of the rollingstock [3, 4].

Therefore, the question of determining the element that leads to the deterioration of the quality of electrical energy onrolling stock is an urgent direction in the development of diagnostic equipment.

The parameters of the quality of electrical energy were determined in various nodes of electrical network schemes. Measurements were performed on the border of the supplier's and consumers' balance sheets.

Electric power quality measurements were performed at the connection of the main network substations of SMES andJSC "Yuzhd" using microprocessor analyzers of voltages andcurrents in electrical networks of the ANTES AK-3F type in accordance with GOST 13109-97.

The total duration of continuous measurements, taking intoaccount the requirements of clause 5.2.1, was at least seven days. The volume of measurements made during NEE monitoring [5] allows us to assume that the values and rangesof changes in the electrical energy parameters in the network, KU and KU(n), are reliably determined. In total, more than 50PKE measurements were carried out at electric power facilities located on the territory connected as consumers to the JSC "Yuzhd" substation located in the city of Kharkiv.

Received detailed protocols of electricity quality measurements for each indicator, which were obtained as a result of measurements at each control point.

Below are some final results of measurements of thequality of electrical energy at the studied facilities, Kharkiv.

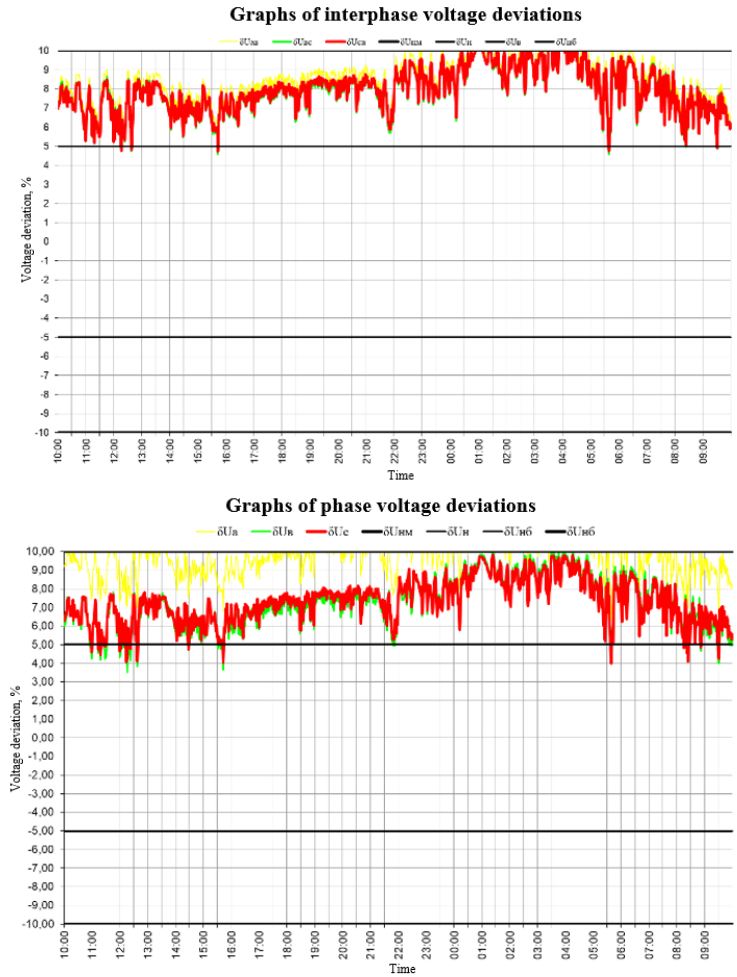


Figure 1. Graphs of interphase and phase voltage deviations.

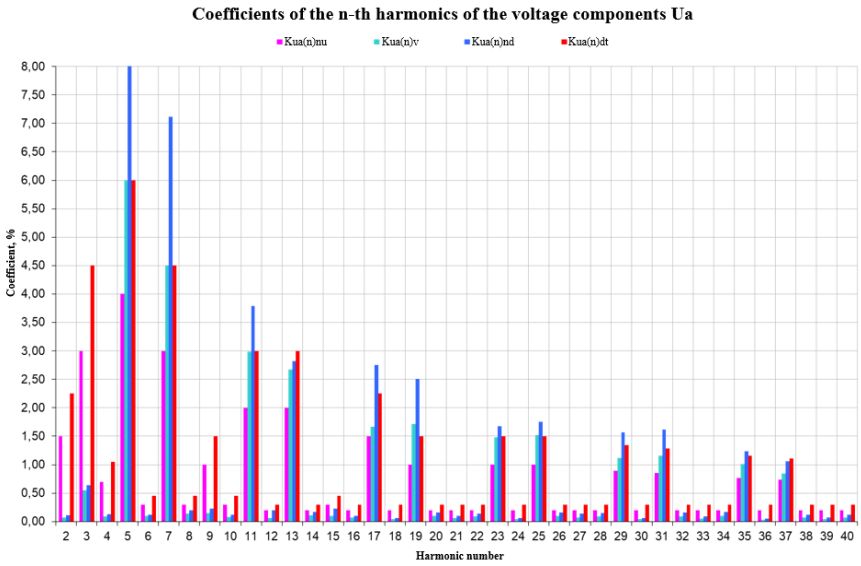


Figure 2. Coefficients of the n-th harmonics of the voltage components Ua

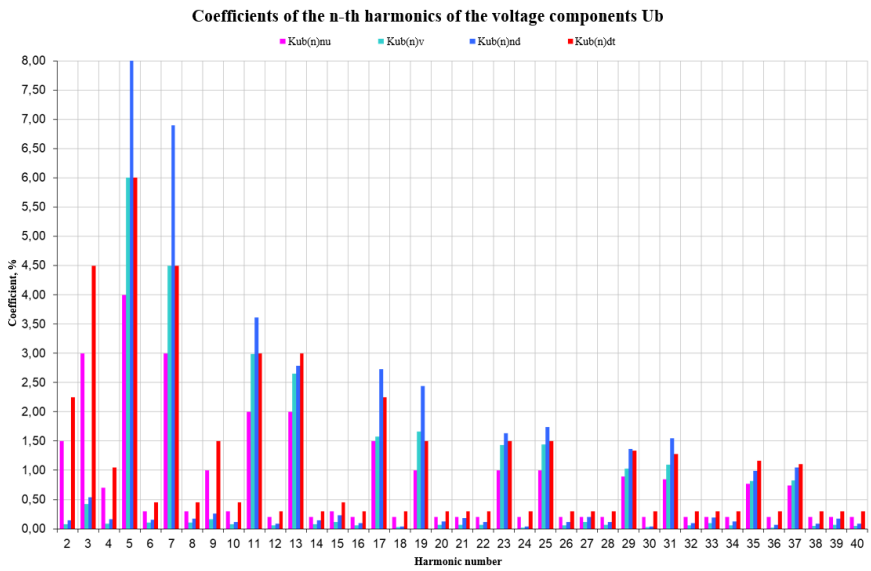


Figure 3. Coefficients of the n-th harmonics of the voltage components Ub

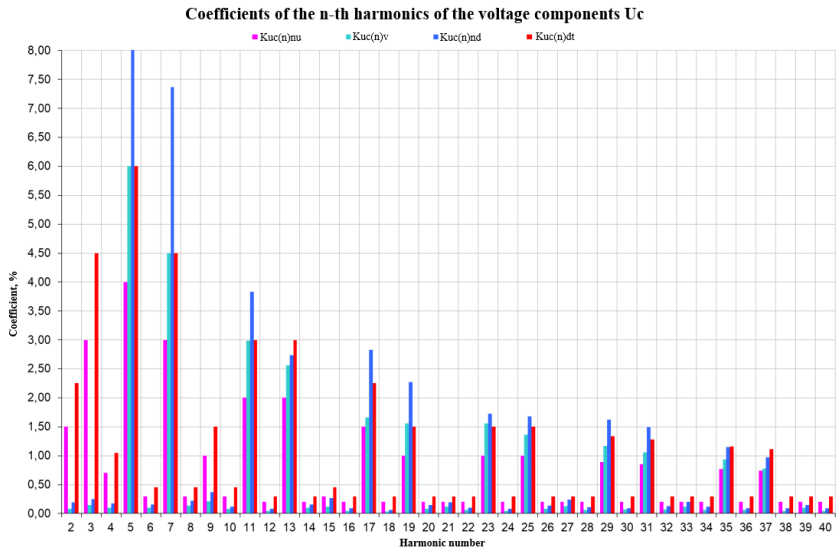


Figure 4. Coefficients of the n-th harmonics of the voltage components Uc

The results of measurements of KU(n) values in the distribution network at the substations where the measurements were made demonstrate a pronounced predominance of the 11th and 13th harmonic components, which significantly exceed the normalized values [6, 7].

The presence of harmonic components in the frequency range of the 11th and 13th harmonic components is due to the influence of the alternating current rectification scheme used in the consumer's power supply system with a non-linear load[8].

In Fig.2-4, it is shown that the 5th and 7th harmonic components prevail in the frequency range, due to the influence of the alternating current rectification scheme usedin the traction load power supply system.

Directly determining the spectral components for elements of rolling stock schemes can be difficult due to the movement of rolling stock and the presence of network disturbances, which can lead to unpredictable results. Additionally, such measurements and devices can be expensive. As a result, developers often place technical control sensors directly on the engines or responsible nodes.

The author of the article propose a technique for quickly determining the deterioration of rolling stock by measuring the quality parameters of electrical energy, which involves structuring electrical disturbances from various elements of rolling stock.

General characteristics of the problems of the influence of poor-quality electric energy on metropolitan rolling stock

Electric system of metro wagon:

The body of the wagon is connected by flexible jumpers tothe frames of the trolleys, and the frames of the trolleys are connected by jumpers through the box body to the wheel pair.Grounding nodes are designed to form a power chain: contactnetwork - current collector - traction motors - grounding nodes – wheel.

The grounding node ensures that the current passes through the wheel, bypassing the bearings of the axles. But inthe case of additional grounding of the wagon body, it is connected to the axle box with a copper busbar. Grounding ofthe wagon body is performed by two tires at the beginning andend of the wagon.

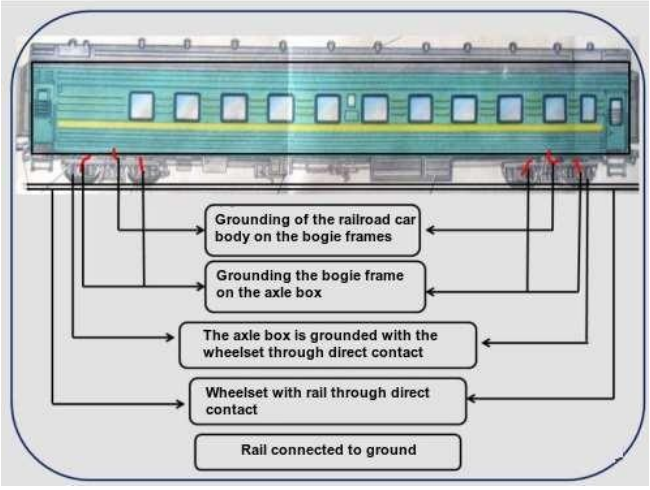


Figure 5. Places of grounding of the wagon body.

Grounding is carried out using shunts located on the bogie:two shunts per wagon on each side of the "body-bogie" and two shunts on each bogie diagonally "bogie-box".

Grounding shunts are installed so that the reverse currentduring the operation of electrical equipment passes through grounding shunts with low resistance, and not through detachable connections, which become non-conductive duringthe movement of the rolling stock under the influence of vibration.

Rolling bearing defects:

- Fatigue surface failures are associated with lubricationproblems (lubricant mismatch, low viscosity, and oil film breaks).
- Atmospheric corrosion. Corrosion is caused by moisture entering the bearing from the atmosphere. Moist air, getting into the middle of the bearing, condenses when the environment cools, breaking the oil film in the places of contact between the bodies andthe raceways.
- Fretting corrosion. Fretting corrosion is very similar tonormal corrosion. It occurs on the surfaces of the bearing on the shaft, as well as on other adjacent surfaces. It is caused by minor (microscopic) loads.
- During brinelling, regular indentations appear on the surfaces of the rings, one by one. This is a consequenceof plastic deformations of the metal in the places of thedents, which arise as a result of overstressing the metal.Brinelling is a consequence of high static or shock loads, incorrect bearing installation technology, strongmechanical shocks that occur when the machine falls.

Supplies of electrical potential on thewagon body

The body of the wagon is one of the conductors for the passage of current through the electric units of the wagon, which include: traction engine, engine control system, automation, low-voltage systems of the wagon (lighting, communication, telemetry system, etc.). The potential

can alsobe induced by air friction during the movement of an electrictrain, or by the influence of electromagnetic waves from power conductors [10].

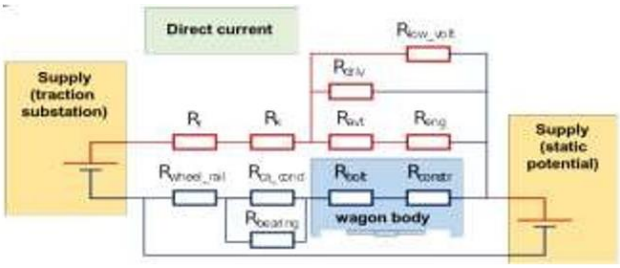


Figure 6. Direct current flow diagram through the wagon body.

The metro's rolling stock is powered by direct current Fig.6. A traction substation is used as a supply of direct current. The supply (traction substation) rectifies the current from the electrical network with an industrial frequency of 50 Hz.

To summarize, the potential on the wagon body during movement is variable and should be calculated for alternating current due to current surges and harmonic components of the current in the power supply network of the rolling stock. Rolling bearings have characteristic frequencies of defects, which can be detected in the spectrum of harmonic bursts in the draining current. The shape of the current at the moment of a surge at the bearing frequency can also provide information about the presence of breakdown or contamination of the bearing lubricant.

Conclusions

The research focused on the impact of current leakage on the support bearing of the metro car. The study revealed that the support bearing undergoes erosion damage due to the purely electric nature of the causes, which are a result of the action of electric discharges: sparking, corona, and arc. Since the bearing is an element of the electric circuit, the surges from the discharges affect other elements of the electric circuit of the metro car, and due to the rough surface of the bearing, its contact receives nonlinear resistance, which also leads to changes in the shape of the current.

References

Books:

1. SOKOL E, GRUB O., STARENKYI V., ZAKOVOROTNYI O., BAZHENOV V.,. VLADIMIROV Yu, KARPALYUK I., *"Relay protection and cyber security of power systems"* (Textbook / Under the general editorship of the corresponding member of the National Academy of Sciences of Ukraine, Doctor of Technical Sciences, Professor E.I. Sokol). - Kharkiv: FOP PANOV O.M., 2019. - 390 p.
2. GRUB O., SENDEROVICH G., SHCHERBAKOVA P. *Actual problems of determining equity participation in liability for violation of the quality of electrical energy.* Science practices DonNTU - Electrical engineering and energy. 2013. No. 1(14). pp. 77-82.
3. ZHEZHELENKO I. V., SAENKO Yu. L.. *Indicators of the quality of electricity and their control at industrial enterprises.* 3rd ed., revised. and additional M: Energoatomizdat, 2000, 252 p.
4. IEEE Std 1159-1995. IEEE Recommended Practice for Monitoring Electric Power. Approved June 1995. IEEE Standards Board. 70 p.
5. GRUB O., DEMIANENKO R. *"Quality monitoring in the electric network under the conditions of digital energy" at the III International Scientific and Technical Conference* "Energy Efficiency and Energy Security of Electric Power Systems (EEES-2019)". Collection of scientific works. Kharkiv: "Madrid Printing House", 2019. - 148 p.
6. GOST 30804.3.2-2013 (IEC 61000-3-2:2009). *The compatibility of technical means is electromagnetic.* Emission of harmonic current components by technical means with a current

consumption of no more than 16 A (in one phase). Test standards and methods. [Enter 2014-01- 01]. M: Standartinform, 2014. 25 p.

7. GOST 30804.3.12-2013 (IEC 61000-3-12:2004). *The compatibility of technical means is electromagnetic.* Norms of harmonic current components, created by technical means with a current consumption of more than 16A, but not more than 75A (in one phase), connected to low-voltage distribution systems of electricity supply. Test standards and methods. [Enter 2014-01-01]. M: Standartinform, 2014. 21 p.
8. ZHEZHELENKO I. *Higher harmonics in power supply systems of industrial enterprises* / ZHEZHELENKO IV. – 4th ed., revised. and additional - M. : Energoatomizdat, 2000. - 331 p.
9. CHERMENSKYI O., FEDOTOV N. "Rolling bearings" in Mashinostroenie, - 2003.576 p.
10. PANCHENKO C., BLINDYUK V., BABAEV M., YATSKO S., VASHCHENKO Y. *„Power supply systems of electric rolling stock of railways and metros”* in UkrDUZT 2018 – 308p.

ANALYSING HOUSEHOLD CONSUMERS' DEMAND RESPONSE THROUGH BEHAVIOURAL LOAD PROFILES

Vladyslav KIIANCHUK

Electric power station, A-2022/ Ph.D. student, IEEE, National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine

Corresponding author: Kiianchuk Vladyslav, Vladyslav.Kiianchuk@ieee.khpi.edu.ua

Tutor/coordinator: Kostiantyn MAKHOTILO, Ph.D., professor, IEEE, National Technical University «Kharkiv Polytechnic Institute»

Abstract. *The study is devoted to the involvement of residential consumers in demand-side management in the power system. The results of a pilot program involving aggregators of residential consumers in demand response in the UK are analysed, and barriers to its implementation are identified. It is proposed to use a behavioural load profile generator to simulate household participation in demand response. Two typical load reduction strategies for a two-adult household are simulated, and their efficiency is compared. It is estimated that shifting the use of high wattage appliances can reduce the household load by half without discomfort, whereas a mistaken strategy of limiting lighting usage gives three to seven times smaller effect. Thus, it is shown that behavioural load modelling can be used to solve the problem of determining effective strategies for the participation of residential consumers in demand response.*

Keywords: *demand response, energy management, load shifting, peak demand reduction*

Introduction

Demand management plays an important role in ensuring the stability of the energy supply and optimizing the costs of electricity production in the energy system. The application of demand management methods allows for maintaining power balance in the system, reducing peak loads, and lowering prices in wholesale markets. There are many studies on this topic, but they are mainly focused on large industrial consumers [1]. The case of small consumers' participation in demand management programs, especially in demand response through aggregators, is less studied and requires additional research.

An aggregator is an entity that brings together multiple low-capacity consumers such as small businesses or individual households and represents them in energy markets. Aggregators play a key role in demand response programs for household consumers, by coordinating energy consumption, optimizing the aggregated load, providing flexibility to the grid and participating in energy markets.

A recent example of an aggregator's capabilities is the demand response program (DRP) of the Octopus Energy company [1] within the Demand Flexibility Service of the National Grid ESO [**Error! Reference source not found.**] in the UK. This is a program that allows customers to earn m oney by reducing their electricity consumption during periods of high demand on the power grid. It is designed to balance the consumption, which can fluctuate significantly throughout the day. Octopus Energy provides enrolled customers with signals to reduce energy consumption by either turning off appliances or shifting their use to a different time. The program is flexible, allowing customers to choose the amount of energy they wish to save and when to do it. Octopus Energy rewards customers for reducing energy consumption with points that can be redeemed for prizes, donations, or account credit.

In general, there are several ways in which a household can participate in DRPs:

- Time-of-use tariffs: DRP offers time-varying tariffs, which means that the price of electricity varies depending on the time of day;

- Energy storage: households can also participate by installing energy storage systems, such as batteries;
- Behaviour modification: households can participate by adjusting their behaviour, e.g., delaying the use of non-essential appliances such as a washing machine during peak demand or simply turning them off.

Challenges in implementing the demand response

Many residential customers who participated in DRPs reported their positive experiences. They found that by regulating the use of appliances during peak demand, they could significantly reduce their electricity bills. But some participants were disappointed by the slight payback and the great inconvenience of their efforts. As more consumers become aware of the benefits of demand response, it is likely that programs like Octopus Energy's will become more popular and widespread.

DRPs have the potential to offer many benefits to both household consumers and the grid. However, severe challenges may arise when attempting to engage and incentive household consumers to participate in demand response activities. One of the most significant of these is limited consumer awareness. Another problem is the weakness of financial incentives or the difficulty of obtaining them.

Behavioural barriers can hinder the engagement of household consumers in DRP. Despite understanding the benefits, some consumers may be unwilling or unable to change their energy behaviour, such as sacrificing comfort or altering daily routines. Scaling DRPs can also be challenging due to variations in habits among different communities.

Simulation of participation in DRP

Thus, the effective design and implementation of DRP require careful planning, ongoing evaluation and adjustment. Through the load simulation, it is possible to estimate the amount of energy that will be needed during peak demand and the possible output of consumer's response. This can be done using behavioural load profile models such as the Load Profile Generator (LPG) [1].

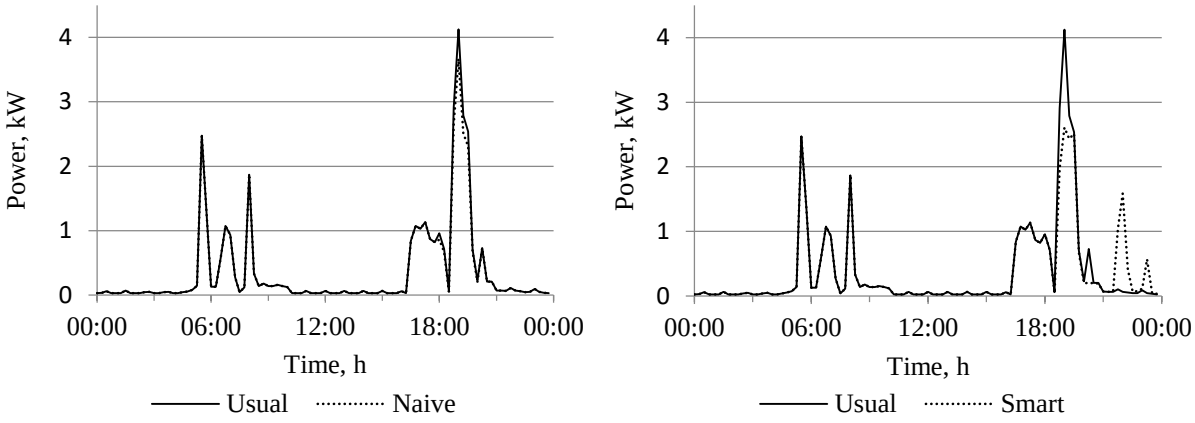
LPG can simulate the energy consumption of a given household or business over a certain period of time, taking into account a number of parameters such as the size of the building, the number of residents or employees, the type of their activities, and more.

In this study, LPG was used to simulate the effect of different strategies of household consumers involved in DRP. The object of the study was a family of two adults who work in an office. Load profiles were generated and investigated for two days when the family used the washing machine in the morning and in the evening. Two different strategies were considered. The first, "Naive" strategy represents inexperienced customers who simply turn off the lights and TV to reduce energy consumption. The second, "Smart" strategy represents customers trying to change the time they use powerful appliances, in this case a washing machine. According to the terms of the DRP, the consumer should reduce consumption during peak load from 8:00 to 10:00 or from 18:00 to 21:00.

Fig. 1, 2 show usual summer working days load profiles. Curves "Usual" show profiles for unlimited power consumption, the other curves show profiles when household participating in DRP under "Naive" (Fig.1a) and "Smart" Fig.1b strategies.

In Fig. 1, evening washing starts at a convenient for the family time from 18:30. A "Naive" consumer simply turns off the appliances in the living room to participate in DRP, while a "Smart" one shifts the start of washing to the night period from 21:30.

The simulation results show that the "Smart" strategy allows to reduce energy consumption during the demand response period (from 18:00 to 21:00) from 4.12 kWh to 2.44 kWh, as well as reduce the peak load by 1.7 kWh or 40%. On the same day, the "Naive" strategy only allows to reduce consumption during demand response period to 3.39 kWh, and reduce the peak load by 0.73 kW or 11%.

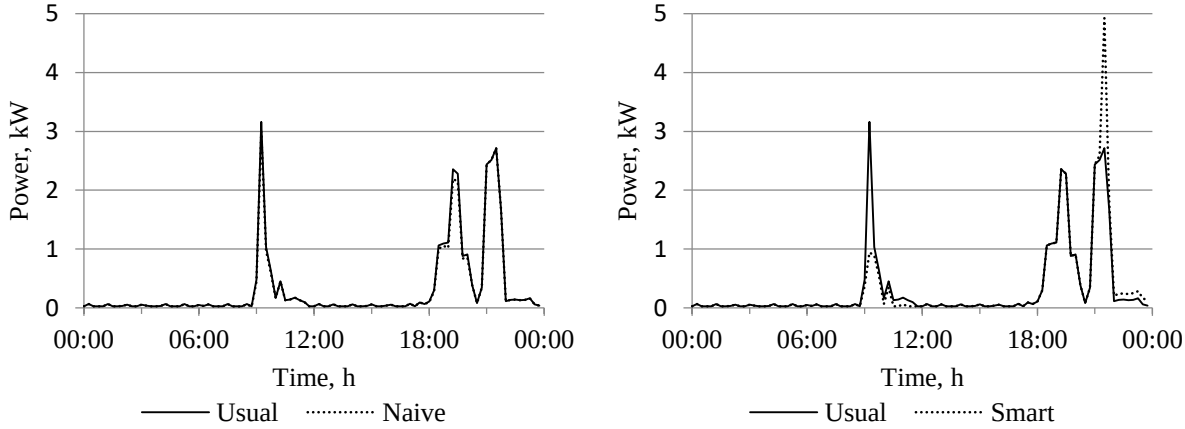


a) **b)**

Figure 1. Load profiles of a household consumer on a working day with washing in the evening

Fig. 2 shows the simulation results for another case where the morning washing starts at 8:30. This time, the consumer is trying to reduce energy consumption during the morning peak from 8:30 to 10:00. Again, a "Naive" consumer Fig.1a) turns off the appliances in the living room, while a "Smart" one Fig.1b) shifts the start of washing to the night period from 21:15.

The simulation results show that the "Smart" strategy allows to reduce energy consumption during the demand response period (from 8:30 to 10:00) from 3.16 kWh to 0.94 kWh, as well as reduce the peak load by 2.21 kWh or 71%. The "Naive" strategy allows to reduce consumption during demand response period to 3.0 kWh, and reduce the peak load by 0.11 kW or 9%.



a) b)

Figure 2. Load profiles of a household consumer on a working day with washing in the morning

These results clearly demonstrate the importance of choosing the right strategies for household consumer's demand response. Shifting washing to the night period almost does not create inconvenience to family, while it provides a 3-7 times greater reduction in consumption compared to almost stopping the use of the living room. It is clear, that the "Naive" strategy makes people uncomfortable, restricts their actions and prevents them from benefiting from the DRP and therefore from being rewarded for participation. Along with this, the chosen "Smart" strategy with similar ones related to other household appliances demonstrates great potential and scalability of residential consumers' engagement in DRPs [4].

Conclusion

DRP for household consumers can be an effective tool for managing energy demand, but these programs face several challenges, including limited consumer awareness and behavioural barriers. To address these challenges, DRP may rely on load profile simulation to find out household consumers' energy usage patterns and strategies for reducing energy consumption.

This study demonstrates the importance of behavioural approaches in promoting sustainable energy consumption practices through DRP in smart grids. It highlights the need for careful design and implementation of DRP programs that are effective, attractive to participants, and address behavioural barriers. It is shown that such programs can significantly influence consumer behaviour and lead to significant energy savings. The study emphasizes the need for carefully designed and implemented DRP programs that are effective and attractive to households, with education and outreach activities to build awareness and trust. With further and development, behavioural approaches can play a critical role in enhancing the effectiveness of DR programs and promoting a more sustainable energy future.

References

1. HONARMAND M., HOSSEINNEZHAD V., HAYES B., SHAFIE-KHAH M. and SIANO P. An Overview of Demand Response: From its Origins to the Smart Energy Community. In *IEEE Access*, 2021, 9, pp. *96851-96876*. http://doi.org/10.1109/ACCESS.2021.3094090
2. Saving Sessions, Octopus. [online]. [Acces at 01.03.2023]. Available: https://octopus.energy/saving-sessions/
3. Demand Flexibility Service, National Grid ES. [online]. [Acces at 01.03.2023]. Available: https://www.nationalgrideso.com/industry-information/balancing-services/demand-flexibility
4. Spectral Database for Organic Compounds, LPG. [online]. [Acces at 01.03.2023]. Available: https://www.loadprofilegenerator.de/
5. KIIANCHUK, V. M., MAKHOTILO, K. V. Estimation of Power Grid Load Transfer Using the Demand Response Service. In: *16th International Scientific and Practical Conference of Master's and PhD Students, Section 2: Electrical engineering and electromechanics, radio engineering and power engineering* - 2022, Kharkiv, 14-16 December 2022. Kharkiv: National Technical University «Kharkiv Polytechnic Institute», 2022, pp. 157-158.

АНАЛИЗ РАБОТЫ КОНДЕНСАЦИОННО-ОХЛАДИТЕЛЬНОГО УЗЛА В УСЛОВИЯХ ПОВЫШЕННОЙ ТЕМПЕРАТУРЫ НАРУЖНОГО ВОЗДУХА

Андрей ГЕЛДЫШЕВ

Студент 4 курса, кафедра теплотехники и энергоэффективных технологий, Институт энергетики, электроники и электромеханики, Национальный технический университет «Харьковский политехнический институт», Харьков, Украина

*Автор-корреспондент: Александр Переселков, e-mail: pereselkovar@gmail.com

Куратор/координатор: Александр ПЕРЕСЕЛКОВ, к.т.н, доцент, Институт энергетики, электроники и электромеханики,, кафедра теплотехники и энергоэффективных технологий, Национальный технический университет «Харьковский политехнический институт», Харьков, Украина

Аннотация. *Работа посвящена анализу эффективности работы действующего конденсационно-охладительного узла, состоящего из барометрического контактного конденсатора и вентиляторной градирни в летний период, когда параметры наружного воздуха превышают проектные.*

Ключевые слова: *барометрический контактный конденсатор, вентиляторная градирня*

Введение

При проектировании вентиляторных градирен их расчеты производятся для наиболее неблагоприятных атмосферных условий в летние месяцы на основе рекомендаций государственных стандартов. Существующие тенденции говорят о том, что климат изменяется в сторону потепления. Предварительные расчеты, проведенные авторами, показали, что почти треть летнего периода в климатических условиях г. Харькова градирня не будет обеспечивать заданного теплосъема (в то время как в нормах заложена возможность превышения параметров наружного воздуха в течение периода от 1 до 10 дней в зависимости от назначения установки). Следовательно, любой теплотехнологический комплекс, где используется градирня, будет работать с превышением расчетных условий. Целью работы является анализ влияния параметров наружного воздуха на работу установки, которая уже находится в эксплуатации, и формирование рекомендации по повышению ее эксплуатационной надежности.

Объектом исследования является конденсационно-охладительный узел вакуум-выпарной установки, состоящий из контактного барометрического конденсатора и вентиляторной градирни. Расчетные параметры конденсатора: расход пара $G_п$ = 8,5 т/ч; давление конденсации $P_к$ = 0,11·10^5 Па (температура конденсации $t_к$ = 47,7 °С), диаметр конденсатора D = 1,6 м. Конструкция конденсатора показана в [1]. Вентиляторная градирня имеет размеры секции 4×4 м, число секций N = 3. Каждая секция оборудована вентилятором 1ВГ-25. Оросительь асбоцементный с горизонтальными гофрами, двухрядный, высота оросителя $h_{ор}$= 2,44 м.

1. Расчет и анализ оптимальных условий эксплуатации контактного барометрического конденсатора

Расход охлаждающей воды, необходимый для конденсации пара, определяется из уравнения теплового баланса, кг/с

$$G_{х.в.} = \frac{G_п\left(i'' - c_{pв}t_{см}\right)}{c_{pв}\left(t_{см} - t_{х.в.}\right)}, \quad (1)$$

где $t_{х.в.}$ – температура охлаждающей воды, °С; $t_{см}$ – температура смеси води и конденсата пара, выходящей из конденсатора (принимается $t_{см}$ = 42 °С); i'' - энтальпия пара при температуре конденсации $t_к$), кДж/кг; $c_{рв}$ – теплоемкость воды, кДж/(кг·К).

Расход смеси воды и пара на выходе из конденсатора находится как $G_{см} = G_{х.в.} + G_п$.

Для задания параметров расчета конденсатора необходимо прежде всего определиться с параметрами наружного воздуха, влияние которых на работоспособность конденсационно-охладительного узла исследуется в настоящей работе. Остановимся на четырех расчетных режимах (1, 2, 3, 4), для которых температура θ_1, относительная влажность φ и температура по мокрому термометру τ указаны в табл. 1 и на Рис. 1 и
Рис. 2. Также в Таб.1 приведены значения расхода охлаждающей воды $G_{х.в.}$ и расхода смеси $G_{см}$ для каждого из принятых режимов.

Таблица 1

Параметры для расчета и анализа работы конденсатора

Режим	Параметры			$G_{х.в.}$, кг/с	$G_{см}$, кг/с
	θ_1, °С	φ, %	τ, °С		
1	24,9	52	18,6	71,7	74
2	30	35	19	75,1	77,4
3	33	33	21,2	86,2	88,6
4	35	30	22,5	92,4	94,8

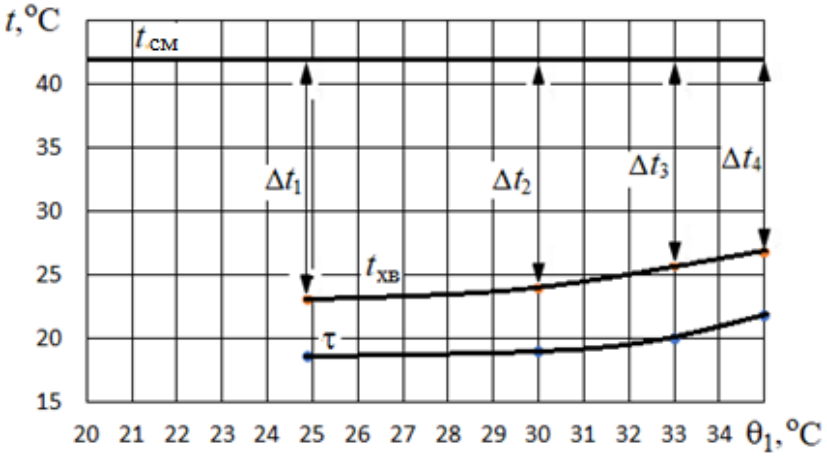


Рисунок 1. Температура холодной воды в зависимости от параметров наружного воздуха

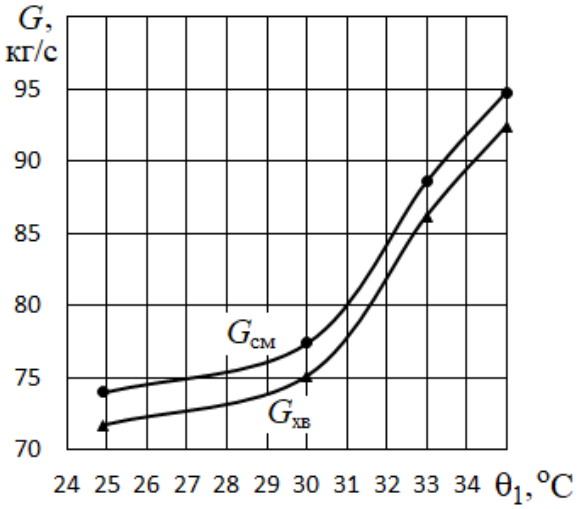


Рисунок 2. Расход воды в конденсаторе при различных параметрах наружного воздуха

На Рис. 3 приведены результаты гидродинамического расчета [3] для определения диаметра конденсатора при принятых расчетных режимах 1–4 по параметрам наружного воздуха. Как видно из Рис. 3, расход воды $G_{х.в.}$ = 90 кг/с является предельно допустимым для конденсатора с D = 1,6 м, во избежание гидроударов, вибраций и т.д., когда эксплуатация аппарата не рекомендуется.

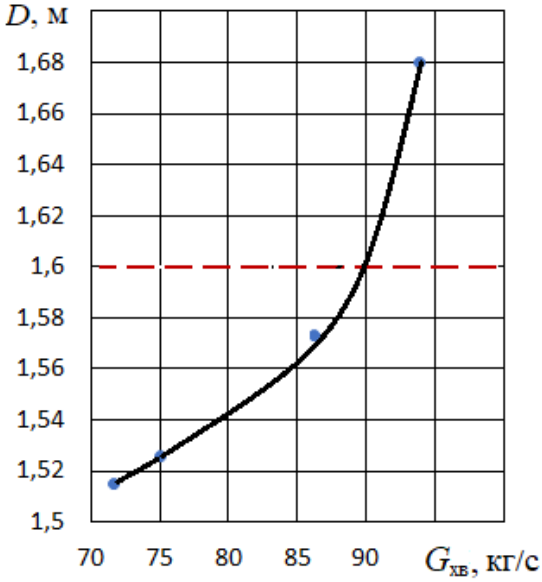


Рисунок 3. Результаты гидродинамического расчета диаметра конденсатора *D* в зависимости от параметров наружного воздуха (режимы 1–4)

2. Анализ работы вентиляторной градирни при повышенной температуре наружного воздуха

В соответствии с методикой расчета вентиляторной градирни [2], решая уравнение теплового баланса, определяем все величины, которые входят в уравнение передачи теплоты

$$Q = \beta_{xv} \cdot V_{ор} \cdot \Delta I_{ср} \cdot k, \quad (2)$$

где β_{xv} – объемный коэффициент массоотдачи, кг/(м^{3}с); $V_{ор}$ – объем оросителя, м^3; $\Delta I_{ср}$ – средний напор по энтальпии, кДж/кг с. в.; k – коэффициент, учитывающий уменьшение количества воды за счет испарения.

Объемный коэффициент массоотдачи

$$\beta_{xv} = A \cdot g \cdot \lambda^m, \quad (3)$$

где A и m – коэффициенты для конкретного типа оросителя (для рассматриваемого оросителя A = 0,72; m = 0,66); g – плотность орошения, кг/(м^{2}с), $g = G_{см} / F_{ор}$.

Из уравнения теплового баланса находится объем оросителя $V_{ор}$, м^3, и высота оросителя, м

$$h = \frac{V_{ор}}{F_{ор}}. \quad (4)$$

Для стандартной градирни с двухъярусным оросителем из волнистых асбоцементных листов $h_{ор}$ = 2,44 м. Если расчетное значение h меньше, чем высота оросителя $h_{ор}$, установленного в градирне, значит, охлаждающая способность градирни достаточна для заданных параметров ее работы.

Как следует из рис. 1, с повышением температуры наружного воздуха уменьшается глубина охлаждения воды Δt_i, а из Рис. 2 видно, что также возрастает расход воды $G_{см}$, подаваемой на градирню, при это возрастает плотность орошения g. Эти факторы способствуют улучшению условий охлаждения воды в градирне. Необходимо отметить, что такая зависимость объемного коэффициента массоотдачи от плотности орошения в градирне объясняется тем, что значительное охлаждение воды происходит в капельной части аппарата,

то есть от форсунок до оросителя, которая для рассматриваемой градирни равна 1 м. Увеличение расхода воды на градирню реализуется за счет повышения давления воды перед форсунками, что приводит к уменьшению размера капель (и, соответственно, к увеличению контактной поверхности между каплями и воздухом) и повышению угла раскрытия факелов центробежных форсунок, что способствует более равномерному заполнению каплями подфакельной зоны. Учитывая это, действующая градирня может эффективнее охладить воду и обеспечить функционирование конденсационно-охлаждающего узла при повышенной температуре наружного воздуха. Если для реализации охлаждающей способности градирни нужно увеличить расход воды выше критического для данного диаметра конденсатора, следует сливать избыток охлаждающей воды в резервуар с теплой водой, выходящей из конденсатора, а затем эту смесь уже подавать на градирню.

Расчет вентиляторной градирни выполнялся для параметров наружного воздуха режимов 1–4, расходах воды $G_{см} = G_в = f(\tau + 5^oC)$ и температуры охлажденной воды, которые приведены в Таб. 1. Результаты расчетов показаны на Рис. 4 (график, обозначенный как $G_в = f(\tau + 5)$ и $\beta=15^o$. Как видно, расчетная высота оросителя меньше, чем стандартная для данной градирни $h_{ор}$ = 2,44 м, т.е. охлаждение воды будет обеспечено в режимах 1–3.

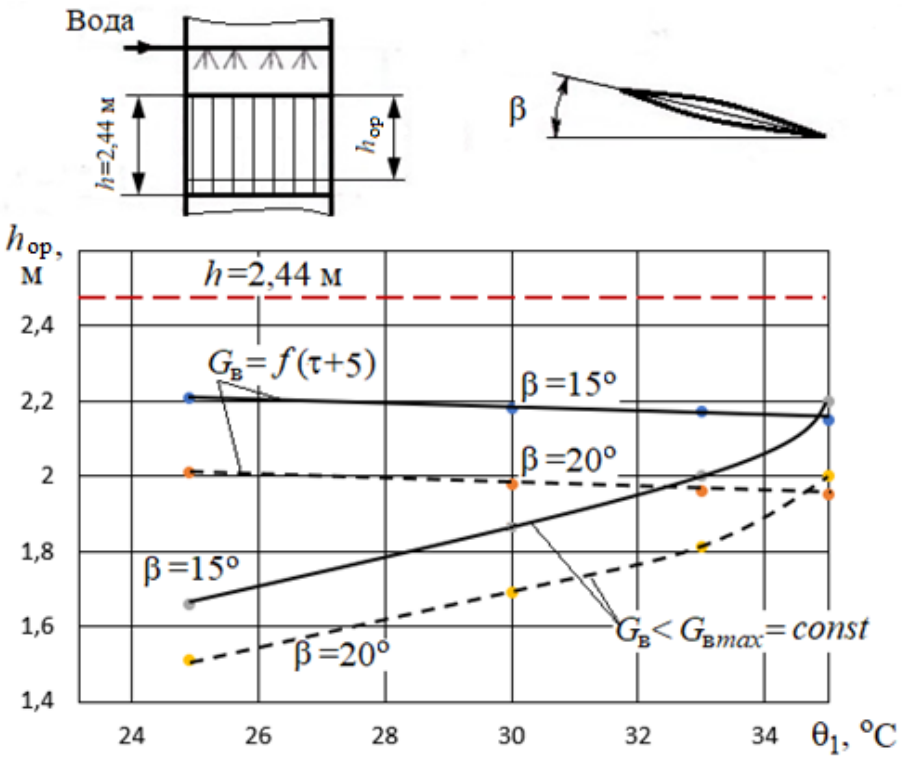


Рисунок 4. Расчетная высота оросителя градирни при повышенных температурах наружного воздуха

3. Интенсификация работы градирни за счет увеличения угла установки лопастей вентилятора

Конструкцией вентилятора 1ВГ25, установленного на градирне, предусмотрена возможность установки лопастей вентилятора на роторе под углами β в 10°, 15° и 20°. Аэродинамические характеристики вентилятора 1ВГ25 [2] получены при оптимальных режимах работы, но действительная подача воздуха зависит от аэродинамического сопротивления на всем пути движения воздуха через градирню. Поэтому согласно методике аэродинамического расчета градирни [2] для данной конструкции оросителя был выполнен расчет суммарного сопротивления при прохождении воздуха через градирню при расходах $G_{вз}$, равных 120; 130; 140; 150 тыс. м^3/ч, и построен график $\Delta P = f(G_{вз})$. Пересечение этого графика с аэродинамическими характеристиками для β=10°; 15°; 20° (рис. 4) показывает

максимальный расход воздуха для данного типа оросителя при разных углах установки лопастей вентилятора.

Проанализируем возможность интенсификации работы градирни при установке лопастей под углом β=20°. По сравнению с β=15° расход воздуха возрастает от 120 тыс. м^3/ч до 143 тыс. м^3/ч. При этом относительный расход воздуха увеличится в λ^m раз, а объем и высота оросителя уменьшаются в 1,1 раза.

Результаты тепловых градирных расчетов на различных режимах относительно параметров наружного воздуха, показанные на рис. 4 (для β = 15° и β = 20°) для стандартных и рекомендуемых режимов работы конденсационно-охлаждающего узла. Из Рис. 4 видно, что с увеличением угла поворота лопастей β увеличивается расход воздуха, а расчетные необходимые объем V и высота оросителя h_{op} уменьшаются (расчетная высота становится меньше стандартной высота оросителя), следовательно, надежность охлаждения воды в градирне возрастает.

Выводы

При повышении температуры наружного воздуха летом днем до 30–35 ºС к эксплуатации конденсационно-охлаждающего узла с контактным барометрическим конденсатором и вентиляторной градирней предъявляются особые требования.

Повышение температуры воздуха уменьшает глубину охлаждения воды в градирне, таким образом, на конденсатор вода поступает с более высокой температурой и ее расход должен возрастать, чтобы обеспечить отвод тепловой нагрузки при конденсации заданного расхода пара. Поэтому на первом этапе в результате гидравлического расчета конденсатора был определен максимальный расход воды, превышение которого значительно ухудшает условия эксплуатации аппарата из-за повышения вероятности гидравлических ударов, вибраций и захлебывания.

На следующем этапе исследования была изучена возможность обеспечения охлаждения воды в градирне при повышенных параметрах наружного воздуха. Необходимо отметить, что при проектировании вентиляторной градирни для барометрического конденсатора, ее площадь в плане и соответственно количество секций выбираются в зависимости от рекомендованной удельной тепловой нагрузки. При таком подходе больший период рабочего времени градирня недогружена (для контактного конденсатора требуется, в сравнении с рекуперативным конденсатором) меньшее количество охлаждающей воды из-за практического отсутствия разности температур конденсации пара и охлаждающей воды на выходе.

Из методики расчета вентиляторных градирен следует, что объемный коэффициент массоотдачи (и, таким образом, интенсивности охлаждения воды) в существенной мере зависит от плотности орошения, другими словами, увеличение расхода воды повышает степень ее охлаждения.

На следующем этапе работы была проанализирована возможность интенсификации градирной работы за счет увеличения угла установки лопастей вентилятора. Для этого предварительно выполнена серия аэродинамических расчетов градирной секции для принятого типоразмера оросителя с целью отметить максимальный расход воздуха при различных углах наклона лопастей. Показано, что поворот лопастей на больший угол приводит к увеличению расхода воздуха, и что повышает охлаждающий потенциал градирни.

В целом, результаты работы могут быть рекомендованы для внедрения в практику эксплуатации конденсационно-охладительных узлов.

Библиография

1. ОСТ 26-01-716-83. Конденсаторы для выпарных аппаратов. Типы, параметры и основные размеры.
2. ПОНОМАРЕНКО, В.С., АРЕФЬЕВ, Ю.И. *Градирни промышленных и энергетических предприятий*. Москва: Энергоатомиздат, 1998.
3. ТРУБ, И.А. *Каскадные конденсаторы смешения*. Москва: Пищевая промышленность, 1969.

PORTOFOLIU STANDARDELOR ENERGIILOR REGENERABILE. PRINCIPII GENERALE

Ruslan BALACCI

Doctorand, D2, Facultatea Energetică și Inginerie Electrică , UTM ,mun. Chișinău, Republica Moldova

*Autorul corespondent: Ruslan Balacci, gembalacci@gmail.com

Îndrumător/coordonator științific: Iurie FRIPTULEAC, inginer, lector universitar UTM

Rezumat. *Lucrarea aceasta este o sineză introductivă pentru cercetarea tezei de doctorat ,cu Titlul "Portofoliu standardelor a energiei regenerabile pentru Republica Moldova". Descrierea performanțelor consumului de energie are ca opțiune de identificare a tehnologiilor performante de energie regenerabile pentru Republica Moldova. Formularea portofoliului de standarde de energie regenerabilă ,va constitui un salt enorm de identificarea tehnologiilor energiei regenerabile pentru implementarea pe teritoriul Republicii Moldova,și definirea direcției Republicii Moldova în privința diversificării surselor de energie. Vor fi expuse în paragraful 1 conceptul esențial a portofoliului de standarde energiei regenerabile. În paragraful 2 va fi expusă bunele practici de implementare a portofoliului de standarde de energie regenerabilă care deja sunt implementate SUA ,și concepte similare în Comunitatea Europeană.*

Cuvinte cheie*: tehnologii, Portofoliu de Standarde a Energiilor Regenerabil, curbă de sarcină, indicatori de performanță..*

Introducere

Pe fonul crizei energetice se discută foarte des despre energie regenerabilă ca posibilă soluție la sporirea securității energetice. Puțin este înțeles că în situația și starea actuală a sistemului electroenergetic energia regenerabilă poate doar să contribuie la reducere din procurările de energie de la CTE Moldovenească (Cuciurgan)/Ucraina într-o măsură moderată indiferent cât de multă energie nu se va produce din surse intermitente (soare și vânt). Deși în realitate este foarte puțin probabil ca să avem vânt și soare concomitentpe întreg teritoriu țării ca să fie acționate toate unitățile de generare intermitente la capacitate maximă totuși surplusurile de capacitate, în special noaptea, sunt destul de semnificative ca să creeze dificultăți serioase cu implicații financiare pe măsură în gestionarea sistemului electroenergetic.

Suplimentar la capacitățile de generare menționate mai sus trebuie luate în considerație și alte capacități ce urmează a fi construite după mecanismul de contorizare netă. Se anticipează ca în prima jumătate a anului 2022 capacitățile fotovoltaice de generare noi construite vor depăși capacitățile totale construite în 2020 și 2021. La fel trebuie considerate capacitățile de generare ce vor fi construite în contextul prevederilor legale actuale (art. 15 din LP 128/2014 privind performanța energetică a clădirilor) potrivit cărora toate clădirile noi construite după 30 iunie 2021 trebuie să aibă un consum de energie aproape egal cu zero iar sistemele fotovoltaice au un rol important în acest sens.

Concluzii apriori

1. Sistemul electroenergetic prin arhitectura sa actuală și profilul consumului de energie nu permite asimilarea volumelor mari de energie regenerabilă din surse intermitente (soare și vânt);
2. Mecanismele de sprijin actuale pentru SER asumate de către guvern în limita capacităților aprobate, fiind valorificate în totalitate, vor ajunge la un nivel de saturație cu energie regenerabilă din surse intermitente (eolian și solar) în anul 2025;

3. Pentru a putea instala capacități suplimentare de generare a energiei din surse regenerabile intermitente este necesar de implementat politici de atenuare a curbei de sarcină (ex. tariful diferențiat pe ore de consum);
4. Atenție prioritară trebuie acordată capacităților de generare pentru balansare, în special surselor de energie regenerabilă neintermitente;
5. Trebuie păstrată rezerva de capacitate fotovoltaică pentru mecanismul de contorizare netă destinată următoarelor categorii de consumatori:
   (i) consumatorii casnici,
   (ii) clădirile publice (conform recentei inițiativei a UE potrivit căreia toate clădirile publice trebuie să dispună de centrale fotovoltaice) .
   (iii) clădirile noi ce urmează să respecte principiul consumului aproape egal cu zero în conformitate cu prevederile legii 128/2014 [2].

Acesta este un punct de analiză ce se petrece la volumul de consum pe curba de sarcină. Da este foarte important de sintetizat indicatorii esențiali de avut stâlpi necesari pentru pornirea ,și identificării surselor de energie și concomitent cu dezvoltarea tehnologiilor inovative de implementare specific relevante pentru Republica Moldova. Putem identifica încă un indicator simplu de consum al energiei, per familie repartizarea consumului pe perioada de iarnă anul curent 2022-2023.Deci avem un consum lunar de gaz natural per luna de 7 000 Lei cu restanța din sezonul de iarnă, curent electric mediu lunar de 350 kWh la 1200 Lei, apă potabile mediu lunar estede 200 Lei, și combustibil pentru automobil mediu 1 500 Lei lunar. Fig.1

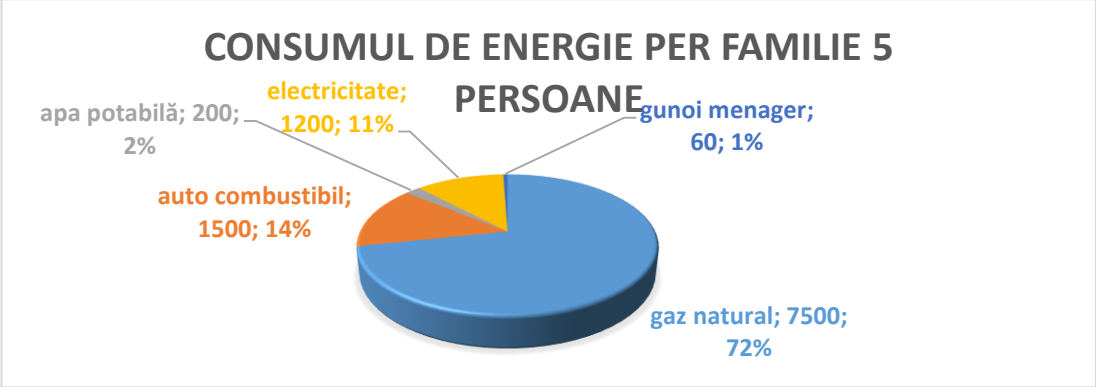


Figura 1 - Repartizarea bugetului de consum a energie pentru o familie de 5 oameni.

Cum vedem se păstrează tendința de consum a energie ,deci avem 5 indicatori pentru care trebuiesa optăm pentru identificarea tehnologiilor de eligibilitate pentru portofoliu standardelor de energie regenerabile pentru Republica Moldova.

Principii Generale, Portofoliu standardelor de energie Regenerabilă.

Un standard de portofoliu de surse regenerabile (RPS) este un mandat de reglementare pentru a crește producția de energie din surse regenerabile, cum ar fi eolian, solar, biomasă și alte alternative la generarea electrică fosilă și nucleară. Este cunoscut și ca standard de energie regenerabilă.
Un RPS are cel mai mare succes în a conduce proiecte de energie regenerabilă atunci când este combinat cu creditul fiscal federal pentru producție. Statele le proiectează adesea pentru a conduceo anumită tehnologie, oferind prevederi „exclude" care impun ca un anumit procent din electricitatea generată să provină dintr-o anumită tehnologie (de exemplu, solar sau biomasă). Statele pot alege să aplice cerința RPS tuturor utilităților sale sau numai utilităților deținute de investitor. Statele pot defini, de asemenea, ce tehnologii sunt eligibile pentru a fi luate în considerare pentru cerințele RPS. A avea o capacitate de transport adecvată pentru a găzdui generarea din resurse regenerabile este importantă pentru succesul unui RPS. Statele cu RPS de succes fie au o transmisie adecvată disponibilă, fie intenționează să o construiască.

Impactul asupra plătitorilor de tarife al unui RPS poate, de asemenea, să deraieze adoptarea acestuia din punct de vedere politic. O contrabalansare a impactului asupra contribuabililor este faptul că mandatele RPS stimulează de obicei creșterea economică locală. În cadrul unui RPS bine conceput, costurile sunt împărțite în mod echitabil de către toți contribuabilii. O altă modalitate dea aborda impactul asupra plătitorilor este de a include prevederi în RPS pentru a preveni escaladarea excesivă a costurilor. Când proiectați un RPS, includeți următoarele bune practici:

a. Țintele RPS ar trebui să fie stabile, să crească constant în timp și să nu fie supuse schimbărilor bruște sau incerte.
b. Un program RPS ar trebui să aibă o durată suficientă pentru a permite contractarea și finanțarea pe termen lung.
c. Un program RPS ar trebui să se aplice tuturor entităților care deservesc încărcătura: deținute de investitori, cooperative municipale și electrice, inclusiv furnizorii de ultimă instanță.
d. Eligibilitatea anumitor tehnologii și generatori de energie regenerabilă ar trebui bine definite.
e. Utilizarea creditelor de energie regenerabilă comercializabile pentru conformitatea cu RPS ar trebui luată în considerare și respectată cu un sistem robust de urmărire.
f. Costul conformității RPS ar trebui să fie alocat în mod echitabil între toți clienții de utilități.
g. Un program RPS ar trebui să fie obligatoriu și să impună sancțiuni de neconformitate acelor entități care nu îndeplinesc cerințele.

O perspectivă generală importantă sugerată de rezultatele inițiale este că un RPS nu este un sistemindependent de politică. Experiența statului arată că rezultatele depind, în special, de politici complementare cele referitoare la transmitere. Statele cu un RPS care au crescut semnificativ utilizarea resursele regenerabile fie au transmisie disponibilitatea, fie au dezvoltat strategii pentrua le construi.

În mod similar, unele state care nu au crescut semnificativ capacitatea de energie regenerabilă, în ciuda faptului că au un RPS de câțiva ani s-a identificat o transmisie inadecvată ca fiind una dintre principalele contribuții de factori de nerealizare [5w].

Exemple de bune practici de portofoliu standardelor de energie regenerabilă

Colaborarea standard de portofoliu de surse regenerabile

Colaborarea a oferit webinarii frecvente, a produs numeroase rapoarte, a oferit asistență tehnică programelor RPS de stat și a găzduit un summit anual RPS la care au participat majoritatea statelorcu programe RPS, precum și alte părți interesate. The Collaborative a publicat, de asemenea, un buletin informativ lunar gratuit; numerele anterioare ale buletinului informativ pot fi vizualizate în arhivele Buletinului informativ RPS . Materialele de la summiturile anuale RPS din 2010 pânăîn 2017 sunt disponibile în arhivele RPS Summit .

Începând din martie 2020, CESA se bazează pe experiența de gestionare a RPS Collaborative pentru a crea și gestiona 100% Clean Energy Collaborative pentru statele care au stabilit obiectiveși mandate 100% pentru energie curată și pentru acele state care iau în considerare dezvoltarea inițiative similare[5w]. Multe dintre inițiativele de a muta sistemul nostru energetic către energie curată și regenerabilă provin de la nivel statal și local. Câteva orașe au atins deja pragul de energiecurată 100%, multe altele s-au angajat să îndeplinească în cele din urmă acest standard. Între timp,politicile și programele de stat au fost esențiale în promovarea revoluției energiei regenerabile. Dela adoptarea standardului de portofoliu regenerabil RPS și a politicilor de stocare a energiei pânăla sprijinirea finanțării energiei verzi și a programelor de incubare de afaceri, statele au și vor continua să joace un rol cheie în conducerea tranziției către energie curată. Iată cinci moduri în care fac acest lucru:

Stabilirea unor standarde puternice

California este un lider incontestabil când vine vorba de energie curată, iar politica sa RPS nu face excepție. Le cere furnizorilor de energie electrică să-și aprovizioneze 50% din energie din energiecurată și regenerabilă până în 2030. Deci, în doar 13 ani, jumătate din toată energia electrică a statului va trebui să provină de la soare, vânt și alte resurse regenerabile. Potrivit Comisiei pentru Energie din California , statul este pe cale de a-și îndeplini RPS 2030, cu aproximativ 27% din energia electrică vândută cu amănuntul în 2016 generată din surse regenerabile. În prezent există un proiect de lege în legislatura statului, deja adoptat de Senat, care cere energie 100% curată pentru electricitate până în 2045. Câteva alte state s-au angajat, de asemenea, la obiective ambițioase. New York a adoptat aceeași țintă ca și California pentru Standardul său de energie curată, 50% până în 2030. În conformitate cu acest obiectiv, statul a anunțat recent o inițiativă de a investi până la 1,5 miliarde de dolari în proiecte majore de energie regenerabilă, cea mai mare achiziție de energie curată de către oricaredintre acestea. stare până în prezent. Vermont are obiectivul de a atinge 90% energie regenerabilăpână în 2050. Hawaii a mers și mai departe cu standardul său , solicitând ca 100% din electricitatesă provină din surse regenerabile până în 2045 [2w].

Oferirea de finanțare inovatoare

Connecticut a găsit o modalitate de a face finanțarea implementării energiei curate mai accesibilăși mai accesibilă pentru consumatori și întreprinderi. În 2011, legislatura statului a creat Connecticut Green Bank , prima bancă verde a națiunii. Utilizează fonduri publice pentru a atrage investiții de capital privat în proiecte de energie verde. Prin mobilizarea investițiilor private, Green Bank crește semnificativ suma totală de finanțare disponibilă pentru proiectele de energie curată. Connecticut Green Bank a sprijinit în total peste 1 miliard de dolari în investiții în energie curată până în prezent. Acest lucru se traduce printr-o capacitate suplimentară de energie regenerabilă de 200 MW. Printre cele mai de succes inițiative ale Băncii Verzi se numără programul Commercial Property Assessed Clean Energy (C-PACE), care permite proprietarilor de proprietăți comercialesă plătească pentru energie curată sau îmbunătățiri ale eficienței în timp prin impozitele pe proprietate. După cum se precizează pe site-ul CT Green Bank, „Modelul funcționează. Implementăm mai multă energie curată mai rapid și mai eficient decât oricând, utilizând în același timp în mod optim finanțarea publică.”[3w]

Sprijinirea afacerilor cu energie curată

Industria energiei curate a cunoscut o creștere semnificativă în Massachusetts, extinzându-se cu 75% între 2010 și 2016. O parte din acest succes se datorează programelor de incubatoare și stagiisusținute de Centrul pentru Energie Curată din Massachusetts. Prin programul Incubate Mass de laMassCEC , start-up-urile cu energie curată au putut beneficia de sprijinul în afaceri și serviciile dementorat ale facilităților incubatoarelor. Finanțarea de către MassCEC a acestor facilități sprijină în mod direct companiile de tehnologie curată și energie curată în stadiu incipient, oferindu-le acces la spații de laborator și birouri, echipamente specializate și alte resurse. Companiile de energie curată beneficiază, de asemenea, de accentul pus de MassCEC pe dezvoltarea forței de muncă. Programul de stagiu pentru energie curată plasează sute de studenți și proaspeți absolvenți cu companii de energie curată participante din Commonwealth. Tinerii cursanți primesc o experiență de lucru practică, valoroasă și plătită, posibilă prin finanțarea oferită de MassCEC. Companiile gazdă, la rândul lor, primesc un impuls pentru forța de muncă. Mulți dintre stagiari continuă să obțină un loc de muncă cu normă întreagă la companiile lor gazdă sau în altă parte a industriei.

De la lansarea programului în 2011, MassCEC raportează că a plasat 1.805 stagiari la 317 companii. New York este un alt stat care a stabilit un program puternic de incubatoare pentru afacerile cu energie curată. Autoritatea de Cercetare și Dezvoltare în Energetică a Statului New York (NYSERDA) finanțează șase incubatoare din stat. Fiecare este afiliat la o universitate de stat. Texas are, de asemenea, un incubator afiliat la universitate, Austin Technology Incubator de la Universitatea Texas din Austin. Incubatorul de energie curată al ATI a fost lansat în 2001 și rămâne unul dintre cele mai vechi incubatoare de energie și tehnologie curată din țară [3-4w].

Transformarea și modernizarea rețelei electrice

New York a întreprins una dintre cele mai îndrăznețe și mai cuprinzătoare inițiative pentru a-și revizui piețele și infrastructura de energie electrică pentru a sprijini creșterea implementării energiei regenerabile. Lansat în 2014, Reforming the Energy Vision (REV)este strategia guvernatorului Andrew Cuomo de a construi un sistem energetic mai curat, mai accesibil și mai rezistent. REV urmărește să modifice modelul tradițional de afaceri de utilități și rețeaua electrică centralizată pentru a găzdui o generație mai distribuită, a sprijini inovația în tehnologia de stocareși a rețelelor inteligente și pentru a accelera măsurile de eficiență energetică și energie curată. Ceitrei piloni de bază ai strategiei REV sunt reforma de reglementare condusă de Comisia pentru Servicii Publice a statului, activarea pieței condusă de NYSERDA și conducerea prin exemplul în implementarea soluțiilor de energie curată conduse de New York Power Authority. Împreună, cele peste 40 de inițiative ale REV reprezintă un angajament de a transforma complet sistemul energetical statului.

Alte state lucrează activ la reforme și modernizări ale rețelei electrice. Conform raportului „50 destate de modernizare a rețelei" al Centrului de tehnologie pentru energie curată din Carolina de Nord , peste jumătate dintre state sunt în prezent implicate în activități de modernizare a rețelei. California a fost statul de top în 2016 Grid Modernization Index, care urmărește progresul statuluiîn modernizarea rețelei electrice și acordă puncte pe baza planurilor și politicilor de stat, a niveluluide implicare a clienților și a implementării tehnologiilor de modernizare a rețelei. Illinois a primitlocul doi în clasamentul general și cel mai bun punctaj în ceea ce privește sprijinul de stat (în 2011, legislatura statului a adoptat Legea de modernizare a infrastructurii energetice). Texas a urmat pe locul trei și a marcat primul la categoria operațiuni pe grilă. Maryland și Delaware au completat primele cinci[5w].

Obligarea stocării energiei

În 2015, Oregon a devenit al doilea stat care a adoptat o lege care stabilește un obiectiv pentru dezvoltarea stocării de energie – o piesă cheie a puzzle-ului atunci când vine vorba de dezvoltarea unui sistem electric cu o penetrare foarte mare a resurselor de energie regenerabilă intermitentă,cum ar fi solară și eoliană. Legea de stocare din Oregon impune utilităților să achiziționeze până în 2020 sisteme de stocare a energiei capabile să stocheze cel puțin 5 MWh de energie și îndrumă Comisia de utilități publice să emită linii directoare pentru această achiziție. În plus, legea ordonă Comisiei să examineze valoarea potențială a stocării, inclusiv ca o completare a altor resurse. California a adoptat primul mandat de stocare a energiei în 2013, necesitând 1,3 gigawați de stocare până în 2020 . Massachusetts a adoptat o lege privind diversitatea energetică în 2016, care a îndrumat Departamentul de Resurse Energetice (DOER) să evalueze dacă să urmărească obiectivele de stocare a energiei, iar DOER avansează acum cu obiectivele obligatorii de stocare. Nevada a adoptat recent o legislație care cere Comisiei de utilități publice să ia în considerare obligarea achizițiilor de depozitare. Comisia are până la 1 octombrie 2018 să decidă în acest sens. În plus, guvernatorul Nevada, Brian Sandoval, a semnat o măsură prin care se stabilește un program de stimulare pentru stocarea energiei solare. Acestea sunt doar câteva dintre multele moduri prin care statele urmăresc un viitor cu energie curată. Ne putem aștepta la progrese și inovații continue în anii următori [4-5w]

Concluzii

1) Din simplele considerente noi suntem la un punct de cotitură pentru resetarea tipurilor de energie regenerabilă. Vedem o experiență și identificare de bune practici la multe state dinlume ,care au deja o reușită.
2) Este important de identificat resursele noastre a Republicii Moldova ,cu promovarea tehnologiilor proprii viabile,promovarea și finanțarea proiectelor de tehnologii a energii regenerabile autohtone.

3) Cu atât mai mult că avem studii complexe a oamenilor de știință din Republica Moldova care au studiat ,și avem identificat tot arsenalul de resurse de energie regenerabile pe teritoriul Republicii Moldova. Putem vedea in bibliografie.
4) Și în final este foarte important de studiat procese și tehnologii complexe ,care vor avea impact pozitiv asupra economiei Republici Moldova.

Referinţe

1. I. BOSTAN, V. DULGHERU, I. SOBOR, V. BOSTAN, A.SOCHIREAN, Sisteme de conversie a energiilor regenerabile Univ. Tehn. a Moldovei. – Ch.: „Tehnica - Info”, 2007, - 592 p. ISBN978-995-63-076-4.
2. I. SOBOR, D. CARAGHEAUR, Ş . NOSADZE; Surse regenerabile de energie: Curs de prelegeri Min.Educaţiei şi Tineretului, Univ. Tehnică a Moldovei- Chişinău; UTM, 2006.- 380 p.ISBN 978-9975-45-020-1.
3. T. AMBROS, V. ARION, A.GUŢU, I.SOBOR, T.TODOS, D.UNGUREANU. Surse regenerabile de energie.–Manual, Chişinău, Editura „Tehnica-Info”, 1999.-434 p. ISBN9975-910-79-3.
4. I. SOBOR, A. CHICIUC,V. RACHIER; Atlasul Resurselor Energetice Eoliene al Republicii Moldova, Univ. Tehn. a Moldovei, AWS Truepower UL (Spania), Wind Power Energy SRL (România).- Chișinău: 2017. - 176 p. ISBN 978-9975-87-215-7.
5. P. TODOS, I. SOBOR, D. UNGUREANU, A. CHICIUC, M. PLEȘ ca. Energia regenerabilă. Studiu de fezabilitate. Ch.: Min. Ecologiei, Conctr. și Dezv. Teritor. UNDP Moldova, Chişinău,2002. – 157 p. ISBN 9975-9581-4-1.
6. Renewable Energy. Power for a Sustainable Future. Second Edition. Editor Godfrey Boyle. Oxford University Press. – 2004, 452 p. ISBN 0-19-926178-4.
7. Agenția Europeană de Mediu .https://www.eea.europa.eu/ro/themes/climate/about-climate-change.
8. Ion Muntean. Publicații Fecebook https://www.facebook.com/ion.muntean.756.
9. Gas supply status report .As at: 1pm, 26 August 2022 https://www.bundesnetzagentur.de/EN/Areas/Energy/Companies/SecurityOfSupply/GasSupply/Downloads/08-Aug_22/20220826.pdf?__blob=publicationFile&v=3.
10. RENEWABLES 2021 GLOBAL STATUS REPORThttps://www.google.com/search?q=RENEWABLES+2021+GLOBAL+STATUS+REPORT&rlz=1C1CHZN_ruMD948MD948&oq=RENEWABLES+2021+GLOBAL+STATUS+REPORT&aqs=chrome..69i57j0i22i30l4.1335j0j15&sourceid=chrome&ie=UTF-8.
11. NREL Transforming Energy https://www.nrel.gov/state-local-tribal/basics-portfolio-standards.html.
12. BerLab https://emp.lbl.gov/publications/us-renewables-portfolio-standards-3. CESA https://www.cesa.org/projects/renewable-portfolio-standards/.
13. PL Nr. 128 din 11-07-2014 privind performanţa energetică a clădirilor. https://www.legis.md/cautare/getResults?doc_id=95262&lang=ro

SUSTENABILITATEA UTILIZĂRII LITIULUI PENTRU BATERII CU ACUMULATOARE DE ENERGIE ELECTRICĂ

Marcel ROTARU

Departamentul Energetică, grupa IME-201, FEIE, UTM, Chișinău, Moldova

*Autorul corespondent: Rotaru Marcel, marcel.rotaru@en.utm.md

Îndrumător: Dumitru BRAGA, dr. lector univ., Facultatea de Energetică și Inginerie Electrică, Universitatea Tehnică a Moldovei, Chișinău, Moldova

Rezumat. *Producția bateriilor cu acumulatoare cu ioni de litiu este esențială pentru dezvoltarea vehiculelor electrice și a altor industrii, cum ar fi echipament electronic de larg consum sau integrarea resurselor regenerabile de energie. Rezervele de litiu sunt limitate și cele mai mari rezerve sunt situate în America de Sud, Australia și China. Cele două metode principale de extracție a litiului sunt din bazinele de saramură și exploatarea rocilor dure. Deși există rezerve semnificative de litiu la nivel global, există îngrijorări cu privire la capacitatea industriei globale de litiu de a satisface cererea tot mai mare de baterii cu acumulatoare cu litiu-ion. În plus, există îngrijorări cu privire la dificultatea de a accesa unele rezerve de litiu, impactul de mediu al industriei de extracție a litiului și factorii politici și economici din țările de extracție a litiului.*

Cuvinte cheie*: baterii cu acumulatoare cu ioni de litiu, vehicule electrice, rezerve de litiu, resurse regenerabile de energie*

1. Rezervele și producția de litiu

După cum știm, bateriile cu acumulatoare cu ioni de litiu sunt o componentă crucială în dezvoltarea vehiculelor electrice, iar producția lor este de așteptat să crească substanțial în următorii ani, pe măsură ce tot mai multe țări se angajează să-și reducă emisiile de gaze cu efect de seră și să realizeze tranziția către un transport sustenabil. Cu toate acestea, ca și în cazul oricărei resurse naturale, rezervele de litiu sunt limitate și există îngrijorări cu privire la faptul dacă avem suficiente rezerve pentru a satisface cererea viitoare de litiu.

Litiul este un element relativ rar găsit în scoarța terestră, cele mai mari rezerve fiind situate în America de Sud, Australia și China. Potrivit United States Geological Survey (USGS), resursele globale de litiu identificate în 2021 au fost estimate la aproximativ 79 de milioane de tone, Chile, Argentina și Australia reprezentând aproximativ 77% din rezervele mondiale de litiu identificate.[2]

Cele două metode principale de extracție a litiului sunt din bazinele de saramură și exploatarea rocilor dure. Majoritatea producției de litiu provine în prezent din bazine de saramură, ceea ce implică extragerea litiului din lacuri cu apă sărată sau acvifere subterane. Exploatarea rocii dure implică extragerea litiului din zăcămintele de pegmatită, care sunt formațiuni de rocă dure care conțin minerale purtătoare de litiu.

Producția globală de litiu a crescut semnificativ în ultimii ani, crescând de la aproximativ 40.000 de tone în 2005 la peste 82.000 de tone în 2019, potrivit USGS, prezentat în Fig.1.

Această creștere a producției a fost în mare măsură determinată de cererea de baterii cu acumulatoare cu litiu-ion, care sunt utilizate într-o varietate de echipament electronic de larg consum, sisteme de stocare a energiei și vehicule electrice.

Cu toate acestea, în ciuda creșterii producției de litiu, există îngrijorări cu privire la capacitatea industriei globale de litiu de a ține pasul cu cererea tot mai mare de baterii cu acumulatoare cu litiu-ion. Acest lucru se datorează unui număr de factori, inclusiv dificultății de a accesa unele rezerve de litiu, preocupărilor de mediu legate de extracția litiului și factorilor politici și economici din unele țări producătoare de litiu.

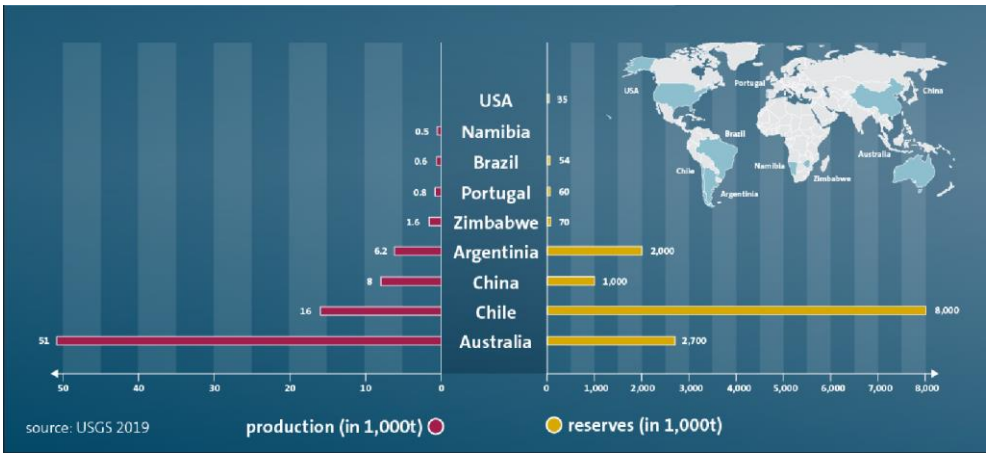


Figura 1. Producția și rezervele globale de litiu în anul 2019

În general, deși există rezerve semnificative de litiu la nivel global, provocarea pentru industrie este să se asigure că nivelurile de producție sunt suficiente pentru a satisface cererea tot mai mare de baterii cu acumulatoare cu litiu-ion, abordând în același timp preocupările de mediu și sociale asociate cu extracția litiului. Cererea de baterii litiu-ion a crescut constant în ultimii ani, determinată în principal de creșterea pieței vehiculelor electrice (EV). Bateriile cu acumulatoare cu litiu-ion sunt tehnologia preferată a bateriilor pentru vehiculele electrice datorită densității mari de energie, duratei de viață lungi și capacității de a fi reîncărcate rapid.

În 2020, piața globală de vehicule electrice a atins o nouă etapă, cu peste trei milioane de vehicule electrice noi vândute în întreaga lume, reprezentând o creștere de 43% față de anul precedent. Se așteaptă că această creștere va continua, previziunile sugerând că piața globală de vehicule electrice va ajunge la peste 145 de milioane de vehicule până în 2030, după cum este arătat în Fig.2.

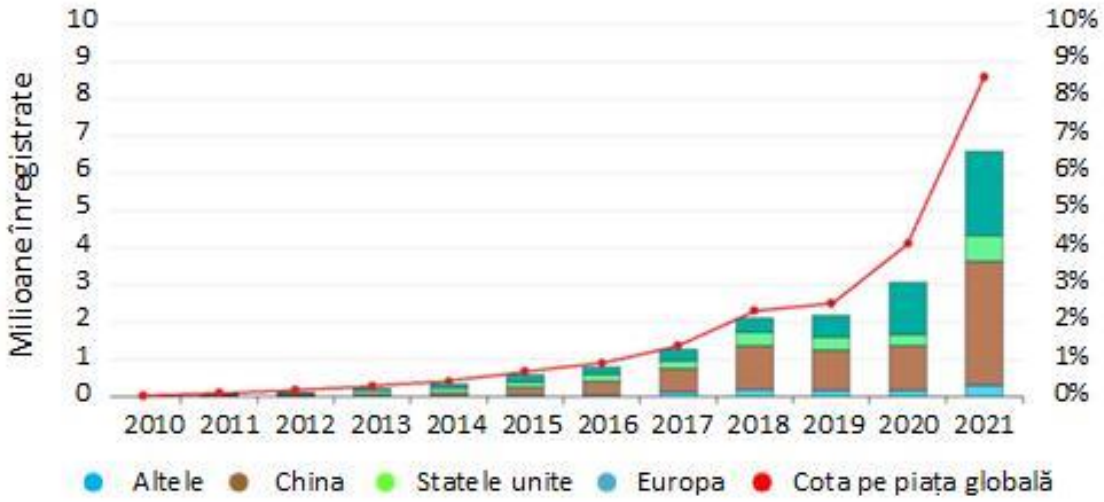


Figura 2. Vehiculele electrice vândute în ultimii ani

Pe lângă piața de vehicule electrice, cererea de baterii litiu-ion este susținută și de alte industrii, cum ar fi electronicele de larg consum, sistemele de stocare a energiei și energia regenerabilă. De exemplu, bateriile litiu-ion sunt folosite în smartphone-uri, laptop-uri și alte dispozitive electronice portabile, precum și în sistemele staționare de stocare a energiei pentru case și afaceri. Potrivit unui raport al McKinsey & Company, cererea de litiu este de așteptat să crească cu cel puțin 20% pe an până în 2030, determinată în primul rând de piața EV.[2]

În timp ce cererea de baterii litiu-ion crește rapid, există o serie de provocări din partea ofertei pe care industria litiului trebuie să le abordeze pentru a asigura o aprovizionare constantă cu litiu în viitor.

2. Preocupări de mediu

Extracția litiului este esențială pentru producerea bateriilor reîncărcabile și pentru a susține industria automobilelor electrice și a energiei din surse regenerabile. Cu toate acestea, această activitate poate avea un impact semnificativ asupra mediului înconjurător.

Una dintre cele mai comune metode de extracție a litiului este prin folosirea bazine cu saramură, care implică pomparea apei subterane și tratarea ei cu substanțe chimice pentru a separa litiul de alte minerale. Acest proces poate duce la epuizarea resurselor de apă dulce, poluarea cu substanțe chimice toxice și modificarea pH-ului solului, ceea ce poate afecta animalele și plantele din ecosistemul local. În plus, exploatarea rocilor dure poate provoca eroziunea solului și alte daune mediului. În acest caz, mineralele sunt extrase prin excavare, ceea ce duce la îndepărtarea solului și a rocilor. Această activitate poate provoca defrișarea pădurilor, degradarea habitatelor sălbatice și pierderea biodiversității.

Pentru a răspunde acestor preocupări, este nevoie ca industria litiului să dezvolte metode mai durabile de extracție care să minimizeze impactul asupra mediului. Una dintre aceste metode este reprezentată de extracția litiului din apă de mare. Această metodă implică extragerea litiului din apele oceanice, ceea ce poate reduce impactul asupra ecosistemelor terestre.

În plus, cercetătorii încearcă să găsească modalități de a recicla mai multe baterii și de a reduce dependența de noi extracții de litiu. Aceste eforturi ar putea ajuta la reducerea cererii de noi extracții de litiu și, prin urmare, la reducerea impactului asupra mediului.

3. Limitări tehnologice

Cu cererea tot mai mare de litiu, este important să găsim noi modalități de a extrage litiul din rezervele existente și să identificăm noi surse de litiu. Tehnologiile actuale de extracție a litiului, cum ar fi folosirea bazinelor cu saramură sau exploatarea rocilor dure, sunt eficiente, dar au și limitări care le fac dificil de utilizat în anumite cazuri.

De exemplu, unele rezerve de litiu pot fi situate în locuri greu accesibile, cum ar fi zonele montane sau regiunile polare, ceea ce poate face dificilă utilizarea tehnologiilor de extracție tradiționale. De asemenea, unele surse de litiu se găsesc în acvifere subterane, care pot fi greu de accesat și de controlat, ceea ce poate duce la poluarea apei și la deteriorarea mediului înconjurător.

Pentru a accesa aceste rezerve de litiu, este nevoie de noi tehnologii de extracție care să poată lucra în astfel de condiții. Unele dintre aceste tehnologii includ folosirea nanotehnologiei și a inteligenței artificiale pentru a identifica și extrage litiul din sursele mai dificile. De exemplu, cercetătorii dezvoltă nanosondelor care pot fi injectate în sol sau în apă subterană și care pot detecta prezența de litiu, ceea ce ar putea facilita extracția.

În plus, există cercetări care încearcă să găsească noi surse de litiu, cum ar fi sursele de litiu din apele oceanice sau din alte materiale cum ar fi argila sau alte minerale. Dezvoltarea unor tehnologii mai eficiente de extracție a litiului din aceste surse ar putea ajuta la reducerea impactului asupra mediului și la creșterea accesibilității litiului.

- **Factori politici și economici**

Producția de litiu este concentrată în câteva țări, în special în Chile, Argentina și Australia. Această concentrare poate crea riscuri geopolitice, deoarece țările încearcă să-și protejeze propriile interese și să controleze piața globală a litiului. În plus, există îngrijorări cu privire la impactul economic al producției de litiu asupra comunităților locale, în special în țările în curs de dezvoltare.

Pentru a aborda aceste provocări, industria litiului va trebui să dezvolte strategii pentru a atenua riscurile geopolitice și pentru a se asigura că comunitățile locale beneficiază de la producția de litiu. Pe măsură ce industria litiului se confruntă cu provocări din partea ofertei, există o serie de soluții alternative care ar putea ajuta la abordarea cererii în creștere pentru baterii litiu-ion.

- **Reciclare**

O soluție potențială este creșterea reciclării bateriilor litiu-ion. În prezent, doar un mic procent din bateriile litiu-ion sunt reciclate, iar acest lucru prezintă o oportunitate de a recupera resurse valoroase de litiu din bateriile uzate. Prin dezvoltarea unor tehnologii de reciclare mai eficiente și mai rentabile, industria ar putea reduce cererea de producție nouă de litiu și ar putea prelungi durata de viață a resurselor existente de litiu.

- **Tehnologii alternative pentru baterii**

O altă soluție potențială este dezvoltarea unor tehnologii alternative de baterii care nu necesită litiu. De exemplu, bateriile cu stare solidă, care folosesc electroliți solizi în loc de electroliți lichizi, sunt o alternativă promițătoare la bateriile litiu-ion. Aceste baterii au potențialul de a oferi o densitate de energie mai mare și timpi de încărcare mai rapizi, folosind în același timp materiale mai abundente și mai puțin sensibile pentru mediu.

- **Explorarea pentru noi surse de litiu**

În cele din urmă, explorarea de noi surse de litiu ar putea ajuta la extinderea rezervelor de litiu disponibile. De exemplu, unii cercetători au sugerat că litiul ar putea fi extras din apa de mare, ceea ce ar crește semnificativ aprovizionarea globală cu litiu. Deși această tehnologie este încă în stadiu experimental, reprezintă o soluție potențial promițătoare la cererea în creștere pentru litiu.

4. Concluzie

În concluzie, pe măsură ce cererea de baterii litiu-ion continuă să crească, industria de litiu va trebui să exploreze soluții alternative pentru a asigura o aprovizionare constantă cu litiu. În ciuda acestor provocări, cererea în creștere pentru vehicule electrice și alte industrii care se bazează pe bateriile litiu-ion reprezintă o oportunitate semnificativă la sprijinirea tranziției către un viitor verde.

Referinţe

1. United States Geological Survey, Disponibil: https://www.volkswagenag.com/en/news/stories/2020/03/lithium-mining-what-you-should-know-about-the-contentious-issue.html
2. Teague Egan, WILL WE RUN OUT OF LITHIUM? Disponibil: https://energyx.com/blog/will-we-run-out-of-lithium/

MODELING ELEMENTS of A BASIC CLOUD SUBSTATION

Vladyslav TSIUPA[1*], Andrii POTRYVAI[1]

[1]*IEEE institute, PhD. student of Department of Electric Power Transmission, NTU "KhPI", Kharkiv, Ukraine*

*Corresponding author: Vladyslav Tsiupa, vladyslav.tsiupa@ieee.khpi.edu.ua

Tutor/coordinator: Veronika CHERKASHYNA, professor, IEEE institute, Department of Electric Power Transmission, NTU "KhPI", Kharkiv, Ukraine**, Dmytro DANYLCHENKO** , docent of IEEE institute, Department of Electric Power Transmission, NTU "KhPI", Kharkiv, Ukraine

Abstract. *In this paper, the main requirements were considered that will allow us to create a complete model of a cloud substation in the future. The mathematical models on the basis of which the basic elements of a cloud substation will be built in the future are described.*

Keywords: *calculation method, WLAN, breaker, remote protection.*

Introduction

In modern design practice, the decision to use digital substations for different voltage classes is becoming more widespread. "Digital Substation" (DSP) is a term applied to substations controlled by intelligent electronic devices (IEDs) linked by a communications network [1]. However, the development of cloud technologies, the emergence of new ways of manipulating the amount of information and increasing the speed of data transfer using the Internet opens up new opportunities for automating the management of electric power facilities.

The idea of a cloud substation (CPS) is based on two requirements. The first is the transfer of all logical processes to the central server, which, based on the data received in real time (on-time), forms a command and sends it to the operational bodies of the PS. The second is getting rid of the overall equipment of substations to reduce its size.

Therefore, the purpose of this work is to determine the work cycle of the OPS, study the features and properties, and determine the laws and models that will make it possible to compile a complete model of the OPS in the Matlab modeling environment.

Initial conditions

To begin with, it is necessary to determine the stages of the work cycle of the OPS. The system interacting with information processing must be able to collect information about the state of the equipment (switch, cubicle door, grounding knives, surge arresters, etc.). An approximate interaction scheme is shown in Fig.1.

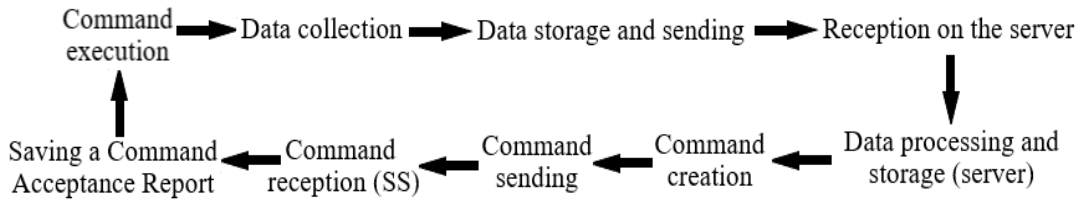


Figure 1. Scheme of the work cycle of the OPS

Intelligent automated processes always start with data collection. Thus, it is necessary, for starters, to determine what data is important for the operation of the OPS and which devices to ensure the collection of this data.

For the operation of the protection of the equipment of the substation, it is necessary to provide the following data: the value of the current and voltage in the phases (possibly the frequency in the system) and the state of the switches. This will allow the data processing program to decide whether to perform switches depending on situations.

To fix the system parameters (current, voltage), it is proposed to use optical current and voltage sensors. This decision was made on the basis that the main parameter in an on-time system is the speed of system responsiveness. Thus, the problem of classical and electronic current transformers is that the processes in them occur at a speed that is calculated by the formula below:

$$T_E = \frac{T_r}{\lambda_R}, where\ \lambda_R = \frac{1}{\sqrt{1 + \frac{k_r^2 R_r}{R_s + R_d}}}$$

This is primarily due to the inertia of the coils and core used in classical current transformers, which under normal conditions are not a problem and do not create a significant delay, but in an on-time system it can significantly affect the speed of operation. The optical system is devoid of these disadvantages, however, it is worth considering the optimal wire length for it, since the length of the path can also affect the response speed of the system.

Method of calculation

Basic elements such as transformer and switches will be provided in building blocks. An example of the simplest implementation is shown in Fig. 2. However, the transformer model will be supplemented with a block that simulates a temperature sensor, which will also send data in real time, since this parameter is also important in making decisions about the execution of operations in the system.

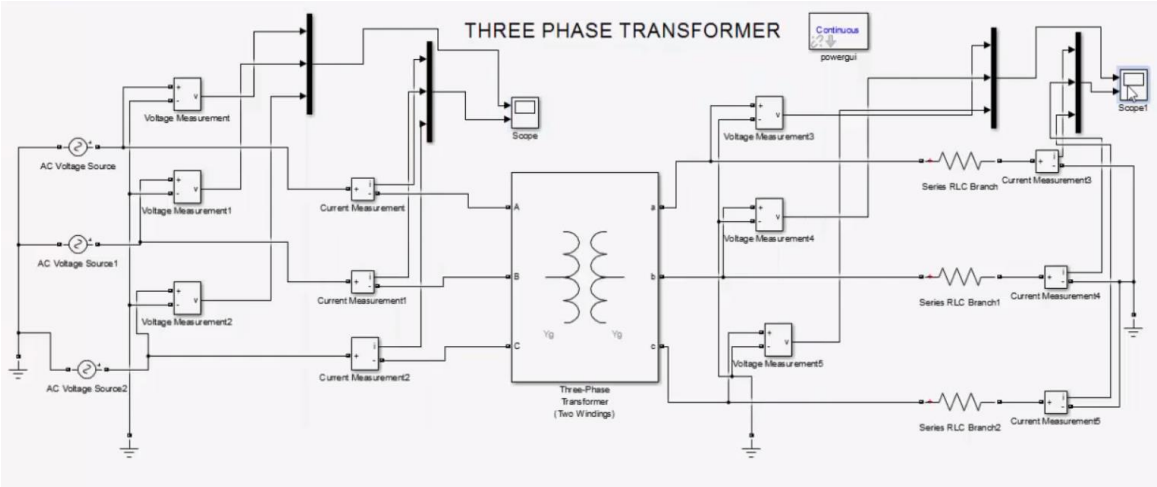


Figure 2. Implementation of the simplest source-transformer-line circuit in *Matlab Simulink*

As mentioned earlier, one of the means of collecting information in the system are optical current and voltage transformers. Their work is based on the Faraday effect. It consists in the rotation of the plane of polarization of electromagnetic radiation (for example, light) propagating in a substance along the lines of force of a constant magnetic field passing through this substance

A magnetized substance in the general case can no longer be characterized by a single refractive index n. The refractive indices n + and n- for radiation of the right and left circular polarizations become different. Linearly polarized radiation passing through an isotropic medium can always be formally represented as a superposition of two waves polarized in the right and left circles with the opposite direction of rotation. The difference between n + and n- leads to the fact that the radiation components polarized in the right and left circles propagate in the medium with different

phase velocities, acquiring a path difference that linearly depends on the optical path length. As a result, the plane of polarization of monochromatic light with a wavelength depends linearly on the strength of the magnetic field H. In the general case, the relation is true:

$$\theta = V \oint \vec{H}\vec{dl} \tag{1}$$

where θ – rotation angle of the polar plane; V - Verde constant; $\vec{H}$ - tension vector magnetic field through light direction; dl - Short length vector faraday material through direction light propagation.

A schematic representation of the fiber optic sensor system and the principle of operation of the sensor itself are shown in Fig.3 a, b.

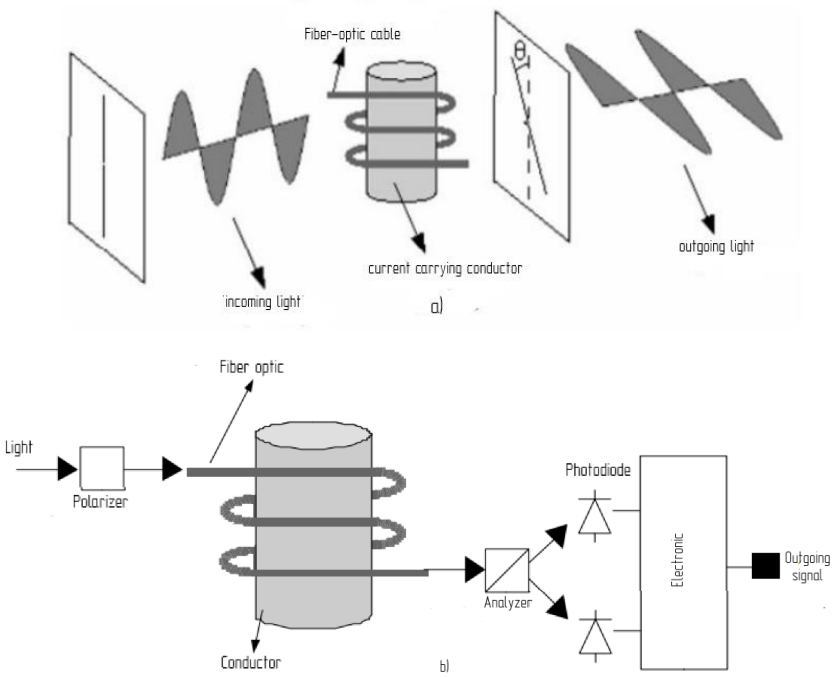


Figure 3. Fiber optic sensor system

However, according to Ampère's law, equation (1) can be reduced to the form

$$\theta = VNI$$

where N - the number of turns of fiber; I - current of the conductor.

A fiber optic sensor in *Matlab* is a complex model consisting of several block models, namely: a laser, a polarizer, a fiber optic sensor, an analyzer, a photodiode, and a computational module.

The laser block model acts as a light source in the system. It has a constant wavelength and is described by the following Ec.:

$$\frac{dN(t)}{dt} = \frac{i(t)}{qV_a} - g_0 \frac{[N(t)-N_0]S(t)}{1-\varepsilon S(t)} - \frac{N(t)}{\tau_n} \tag{2}$$

$$\frac{dS(t)}{dt} = Gg_0 \frac{[N(t)-N_0]S(t)}{1-\varepsilon S(t)} - \frac{S(t)}{\tau_p} + \frac{G\beta}{\tau_n} N(t) \tag{3}$$

$$\frac{d\phi(t)}{dt} = \frac{1}{2}\alpha_L \left[Gg_0[N(t) - N_0] - \frac{1}{\tau_p} \right] \tag{4}$$

where $i(t)$ - laser current; $S(t)$ – photon density; G – optical factor confinement; g_0 – slope gain; $N(t)$ – carrier density; N_0 – carrier density at transparency; ε – saturation parameter; τ_p – photon lifetimeβ - spontaneous emission factor; τ_n – carrier lifetime; V_a – volume of the active region; α_L – line-width enhancement factor; q – electron charge; $\phi(t)$ – phase of the laser electric field.

The output optical power of a laser diode is mathematically expressed as follows [2].

$$P(t) = \frac{V_a \eta h v}{2G\tau_p} \tag{5}$$

where $P(t)$ – line-width enhancement factor; η – quantum efficiency; h – phase of the laser electric field; v – frequency of the laser.

Since laser light is an electromagnetic field, the output power of laser radiation can be expressed as like an electric field [-].

$$E_{laser}(t) = \sqrt{P(t)} e^{i\phi(t)} \tag{6}$$

In the polarizer model shown in Fig.6, the light from the laser is perfectly linearly polarized in the x and y directions. The mathematical expression for this is as follows [3].

$$E_{pol} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} E_{laser} \tag{7}$$

If we neglect the linear birefringence of light inside the fiber, the Faraday effect can be mathematically written as an M-matrix similar to [4].

$$M = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \tag{8}$$

The attenuation of the electric field in the fiber can be expressed as follows, analogue [-].

$$E_{alfa} = E_{pol} \sqrt{10^{-\frac{\alpha_L}{10}}} \tag{9}$$

Since Rayleigh scattering dominates at short wavelengths and IR absorption loss dominates at long wavelengths, the damping factor can be expressed as:

$$\alpha(\lambda) = 7.81 \cdot 10^{11} e^{-\left(\frac{48.48}{\lambda}\right)} + \frac{1}{\lambda^4} \tag{10}$$

The change in the electric field of light due to the Faraday effect is also expressed as follows

$$E_{fiber} = M E_{alfa} \tag{11}$$

The analyzer model ideally divides the electric field of the light coming out of the fiber optic cable into x and y components. The equation below represents the x and y components of the electric field of light:

$$E_{x,y} = E_{laser} \frac{1}{\sqrt{2}} \sqrt{10^{-\frac{\alpha_L}{10}}} (cos\theta \pm sin\theta) \tag{12}$$

The current passing through the photodiode conductor can be found as follows:

$$I = \frac{1}{2VN} \arcsin\left(\frac{I_y - I_x}{I_y + I_x}\right) \tag{13}$$

Switch connection and wireless transmission model

Another important link is the switch. Own design experience allows us to accept the Easypact EXE 10 kV circuit breaker from Schneider Electric as a simulation, as it is quite reliable, its contact diagram is freely available, and it also uses a spring-motor drive, which has proven itself better than electromagnetic . Its contacts make it easy to connect the switch to the I/O module. Thus, using contacts OF1, PF and B1, you can get a reliable picture of the state of the circuit breaker, because they are responsible for the position of the switch, readiness for switching on and the position of the contact for turning on the spring winding motor, respectively Fig. 4.

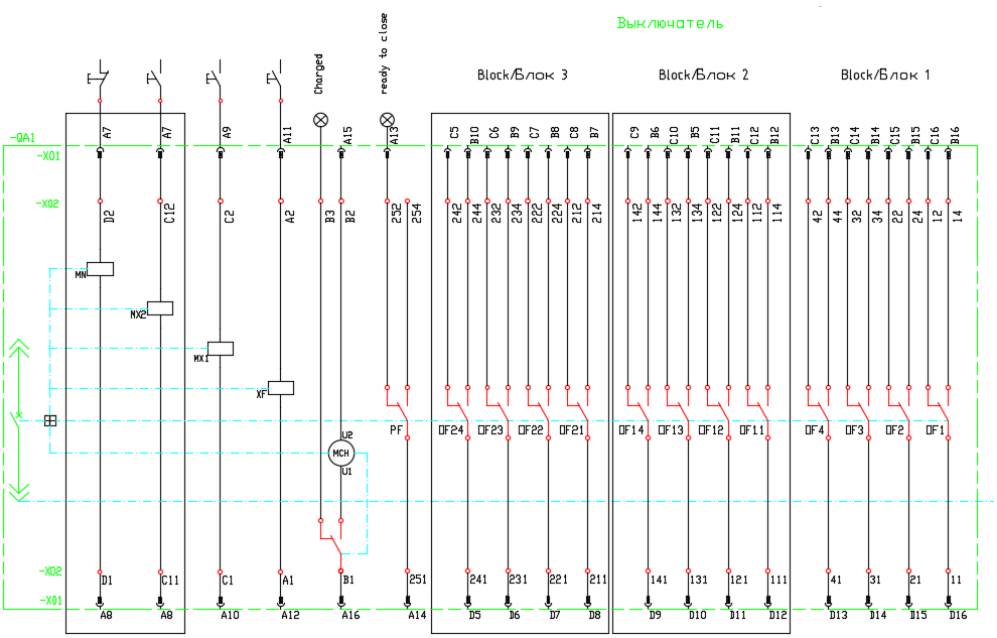


Figure 4. EasyPact EXE Wiring Diagram

In current design practice, for processing relay protection and automation (RPA) assembled from switches and terminals, a processor module is installed between the I/O module and the router, however, in our system there is no certainty that these functions need to be implemented in this way, since the device for converting discrete / analog signals and the formation of data packets for sending can be implemented in a much simpler and more compact design.

The standard functionality of the "switch" block will definitely not be enough, because even the basic model will require the implementation of relay protection in order to be able to manipulate the power switch. And since the idea of the OPS project is to transfer all logical processes to a remote server, it is necessary to model the process of information exchange between the executive bodies and the decision-making center. This requires a model of wireless data transmission.

One of the key questions that will be asked of this model will be the speed of data transfer, because it is necessary to ensure a sufficient speed of issuing commands. This will be achieved by optimizing the size of data packets and researching the fastest ways to transmit them. However, for the base model, it was decided to use the standard 4g network model, which, despite its familiarity, is an extremely difficult model to implement. However, the implementation algorithm is clear and its results are shown in Fig.5.

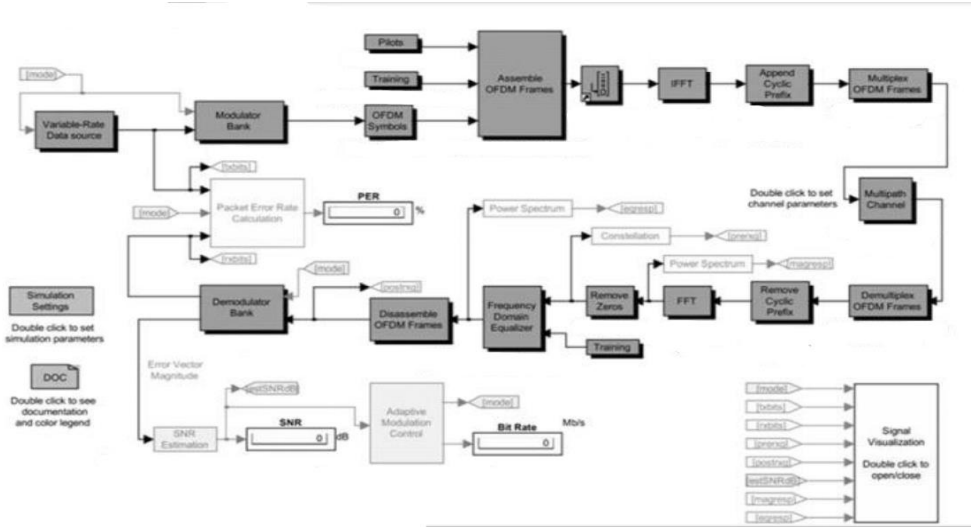


Figure 5. Results of the algorithm implementation

Conclusions

In this paper, the basic for mathematical models of the elements of the basic cloud substation was considered. The basic requirements were determined, which formed the basis for the creation of models and the choice of equipment. The properties of some elements of a cloud substation are considered.

References

1. What is a Digital Substation [Electronic resource] / RZA Project. — Access mode: https://pro-rza.ru/chto-takoe-tsifrovaya-podstantsiya/ (date of access: 02/11/2020).
2. YURIEVICH SHEVCHENKO S., OKUN A.A.: Анализ методов расчета электрических полей установок высоких напряжений [Analysis of methods for calculating the electric fields of high voltage installations]. In: *Электротехника и электромеханика* , 2010, pp.59-62.
3. SCHAER, T., RUSNOV, R., EAGLE, S., JASTREBSKI, J., ALBANESE, S., FERNANDO, X., “A Dynamic Simulation Model for Semiconductor Laser Diodes”, IEEE CCECE Canadian Conference, 293, 297, (2003)..
4. JIAO, B., WANG, Z., LIU, F., BI, W., “Interferometric Fiber-Optic Current Sensor with Phase Conjugate Reflector”, Proceeding of the 2006 IEEE International Conference on Information Acquisition, China, 707, 708, (2006)
5. IOSSEL Yu. Ya., KOCHANOV E. S., STRUNSKY M. G. Calculation of electrical containers. - 2nd ed., revised. and additional - L .: Energoizdat, 1981 .- 240 pp.
6. ÖZKAN, M., “Fiber Optik Kablolar ile İletişim ve Fiber Optik Dalgakılavuzları”, Yüksek lisans tezi, Osmangazi Üniversitesi Fen Bilimleri Enstitüsü, Eskişehir, 10, (1998)
7. SHEVCHENKO S.; DANYLCHENKO D.; DRYVETSKYI S,; POTRYVAI A., Modernization of a Simulation Model of a Photovoltaic Module, by Accounting for the Effect of Snowing of Photovoltaic Panels on System Performance with Correction for Panel Cleaning for Matlab Simulink// 2021 IEEE 2nd KhPI Week on Advanced Technology (KhPIWeek). doi**:** 10.1109/KhPIWeek53812.2021.9570030

IMPLEMENTAREA SOFTULUI PENTRU OPTIMIZAREA PARAMETRILOR TEHNICI A SISTEMELOR ELECTROMECANICE DISTRIBUITE ÎN CADRUL STAȚIEI ORĂȘENEȘTI DE EPURARE A APELOR UZATE

Artiom MOLDOVAN

Universitatea Tehnică a Moldovei, Facultatea Energetică și Inginerie Electrică, Departamentul Inginerie Electrică, Școala Doctorală „Știința Calculatoarelor, Electronică și Energetică”, Chișinău, Republica Moldova

Autorul corespondent: Artiom Moldovan, artiom.moldovan@ie.utm.md

Îndrumător/coordonator științific: Ilie NUCA, conf.univ., Universitatea Tehnică a Moldovei, Facultatea Energetică și Inginerie Electrică, Departamentul Inginerie Electrică,

Rezumat. *În acest proiect, problema a fost dezvoltarea și implementarea softului pentru optimizarea parametrilor tehnici a sistemelor electromecanice distribuite programabile bazate pe soluții tehnico-științifice ce duc la creșterea exponențială a calității și a eficienței energetice.. Problema științifică importantă rezolvată este cercetarea și elaborarea unui soft standart pentru mai multe tipuri de stații de tratare a apei uzate și anume calcularea corectă a parametrilor tehnici în baza celor tehnologici pentru eficientizarea proceselor tehnologice.*

Cuvinte cheie*: stații de tratare a apelor uzate, pompe, motoare electrice, convertoare statice, sisteme electromecanice distribuite, algoritmi de comandă, controlere programabile, sistem SCADA.*

Introducere

Apa uzată sau apa reziduală conține cantități semnificative de poluanți, care duc la epuizarea oxigenului din apă atunci când este deversată direct în apele de suprafață. Compoziția apelor reziduale variază în mare măsură în funcție de zonele de unde sunt colectate. Substanțele poluante care contaminează apa sunt, în general, solide sau compuși biodegradabili sau lent biodegradabili, nutrienți, substanțe toxice, organisme patogene etc[1]. Astfel, o instalație de tratare a apelor reziduale uzuale, așa cum se arată și în Fig. 1, include mai multe procese de tratare în mai multe etape. Fiecare etapă de tratare este proiectată pentru a elimina un anumit tip de substanță poluantă. Obiectivul principal al tratării apelor uzate este de a permite deversarea apelor uzate urbane în apele de suprafață, asigurând, astfel, protecția mediului și a comunității umane[2].

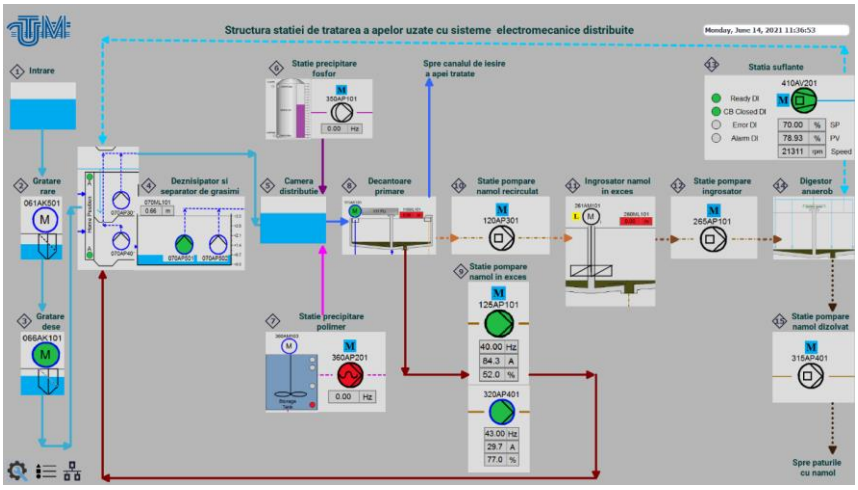


Figura 1. Schema generală a unei stații de tratare [1].

Structura unei stații de tratare a apei uzate este alcătuită din următoarele obiecte, care sunt obligator prezente:

1) Gratare rare si fine – obiectele in care apa trece o tratare mecanica primara, și este despărțită de materialele ce nu pot trece sita gratarelor.

2) Deznisipator și separator de grăsimi – obiectul în care este sedimentat nisipul din apă, și pompat in exterior. La fel tot aici din cauza căderii libere cu un debit mare a apei, este eliminată spuma, care la fel contine murdării.

3) Stațiile de precipitare fosfor și polimer – aceste stații dozeaza cantitatea dorită de fier pentru a elimina fosforul din apă, și cantitatea dorită de polimer pentru a sedimenta mai bine nisipul marunt ăn decantoarele primare.

4) Decantoare primare – locul unde se sedimenteaza particulele ce u au fos sedimentate in deznisipator.

5) Stația de pompare nămol recirculate – nămolul care este sedimentat la fundul decantorului este pompat în îngroșătorul de nămol.

6) Stația de pompare nămol în exces – nămolul amestecat cu apă este repompat in deznisipator, deoarece este o cantitate în exces.

7) Îngroșătorul de nămol – este bazinul în care se amestecă cantitatea de nămol, pentru a ridica viscozitatea, si de a elimina apa din el, să fie mai ușor de dizolvat.

8) Stație de pompare îngroșător – pompeaza nămoul din îngroșător în bazinul de dizolvare.

9) Stația suflante – obiectul cu un număr calculate de suflante, care pompeaza aer în bazinul de aerare a deznisipatorului și în bazinul de dezolvare a nămolului.

10) Digester anaerob – bazinul unde nămolul se dizolvă.

11) Stația de pompare nămol dizolvat – stația care pompeaza nămolul dizolvat spre paturile cu nămol [3].

În funcționarea unei stații de tratare a apelor uzate, principala funcție este de a elimina în mod eficient poluanții din apa uzată sau transformarea lor în compuși mai puțin dăunători, astfel încât efluentul să îndeplinească cerințele de deversarare utilizând tehnologii de tratare adecvate și la cele mai scăzute costuri posibile. Reducerea costului se poate realiza și prin recuperarea unor resurse din proces, cum sunt substanțele nutritive, biogazul și apa pentru reutilizare. Datorită interacțiunilor complexe ale diferitelor variabile legate de funcționarea sistemului de tratare, este necesar să se adoptate strategii de control avansate care să mențină concentrațiile efluentului sub limita legală cu un cost minim.

Analiza unui exemplu de configurare a stației în softul elaborat

În regim de simulare a fost configurată o stație fară optimizare, lipsită de un sistem de automatizare, informativ și de control și o variant de optimizare cu schimbarea sarcinei de lucru, schimbarea timpului de funcționare ce au dus la schimbari de consum de energie, respectiv și preți pentru energie, cazul a fost analizat pe o perioadă de un an.

În Fig. 2 avem datele stației fară optimizare cu consumul sau și cheltuielele respective.

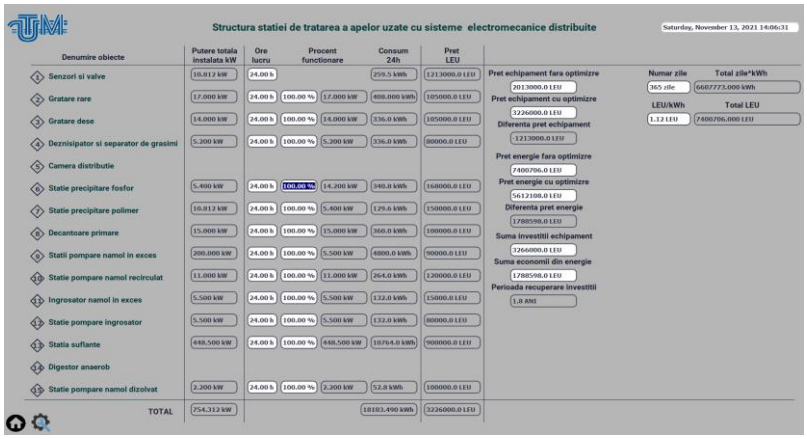


Figura 2. Reglarea automata a grătarelor rare si fine [8]

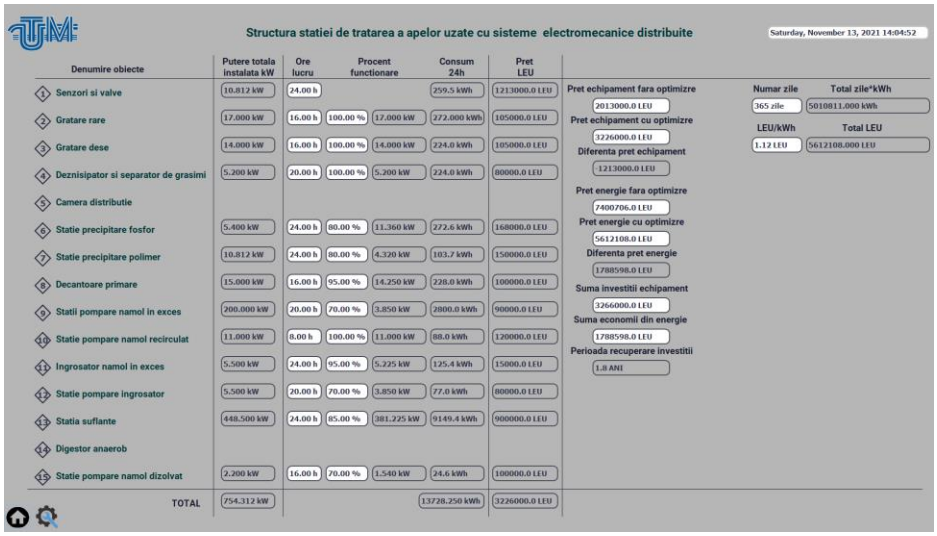


Figura 3. Parametrii de reglare ai deznisipatorului [8]

În Fig. 3 este modelată stația optimizată, cu ore de funcționare conform procesului tehnologic, cu posibile încărcări aproximativ aproape de date reale. La optimizarea stației am primit o suma de bani, care in decursul a 2-3 ani de zile, ne rascupara investițiile în echipament. Desigur constrcția unei stații mai are partea civilă, dar daca fiecare parte constructivă a stației ar fi simulată, s-ar putea de optimizat proiectarea ei, care la rândul său, o să aducă beneficii pe tot parcursul funcționării stației[4]. Cu ajutorul acestui soft, a fost proiectata partea electrica a sistemului electromecanic distribuit in cadrul statiei de tratare a apei uzate din orasul Cantemir, Republica Moldova. Sarcina de baza a fost reducerea puterii instalate. Inca o sarcina importanta a fost calcularea consumului de current electric, pentru a calcula care sunt cheltuielile pentru intretinerea acestei statii. Echiparea electromecanica si modul de functionare a fiecarui obiect in parte va fi descrisa mai jos in capitolele urmatoare.

Analiza parametrilor tehnici și tehnologici prin instrumentul grafic în stația de tratare a apei uzate

Partea grafica a starii echipamentelor de măsurare analogice, sunt redate in trei ferestre a sistemului SCADA.

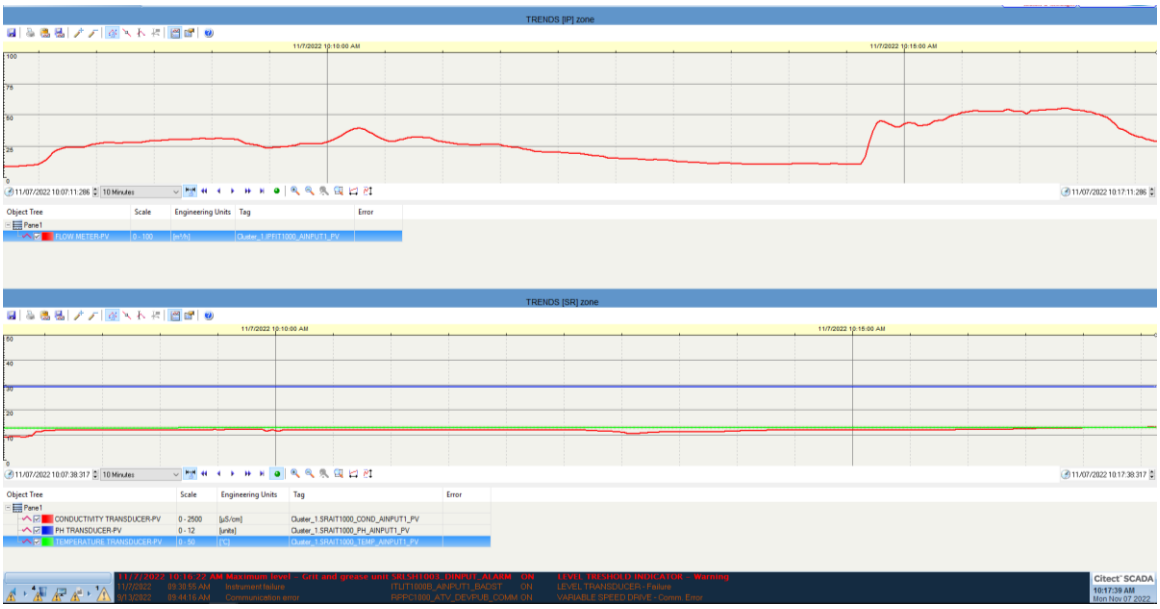
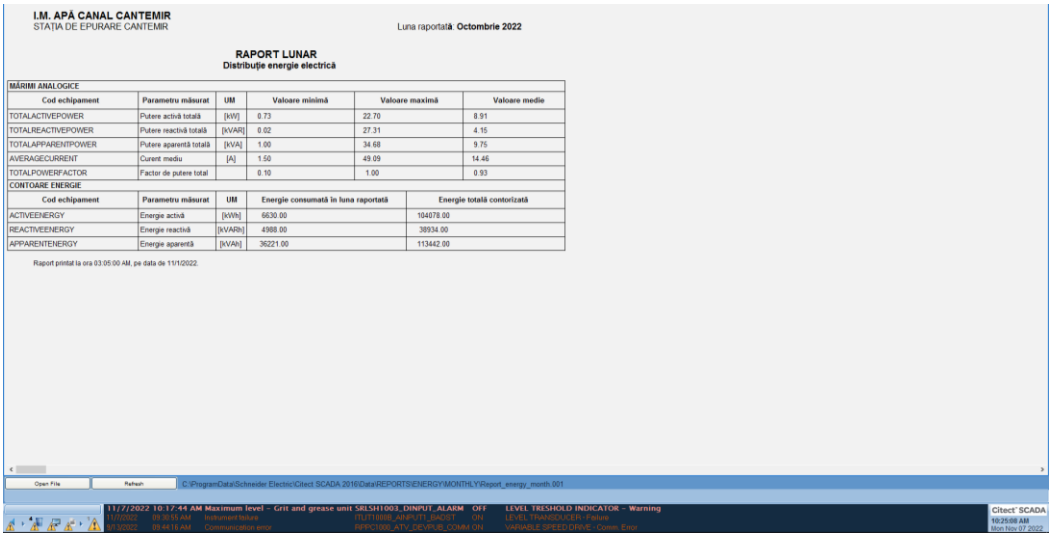


Figura 4. Exemplu pagină grafice pentru pretratare [7]

Toate variabilele analogice așa ca senzorii de nivel, traductorii de presiune, debitmetrele și analizatoarele de proces sunt afișați in sistemul SCADA prin instrumentul Process Analize. Operatorul poate selecta orice configuratie dorită, pe o perioada preselectată pentru a analiza procesul tehnologic. Toate datele graficelor sunt salvate in arhiva softului și pot fi analizate și printate in orice moment de timp.

Pentru a analiza în timp procesul tehnologic au fost create căteva tipuri de rapoarte zilnice, lunare și anule:

1. Consum de energie.
2. Orele de funcționare.
3. Parametrii de proces (debite, nivele, presiuni,pH, umiditate etc.) grupați pe trepte de proces.
4. Volume de apă brută/trataă, nămol, substanțe dozate ș.a.



I.M. APĂ CANAL CANTEMIR
STAȚIA DE EPURARE CANTEMIR

Luna raportată: **Octombrie 2022**

RAPORT LUNAR
Distribuție energie electrică

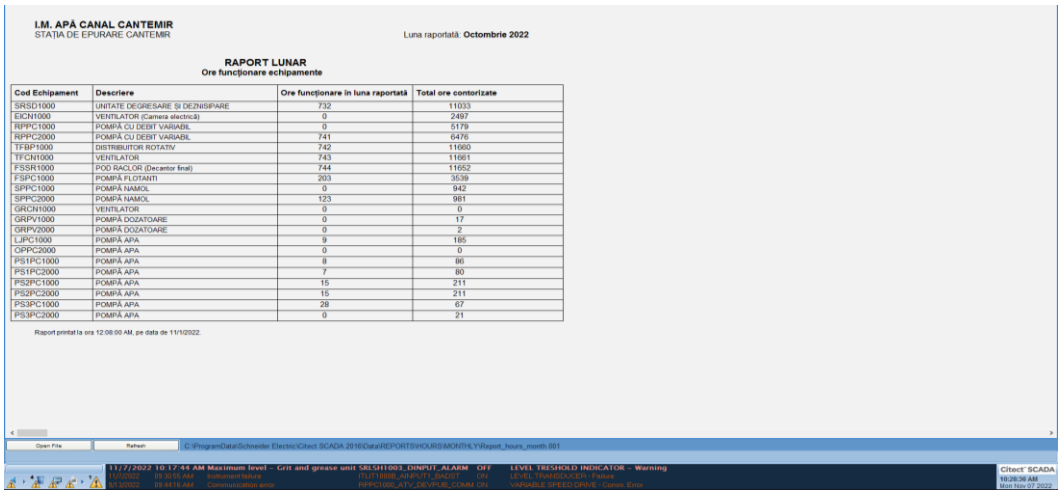
MĂRIMI ANALOGICE					
Cod echipament	**Parametru măsurat**	**UM**	**Valoare minimă**	**Valoare maximă**	**Valoare medie**
TOTALACTIVEPOWER	Putere activă totală	[kW]	0.73	22.70	8.91
TOTALREACTIVEPOWER	Putere reactivă totală	[kVAR]	0.02	27.31	4.15
TOTALAPPARENTPOWER	Putere aparentă totală	[kVA]	1.00	34.68	9.75
AVERAGECURRENT	Curent mediu	[A]	1.50	49.09	14.46
TOTALPOWERFACTOR	Factor de putere total		0.10	1.00	0.93

CONTOARE ENERGIE				
Cod echipament	**Parametru măsurat**	**UM**	**Energie consumată în luna raportată**	**Energie totală contorizată**
ACTIVEENERGY	Energie activă	[kWh]	6630.00	104078.00
REACTIVEENERGY	Energie reactivă	[kVARh]	4988.00	38934.00
APPARENTENERGY	Energie aparentă	[kVAh]	36221.00	113442.00

Raport printat la ora 03:05:00 AM, pe data de 11/1/2022.

Figura 5. Exemplu raport energie electrică lunar [7]

Conform raportului din Fig. 5 se poate analiza consumul total de energie electrică. La cazul dat lunar. Cu ajutorul softului elaborate, prin simulare, sau ajuns la aceste date pana a fi construită stația. Acest consum a fost mai mare, dar fiind faptul ca am putut micșora puterile instalate si automatiza procesul tehnologic în așa fel ca echipamentele să funcționeze la maxima eficiență energetică, am putut micșora acest consum. Ce a dus la micșorarea chetuielilor lunare, în unele cazuri și micșorarea prețului pe echipament



I.M. APĂ CANAL CANTEMIR
STAȚIA DE EPURARE CANTEMIR

Luna raportată: **Octombrie 2022**

RAPORT LUNAR
Ore funcționare echipamente

Cod Echipament	Descriere	Ore funcționare în luna raportată	Total ore contorizate
SRSD1000	UNITATE DEGRESARE ȘI DEZNISIPARE	732	11033
EICN1000	VENTILATOR (Camera electrică)	0	2497
RPPC1000	POMPĂ CU DEBIT VARIABIL	0	5179
RPPC2000	POMPĂ CU DEBIT VARIABIL	741	6476
TFBP1000	DISTRIBUITOR ROTATIV	742	11660
TFCN1000	VENTILATOR	743	11661
FSSR1000	POD RACLOR (Decantor final)	744	11652
FSPC1000	POMPĂ PLOTANTI	203	3539
SPPC1000	POMPĂ NAMOL	0	942
SPPC2000	POMPĂ NAMOL	123	981
GRCN1000	VENTILATOR	0	0
GRPV1000	POMPĂ DOZATOARE	0	17
GRPV2000	POMPĂ DOZATOARE	0	2
LJPC1000	POMPĂ APA	9	185
OPPC2000	POMPĂ APA	0	0
PS1PC1000	POMPĂ APA	8	86
PS1PC2000	POMPĂ APA	7	80
PS2PC1000	POMPĂ APA	15	211
PS2PC2000	POMPĂ APA	15	211
PS3PC1000	POMPĂ APA	28	67
PS3PC2000	POMPĂ APA	0	21

Raport printat la ora 12:08:00 AM, pe data de 11/1/2022.

Figura 6. Exemplu raport ore funcționare lunar [7]

Conform raportului din Fig. 6 se poate analiza diagrama de funcționare a fiecărui echipament în parte. La fel cu ajutorul softului proiectat au fost calculate orele aproximative lunare, și ajustate

pentru a calcula consumul de energie electrică totală preventiv. Conform acestor date este vizibil că unele echipamente lucrează fară oprire, unele sunt echipate cu convertizor de frecvență și pe langa faptul ca funcționeaza mai puțin în timp, dar și mai eficient. Cu ajutorul PLC-ul și modului de funcționare automatizat echipamentele funcționeaza doar cand e necesar conform sarcinei tehnologice, micșorînd cheltuielile și mărind termenul de funcționare fără defecte mecanice sau electrice [5].

Pentru o funcționare economică și fiabilă a unei stații moderne de tratare a apelor uzate, este inevitabil ca operatorii stației să obțină toate informațiile relevante și să aibă o imagine de ansamblu asupra stării actuale a tuturor proceselor instalației din camera de control.

În afară de aceste cerințe de control al instalației, reglementările locale și cerințele autorităților pentru înregistrarea, manipularea datelor, răspunsul la alarmă și arhivarea sunt îndeplinite prin utilizarea unui sistem de automatizare, control și SCADA de ultima generație.

Controlul de supraveghere și achiziția de date (SCADA) și sistemele de control automat PLC sunt furnizate pentru controlul și monitorizarea centralizate a instalației. Software-ul SCADA este CITECT. Aranjamentul general al controlului automat este inclus ca ecrane în SCADA.

Sistemul SCADA este amplasat într-o cameră de control special construită. Sala de control este dotata cu mobilier respectiv echipamentelor sistemului SCADA. Pentru parametrii critici sunt furnizate facilități de rezervă pentru stocarea pe disc de date.

Sistemele SCADA sunt echipate cu o sursă de alimentare neîntreruptibilă on-line, cu baterie limitată, pentru a permite monitorizarea stării de bază a centralei în timpul întreruperilor scurte de alimentare.

Sistemul de supraveghere și control al instalației (SCADA) cuprinde[6]:

- Un server SCADA care este și o stație de operare pentru monitorizare, control și supravegherea operațiunilor în stație.
- Loc de muncă SCADA pentru administrare (numai citire).
- O rețea de proces Ethernet pentru a lega PLC-ul principal de proces și serverul SCADA.
- Un commutator.
- Un router GSM.
- Controlere logice programabile (PLC-uri) pentru controlul procesului automatizat.

Serverul de bază are următoarea configurație tehnică:

5. Processor – intel Core i7 3,0 GHz.
6. Memorie – 16 GB RAM.
7. HDD – 1 TB.
8. Ethernet Gigabit controller incorporate.
9. Placa video incorporateă.
10. Tastatură USB.
11. Maus USB.
12. Monitor LED cu diagonala de 21.5”

Adițional serverul este echipat cu printer A3 și cu o memorie hard externă pentru back-up informație.

Conform Fig.7 comunicația intre echipamente este realizată prin interfața de comunicare paralelă ETHERNET. Comunicația este realizată intre serverele SCADA, PLC, HMI si convertizoare de frecvență.La fel este prezentă și comunicația prin GSM cu stațiile de pompare a apei brute.

Sistemul SCADA CITECT este echipat cu o licență de 1500 variabile, care poate fi reînoită la necesitate. Sistema SCADA indeplinesște următoarele funcții de bază:

13. Afișarea dinamică/statică a echipamentelor din stație.
14. Arhivarea pe termen lung/scurt a tuturor datelor și acțiunilor.
15. Vizualizarea graficicelor.
16. Afișarea alarmelor curente/arhivă, la fel și luarea la cunoștință de către operator.
17. Printarea alarmelor, graficelor cu legarea de timp real/timp arhivă.
18. Arhivarea evenimentelor.

19. Mentenanța echipamentelor și raportarea.
20. Monitorizarea și managmentul proceselor automatizate.

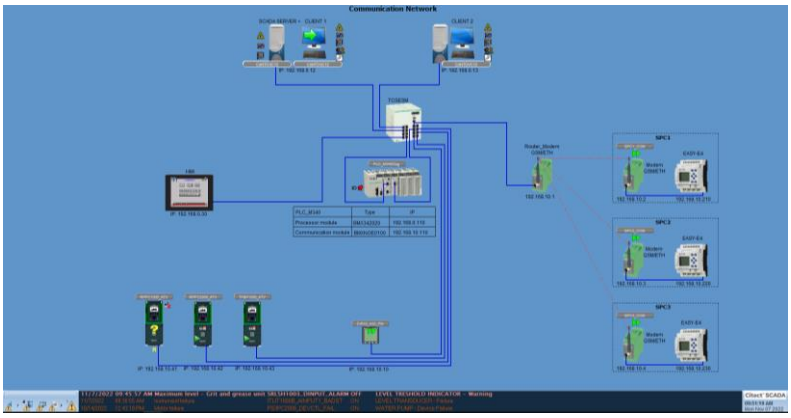


Figura 7. Sistem de comunicație în stație și extern [7]

Este prevăzut un sistem ierarhic de protecție prin parolă pentru a preveni accesul neautorizat la diferitele niveluri și funcții din cadrul sistemului de control.

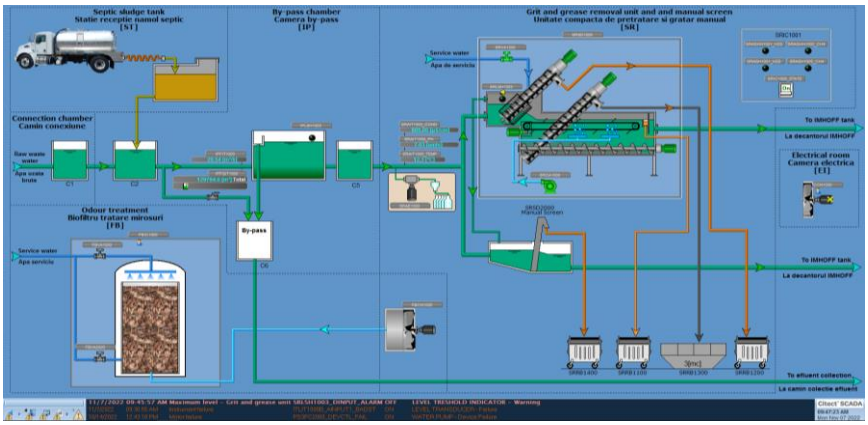


Figura 8. Exemplu de pagină a sistemului SCADA [7]

Diagramele cu obiecte și echipamente sunt afișate pe sistemul SCADA. De obicei, vizualizarea lor se bazează pe P&ID-ul proiectului și acoperă toate zonele de proces, după caz Fig. 8.

PLC-ul este utilizat pentru a efectua următoarele operații de bază:

1. Furnizarea de comenzi către diferite unități și actuatoare pe baza secvenței predefinite de funcționare pentru îndeplinirea interblocării de siguranță și optimizarea funcționării.
2. Citirea valorilor diferiților parametri de proces din instrumentele de câmp și analizatoare, run și condițiile de declanșare/defecțiune ale convertizoarelor de frecvență, starea deschis/închis și condițiile de declanșare/defecțiune ale supapelor motorizate, consumul de energie de la analizatoarul de rețea etc.
3. Comunicarea cu software-ul SCADA bazat pe PC pentru a obține funcționalitatea, stocarea datelor și raportarea lor.

Concluzii

1. In baza softului elaborat pentru proiectarea stațiilor de tratare a apei uzate, a fost proiectată stația de epurare din orașul Cantemir.
2. În baza calculelor primite dupa simularea softului au fost selectate corect echipamentele electromecanice din cadrul stației de tratare.
3. În baza simulării softului a fost consumul energetic a stației cu aproximativ 20%, durata de exploatare a fost marită cu aproximativ 2-3 ani și cheltuielile pe intreținere micșorate cu aproximativ cu 5 %.

4. Alegerea echipamentelor a fost optimizată din punct de vedere a eficienței energetice și a modului de funcționare.
5. Conform datelor din graficele extrase pe o perioada de un an de zile în funcționare, este evident ca stația de tratare își indeplinește toate funcțiile tehnologice cu ajutorul sistemelor electromecanice distribuite elaborate corect la etapa proiectării.
6. În baza rapoartelor a orelor de funcționare a echipamentelor electromecanice este clar că sistemul electromecanic distribuit este corect automatizat și echipat cu echipament respectiv funcționării eficiente și optime.
7. Datele asupra consumlui energetic după simularea softului sunt aproximativ egale cu datele primite în rapoartele energetice dupa funcționarea a unui an în regim autonom. Concluzie ce demonstreaza veridicitatea datelor calculate.
8. Softul proiectat a fost un instrument de baza la momentul proiectării acestei stații de tratare a apei uzate, cu ajutorul căruea au fost selectate echipamente de noua generație, cu un randament de funcționare înalt și cu o eficiență energetică ridicată. Toate aceste proprietăți tehnice au adus la o buna funcționalitate a sistemelor electromecanice distribuite ce își îndeplinesc funcția la cel mai inalt nivel conform procesului tehnologic prestabilit.

Referinţe

1. MOLDOVAN A, NUCA, I. "*Automation of Wastewater Treatment Plant*", SIELMEN 2019. Moldova, Chisinau.
2. MOLDOVAN A., "*Algoritmi pentru controlul si reglarea automată a sistemelor electromecanice distribuite programabile pentru stațiile de tratare a apelor uzate*", Technical Scientific Conference of Undergraduate, Master, PhD students Technical University of Moldova, 29-31 martie 2022.
3. NUCA I., *Acţionări electrice. Note de curs*. Chişinău, 2011, 90 p. (format electronic). http://elearning.utm.md/moodle/course/view.php.
4. IVANOV V., *Sisteme integrate de monitorizare si control pentru echipamente electrice*, Editura Universitaria, Craiova, 2008.
5. MILICI L.D., MILICI M.R, *Aplicaţii ale sistemelor de monitorizare şi transmisii de date. Editura didactică şi pedagogică,* Bucureşti, 2014,ISBN 978-973-30-3639-5, 202 pag.
6. *Procedee și echipament de epurare a apelor. Romania*, Universitatea "Petru Maior" TG. MUREȘ, 2011, 200p. (format electronic). http://www.amac.md/Biblioteca/data/17/08/Romania/80-Procedee-Si-Echipamente-de-Epurare-a-Apei.pdf
7. Sistemul SCADA în cadrul staţiei de epurare Cantemir, nepublicat.
8. Aplicație dezvoltată în softul TiaPortal V17.

ABOUT UNDERGROUND OVERHEAD POWER LINE

Roman GANUS[1*], Andrii POTRYVAI[1]

[1]*IEEE institute, PhD. student of Department of Electric Power Transmission, NTU "KhPI", Kharkiv, Ukraine*

*Corresponding author: Roman Ganus, mrgans.roma@gmail.com

Tutor/Coordinator: Sergii SHEVCHENKO, IEEE institute, prof. of Department of Electric Power Transmission, NTU "KhPI", Kharkiv, Ukraine**, Dmytro DANYLCHENKO** IEEE institute, docent of Department of Electric Power Transmission, NTU "KhPI", Kharkiv, Ukraine

Abstract. *In this paper, the main requirements and conditions for placing overhead power lines underground, the reasons for the relevance of such an implementation, and a methodology for calculating the optimal interphase distance for the proposed line design were determined.*

Keywords: *calculation method, line capacity, phase distance, power line protection.*

Introduction

Under the prevailing conditions in the world, when the demand and price for land plots are growing rapidly, the experience of war shows the vulnerability of the means of transmission of electrical energy, and the interest in new designs of overhead power lines (TL) is growing, it must be recognized that their traditional design is no longer always the best for solving problems of transmission of electrical energy.

The solution to all of the above problems, in our opinion, is to move overhead power lines underground. This solution has many advantages, however, the standard recommendations for the installation of overhead power lines will not create an optimal structure for construction and operation, and therefore it is necessary to revise many existing rules in order to form new ones that will achieve maximum efficiency and reliability of the structure.

Therefore, the objectives of this work are to consider the advantages and disadvantages of underground overhead power transmission lines (UOTL), review existing standards and identify issues that need to be addressed in the implementation of such a project.

Reasons for using

One obvious solution to the problems described above is the use of cable lines. The problem of using cable lines to transmit electricity over long distances is their large capacity. Thus, an attempt to transmit high voltage over a distance of 40 km or more will lead to the fact that the cable will act as a source of reactive power and it will not be possible to transmit active power anywhere. Therefore, cable lines are used mainly at short distances and small, relative to overhead power lines, voltages.

Overhead transmission lines do not have such problems, however, their dimensions and the high electric field strength around the conductors do not allow them to be brought deep enough into the city. And even if there is a path that allows you to build a line, for example, 330 kV, then the cost of these land plots will be extremely high, which will lead to the unprofitability of such a project.

Therefore, it is necessary to develop a solution that will allow transmitting significant power over long distances, will allow running a line deep into the city, while maintaining profitability, and also, for Ukraine, the requirement to protect lines from aircraft weapons will be relevant.

Details about OULT

The first disadvantage of the traditional design of high-voltage overhead power lines is their dimensions. This is due to the observance of the recommended distances between the phases and the need to create a protective zone.

The distance between the phases is affected by many parameters, but the key ones are the fluctuations that occur due to the influence of the wind, which can lead to wires lashing and breakdown of the air gap in the event of a critical approach of the wires.

The most common types of fluctuations are:

Vibration of wires (cables) - periodic oscillations of wires or cables in a run with a frequency of 3 to 150 Hz, which occur in a vertical plane during wind and form standing waves with a swing that can exceed the diameter of the wires (cables).

Dance of wires (cables) - constant periodic low-frequency (0.2 - 2 Hz) vibrations of wires (cables) in the run, which form standing waves (sometimes combined with running ones) with a number of half-waves from one to twenty and an amplitude of 0.3 - 5 m .

Sub-oscillations (oscillations of wires in sub-runs) are periodic wind-induced oscillations of a horizontally placed pair of split-phase wires that occur with the same or several half-waves in sections (called sub-runs) between adjacent intra-phase spacers with nodal points at the places where the spacers are installed.

According to [1], the protection of lines from the above types of vibrations is reduced to the use of appropriate devices (dampers) and compliance with the norms of the length of runs and phase-to-phase distances. Also in the norms "SOU 45.2-00100227-24: 2010 (Protection of wires and cables of repeated lines of electricity in the form of wind fluctuations (vibration, galloping, subfluctuations). Methodical wiring" in table B.2. the recommended values of the phase-to-phase distance for lines of different types of voltages are indicated (from 35 to 750 kV) Based on it, the choice of phase-to-phase distance is selected based on the length of the sag and the vertical distance H. Compliance with these recommendations allows you to be sure that whipping or wire breakage will not occur under most wind loads.

A fragment of this table for a voltage of 110 kV is given below, Tab.1.

The smallest displacements of wires of adjacent tiers horizontally on intermediate supports in areas with moderate dance (with a frequency of less than once every five years) for 110 kV.

Table 1.

OTL voltage, kV	**Vertical distances (H), m**	**Horizontal distance, m, with sagging booms at 0°C, м**							
		4	**5**	**6**	**8**	**12**	**16**	**20**	**>30**
110	3,0	-	-	1,15	1,70	2,40	2,80	3,50	4,15
	3,5	-	-	-	1,50	2,40	2,70	3,40	4,10
	4,0	-	-	-	1,20	2,20	2,65	3,40	4,10
	4,5	-	-	-	-	2,00	2,60	3,35	4,05
	5,0	-	-	-	-	1,80	2,50	3,25	4,00
	5,5	-	-	-	-	1,50	2,45	3,30	4,10
	6,0	-	-	-	-	1,20	2,30	3,20	4,00
	6,5	-	-	-	-	-	2,10	3,05	3,80
	7,0	-	-	-	-	-	2,00	2,90	3,70
	7,5	-	-	-	-	-	1,60	2,75	3,65
	8,0	-	-	-	-	-	1,20	2,60	3,50

In addition, the issue of interphase distance is considered in the REI (ПУЕ). Therefore, for example, the distance between the phases for an overhead line (OTL) of 110 kV according to the formula 2.5.22 [2] will be 3.3-3.6 m for a standard single-circuit version.

Moreover, this distance significantly depends on climatic conditions, such as: wind, ice loads, temperature influence and zoning according to thunderstorm activity. It is worth noting that these standards will not be relevant for UOTL, since underground climatic features will not have a significant impact, which means that these requirements can be neglected when searching for the optimal design. Then the search for the optimal design is reduced to providing the most favorable conditions for the line capacity, which directly depends on the distance between the phases [3]. The only condition that must be observed during layout is to ensure sufficient distance between the phases to prevent breakdown of the air space between the conductors.

Based on this, you can refer to the source [4], from which to determine the minimum allowable air gap Fig. 1-2.

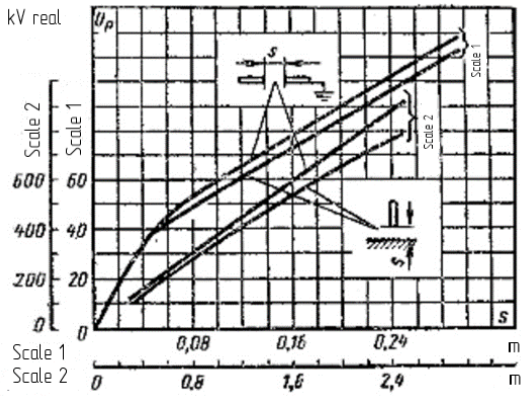


Figure 1. Discharge voltages for air gaps rod-to-rod and rod-to-plane at 50 Hz

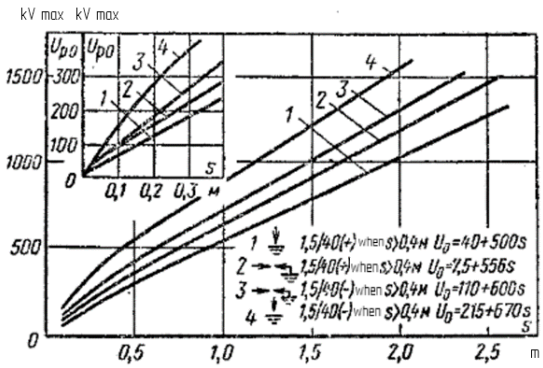


Figure 2. Discharge 50% - impulse voltage for air gaps rod - rod and rod – plane

However, the problem with these curves is that they are designed for bar-to-bar gaps. When it comes to significant phase convergence, which UOTL allows to perform, the error can have a significant impact, therefore, it becomes necessary to create curves for air gaps wire-wire and wire-plane. Accurate data can be built only by conducting an experiment, which is what the next work is planned to be devoted to.

The location of the lines underground, also, provide them with high security from enemy assets that damage the infrastructure.

To determine the required depth, it is worth referring to the statistics collected during the war in Ukraine.

For attacks on the energy infrastructure, cruise missiles of the “Caliber” and X-101 types were used. The conical shape of the funnel characterizes the charge as a normal ejection charge. This makes it possible, after a series of transformations of the formulas from the source [5], to calculate the depth of the funnel using the formula:

$$W = \sqrt[3]{\frac{L}{1{,}83 \cdot P}} \tag{1}$$

Where L is the weight of the warhead; P is the specific consumption of explosive per m^3 of blasted rock, equal to 2.43.

To prevent collapses and strengthen, it is necessary to make a layer of reinforced concrete, to clarify the thickness of which it is necessary to calculate the mechanical resistance of the material.

Thus, the UOTL takes the form of a line that is stretched inside a reinforced concrete collector of a cylindrical or parallelepiped shape with right angles. Options are shown in Fig. 3.

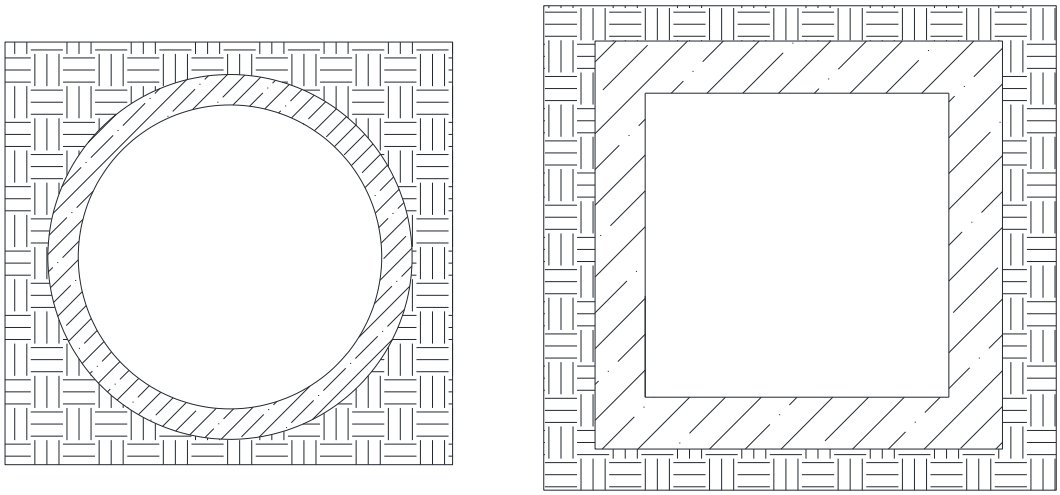


Figure 3. Schematic representation of the proposed type of the UOTL tunnel

However, this location is not without its downsides. Such a design implies the need to reduce the humidity inside the collector and adds to the calculation the need to take into account the moisture discharge voltage for the support insulators.

The need to control the humidity of the environment in the collectors can lead to the use of a large amount of expensive equipment, which will lead to a decrease in the profitability of the UOTL, but this issue needs to be considered in more detail during the feasibility study.

Conclusions

In this paper, the concept of UOTL was proposed, which is devoid of the disadvantages of classical solutions: overhead power lines of standard designs and cable lines. The advantages and disadvantages of this concept were considered.

Among the advantages, the possibility of changing the interphase distance, based solely on ensuring the highest throughput, is highlighted, since the lines are not affected by atmospheric phenomena (wind, ice, etc.). The underground location will allow the line to run deep into the city, which will reduce line losses and save on the cost of land plots through which the line could potentially pass, which significantly increases its profitability.

A method for calculating the depth of the formation of a funnel was determined, which will help determine the depth of the UOTL in the future. Two proposed types of collector are proposed.

The shortcomings of both traditional transmission lines and the proposed UOTL are considered. Its key disadvantage is the formation of condensate inside the collector, which leads to the need to control the humidity inside the system and take into account the moisture discharge voltage.

References

1. POSTOLATY V. M. *Technical characteristics of controlled self-compensating transmission lines and feasibility applying them to form backbone network of the Unified energy system of the USSR* / Proceedings of the Academy of Sciences of the USSR, Energy and transport. - Moscow, 1989. - S. 54-64..
2. *Правила улаштування електроустановок.* Kyiv: 2017, pp. 332
3. SHEVCHENKO S.Y., OKUN A.A: Анализ методов расчета электрических полей установок высоких напряжений [Analysis of methods for calculating the electric fields of high voltage installations]. In: *Электротехника и электромеханика* , 2010, pp.59-62.
4. PIGINI A., RIZZI G., GARBAGNATI E., PORRINO A., BALDO G., PESAVENTO G. *Performance of large air gaps under lightning overvoltages: experimental study and analysis*

of accuracy predetermination methods // IEEE Transactions on Power Delivery. – 1989. – vol.4. – no.2. – pp. 1379-1392. doi: 10.1109/61.25625.

5. OSNOVIN S. *Methods and formula for calculation located charges*, Izvestiya of the tomsk order of labor red banner polytechnical volume 84 institute named after S. M. Kirov 1956
6. IOSSEL Yu. YA., KOCHANOV E. S., STRUNSKY M. G. *Calculation of electrical containers. - 2nd ed*., revised. and additional - L .: Energoizdat, 1981 .- 240 pp.
7. POSTOLATY V. M. SHAKARYAN Yu.G., *Применение компактных управляемых линий электропередачи для формирования межсистемных и межгосударственных транзитов* [Compact applications managed lines power transmission for formation of inter-system and interstate transits]., энергия единой сети №3 (8) июнь – июль 2013
8. SHEVCHENKO S.; DANYLCHENKO D.; DRYVETSKYI S,; POTRYVAI A., *Modernization of a Simulation Model of a Photovoltaic Module, by Accounting for the Effect of Snowing of Photovoltaic Panels on System Performance with Correction for Panel Cleaning for Matlab Simulink*// 2021 IEEE 2nd KhPI Week on Advanced Technology (KhPIWeek). doi**:** 10.1109/KhPIWeek53812.2021.9570030

SECŢIA CALCULATOARE, INFORMATICĂ ŞI MICROELECTRONICĂ

SECTION OF COMPUTERS, INFORMATICS AND MICROELECTRONICS

SUBSECŢIA INGINERIA ŞI ŞTIINŢA CALCULATOARELOR

ENGINEERING AND COMPUTER SCIENCE SUBSECTION

DOMAIN-SPECIFIC LANGUAGE FOR ANALYZING MEDICAL RESULTS

Maia ZAICA[1*], Felicia LUPAȘCU[1], Cristian BRÂNZA[1], Nichifor POPESCU[1]

[1]*Department of Software Engineering and Automatics, Group FAF-212, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chișinău, Republic of Moldova*

*Corresponding author: Maia Zaica, maia.zaica@isa.utm.md

Abstract. *The following article aims to describe the way how the Medical Domain could benefit from a Domain Specific Language. The article explores the advantages of using Domain Specific Language in medical result analysis, such as improved accuracy, faster processing times, and reduced error rates. It also could benefit in collecting, updating, and keeping data. Finally, the article discusses potential future applications of Domain Specific Language in the medical field and how it can continue to contribute to the way medical results are analyzed and interpreted.*

Keywords: *analysis, data, domain-specific language, grammar, medical result, specific problem.*

Introduction

The analysis of medical results is an essential task in the healthcare industry, providing insights that aid in the diagnosis, treatment, and prevention of diseases. However, analyzing large volumes of medical data from different sources can be a challenging task, especially for healthcare professionals who may not have the necessary technical skills.

Domain-Specific Languages (DSLs) provide a promising solution to this problem by offering a specialized programming language that is tailored to the needs of the medical industry. In this article, a DSL designed specifically for the analysis of medical results is presented. The authors begin with an overview of domain analysis, discussing the key challenges associated with medical data analysis. An overview of the DSL is provided, highlighting its key features, including its ability to simplify complex data analysis tasks and improve the accuracy of medical diagnoses.

The grammar of the DSL is designed to be user-friendly and easy to understand. The potential impact of the DSL on the medical industry, including improved patient outcomes and reduced costs, is also discussed. Overall, the article provides a comprehensive overview of a domain-specific language for medical data analysis, offering insights into its design, implementation, and potential benefits.

Domain Analysis

General-purpose languages (GPLs) are robust and versatile, but they have weaknesses like everything. Comparing GPLs with general doctors:

"They are good at solving the first layer of diseases (problems), but they can't go deep, such as having a full diagnosis of heart disease (domain-related problems)" [1].

Thus, DSLs were created to provide a more effective way to address specific problems in a particular domain or industry. Starting with the definition of domain-specific language. "*Domain-specific languages are languages tailored to a specific application domain. They offer substantial gains in expressiveness and ease of use compared with general-purpose programming languages in their domain of application"* [2]. Rather than relying on general-purpose programming languages, which may not be optimized for a specific task or domain, DSLs allow developers to create languages that are tailored to the specific needs of a particular domain. This can result in increased productivity, better code readability, and more efficient code execution [3].

Developing a Domain-Specific Language is a challenging task that requires a combination of expertise in both language development and the specific domain the language is intended for.

A healthy population is essential for sustainable development and a strong macroeconomy. In turn, a strong economy is necessary to generate sufficient resources for health systems. Yet, despite this mutually beneficial relationship, it is often overlooked and health systems expenditure isn't taken into consideration most of the time. They may support the goal of universal health coverage, but have concerns that spending on health may be a drain on the economy [4].

The Republic of Moldova has a universal health care system. The financing of healthcare services in the country is based on the principles of fairness and unity, ensuring that everyone has equal access to healthcare services regardless of their financial status. However, over 10% of the population lacks health insurance coverage [5]. The level of transparency of the domestic health system is 36%, which ranks us among the countries with an unsatisfactory level of transparency. It is relatively simple to increase transparency by publishing data that institutions have already created and utilized internally [6].

To identify potential issues within the healthcare system that could be addressed or improved, secondary research was conducted. Secondary research involves analyzing and utilizing pre-existing data sources rather than collecting new data through primary research methods. This approach allows for the examination of a large amount of information and can provide insights into current trends, patterns, and potential areas for improvement. Healthcare organizations should look for advanced solutions that support: Easy Information Gathering, Real-Time Data Updates, Clear Data Visualization, and Simple Reporting.

Domain-specific language can be an effective solution to address the lack of system interoperability in healthcare. Healthcare systems often use different data formats and structures, which can make it difficult to share data between systems. By creating a DSL specifically designed for healthcare data, healthcare professionals can standardize the way that data is stored, analyzed, and shared across different systems. By using a common language, DSL can help to reduce errors and inconsistencies that can arise when data is transferred between different systems, ultimately improving patient care and outcomes.

A markup language annotates text with tags or codes to provide meaning and context to content. Tags indicate how text should be displayed or structured, and are embedded in the text itself. Markup languages make it easy to display documents on various devices and process them with software applications.

Grammar

A grammar G is an ordered quadruple $G = (V_N, V_T, P, S)$, where:

$-V_N$ is a finite set of non-terminal symbols

$-V_T$ is a finite set of terminal symbols

$$V_N \cap V_T = \emptyset$$

$-P$ is a finite set of production rules

$$P \subseteq (V_N \cup V_T)^* V_N (V_N \cup V_T)^* \times (V_N \cup V_T)^*$$

$-S$ is the start symbol

In a markup language, the terminal symbols represent the actual content of the document or data being marked up. The non-terminal symbols represent the structure or syntax of the markup language itself.

The production rules describe how non-terminal symbols can be replaced or expanded into other non-terminal or terminal symbols. These rules define the syntax and structure of the markup language.

To specify the grammar representation for the markup language we need to use meta notations (table 1):

Table 1

Meta notations

Symbol	Definition
< >	Non-terminal symbol
*	Zero or more occurrences
+	One or more occurrences
|	Separates the alternative symbols
->	Derivation
~	Except: any character except
\\	Comment

$V_N =$ {medical_results, test_result, imaging_result, lab_result, query, query_type, result_type, patient_data, patient_id, birthday, date, test_name, imaging_type, lab_name, result, unit, reference_range, normal_range, high_range, low_range, image_location, location}

$V_T =$ {"test", "imaging", "lab", "show", "for", "all", "average", "tests", "imaging", "labs", "patient", "age", "-", ">", "<", "units", "normal range", "high range", "low range"}

$P =$ {<medical_results> -> <test_result> | <imaging_result> | <lab_result> | <query>
<test_result> -> "test" <test_name> ":" <result> ["units" <unit>] [<reference_range>]
<imaging_result> -> "imaging" <imaging_type> ":" <result> ["units" <unit>] [<image_location>]
<lab_result> -> "lab" <lab_name> ":" <result> ["units" <unit>] [<reference_range>]
<query> -> "show" <query_type> "for" <patient_data>
<query_type> -> "all" <result_type> | "average" <result_type> "for" <time_frame>
<result_type> -> "tests" | "imaging" | "labs"
<patient_data> -> "patient" <patient_id> | "age" <birthday>
<patient_id> -> <word>+
<birthday> -> <date>
<date> -> <year> "-" <month> "-" <day>
<year> -> <number>
<month> -> <number>
<day> -> <number>
<test_name> -> <word>+
<imaging_type> -> <word>+
<lab_name> -> <word>+
<result> -> <number>
<unit> -> <word>+
<reference_range> -> "normal range" <normal_range> | "high range" <high_range> | "low range" <low_range>
<normal_range> -> <number> "-" <number> <unit>
<high_range> -> ">" <number> <unit>
<low_range> -> "<" <number> <unit>
<image_location> -> "location" <location>
<location> -> <word>}

An example of code using this grammar representation is:

```
Description {
      test glucose= 120
      units= mg/dL
      stage= normal
      range= 70-100
}
```

```
Setting {
        patient= John Smith,
        age= 1978-01-01
        gender= male,
        weight= 175 lbs,
        height= 5'11"
}
Response {
        show all tests for patient age 1978-01-01
         // Output: glucose: 120 mg/dL (normal)
}
Response {
        show average labs for last 6 months for patient John Smith
         // Output: No lab results found for patient John Smith in the last 6 months.
}
Response {
        show all imaging for patient John Smith
         // Output: No imaging results found for patient John Smith.
}
Response {
        length 300
        prompt "The patient's glucose test results were higher than normal and
        indicate the presence of diabetes. The patient is advised to follow up with
        a healthcare provider for further evaluation and treatment."
}
```

Parsing

Parsing algorithm:

A common approach for parsing context-free grammars like this one is to use a technique called recursive descent parsing. Here is how it could be done for this grammar:

1. Start with the <medical_results> rule.
2. Check if the next token matches <test_result>, <imaging_result>, <lab_result>, or <query>.
3. If it matches one of those rules, call the corresponding parsing function.
4. If it doesn't match any of those rules, return an error.
5. The parsing function for <test_result> would look for the "test" keyword, then call the <test_name>, <result>, and <unit> parsing functions in order. It would also check if there is a <reference_range> and parse it if it's there.
6. The parsing function for <imaging_result> would look for the "imaging" keyword, then call the <imaging_type>, <result>, and <unit> parsing functions in order. It would also check if there is an <image_location> and parse it if it's there.
7. The parsing function for <lab_result> would look for the "lab" keyword, then call the <lab_name>, <result>, and <unit> parsing functions in order. It would also check if there is a <reference_range> and parse it if it's there.
8. The parsing function for <query> would look for the "show" keyword, then call the <query_type> and <patient_data> parsing functions in order.
9. The parsing function for <query_type> would look for either "all tests", "imaging", "labs", "average tests", or "average imaging", and parse the associated data if necessary.
10. The parsing function for <patient_data> would look for either "patient" and a patient ID, or "age" and a date of birth.
11. The parsing function for <patient_id> would parse any string of alphanumeric characters.
12. The parsing function for <birthday> would look for a date in the format "YYYY-MM-DD".
13. The parsing function for <test_name>, <imaging_type>, <lab_name>, <number>, <unit>, <normal_range>, <high_range>, and <low_range> would all parse their respective data according to the grammar rules.

Recursive descent parsing works by recursively calling parsing functions for each rule in the grammar until the entire input is parsed. If the input is valid according to the grammar, a parse tree is constructed that represents the structure of the input. If the input is not valid, an error is returned.

Conclusions

In conclusion, the development of a Domain-Specific Language for analyzing medical results offers a promising solution to the challenges associated with medical data analysis. The DSL is designed to simplify the process of analyzing large volumes of medical data from different sources, improving the accuracy of medical diagnoses and reducing costs in the healthcare industry. Its user-friendly and intuitive nature, combined with a simple user interface, makes it accessible to healthcare professionals who may not have programming skills.

Overall, the DSL has the potential to transform the way medical data is analyzed, leading to improved patient outcomes and increased efficiency in the healthcare industry. As such, the development and adoption of this DSL should be a priority for healthcare professionals and organizations looking to improve their medical data analysis capabilities.

Acknowledgments

This article and the research behind it would not have been possible without the direction of our coordinator, Irina Cojuhari.

We'd like to express our gratitude and appreciation for the invaluable contribution to the completion of our article. The guidance and expertise provided were of utmost importance, and we appreciated the approachability and willingness to assist in every way possible, including providing mentorship and offering constructive feedback.

References

1. ALVES, I.R. Making your life easier with domain-specific languages (DSLs). In: *WAES*, January 2023. [online]. [accessed 06.03.2023]. Available: https://medium.com/wearewaes/making-your-life-easier-with-domain-specific-languages-dsl-1838d351d35
2. MERNIK, M., HEERING, J., SLOANE, A.M. When and how to develop domain-specific languages. In: ACM Computing Surveys, December 2005, p. 1 https://dl.acm.org/doi/10.1145/1118890.1118892
3. JetBrains. *Domain-Specific Languages*. [online]. [accessed 06.03.2023]. Available: https://www.jetbrains.com/mps/concepts/domain-specific-languages/
4. Euro Health Observatory (WHO). *Health and economy.* [online] [accessed 12.02.2023]. Available: https://eurohealthobservatory.who.int/themes/observatory-programmes/health-and-economy
5. Euro Health Observatory (WHO). *Country overview*. [online]. [accessed 12.02.2023]. Available: https://eurohealthobservatory.who.int/countries/republic-of-moldova
6. CARP, M.,GOMA, L. Efficiency of medical services in the Republic of Moldova. In: *Cercetarea în biomedicină şi sănătate: calitate, excelenţă şi performanţă.* (R), october 20-22, 2021, Chişinău. Chişinău, Republica Moldova: 2021, p. 64. https://ibn.idsi.md/vizualizare_articol/144098

SMART CITY – O SIMBIOZĂ DINTRE MEDIU, TEHNOLOGII, DATE, CUNOȘTINȚE ȘI CETĂȚENI

Ana ȚURCAN

Departamentul Informatica și Ingineria Sistemelor, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Ana Țurcan, e-mail: ana.turcan@fcim.utm.md

Îndrumător/coordonator științific: Victor ABABII, conf.univ.,dr., DIIS, UTM

Rezumat. În lucrarea de față orașul inteligent este definit ca o entitate ce exploatează tehnologiile informației și ale comunicațiilor pentru a îmbunătăți calitatea vieții cetățenilor săi, oferindu-le servicii de calitate, asigurând în același timp utilizarea eficientă a resurselor naturale oferite de mediul de activitate. Obiectul de cercetare este legtura dintre diferitele componente ale orașului inteligent care prezintă o simbioză dintre mediul de activitate, tehnologiile aplicate, mulțimea de date și cunoștințele explorate, și interacțunea acstora cu cetățenii. Fiecare component al orașului inteligent este interpretat ca o parte componentă a altuia și invers.

Cuvinte cheie*: Smart City, simbioză, mediul ambiant, tehnologii informaționale și de producere, baze de date, cunoștințe, cetățeni, servicii.*

Introducere

Procesul de urbanizare, care a devenit în ultimii ani tot mai intens, cu care se confruntă astazi omenirea impune și unele criterii de calitate față de infrastructură și serviciile oferite de agenții economici, serviciile de management și monitorizare. Reieșind din criteriile care determină calitatea vieții pot fi identificate cele mai importante servicii integritatea cărora se află la baza unui oraș inteligent [1-3]: servicii industriale și de producere, servicii de transport public, servicii de logistică, servicii de marketing, servicii medicale, servicii de asigurare cu apă și canalizare, servicii de mediu, servicii educaționale, servicii de asigurare cu energie electrică și gaze, servicii publice, de informare și asistență juridică etc. Toate acestea pot fi interpretate cît independent unul față de altul, dar și în colaborare care prezintă o simbioză de servicii care determină calitatea vieței oferită de mediul de activitate. După datele prezentate în Internet cele mai recunoscute 10 orașe inteligente din lume sunt considerate: Singapore, Dubai, Oslo, Copenhaga, Boston, Amsterdam, New York, Londra, Barcelona și Honk-Kong. Calitatea serviciilor oferite de Smart City depinde de complexitatea tehnologică a resurselor implicate și modelele matematice și tehnologice aplicate în funcționalitatea acestora [4].

Conform datelor oferite de sursa Internet [5] în Republica Moldova locuesc 4,028,702 de oameni (08.03.2023), dintre care 1,721,540 de oameni locuesc în orașe ceea ce constituie peste 42%. Pe 15 Noiembrie 2022 au fost inregistrati 8 Billion de oameni pe Pămînt. Pentru prima dată în istorie, peste 50 % din populaţia mondială trăieşte în zone urbane [6]. Este posibil ca până în 2050, circa 70 % din populaţie să fie urbană. Demografii estimează că Asia va ajunge până în 2050 să găzduiască peste 50 % din populaţia urbană mondială.

În lucrarea de față se propune dezvoltarea unui model al orașului inteligent în care se scoate în evidență importanța și modul de interacțiune a principalelor componente care prezintă o simbioză dintre: mediu, tehnologii, date, cunoștințe și cetățeni.

Simbioza Orașului Inteligent

Orașul inteligent [7] prezintă una din cele mai complexe sisteme orientată pentru a oferi servicii de calitate cetățenilor săi. Aceste sisteme fiind un proces social aplică modele și metode bazate pe cunoștințe inspirate din natură și au un caracter colectiv/colaborativ [8-10]. În Figura 1 este prezentată diagrama simbioza orașului inteligent.

Orașul inteligent (***Smart City***) integrează următoarele componente de bază:

Environment – mediul de activitate care integrează spațiile vierzi, bazinele cu apă terestră și subterană, arhitectura și geografia urbană, procese tehnologice și de producere, etc. Mediul de activitate prezintă principala sursă de date necesară pentru generarea deciziilor corecte și eficiente pentru protecția, dezvoltarea și ocrotirea mediului;

Data – mulțimea de date achiziționate de la setul de senzori de stare a mediului de activitate, și datele stocate pe dispozitive electronice utilizate pentru gestiunea eficientă a orașului inteligent;

Technology – tehnologii informaționale și de comunicații destinate pentru procesarea datelor și comunicarea dintre componentele sistemului. Întreprinderile cu procesele tehnologice și de prodicere sunt entitățile principare care asigură partea materială a orașului inteligent;

Knowledge – mulțimea de cunoștințe acumulate pe parcursul anilor și modele de inteligență artificială adaptate pentru gestiunea orașului inteligent;

Citizen – cetățenii orașului inteligent care sint implicați în procesele de producere, oferă și consumă servicii și produse specifice funcționării unui oraș inteligent.

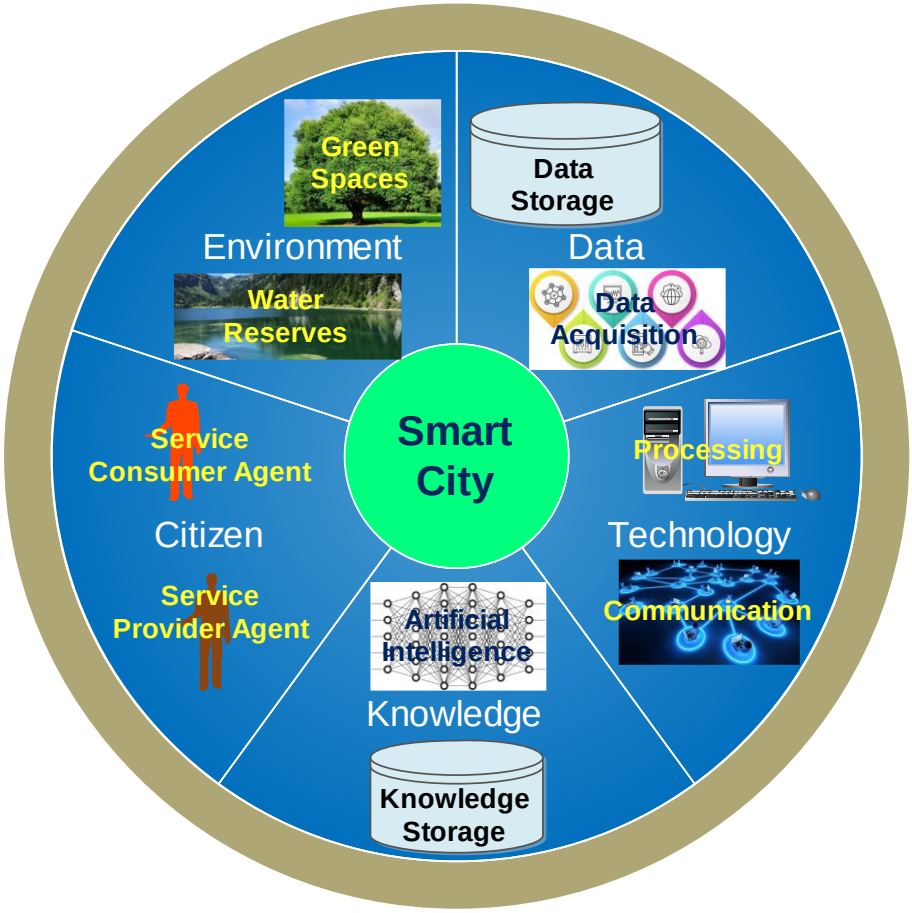


Figura 1. Simbioza unui Oraș Inteligent

Concluzii

În lucrarea de față sunt prezentate rezultatele proiectării și cercetării structural-funcțională a orașelor inteligente. Conceptul de oraș inteligent este prezentat ca o simbioză dintre mediu, date, tehnologii, cunoștințe și cetățeni.

Mulţumiri. Cercetările realizate fac parte din tematica tezelor de doctorat realizate la Departamentul Informatica și Ingineria Sistemelor, Universitatea Tehnică a Moldovei.

Referinţe:

1. D'ANIELLO, G., GAETA, M., ORCIUOLI, F., SANSONNETI, G. and SORGENTE, F. Knowledge-Based Smart City Service System, *Electronics 2020, 9, 965,* 22p., doi:10.3390/electronics9060965.
2. ANGELAKIS, V., TRAGOS, E., POHLS, H.C., KAPOVITS, A., BASSI, A. *Designing, Developing, and Facilitatings Smart Cities. Urban Design to IoT Solutions,* Springer, 2017, 338p., ISBN: 978-3-319-44924-1. DOI: 10.1007/978-3-319-44924-1.
3. PELTON, J.N., SINGH, I.B. *Smart Cities of Today and Tomorrow. Better Technology, Infrastructure and Security,* Springer, 2019, 267p., ISBN: 978-3-319-95822-4. DOI: 10.1007/ 978-3-319-95822-4.
4. ABABII, V., SUDACEVSCHI, V., BRANISTE, R., TURCAN, A., ABABII, C., MUNTEANU, S. Adaptive computing system for distributed process control. In: *International Journal of Progressive Sciences and Technologies. Vol. 22, No 2, September 2020,* pp. 258-264. ISSN 2509-0119.
5. *Moldova Population.* [accesat 01.02.2023] Disponibil: https://www.worldometers.info/world-population/moldova-population/.
6. *Populația lumii.* [accesat 01.02.2023] Disponibil: https://www.eea.europa.eu/ro/semnale/semnale-de-mediu-2011/megatendinta-mondiala-a-pamantului-pentru-2050/viata-intr-o-lume-urbana.
7. TURCAN, A., ABABII, V., SUDACEVSHI, V., MELNIC, R., ALEXEI, V., MUNTEANU, S., ABABII, C. Smart City Services based on Spatial – Temporal Logic. In: *Journal of Engineering Science 2022, 29 (3),* pp. 78-85, ISSN: 2587-3474 / E-ISSN: 2587-3482, https://doi.org/10.52326/jes.utm.2022.29(3).07.
8. MUNTEANU, S., SUDACEVSCHI, V., ABABII, V., BRANISTE, R., TURCAN, A., LEASHCENCO, V. Cognitive Distributed Computing System for Intelligent Agriculture. In: *International Journal of Progressive Sciences and Technologies. Vol. 24, No 2, January 2021,* pp. 334-342. ISSN 2509-0119.
9. MUNTEANU, S., SUDACEVSCHI, V., ABABII, V. Computer Systems Synthesis Inspired from Biologic Cells Structures. In: *Journal of Engineering Science, June, 2022, Vol. XXIX (2),* pp. 91-107, ISSN: 2587-3474 / E-ISSN: 2587-3482, DOI: 10.52326/jes.utm.2022.29(2).09.
10. TURCAN, A., BOROZAN, O., ABABII, V., SUDAEVSCHI, V., MUNTEANU, S. Decision Making System based on Collaborative Agents. In: *Proceedings of the 12th International Conference on Electronics, Communications and Computing (ECCO-2022), 20-21 October, 2022,* pp.257-260. ISBN: 978-9975-45-898-6. https://doi.org/10.52326/ic-ecco.2022/CE.09.

DOMAIN SPECIFIC LANGUAGE FOR PLC

Inga PALADI[1*], Vladislav FRUNZE[1], Vlad URSU[1], Roman TRANDAFIR[1], Irina RACOVCENA[1]

[1]*Department of Software Engineering and Automatics, group FAF-212, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chişinău, Republic of Moldova*

*The corresponding authort: Inga Paladi, inga.paladi@isa.utm.md

Scientific coordonator/ mentor : Irna COJUHARI, conf. univ., dr., Technical University of Moldova

Abstract. *This article discusses the potential of the domain specific language (DSLs) for programmable logic controller (PLCs), it presents the grammar and it is provided an example of code that could be used to create programs for the PLC. The grammar includes essential elements such as input, memory, and output variables, logical operations, and program structure. DSLs for PLCs offer several benefits, including efficiency, maintainability, and flexibility, making them valuable tools in the development of industrial automation.*

Keywords*: domain specific language, programmable logic controller, grammar, syntax, parser.*

Introduction

Programmable Logic Controllers (PLCs) are used as a vital component in modern industry which helps to control the manufacturing processes. In order to use the PLCs, it is needed to program them and the most used programming languages for it are Ladder Logic, Function Block Diagram, Sequential Function Charts, Structured Text, Instruction List [1].

Nevertheless, as the industry evolves, more complex programs are needed. And to do them in simpler ways it is necessary to develop a domain-specific language (DSL). A DSL is a programming language that specializes in a particular domain of control systems.

DSLs for PLCs give the possibility to create more maintainable, readable, and, in general, simpler programs that will govern industrial processes. DSLs provide us high-level abstractions which could help to program complex control logic in a more intuitive manner. As a feature, DSLs for PLCs could have code generation, which would significantly reduce the time necessary for PLC programming.

Generally, DSLs for PLCs have great potential for the development of industrial automation. They can provide efficiency, maintainability, and flexibility over ”ladder logic”. Therefore, while PLCs will have a critical role in the present manufacturing processes, the DSLs will become more helpful and in demand.

Introduction in PLCs

PLCs, or Programmable Logic Controllers, industrial-grade computers that can be programmed to operate specific control tasks. Some characteristics of the PLC include significantly reduced amounts of hardwiring used for traditional relay control circuits, easy programming, and installation, fast control, network interoperability, easy troubleshooting and testing, etc.

A programmable logic controller is supposed to work with a variety of input and provide output configurations, operate over a wide array of temperatures, be immune to electrical noise, and be impact and vibration resistant. Typically, battery-backed or non-volatile memory is where software for controlling and operating industrial process equipment and machinery is kept. A PLC is an illustration of a real-time system since the output of the system it controls is dependent upon the input circumstances.

The PLC was first used to replace relay logic, but due to its wide range of capabilities, it is now utilized in a variety of applications. A PLC can execute relay switching operations as well as other applications like timing, counting, calculating, comparing, and the processing of analog signals because of the fact that its structure is built on the same concepts as those used in computer architecture[1].

When discussing PLCs in relation to a traditional relay kind of control, programmable controllers have a number of benefits. To carry out a given task, relays must be hardwired. The relay wiring has to be altered or modified as soon as the system requirements change. In severe circumstances, such as the automotive industry, whole control panels had to be rebuilt because it was not practical to rewire the old panels for each model change. The programmable controller has significantly reduced the amount of hardwiring required for traditional relay control circuits. Compared to other relay-based process control systems, it is compact and reasonably priced. Relays are still a part of contemporary control systems, but logic is rarely applied to them.

Grammar

An example of a grammar that could help create the Domain Specific Language (DSL) for programming Programmable Logic Controllers (PLCs) is given below [2].

```
<program> ::= <start> <programName> <inputDeclarations> <outputDeclarations>
<memoryDeclarations> <logicStatements> <endProgram>
<start> ::= "BEGIN"
<endProgram> ::= "END"
<programName> ::= <identifier>
<identifier> ::= <alpha> <alphaNum>*
<alpha> ::= "a" | "b" | ... | "z" | "A" | "B" | ... | "Z"
<alphaNum> ::= <alpha> <digit>
<digit> ::= "0" | "1" | "2" | "3" | ... | "9"
<inputDeclarations> ::= "INPUT" <var> {"." <var>} ";"
<outputDeclarations> ::= "OUTPUT" <var> {"." <var>} ";"
<memoryDeclarations> ::= "RAM" <ram> {"." <ram>} ";"
<var> ::= "I" <x1> <x2> | "Q" <x1> <x2> | "M" <x1> <x2>
<x1> ::= "0"
<x2> ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
<ram> ::= <CN01> "." "AND" <coils> | <CN02> "." "OR" <coils> | <CN03> "." "XOR"
<coils> | <CN04> "," "NOT" <coils>
<CN01> ::= "CN01"
<CN02> ::= "CN02"
<CN03> ::= "CN03"
<CN04> ::= "CN04"
<coils> ::= <coil> {"." <coil>}
<coil> ::= <var>
<logicStatements> ::= <statement> {"." <statement>}
<statement> ::= <var> ":=" <logicOp>
<logicOp> ::= <var> "AND" <var>
        | <var> "OR" <var>
        | <var> "XOR" <var>
        | "NOT" <var>
```

The most important parts of this grammar are the following:

1. <program> production rule: This rule suggests the starting point of this grammar and defines the whole structure of a PLC program. It is made of several sub-rules for a program name, logic statements, input, and output declarations, and so on.
2. <var> production rule: This rule suggests the syntax and variable declaration. The DSL in question has three types of variables: input (I), output (Q), and memory (M). The variables are defined by first writing a single letter and then two digits. In this way, it will identify accurately where the inputs come from and where the outputs should go.
3. <ram> production rule: This rule denotes the syntax for memory declaration. The memory can be identified by a four-letter code in combination with logical operations (AND, OR, XOR, NOT) and a list of coils. Coils imply the variables which can be connected together to create complex operations.
4. <logicStatements> production rule: This rule attributes the result of a logical operation to a variable. The logical operation can be a simple AND, OR, XOR, or a NOT operation applied to a variable.

All in all, this DSL with the above grammar is created in order to help engineers in programming PLCs with a programming language that is similar to the ladder logic, but which is easier to learn and to use.

Code example

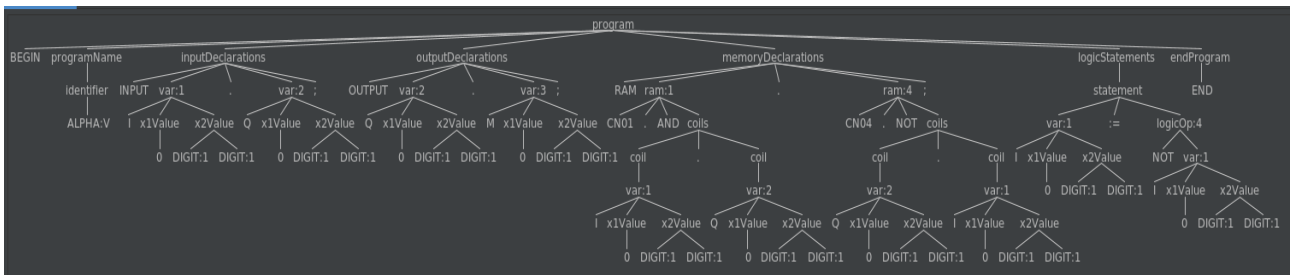
A presentation of a code snippet, that was constructed using the DSL described in this paper is shown in Fig. 2:

```
1   BEGIN
2   v
3   INPUT
4   I 011. Q 011;
5   OUTPUT
6   Q 011. M 011;
7   RAM
8   CN01.AND I 011. Q 011 . CN04.NOT Q 011. I 011;
9   I 011 := NOT I 011
10  END
```

Figure 1. DSL code example

Derivation Tree

A derivation tree or parse tree is a graphical representation that illustrates the way in which strings in a language are being derived, taking in consideration the rules of the grammar. Fig. 3 illustrates the derivation tree obtained from an example of code, by using ANTLR as a tool for parsing [3].


program
BEGIN
programName
inputDeclarations
outputDeclarations
memoryDeclarations
logicStatements
endProgram
identifier
INPUT
var:1
var:2
OUTPUT
var:3
RAM
ram:1
ram:4
statement
END
ALPHA:V
I
x1Value
x2Value
Q
M
CN01
AND
coils
CN04
NOT
var:1
:=
logicOp:4
coil
var:2
var:1
var:2
var:1
var:1
var:2
var:2
var:1
var:1
var:1

Conclusion

In conclusion, Programmable Logic Controllers (PLCs) are essential components in modern industry for controlling manufacturing processes. They are easy to program, fast, interoperable, and can operate in a wide array of conditions.

Furthermore, the most commonly used programming language for PLCs is "ladder logic", but as industry evolves, more complex and maintainable programs are needed. Domain-specific languages (DSLs) provide high-level abstractions that can simplify the programming process, reduce programming time, and increase maintainability.

DSLs for PLCs can be designed with a grammar that includes essential elements such as variables, logical operations, and program structures. By utilizing a DSL for PLCs, the programs created can be more maintainable, efficient, and flexible compared to those created with ladder logic.

Although relays are still used in contemporary control systems, PLCs have significantly reduced the amount of hardwiring required for traditional relay control circuits. PLCs have become essential for replacing relay logic in many applications.

As a result, with the potential benefits of DSLs for PLCs, it is likely that their demand will increase as industry continues to evolve, and they will become an increasingly vital component in industrial automation.

References:

1. PETRUZELLA, D.F. Programmable Logic Controllers 4th Edition, McGraw Hill, 2010, p.2.
2. AHO A.V., LAM M.S., SETHI R., ULLMAN J. D., *Compilers, Principles, Techniques, & Tools, second Edition*, p. 42-45.
3. PARR T., *The Definitive ANTLR Reference, Building Domain-Specific Languages*, 2007.

THE DEVELOPMENT OF A DOMAIN SPECIFIC MARKUP LANGUAGE FOR INTERACTIVE STORYTELLING

Denis PRODAN[1] , Rodica PRODAN[1],
Sandu GAZEA[1], Daniel BUCĂTARU[1],
Nicolae CHIPERI[1]

[1]*Department of Software Engineering and Automation, Group FAF-211, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chisinau, Republic of Moldova*

*Corresponding author: Denis Prodan, denis.prodan@isa.utm.md

Scientific coordinator: Braga Vasili, University lecturer, Technical University of Moldova

Abstract. The problem being dealt with is the need for a more intuitive and user-friendly DSL for creating interactive stories with branching paths and multiple outcomes. The language should incorporate declarative and imperative syntax, objects, variables, and collections, and require user input in the form of choices and interactions. User-generated content and various types of output can enhance the storytelling experience. The goal is to create a program that allows for dynamic, personalized stories that engage the reader and encourage active participation.

Keywords*: DSL, storytelling, outcomes, branches, grammar, parse tree.*

Introduction

Interactive storytelling is a rapidly growing field that involves creating stories where the reader or player performs an active role in shaping the narrative. While there are numerous programming languages available for game and simulation development, creating interactive stories with different pathways and multiple outcomes can be a complex and time-consuming task. This is where a domain-specific language (DSL) can prove to be incredibly beneficial. A domain-specific language (DSL) is a programming language that is designed for a specific domain or task, rather than being a general-purpose language. A DSL that is specifically designed for interactive storytelling can provide domain experts, such as writers, game designers, and virtual reality enthusiasts, with a more intuitive and user-friendly way to create compelling stories with branching paths and dynamic outcomes. By using a DSL, domain experts can focus more on the creative aspects of storytelling and less on the technical details of programming, thereby enabling them to bring their ideas to life more efficiently and effectively. Furthermore, a DSL has more benefits, like: productivity, quality, validation and verification, data longevity, platform independence, and domain involvement [1].

Domain Analysis

The problem being addressed is the creation of a DSL that provides a more intuitive and concise way to create interactive stories. While there are numerous programming languages available for developing games and simulations, these languages don’t always make it simple to create interactive stories with different pathways and multiple outcomes. There were implemented several tools for operating with narrative text, for example Inky editor [2].

These types of languages are used by people who like to engage in action completely, so the aim is to please different groups of people, including: young adults, gamers and virtual reality enthusiasts. It should allow users to easily create branching story paths, define character actions, dialogue, and create different outcomes based on user input. This language should also be easy to learn, so that even people who have no experience with programming can use it to create compelling interactive stories.

Language Overview

Interactive storytelling is a type of storytelling in which the reader or player actively shapes the narrative. There are several fundamental structures that must be understood before developing a program or domain-specific language (DSL) for interactive storytelling. The imperative model specifies the sequence of events in the story as well as the conditions that must be met for certain events to occur. The declarative model establishes the story's characters, settings, and objects, as well as their relationships and interactions. The functional model may also be used to process and manipulate data to determine the next steps in the narrative. Objects, variables, and collections are the basic data structures in this type of DSL. Objects in the story represent characters, settings, and objects, as well as their attributes and properties. Variables store and manipulate data, such as the reader's choices and the narrative's current location. Collections group together related data, such as a list of characters or potential story paths.

The user would typically use a combination of declarative and imperative syntax to control the flow of the narrative, including branching structures to evaluate the reader's choices and actions and looping structures to repeat a set of actions multiple times. This type of program or DSL requires user input in the form of choices, interactions, and other types of user input from the reader or player. The program would use this input to control the narrative's flow and manipulate data to update the story's state based on the reader's choices. Reader choices, interactions, and inventory management are some of the specific types of input that a program in this DSL may require.

User-generated content is another important type of input that a program in our DSL for interactive storytelling may require. User-generated dialogue, character names, locations, and even entire story paths are examples of this. User-generated content can add a high level of customization and personalization to a story, making readers feel more connected and engaged with it. A program the DSL may include various types of output in addition to reader input to improve the reader's experience. Output could include, for example, dialogue, images, sound effects, and music. These elements can aid in the creation of a more immersive and engaging storytelling experience, drawing the reader deeper into the story.

Finally, a program or DSL for intuitively narrating would have to incorporate a wide run of components and usefulness to form locks in and intelligently accounts that can adjust to the reader's choices and activities. By combining revelatory and basic language structure, essential information structures, control structures, and input/output usefulness, such a program might permit journalists and engineers to make energetic, personalized stories that capture the creative ability and motivate perusers to be dynamic members within the story.

Grammar

VN = {<program>, <var>, <knot>, <ID>, <content>, <text>, <goto>, <print>, <img>, <choice>, <var_op>, <expr>, <int>, <str>, <value>, <var_name>, <knot_name>, <img_name>, <option_text>, <EQ>, <LPAREN>, <RPAREN>, <LCURLY>, <RCURLY>, <GH>, <EXLAM>, <IMG_NAME>, <INT>, <ID>, <WS>, <EOF>}

VT = {'EOF', '(', ')', '*', '=', '<', '>', '[', ']', '{', '}', '!', '+', '-', '/', '(', ')(', ')', '<', '>', '=', '[', ']', '!', a'{', '}', ',', '.png', '.jpg', ‘0’..’9’, ‘a..z’, ‘A’..’Z’}

Table 1.

Meta Notation

Notation	Meaning
<foo>	means foo is a nonterminal
foo	foo in bold means foo is a terminal
x*	zero or more occurrences of x
x+	zero or more occurrences of x
|	separates alternatives
→	derives

```
<program>→ (<var>)* <knot>+ EOF

<knot> → <ID> '{' <content>* '}'
<var> → <var_name> '=' <value>
<value>→ <int>|<str>
<int>→<INT>
<str>→'"'(<ID>|<INT>)*'"'
<content>→ <text>
          | <goto>
          | <print>
          | <img>
          | <choice>
          | <var_op>
<var_op> : '[' <var_name> '=' <expr> ']'
<expr>:   <expr> ('*'|'/') <expr>
          |   <expr> ('+'|'-') <expr>
          |   <int>
          |   '(' <expr> ')'
          |   <str>
          | <var_name>
<goto> → '(' <knot_name> ')'
<print> → '(''(' <var_name> ')'')'
<img> →'('!' <img_name> ')'
<choice>→ '(' '(' <pair>*  ')'')'
<pair>→'!'<option_text> <goto>
<option_text>→ (<ID>|<INT>)*
<knot_name>→ <ID>
<var_name>→<ID>|<INT>
<img_name>→<IMG_NAME>
<text>→ <ID>|<INT>;
<EQ> → '='
<LPAREN> → '('
<RPAREN> → ')'
<LCURLY> → '{'
<RCURLY> → '}'
<GH>→'"'
<EXLAM>→ '!'

<IMG_NAME>→ <ID> '.png'| <ID> '.jpg'
<INT> → [0-9]+
<ID>→ [a-zA-Z_][a-zA-Z_0-9]*
<WS>→ [ \t\n\r\f]+ -> skip
```

Parsing Tree

A parsing tree or concrete syntax tree is an ordered, rooted tree that represents the syntactic structure of a string according to some context-free grammar. The term parse tree itself is used primarily in computational linguistics. In theoretical syntax, the term syntax tree is more common [3]. For the following code snippet, the corresponding parse tree was generated (Fig. 1):

```
myvar="hello world"
second=12
f {
        ce faci132
        (y2)
        ((j))
        (!car.jpg)
        ((!what are you doing (y2)
        !here (y4)))
  }
y{
hi io hi
}
```

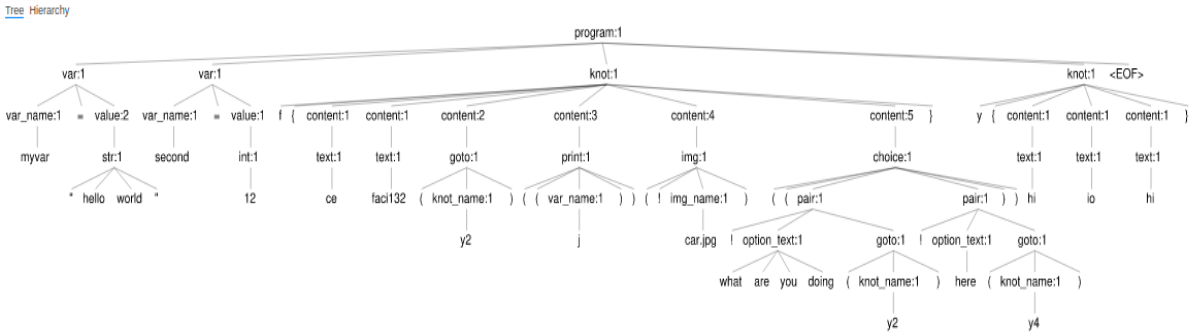


Figure 1. Parse Tree

Conclusions

Domain Specific Language can be useful for different tasks, purposes and people. The difference between Interactive storytelling DSLs and other programming languages is that they must cope with outcomes, branches and characters. These notions are naturally understood by humans, but can be complicated to be transposed into code. This article introduces a new DSL concept, which develops the idea of a more friendly markup language for interactive storytelling.

References

1. VOELTER, M. DSL Engineering: Designing, Implementing and Using Domain-Specific Languages. Germany, 2013.
2. Writing web-based interactive fiction with ink, [online]. [accessed 05.03.2023]. Available: https://www.inklestudios.com/ink/web-tutorial/.
3. Compilers: Principles, Techniques, and Tools [online]. (accessed 05.03.2023]. Available: *https://en.wikipedia.org/wiki/Compilers:_Principles,_Techniques,_and_Tools#cite_note-1*

SINTEZA ALGORITMULUI DE REGLARE LA MODELUL OBIECTULUI CUANTICIPAŢIE-ÎNTÂRZIERE DE ORDINUL TREI CU OSCILAȚII

Dumitru MORARU

Departamentul Ingineria Software şi Automatică, Facultatea Calculatoare Informatică şi Microelectronică, Universitatea Tehnică a Moldovei, Chişinău, Republica Moldova

Autorul corespondent: Dumitru MORARU, e-mail dumitru.moraru@ati.utm.md

Rezumat. *În lucrare s-a elaborat o procedură de acordare a algoritmului PID la modelul stabil al obiectului de reglare cu anticipare-întârziere de ordinul trei cu oscilații cu parametrii cunoscuți după metoda gradului maximal de stabilitate cu iterații. Din ecuația caracteristică a sistemului închis, prin operații de derivare pe variabila necunoscută a gradului maximal de stabilitate, se obțin expresiile analitice ale parametrilor de acord ai regulatorului PID ca funcții neliniare de argumentul - gradul de stabilitate și liniare de parametrii cunoscuți ai modelului obiectului de reglare. Variind argumentul gradul maximal de stabilitate aceste funcții se construiesc în formă grafică. Prin proceduri de iterații pe aceste curbe pentru aceiași valoare a argumentului se aleg seturi de valori ale parametrilor regulatorului PID și prin simulare se verifică performanțele sistemului automat. Pentru verificarea și compararea rezultatelor obţinute s-a examinat un exemplu după metoda elaborată și după metoda polinomială cu simularea sistemului pe calculator. S-au variat parametrii modelului obiectului de la valorile nominale și s-a verificat robustețea sistemului. S-a verificat dinamica sistemului la acțiunea perturbației de tip treaptă unitară. Se evidențiază avantajele procedurii elaborate prin calcule grafo-analitice reduse şi timp minim, care conduc la simplificarea procedurii de acordare a parametrilor algoritmului de reglare PID pentru aceste modele de obiecte.*

Cuvinte cheie*: model stabil de obiect de reglare cu inerție, funcție de transfer, regulator PID, parametrii de acord, sistem automat, metode de acordare, acordarea regulatorului, răspunsul sistemului, performanțele sistemului.*

Introducere

În lucrare se utilizează modelul matematic al obiectului de reglare cu anticipație-întârziere de ordinul trei cu oscilații [1-4]:

$$H(s) = \frac{k_1}{T_2 s+1}\frac{k_2(T_1 s+1)}{s^2+2qws+w^2} = \frac{b_0 s+b_1}{a_0 s^3+a_1 s^2+a_2 s+a_3}, \quad (1)$$

unde k_1, k_2 sunt coeficienții de transfer, T_1, T_2 – constantele de timp, q -gradul de amortizare și w - pulsația naturală ale obiectului tehnic, iar $b_0 = k_1 k_2 T_1$, $b_1 = k_1 k_2$, $a_0 = T_2$, $a_1 = 2qwT_2 + 1$, $a_2 = 2qw + T_2 w^2$, $a_3 = w^2$.

Modelul (1) conține un zerou și trei poli alocați în semiplanul stâng al planului complex al rădăcinilor.

La automatizarea proceselor industriale au aplicare regulatoarele de tipul PID [3-6].

Există mai multe metode de acordare a algoritmului PID care se aplică pentru acordarea regulatorului la modelul obiectului (1): metoda frecvențială, metoda de alocare poli-zerouri, metoda polinomială etc. [4-6].

Aplicarea metodei frecvențiale este însoțită de calcule în domeniul frecvență și construcții grafice care conduc la dificultăți la sinteza algoritmilor de reglare [4,5].

Metoda alocării poli–zerouri este o metodă analitică și se utilizează pentru sinteza algoritmii de tipul PID, dar procedura de sinteză este cu iterații și devine dificilă [4,5].

Metoda polinomială prezintă o metodă analitică, care rezolvă problema de sinteză a algoritmului de reglare, dar este dificilă la etapa construirii polinomului caracteristic [6].

În lucrare s-a elaborat metoda gradului maximal de stabilitate cu iterații (GMSI) [7-9] de acordare a regulatorului PID la modelul obiectului de reglare (1).

Se studiază un exemplu de acordare a regulatorului PID la modelul obiectului (1) și se variază parametri obiectului de la valorile nominale și se analizează performanțele sistemului.

Algoritmii de acordare a regulatorului PID

În studiu se utilizează schema structurală a sistemului automat, alcătuită din modelul obiectului cu f.d.t. $H_P(s)$ și regulatorul cu f.d.t. $H_R(s)$ supus acțiunii intrării treaptă unitară.

Se consideră algoritmul PID standard în conexiune paralelă descris cu funcția de transfer:

$$H_{PID}(s) = k_p + \frac{k_i}{s} + k_d s = \frac{k_d s^2 + k_p s + k_i}{s}, \tag{2}$$

unde k_p, k_i, k_d sunt parametrii de acord ai componentei proporționale, integrative și derivative ale algoritmului PID.

Procedura de acordare a algoritmului PID după metoda GMSI se reduce la alcătuirea ecuației caracteristice a sistemului închis, se introduce noțiunea de grad de stabilitate în ecuația caracteristică ca o nouă variabilă necunoscută și prin operații de derivare a acesteia se obțin relații care prezintă parametrii PID ca funcții $k_p = f_p(J)$, $k_i = f_i(J)$, $k_d = f_d(J)$ analitice neliniare de parametrii cunoscuți ai modelului obiectului și de variabila necunoscută - gradul de stabilitate al sistemului [8-9].

Pentru sistemul automat cu algoritmul PID se obțin trei expresii de calcul ai parametrilor:

$$k_d = \frac{d_0 J^5 - d_1 J^4 + d_2 J^3 - d_3 J^2 + d_4 J - d_5}{2(b_1 - b_0 J)^4} = f_d(b_0, b_1, a_0, a_1, a_2, a_3, J) = f_d(J), \tag{3}$$

$$k_p = \frac{-d_0 J^4 + d_1 J^3 - d_2 J^2 + d_3 J}{(b_1 - b_0 J)^2} + 2k_d J = f_p(b_0, b_1, a_0, a_1, a_2, a_3, J) = f_p(J), \tag{4}$$

$$k_i = \frac{-a_0 J^4 + a_1 J^3 - a_2 J^2}{b1 - b_0 J} - 2k_d J + k_p J = f_i(b_0, b_1, a_0, a_1, a_2, a_3, J) = f_i(J), \tag{5}$$

unde coeficienții din expresiile (3)–(5) se exprimă prin parametrii obiectului.

Se variază $J = 0 \cdots \infty$, se calculează și se construiesc curbele (3)–(5) $k_p = f_p(J)$, $k_i = f_i(J)$, $k_d = f_d(J)$ pentru algoritmul PID. Pe aceste curbe se aleg seturi de valori ai parametrilor regulatorului PID $J_i - k_{pi}$, k_{ii}, k_{di} și se simulează sistemul automat, se înregistrează răspunsurile la semnal treaptă unitară după care se determină performanţele cele mai ridicate ale sistemului, care ar satisface condițiile de performanță impuse sistemului.

lu al modelului obiectului de reglare (1) se analizează procedurile de acordare a algoritmului PID după metoda GMSI și metoda polinomială, se simulează sistemul și se analizează performanțele sistemului proiectat.

Se variază parametrii modelului (1) cu ±50% de la valorile nominale și se analizează performanțele sistemului.

Aplicații și simulare pe calculator

Pentru verificarea rezultatelor obținute la acordarea algoritmilor PID pentru modelul (1) după metoda gradului maximal de stabilitate cu iterații se prezintă exemplu de calcul.

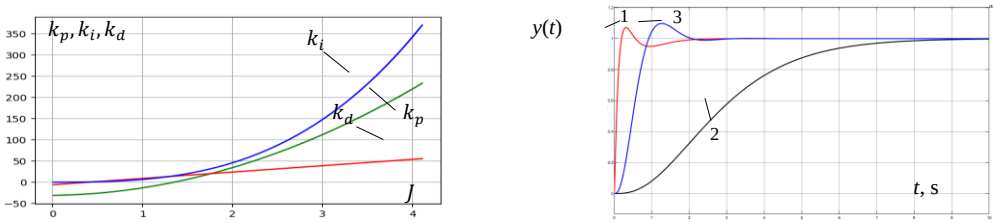
Exemplu. Se consideră obiectul tehnic descris ca modelul obiectului de reglare stabil cu inerție de ordinul $n = 3$ cu parametrii cunoscuți: coeficienții de transfer $k_1 = 1$, $k_2 = 0.2$, constantele de timp $T_1 = 0.1$ s, $T_2 = 0.1$ s, gradul de amortizare $q = 0.3$ și pulsația naturală $w = 2.5\ s^{-1}$ prezentat de funcția de transfer de forma [2]:

$$H(s) = \frac{k_1}{T_2 s + 1} \frac{k_2(T_1 s + 1)}{s^2 + 2qws + w^2} = \frac{10}{s + 10} \frac{k_2(T_1 s + 1)}{s^2 + 2qws + w^2} = \frac{b_0 s + b_1}{a_0 s^3 + a_1 s^2 + a_2 s + a_3}, \tag{6}$$

unde $b_0 = 10k_2T_1 = 0.2$, $b_1 = 10k_2 = 2$, $a_0 = 1$, $a_1 = 2qw + 10 = 11.5$, $a_2 = 20qw + w^2 = 21.25$, $a_3 = 10w^2 = 62.5$.

Pentru performanțele impuse sistemului timpul de reglare $t_r \leq 2$ s și suprareglarea $\sigma \leq 10$ % să se sintetizeze algoritmul de reglare PID la modelul obiectului (6) prin metoda GMS cu iterații. Pentru verificare și comparare se aplica metoda polinomială.

Soluționare. S-au efectuat calculele de acordare a parametrilor regulatorului PID după metoda GMS cu iterații după relațiile (3)–(5) pentru modelul (6) și curbele sunt date în fig. 1, *a*. S-a simulat sistemul automat cu regulatorul PID (fig. 2) și răspunsul optimal se prezintă în fig. 1, *b* (curba 1, rândul 3 din tabelul 1), iar performanțele sistemului sunt date în tabelul 1.



a) *b*)

Fig. 1. Funcțiile $k_p = f(J)$, $k_i = f(J)$, $k_d = f(J)$

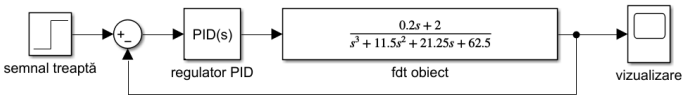


Figura 2. Schema de simulare a sistemului automat

Tabelul 1

Performanțele sistemului automat pentru modelul obiectului din exemplu

Nr. iter.	Metoda acordare	Tip reg.	Parametrii regulatorului					Performanțele sistemului			
			J	k_p	k_i	T_i, s	k_d, s	t_c, s	d, %	t_r, s	n
1	GMS	PID	3.2	122.35	163.48	0.006	40.5	0.25	-	1,56	-
2	GMS	PID	3,5	152.5	214.4	0.0047	45	0.21	3.68	1.32	1
3	**GMS**	**PID**	**3,9**	**196.9**	**296.6**	**0.0034**	**51**	**0.17**	**6.57**	**0.42**	**1**
4	MP1							7.6	-	7.6	-
5	MP2							0.85	9.69	1.68	1

Metoda polinomială. După această metodă se construiesc două modele a algoritmului de reglare.

În prima procedură (MP1) se adoptă cazul când rădăcinile polinomului caracteristic sunt multiple și care asigură eroarea staționară $e = 0$ la semnal de referinţă de tip treaptă unitară și se construiește funcția de transfer a algoritmul de reglare de forma:

$$H_R(s) = \frac{Q(s)}{P(s)} = \frac{m_0\,(a_0s^3+a_1s^2+a_2s+a_3)}{(b_0s+b_1)(n_0s^2+n_1s+n_2)s} = \frac{s^3+11.5s^2+21.25s+62.5}{(0.2s+2)(s^3+3s^2+3s)} = \frac{q_0s^3+q_1s^2+q_2s+q_3}{p_0s^4+p_1s^3+p_2s^2+p_3s}, \quad (7)$$

unde parametrii algoritmului au valorile numerice $q_0 = 1$, $q_1 = 11.5$, $q_2 = 21.25$, $q_3 = 62.5$, $p_0 = 0.2$, $p_1 = 2.6$, $p_2 = 6.6$, $p_3 = 6$.

În a doua procedură (MP2) se construiește polinomul caracteristic dorit al sistemului după polii dominanți $s_{1,2} = -2.0005 \pm j2.7271$ și se mai impune un pol suplimentar $p_3 = -10$, după care se construiește funcția de transfer a algoritmului de reglare, în care se include și astatism de gradul unu pentru a asigura eroarea staționară $\varepsilon = 0$ la semnalele de referință și perturbație de tip treaptă unitară:

$$H_R(s) = \frac{Q(s)}{P(s)} = \frac{m_0\,(a_0s^3+a_1s^2+a_2s+a_3)}{(b_0s+b_1)(n_0s^2+n_1s+n_2)s} = \frac{q_0s^3+q_1s^2+q_2s+q_3}{p_0s^4+p_1s^3+p_2s^2+p_3s}, \quad (8)$$

unde parametrii algoritmului au valorile numerice $q_0 = 114.395$, $q_1 = 1315.5425$, $q_2 = 2430.8937$, $q_3 = 7149.6875$, $p_0 = 0.2$, $p_1 = 4.8002$, $p_2 = 38.2927$, $p_3 = 102.903$.

S-a simulat sistemul automat cu algoritmul de reglare sintetizat după metoda polinomială MP1 și MP2 și răspunsurile sunt prezentate în fig. 1, *b*, (curbele 2 și 3), iar performanțele sunt date în tabelul 1, rândurile 4 și 5.

Concluzii

În rezultatul studiului efectuat se formulează următoarele concluzii.

1. Sistemul cu regulatorul acordat după metoda GMS cu iterații are performanțe mai ridicate în comparație cu sistemul cu regulatorul acordat după metoda polinomială MP1 și MP2.
2. Pentru acordarea regulatorului la modelul obiectului (1) se recomandă de utilizat metoda GMS cu iterații, pentru care calculele sunt mai reduse.
3. Sistemul cu regulatorul acordat după metoda GMSI este mai robust de 6.3 ori în raport cu sistemul cu regulatorul acordat după metoda polinomială (gradul de stabilitate $J = -3.8899$, iar a sistemului cu regulatorul acordat după metodele MP1 $J = -1.2457$, MP2 $J = -0.75$).
4. La variația parametrilor modelului obiectului cu $\pm 50\%$ de la valorile inițiale sistemul cu regulatorul acordat după metoda GMSI este mai robust decât sistemul cu regulatorul acordat după metodele polinomiale.

Bibliografie

1. VETLINSKII, V.N., OSIPOV, A.V. *Avtomaticheskie sistemy upravlenia dvijeniem avtotransporta.* L.: Mashinostroenie, Leningr. otd-nie, 1986.
2. SHATALOV, A.S., TOPCHEEV, Iu.I. KONDRATIEV, B.S. *Letatelnye apparaty kak obiecty upravlenia*. M.: Mashinostroenie, 1972.
3. DIB, F., BENAYA, N., BEN MEZIANE, K., BOUMHIDI, I. *Comparative Study of Optimal Tuning PID Controller for Manipulator Robot*. The Proceedings of the International Conference on Smart City Applications SCA 2022: Innovations in Smart Cities Applications Volume 6. [Accesat 6.03.2023]. https://doi.org/10.1007/978-3-031-26852-6_23
4. DORF, R.; BISHOP, R. *Sovremennye sistemy upravlenia* (*Modern Control Systems).* Moskva: Laboratoria Bazovyh Znanii, 2004, 832 s. ISBN 5-93208-119-8.
5. DUMITRACHE, I. *Ingineria reglării automate*. București: Politehnica Press. V.1, 2016, ISBN 978-606-515-686-9.
6. KIM, D. P. *Teoria avtomaticheskogo upravlenia. T.* 1. *Lineinye sistemy.* M.: FIZMATLIT, 2003, ISBN 5-9221-0379-2.
7. IZVOREANU, B., FIODOROV, I., *The Synthesis of Linear Regulators for Aperiodic Objects with Time Delay According to the Maximal Stability Degree Method*. In: *Preprints the Fourth IFAC Conference on System Structure and Control*, București.: Ed. Tehnică, 1997, pp. 449-454.
8. IZVOREANU, B., FIODOROV, I., Pisarenco, M. *Comparative Analysis of Regulators Tuning Methods to Models of Objects with Inertia*. În *Buletinul Institutului Politehnic din Iaşi*, Tomul L(LIV), Fasc. 5A, Electrotehnică, Energetică, Electronică, 2004, pp. 63-68.
9. IZVOREANU, B.; COJUHARI, I.; FIODOROV, I.; MORARU, D.; SECRIERU, A. *Tuning the PID Controller to the Model of Object with Inertia Second Order According to the Maximum Stability Degree Method with Iteration.* Annals of the University of Craiova. Electrical Engineering series, No. 43, Issue 1, 2019, pp. 79-85. ISSN-1842-4805.

SISTEM DE CALCUL COLABORATIV PENTRU COMANDA ROBOȚILOR MOBILI

Olesea BOROZAN[1]*, Constantin ABABII[1,2], Victor LAȘCO[1], Ianuș FORTUNA[3], Vasile IONIȚA[3]

[1]*Departamentul Informatica şi Ingineria Sistemelor, Facultatea Calculatoare, Informatică şi Microelectronică, Universitatea Tehnică a Moldovei, Chişinău, Republica Moldova*
[2]*IAW Internationale Academie, Germany*
[3]*Departamentul Informatica şi Ingineria Sistemelor, grupa CR-202, Facultatea Calculatoare, Informatică şi Microelectronică, Universitatea Tehnică a Moldovei, Chişinău, Republica Moldova*

*Autorul corespondent: Olesea Borozan, e-mail: olesea.borozan@ia.utm.md

Îndrumător/coordonator ştiinţific: Victor ABABII, conf.univ.,dr., DIIS, UTM

Rezumat. *Lucrarea de față prezintă rezultatele proiectării și cercetării unui sistem de calcul colaborativ pentru comamanda în timp real a roboților mobili. Sistemul de calcul prezintă o mulțime de dispozitive pentru procesarea distribuită a datelor care sunt plasate pe platforme mobile ale roboților mobili. Fiecare robot mobil conține o mulțime de senzori care determină starea mediului de activitate și poziția acestuia în spațiu, și dispozitive MCU pentru procesarea și organizarea schimbului de date dintre roboți. În lucrare a fost elaborată: structura sistemului de caclul colaborativ; schema de ansamblu al robotului mobil; și s-au efectuat testări pentru evaluarea timpului de întîrziere a mesajelor în raport cu numărul de roboți mobili în structura sistemului.*

Cuvinte cheie*: robot mobil, calcul colectiv, inteligență de roi, Inteligența Artificială, MQTT, broker, publisher, subscriber.*

Introducere

Astăzi, este imposibil de imaginat activitatea umană fără de implicarea sistemelor robotice de diverse forme. Fie, acesta este un asistent în formă de o aplicație software, care oferă careva servicii de automatizare a proceselor de calcul, sau aceasta este o jucărie pentru un copil, sau este un sistem complex de automatizare a proceselor tehnologice și de producere. Toate exemplele prezintă sisteme robotice la baza cărora se află efectuarea unor operații automate respectîndu-se unele criterii comportamentale sau de socializare bazate pe modele de Inteligență Artificială (IA).

Caracteristica fundamentală a unui sistem robotic, este autonomia acestuia [1-3], care implică aplicarea unor decizii, sau efectuarea unor operații strict definite de algoritmul de funcționare, așa ca: scanarea mediului de activitate - prin itermediul unui set de senzori; procesarea datelor - prin aplicarea unor modele bazate pe IA; și acțiunea asupra mediului de activitate sau de mobilitate spațială a acestuia.

Un pas important în dezvoltarea sistemelor robotice mobile îl constituie posibilitatea de cooperare și colaborare dintre roboți în procesul efectuării unor sarcini complexe comune [4]. Obiectivele țintă ale procesului de cooperare sunt orientate spre a utiliza colonii de roboți inteligenți unde fiecare dintre aceștea dispune de resurse de calcul limitate. Din punct de vedere metodologic, inteligența de roi este un set de soluții euristice inspirate din comportamentele roiurilor de animale și capabile să ofere soluții empirice la multe probleme dificile din punct de vedere computațional, care aparțin mai multor domenii de aplicare. În rezultatul cooperării dintre sistemele robotice și inteligența de roi a apărut o nouă direcție de cercetare numită Swarm Robotics [5]. Domeniul științific Swarm Robotics se ocupă de coordonarea și integrarea unor grupuri mari de roboți simpli care sunt controlați folosind mijloace de calcul locale. Roiul de roboți este format dintr-o mulțime de roboți mici, care sunt autonomi și portabili. Acești roboți sunt independenți, sau pot crea o structură de calcul complexă. Costurile acestei structuri sunt reduse, iar sistemul este mult mai stabil și mai robust întrucât interacțiunea este locală și probabilitatea de eroare pentru roboții individuali este minimizată.

Conceptual, aceste roiuri sunt auto-adaptabile, auto-organizate și descentralizate. Aplicațiile potențiale ale roboticii de roi sunt sarcini care pot fi miniaturizate, necesită resurse limitate, sunt periculoase pentru oameni și roboți, cum ar fi ajutorarea în caz de dezastre, operațiuni de căutare sau aplicații militare.

La rîndul său, inteligența de roi, este inspirată din natură și se bazează pe comportamentul ființelor vii care respectă unele reguli de socializare sau viață și activitate colectivă. Până în prezent, în literatură au fost propuse mai multe modele de inteligență de roi bazate pe diferite sisteme naturale de roi și aplicate cu succes în multe aplicații din viața reală. Exemple de modele de inteligență roi sunt [6-8]: Optimizarea coloniilor de furnici, Optimizarea roiului de particule, Colonia de albine artificiale, Furajarea bacteriilor, Optimizarea roiului de pisici, Sistemul imunitar artificial etc.

În această lucrare se propune dezvoltarea unui sistem multi robot cu activitate colaborativă care să urmărească condiția de valori optime ale spațiului de activitate. Această problemă este cracteristică pentru diverse situații excepționale, căutarea sursei de poluare, căutarea zăcămintelor naturale minerale, etc.

Structura sistemului de calcul colaborativ

Sistemul de calcul colaborativ este format dintr-o mulțime de dispozitive pentru procesarea datelor plasate pe platforme de roboți mobili. Dispozitivele de procesare a datelor formează între ele o rețea comună de comunicare care asigură transferul de date în baza protocolului MQTT [9-11].

MQTT este un protocol de comunicare machine-to-machine (M2M), în special pentru aplicațiile Internet of Things. Este conceput ca un sistem de transport de mesaje, pentru publicarea și accesul la date prin abonare. MQTT este util pentru conexiunile cu locații distribuite în care este necesar un cod mimin de program cu un acces de lățime de bandă la nivel superior. Protocolul MQTT este unilizat în rețelele de senzori care comunică cu un broker pentru a distribui datele la mulțimea de abonați. Este un model ideal pentru dispozitivele mobile datorită resurselor solicitate reduse, consumului redus de energie, pachetelor de date minimizate și distribuției eficiente a informațiilor către unul sau mai mulți abonați [9, 21, 22].

Structura sistemului de calcul colaborativ [12, 13, 18, 19, 22] pentru comanda roboților mobili [17, 20] este prezentată în Figura 1, unde: **Sensor** – este mulțimea de senzori care livrează date pentru **Broker** prin funcția **Publishe**; **Application** – este mulțimea de abonați care accesează datele de pe **Broker** prin funcția **Subscribe** și le utilizează în conformitate cu algoritmul de funcționare a acestuia; **Actuator** – este mulțimea de abonați care accesează datele de pe **Broker** prin funcția **Subscribe** și le utilizează pentru a genera acțiuni asupra obiectelor sau a mediului de activitate; **MQTT Broker** – este un serviciu pe Cloud care stochează datele de la setul de senzori și le distribue către abonați. Mesajele transportate de protocolul MQTT au o structură strict definită care include numele abonatului, numele serviciului (senzor) și valoarea calitativă/cantitativă sau starea acestuia.

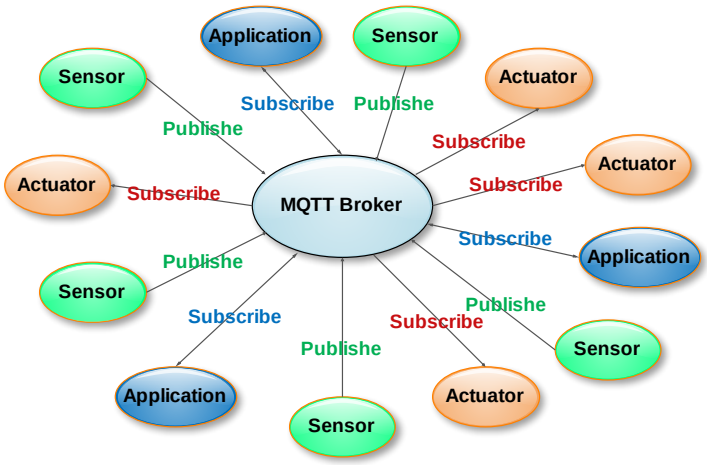


Figura 1. Structura sistemului de calcul colaborativ

Modul de funcționare a sistemului de calcul colaborativ în baza protocolului MQTT.

Un sistem în baza protocolului MQTT implică în procesul de funcționare următoarele dispozitive și servicii: Clientul, Serverul, Abanamentul, Nume subiect, Filtrul de subiecte, Sesiunea.

Clientul – este un produs program sau un dispozitiv care utilizează protocolul MQTT. Un client stabilește întotdeauna o conexiune la rețea prin care apelează serviciile oferite de server. Clientul publică pe Server mesajele de care Clienții pot fi interesați.

Serverul – este un program sau un dispozitiv care acționează ca intermediar între Clienții care publică Mesaje și Clienții care utilizează aceste Mesaje. Un server acceptă conexiuni din rețea de la Clienți și acceptă Mesajele publicate de aceștea.

Abonamentul – este un drept care validează accesul la serviciile oferite de sistem.

Nume subiect – este eticheta atașată unui Mesaj de aplicație care se potrivește cu Abonamentele cunoscute de Server. Serverul trimite o copie a Mesajului fiecărui Client care are un Abonament corespunzător.

Filtru de subiecte – este o expresie conținută într-un Abonament, pentru a indica un interes pentru unul sau mai multe subiecte.

Sesiune – este o interacțiune dintre un Client și un Server. Unele sesiuni durează doar atâta timp cât conexiunea la rețea, altele se pot dura pe mai multe conexiuni consecutive.

Sinteza robotului mobil

Schema de ansamblu al robotului mobil [17, 20] este prezentată în Figura 2, unde:

Ultra Sonic – senzor ultrasonic pentru identificarea distanței pînă la obstacolele amplasate pe direcția vectorului de deplasare a robotului mobil;

ESP32 CAM – cemara video pentru transmiterea vederii din direcția de deplasare a robotului. Imaginea video în timp real este difuzată pe un canal separat și este accesată de abonați în browser;

Gas Sensor – senzorul de gaze identifică prezența acestora în mediul de activitate al robotului mobil;

Giroscope – setul de senzori ale dispozitivului giroscop care indică direcția de deplasare și accelerația acestuia;

ESP8266 – modulul WiFi care prezintă un SOC cu protocolul TCP/IP integrat care asigură conectarea la rețeaua WLAN. Este modulul principal al sistemului care acumulează datele de la senzori și în baza protocolului MQTT le livrează la MQTT Broker;

GPS – dispozitiv pentru poziționarea robotului în sistemul de coordonate absolute Globale;

Mocrophone – senzor pentru achiziția sunetului din mediul de activitate;

T0/H Sensor – senzor de umiditate și temperatură a aerului;

LCD Display – indicator LCD pentru afișarea stării curente a robotului mobil;

Batery 1 – sursa de alimentare pentru partea electronică a sistemului de control;

Wheel – două roți pentru deplasarea robotului mobil. Viteza de rotație a acestora determină și direcția de deplasare a robotului mobil.

Arduino UNO – dispozitiv MCU realizează algoritmul de comandă cu motoarele de deplasare a robotului mobil și comunicare cu setul de senzori;

Driver DC Motor – driver pentru comanda cu motoarele DC de deplasare a robotului mobil;

DC Motor – două motoare DC pentru antrenarea roților în procesul de deplasare a robotului mobil;

Universal Wheel – roată cu rotire liberă pe planul orizontal.

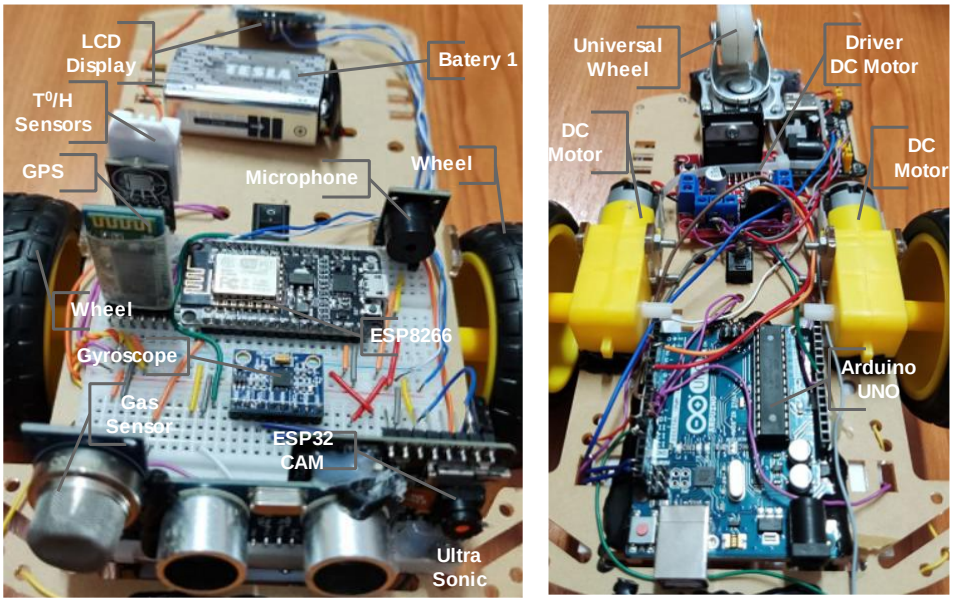


Figura 2. Schema de ansamblu a robotului mobil

Evaluarea timpului de întîrziere a mesajului

Arhitectura sistemului de calcul colaborativ pentru comanda roboților mobili prezintă o arhitectură de calcul distribuit [14, 15] care include o mulțime de dispozitive MCU pentru procesarea datelor. Activitatea robotului depinde de starea globală a mediului de activitate și de starea fiecărui robot în parte. Pentru a efectua acțiunile respective este necesar ca robotul în cauză să primească datele de la toți roboții cu care se află în colaborare. Această operație de comunicare durează în timp. Pentru a evalua timpul de întîrziere a mesajelor au fost efectuate mai multe teste practice rezultatele cărora sunt prezentate în Figura 3, unde: pe orizontală sunt indicate numărul de dispozitive MCU conectate la rețea; pe verticală este indicat timpul mediu de întîrziere (*ms*).

Excplicații pentru Figura 3:

1 – 9 MCU conectate în sistem – timpul de întîrziere a mesajelor este constant în mediu *130 ms*;

9 – 35 MCU conectate în sistem – are loc creșterea liniară a timpului de întîrziere de la *130 ms* la *1700 ms*, fenomenul este explicat prin capacitatea de deservire a punctului de acces la rețea;

35 – 40 MCU conectate în sistem – timpul de întîrziere este constant în mediu *1700 ms*.

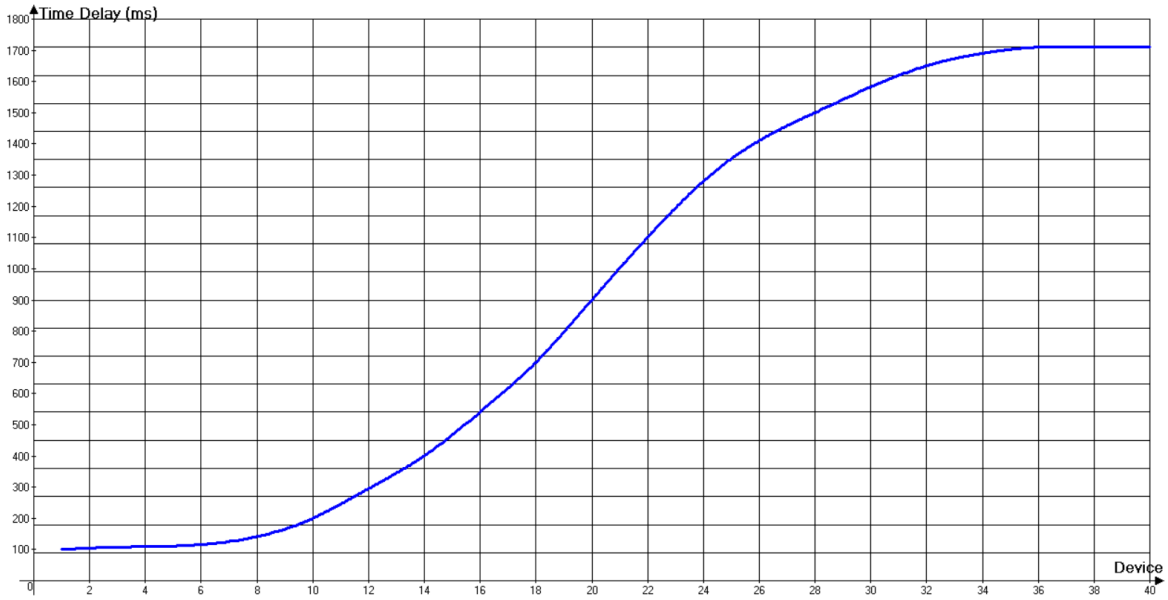


Figura 3. Timpul de întîrziere a mesajelor

Concluzii

Roboții mobili colaborativi și-au găsit aplicație practic în toate domeniile științei și tehnicii. Principala problemă care apare în sistemele multi robot este organizarea schimbului de date dintre dispozitivele implicate în procesul de calcul. Cele mai frecvente modele de colaborare ale roboților sunt inspirate din natură care simplifică esențial algoritmul de funcționare și reduce volumul de calcul.

În lucrare de față sunt prezentate rezultatul proiectării și cercetării unui sistem de calcul colaborativ care va permite comanda unei mulțimi de roboți mobili care vor efectua o activitate comună. Este prezentată structura sistemului la niver abstract care include mulțimea de clienți conectați la un server în baza protocolului MQTT. S-a elaborat schema de ansamblu al robotului mobil cu descrierea setului de senzori, elementele de acțiune, și dispozitivele pentru procesarea și organizarea schimbului de date.

Pentru evaluarea timpului de întîrziere a mesajelor în procesul transferului de date s-au efectuat măsurări rezultatele cărora sunt prezentate în forma de grafic: Numărul de dispozitive MCU – Timpul de întîrziere a mesajelor (*ms*).

Mulţumiri. Proiectarea și cercetarea sistemului de calcul colaborativ pentru comanda roboților mobili au fost realizate cu suportul tehnic și tehnologic oferit de laboratorul ”Inteligența Artificială și Sisteme Multi-Agent” care face parte din cadrul Departamentului Informatica și Ingineria Sistemelor, Universitatea Tehnică a Moldovei.

Referinţe:

1. CORELL, N. *Introduction to Autonomuos Robots: Kinematics, Perception, Localization and Planing.* 1st edition, 2016, 226p., ISBN: 978-149377-30-77.
2. HOLLAND, J. *Designing Autonomous Mobile Robots.* ELSEVER, 2004, 335p., ISBN: 0-7506-7683-3.
3. SIEGWART, R., NOURBAKHSH, I., SCARAMUZZA, D. *Introduction to Autonomous Mobile Robots.* Second Edition, MIT Press, 2011, 453p., ISBN: 978-0-262-01535-6.
4. ZOGHBY, N., LOSCRI, V., NATALIZIO, E., CHERFAOUI, V. Robot Cooperation and Swarm Intelligence, *Wireless Sensor and Robot Networks, 2013, HAL,* pp. 163-201, https://hal.archives-ouvertes.fr/hal-00917542.
5. PRASAD, S., RAWOOL, S. Swarm Robotics: Nature Inspired Systems, *I.J. of Engineering Research and General Science, Vol. 4, Issue 5, 2016*, pp. 168-174, ISSN: 2091-2730.
6. RAY, S., BISWAS, S., CHAUDHURI, S.S. Nature-Inspired Swarm Intelligence and Its Applications, *I.J. Modern Education and Computer Science, 2014, 12*, pp. 55-65. DOI: 10.5815/ijmecs.2014.12.08.
7. AHMED, H., GLASGOW, J. *Swarm Intelligence: Concepts, Models and Applications*, Technical Reports 2012-585, 2012, 50p.
8. THAKARE, S.A. Comparison of Swarm Intelligence Techniques, *I.J. of Computer Science and Business Informatics, Vol. 1, No. 1, May 2013*, pp. 1-11, ISSN: 1694-2108.
9. JoramMQ, a distributed MQTT broker for the Internet of Things. White paper and performance evaluation, V.1.2, ScalAgent, 2014, 52p. [online]. [accesat 01.03.2023]. Disponibil: https://www.oasis-open.org/news/announcements/60-day-public-review-for-mqtt-version-3-1-1-cos01-ends-september-4th.
10. BOYD, B., GAUCI, J., ROBERTSON, M.P., DUY, N.V., GUPTA, R., GUCER, V., KISLICINS, V. *Building Real-Time Mobile Solutions with MQTT and IBM MessageSight.* IBM, 2014, 250p., ISBN: 07-384-400-51.
11. SHANKAR, S., PALANIVEL, S., VENKATESWARLU, S.C., SAWMAYA, M. MQTT in Internet of Things. *International Research Journal of Engineering and Technology, Vol. 06, Issue 12, 2019*, pp.1-3. E-ISSN: 2395-0056.

12. ПОДУБНЫЙ, М., САФОНОВ, Г., НЕГАРЭ, Е., МОРОШАН И., БОРДИАН Д., АБАБИЙ В. Система коллективного принятия решений для управления мобильными роботами. *// Ежемесячный научный журнал "Евразийский Союз Ученых (ЕСУ)", 2016, № 6-2 (27)*, С. 5-8, ISSN 2411-6467.
13. ABABII, V., SUDACEVSCHI, V., SAFONOV, Gh. Designing a Collective Agent for synthesis of Adaptive Decision-Making Systems. *Sciences of Europe (Praha, Chech Republic), Vol 1, No 17(17), 2017*, pp. 70-75, ISSN 3162-2364.
14. ABABII, V., SUDACEVSCHI, V., BRANISTE, R., TURCAN, A., ABABII, C., MUNTEANU, S., Adaptive computing system for distributed process control. *International Journal of Progressive Sciences and Technologies. Vol. 22, No 2, September 2020*, pp. 258-264. ISSN 2509-0119.
15. MUNTEANU, S., SUDACEVSCHI, V., ABABII, V., BRANISTE, R., TURCAN, A., LEASHCENCO, V., Cognitive Distributed Computing System for Intelligent Agriculture. *International Journal of Progressive Sciences and Technologies. Vol. 24, No 2, January 2021*, pp. 334-342. ISSN 2509-0119.
16. MUNTEANU, S., SUDACEVSCHI, V., ABABII, V. Computer Systems Synthesis Inspired from Biologic Cells Structures. *Journal of Engineering Science, June, 2022, Vol. XXIX (2)*, pp. 91-107, ISSN: 2587-3474 / E-ISSN: 2587-3482, DOI: 10.52326/jes.utm.2022.29(2).09.
17. ABABII, V., SUDACEVSCHI, V., BRANISTE, R., NISTIRIUC, A., MUNTEANU, S., BOROZAN, O. Multi-Robot System Based on Swarm Intelligence for Optimal Solution Search. *The International Congress on Human-Computer Interaction, Optimization and Robotic Applications, HORA-2020, June 26-28, 2020, Ankara, Turkey.* pp. 269-273, Publisher: IEEE Catalog Number CFP20X32-ART, ISBN: 978-1-7281-9352-6, DOI: 10.1109/HORA49412.2020.9152926.
18. MUNTEANU, S., SUDACEVSCHI, V., ABABII, V., BOROZAN, O., BORDIAN, D., ȚURCAN, A. Коалиционная Много-Агентная Система Принятия Решений на базе Мембранных Вычислений. *IV Міжнародна науково-практична конференція "ПРИКЛАДНІ НАУКОВО-ТЕХНІЧНІ ДОСЛІДЖЕННЯ", 05 - 07 квітня 2021р., АКАДЕМІЯ ТЕХНІЧНИХ НАУК УКРАЇНИ,* Івано-Франківськ, Україна. pp. 43-45, ISBN: 978-617-7926-12-1.
19. MUNTEANU, S., SUDACEVSCHI, V., ABABII, V., BOROZAN, O., LASCO, V. Design of Embedded Collective Computing System based on Membrane Computing Models. *The 11th International Conference „Electronics, Communications and Computing"(IC | ECCO – 2021). 21 - 22 October, 2021*. Technical University of Moldova, Chisinau, Republic of Moldova, pp. 41, ISBN: 978-9975-45-776-7.
20. ABABII, V., SUDACEVSCHI, V., CARBUNE, V., MUNTEANU, S., ALEXEI, V., LASCO, V. A Method of Hardware Implementation of Membrane Computing Architecture for Mobile Robot Control. *The 16th International Conference on Development and Application Systems (DAS-2022), Suceava, Romania, May 26-28, 2022, pp. 52-56*, IEEE Catalog Number: CFP2265Y-USB, ISBN: 978-1-6654-8161-8. DOI: 10.1109/DAS54948.2022.9786079 https://ieeexplore.ieee.org/xpl/conhome/9106464/proceeding.
21. MUNTEANU, S., TURCAN, A., ALEXEI, V., SUDACEVSCHI, V., ABABII, V., CARBUNE, V., BORDIAN, D. Multi-Objective Optimal Solution Search based on Genetic Algorithms. *Proceedings of the 12th International Conference on Electronics, Communications and Computing (ECCO-2022), 20-21 October, 2022*, pp.247-252. ISBN: 978-9975-45-898-6, https://doi.org/10.52326/ic-ecco.2022/CE.07.
22. TURCAN, A., BOROZAN, O., ABABII, V., SUDAEVSCHI, V., MUNTEANU, S. Decision Making System based on Collaborative Agents. *Proceedings of the 12th International Conference on Electronics, Communications and Computing (ECCO-2022), 20-21 October, 2022*, pp.257-260. ISBN: 978-9975-45-898-6, https://doi.org/10.52326/ic-ecco.2022/CE.09.

MARKUP LANGUAGE FOR LABEL DESIGN

Călin RADU[1], Maxim COJOCARI-GONCEAR[1*], Sorin IAȚCO[1], Nichita CHIHAI[1], Andrei BARDIER[1]

[1]*Department of Software Engineering and Automatics, FAF-213, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chișinău, Republic of Moldova*

*Corresponding author: Maxim Cojocari-Goncear, maxim.cojocari-goncear@isa.utm.md

Scientific advisor/coordinator: Vasili BRAGA, university assistant, DISA, FCIM

Abstract. *This article is about Domain Specific Language (DSL) for label design, its architecture and conceptual implementation. Labels are crucial for branding and marketing, and a well-designed label can capture consumers' attention and convey important information about a product. Markup language, which refers to the visual and textual elements that comprise a label's design, plays a significant role in creating a label's overall aesthetic and communicating the desired message to consumers. This article discusses the various components of markup language, including color, typography, imagery, and layout, and how they can be used to create effective label designs.*

Keywords*: markup language, label design, DSL, grammar*

Introduction

A domain-specific language (DSL) is a language designed for usage within a given domain. A domain might be either a business or an application setting (e.g., an app extension, a plugin etc.). A DSL does not try to please everyone. Instead, it is designed for a certain set of issues and solutions, but it is powerful enough to express and handle those problems and answers. To exemplify, HTML is a DSL for web application development [1].

Generally speaking, a *label* is a little piece of cloth, paper, or plastic that is attached to an object. A label typically includes branding elements such as the company logo and colors, as well as product information such as the product name, description, ingredients, and nutritional information [2]. Labeling products not only provides crucial information and instructions to consumers but can also help your product stand out. In fact, 85% of shoppers say that their decision to buy a product is informed by reading a product's packaging while they're shopping [3].

Label design is the process of creating visual and textual elements that are printed or affixed to a product or package in order to identify and provide information about the product. The design of a label can vary depending on the product, brand, and target audience. Sometimes the label design process can be challenging, laborious and a long process. A DSL in this case, can become a handy solution to those who are looking for a specialized tool and want to save time and material resources in order to create an appropriate label. This tool can also help professionals work more efficiently, accurately, and flexibly, while also promoting standardization and collaboration within an organization or industry.

Language overview

A flexible, fully extendable, modular and user-friendly language, can be counted as the summary for DSL which we want to develop. This is a unique tool, in the field of languages, seen from the AI generated content perspective.

It is very easy to get to know the language, as it is similar to other markup languages. It has two main sections, in the *document* section: *meta* and *data*.

In the *meta section* the user defines all the auxiliary bits of information that will be bound to the output file, like the author of the file, the creation date or some short description. The document is created by the components shown in the *data section*. In each section we have a tag, with some

parameters, which dictates the component's behavior. The developed DSL is just a markup language for the label design, there are no data structures available, nor is there a way to manipulate, copy or produce data. The output will be a file, either in PDF format, or as a series of photos in .png format.

There will be no complex runtime error handling or checking, nor will there be a strict and well defined error checking system. This markup language has very easy rules, and we believe that the user gets a hand of it without much effort, and without the need of being guided by the interpreter.

Some components will have the required parameters for an AI generated image. The DSL will talk to an AI, and use the generated image as the value for the component. Extensibility and modularity will be provided to the user, so that any AI API, with some constraints could be used, if implemented.

Grammar definition

The grammar for a Domain Specific Language includes rules for defining the structure of statements, expressions, and other constructs that are specific to the domain. To enhance comprehension of the proposed DSL, its structure is presented below. Initially, the terms which the grammar utilizes need to be defined. The grammar of the proposed language is described as $G = (V_N, V_T, S, P)$ where V_N is the finite set of non-terminal symbols, V_T is the finite set of terminal symbols, S is the start symbol, and P is the finite set of production rules.

Table 1

Meta notations

Notation	Meaning
<text>	a nonterminal parameter is written between <>
text	a terminal parameter is written in bold
text*	the symbol appears zero or more times
text$^+$	the symbol appears one or more times
|	an alternative follows

V_N = {<root_component>, <parent_component>, <terminal_symbol>, <component>, <left_square_bracket, <right_square_bracket>, <equal_sign>, <digits>, <component_name>, <double_quotes>, <component_param>, <text_char>, <root_component_name>, <parent_component_name>, <component_param_name>, <component_param_value>};

V_T = {**document**, **meta**, **data**, **author**, **name**, **format**, **path**, **size**, **title**, **text**, **image**, **barcode**, **batch_number**, **date**, **[**, **]**, **/**, **=**, **"**, **-**, **[a-z]**, **[A-Z]**, **[0-9]**, **value**, **type**, **width**, **height**, **font_size**, **font_weight**, **position_x**, **position_y**, **src**, **prompt**};

S = {<root_component>};

P = {

<root_component> → <left_square_bracket><root_component_name> <right_square_bracket><parent_component>*<left_square_bracket><terminal_symbol> <root_component_name><right_square_bracket>

<parent_component> → <left_square_bracket><parent_component_name> <right_square_bracket><component>*<left_square_bracket><terminal_symbol> <parent_component_name><right_square_bracket>

<left_square_bracket> → **[**

<right_square_bracket>→ **]**

<terminal_symbol> → **/**

<equal_sign> → **=**

<double_quotes> → **"**

<text_char> → **a** | **b** | ... | **z** | **A** | **B** | ... | **Z** | **ε** | **-**

<digits> → **0** | **1** | ... | **9**

<component_name> → **author** | **title** | **text** | **image** | **barcode** | **batch_number** | **date** | **format** | **name** | **path** | **size**

<parent_component_name> → **meta** | **data**

<root_component_name0> → **document**

<component_param_name> → **value** | **type** | **width** | **height** | **font_size** | **font_weight** | **position_x** | **position_y** | **src** | **prompt**

<component_param_value> → <text_char> | <digits> | <digits><text_char>*

<component_param> → <component_param_name><equal_sign><double_quotes> <component_param_value><double_quotes>

<component> → <left_square_bracket><component_name> <component_param>*<right_square_bracket>

};

Syntax example

For a better understanding of how the DSL will look like, an example of syntax is presented in the code snippet below.

```
[document]
    [meta]
        [name value="pinot_grigio_purcari.pdf"]
        [format value="pdf"]
        [path value="C:\Users\andre\OneDrive\Desktop"]
        [size value="320KB"]
        [date value="08-03-2023"]
        [author value="Bardier Andrei"]
    [/meta]
    [data]
        [title value="Pinot Grigio de Purcari" font_size="14"
         font_weight="2"]
        [image prompt="A label logo for a bottle of Moldavian wine Pinot Grigio
de Purcari. The logo has a white background with black text and a gold border.
The text says 'Purcari' in large cursive letters at the top and 'Pinot Grigio'
in smaller serif letters below it. There is also a small emblem of a grapevine
with green leaves and purple grapes on the left side of the logo. The image is
realistic, high-resolution (768x768), and sharp with no blurriness or noise."
src="C:\Users\andre\OneDrive\Desktop\logo.png"]
        [text type="origin" value="Vin cu indicatie geografica Purcari"]
        [text type="wineType" value="Alb sec"]
        [text type="bottled" value="2013"]
        [text type="alc&vol" value="Alc. 14,0% vol. 0,75l."]
        [text type="manufacturer" value="IM 'Vinaria Purcari'"]
        [barcode]
        [batch_number]
        [date]
    [/data]
[/document]
```

Parsing tree

A parse tree is a hierarchical representation of the syntactic structure of a string of symbols or code, typically in the form of a tree or a directed acyclic graph. It is created by a parser during the process of parsing, which is the process of analyzing the syntax of a program and determining its grammatical structure.

The parsing tree for a sequence of code, according to the grammar defined in the previous section, is shown in Fig. 1.

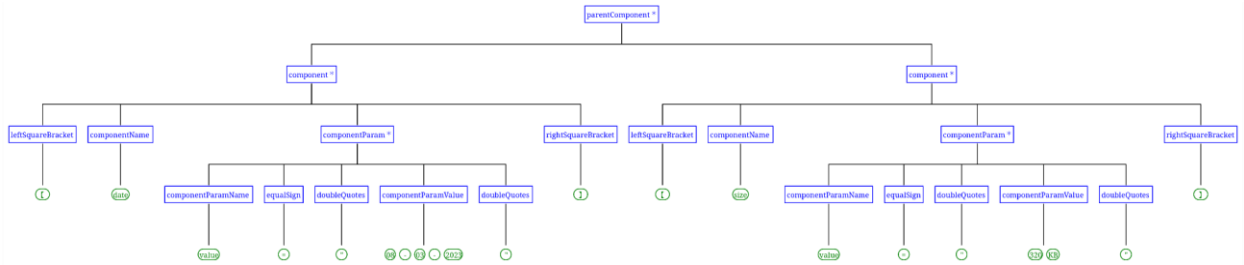


Figure 1. Parsing tree example

Conclusions

To summarize, by using a markup language described above, label designers can easily create and manage label designs with a high degree of precision and consistency. This is particularly important for businesses that need to produce large volumes of labels, as markup language makes it easy to generate labels in bulk and maintain consistency across different label designs. By using markup language, designers can also integrate dynamic data sources to populate label fields with up-to-date information. Furthermore, this DSL may be an essential tool for label design, offering numerous benefits in terms of efficiency, accuracy, and consistency. As more businesses recognize the advantages of using markup language for their label design needs, we can expect to see even greater adoption of this technology in the future.

References

1. MANAGOLI, G., *What developers need to know about domain-specific languages* [online]. 2020. [accessed 26.02.2023].
   Available: https://opensource.com/article/20/2/domain-specific-languages
2. What is a label? Definition and examples. [online]. [accessed 26.02.2023]. Available: https://marketbusinessnews.com/financial-glossary/label/
3. Why Is Product Labeling So Important? [online]. [accessed 26.02.2023]. Available: https://www.luminer.com/articles/why-is-product-labeling-important/

CREATING A GENERAL PURPOSE LANGUAGE: DESIGN AND IMPLEMENTATION OF PRISM

Valeria COZLOV[1*], Tudor SÎRGHI[1], Beatricia GOLBAN[1], Ion VASLUIAN[1]

[1]*Department of Software Engineering and Automatics, FAF-213, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chișinău, Republic of Moldova*

*Corresponding author: Valeria Cozlov, valeria.cozlov@isa.utm.md

Scientific coordinator: Alexandr VDOVICENCO, lector-assistant, DISA.

Abstract. *This article presents the development of a new general-purpose programming language, PRISM. The language was designed to be simple, intuitive, and versatile, with features that cater to both beginners and experienced programmers. The article discusses the language's grammar, data structures, and built-in functions, highlighting its potential applications and benefits for various industries.*

Key words*: General Purpose Language, Grammar, Abstract Syntax Tree (AST), Interpretation, Parse Tree.*

Introduction

Special purpose (domain specific) and general purpose languages are the two types of high-level programming languages used in computer programming. Although general purpose languages can be used in a wide range of computer applications, hardware setups, and operating systems, they are fundamentally asynchronous and non-deterministic. They enable portability by allowing the same software to execute equally on a single CPU or on a network of computers [1].

Creating a general-purpose programming language can be a complex and challenging task, but it is also a highly rewarding one. A general-purpose language is designed to be used for a wide variety of applications and can be used to write software for everything from web applications to operating systems.

The process of creating a language involves defining its syntax, grammar, and semantics, as well as designing its features and capabilities. This can be done using various tools and techniques, such as lexers, parsers, and compilers.

When designing a language, it's important to consider factors such as readability, ease of use, and performance. The language should also be designed to allow for efficient execution of programs and offer features that make it easy for developers to write and maintain code.

Overview of language implementation

One of the key features of PRISM is its simplicity. The language has a clean, easy-to-read syntax that is designed to be intuitive for developers. It also supports type inference, which makes it easier to write code without having to specify the data types of variables.

PRISM provides a rich set of data types, including, but not limited to:

- integers;
- floating-point numbers;
- strings;
- arrays.

It also supports built-in functions for common operations, such as string manipulation and mathematical calculations. Interpreting PRISM involves understanding its syntax, grammar, semantics, and features. Here are some steps that are followed to interpret this language:

1. Read the language documentation: The first step is to read the language documentation to understand the syntax, grammar, and semantics of the language. The documentation should provide examples and explanations of how the language works.
2. Understand the lexical structure: The next step is to understand the lexical structure of the language. This involves understanding the basic building blocks of the language, such as keywords, identifiers, literals, and operators.
3. Identify the grammar rules: Once the lexical structure is understood, the grammar rules of the language should be identified. This involves understanding the syntax of the language and how different components of the language fit together.
4. Identify the language constructs: The next step is to identify the language constructs, such as loops, conditionals, functions, and classes. These constructs define how code is organized and executed in the language.
5. Implement a parser: After understanding the grammar rules and constructs of the language, a parser can be implemented to parse the language code. The parser takes in the code as input and produces an abstract syntax tree (AST) as output.
6. Implement an interpreter: Once the AST is produced, an interpreter can be implemented to execute the code. The interpreter reads the AST and executes the code according to the rules of the language.

Grammar of the GPL

A mechanism is required to characterize it in order to investigate languages mathematically. Informal descriptions in English are frequently insufficient due to the imprecision and ambiguity of everyday language. Though it has limitations, the set notation is more appropriate. The concept of grammar must therefore be introduced for this reason [2].

A grammar *G* is defined as a quadruple $G = (V, T, S, P)$, where:

- V is a finite set of objects called non-terminal symbols;
- T is a finite set of objects called terminal symbols;
- $S \in V$ is a special symbol called the start variable;
- P is a finite set of productions

Notation:

<> - Nonterminal parameter;

Bold – terminal parameter

| - separator of alternatives

V = {<code>, <statement>, <variable_declaration>, <comment>, <text>, <integer_literal>, <float_literal>, <string_literal>, <boolean_literal>, <array_literal>

<expression>, <function_call>, <identifier>, <binary_operator>,

<unary_operator>, <type_name>, <function_name>, <digit>, <equality_operator>, <arithmetic_operator>, <comparison_operator>, <condition_operator>, <location>, <block>}

T = { if, else, from, for, to, downto, while, int, float, string, boolean, not, and, or, //, -, ==, !=, +, -, *, /, %, >, <, <=, >=, true, !, ., &&, ||, false, “ASCII characters”, {, },[,], a , A, b, B, c, C, d, D, e, E, f, F, g, G, h, H, I, I, j, J, k, K, l, L, m, M, n, N, o, O, p, P, q, Q, r, R, s, S, t, T, u, U, v, V, w, W, x, X, y, Y, z, Z, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

P = {<code → <statement>[<comment><text>]

<statement> → <variable_declaration> |<location> = <expression> | <function_call>| if<expression><statement> [else <statement>] | **for** <identifier> **from** <expression> [to, downto]*<expression> <statement> | **while** <expression> <statement>

<expression> → <unary_operator> <expression>| (<expression>)

<block> → {<statement>}

<variable_declaration> → <type_name> <identifier>= <expression> | <function _call>

<function_call> → <function_name> ([<expression>])

<identifier> → <letter>*|<digit>*
<type_name> → **int** | **float** | **string** | **boolean** | [[ ]]
<function_name> → <identifier>
<location> → <identifier>
<binary_operator> → <equality_operator>| <comparison_operator>| <condition_operator>| <arithmetic_operator>
<text> → <string_literal>
<comment>→ //
<unary_operator> → **not** | **-**
<equality_operator> → **==** | **!=**
<arithmetic_operator> → + | **-** | ***** | **/** | **%**
<comparison_operator> → **>** | **<** | **<=** | **>=**
<condition_operator> → **and** | **or**
<literal> → <integer_literal>|<boolean_literal>|<float_literal> |<string_literal> |<array_literal>
<integer_literal> → [-] <digit>
<boolean_literal> → **true** | **false**
<float_literal> → <integer_literal>. <digit>*
<string_literal> → "ASCII characters"
<array_literal>→ [<expression>]
<letter> → **a | A | b | B | c |C | d | D | e | E | f | F | g | G | h | H | i | I | j | J | k | K | l | L | m | M | n | N | o | O | p | P | q | Q | r | R | s | S | t | T | u | U | v | V | w | W | x | X | y | Y | z | Z**
<digit> → **0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9**}

Input and output

The code for the PRISM program must be written in a file with the .psmx extension or directly in the command line for the Abstract Syntax Tree Interpreter to run the program.

The command line will show the direct output of a specific .psmx file. An executable file won't be produced because PRISM is an interpreted language and these files are only compatible with compiled programming languages like C or C++.

Conclusion

In conclusion, creating a new general-purpose language is a complex and challenging task that requires a deep understanding of programming concepts, syntax, and semantics. PRISM, the newly created language, is designed to be easy to learn and use, with a modern syntax and a rich set of features that enable developers to create powerful and efficient applications. With its focus on simplicity, clarity, and flexibility, PRISM has the potential to become a popular language among developers who value productivity and expressiveness.

References

1. GÉRARD BERRY *Real time programming: special purpose or general purpose languages* [online], 1989, [cite: 06.03.2023] Available: https://hal.inria.fr/inria-00075494/document
2. PETER LINZ *An introduction to formal languages and automata* [online], 2012, [cite: 06.03.2023] Available: https://fall14cs.files.wordpress.com/2017/04/an-introduction-to-formal-languages-and-automata-5th-edition-2011.pdf

DOMAIL SPECIFIC LANGUAGE FOR COMPUTING TAXES

Diana BELAIA[1*], Corneliu CATLABUGA[1], Ana BZOVII[1], Igor MIHAILUIC[1], Constantin BAXANEAN[1]

[1]*Department of Software Engineering and Automation, group FAF-213, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chisinau, Republic of Moldova*

*Author correspondent: Diana Belaia, diana.belaia@isa.utm.md

Scientific coordator: Irina Cojuhari, conf. univ. dr., Gabriel Zaharia, univ. assist.

Abstract: *The aim of the following article is to describe the notion of taxes, the rules of computations in different countries and one of possible ways to develop a Domain Specific Language (DSL) to make computing taxes easy for an average user. It contains some general information of taxes to make readers familiar with this field. Further, it explains how DSL works, the identified problem which is solved by DSL and it is presented the grammar.*

Keywords*: taxes, Domain Specific Language, grammar, problem.*

Introduction

Computing taxes is the process of calculating how much tax an individual or organization owes to the government based on their income, assets, or other factors. The specific method of computing taxes varies depending on the country and tax system in question, but in general, it involves the following steps:

1. **Determine taxable income:** Taxable income is the amount of income that is subject to taxation. It is usually calculated by subtracting certain deductions and exemptions from gross income.
2. **Determine tax rate:** The tax rate is the percentage of taxable income that must be paid as tax. The tax rate can vary depending on the type of income, the taxpayer's income level, and other factors.
3. **Calculate tax liability:** To calculate tax liability, multiply taxable income by the applicable tax rate. Some tax systems also have additional taxes or fees that must be added to the tax liability.
4. **Apply credits and deductions:** Tax credits and deductions can reduce the amount of tax liability owed. Tax credits are typically dollar-for-dollar reductions in the amount of tax owed, while deductions reduce taxable income.
5. **File tax returns:** Taxpayers are required to file tax returns with the government to report their income, deductions, and tax liability. Tax returns are usually due on an annual basis, although some systems may require more frequent reporting.

It is important to note that computing taxes can be a complex process, and mistakes or omissions can result in penalties or legal problems. Many individuals and organizations choose to use tax software or hire professional tax preparers to help ensure accuracy and compliance with tax laws.

Taxes are mandatory contributions levied on individuals or corporations by a government entity (local, regional or national). From an accounting perspective, there are various types of taxes [1]:

1. **Income taxes:** A percentage of generated income that is relinquished to the state or federal government
2. **Corporate taxes:** A percentage of corporate profits taken as tax by the government to fund federal programs
3. **Estate taxes:** rate is applied to the fair market value of property in a person’s estate at the time of death; the total estate must exceed thresholds set by state and federal governments

4. **Tariff:** Taxes on imported goods; imposed with the aim of strengthening domestic businesses
5. **Property taxes:** Based on the value of land and property assets
6. **Sales taxes:** Taxes levied on certain goods and services; varies by jurisdiction [2].

Taxes in different countries:

Computing taxes can vary from country to country since each nation has its own tax system and rules. Here is some general information about computing of taxes in different countries:

1. **United States:** income taxes are calculated based on a progressive tax system, meaning the tax rate is increasing as income is increasing. The Internal Revenue Service collects taxes on income, capital gains and dividends.
2. **United Kingdom:** taxes are collected by HM Revenue and Customs. The tax rate is based on a progressive tax system and the rates are different for different types of income.
3. **Canada:** the federal government and the provincial government both collect income taxes. The federal tax rate is based on the progressive rate system and each province has its own tax rates.
4. **Moldova:** the personal income tax rate in Moldova is progressive with rates ranging from 12% and 18% depending on the amount income earned.

The problem of computing taxes and DSL benefits

Nowadays computing of taxes can become a real problem for those who want to start their own business or for people who are not familiar with economics and law. Since there are a lot of tax rate systems and these systems vary from country to country it would be a saving solution to develop a Domain Specific Language with user friendly interface which would allow us to compute the amount of the tax for our income, to check the correctness of computations offered by special instances. Moreover, one of the advantages for those individuals who develop their own small business will be a possibility not to have an accountant or a special person who is usually responsible for this part of finance in the company.

Language overview:

The DSL for computing taxes is designed to provide a streamlined and user-friendly solution for tax preparation. The language uses several basic data structures to manipulate and store data, including strings for names, rates for tax rates between 0 and 1, percentages for tax rates between 0 and 100 with % at the end, and positive floats for income. Users can create and manipulate data by specifying the appropriate data structure and assigning values to them.

The DSL also includes basic control structures for specifying tax brackets and computing tax deductions. Users can declare tax brackets with multiple income ranges, each with an assigned tax rate or percentage. After the brackets are declared, users can specify the target income and any applicable deductions to compute the required tax deduction. Control flow is manipulated by specifying the appropriate brackets and income ranges.

Programs in the DSL require input such as the income and tax bracket ranges, as well as any applicable deductions. The output of the program is the required tax deduction based on the income, bracket, and deductions specified by the user.

The language also includes error handling mechanisms to prevent common errors. Programs can go wrong if the range declarations do not start from 0 or end at the maximum value, or if the ranges overlap. The language communicates errors to the user through error messages and warnings, helping users to identify and fix errors in their code.

Overall, the DSL for computing taxes is designed to simplify tax preparation and provide a more efficient and reliable solution for individuals and businesses. By providing a user-friendly interface and comprehensive error handling, the language aims to make tax preparation easier and more accessible for everyone.

Syntax

Syntax of programming language is considered to be notations and rules to control the structure. Nowadays all scripts and programs which are supposed to compute the taxes are written in compiled programming languages since it is faster. Our goal is to develop a programming language the syntax pretty similar to Python syntax since it is easy to understand and use even for people who do not have deep knowledge of mathematics and do not want to spend time on learning the usage of DSL, its specific structure.

Grammar of DSL

To better understand the grammar of a particular programming language it is needed to refer to some general notions. The notations necessary for grammar:

V_n – the finite set of non-terminal symbols.

V_t – the finite set of terminal symbols which, also, defines the alphabet of the language

S – the starting symbol

P – the finite set of productions (allowed combinations to generate the strings)

DSL grammar:

V_n = {
<program>, <newtaxbracket>, <newrange>, <text>, <character>, <real>, <integer>, <rate>, <percentage>, <letter>, <symbol>, <digit>, <newline>, <newtaxcompute>, <deductions>, <charity>, <standart>
}

V_t = {
tax_bracket, range, tax_compute, amount, deductions, charity_deduction, standart_deduction, a..z, A..Z, 0..9, max, ., -, <, >, =, %, ", \, /, @, #, !, ?, +, *, $, NEWLINE, IDENTATION
}

S = {<program> }

P = {
<program > -> <newtaxbracket> NEWLINE <newline> <newtaxcompute>
<newtaxbracket> -> tax_bracket "<text>" NEWLINE <newline> <newrange> | tax_bracket "<text>" NEWLINE <newline> <newrange> <newbracket>
<newline> -> NEWLINE | NEWLINE <newline> | ε
<text> -> <character> | <character><text> | ε
<character> -> <letter> | <symbol> | <digit>
<real> -> <integer>.<integer> | <integer> | max
<integer> -> <digit><integer> | <digit>
<rate> -> 0.<integer> | 1 | 0
<percentage> -> <digit><digit>.<integer> % | 100 % | 0 %
<letter> -> a | A | b | B | ... | z | Z
<symbol> -> . | ! | ? | , | : | ; | - | < | > | @ | # | % | \ | / | + | * | $
<digit> -> 0 | 1 | 2 | ... | 9
<newrange> -> IDENTATION range <real>..<real> -> <percentage> NEWLINE <newline> | IDENTATION range <real>..<real> -> <rate> NEWLINE <newline> | IDENTATION range <real>..<real> -> <percentage> NEWLINE <newline> <newrange> | IDENTATION range <real>..<real> -> <rate> NEWLINE <newline> <newrange>
<newtaxcompute> -> tax_compute "<text>" NEWLINE <newline> IDENTATION bracket = "<text>" NEWLINE <newline> IDENTATION income = <real> <newline> | tax_compute "<text>" NEWLINE <newline> IDENTATION bracket = "<text>" NEWLINE <newline> IDENTATION income = <real> NEWLINE <newline> | ε

<deductions> -> IDENTATION deductions NEWLINE <newline> <charity> <standart> | IDENTATION deductions NEWLINE <newline> <charity> | IDENTATION deductions NEWLINE <newline> <standart> | ε
<charity> -> INDENTATION INDENTATION charity_deduction = <float> NEWLINE <newline>
<standart> -> INDENTATION INDENTATION standart_deduction = <float> NEWLINE <newline>
}

Syntax description:

The program file should start by declaring all the tax brackets that might be used later in the program. Each tax bracket should have a unique name incased in double quotation marks ["…"], used for identification. After name declaration the user should define one or multiple ranges, where the max-income has its specific tax rate, in increasing order, the last range having a max income and to each range being assigned a rate [x.y…] or percent [x.y… %] value.

Following the tax bracket(s) declaration(s) the user should declare the tax compute(s) target(s), each having assigned a name, brackets in which that tax compute should be classified and income. Optionally the user can declare the deductions where standard and/or donation deductions are specified.

The program should output the tax compute target along its required tax.

Code Example

```
tax_bracket "moldova:mdl"
    range 1000 -> 0.07
    range 3000 -> 0.08
    range 5000 -> 0.1
    range max -> 0.12

tax_bracket "romania:eur"
    range 500 -> 7 %
    range 1000 -> 0.09
    range max -> 11.23 %

tax_compute "Alex"
    bracket = "moldova"
    income = 1170

tax_compute "Ion"
    bracket = "moldova"
    income = 3053

    deductions
        donation_deduction = 150

tax_compute "Alan"
    bracket = "romania"
    income = 782

    deductions
        standart_deduction = 72
```

Figure 1. Code example

Parse tree:

Parsing tree represents the detailed way to generate the code using allowed productions. On Figure 1 the part of parsing tree for DSL for computing taxes is presented:

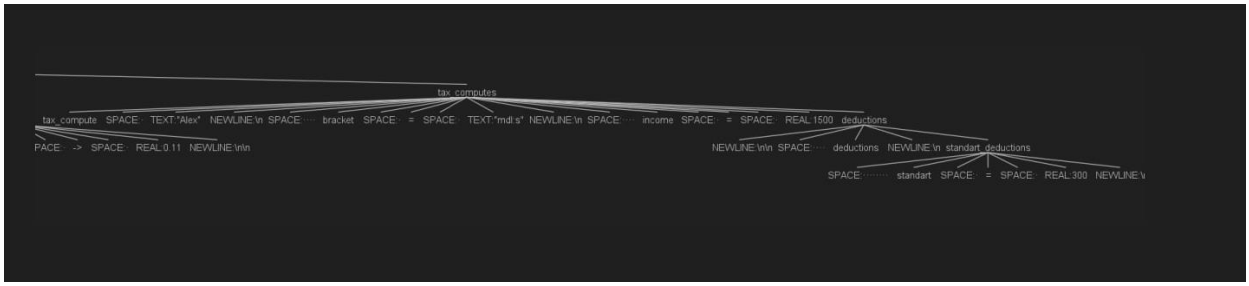


Figure 2. Parsing tree example

Conclusions

Domain Specific Language for computing of the taxes is a good solution to unite the income taxes from most part of the countries and to make tax computation easier for bisiness. Despite on the narrow field of application, DSL for taxes can help to improve people's knowledge about the taxes, economics and law in general. Moreover, DSL can reduce the number of specialists who are needed to be hired to solve the computational problems and this is the greate advantage for bussiness since we have to pay less.

References:

1. Basis of tax computation. [online]. [accessed 24.02.2023]. Available: https://www.accountingnotes.net/international-taxation/domestic-tax/basis-of-tax-computation/7469
2. Preparing a Tax Computation. [online]. [accessed 25.02.2023]. Available: https://www.iras.gov.sg/taxes/corporate-income-tax/form-c-s-form-c-s-(lite)-form-c-filing/preparing-a-tax-computation
3. Domain Specific Language. [online]. [accessed 10.03.2023]. Available:https://stackoverflow.com/questions/809574/what-is-a-domain-specific-language-anybody-using-it-and-in-what-way

FRACLANG - A DOMAIN SPECIFIC LANGUAGE FOR BUILDING FRACTALS

Ciprian BOTNARI[1*], Grigore GUZUN[1], Emanuil CUCOȘ[1], Artiom MIROVSKI[1]

[1]*Department of Software Engineering and Automatics, Group FAF-213, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chișinău, Republic of Moldova*

*Corresponding author: Ciprian Botnari, ciprian.botnari@isa.utm.md

Scientific coordinator: Mariana CATRUC, University Lecturer

Abstract. *Fractals are mathematical structures that are fascinating due to their self-similar patterns, which repeat at different scales. However, building fractals requires a considerable amount of mathematical knowledge and expertise. To simplify the process, domain-specific languages (DSLs) have been developed that allow users to create fractals using simple syntax and commands. This article discusses a domain-specific language that is flexible and convenient for building fractals, FracLang.*

Keywords*: domain specific language, fractal, fractal language, grammar, parse tree*

Introduction

Mathematicians and computer scientists have been fascinated by fractals for many years. Several natural phenomena, including coastlines, clouds, snowflakes, and even the human body, contain fractals. They have also been utilized to produce stunning animations and digital art. Fractal geometry is an area of mathematics that examines objects that exhibit self-similarity at various scales [1].

Mathematicians and computer scientists have created algorithms that iteratively apply a set of rules to a geometric object in order to produce fractals. To create a new version of the item, these rules specify how the object should be rotated, scaled, and modified. Recursively repeating the technique results in a string of items that eventually congregate into a fractal shape. Domain-specific languages (DSLs) have been created by programmers to make it easier for users to explain fractal algorithms in a clear and understandable manner.

Language Design

A DSL is a programming language designed for a specific domain and is created to simplify programming tasks in a specific field, and has a lower learning curve than a general-purpose programming language [2]. DSLs can be either external or internal. External DSLs have their own syntax and are often compiled to a target language. Internal DSLs are embedded within a host language and use the syntax and structure of the host language [3]. FracLang is a language that enables users to create fractals with minimal coding as it has a set of predefined statements that are easy to use and learn, and addresses the challenges associated with building fractals.

Grammar

Grammar in computer science refers to the body of rules that specify a language's syntax. It outlines the proper combinations of a language's various building blocks, including its keywords, operators, variables, and expressions [4].

Formal notations like Backus-Naur Form (BNF) or Extended Backus-Naur Form (EBNF), which offer a succinct approach to define a language's syntax, can be used to represent the grammar of a language. Compilers and interpreters employ grammar to parse and evaluate source code because grammar defines a computer language's structure.

Table 1

Grammar notations

Notation	Description
<foo>	foo is nonterminal
foo	**foo** is terminal
|	separates alternative
→	derives into

$G = (V_N, V_T, P, S)$
V_N - finite set of non-terminal symbols.
V_T - finite set of terminal symbols.
P - finite production rules.
S - start symbol.
S = { <program> }
V_N = { <program>, <statement>, <size_statement>, <color_statement>, <angle_statement>, <iterations_statement>, <shape_statement>, <move_statement>, <scale_statement> <rotate_statement>, <mirror_statement>, <axis>, <draw_statement>, <save_statement>, <filename> }
V_T = { **repeat**, **times**, **start**, **with**, **shape**, **circle**, **square**, **triangle**, **polygon**, **color**, **background**, **scale**, **rotate**, **save**, **as**, **PNG**, **JPG**, **[A-Z]**, **[a-z]**, **[0-9]**, =, **.**, **,**, **[**, **]** }

P = { <program> → <statement> | <statement> <program>
<statement> → <size_statement> | <color_statement> | <angle_statement> | <iterations_statement> | <shape_statement> | <move_statement> | <scale_statement> | <rotate_statement> | <mirror_statement> | <draw_statement> | <save_statement>
<size_statement> → **size** <value>
<color_statement> → **color** <value>
<angle_statement> → **angle** <value>
<iterations_statement> → **iterations** <value>
<shape_statement> → **shape** <shape>
<move_statement> → **move** <value> <value>
<scale_statement> → **scale** <value>
<rotate_statement> → **rotate** <value>
<mirror_statement> → **mirror** <axis>
<axis> → **x** | **y**
<draw_statement> → **draw**
<save_statement> → **save** <filename>
<filename> → <string>
<shape> → **circle** | **square** | **triangle** | **polygon**
<value> → <digit> | <digit> <value> | <string>
<digit> → **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9**
<string> → <char> | <char> <string>
<char> → [**A-Z**] | [**a-z**] | [**0-9**] | **=** | **.** | **,** | **[** | **]** | ' ' | '_'
}

Program showcase

The program generates a fractal image with a red color scheme and a square shape, using an angle of 45 degrees and three iterations of the fractal algorithm. The fractal image is moved to the coordinates (100, 100) on the screen, scaled to half its original size, and rotated 90 degrees. The image is then mirrored along the y-axis before being drawn to the screen. Finally, the resulting image is saved as a PNG file named "fractal.png".

size 800
color red
angle 45
iterations 3
shape square
move 100, 100
scale 0.5
rotate 90
mirror y
draw
save fractal.png

Parsing tree

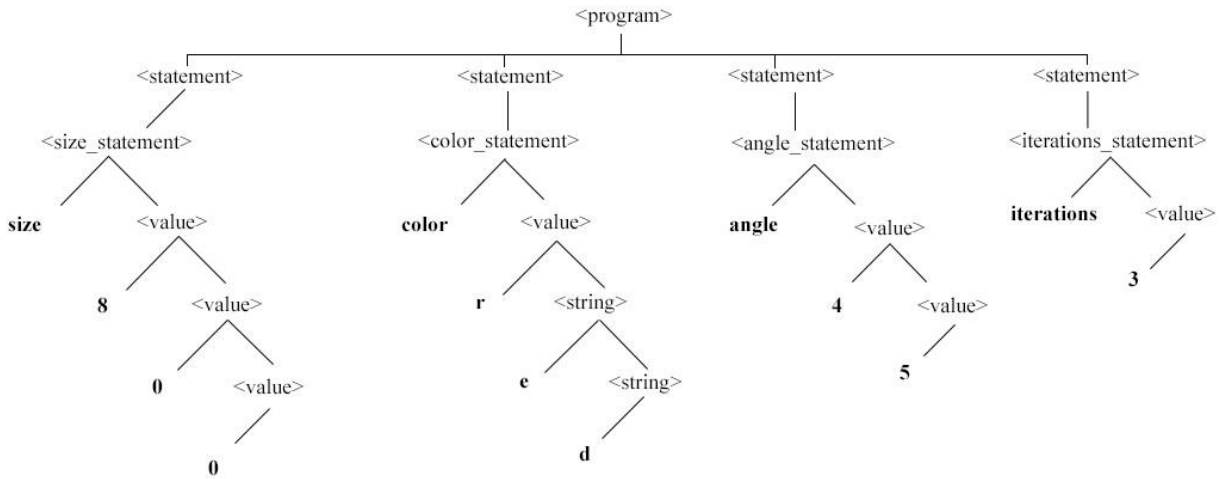


Figure 1. Parsing tree

In the context of domain-specific languages, the parse tree for FracLang can be quite extensive due to the presence of numerous statements. To alleviate the cognitive burden of reading such complex trees, a limited display of statements is proposed. The starting point for FracLang is the <program> production, which allows for the addition of an infinite number of statements. The <statement> non-terminal symbol contains several productions, including the <size_statement>, <color_statement>, <angle_statement>, <iterations_statement>, <scale_statement>, and <rotate_statement>. Each of these statements has similar construction rules and serves a specific purpose in the construction of fractals.

For instance, the <color_statement> determines the color of the fractal by accepting a string representing a color name or a tuple of three integers representing RGB values. The <iterations_statement> is used to control the number of repetitions of the fractal pattern. Too few iterations may mask the fractal's intricate beauty, whereas too many iterations may place a significant strain on the device hardware. Hence, it is important to strike a balance between the iterations and the complexity of the fractal design.

Conclusion

FracLang is an innovative domain-specific language that is specifically designed to make it easier for both experts and novices to generate stunning fractals. With its intuitive grammar and user-friendly interface, The language is accessible to users with varying levels of experience, making it an ideal tool for anyone interested in exploring the fascinating world of fractals.

What makes FracLang stand out from other programming languages is its ability to simplify the complex task of generating fractals. FracLang achieves this by providing a set of easy-to-use

commands that allow users to create complex fractal patterns with ease. This makes it possible for anyone, regardless of their experience level, to create intricate and beautiful fractal designs without having to learn complex programming languages.

It is just one of many domain-specific languages developed to simplify complex tasks in various fields. Its success serves as a testament to the power of DSLs in making complex tasks more accessible to a wider audience. As technology advances, it is likely that we will see more DSLs developed for other fields, making it easier for people to accomplish complex tasks and achieve their goals.

In conclusion, FracLang is a powerful and innovative tool that simplifies the process of generating fractals for both experts and novices. Its intuitive grammar and user-friendly interface make it accessible to users of varying levels of experience. As the demand for DSLs continues to grow, it is exciting to see what new innovations and discoveries will be made possible in the future.

References

1. MANDELBROT, B. *The fractal geometry of nature*. Times Books, 1982
2. MERNIK, M., HEERING, J., & SLOANE, A. M. When and how to develop domain-specific languages. In: *ACM Computing Surveys* (CSUR), 2005, 37(4), 316-344.
3. FOWLER, M. *Domain-specific languages*. Addison-Wesley Professional, 2010
4. AHO, A. V., LAM, M. S., SETHI, R., & ULLMAN, J. D. *Compilers: principles, techniques, and tools. Pearson Education.* Addison-Wesley, 2006

DOMAIN SPECIFIC LANGUAGE FOR UNIT TESTING

Andreea CHIPER[1], Denis SMOCVIN[1], Cezar GUZUN[1], Vladimir LUCHIAOV[1], Anatolie TELUG[1]

[1]*Department of Software Engineering and Automation, group FAF-212, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chisinau, Moldova*

*Corresponding author: Andreea Chiper, andreea.chiper@isa.utm.md

Scientific coordinator: Vasile DRUMEA, university assistant, DISA

Abstract. *The purpose of this article is to present a project based on the development of a domain specific language for unit testing. Unit testing is the process of testing individual pieces of code. Even while it is incredibly useful, it has a few drawbacks that can be solved by developing a domain-specific language. It will be related about the importance of a domain specific language in unit testing and how to create one so that users can get the most out of it.*

Key words*: domain specific language, bugs, developers, module, parser, grammar, suite.*

Introduction

A domain specific language (DSL) is a collection of functions named after intuitively obvious behaviors in a certain environment. A customized language that focuses on a certain issue domain. It defines the problem at a greater level of abstraction than general purpose languages. It simply presents another angle on the same issue.

Unit testing is a testing technique that isolates individual modules to find, analyze, and remedy any faults [1]. The goal is to ensure that each unit of software code operates as planned. The developers test an application throughout the development named code phase. Unit tests isolate and check the validity of a section of code. A unit is a single function, method, procedure, module, or object.

By decreasing errors, enhancing architectural conformance, and increasing maintainability, domain specific languages can improve product quality. It can help specialists write more expressive and legible test cases, reduce repetitive code, enforce domain-specific rules, and improve communication and cooperation. It streamlines the writing, management, and execution of tests, resulting in higher quality, consistency, and automation.

Domain analysis of unit testing

Testing is a wide and important domain in application development. Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation [1].

Unit testing is important because software developers often try to save time by performing minimal unit testing; however, this is a myth because insufficient unit testing leads to high-cost defect fixing during system testing, integration testing, and even beta testing after the application is built [2]. Proper unit testing during early development saves time and money in the long run.

The following are the primary reasons for performing unit testing in software engineering:

- Unit tests aid in the early detection of errors and the reduction of development costs.
- It allows developers to quickly comprehend and improve the testing code base.
- Unit tests help with code reuse.

As any other type of product, when developing a language, it has to be taken into consideration the potential users that will be interested in the service.

As it may be believed, only testers are going to use it because their job involves testing the developed software product and unit tests are an often used technique for doing tests.

Sometimes a team doesn't have a tester, which means that one software developer or a group of developers will take on the responsibility to perform automated tests on the developed software [3].

Similar to software developers, library developers, when working on a library its individual units should be tested to ensure that the many users it will have will not have any problems, errors or crashes while using the library.

Although it is not usually the case, cybersecurity specialists may write unit tests to validate the behavior of security-critical components in the code, such as encryption and authentication modules.

Unit testing challenges that a DSL could solve

Unit testing is the process of evaluating individual software modules or components to ensure they work as expected. A domain specific language can help unit testing in a variety of ways, as specified below.

Improved readability: a DSL for unit testing can offer a shorter, clearer syntax, making it simpler to design, comprehend, and manage tests. This can improve the accuracy and reliability of unit tests and make it simpler to find and address problems [4].

Abstraction: a DSL can provide a level of abstraction that makes testing complicated or challenging to understand code simpler.

Customization: it is simpler to develop tests that are pertinent and useful for a given context when a DSL has been adapted to the particular requirements of a project or team. This can enhance the efficacy and quality of unit tests and point out areas where the code should be better managed.

Improved teamwork: a DSL can make it simpler for members of the team to work together on unit testing.

Improved automation: by offering a clear and uniform syntax that automated testing tools can quickly parse, a DSL can make it simpler to automate unit testing. This makes it simpler to detect faults and regressions early in the development cycle and can speed up, reliably, and scale unit tests.

Basic data structures in DSL

A domain-specific language for unit testing often has a number of data structures that aid in the definition and organization of test cases. The following are some of the most commonly used data structures in unit testing DSLs:

A test case is a unit of testing that represents a single set of input values and expected output for a particular function or feature that is being tested. In general, test cases are stated in terms of input data, expected output, and any additional context or requirements that must be met.

A test suite is a collection of test cases that are organized according to their purpose, function, or area of concentration. Test suites allow you to organize tests into logical groups and manage huge numbers of test cases more easily.

A statement that outlines an expected condition or behavior for a certain test case is known as an assertion. Assertions are used to ensure that the actual results of a test correspond to the expected results and are frequently defined using a specific syntax or function within the language.

A fixture is a set of preconditions that must be met in order to execute a certain test case. Fixtures are used to build up the environment or context in which a test case will be conducted and can contain things like the initialization of variables, database connections, or other resources.

Mock object is a simulated object that is used to test a specific feature or behavior by replacing a genuine item. Mock objects are often used to isolate specific components of an application or system, and their methods, properties, and intended behavior are described within the DSL.

Grammar overview

Domain specific languages can be used to simplify the process of writing unit tests by providing a custom syntax that is tailored to the needs of the software being tested. It will be introduced a grammar for a DSL that can be used for unit testing, exploring its syntax and capabilities, and demonstrating how it can be used to write effective and efficient unit tests [5].

Firstly, it will be introduced with a bunch of examples that show how the DSL will work for some base cases. Then it will be presented the grammar that is used to parse the presented code.

In Figure 1 it can be seen some examples that are already possible to be parsed in the DSL

```
Suite mySuite

Test foo -skip -repeat 5 times
When param1=1
Then result should be 1

Test foo2
When param2=True, param3="John"
Then result should be empty

Test foo3 -skip
When no parameters
Then result should be 1

Execution order: foo1, foo3, foo2
```

Figure 1: DSL code examples

First, notice the **Suite** keyword. A suite defines a collection of functions. This DSL is for the Python programming language. It will search for the file that contains the name of the suite. It will then search for functions in this file.

Test *functionName* [flags]

Notice that Test is bolded, meaning it is a keyword. The following is an identifier of a function. The interpreter will search for the function in the file specified in the suite definition. This function will be tested in this test. Next, optionally, flags can be specified. Using the flags for skipping a test (*-test*), for repeating a test multiple times (*-repeat N times*).

After specifying a test's name and flags, there is the **When** [params]. This is the line that specifies the parameters of the function that will be called. Parameters are specified as *paramName=paramValue.* Notice that if a function doesn't require any parameters at all, then you can just type *no parameters* and the function will be called without any parameters.

Then result should be [value] - specifies the expected return value of the function call. Also, no return value (void function) still works. It is required to type *empty* to mention that no value should be returned.

The keyword *empty* when specifying the result value is interchangeable with *Empty, void, Void, none and None*. Similarly, the *no parameters* is interchangeable with *None, none, Void and void*.

Lastly, it can be specified the execution order of your tests. This is done by typing *Execution order:* [functionName1], [functionName2] and so on.

The following is a formal definition of the grammar in Backus-Naur Form (BNF):

V_T = {
<letter>,
<digit>,
<STRING>
<BOOL>,
<EmptyParameters>,
<EmptyResult>
},

```
V_N = {
        <ID>, <INT>, <FLOAT>, <NUMBER>,
        <Suite>, <Test>, <Result>, <Parameter>,
        <Datatype>, <Flag>, <Repeat>, <Order>
}

<letter> ::= "A" | "B" | ... | "Z" | "a" | "b" | ... | "z"
<digit> ::= "0" | "1" | ... | "9"
<ID> ::= <letter> (<letter> | <digit>)*
<STRING> ::= "" (~"")* ""
<INT> ::= <digit>+
<FLOAT> ::= <digit>+ "." <digit>+
<NUMBER> ::= <INT> | <FLOAT>
<BOOL> ::= "True" | "False"
<Suite> ::= "Suite" <ID> <Test>* <Order>?
<Test> ::= "Test" <ID> <Flag>* "When" <Parameter>+ <Result>
<Result> ::= "Then" ("result" "should" "be")? (<Datatype> | <EmptyResult>)
<Parameter> ::= (<ID> "=" <Datatype>) | <EmptyParameters>
<Datatype> ::= <STRING> | <INT> | <FLOAT> | <NUMBER> | <BOOL>
<EmptyParameters> ::= "no parameters" | "None" | "none" | "void" | "Void"
<EmptyResult> ::= "empty" | "Empty" | "void" | "Void" | "none" | "None"
<Flag> ::= "-" ("skip" | "Skip" | <Repeat>)?
<Repeat> ::= ("repeat" | "Repeat") <INT> ("times" | "Times")
<Order> ::= "Execution order:" <ID> ("," <ID>)*
```

The language is defined as follows:

$L(G) = (V_N, V_T, P, S)$, where

```
S = { <Suite> },
P = {
        <letter> => "A" | "B" | ... | "Z" | "a" | "b" | ... | "z"
        <digit> => "0" | "1" | ... | "9"
        <ID> => <letter> (<letter> | <digit>)*
        <STRING> => "" (~"")* ""
        <INT> => <digit>+
        <FLOAT> => <digit>+ "." <digit>+
        <NUMBER> => <INT> | <FLOAT>
        <BOOL> => "True" | "False"
        <Suite> => "Suite" <ID> <Test>* <Order>?
        <Test> => "Test" <ID> <Flag>* "When" <Parameter>+ <Result>
        <Result> => "Then" ("result" "should" "be")? (<Datatype> | <EmptyResult>)
        <Parameter> => (<ID> "=" <Datatype>) | <EmptyParameters>
        <Datatype> => <STRING> | <INT> | <FLOAT> | <NUMBER> | <BOOL>
        <EmptyParameters> => "no parameters" | "None" | "none" | "void" | "Void"
        <EmptyResult> => "empty" | "Empty" | "void" | "Void" | "none" | "None"
        <Flag> => "-" ("skip" | "Skip" | <Repeat>)?
        <Repeat> => ("repeat" | "Repeat") <INT> ("times" | "Times")
        <Order> => "Execution order:" <ID> ("," <ID>)*
}
```

In Figure 2 it is represented the meta-model for the language as generated by TextX.

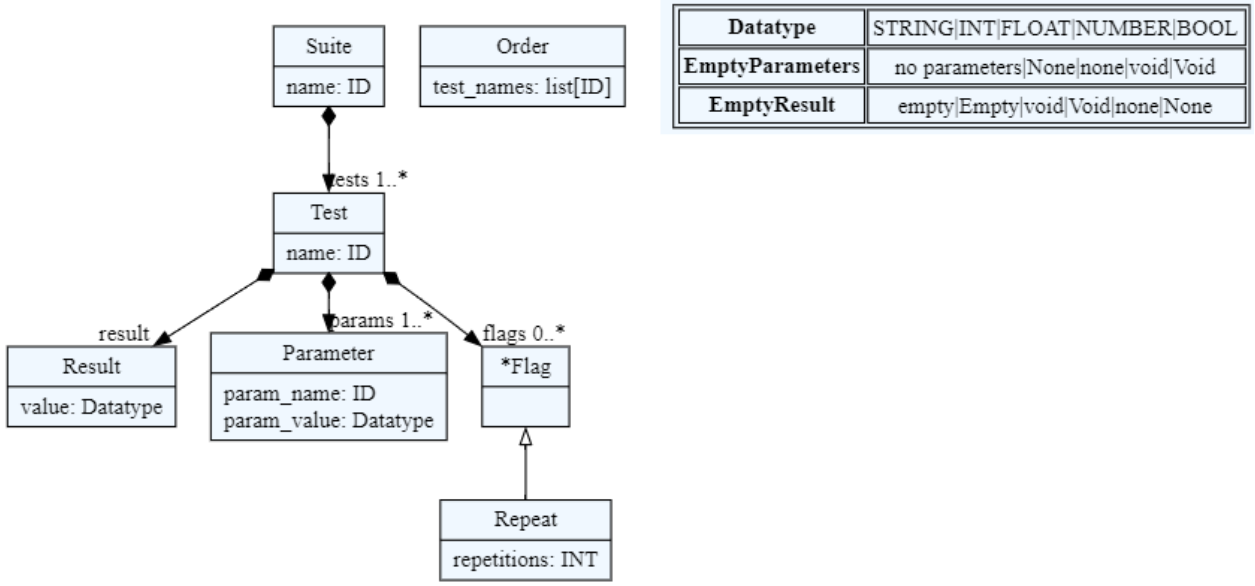


Figure 2: Language meta-model

An example parse tree would be useful.

Suite mySuite

Test foo -skip -repeat 5 times
When param1=1, param2=True
Then result should be 42

The obtained parse tree from the above piece of code is constructed in Figure 3.

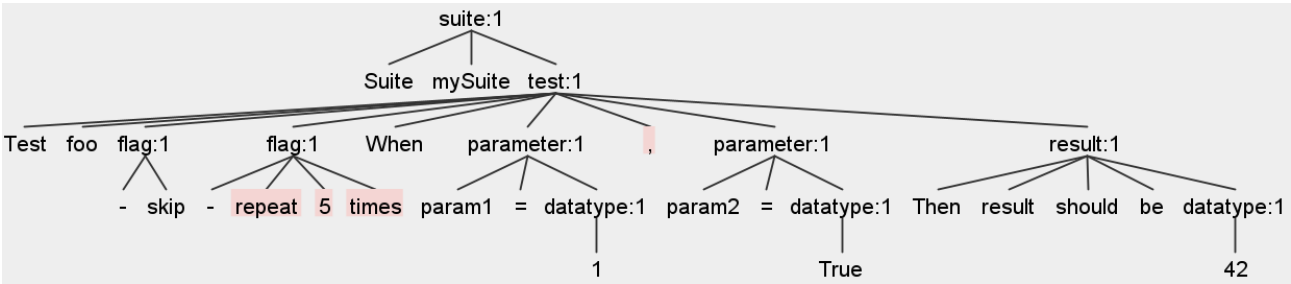


Figure 3: Parse Tree for example program

Conclusions

All things considered, a domain specific language for unit testing may simplify the development, management, and execution of tests while also improving the quality, consistency, and automation of unit tests. This allows for the rapid and effective identification and resolution of problems. It is a great tool that could be used by developers and testers to make their work easier and save their time.

Users create and change data by writing code that defines test cases and related structures. The DSL allows users to express their test cases in a succinct and legible manner, as well as a framework for executing and reporting on the results of those tests.

References:

1. Unit Testing, TechTarget [online]. [accessed 20.02.2023] Available: https://www.techtarget.com/searchsoftwarequality/definition/unit-testing

2. Domain overview, GURU99 [online]. [accessed 22.02.2023] Available: https://www.guru99.com/unit-testing-guide.html
3. Who performs unit testing?, METHODPOET [online]. [accessed 22.02.2023] Available: https://methodpoet.com/who-performs-unit-testing/
4. Unit Testing Problems, Mozaic Works [online]. [accessed 22.02.2023] Available: https://mozaicworks.com/blog/5-common-unit-testing
5. Security unit tests, DaDario [online]. [accessed 28.02.2023] Available: https://dadario.com.br/security-unit-tests-are-important/

DOMAIN SPECIFIC LANGUAGE FOR PETRI NETS

Maxim CERNETCHI[1], Corneliu NASTAS[1],
Alexandr CARA[1], Vladislav CRUCERESCU[1]

[1]*Department of Software and Automation Engineering, group FAF-212, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chisinau, Moldova*

*Corresponding author: Cernetchi Maxim, maxim.cernetchi@isa.utm.md

Scientific coordinator: Irina Cojuhari, conf. univ., dr., DISAa

Abstract. This scientific paper presents the development of a Domain Specific Language (DSL) for Petri nets, a formalism used for modeling concurrent systems. The DSL provides an intuitive syntax for modeling Petri nets, which enables domain experts to design and analyze complex systems without the need for formal training in the underlying theory. The paper outlines the syntax and semantics of the DSL and demonstrates its applicability through a set of case studies. The results show that the DSL improves the productivity and efficiency of the modeling process and makes the analysis of Petri nets more accessible to a wider audience. The paper concludes with a discussion of the benefits and limitations of the DSL and its potential for future research and development in the field of Petri nets.

Keywords*: Petri Nets, transitions, places, token, dsl.*

Introduction

Petri Nets are a powerful mathematical tool used to model complex systems and processes [1]. They provide a graphical representation of a system's state and its transitions, making them an ideal choice for modeling a wide range of scenarios. However, creating and analyzing Petri Nets can be a time-consuming and error-prone process, especially for large and complex systems. To overcome this challenge, Domain-Specific Languages (DSLs) can be used to provide a more efficient and convenient way to work with Petri Nets. A DSL is a programming language that is tailored to a specific domain, making it easier to express domain-specific concepts and operations. In this context, creating a DSL for Petri Nets can simplify the process of modeling, simulating, and analyzing Petri Nets, improving both the accuracy and the speed of the analysis. This can have significant benefits in areas such as manufacturing, logistics, and systems engineering, where the use of Petri Nets is common. In this article, we will explore the process of creating a DSL for Petri Nets, discussing the benefits of this approach and providing practical examples to illustrate its usage.

Domain analysis

Petri Nets are a mathematical modeling language used to represent and analyze the behavior of discrete systems. Petri Nets are graphical representations of the interactions between different components in a system and provide a visual representation of the underlying relationships between states and transitions.

Key concepts and principles:

Petri Nets are a graphical modeling language used to represent and analyze the behavior of discrete systems. The key concepts and principles of Petri Nets include:

1. *Places*: places represent the components of a system, such as resources, queues, or intermediate stages of a process. They are represented by circles in a Petri net diagram.

2. *Transitions*: transitions represent events or actions that change the state of the system. They are represented by rectangles in a Petri net diagram.

3. *Tokens*: tokens are used to represent the presence of a resource or the completion of an event. Tokens are represented by dots in a Petri net diagram and are placed in places.

4. *Arcs*: arcs represent the relationships between places and transitions. They are represented by directed lines connecting places to transitions. Inbound arcs (place to transition) indicate the input required for a transition to fire, while outbound arcs (transition to place) indicate the output produced when a transition fires.

5. *Firing*: firing is the process by which a transition changes the state of the system by consuming tokens from input places and producing tokens in output places. A transition can only fire if there are enough tokens in its input places and enough free space in its output places.

6. *State*: the state of a Petri net is represented by the distribution of tokens in the places. A Petri net can be in multiple states at the same time, representing different scenarios or parallel processes.

7. *Reachability*: reachability is the property of a Petri net that allows for the analysis of the possible sequences of transitions that can lead to a particular state. This property is useful for verifying the behavior of the system and for analyzing the impact of changes to the system.

8. *Boundedness*: boundedness is the property of a Petri net that ensures that the number of tokens in a place that remains within a specified range. This property is important for ensuring the stability and reliability of the system.

These are the key concepts and principles of Petri Nets. Understanding these concepts is essential for effectively using Petri Nets to model and analyze discrete systems.

Tools and techniques:

There are several tools and techniques that are commonly used in the analysis and design of systems modeled with Petri Nets. Some of the most important tools and techniques include:

1. *Modeling and simulation*: Petri Nets can be used to create models of systems, which can then be simulated to visualize and analyze the behavior of the system.

2. *State space analysis*: Petri Nets can be analyzed to determine the number of reachable states, the number of transitions between states, and other properties of the system. This analysis is useful for verifying the behavior of the system and for analyzing the impact of changes to the system.

3. *Verification and validation*: Petri Nets can be used to verify and validate system models by checking the consistency and completeness of the model and by comparing the behavior of the model with the behavior of the real system.

4. *Performance analysis*: Petri Nets can be used to analyze the performance of systems by calculating various performance metrics, such as response time, throughput, and utilization.

5. *Control and optimization*: Petri Nets can be used to control and optimize systems by determining the optimal policies for controlling the transitions and by finding the optimal settings for the parameters of the system.

These are some of the most important tools and techniques for Petri Nets. The choice of tool or technique will depend on the specific requirements of the system being modeled and the goals of the analysis.

Applications and use cases:

Petri Nets have a wide range of applications and use cases in various domains, including but not limited to:

1. *Manufacturing and production*: Petri Nets can be used to model and analyze the behavior of production systems, such as assembly lines, and to optimize the production process.

2. *Computer networks*: Petri Nets can be used to model and analyze the behavior of computer networks, such as communication protocols and data flow, and to optimize the performance of the network.

3. *Software engineering*: Petri Nets can be used to model and analyze the behavior of software systems, such as workflow systems and user interfaces, and to optimize the design and implementation of the software.

4. *Business process modeling*: Petri Nets can be used to model and analyze the behavior of business processes, such as supply chain management and customer service, and to optimize the performance of the processes.

5. *Healthcare*: Petri Nets can be used to model and analyze the behavior of healthcare systems, such as patient flow and resource utilization, and to optimize the delivery of care.

6. *Transportation*: Petri Nets can be used to model and analyze the behavior of transportation systems, such as air traffic control and traffic flow, and to optimize the performance of the transportation system.

These are just a few examples of the wide range of applications and use cases for Petri Nets. The versatility of Petri Nets makes them suitable for modeling and analyzing a wide range of discrete systems in various domains.

Grammar and Language Design.

Grammar refers to the set of rules governing the structure of a language, including the way words are formed, combined, and used to convey meaning. It encompasses everything from the parts of speech and their functions to sentence structure and punctuation.

The language design started from identifying what the DSL language can and must do. We identified the basic rules for the petri Nets:

1. Petri Nets have transitions and places

The syntax for this would be simple instantiation:

```
place p;
place p1, p2, p3;
tran t;
tran t1, t2, t3;
```

Where pos, tran is the datatype of the variable. Notice that it uses semicolons to represent the end of the declaration.

2. The places' state describes the amount of token it has, it also has a token maximum capacity which can be finite or infinite.

This can be done with this simple representation:

```
place p;
p.amm = 3;
p.cap = 9;
```

The token amount of the place p is 3, whilst its maximum capacity is 9. The default quantity for the token amount is zero, whilst for the maximum capacity is infinity.

3. Transitions and places are connected through inbound and outbound arcs:

Transitions and places can connect with multiple places and transitions.

```
place p, p1, p2, p3;
tran t, t1, t2;
t.in = {p1, p2};
t.out = {p3};
p.in = {t1, t2};
p.out = {t};
```

The inbound arcs for the transition t, connect it to p1 and p2, whilst the outbound arc connects it to p3. p is connected to t1 and t2 with outbound arcs, which should not be confused with the syntax p.in. For simplicity, it means whether the arrow in the graphical representation is pointing towards the place or the transition. Notice that "p.out = {t}" connects itself to t which already has arcs. Rather than overriding t's arcs, it would add to the inbound arcs t already has, making t.in = {p, p1, p2}.

4. Arcs have weights:

Inbound arcs' weight attribute describes the amount of tokens it will consume during firing. Outbound arcs' attribute describes the amount of tokens it will produce during firing.

place p1, p2, p3, p4;
tran t1, t2;
t1.in = {p1 : 2, p2};
t2.out = {p3, p4};
p3.in = {t1 : 3};
p3.out = {t2};

The inbound arc "p1->t1" has the weight of 2. For the variables where there is no ": <number>" notation, the weight is set to the default value 1.

With this we designed the following grammar for the DSL:

```
V_N = {<program>, <declarationlist>, <declaration>, <instantiation>, <placefield>,
<connection>, <type>, <varlist>, <var>, <number>, <arcing>, <arclist>, <arc>,
<nonzero>, <digits>, <digit>, <alpha>, <string>, <char>}
V_T =
{a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,
Q,R,S,T,U,V,W,X,Y,Z,_,0,1,2,3,4,5,6,7,8,9,.,,,:,{,},=,;}
P = {
<program> -> <declarationlist>,
<declarationlist> -> <declaration> ; <declarationlist> | ε,
<declaration> -> <instantiation> | <placefield> | <arcing>,

<instantiation> -> <type> <varlist>,
<type> -> place | tran,
<placefield> -> <var>.amm = <number> | <var>.cap = <number>,
<arcing> -> <var>.in = { <arclist> | <var>.out = { <arclist>,
<arclist> -> <arc> , <arclist> | <arc> },
<arc> -> <var> : <number> | <var>,

<number> -> <nonzero><digits>,
<digits> -> <digit><digits> | ε,
<digit> -> 0|<nonzero>,
<nonzero> -> 1|2|3|4|5|6|7|8|9,

<varlist> -> <var> , <varlist> | <var>,
<var> -> <alpha><string>,
<string> -> <char><string> | ε,
<char> -> _|<digit>|<alpha>,
<alpha> -> a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z|A|B|C|D|E|F|G|H|I|
J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z
}
```

Figure 1. Grammar Definition of DSL syntax

Example:

place p1, p2, p3;
tran r1, r2;
p1.amm = 4;
p2.amm = 3;
p3.cap = 5;
r.in = {p1 : 2, O2};
p3.in = {r1 : 2};
p3.out = {r2};

Would generate something like this:

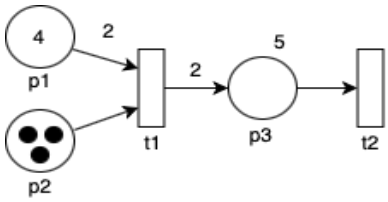


Figure 2. Petri Net representation

Conclusion.

Petri Nets are a valuable tool for modeling complex systems and processes, providing a graphical representation of a system's state and its transitions. However, working with Petri Nets can be a time-consuming and error-prone process, especially for large and complex systems. Creating a Domain-Specific Language for Petri Nets can simplify the process of modeling, simulating, and analyzing Petri Nets, improving both the accuracy and the speed of the analysis. By tailoring a programming language to the specific needs of Petri Nets, a DSL can make it easier to express domain-specific concepts and operations, allowing users to focus on the problem at hand instead of the technical details of the modeling process. The benefits of using a DSL for Petri Nets are significant, and can have a positive impact on fields such as manufacturing, logistics, and systems engineering. As such, studying Petri Nets and exploring the creation of DSLs for them is an important area of research, with the potential to improve the efficiency and effectiveness of many real-world applications.

References:

1. Păstrăvanu O., Matcovschi M., Mahulea C. *Aplicații ale rețelelor Petri în studierea sistemelor cu evenimente discrete*. Iași: Editura Gh. ASACHI , 2002.

GENERAL PURPOSE LANGUAGE

Maria PROCOPII[1*], Alexei CIUMAC[1], Dorin OTGON[1], Andreea MANOLE[1], Alexandru DOBROJAN[1]

[1]*Department of Software Engineering and Automatics, FAF-212, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chișinău, Republic of Moldova*

*Corresponding author: Maria Procopii, maria.procopii@isa.utm.md

Scientific coordinator: Irina COJUHARI, conf. univ., dr., DISA

Abstract. *General-purpose programming languages are those that are meant to handle a broad variety of activities, such as scientific computing, data analysis, web development, and software engineering. The objective of this paper is to offer an overview of general-purpose languages, covering their characteristics, usage, and features. The article examines the advantages and disadvantages of utilizing general-purpose languages, as well as some of the most popular and commonly used general-purpose languages.*

Keywords*: General-purpose language, programming languages, software engineering, design, grammar*

Introduction

A general-purpose programming language is one that is designed to be flexible and adaptable enough to be used for a wide range of programming tasks. These languages are not restricted to a specific application area or field, but may be used to construct a broad variety of software applications, such as desktop and mobile apps, internet applications, and games [1].

General-purpose programming languages provide developers with a wide range of tools and capabilities, making it easier to construct complex programs. They frequently have a high level of abstraction, allowing developers to express complex notions in a plain and succinct manner without having to worry about the underlying complexity of how the machine actually executes the program.

One of the key benefits of using a general-purpose language is its versatility. Instead than learning and utilizing a distinct language for each activity, developers may use the same language to construct a variety of different apps. This facilitates jumping between projects and interacting with other developers who may be working on a variety of apps. A general-purpose program written effectively may take less time to execute than an equivalent program written in a simulation language. This is because a simulation language is designed to represent a wide range of systems using a single set of building blocks, whereas a general purpose program may be tailored to a specific application. While general purpose languages are present on almost every computer, a specific simulation language may be unavailable on the system that the analyst intends to use [2].

Language overview

A very user-friendly general-purpose language, with a small set of syntax structures that can easily build-up complex programs that can solve bigger problems, is the short description of our language. In order to succeed, this general-purpose language has the needed functionality for variables binding, if-else conditionals, functions and cycles.

The language is partially static typed, i.e. in the functions we are obliged to give a type for the parameters of the function, whereas when declaring a variable, for instance, we don't have to precisely specify a type for the variable. Although, the language support all the classical logical operators met in other programming languages, we tend to give a more friendly and easily understandable control flow experience to the user, thus we've add some rarely seen conditional syntax structure, by the use of the *unless* and *then* keywords.

Grammar definition

A computer language's syntax is an essential component. A grammar is a collection of rules that govern how a language's syntax works. It specifies the sequence of symbols and constructs that may be used to create expressions and statements in the language. The grammar of a programming language determines how the language is processed and code is executed.

Grammar in programming languages is related to the language's syntax. The syntax specifies how statements and expressions are built and combined to form more complex expressions. The grammar of a computer language is often specified using formal notation such as Backus-Naur Form (BNF) or Extended Backus-Naur Form (EBNF) (EBNF). These notations allow for an accurate representation of the syntax of the language, as well as the allowable sequence of symbols and constructs that compose the language's expressions and statements [3].

The syntax of a computer language determines its utility and adaptability. A sophisticated or restrictive syntax may be more difficult to learn and use than a simple and flexible grammar. Grammar also has an impact on linguistic expressiveness. A more expressive grammar may permit more concise and readable code, whereas a less expressive grammar may necessitate more lengthy and difficult-to-read code. The grammar of the proposed language is described as $G = (VN, VT, S, P)$ where VN is thefinite set of non-terminal symbols, VT is the finite set of terminal symbols, S is the start symbol, and$P$ is the finite set of production rules.

VN = { <program>, <statement>, <expression>, <return expression>, <literal>, <id>, <comment>,
<text_char>, <conditional>, <for expression>, <array>, <element>, <function call>,
<args>, <function>, <param>, <bool>, <type>, <letter>, <digit> };
VT = { return, true, false, [], [a-z], [A-Z], [0-9], “, _, !, @ , # , $, %, && , || , > , < , >= , <= , == | !=, *,
/, +, -, ++, --, = };
S = { <program>};
P = {<program> -> <statement> | <statement><program>
<statement> -> <expression> |<expression><statement>
<expression> -> <id> | <literal> | <conditional>| <id><assign><literal>
<id><assign><expression> <for expression> |<array> | <bool><function> |<function call> |
<expression><logical operator><expression> |
<expression><arithmetic operator><expression> |<sufix><id> | <id><suffix> | <comment>
|<returnexpression>
<return expression> -> return <expression> | return
<literal> -> <string> | <int> | <bool>
<id> -> <text_char><id> | <text_char>
<comment> -> //<string>
<text_char> -> <letter><digit><symbol><text_char> | <letter><digit><symbol>
<string> -> "<text_char>"
<int> -> <digit><int> | <digit>
<conditional> -> if (<expression>) {<program>} |unless (<expression>) {<program>} |
if (<expression>) {<program>} else {<program>} |unless (<expression>) {<program>} then
{<program>}<for expression> -> for (<expression>;<expression;<expression>) {<program>} |for
(;<expression>;<expression>) {<program>}
<array> -> [<element>] | []
<element> -> <id>,<element> |<literal>,<element> |<id> | <literal>
<function call> -> <id>(<args>) | <id>()
<args> -> <expression> | <expression>, <args>

<function> -> def <id>(<param>) {<program>} |def <id>() {<program>}
<param> -> <type><id> | <type><id>, <param>
<bool> -> true | false
<type> -> <int> | <string> | <bool>
<letter> -> a | b | ... | z | A | B | ... | Z | _
<digit> -> 0 | 1 | ... | 9 | ε
<symbol> -> ! | @ | # | $ | % | ... | ε
<logical operator> -> && | || | > | < | >= | <= | == | !=
<arithmetic operator> -> * | / | + | -
<sufix> -> ++ | --
<assign> -> =
}

Parsing tree

A parse tree is a hierarchical representation of a string of symbols or code's syntactic structure, generally in the form of a tree or a directed acyclic graph. It is generated by a parser during the parsing process, which is the act of examining a program's syntax and establishing its grammatical structure. Fig. 1 depicts the parsing tree based on the grammar developed in the previous section.

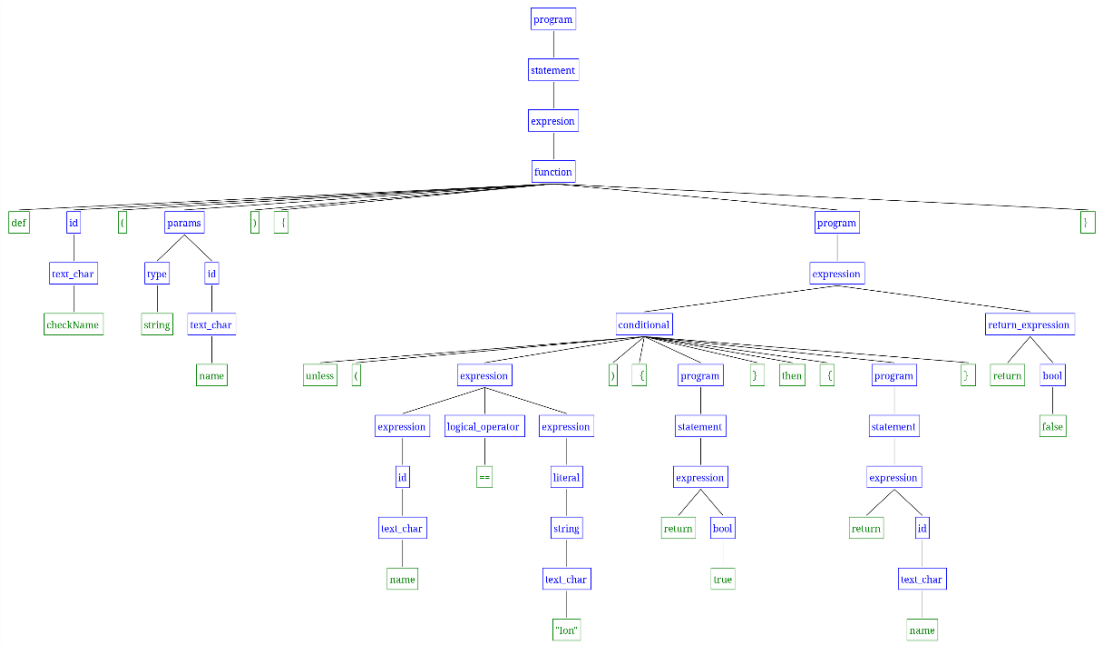


Figure 1. Parsing tree example

Syntax example

The code sample below is an example of syntax.

def checkName(**string** *name*) {
unless (*name* == "Ion") {
    return true
} **then** {
    return *name*
}
return false
}

Conclusions

General-purpose programming languages are crucial tools for software development, scientific computing, data analysis, and web development. They are versatile and adaptable, allowing programmers to utilize them in a variety of scenarios. The programming language used is determined by the application domain as well as the programmer's expertise and talents. While general-purpose languages have many advantages, they do have certain disadvantages, such as slower performance or a lack of support for specialized features or libraries. Overall, general- purpose languages are crucial components of modern software development, and their relevance will only rise in the future.

A programming language's grammar is a fundamental component that specifies how the language's syntax operates. It influences the language's usability, flexibility, and expressiveness, making it an important factor to consider when selecting a programming language. A solid knowledge of a programming language's syntax may help programmers produce more succinct,legible, and maintainable code.

References

1. MARGARET ROUSE, TECH TARGET, *What is a General-Purpose Programming Language?* [online]. [accessed 26.02.2023]. Available: https://searchapparchitecture.techtarget.com/definition/general-purpose-programming-language
2. KING FAHD, *Advantages of General Purpose Languages* [online].[accessed 26.02.2023]. Available: https://faculty.kfupm.edu.sa/SE/habboubi/Courses/SE405_STOCHASTIC_SYSTEM_SIMULATION/Advantages_of_Simulation_languages.doc#:~:text=General%E2%80%94purpose%20languages%20allow%20greater,are%20not%20easy%20in%20GP SS.
3. ALFRED V. *AHO, MONICA S. L., Compiles Principles, Techniques, and Tools* [online]. [accessed 28.02.2023]. Available: https://repository.unikom.ac.id/48769/1/Compilers%20-%20Principles%2C%20Techniques%2C%20and%20Tools%20%282006%29.pdf

JADE - A GENERAL PURPOSE LANGUAGE

Dinu GUȚU[1*], Adelia BRĂGUȚA[1], Andrei SĂRĂTEANU[1], Alexandra KONJEVIC[1], Milena COJUHARI[1]

[1]*Department of Software Engineering and Automatics, Group FAF-213, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chișinău, Republic of Moldova*

*Corresponding author: Dinu Gutu, dinu.gutu@isa.utm.md

Scientific coordinator: Alexandru VDOVICENCO, University Lecturer

Abstract. *General-purpose programming languages are crucial tools for the development of computer programs and are created to offer a broad range of capabilities and flexibility. The idea of general-purpose languages, their traits, their significance in contemporary software development, and how one can be developed are all covered in this article.*

Key words*: General purpose language, grammar, interpreter, lexer, parser.*

Introduction

The ability to express complex algorithms and create software systems is made possible by programming languages, which are crucial to software engineering. Despite the fact that there are numerous programming languages on the market right now, developing a brand-new general-purpose language from scratch takes considerable time and knowledge. Nevertheless, learning a new language can offer a lot of advantages, such as greater performance, readability, and expressiveness [1].

In recent years, there has been growing interest in creating new general-purpose programming languages that are better suited to the needs of modern software engineering. These new languages are designed to provide more efficient, scalable, and maintainable solutions to the challenges posed by complex software systems. Researchers and practitioners have been exploring new approaches to programming language design, seeking to improve upon existing languages and create new ones that meet the evolving needs of software development [2].

Creating a general-purpose programming language from scratch is examined in this work, with a focus on the creation of a lexer and the design of the language's grammar. In detail are laid out the theoretical foundations of programming language design, the principles of lexical analysis, parsing, and semantic analysis. The tradeoffs between expressiveness, efficiency, and readability are only a few examples of the practical factors that must be taken into account when creating a language.

A case study of general-purpose programming that makes use of the aforementioned methodologies was offered in this article to illustrate these findings. In addition, the grammar and lexer are discussed in detail, along with any difficulties that can arise during the development process. The analysis shows that it is possible to develop a general-purpose language that is both descriptive and effective with careful planning, attention to detail, and a thorough understanding of the principles of programming and language design.

Language overview

A general-purpose programming language (GPL) is a type of programming language that can be used to create software for a wide variety of application domains. Although their use within a specific domain is somewhat inferior to that of domain specific languages, the existence of libraries that are extensively tested and optimized should in theory bridge the gap between them [3].

Jade is a general-purpose language that is designed to be simple, versatile, and efficient. The language is based on a minimalist syntax that relies heavily on indentation and white space. This makes it easy to write and read code, and reduces the amount of typing required. Jade also supports

templates and mixins, which can greatly simplify the development process by reducing the amount of repetitive code that needs to be written.

The languages' abstract syntax tree interpreter will run the program written in a file with .jd extension, or in the command line. The interpretation of the source code will follow the reconstruction of the code into an abstract syntax tree and each sentence will be translated and executed at a time [4]. Its lexer and parser have the role of managing the grammar. This will perform in the following way: the lexer will analyze the input sequence and convert it into a string of tokens, then, the parser will take over the generated tokens and use them to create the abstract syntax tree.

Tokens are the representation of several terminal symbols existing in the grammar. They are composed of the: identifiers, literals, operators and punctuations. These symbols are the main component that makes possible the creation of rules of the set of non-terminal symbols.

Grammar

In formal language theory, a grammar (sometimes referred to as a formal grammar when the context is not specified) describes how to construct strings from an alphabet that are valid in accordance with the syntax of the language. The meaning of the strings or what can be done with them in any context are not described by a grammar; only their form is. A formal grammar is a collection of rules for creating these strings in a formal language. In addition to that, the grammar of a language can be represented using formalized notations like Backus-Naur Form (BNF) or Extended Backus-Naur Form (EBNF), which provide a concise method to define a language's syntax. Because grammar establishes a computer language's structure, compilers and interpreters use it to parse and evaluate source code [5].

Table 1.

Meta notations

Notations	Meaning
<text>	a nonterminal symbol
"text"	a terminal symbol
text*	a symbol that appears zero or more times
|	an alternative follows

$G = (V_N, V_T, P, S)$
V_N - finite set of non-terminal symbols.
V_T - finite set of terminal symbols.
P - finite production rules.
S - start symbol.

V_T = { "identifiers", "literals", "operators", "punctuation" }

- identifiers: a set of alphanumeric strings starting with a letter or underscore
- literals: a set of numeric, string, and boolean values
- operators: a set of symbols representing arithmetic, logical, and bitwise operations
- punctuation: a set of symbols used for syntax, such as parentheses, braces, and semicolons

V_N = {<Program>, <Statement>, <Expression>, <AssignmentExpression>, <ConditionalExpression>, <LogicalExpression>, <BinaryExpression>, <UnaryExpression>, <CallExpression>, <MemberExpression>, <PrimaryExpression>, <FunctionExpression>, <ObjectExpression>, <ArrayExpression>, <VariableDeclaration>, <IfStatement>, <WhileStatement>, <ForStatement>, <FunctionDeclaration>, <ReturnStatement>,

<BlockStatement>, <Identifier>, <Literal>, <NumericLiteral>, <StringLiteral>, <BooleanLiteral>, <Arguments>, <Property> }

S = {<Program>}

P = {

```
        <Program> → <Statement>*
        <Statement> → <Expression>
                      | <VariableDeclaration>
                      | <IfStatement>.
                      | <WhileStatement>
                      | <ForStatement>
                      | <FunctionDeclaration>
                      | <ReturnStatement>
        <Expression> → <AssignmentExpression>
                      | <ArrayExpression>
                      | <ConditionalExpression>
                      | <LogicalExpression>
                      | <BinaryExpression>
                      | <UnaryExpression>
                      | <CallExpression>
                      | <MemberExpression>
                      | <FunctionExpression>
                      | <PrimaryExpression>
                      | <ObjectExpression>
                      | <Literal>
        <AssignmentExpression> → <Expression> "=" <Expression>
        <ConditionalExpression> → <LogicalExpression> "?" <Expression> ":" <Expression>
        <LogicalExpression> → <LogicalExpression> "&&" <LogicalExpression>
                            | <LogicalExpression> "||" <LogicalExpression>
        <BinaryExpression> → <Expression> "+" <Expression>
                            | <Expression> "-" <Expression>
                            | <Expression "" Expression>
                            | <Expression "/" Expression>
                            | <Expression "%" Expression>
                            | <Expression "==" Expression>
                            | <Expression "!=" Expression>
                            | <Expression "<" Expression>
                            | <Expression ">" Expression>
                            | <Expression "<=" Expression>
                            | <Expression ">=" Expression>
                            | <Expression "<<" Expression>
                            | <Expression ">>" Expression>
                            | <Expression "&" Expression>
                            | <Expression "|" Expression>
                            | <Expression "^" Expression>

        <UnaryExpression> → "!" <Expression>
                            | "-" <Expression>
                            | "+" <Expression>
                            | "~" <Expression>
        <CallExpression> → <MemberExpression> <Arguments>
        <Arguments> → "(" [<Expression> ("," <Expression>)] ")"
```

```
<MemberExpression> → <PrimaryExpression> ("." <Identifier> | "[" <Expression> "]")*
<PrimaryExpression> → <Identifier>
                      | <Literal>
                      | "(" <Expression> ")"
                      | <FunctionExpression>
                      | <ObjectExpression>
                      | <ArrayExpression>
<FunctionExpression> → "function" [<Identifier>] "(" [<Identifier> ("," <Identifier>)] ")"
<BlockStatement>
<ObjectExpression> → "{" [<Property> ("," <Property>)] "}" <Property>→ <Identifier> ":"
<Expression>
<ArrayExpression> → "[" [<Expression> ("," <Expression>)] "]"
<VariableDeclaration> → "var" <Identifier> ["=" <Expression>] ";"
<IfStatement> → "if" "(" <Expression> ")" <BlockStatement> ["else" <BlockStatement>]
<WhileStatement> → "while" "(" <Expression> ")" <BlockStatement>
<ForStatement> → "for" "(" [<VariableDeclaration>
                      | <Expression>] ";" <Expression> ";" <Expression> ")"
                <BlockStatement>
<FunctionDeclaration> → "function" <Identifier> "(" [<Identifier> ("," <Identifier>)] ")"
<BlockStatement>
<ReturnStatement> → "return" [<Expression>] ";"
<BlockStatement> → "{" <Statement>* "}"
<Identifier> → /[a-zA-Z_][a-zA-Z0-9_]/
<Literal> → <NumericLiteral>
            | <StringLiteral>
            | <BooleanLiteral>
<NumericLiteral> → /[0-9]+(.[0-9]+)?/
<StringLiteral> → /"([^"\]|\.)"/
<BooleanLiteral> → "true"
                      | "false"
}
```

Sample code

The array arr is defined in the following program using the values [1, 5, 9, 11, 0]. Then, it creates the function doubleArray, which multiplies each value in an array by two before returning the transformed array. Next, after supplying the arr array as input, it calls the doubleArray method. All of the values in the initial array are effectively doubled when the resultant array provided by doubleArray is set to the variable arr. As a result, the array will contain the values [2, 10, 18, 22, 0] at program's completion.

```
var arr = [1, 5, 9, 11, 0];

function doubleArray(arr) {
  var doubledArr = [];
  for (var i = 0; i < arr.length; i++) {
    doubledArr[i] = arr[i] * 2;
  }
  return doubledArr;
}
var doubleArr = doubleArray(arr);
```

Parsing tree

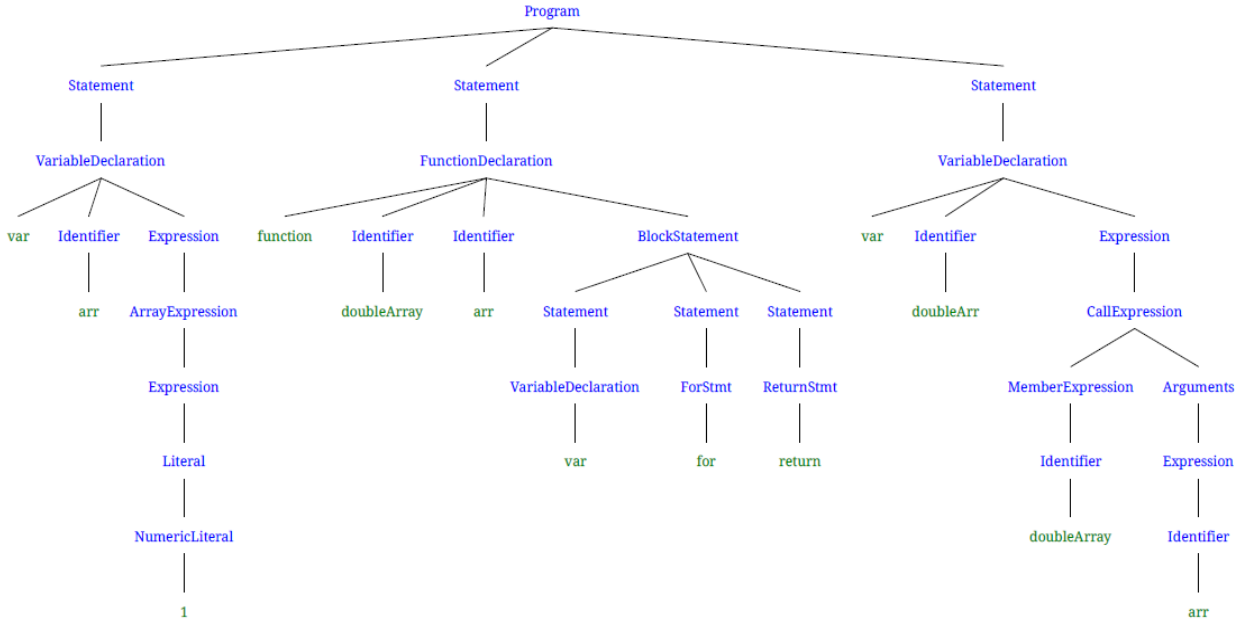


Figure 1. Parsing tree

The parsing tree represents the abstract syntax tree of the program that doubles an input array. It has been significantly reduced due to the big number of branchings that take place. The beginning is the <Program> which disperses into as many statements as needed, in this case, 3. Then each <Statement> will into a <VariableDeclaration> and a <FunctionDeclaration>. The former has a <Identifier> that represents the name of the variable and a <Expression> that can begin to represent what the variable will be. The latter, that is the <FunctionDeclaration>, also separates into identifiers to represent the name of the function and input, and also it branches off into a <BlockStatement> which covers the logic of the function.

Conclusion

Jade is a general-purpose language that is designed to be simple, versatile, and efficient. Numerous things, including software, research projects, mathematical calculations, and so forth, can be done using it. Its simple grammar, which places a heavy emphasis on indentation and white space, enables rapid and simple coding. Additionally, the use of templates and mixins greatly simplifies the development process, reducing the amount of repetitive code that developers need to write.

General-purpose languages, such as Jade, are designed to be flexible and all-round, with the goal of allowing developers to write code that can be used in a wide range of applications.
These languages can be used to create any kind of program, including windows software, web applications, and analytical models. They are not constrained to certain industries or program kinds. The main benefit of general-purpose languages is that they enable programmers to create code that can be applied in a number of situations, allowing them to reuse code and reduce effort redundancy. Additionally, they frequently have sizable, vibrant communities that offer a wealth of tools to aid developers in their learning and skill-building.

In conclusion, general-purpose languages are an important resource for developers who need to work across different domains or who want to write code that is more broadly applicable, even though specialized languages might be more effective in some situations. The final decision about the language will be made in light of the project's particular requirements as well as the preferences of the development team.

References:

1. AABY, A. *Introduction to Programming Languages* [online]. [visited 09.03.2023]. Available: https://web.archive.org/web/20121108043216/http://www.emu.edu.tr/aelci/Courses/D-318/D-318-Files/plbook/intro.htm.
2. Statistics and Data. [online]. [visited 09.03.2023]. Available: https://statisticsanddata.org/data/the-most-popular-programming-languages-1965-2021.
3. KOSAR, T. *Comparing General-Purpose and Domain-Specific Languages: An Empirical Study*. In: *Computer Science and Information Systems*. 2010, pp. 247–264
4. VAIDEHI J. *A Deeper Inspection Into Compilation And Interpretation* [online]. [visited 02.03.2023]. Available: https://medium.com/basecs/a-deeper-inspection-into-compilation-and-interpretation-d98952ebc842.
5. ATTENBOROUGH M. Language theory. In: Attenborough M., ed *Mathematics for Electrical Engineering and Computing*. 2003, pp. 479-490.

MARKUP LANGUAGE FOR PSYCHOLOGICAL ARCHETYPES

Alexandru BUZU[1*], Bogdan ZLATOVCEN[1], Dumitru MORARU[1], Maria LEȘENCO[1], Marius BOTEZATU[1]

[1]*Faculty of Computers, Informatics and Microelectronics, Software Engineering, Group FAF-212, Technical University of Moldova, Chișinău, The Republic of Moldova*

*Corresponding author: Alexandru Buzu, alexandru.buzu@isa.utm.md

Coordinator: Vasili BRAGA, conf. univ, TUM

Abstract. *The main topic of this scientific article is the basics for development of a markup language based on NLP. The proposed solution is oriented to help content creators, and especially game developers, to create original behavior and background history for NPCs and other characters. The markup language aims to have a very user friendly syntax, and operate with ChatGPT API to give original and AI based responses.*

Keywords: *formal language, grammar, AI, game development, NPC.*

Introduction

An AI-based markup language for NPC stories can revolutionize the way game developers create immersive and engaging game experiences. By automating the process of NPC story generation, an AI-based markup language can significantly reduce the amount of time and resources required to create unique and high-quality stories. This can be especially advantageous for indie game developers and smaller studios with limited resources, enabling them to focus on other critical aspects of game development.

Moreover, AI-powered language models can generate dynamic NPC stories that respond dynamically to player actions, resulting in more engaging and immersive game experiences. The variability of the generated stories can lead to increased replayability for players, as each playthrough may have different NPC storylines. This can significantly improve the quality and variability of the game, making it more enjoyable for players.

Additionally, an AI-based markup language for NPC stories can provide game developers with a platform for experimentation and innovation in storytelling. With the ability to generate dynamic NPC stories based on player input, developers can explore new narrative structures and gameplay mechanics, leading to the creation of more innovative and engaging games. This can open up new opportunities for developers to experiment with different narrative styles and explore new and creative storytelling techniques.

Although there are several existing tools and technologies that game developers use to create NPC stories, including markup languages like Ink and Yarn, an AI-based markup language that generates NPC stories automatically is a relatively new concept. The use of AI and machine learning in game development is becoming increasingly common, as developers seek to create more immersive and engaging game experiences for players.

An example of an existing solution is AI Dungeon - a text-based adventure game that uses natural language processing and machine learning to create dynamic storylines based on user input. It enables users to create custom characters and storylines, which are generated using an AI-powered language model [1]. Artie, on the other hand, is an AI-based storytelling platform that enables users to create interactive stories with animated characters. Artie's AI engine generates stories based on user input, creating custom narratives for each character based on their traits and personalities.

Overall, an AI-based markup language for NPC stories should provide game developers with a powerful tool to create immersive and engaging game experiences for players. It should enable the

creation of dynamic NPCs that respond to player choices and actions, while also being accessible to developers with varying levels of experience in programming and game development. The AI model used by the markup language should be well-trained and maintained, with regular updates and improvements to ensure high-quality output. By leveraging the power of AI and machine learning, game developers can create truly immersive and engaging game experiences that captivate and entertain players for hours on end.

Language overview

- Basic Computation: The basic computation that this DSL performs is data manipulation and analysis related to psychological archetypes.
- Data Structures: The basic data structures in this DSL are archetypes and their attributes, which can be created and manipulated by the user.
- Control Structures: The user can specify control flow using different key parameters and their attributes.
- Input and Output: A program in this DSL requires input in the form of archetypal data and produces output in the form of analysis, manipulation of that data and a virtual chat with the described person.
- Error Handling: Programs may go wrong due to invalid input or incorrect use of syntax. The language will communicate these errors to the user through informative error messages.
- Other DSLs for this Domain: There are currently no widely-used DSLs specifically for the domain of psychological archetypes. However, there are general-purpose programming languages that can be used to manipulate archetypal data, such as Python or R. This language will be more user-friendly and focused on the needs of archetypal data analysis.

Grammar

For a markup language for psychological archetypes, we can define its grammar as follows:
L(G) = (S, P, VN, VT), where:

- S is the start symbol
- P is a finite set of production rules
- VN is a finite set of non-terminal symbols
- VT is a finite set of terminal symbols

The terminal symbols represent the actual content of the document or data being marked up. The non-terminal symbols represent the structure or syntax of the markup language itself.
The production rules describe how non-terminal symbols can be replaced or expanded into other non-terminal or terminal symbols [2]. These rules define the syntax and structure of the markup language.
To specify the grammar representation for the markup language we need to use meta notations:

Table 1

Meta notation

Symbol	Definition
< >	Non-Terminal symbol
*	Zero or more occurrences
+	One or more occurrences
|	Separates the alternative symbols
→	Derivation
~	Except: any character except

The grammar representation for markup language project:

<markup-language> → <description> <setting> <response>
<description> → **Description** "{" <description-property>* "}"
<description-property> → <name-property> | <type-property> | <mbti-property> |<role-property> | <background-property>
<name-property> → **name** = <string-literal>
<type-property> → **type** = <string-literal>
<mbti-property> → **mbti =** <string-literal>
<role-property> → **role =** <string-literal>
<background-property> → "background" "=" <string-literal>
<setting> → **Setting** "{" <setting-property>* "}"
<setting-property> → <type-property> | <category-property> | <background-property>
<category-property> → **category =** "[" <string-literal> ("," <string-literal>) * "]"
<response> → **Response** "{" <response-property>* "}"
<response-property> → <length-property> | <prompt-property>
<length-property> → **length =** <number-literal>
<prompt-property> → **prompt =** <string-literal>
<number-literal> → <digit>+
<digit> → 0 | 1 | 2 | ... | 9
<string-literal> → <char>*
<char> → <digit> | a | b | c | … | z | A | B | C | … | Z | - | _ | , | ; | “ | ” | : | . | ‘ | ? | !

An example of code using this grammar representation is:

```
Description {
	name=John
	type=NPC
	mbti=intj
	role=protagonist
	background="John is a witch hunter"
}
Setting {
	type=game
	category=[adventure, magic, open-world]
	background="A world similar to Mars, where witches are living in forests and fight with humans"
}
Response {
	length=300
	prompt="I need the background story of the character. Add some info about his family. Give 3 possible stories in which John is a side character"
}
```

Conclusions

In a world where automation takes a vital role in many fields, it is important for content creators to have various tools which will simplify some routine processes and boost their creativity and performance. The niche of DSL and Markup languages for defining NPC behavior and other topics related to creating the world, is not occupied at all, and the content creators are in need of solutions. That is why, the development of a markup language which will define the NPCs and other characters, is a very practical solution, and not so hard to realize with the help of tools like ChatGPT API.

References

1. AI Dungeon. [online]. [accessed 10.03.2023]. Available: https://aidungeon.io
2. TOMASSETTI, G. *A Guide to Parsing: Algorithms and Terminology* [online]. [accessed 10.11.2023]. Available: https://tomassetti.me/guide-parsing-algorithms-terminology/

ANALIZA METODELOR DE ACORDARE A REGULATORULUI LA MODELUL OBIECTULUI CU ANTICIPAȚIE-ÎNTÂRZIERE DE ORDINUL DOI CU ASTATISM

Dumitru MORARU

Departamentul Ingineria Software și Automatică, Facultatea Calculatoare Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Dumitru MORARU, e-mail dumitru.moraru@ati.utm.md

Rezumat. *În lucrare se efectuează analiza comparativă a procedurilor de acordare a algoritmilor la modelul obiectului de reglare cu anticipație-întârziere de ordinul doi cu oscilații și astatism de gradul unu cu parametrii cunoscuți după metodele gradului maximal de stabilitate cu iterații, polinomială și poli-zerouri. Pentru acordarea algoritmilor de tipul PID la modelul obiectului cu anticipație-întârziere s-a elaborat procedura utilizând metoda gradului maximal de stabilitate cu iterații și se obțin expresiile analitice ale parametrilor de acord ai algoritmilor de tipul PID ca funcții neliniare de argumentul - gradul de stabilitate și liniare pe parametrii cunoscuți ai modelului obiectului de reglare. Se variază gradul de stabilitate și aceste funcții se construiesc în formă grafică și, prin proceduri de iterații pe aceste curbe pentru aceiași valoare a argumentului, se aleg seturi de valori ale parametrilor algoritmilor de tipul PID și prin simulare se verifică performanțele sistemului automat. Pentru verificarea și compararea rezultatelor obținute s-a studiat un exemplu după metoda propusă, metoda polinomială și poli-zerouri cu simularea sistemului pe calculator. S-au variat parametrii modelului obiectului de la valorile nominale și s-a verificat robustețea sistemului. Se evidențiază avantajele metodei gradului maximal de stabilitate cu iterații cu calcule reduse și timp minim, care conduc la simplificarea procedurii de acordare a parametrilor algoritmului de tipul PID pentru aceste modele de obiecte.*

Cuvinte cheie: *modelul obiectului cu anticipație-întârziere și astatism, funcție de transfer, algoritm de reglare, parametrii de acord, metode de acordare, răspunsul sistemului, performanțele sistemului.*

Introducere

În lucrare se utilizează modelul matematic al obiectului de reglare cu anticipație-întârziere de ordinul doi cu oscilații și astatism descris cu funcția de transfer de forma [1-4]:

$$H(s) = \frac{k(T_1 s+1)}{s(T_2^2 s^2+2T_2 q s+1)} = \frac{b_0 s+b_1}{a_0 s^3+a_1 s^2+a_2 s}, \quad (1)$$

unde k este coeficientul de transfer, T_1, T_2 – constante de timp, q - gradul de amortizare ale obiectului tehnic-automobil, iar $b_0 = kT_1$, $b_1 = k$, $a_0 = T_2^2$, $a_1 = 2qT_2$, $a_2 = 1$.

Modelul obiectului de reglare (1) conține un zerou, un pol în origine și doi poli alocați în semiplanul stâng al planului complex al rădăcinilor.

Pentru modelul obiectului de reglare (1) este necesar de sintetizat algoritmul de conducere.

Există mai multe metode de acordare a algoritmului de tipul PID care se aplicată pentru acordarea regulatorului la modelul (1): metoda polinomială, alocării poli-zerouri etc. [3-7].

Metoda poli–zerouri este o metodă analitică de sinteză a algoritmului de reglare, dar care are dezavantaje la construirea polinomului caracteristic și soluționarea ecuațiilor matriceale [3, 6].

După metoda polinomială se sintetizează algoritmul de reglare, dar care este dificilă la determinarea polinomului caracteristic al sistemului proiectat [7].

În lucrare se propune procedura de acordare a regulatorului de tipul PID la modelul obiectului (1) în baza metodei gradului maximal de stabilitate cu iterații (GMSI) [8-9].

Se analizează un exemplu de acordare a algoritmului de reglare la modelul obiectului (1) cu anticipație-întârziere și astatism utilizând metodele indicate mai sus.

Algoritmii de acordare a regulatorului

Procedura de acordare a regulatorului de tipul PID după metoda GMSI se reduce la alcătuirea ecuației caracteristice a sistemului închis, în care se introduce noțiunea de grad de stabilitate J ca o nouă variabilă necunoscută și, prin operații de derivare a ecuației, se obțin relațiile $k_p = f_p(J)$, $k_i = f_i(J)$, $k_d = f_d(J)$, care prezintă parametrii algoritmului de tipul PID ca funcții neliniare de variabila J și de parametrii cunoscuți ai modelului obiectului de reglare (1).

În continuare se prezentă sistemul de funcții ale parametrilor $k_p = f_p(J)$, $k_i = f_i(J)$, $k_d = f_d(J)$ ai algoritmului P PD, PI și PID respectiv de forma:

$$k_p = \frac{a_0J^3 - a_1J^2 + a_2J}{b_1 - b_0J} = f_p(J), \tag{2}$$

$$k_d = \frac{d_0J^3 - d_1J^2 + d_2J - d_3}{(b_1 - b_0J)^2} = f_d(J),\ k_p = \frac{a_0J^3 - a_1J^2 + a_2J}{b_1 - b_0J} + k_dJ = f_p(J), \tag{3}$$

$$k_p = \frac{-d_0J^4 + d_1J^3 - d_2I^2 + d_3J}{(b_1 - b_0J)^2} = f_p(J),\ k_i = k_pJ - \frac{a_0J^4 - a_1J^3 + a_2J^2}{b_1 - b_0J} = f_i(J). \tag{4}$$

$$k_d = \frac{d_0J^5 - d_1J^4 + d_2J^3 - d_3J^2 + d_4J - d_5}{2(b_1 - b_0J)^4} = f_d(J),\ k_p = \frac{-d_0J^4 + d_1J^3 - d_2J^2 + d_3J}{(b_1 - b_0J)^2} + 2k_dJ = f_p(J),$$

$$k_i = \frac{-a_0J^4 + a_1J^3 - a_2J^2}{b_1 - b_0J} - k_dJ + k_pJ = f_i(J), \tag{5}$$

La variația lui J ca variabilă independentă, se calculează și se construiesc curbele (2)–(5) $k_p = f_p(J)$, $k_i = f_i(J)$, $k_d = f_d(J)$ pentru algoritmii PD, PI și PID și pe aceste curbe se aleg seturi de valori ai parametrilor regulatorului PD, PI și PID $J_i - k_{pi}$, k_{ii}, k_{di} şi se simulează pe calculator sistemul automat, se ridică răspunsurile indiciale, după care se determină performanțele cele mai ridicate ale sistemului, care ar satisface condițiile de performanță impuse sistemului.

Pentru un exemplu al modelului obiectului (1) se analizează procedurile de acordare a algoritmilor de reglare după metoda GMS, metoda poli-zerouri și metoda polinomială și se analizează performanțele sistemului.

Aplicații și simulare pe calculator

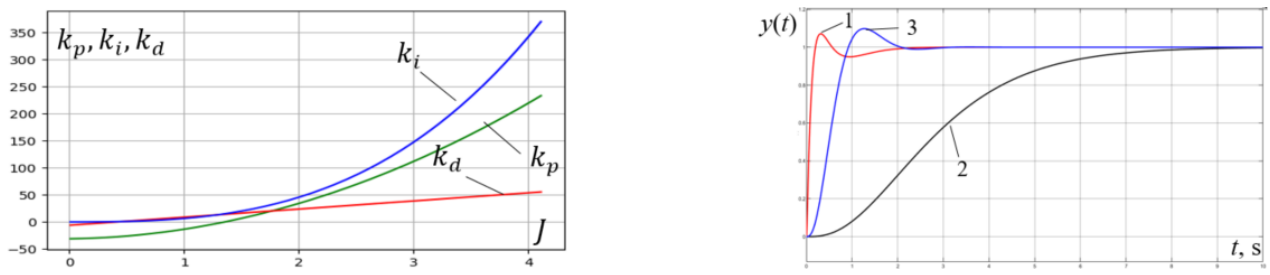
Pentru verificarea procedurii de acordare a algoritmilor PD, PI şi PID la modelul (1) după metoda GMS cu iterații se prezintă exemplu de calcul.

Exemplu. Se consideră automobilul ca obiect de reglare, descris ca modelul obiectului de reglare cu anticipație-întârziere cu oscilații și astatism de ordinul unu, cu parametrii determinați din nomograma fig. 1.4 [1] la viteza de 20 m/s: coeficientul de transfer $k = 1.9$, constantele de timp $T_1 = 0.44$ s, $T_2 = 0.04$ s^2, gradul de amortizare $q = 0.5$ şi prezentat cu funcția de transfer:

$$H(s) = \frac{k(T_1s+1)}{s(T_2^2s^2 + 2T_2qs + 1)} = \frac{b_0s + b_1}{a_0s^3 + a_1s^2 + a_2s} = \frac{0.076s + 1.9}{0.1936s^3 + 0.44s^2 + s}, \tag{6}$$

Pentru performanțele impuse sistemului automat timpul de reglare $t_r \leq 2$ s și suprareglarea $\sigma \leq 10$ % să se sintetizeze algoritmul de reglare pentru modelul (1) prin metoda GMSI

Soluționare. S-au efectuat calculele de acordare a parametrilor de acord ai regulatorului P, PD, PI, PID după metoda GMSI după relațiile (2)–(5) (fig. 1, *a*) - curba pentru P, *b*) - curba pentru PI). Din analiza curbelor algoritmului PD și PID se constată că metoda GMSI nu este aplicabilă pentru modelul (1). S-a simulat sistemul automat cu regulatorul P și PI și răspunsurile indiciale (curbele 1 și 2) sunt prezentate în fig. 2, iar performanțele sistemului sunt date în tabelul 1, rândurile 1 și 2.



a) *b)*

Fig. 1. Funcțiile $k_p = f(J)$, $k_i = f(J)$, $k_d = f(J)$

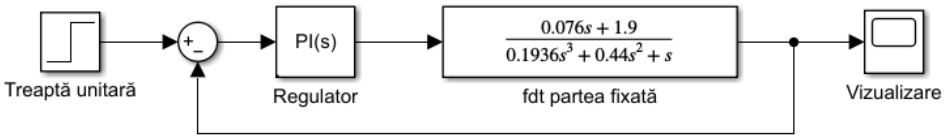


Fig. 2. Schema de simulare a sistemului automat

Tabelul 1

Performanțele sistemului automat pentru modelul obiectului din exemplu

Nr. crt.	Metoda acordare	Tip reg.	Parametrii regulatorului				Performanțele sistemului			
			J	k_p	k_i	T_i, s	t_c, s	d, %	t_r, s	n
1	GMS	P	0.7412	0.355	-	-	4.27	-	4.27	-
2	GMS	PI	0.5916	0.824	0.217	4.61	1.62	31.92	7.81	1
3	MP1						6.27	-	6.27	-
4	MP2						0.85	9.69	1.68	1
5	PZ						0.36	11.22	0.97	1

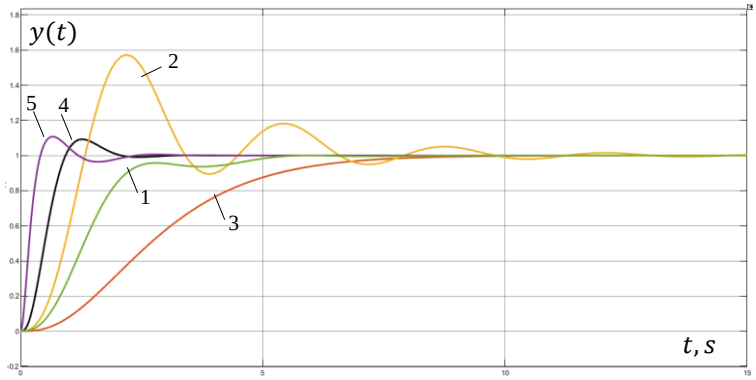


Fig. 3. Răspunsurile indiciale ale sistemului automat

Pentru verificare s-a sintetizat algoritmul de reglare după metodele polinomiale MP1, MP2 și poli-zerouri, care sunt prezentați:

$$H_R(s) = \frac{m_0\,(a_0s^2+a_1s+a_2)}{(b_0s+b_1)(n_0s^2+n_1s+n_2)} = \frac{0.1936s^2+0.44s+1}{(0.076s+1.9)(s^2+3s+3)} = \frac{0.1936s^2+0.44s+1}{0.076s^3+2.128s^2+5.928s+5.7}, \quad (7)$$

$$H_R(s) = \frac{Q(s)}{P(s)} = \frac{m_0\,(a_0s^2+a_1s+a_2)}{(b_0s+b_1)(n_0s^2+n_1s+n_2)} = \frac{22.1461s^2+50.332s+114.391}{0.076s^3+2.9641s^2+30.512s+97.7531}, \quad (8)$$

$$H_R(s) = \frac{Q(s)}{P(s)} = \frac{q_0s^2+q_1s+q_2}{p_0s^2+p_1s+p_2} = \frac{792.5127s^2+2708.2368s+6020.5789}{5.1653s^2+163.8858s+1829.516}. \quad (9)$$

S-a simulat sistemul cu algoritmii sintetizați și răspunsurile sunt prezentate în fig. 2 (curbele 3, 4, 5), iar performanțele sunt date în tabelul 1, rândurile 4, 5 și 6.

S-au variat parametrii modelului (1) cu ±50% de la valorile nominale și s-au analizat performanțele sistemului.

Concluzii

În rezultatul studiului efectuat se formulează următoarele concluzii.

1. Conform cerințelor de performanță impuse sistemului se obțin după metoda MP2.

2. Metoda GMSI de acordare este cea mai redusă în calcule și are performanțe satisfăcătoare în comparație cu metodele MP1, MP2 și PZ, dar care au calculele mai dificile.

3. Sistemul cu regulatorul acordat după metoda GMSI are gradul optimal de stabilitate $J == -0.4175$, iar MP1 $J = -0.6239$, MP2 $J = -1.1364$, PZ $J = -1.5669$.

4. La variația parametrilor modelului obiectului cu $\pm 50\%$ de la valorile inițiale sistemul cu regulatorul acordat după metoda GMSI este mai robust decât sistemul cu regulatorul acordat după metodele MP1, MP2, PZ.

Bibliografie

1. VETLINSKII, V.N., OSIPOV, A.V. Avtomaticheskie sistemy upravlenia dvijeniem avtotransporta. L.: Mashinostroenie, Leningr. otd-nie, 1986.
2. SHATALOV, A.S., TOPCHEEV, Iu. I., KONDRATIEV, B.S. Letatelnye apparaty kak obiecty upravlenia. M.: Mashinostroenie, 1972.
3. DORF, R.; BISHOP, R. Sovremennye sistemy upravlenia (Modern Control Systems). Moskva: Laboratoria Bazovyh Znanii, 2004, 832 s. ISBN 5-93208-119-8.
4. DIB, F., BENAYA, N., BEN MEZIANE, K., BOUMHIDI, I. Comparative Study of Optimal Tuning PID Controller for Manipulator Robot. The Proceedings of the International Conference on Smart City Applications SCA 2022: Innovations in Smart Cities Applications Volume 6. [Accesat 6.03.2023]. https://doi.org/10.1007/978-3-031-26852-6_23
5. SULE, A.H. Studies of PID Controller Tuning using Metaheuristic Techniques: A Review. International Journal of Innovative Scientific & Engineering Technologies Research 10(4):44-63, Oct.-Dec., 2022. ISSN: 2360-896X.
6. DUMITRACHE, I. Ingineria reglării automate. București: Politehnica Press. V.1, 2016, ISBN 978-606-515-686-9.
7. KIM, D. P. Teoria avtomaticheskogo upravlenia. T. 1. Lineinye sistemy. M.: FIZMATLIT, 2003, ISBN 5-9221-0379-2.
8. IZVOREANU, B., FIODOROV, I., The Synthesis of Linear Regulators for Aperiodic Objects with Time Delay According to the Maximal Stability Degree Method. In: Preprints the Fourth IFAC Conference on System Structure and Control, București.: Ed. Tehnică, 1997, pp. 449-454.
9. IZVOREANU, B.; COJUHARI, I.; FIODOROV, I.; MORARU, D.; SECRIERU, A. Tuning the PID Controller to the Model of Object with Inertia Second Order According to the Maximum Stability Degree Method with Iteration. Annals of the University of Craiova. Electrical Engineering series, No. 43, Issue 1, 2019, pp. 79-85. ISSN-1842-4805.

DOMAIN SPECIFIC LANGUAGE FOR SOUND PROCESSING

Mihail ECHIM[1*], Anastasia CUNEV[1], Iulian BERCU[1], Stefan NISTOR[1]

[1]*Department of Software Engineering and Automation, FAF-211,Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chisinau, Moldova*

*Corresponding author: Mihail Echim, mihail.echim@isa.utm.md

Scientific coordinator: Gabriel ZAHARIA, University Assistant, Technical University of Moldova

Abstract: *This article relays an analysis of the implementation of a Domain-Specific Language for sound processing. In this article, both the technical and the non-technical aspects of the Domain-Specific Language implementation were analyzed and it was explained in detail the process and the steps that were taken in order to develop the Domain-Specific Language for Sound Processing, as well as its rules and specifics were conveyed.*

Key words*: DSL, language, grammar, semantics, syntax, sound processing*

Introduction

Programming presents a wide variety of problems that programmers solve using various existing tools. However, sometimes a new tool is developed to make certain tasks easier such as a Domain-Specific Language (DSL). A DSL presents tools for certain specific tasks that prove more useful and easy to use than General-Purpose Languages (GPL), because DSLs are created with a specific focus in mind to accomplish certain tasks as frictionlessly as possible [1].

When creating DSLs, programmers have the opportunity to shape the tool to their needs as closely as possible which can greatly speed up development processes on bigger projects.

Additionally, DSLs can provide a higher level of abstraction than GPLs, which can make them more approachable to non-programmers or those with less programming experience. This can be especially beneficial in fields where technical knowledge is not the primary focus, such as finance, biology, or law [2]. A DSL is a user empowerment tool for increasing software system development productivity. By creating a DSL that speaks the same language as the experts in those fields, programmers can empower those experts to more easily translate their knowledge into code and automate repetitive tasks. Ultimately, DSLs provide a powerful tool for solving specific problems in a more intuitive and efficient way, and their development can greatly benefit both programmers and non-programmers alike [3].

Domain analysis

The problem that has to be solved is creating a DSL that provides a more intuitive and concise way of expressing algorithms for sound processing, making it easier for practitioners and researchers in the sound processing field to express their ideas and techniques.

The problem solution also involves the creation of a special syntax and constructs for improved readability, especially for those familiar with the sound processing domain. And in general, it will be able to do all the general sound processing techniques such as filtering, spectral analysis, and compression.

A DSL is one of the best solutions for these problems because DSLs provide a more intuitive and concise way of expressing sound processing algorithms, as compared to general-purpose programming languages. This leads to increased productivity and improved readability of code. In addition, DSLs can also provide domain-specific optimizations, leading to improved performance and scalability.

For the project, the research was conducted in the field of sound processing to explore and analyze the existing DSLs. Some popular DSLs for sound processing include Max/MSP, Pure Data, and SuperCollider [4]. The plan of the project is to look at all the advantages that the languages provide and one up them to create a truly intuitive and convenient sound processing DSL.

The project is specifically meant for audio engineers who often work with complex sound processing algorithms, and a DSL can provide a more intuitive and efficient way of expressing these algorithms. Music producers who use it for sound processing to quickly create and manipulate sounds, without having to write complex code. Researchers in the fields of audio signal processing and music technology who can use a DSL to prototype and evaluate new sound processing algorithms and techniques. And finally, people interested in music production who want to learn code in a fun and interactive way.

Grammar

In Table 1 are presented special notations, which are useful for better understanding of the grammar.

Table 1

Meta Notations

Notations	Meaning
<x>	x is a nonterminal symbol
x	x is a terminal symbol (text in **bold** font)
[x]	means zero or one occurrence of x
x*	means zero or more occurrence of x
x^+	means one or more occurrence of x
|	separates alternatives

To design a grammar, there are several tuples that need to be implemented. Every Grammar consists of 4 tuples as follows G = (V_T, V_N, P, S), where:

1. V_T - is a finite set of terminal symbols.
2. V_N - is a finite set of non-terminal symbols.
3. P - is a finite set of productions of rules.
4. S - is a start symbol [5].

S = {<program>}

V_T = {**0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, …, x, y, z, A, B, C, …, X, Y, Z, true, false, +, -, /, *, Sine, Sawtooth, Triangle, Square, lowpass, highpass, distortion, phase, time, envelope, play, record, volume, pan, reverb, gain, freq, :, ;, (,), >, <, }, {, “, func process(), func compose(), if, else, for, return, break, continue, >, >=, ==, !=, <, <=, //**}

Vn = {<program>, <statement>, <query_invocation>, <comments>, <variable_declaration>, <method_invocation>, <assignment_statement>, <identifier>, <value>, <values> <characters>, <character>, <string>, <digits>, <digit>, <non_zero_digit>, <method_name>, <comment>, <method_declaration>, <expression>, <term>, <operation>, <arithmetic_operation>, <comparative_operation>, <variable>, <constant>, }

P = {

<program> -> **func process**() {<statement>$^+$}* **func compose**() {<statement>$^+$}

<statement> -> <query_invocation>

| <comments>

| **if** (<expression>) <statement>**else** <statement>

| **for** <id> **=** <expression> **,** <expression> <statement>

| **return** <expression>

| **break**
| **continue**
| <statement>
<query_ivocation> -> <variable_declaration>
| <method_invocation>
| <method_declaration>
| <assignment_statement>
<expression> -> <term>| <expression> <operation> <expression> | (<expression>)
<term> -> <variable> | <constant>
<variable> -> <identifier>
<constant> -> <digits>
<operation> -> <arithmetic_operation>| <comparative_operation>
<arithmetic_operation> -> **+**| **-**| *****| **/**
<comparative_operation> -> **>**| **>=**| **==**| **<**| **<=**| **!=**
<variable_declaration> -> <identifier>=<value> | <identifier> =
<method_invocation>
<identifier> -> <characters>| <characters><string>
<values> -> <value> | <value>,<values>
<value> -> <digits> | <characters>
<digits> → <digit> |<digit><digits>
<digit> → **0**| <non_zero_digit>
<non_zero_digit> → **1**| **2**| **3**| **4**| **5**| **6**| **7**| **8**| **9**
<characters> -> <character> |<character><characters>
<string> -> **‘**<characters>**’** | **‘**<digits>**’**
<character> → **a**| **b**| **c** … **x**| **y**| **z** … **A**| **B**| **C**| … **X**| **Y**| **Z**
<method_invocation> -> <method_name>([<values>$^{+}$])
<method_name> -> load| delay| set_gain| set_reverb|...
<assigment_statement> -> <identifier> = <value>
| <identifier> = <identifier>
| <identifier> = <string>
<comments> → <comment>| <comments><comment>
<comment> → // <string> }

Code Example:

```
func process(){
        mysound = load(sound.mp3)
        setreverb(15 mysound)
        delay(mysound)
}
func compose(){
        play(mysound)
}
```

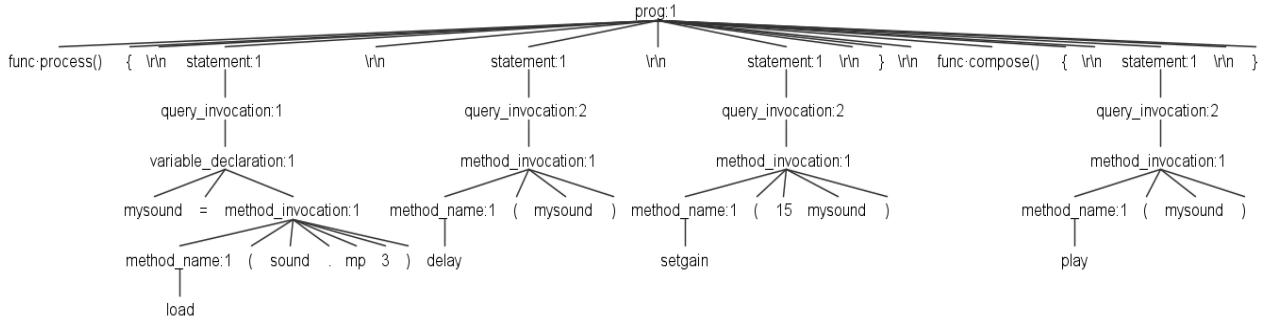


Figure 1. Parse Tree

Semantics and lexicon

In the first block of the program, the user will be able to manipulate and modify sounds. This can include changing the pitch, adding effects such as reverb or distortion, or chopping up the sound into smaller samples. The DSL will provide a variety of tools and options for the user to experiment with, allowing them to create unique and interesting sounds. Additionally, the program may allow the user to import their own audio files to use in their compositions, giving them even more flexibility and creative control.

Once the user has created and modified their sounds in the first block of the program, they will move on to the second block to compose their track. In this block, the user will be able to place their sounds on a timeline, specifying the exact moment when each sound should be played. They may also be able to adjust the volume and other parameters of each sound within the track, and add additional effects or transitions between sounds. As the user works on their composition, they may be able to listen to the track as it is being built, allowing them to make adjustments and refine the sound until it is exactly what they are looking for. Ultimately, the DSL will allow the user to express their creativity and musical ideas in a tangible way, creating a unique and personalized composition that is entirely their own.

Conclusions

In conclusion, this project aims to create a DSL for sound processing that provides a more intuitive way to express sound processing algorithms. DSLs are designed with a specific focus in mind and can provide easy-to-use tools. By creating a DSL, programmers have the opportunity to customize the tool to their specific needs, which can significantly speed up the development process for larger projects. Compared to General Purpose Language, a Domain Specific Language can provide a higher level of abstraction, which can make them more approachable to non-programmers or those with less programming experience.

References

1. FOWLER, M., PARSONS, R., *Domain Specific Languages* [online][accessed 03.03.2023]. Available: *https://martinfowler.com/books/dsl.html*
2. VOELTER, M., *DSL Engineering Designing, Implementing and Using Domain-Specific Languages* [online]. [accessed 09.03.2023]. Available: http://voelter.de/data/books/markusvoelter-dslengineering-1.0.pdf
3. BARKATI, K., JOUVELOT, P., *Synchronous programming in audio processing* [online]. [accessed 07.03.2023]. Available: https://hal-mines-paristech.archives-ouvertes.fr/hal-01540047/document
4. BARISIC, A., GOULÃO, M., AMARAL, V., *Usability Driven DSL development with USE-ME* [online]. [accessed 09.03.2023]. Available: https://hal.science/hal-03159941/file/Bari%C5%A1i%C4%87,%20Amaral,%20Goul%C3%A3o%20-%202018%20-%20Usability%20Driven%20DSL%20development%20with%20USE-ME-annotated.pdf
5. COJUHARI, I., DUCA, L., FIODOROV, I., *Formal Languages and Finite Automata* [online]. [accessed 03.03.2023]. Available: https://else.fcim.utm.md/pluginfile.php/110458/mod_resource/content/0/LFPC_Guide.pdf

IMPLEMENTAREA ALGORITMILOR DIN INTELIGENȚA ARTIFICIALĂ PENTRU DEPĂȘIREA DECALAJULUI DE COMUNICARE ÎNTRE PERSOANELE CU DIZABILITĂȚI (SURDO-MUȚI) ȘI OAMENII FĂRĂ DIZABILITĂȚI.

Dragoș COJOCARI

Departamentul Ingineria Software și Automatică, TI-214, Facultatea calculatoare, informatică și microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Moldova

Îndrumător/coordonator științific: Veronica ANDRIEVSCHI-BAGRIN, asist. univ., UTM

Rezumat. *În articolul dat se va analiza posibilitatea de a integra algoritmii din Inteligența Artificială pentru a ajuta la depășirea decalajului de comunicare între persoanele cu dizabilități (surdo-muți) și oamenii fără dizabilități. Ca exemplu a fost ales limbajul de semne americane (ASL), el fiind cel mai des folosit în lume de oamenii surdo-muți. Pentru realizarea scopului dat se va folosi algoritmul "Support Vector Machine" (SVM) din Machine Learning (ML). Toate ecuațiile prezente sunt deja bine cunoscute, demonstrațiile vor fi ilustrate grafic și reprezentate prin exemple. În articol de asemenea este prezentată exactitatea prezicerii algoritmului și explicarea cauzei erorilor și inexactități algoritmului în unele momente. Realizarea algoritmului se va face în limbajul de programare Python cu ajutorul bibliotecilor "Mediapipe", "scikit-learn", "OpenCV" si "Turtle".*

Cuvinte cheie*: Support Vector Machine, Inteligenta Artificiala, Python, limbajul de semne americane*

Introducere

Potrivit Organizației Mondiale a Sanatatii (OMS), peste 5% din populația lumii suferă de dizabilități ale auzului și surditate. Decalajul de comunicare dintre persoanele fără dizabilități și surdo-mute poate prezenta provocări pentru o comunicare eficientă, surdo-muții pentru comunicare se bazează pe limbajul semnelor, cum ar fi ASL, iar oamenii fără dizabilități se bazează pe convorbirea verbală. Persoanele fără dizabilități pot să nu fie familiarizate cu forma de comunicare prin gesturi, ceea ce poate duce la ne înțelegere și dificultăți în comunicare, pentru a depăși acest decalaj se poate folosi algoritmul SVM pentru a transforma gesturile în text în timp real.

În cele ce urmează ne interesează:

- să minimizăm timpul de răspuns al aplicației;
- exactitatea algoritmului să fie cât mai mare.

Descrierea algoritmului

În ML SVM face parte din modelele cu "Învățarea supravegheată" (SL), ceea ce înseamnă că pentru a antrena astfel de model trebuie să-i transmitem un set de date cu N atribute, în care valorile au rezultatul predefinit, iar algoritmul deja la rândul său va modifica parametrii săi astfel încât să ajungă la rezultatul dorit de noi [1].

Scopul principal al algoritmului SVM este de a găsi un hiperplan optimal într-un domeniu N-dimensional. Hiperplanul pentru domeniul N-dimensional se află în N-1 dimensiune, ca spre exemplu pentru planul bidimensional hiperplanul va fi o linie prezentată în figura 1, iar pentru planul tridimensional hiperplanul va fi un plan bidimensional prezentat în figura 2.

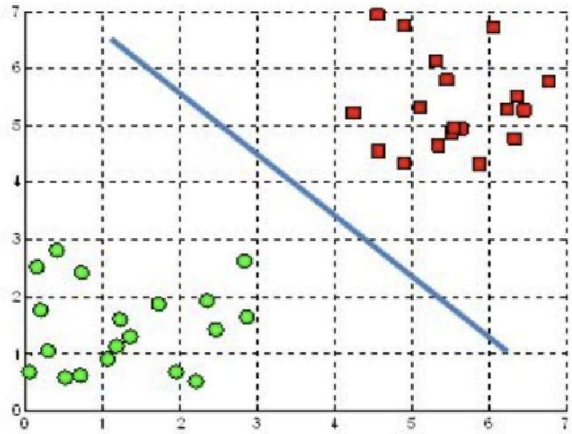


Figura 1. Hiperplanul optim pentru modelul cu două atribute (plan bidimensional)

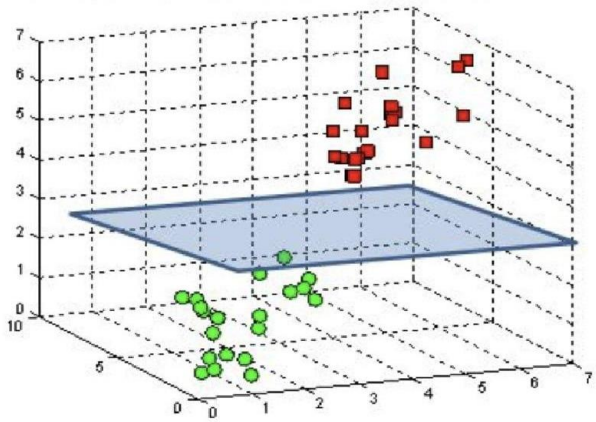


Figura 2. Hiperplanul optim pentru modelul cu trei atribute (plan tridimensional)

Hiperplanul se defineste prin formula:

$$x \cdot \underline{w} + b = 0 \tag{1}$$

unde $\underline{w}$ este n-vector, iar b este un număr real. Pentru reprezentarea grafică vom lua $\underline{w} = 1$, iar pentru calcularea distanței dintre hiperplan și punctul x vom egala formula (1) cu y.

$$x \cdot \underline{w} + b = y \tag{2}$$

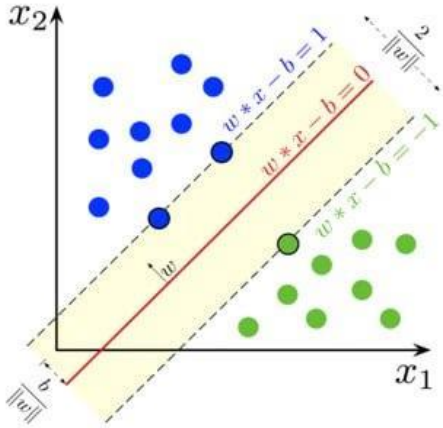


Figura 3. Distanța dintre hiperplan și punctul x

Pentru comoditate în figura 3 y a fost inlocuit cu ± 1 și vom primi că condiția de împărțire în clase pentru modelul nostru va fi:

$$y_n \left\{ \begin{matrix} +1 & x \cdot \underline{w} + b \geq 1 \\ -1 & x \cdot \underline{w} + b \leq -1 \end{matrix} \right\} \quad (3)$$

Respectiv formula (3) poate fi rescrisa ca:

$$y_n (x \cdot \underline{w} + b) \geq 0 \quad (4)$$

sau

$$min \frac{1}{2} ||w||^2 \quad (5)$$

Pentru a găsi hiperplanul optim pentru modelul nostru trebuie să alegem planul astfel încât distanța dintre vectorii suport și hiperplan sa fie maximă, vectorii suport fiind punctele care se află cel mai aproape de hiperplan și influențează poziția și orientarea sa.

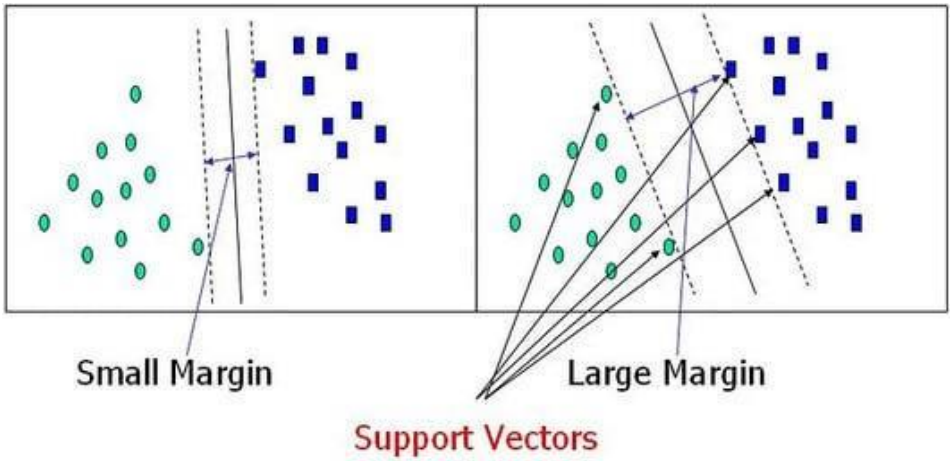


Figura 4. Vectorii suport și distanța lor de la hiperplan

În exemplele anterioare setul de date era unul ideal, fără erori sau valori aberante ce se vizualizează în figura 5, însă în realitate așa ceva nu se întâlnește și trebuie o modalitate de a aborda valorile date ca efectul lor asupra modelului să fie minimizat.

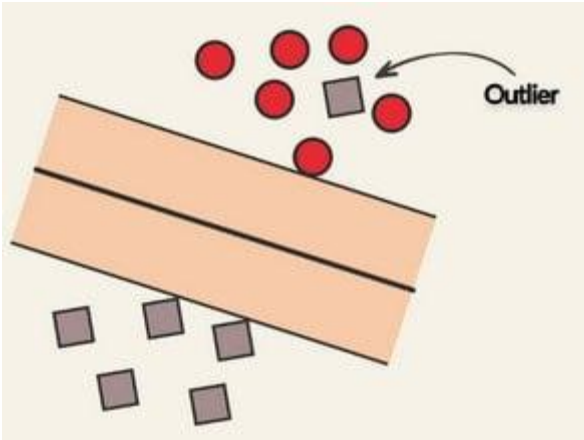


Figura 5. Valoare aberantă

Pentru a afla eroarea produsă de astfel de valoare vom folosi formula (6), graficul căruia poate fi vizualizat în figura 6.

$$\xi = max\{0, 1 - y_n(\underline{w} \cdot x + b)\} \quad (6)$$

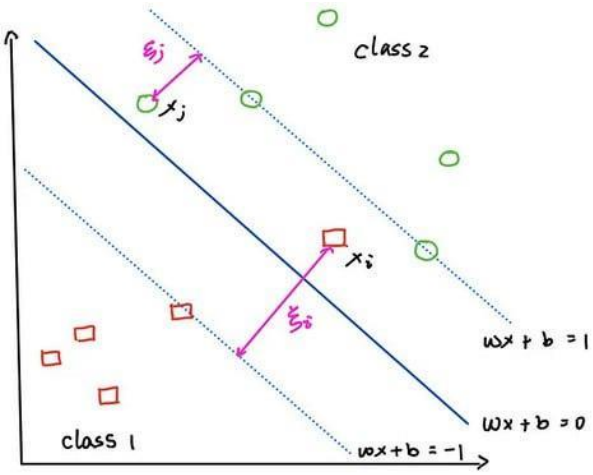


Figura 6. Eroarea valorilor aberante

Pentru a găsi eroarea produsă de toate valorile aberante vom lua suma erorilor care se calculează după formula (6):

$$\sum_{i=1}^{n} \xi_i \quad (7)$$

Unele valori aberante însă nu aduc o eroare mare în model și pot fi ignorate, pentru aceasta introducem o variabila C care va reprezenta nivelul de ignoranță față de valorile aberante și introducandu-se in formula vom primi:

$$C \cdot \sum_{i=1}^{n} \xi_i \quad (8)$$

După cate observăm din figura 7 cu cât mai mica e valoarea variabilei C cu atât mai tare se ignoră valorile aberante, iar cu cat e mai mare valoarea variabilei C cu atît mai mult se iau valorile date în considerație, aceasta fiind ilustrat în figura 7. Pentru a crea un model cât mai aproape de ideal trebuie de găsit valoarea optimă pentru C.

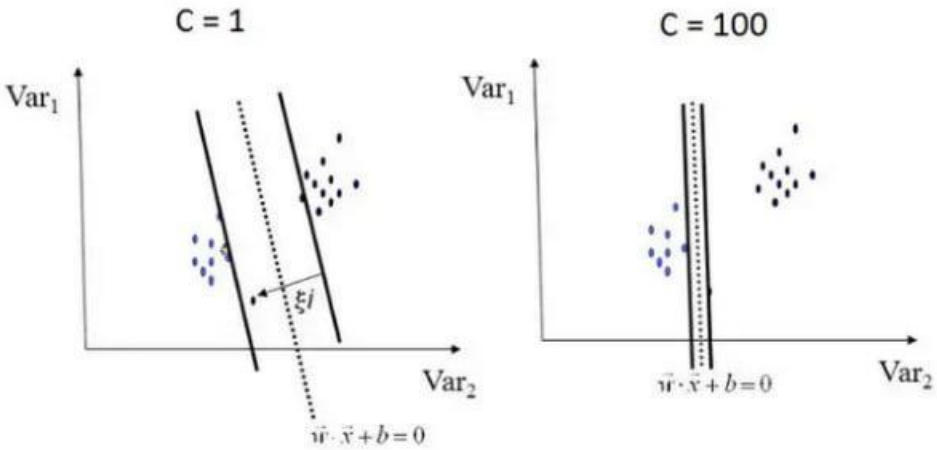


Figura 7. Impactul introducerii nivelului de ignoranță față de valorile aberante

Într-un final folosind formulele (5) și (8) vom primi formula cu ajutorul careia vom găsi hiperplanul optim:

$$min\frac{1}{2}\left|\left|w\right|\right|^2 + C \cdot \sum_{i=1}^{n} \quad \xi_i \tag{9}$$

Pentru a găsi valorile optime pentru w și b se folosesc diferite modalități, cele mai populare fiind dualitatea și descendența gradientului [2-3].

Implementarea algoritmului

Pentru implementarea algoritmului a fost ales limbajul de programare Python cu următoarele biblioteci:

- tkinter - crearea interfeței pentru aplicație;
- scikit-learn - crearea și antrenarea modelului [4];
- joblib - salvarea modelului;
- opencv - lucru cu camera web;
- mediapipe - căutarea coordonatelor mainii în spațiu.

Inițial pentru a primi coordonatele mâinii persoanei cu care vorbim vom folosi biblioteca mediapipe, care cu ajutorul la opencv capteaza imaginea și după ne transmite coordonatele la 21 de puncte de pe mână prezentată în figura 8 [5].

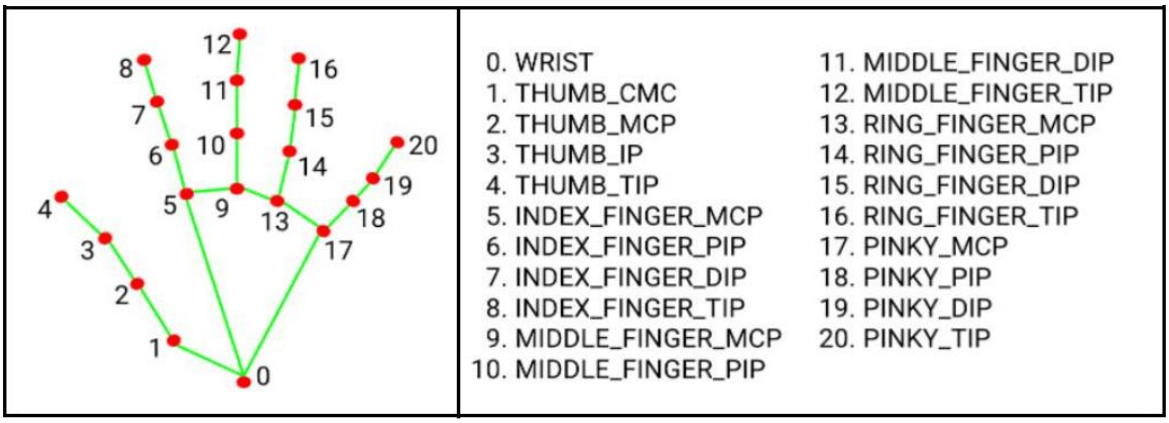


Figura 8. Punctele de pe mână luate de mediapipe.

După care informația dată se transmite modelului antrenat anticipat cu ajutorul scikit-learn, iar răspunsul dat de model va fi afișat în interfața noastră creată cu ajutorul bibliotecii tkinter ce se poate vizualiza în figura 9.

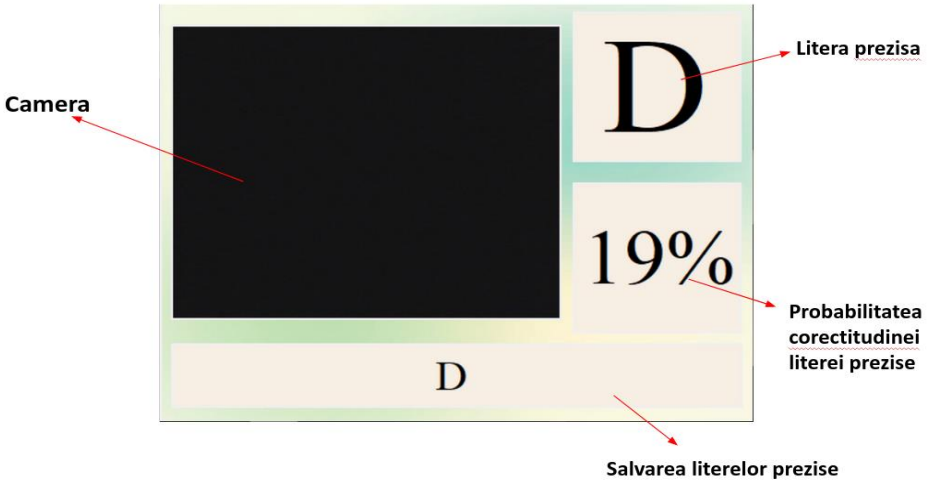


Figura 9. Interfața aplicației.

Analiza rezultatelor

Folosind funcția incorporata a bibliotecii scikit-learn vom vedea că modelul nostru are o precizie de ~82.6% și analizând rezultatele cu ajutorul matricei de confuzie figura 10 observăm că modelul nostru întâmpină greutăți cu înțelegerea literelor ‘m’ și ‘n’, deseori se încurcă ‘a’ cu ‘s’, ‘q’ cu ‘p’, ‘r’ cu ‘u’ etc. Unele din cauza apariției erorilor date pot fi setul de date necalitativ și volumul mic de informație.

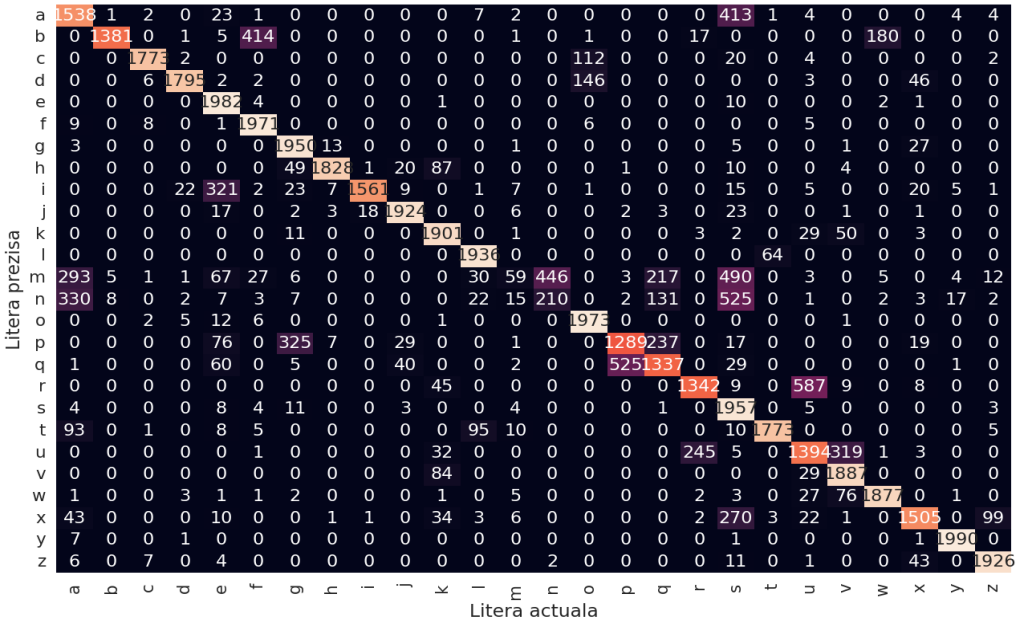


Figura 10. Matricea de confuzie al modelului

Concluzie

Ecuațiile deduse incă în anii 90 datorită hardware-ul din zilele noastre sa arătat ca fiind foarte eficiente. SVM este un algoritm care chiar și cu un set de date mic poate avea o precizie mare, în cazul nostru precizia este de aproximativ 83%, datorită diferitor posibilități care ne dă funcția, ea fiind efectivă indiferent de dimensiunea în care se folosește și are posibilitatea de a aborda valorile aberante folosind constanta *C*.

După cum observăm cu un set de date necalitativ chiar dacă ajungem la o precizie mare, oricum modelul are foarte multe erori, de unde înțelegem ca pentru a antrena bine modelul trebuie pentru început să lucrăm cu informația pe baza căreia îl vom antrena, pe lângă aceasta, valorile pentru constanta *C*, *W* și *B* deduse pentru modelul nostru nu vor fi bune pentru alte modele.

Deci pentru a crea un model cu o precizie mare cu ajutorul SVM se poate folosi un set de date mici, însă cu informație calitativă, și de asemenea ne trebuie să alegem valori optime pentru constante.

Referinţe

1. Explain Support Vector Machines in Mathematic. Details Hyperplane, maximal margin, hard-margin, soft-margin in math. [online]. [accesat 07.03.2023]. Disponibil: https://towardsdatascience.com/explain-support-vector-machines-in-mathematic-details-c7cc1be9f3b9
2. Duality (optimization) [online]. [accesat 07.03.2023]. Disponibil: https://en.wikipedia.org/wiki/Duality_(optimization)
3. Gradient Descent Algorithm — a deep dive [online]. [accesat 07.03.2023]. Disponibil: https://towardsdatascience.com/gradient-descent-algorithm-a-deep-dive-cf04e8115f21
4. Support Vector Machines [online]. [accesat 07.03.2023]. Disponibil: https://scikit-learn.org/stable/modules/svm.html
5. MediaPipe Hands [online]. [accesat 07.03.2023]. Disponibil: https://google.github.io/mediapipe/solutions/hands.html

MARKUP LANGUAGE FOR DESIGNING DIAGRAMS

Maria AFTENI[1], Sevastian BAJENOV[1], Mihaela CUȘNIR[1*], Alexandru FURDUI[1], Gabriel GÂTLAN[1]

[1]*Department of Software Engineering and Automatics, FAF 213, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chișinău, Moldova*

*Corresponding author: Mihaela CUȘNIR, mihaela.cusnir@isa.utm.md

Tutor/coordinator: Vasile DRUMEA, univ. asis., *Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chișinău, Moldova*

Abstract. *This article represents a contemplation and analysis of domain-specific language in developing a markup language for designing diagrams. Because it is already a well-known fact that UML diagrams have a considerable purpose in visualizing the future of a project (not only in the software engineering domain), this paper aims to present an efficient approach, explaining the grammar and functionality of such language in creating and editing diagrams.*

Keywords: *diagram, Domain-Specific Language, grammar, Unified Modeling Language.*

Introduction

According to Girish Managoli in his article "What developers need to know about domain-specific languages", Domain-Specific Language (also known as DSL) is a basic language created to be used in specific, but vast areas [1]. Furthermore, even if it has limited influence on users' domains, it is excellent when used in its spectrum of applicability.

Not too far, the Markup Languages is one of the DSL's spheres and noted when it comes to "annotating an electronic document" [2]. In other words, a Markup Language will "rule" how the document, or any kind of information, will be displayed. The most popular examples are HTML, XML, SGML and MathML.

Unified Modeling Language is a standardized language used in software engineering for creating visual models of software systems [3]. Beyond the idea of being used in software development, UML diagrams are used in system engineering, business analysis, database design, network engineering, and also electrical engineering. So there is no surprise that having a great Markup Language for designing diagrams is more than just a tool, it is a must have for a successful project.

Language Overview:

Generally speaking, there is no concrete concept of a data structure in a markup language. The entities existing in our DSL are being created with the help of tags, not variables, which are actually defining only the behavior of an object. Because the DSL is being created in order to simplify designing UML diagrams, then the main object of the language will be <diagram>. It will describe each entity representing a diagram and its parameters will be listed using attributes (type=state machine, for instance). The next data structures will be constructed in a smaller scale and dependent on the type of the diagram they belong to. For instance, use-case diagrams will have substructures <actor> or <use case> which on their own will be described by more attributes and interconnected using various types of relations. Relations also will be dependent on the diagram category and have their own parameters (for example, <relation type=dependency...>).A markup language for creating UML diagrams can take different inputs, including textual, graphical, programmatic, and mixed.

Our markup language relies on tags, which are a form of programmatic input. These tags will allow us to define a wide range of elements, including packages, systems, diagrams, and the relationships between them. By using tags to define these elements, can be created a highly structured

and flexible language that can be used to represent a wide range of systems, from simple diagrams to complex software architectures. As for the output, a program in this markup language for UML would produce diagrams that represent the UML elements and their relationships defined by the input tags. The output could be in a variety of formats, such as image files or vector graphics, depending on the needs of the user. Additionally, the program could also produce textual representations of the UML diagrams, which could be useful for documentation or further analysis.

Grammar

When referring to formal language theory, a grammar is a set of production rules or principles used in creating valid strings from an alphabet, except for their meaning or how they can be used in any context.

Table 1

Meta notation

Notation	Meaning
<symbol>	A nonterminal symbol
symbol	A terminal symbol
*	Repeated 0 or more times
+	Repeated 1 or more times
?	Means 0 or 1 times
|	Separates alternatives

Since UML has more than one type of diagram, the grammar for each one can differ, as presented below:

<uml> ::= <use-case-diagram>| <class-diagram> | <component-diagram> |
<deployment-diagram> | <activity-diagram> | <sequence-diagram> |state-diagram|

Aspects of the grammar:

<type> ::= **int** | **float** | **double** | **string** | **boolean** | **<class-name>**
<identifier> ::= <letter> (<letter> | <digit> | "_")
<letter> ::= **a** | **...** | **z** | **A** | **...** | **Z**
<digit> ::= **0** | ... | **9**
<text> ::= (<letter> | <digit> | <punctuation> | " ")
<punctuation> ::= **.** | **,** | **;** | **:** | **!** | **?** | **-** | **/** | **** | **"** | **"** | **(** | **)** | **[** | **]** | **{** | **}**

1. Use Case Diagram:

<use-case-diagram>::=<use-case-diagram><use-case>|<use-case-actors></use-case-diagram> <use-case> ::= <use-case> <use-case-name> </use-case>
<use-case-name> ::= “ <identifier>”
<use-case-actors> ::= <actors> λ|<actor>| <use-case-actors> </actors>
<actor> ::= “<identifier>”

2. Class Diagram:

<class-diagram> ::= <class-diagram> <class></class-diagram>
<class> ::= <class> <class-name> <class-attributes> | <class-methods> </class>
<class-name> ::= “<identifier>”
<class-attributes> ::= <attributes> <attribute> </attributes>
<attribute> ::= <type> <identifier>
<class-methods> ::= "<methods>" <method>* "</methods>"
<method> ::= <type>? <identifier> "(" <parameters>? ")" <method-body>?
<parameters> ::= <parameter> (",” <parameter>)*
<parameter> ::= “<type><identifier>”

3. Component Diagram:

<component-diagram>::=<component-diagram><component><connector></component-diagram>
<component>::=<component><component-name><component-attributes></component>
<component-name> ::=" <identifier>"
<component-attributes> ::= <attributes> <attribute></attributes>
<attribute> ::= "<type> <identifier>"

4. Deployment Diagram:

<deployment-diagram> ::= <deployment-diagram> <node> <artifact> </deployment-diagram>
<node> ::= <node> <node-name> <node-type> <node-attributes></node>
<node-name> ::= "<identifier> <node-type> "::= physical | virtual
<node-attributes> ::= <attributes> <attribute></attributes>
<attribute> ::= "<type> <identifier>"

5. Activity Diagram:

<activity-diagram> ::= <activity-diagram> <activity> <control-flow> </activity-diagram>
<activity> ::= <activity> <activity-name> <activity-nodes></activity>
<activity-name> ::= "<identifier>"
<activity-nodes> ::= <nodes> (<initial-node> | <activity-node> | <final-node> | <decision-node> | <merge-node>)</nodes>
<initial-node> ::= <initial-node> <node-name> </initial-node>
<activity-node> ::= <activity-node> <node-name> <node-attributes><activity-node>
<node-name> ::= "<identifier>"
<node-attributes> ::= <attributes> <attribute> </attributes>
<attribute> ::= "<type> <identifier>"

6. Sequence Diagram:

<sequence-diagram> ::= <sequence-diagram> <lifeline> <message></sequence-diagram>
<lifeline> ::= <lifeline> <lifeline-name> <lifeline-attributes><lifeline>
<lifeline-name> ::= " <identifier>"
<lifeline-attributes> ::= <attributes> <attribute></attributes>
<attribute> ::= "<type> <identifier>"
<message> ::= ("<synchronous-message>" | "<asynchronous-message>" | "<reply-message>") <message-name> <from> <from-lifeline> </from><to> <to-lifeline> </to> <message-attributes><message>
<message-name> ::= "<identifier>"
<message-attributes> ::= <attributes> <attribute></attributes>
<from-lifeline> ::= "<identifier>"
<to-lifeline> ::= "<identifier>"

7. State Diagram:

<state-diagram> ::= <state-diagram><state> <transition></state-diagram>
<state> ::= <state> <state-name> <state-attributes> <state-entry> <state-exit><state>
<state-name> ::= "<identifier>"
<state-attributes> ::= <attributes> <attribute></attributes>
<attribute> ::= "<type> <identifier>"
<state-entry> ::= <entry> <activity</entry>
<state-exit> ::= <exit> <activity></exit>
<transition>::=<transition><transition-name><from-state><to-state><transition-trigger><transition-guard> <transition-action></transition>
<transition-name> ::= "<identifier>"

<from-state> ::= <state-name>
<to-state> ::= <state-name>
<transition-trigger> ::= <trigger> <trigger-event></trigger>
<trigger-event> ::= “<identifier>”
<transition-guard> ::= <guard> <guard-condition></guard>
<guard-condition> ::= “<expression>”
<transition-action> ::= <action> <activity></action>
<activity> ::= “<identifier>”

Types of relationship in diagrams:

<relationship> ::= <relationship> <relationship-type> <relationship-source>
<relationship-target> |<relationship-properties> </relationship>
<relationship-type> ::= "association" | "aggregation" | "composition" | "inheritance" | "realization"
<relationship-source>::=<source><source-name> <source-cardinality>? <source-role></source>
<relationship-target> ::= <target> <target-name> <target-cardinality> <target-role> </target>
<relationship-properties> ::= <property> <property-name> = <property-value> </property>
<source-name> ::= “ <identifier>” <source-cardinality> ::= <cardinality> <cardinality-value> "</cardinality>" <source-role> ::= <role> <role-name> </role>
<target-name> ::= “ <identifier> “
<target-cardinality> ::= <cardinality> <cardinality-value> </cardinality>
<target-role> ::= <role> <role-name> </role>
<property-name> ::= “<identifier>”
<property-value> ::= “<text>”
<cardinality-value> ::= <cardinality-range> | <cardinality-exact>
<cardinality-range> ::= <integer> .. <integer>
<cardinality-exact> ::= <integer> <role-name> ::= “<text>”

Code example and its parsing Tree

Since it is known how important visualization is when working on a complex project, the derivation of the grammar is no exception. For presenting it hierarchically, parse tree helps in following the set of production rules for each diagram [4]. Beneath, in the figure 1 and 2, are examples of the parse tree for two different kinds of diagrams.

Code:

```
use-case-diagram:
actor "User" as User
rectangle Web_Records {
  usecase "SignUp" as UC1
  usecase "LogIn" as UC2
  usecase "Post" as UC3
  usecase "Edit" as UC3
}
User --> UC1
User --> UC2
User --> UC3
/use-case-diagram
```

The next figure, “Parse tree for a Use Case diagram”, follows the set of production for creating a Use Case diagram.

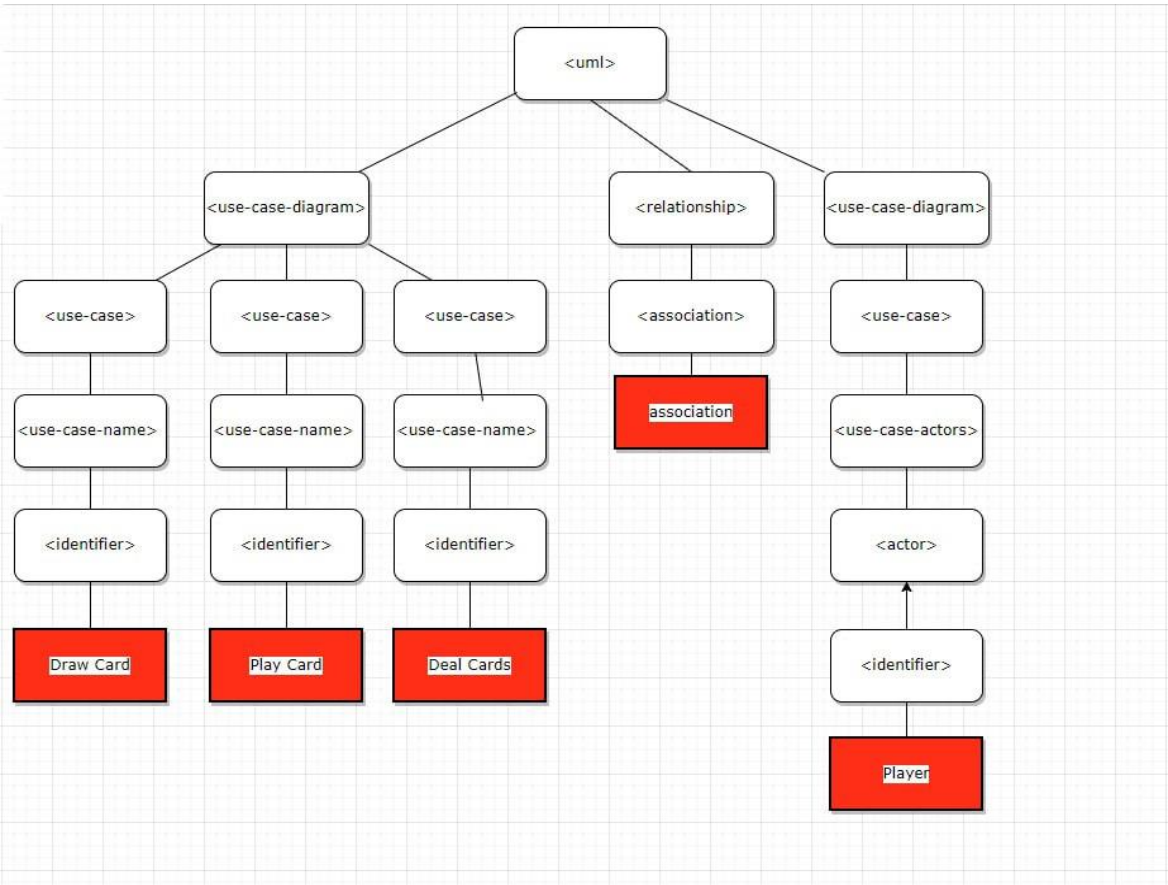


Figure 2. Parse tree for a Use Case diagram

Conclusions

Domain Specific Language is a great instrument for developers and creators around numerous domains, and there is no doubt that its subset, Markup Languages for designing diagrams, make the path of the project towards success ensured. One step, like visualization of the architecture or the project plan, can be a major and decisive stage before the final verdict. Consequently, having a flexible and manageable language for devising corresponding diagrams is indispensable. In other words, having such Markup Language is more than an advantage in efficient working.

References

1. MANAGOLI, G. *What developers need to know about domain-specific languages* [online], 2020. [accessed 06.03.2023]. Disponible: https://opensource.com/article/20/2/domain-specific-languages
2. UNIVERSITY OF LONDON, School of Advanced Study, *An introduction to markup* [online], 2022. [accessed 06.03.2023]. Disponible: https://port.sas.ac.uk/mod/book/view.php?id=568&chapterid=336
3. OBJECT MANAGEMENT GROUP, *Introduction To OMG's Unified Modeling Language™ (UML®)* [online], 2005. [accessed 06.03.2023]. Disponible: http://www.uml.org/what-is-uml.htm
4. DEEPANSHU, R. *Parse Tree in Compiler Design*, [online], 2022. [accessed 06.03.2022]. Disponible: https://www.geeksforgeeks.org/parse-tree-in-compiler-design/

ANALIZA CATEGORICĂ A ALGORITMILOR DE SORTARE

Petru CERVAC

Faculty of Computers, Informatics andMicroelectronics, Technical University of Moldova, Chișinău, Moldova

Autorul corespondent: Petru CERVAC, e-mail cervac.petru@doctorat.utm.md

Coordonator științific: Viorica SUDACEVSCHI, dr. conf. Univ., UTM

Rezumat. *Acest articol explorează posibilitatea utilizării teoriei categoriilor pentru a analiza complexitatea algoritmilor de sortare. Sunt analizați diferiți algoritmi de sortare, cum ar fi bubble sort, insertion sort, selection sort, merge sort, quicksort și heapsort, din punct de vedere al complexității lor în timp și spațiu. Apoi sunt descrise conceptele din teoria categoriilor, inclusiv functorialitatea, transformările naturale și adjuncțiile, și se explică cum acestea pot fi utilizate pentru analiza algoritmiilor de sortare într-un mod sistematic și unificat. Studiul propus în lucrare demonstrează potențialul teoriei categoriilor în analiza comportamentului algoritmilor de sortare și oferirea unei abordări pentru compararea și contrastarea diferitor algoritmi. Sunt discutate limitările și provocările utilizării teoriei categoriilor în practică și se propun căi pentru cercetarea viitoare. Lucrarea contribuie la dialogul continuu despre rolul teoriei categoriilor în informatică și complexitatea computațională.*

Cuvinte cheie*: algoritmi de sortare, teoria categoriilor, complexitate, eficiență.*

Introducere

Algoritmii de sortare sunt esențiali în informatică, servind drept fundament pentru mulți alți algoritmi. Aceștia se numără printre cei mai frecvent utilizați algoritmi în calculatoare. În plus, algoritmii de sortare sunt fundamentali pentru dezvoltarea algoritmilor mai avansați, cum ar fi căutarea binară, algoritmul lui Dijkstra și algoritmul lui Prim. În plus, algoritmii de sortare sunt frecvent utilizați în mediul academic pentru a-i introduce pe studenți în noțiunile de bază ale algoritmilor. Acest lucru se datorează faptului că sortarea poate fi realizată printr-un program de calculator relativ simplu, dar nebanal, implementat într-o varietate de moduri.

Sortarea este un proces de aranjare a datelor într-o anumită ordine, bazată pe relația dintre elemente. Aceasta poate implica sortarea datelor într-o ordine crescătoare sau descrescătoare. Deși adesea asociată cu calculatoarele, sortarea este o sarcină comună în multe scenarii din lumea reală. De exemplu, bibliotecile organizează cărțile după numele autorului și categorie, dicționarele sortează cuvintele în ordine alfabetică, facilitățile poștale sortează coletele pentru livrare, iar producătorii agricoli aranjează produse precum merele sau morcovii după mărime.

Sortarea a fost una dintre problemele pe care tehnologia a ajutat să le optimizeze. De fapt, prima mașină de sortare a fost dezvoltată în Statele Unite în secolului al XIX-lea pentru a ajuta la recensământul populației. Ulterior, John von Neumann a dezvoltat primul procedeu de sortare pentru un calculator, mai exact algoritmul de sortare prin interclasare pentru computerul EDVAC [1].

Sortarea este un subiect de mare interes printre cercetători. În ciuda aparentei sale simplități, sortarea ascunde o mulțime de mistere și întrebări fără răspuns. De exemplu, numărul minim de comparații necesare pentru a sorta un tablou de peste 47 de elemente, existența unui algoritm care utilizează mai puține comparații decât algoritmul Ford-Johnson, proiectarea unui algoritm mai eficient pentru rețele de sortare decât sortarea prin interclasare bitonică,schema optimă de divizare pentru algoritmul Shell - sunt doar unele întrebări care rămân fără răspuns.

Cea mai cuprinzătooare resursă despre sortare rămâne cartea "Arta programării calculatoarelor" de Knuth [1]. A fost lansată inițial în 1967, cea mai recentă fiind a doua ediție, lansată în 1998. Cartea descrie majoritatea familiilor de algoritmi de sortare, proprietățile matematice ale sortării, precum și analiza optimă a sortării. Cartea menționează o serie de întrebări deschise în

domeniu care au servit sursă de inspirație pentru cercetarea în secolul al XXI-lea. Un subiect pe care Knuth nu l-a inclus în cartea sa este analiza optimă a sortării bazate pe alte metode decât comparația, ceea ce este regretabil.

Sortarea este o operație crucială în informatică, iar căutarea celei mai optime metode a fost întotdeauna de mare interes. Minimizarea resurselor utilizate de un algoritm de sortare este un obiectiv important. Deși în anii 1960 au fost dezvoltate tehnici pentru analizarea complexității algoritmilor , puține activități în acest domeniu au avut loc de atunci. Autorul consideră că teoria categoriilor ar putea fi un domeniu aplicat pentru analizarea sortării și a algoritmilor în general. Deși există puține lucrări care explorează această idee, lucrările recente ale lui Hinze [2, 3] referitor la proprietățile categorice ale familiilor de sortare prin inserție și selecție și ale lui Basu [4] asupra complexității categorice, arată promițător. Lipsa cercetării în domeniul analizei algoritmice prin prisma teoriei categoriilor este surprinzătoare, dat fiind rolul său tradițional ca sursă de inspirație pentru noi dezvoltări în proiectarea limbajelor de programare, în special în limbajele de programare funcționale. Teoria categoriilor este o unealtă potențială pentru analiza performanței algoritmului și ar trebui să fie explorată în continuare în acest context.

Scopul acestui articol este studierea perspectivelor utilizării teoriei categoriilor pentru analiza complexității algoritmilor. În Capitolul 1 este discutată starea actuală a cercetării în domeniul sortării. În Capitolul 2 sunt prezentați algoritmii de sortare.Capitolul 3 este dedicat teoriei categoriilor. În Capitolul 4 este discutată complexitatea algoritmilor.

Starea actuală a cercetării

În prezent, cercetarea în domeniul de sortare se concentrează pe mai multe direcții, inclusiv dezvoltarea de proceduri noi pentru arhitecturile paralele precum GPU, procesoare multithreaded și FPGA. Începând cu mijlocul anilor 2000, legea lui Moore a fost susținută prin îmbunătățirea numărului de nuclee pe un singur procesor, dar programarea multithreaded este notoriu dificilă. În timp ce algoritmii cu un singur fir sunt mai ușor de înțeles și de implementat, crearea unei versiuni multithreaded a algoritmilor utilizați frecvent este necesară pentru a utiliza pe deplin potențialul computerelor moderne.

Dezvoltarea algoritmilor de sortare pentru arhitecturi paralele, cum ar fi GPU-urile, este deosebit de interesantă, deoarece acestea au mii de nuclee și pot fi destul de eficiente în rezolvarea anumitor tipuri de probleme. Mai multe lucrări au explorat acest domeniu [5, 6].

Sortarea pe FPGA este, de asemenea, destul de interesantă. FPGA sunt circuite complexe care pot fi reprogramate. FPGA pot fi folosite pentru a construi cipuri personalizate care rezolvă o problemă specifică foarte bine. Acestea sunt de obicei implementate în aplicații în care viteza de procesare a datelor de intrare este critică, cum ar fi avionica, automobilele și tranzacțiile. În multe astfel de aplicații, sortarea este una dintre etapele principale. Accelerarea procesului de sortare folosind FPGA ar putea fi o soluție viabilă. Există eforturi de portare a algoritmilor de sortare existenți, cum ar fi bubble sort pe FPGA [7]. Există, de asemenea, eforturi în dezvoltarea de algoritmi de sortare special concepuți pentru a fi implementați pe FPGA. Rețelele de sortare sunt deosebit de potrivite pentru FPGA [8]. Numărul minim de comparatoare necesare pentru a sorta mai mult de 16 elemente este o întrebare actuală [9], precum și dezvoltarea unui algoritm care oferă o rețea de sortare mai optimă decât bitonic mergesort [1].

Big data este un alt subiect dificil, când vine vorba de sortare. Cantitatea de date crește exponențial, iar mai multe aplicații utile sunt dezvoltate pentru a utiliza big data. Cantitatea de date limitează abordările tradiționale de procesare a datelor. Sortarea este un pas cheie în multe dintre procesările de big data, așadar este important să dezvoltăm algoritmi de sortare eficienți pentru astfel de scenarii [10].

Proprietățile teoretice ale sortării sunt, de asemenea, analizate, chiar dacă nu la fel de mult, ca în deceniile anterioare. În 2022 a fost calculat numărul minim de comparații necesare pentru a sorta 16, 17 și 18 elemente [11]. Alte cercetări în acest domeniu sunt analizate în [12] și [13]. De asemenea, sunt dezvoltate noi algoritmi de sortare de uz general [14, 15].

Un domeniu care merită mai multă atenție este analiza algoritmilor de sortare non-comparație. Cercetarea activă în această direcție a început în anii 1890 cu lucrarea lui Kirkpatrick și David [16], care a arătat limitele superioare pentru sortarea numerelor întregi pe modelul RAM nelimitat. Întrebările deschise și limitările din această lucrare au servit ca sursă de inspirație pentru cercetarea efectuată în anii 1990 și 2000 [17, 18] (Han & Thorup, 2002)). Cel mai recent articol găsit de autor este din anul 2020, în care se introduce un nou algoritm de sortare a numerelor întregi numit *RadixInsert*, utilizat pentru sortarea numerelor cu virgulă mobilă [15]. Starea actuală a cercetării algoritmilor de sortare non-comparație este importantă, deoarece acești algoritmi sunt mai eficienți decât cei bazati pe comparație, datorită utilizării tuturor proprietăților datelor de intrare și ale mașinii subiacente. Studierea algoritmilor de sortare non-comparație este utilă și pentru înțelegerea tuturor algoritmilor în general, ca un studiu de caz privind modul în care structura datelor de intrare influențează performanța algoritmului.

Clasificarea algoritmilor de sortare

Sortare internă vs externă. Algoritmii de sortare pot fi clasificați ca fiind interni sau externi, în funcție de faptul dacă înregistrările sunt păstrate integral în memoria calculatorului (intern) sau dacă se utilizează stocarea externă și doar o cantitate limitată de înregistrări este procesată într-un moment dat (extern) [16]. Algoritmii de sortare interni sunt mai des utilizați în zilele noastre, deoarece calculatoarele au o cantitate mare de memorie și nu au o problemă în stocarea cantității de date cu care interacționează un utilizator tipic. Sortarea externă este un subiect de cercetare aparte.

Sortarea bazată pe comparații vs sortarea nebazată pe comparații. Dacă un algoritm de sortare utilizează o operație de comparare, acesta este denumit sortare bazată pe comparații, în caz contrar, este un algoritm nebazat pe comparații. Cei mai cunoscuți algoritmi de sortare (ex. BubbleSort, MergeSort, SelectionSort, QuickSort etc.) sunt algoritmi de sortare bazată pe comparații. Algoritmii de sortare nebazati pe comparații (ex. BucketSort, RadixSort) utilizează proprietățile matematice ale înregistrărilor întregi pentru a elimina complet operația de comparare.

Sortare stabilă vs instabilă. O sortare este numită *stabilă* dacă înregistrările cu chei egale își păstrează ordinea originală. În caz contrar, algoritmul de sortare este considerat *instabil* [1].

Conservator vs non-conservator. Un algoritm de sortare este considerat *conservator* dacă toate rezultatele intermediare încap în memoria computerului, în caz contrar, algoritmul este *non-conservator* [16].

Sortarea prin inserție. Ideea din spatele acestei metode este de a lua un element la fiecare pas și de a-l insera într-un tablou deja sortat. Implementarea directă a algoritmului are o complexitate de timp de $O(n^2)$. Aceasta poate fi redusă prin utilizarea inserției binare și a inserției cu două direcții. Această tehnică este cea mai potrivită pentru tablouri implementate ca liste înlănțuite, deoarece acestea minimizează timpul necesar pentru inserarea elementului la loc și se reduce la înlocuirea valorii unui pointer cu unul diferit. Unul dintre cei mai cunoscuți algoritmi din această categorie este shellsort, dezvoltat de Donald L. Shell, în 1959. Acesta funcționează prin subdivizarea tabloului original în grupuri de subtablouri și inserarea elementelor în fiecare subtablou. Cu toate acestea, încă există întrebări de cercetare deschise despre acesta, cum ar fi care este cea mai bună secvență de creștere pentru valori mari ale lui N. Shellsort este locul de naștere al rețelelor de sortare, o poveste fascinantă în sine. Această tehnică este cea mai potrivită pentru structurile de date care minimizează timpul de inserție, cum ar fi listele înlănțuite. O aranjare structurată ca un arbore este, de asemenea, potrivită pentru aceste tipuri de algoritmi.

Sortarea prin interschimbare. Ideea din spatele sortării prin interschimbare este de a interschimba sistematic o pereche de elemente care sunt în ordine greșită până când nu mai există astfel de elemente. Exemple de sortări din această familie de algoritmi sunt bubble sort și quick sort.

Sortare pentru selectare. Ideea din spatele sortării prin selecție este de a selecta continuu valoarea minimă dintr-un array și plasarea ei la începutul părții sortate a array-ului. Din această familie de sortare fac parte sortarea prin selecție directă și sortarea prin intercalare (heap sort).

Sortare prin combinare. Merge sort este un algoritm de sortare popular care utilizează o strategie ***divide and conquer*** pentru a sorta o listă de elemente. Algoritmul se bazează pe împărțirea recursivă a listei în subliste mai mici, sortarea fiecărei subliste și apoi combinarea sublistelor sortate împreună pentru a produce lista finală sortată.

Sortare prin distribuție. Sortarea prin distribuție, cunoscută și sub denumirea de sortare prin găleți sau sortare radix, este un algoritm de sortare care funcționează prin gruparea elementelor unei liste în găleți, bazate pe valorile lor, apoi sortând elementele în fiecare găleată individual și, în final, combinând gălețile sortate pentru a obține lista sortată finală.

Teoria categoriilor pentru analiza complexității

Teoria categoriilor este o ramură a matematicii care studiază o structură particulară numită categorie. O categorie este o colecție de obiecte care sunt legate între ele prin morfisme. Există câteva reguli la care morfismele și obiectele trebuie să se conformeze:

- **Composabilitate**: Dacă $f: a \to b,\ g: b \to c$, atunci $g \circ f: a \to c$
- **Asociativitate**: Dacă $f: a \to b, g: b \to c$ și $h: c \to d$ atunci $h \circ (g \circ f) = (h \circ g) \circ f$
- **Identitate**: pentru fiecare obiect x, există un morfism $id_x: x \to x$ numit morfismul identității pentru x, astfel încât fiecare morfism $f: a \to x$ să satisfacă $id_x \circ f = f$ și fiecare morfism $g: x \to b$ să satisfacă $g \circ id_x = g$.

Deși conceptul de categorie poate fi considerat rudimentar, aplicabilitatea sa în formalizarea diverselor structuri matematice este extensivă. Teoria categoriilor ajută la descrierea modului în care conceptele sunt legate între ele, fiind vorba despre relațiile dintre specii de animale sau relațiile dintre structuri matematice abstracte, cum ar fi grupuri și spații topologice. Aplicarea conceptelor teoriei categoriilor a devenit o sursă semnificativă pentru stimularea inovației în domeniul dezvoltării de limbaje de programare, în special a limbajelor de programare funcționale. Haskell, un limbaj de programare funcțional prominent, este un exemplu elocvent despre modul în care ideile din teoria categoriilor au fost instrumentale în modelarea design-ului și implementării limbajului.

Teoria categoriilor nu este utilizată atât de mult în domeniul analizei computaționale. Hinze și James au utilizat teoria categoriilor pentru a analiza tehnici de sortare în programarea funcțională prin intermediul bialgebrelor și legilor distributive [3] (Hinze, James, Harper, Wu, & Magalhães, 2012). Aceștea arată dualitatea abordărilor de îmbinare și de selecție pentru sortare. Ei își continuă analiza asupra dualității metodelor de sortare într-un articol ulterior [2]. Basu introduce domeniul complexității categorice și al calculului categoric. El îl utilizează pentru a deriva complexitatea mai multor structuri matematice [4].

Autorul crede că teoria categoriilor are potențial în ceea ce privește analiza algoritmilor. În cea mai rudimentară formă, un algoritm poate fi înțeles ca o transformare între intrare și ieșire. Intrarea și ieșirea au anumite structuri asociate lor. Înțelegerea structurii intrării și ieșirii va aduce evident o mai bună înțelegere a transformării dintre ele. Un algoritm de sortare poate fi perceput ca o transformare între categoria de liste nesortate și categoria de liste sortate.

Complexitate și eficiență

Algoritmii de sortare sunt caracterizați prin complexitatea lor în timp și spațiu, care sunt măsurile fundamentale ale eficienței lor. Complexitatea în timp se referă la cantitatea de timp necesară pentru ca un algoritm să ruleze în funcție de dimensiunea datelor de intrare. Complexitatea în spațiu, pe de altă parte, se referă la cantitatea de memorie necesară pentru ca un algoritm să ruleze în funcție de dimensiunea datelor de intrare. Aceste complexități sunt adesea exprimate utilizând notația big O, care oferă o limită superioară a ratei de creștere a utilizării resurselor algoritmului în funcție de datele de intrare.

Notația big O este definită ca fiind mulțimea tuturor funcțiilor care nu cresc mai rapid decât o anumită funcție dată, până la un factor constant. Mai formal, $O\big(f(n)\big)$ denotă mulțimea tuturor funcțiilor $g(n)$ astfel încât există constante pozitive C și n_0 astfel încât $|g(n)| \leq Cf(n)$, pentru toate $n \geq n_0$. Notația big Omega, denumită $\Omega\big(f(n)\big)$, furnizează o limită inferioară a ratei de creștere a

utilizării resurselor algoritmului în funcție de datele de intrare. Conceptul de notație big O și big Omega pentru analizarea performanței algoritmilor a fost popularizat de Donald Knuth.

Analiza performanței algoritmilor este o problemă complexă și subiectivă la mulți factori, inclusiv arhitectura calculatorului, calitatea implementării, limbajul de programare și altele. Există două abordări principale pentru analiza algoritmilor: teoretică și empirică.

În abordarea empirică, un algoritm este executat pe un hardware specific cu diferite dimensiuni ale datelor de intrare pentru a observa cantitatea de resurse pe care o consumă. Această abordare este utilă pentru a determina comportamentul unui algoritm pe hardware-ul real și poate ajuta la răspunsul unor întrebări care sunt dificil de rezolvat teoretic. Cu toate acestea, această abordare are și dezavantaje, inclusiv riscul de a afecta accidental performanța prin arhitectură, imposibilitatea de a răspunde la toate întrebările și dificultatea de a valida rezultatele pentru toate scenariile.

Analiza teoretică a algoritmilor implică utilizarea unui model computațional, cum ar fi o mașină Turing sau lambda calculus, pentru analiza algoritmilor. Avantajul acestei abordări este că permite demonstrația matematică a rezultatelor, iar operațiile definite în model pot fi folosite pentru a deduce numărul de resurse utilizate în funcție de dimensiunea datelor de intrare. Cu toate acestea, rezultatele analizei teoretice pot diverge de performanța reală, deoarece modelele de calcul sunt adesea mai simple decât arhitecturile realizate, iar operațiile definite în modele pot fi departe de implementările din lumea reală.

Analiza sortării bazate pe comparații. Numărul de comparații utilizat de un algoritm este un factor important pentru determinarea eficienței sale, deoarece comparațiile sunt operații costisitoare, în mod tradițional. Prin minimizarea numărului de comparații necesare, putem îmbunătăți eficiența algoritmului. Se știe că un algoritm de sortare bazat pe comparații are o complexitate a timpului de cel mult $O(n \log n)$ datorită cantității de informații pe care o oferă fiecare comparație. Cu toate acestea, nu există un algoritm cunoscut care să atingă această limită teoretică pentru toate dimensiunile datelor de intrare. Determinarea numărului exact de comparații necesare pentru a sorta un număr dat de intrări este o problemă computațională dificilă. Merge Insertion sort, cunoscut și sub numele de algoritmul de sortare Ford-Johnson, este în prezent cel mai eficient algoritm cunoscut pentru sortarea listelor cu mai puțin de 47 de elemente [19]. Acesta atinge limita inferioară teoretică a informației pentru $N \leq 22$, iar numărul exact de comparații necesare a fost demonstrat pentru matrice cu mai puțin de 22 de elemente [1, 11, 20, 21]. Cu toate acestea, algoritmul nu este cel mai eficient pentru toate valorile de N [22].

Analiza sortării non-bazate pe comparații. Algoritmii de sortare non-bazați pe comparații au, de obicei, o performanță asimptotică mai bună decât cei care se bazează pe comparații, deoarece nu sunt restricții la utilizarea doar a operatorului de comparație. În schimb, aceștia pot folosi mai multe informații disponibile în cheie. Algoritmii de sortare non-bazați pe comparații pot gestiona structuri de chei mai complexe, cum ar fi numerele întregi, ceea ce permite utilizarea unei game mai largi de operații. Este important să se aleagă modelul corect de calcul atunci când se evaluează performanța algoritmilor de sortare. Limita inferioară pentru sortarea bazată pe numere întregi a fost determinată a fi $O(n)$, deși pentru constante prea mari pentru a fi practice [16]. În timp ce algoritmii teoretici de sortare a numerelor întregi au o performanță asimptotică bună, aceștia nu sunt practici pentru a fi implementați pe hardware real. Cercetarea în domeniul sortării non-bazați pe comparații a început în anii 1980, cu cei mai eficienți algoritmi de sortare non-bazați pe comparații, și anume counting sort, radix sort, bucket sort, RadixInsert și Signature sort [18]. Acești algoritmi sunt, de asemenea, cunoscuți sub numele de algoritmi de sortare bazate pe numere întregi, care au realizat limite teoretice inferioare mari. Cercetarea curentă se concentrează pe furnizarea de implementări paralele ale algoritmilor de sortare a numerelor întregi, în special pe GPU. În prezent, eforturile sunt îndreptate spre cercetarea algoritmilor de sortare a numerelor întregi care pot fi implementați pe o arhitectură paralelă. Algoritmii de sortare conservativă, care utilizează doar comparații și sunt concepuți pentru o gamă fixă de chei, au două idei importante pentru algoritmii de sortare a numerelor întregi: reducerea gamei și sortarea împachetată. O implementare interesantă a sortării întregi este AC^0, care poate fi făcută în timp $O(n)$ și cu cerințe de spațiu $O(1)$ [23].

Concluzii

În concluzie, analiza complexității algoritmilor este crucială în găsirea celei mai eficiente modalități de a realiza sarcina propusă. Teoria categoriilor poate furniza un framework util pentru analiza algoritmilor, iar algoritmii de sortare, în special, sunt un punct de plecare excelent datorită argumentării matematice solide. Combinând teoria categoriilor cu algoritmii de sortare, putem explora întrebări interesante precum definirea de vectori și vectori sortați în contextul categoric, precum și exprimarea proprietăților subiacente ale datelor. Aceste întrebări reprezintă domenii importante pentru cercetare care ar putea duce la noi perspective în analiza algoritmilot. În general, intersecția teoriei categoriilor și a algoritmilor de sortare prezintă o direcție promițătoare pentru avansarea înțelegerii noastre a algoritmilor și a complexității lor.

Bibliografie

1. D. E. KNUTH, The Art of Computer Programming: Volume 3, vol. 3, Addison-Wesley Pub. Co., 1997.
2. R. HINZE, J. P. MAGALHÃES ȘI N. WU, „A Duality of Sorts,” în Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2013, p. 151–167.
3. R. HINZE, D. JAMES, T. HARPER, N. WU și J. MAGALHÃES, „Sorting with Bialgebras and Distributive Laws,” 2012.
4. S. BASU și M. U. ISIK, „Categorical Complexity,” Forum of Mathematics, Sigma 8 (2020) e34, vol. 8, 2016.
5. S. ABDEL-HAFEEZ, A. GORDON-ROSS și S. ABUBAKER, „A comparison-free sorting algorithm on CPUs and GPUs,” The Journal of Supercomputing, vol. 74, p. 6369–6400, 1 November 2018.
6. N. F. A. S. GHRERA, „Performance Evaluation of Parallel Count Sort using GPU Computing with CUDA,” Indian Journal of Science and Technology, vol. 9, p. 1–12, 14 April 2016.
7. N. LEISCHNER, V. OSIPOV și P. SANDERS, „GPU sample sort,” în 2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS), 2010.
8. H. PETERS, O. SCHULZ-HILDEBRANDT și N. LUTTENBERGER, „Fast In-Place Sorting with CUDA Based on Bitonic Sort,” în Parallel Processing and Applied Mathematics, Berlin, 2010.
9. P. PAPAPHILIPPOU, W. LUK și C. BROOKS, „FLiMS: A Fast Lightweight 2-Way Merger for Sorting,” IEEE Transactions on Computers, vol. 71, p. 3215–3226, December 2022.
10. T. EHLERS, "Merging almost sorted sequences yields a 24-sorter," Information Processing Letters, vol. 118, p. 17–20, February 2017.
11. D. BUNDALA and J. ZÁVODNÝ, "Optimal Sorting Networks," in Language and Automata Theory and Applications, Cham, 2014.
12. SHATNAWI, Y. ALZAHOURI, M. A. SHEHAB, Y. JARARWEH și M. AL-AYYOUB, „Toward a new approach for sorting extremely large data files in the big data era,” Cluster Computing, vol. 22, p. 819–828, 1 September 2019.
13. F. STOBER și A. WEIß, „Lower Bounds for Sorting 16, 17, and 18 Elements,” 2022.
14. K. IWAMA și J. TERUYAMA, „Improved average complexity for comparison-based sorting,” Theoretical Computer Science, vol. 807, p. 201–219, 6 February 2020.
15. S. EHARA, K. IWAMA și J. TERUYAMA, „Small Complexity Gaps for Comparison-Based Sorting,” H. Böckenhauer, D. Komm și W. Unger, Ed., Cham, Springer International Publishing, 2018, p. 280–296.
16. S. P. Y. FUNG, „Is this the simplest (and most surprising) sorting algorithm ever?,” October 2021.
17. MAUS, „RadixInsert, a much faster stable algorithm for sorting floating-point numbers,” 2020.
18. D. KIRKPATRICK and S. REISCH, "Upper bounds for sorting integers on random access machines," Theoretical Computer Science, vol. 28, p. 263–276, January 1983.

19. Y. HAN și M. THORUP, „Integer sorting in O(n/spl radic/(log log n)) expected time and linear space,” în The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings., 2002.
20. M. THORUP, "Randomized Sorting in O(nloglogn) Time and Linear Space Using Addition, Shift, and Bit-wise Boolean Operations," Journal of Algorithms, vol. 42, p. 205–230, February 2002.
21. ANDERSSON, T. HAGERUP, S. NILSSON and R. RAMAN, "Sorting in Linear Time?," Journal of Computer and System Sciences, vol. 57, p. 74–93, August 1998.
22. L. R. FORD și S. M. JOHNSON, „A Tournament Problem,” The American Mathematical Monthly, vol. 66, p. 387–389, May 1959.
23. M. PECZARSKI, "New Results in Minimum-Comparison Sorting," Algorithmica, vol. 40, p. 133–145, October 2004.
24. M. PECZARSKI, "The Ford–Johnson algorithm still unbeaten for less than 47 elements," Information Processing Letters, vol. 101, p. 126–128, February 2007.
25. G. K. MANACHER, „The Ford-Johnson Sorting Algorithm Is Not Optimal,” Journal of the ACM, vol. 26, p. 441–456, July 1979.
26. G. FRANCESCHINI, S. MUTHUKRISHNAN ȘI M. PATRASCU, „Radix Sorting With No Extra Space,” 2007.

VINSENT: AN IOT SOLUTION FOR OPTIMAL WINE BARREL MANAGEMENT IN MOLDAVIAN WINERIES

Vlada MAGAL[1], Irina TIORA[1], Viorel NOROC[1], Marcel VLASENCO[1], Maria-Mădălina UNGUREANU[1*]

[1]*Department of Software Engineering and Automatics, FAF-203, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chișinău, Republic of Moldova*

*Corresponding author: Maria-Mădălina Ungureanu, maria-madalina.ungureanu@isa.utm.md

Coordinator: Alexei MARTÎNIUC, research scientist, Department of Software Engineering and Automatics, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova

Abstract. *This piece of work was produced as part of an IoT project that follows Problem Based Learning (PBL) methodology. The project focuses on providing an accurate and efficient IoT solution to monitor the fermentation and ageing processes in winemaking. The usage of this device can result in a better final product that is more consistent and has higher quality.*

Keywords: *Internet of Things (IoT), sensor technology, real-time monitoring, winemaking*

Introduction

The art of producing wine has been practised for thousands of years. Nowadays it evolved into a sophisticated and intricate process, involving a blend of art, science and tradition.

Winemakers face many challenges in the production process, especially in monitoring the fermentation process and ageing. The process requires careful observation of temperature, yeast, and oxygen levels. If the fermentation conditions are not just right, the wine may spoil or turn into vinegar. This can also help the winemaker to identify any potential issues with the ageing process, such as fluctuations in temperature or oxygen levels.

Traditional methods for monitoring this process can be time-consuming, inaccurate, and inefficient, requiring frequent manual checks of each barrel.

Problem statement

Monitoring the temperature, oxygen, and timing is critical in wine fermentation and ageing. High temperatures can cause yeast to die, while low temperatures can slow down or stop the fermentation process. Exposure of wine to a limited amount of oxygen can result in enhancing the wine's flavour by making it deeper and more complex, but too much oxygen can cause wine to oxidize and develop off-flavours. Timing is also critical, as fermentation and ageing processes must be timed correctly to achieve the desired alcohol level and flavour profile [1-2].

Other important factors that need to be monitored in winemaking are humidity, light, pH and CO_2 also it's beneficial to check the level of wine in the barrels to avoid leaks. Humidity levels should be kept constant and relatively high (75-85%) to prevent the evaporation of alcohol and other volatile compounds from the wine. If the relative humidity is lower than this range, it can accelerate the wine's maturation, which could result in a substandard product. Conversely, if the relative humidity exceeds 85%, there is a possibility of fungus and mould growth, which can affect the wine's quality. Exposure to light, especially UV light, can cause wine to develop an unpleasant flavour. The pH of the must (grape juice) is an important factor in the fermentation process. Yeast performs best at a pH between 3.0 and 4.0, so winemakers may need to adjust the pH of the must. Generally, winemakers want to maintain a certain level of CO_2 in the barrel during fermentation to ensure that the yeast is able to carry out the fermentation process properly [3-5].

Objectives

As Moldavian wines are continuously gaining popularity and recognition at various wine competitions, our team wants to help local winemakers make good quality wine that will increase interest in Moldavian products.

The main objective of this project is to develop a smart cork device that can be inserted into wine barrels. It must increase the efficiency of the fermentation and wine ageing procedures by reducing the need for manual checks and adjustments. This can allow winemakers to focus on other aspects of the winemaking process and increase overall productivity.

The proposed project aims to develop a functional prototype of the device, evaluate its performance, and provide recommendations for its commercialization and implementation. The outcomes of the project could be of significant value to the wine industry in Moldova.

Solution concept

The development of VinSenT - a smart cork device designed for winemaking process monitoring using sensor technology, could provide a more accurate and efficient method for observing fermentation and ageing processes, allowing winemakers to optimize them. The device will be designed and developed using ESP32 MCU [6] and smart sensor technology. It will be equipped with temperature, pressure, humidity, and other sensors to collect data on the wine-making processes. The data collected will be transmitted wirelessly to a centralized system for real-time monitoring and analysis. A diagram of the project architecture can be viewed in Fig. 1 from below.

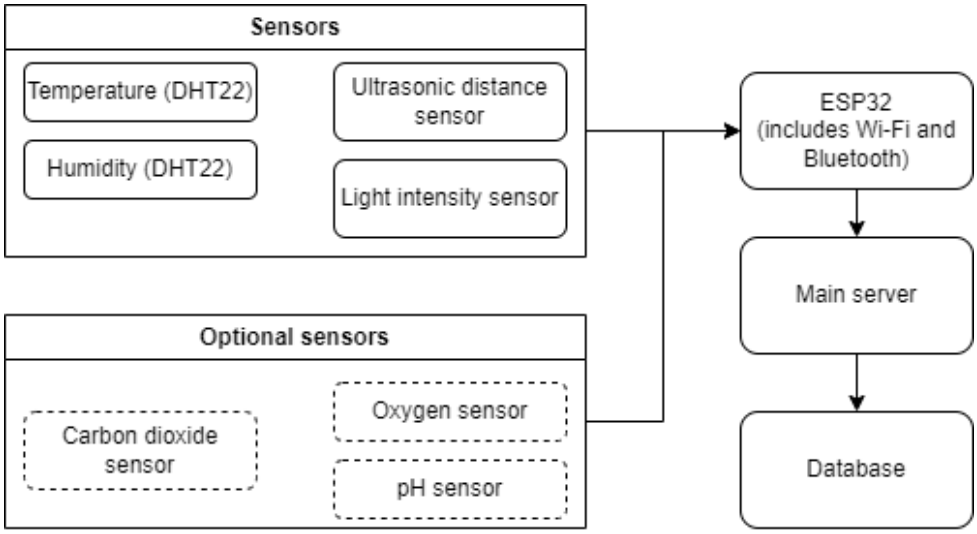


Figure 1. Project Architecture

The IoT project will involve various sensors to collect temperature, humidity, light, distance, and potentially oxygen, CO2, and pH data. The gathered information will be transmitted to the ESP32 MCU, which will then send all the collected data to the server. This data will be presented in an application that will notify the user in case of any deviations from the desired settings.

Conclusion

To sum up, the creation of VinSenT (Wine Sensor Technology), a device equipped with smart sensors for monitoring the winemaking process, can enhance the precision and effectiveness of observing fermentation and ageing processes in winemaking. This can enable winemakers to improve the quality and consistency of the final product while optimizing productivity. The project's outcomes have the potential to bring substantial benefits to the wine industry in Moldova.

References

1. What's the best temperature for wine fermentation? [online]. 2020. [cited 03.03.2023]. Available: https://olavin.com/wine-fermentation-temperature.html
2. What is Oxidation and What Is It Doing to Your Wine? [online]. 2019. [cited 03.03.2023]. Available: https://www.jjbuckley.com/wine-knowledge /blog/what-is-oxidization-and-what-is-it-doing-to-your-wine-/1015
3. Is Sunlight Bad For Wine? [online]. 2021. [cited 05.03.2023]. Available: https://dunavox.com/page/news/details/is-sunlight-bad-for-wine_109
4. Humidity Control in Wine Cellars and Barrel Rooms. [online]. 2023 [cited 05.03.2023] Available: https://www.coolmist.com.au/humidity-control-in-wine-cellars-and-barrel-rooms
5. The Importance of pH in Wine Making. [online]. 2022. [cited 05.03.2023]. Available: https://sensorex.com/2022/11/27/ph-wine-making/
6. Internet of Things with ESP32. [online]. 2018. [cited 08.03.2023]. Available: http://esp32.net/

DOMAIN SPECIFIC LANGUAGE FOR GEOMETRIC CALCULATIONS

Ion ROTARU[1], Alexandru GRAMA[1*], Ahmed AL HAJ[1], Mihai CORETCHI[1], Maxim ZADAROJNII[1]

[1]*Technical University of Moldova, Faculty of Computers, Informatics and Microelectronics, Software Engineering and Automation Group FAF-211, Chisinau, The Republic of Moldova*

*Corresponding author: Alexandru Grama

Coordinator: Catruc Mariana

Abstract: *This article analyzes the use of a domain-specific language for mathematics using geometric shapes and bodies. The paper analyzes both technical and non-technical topics with the intention of describing the Domain Specific-Language implementation process in detail, step-by-step, and highlighting the priorities and regulations. The fundamental characteristics of the Domain Specific-Language, as well as the fundamental semantic rules and vocabulary of the grammar for the Domain Specific-Language, were stated while highlighting the prerequisites and branches that should be included in the language.*

Keywords: *Domain-Specific Language, language, grammar, semantics, syntax, geometric calculations.*

Introduction

A Domain-Specific Language (DSL) is a computer language that's targeted to a particular kind of problem, rather than a general purpose language that's aimed at any kind of software problem. Domain-specific languages have been talked about, and used for almost as long as computing has been done [1].

A Domain specific language is usually less complex than a general-purpose language, such as Java, C, or Ruby. Generally, DSLs are developed in close coordination with the experts in the field for which the DSL is being designed. In many cases, DSLs are intended to be used not by software people, but instead by non-programmers who are fluent in the domain the DSL addresses.

There are two fundamentally different ways of how traditional code and DSL code can be integrated. The first one keeps DSL code and regular code in separate files. The DSL code is then transformed into programming language code by an automated code generator, or alternatively the program loads the domain-specific code and executes it. This first approach, with separated General Purpose Language (GPL) and DSL code is termed external DSLs. Think of SQL, MATLAB as an example of an external DSL [2].

1. Domain description and analysis

A Domain-Specific Language for geometry could address a wide range of domains related to geometry, including: Computer-Aided Design, Computer Graphics and Animation, Robotics, Architecture and Construction and Computational Geometry. The propoused DSL will address Computational Geometry, which means that it could be used to solve mathematical problems related to geometry, such as calculating intersections, distances, or areas between shapes, or performing geometric transformations. This could include tools for solving geometric algorithms or defining geometric primitives.

A programming language called a Domain-Specific Language is created to handle a particular issue domain. A DSL might be developed to aid in the description and manipulation of geometric forms in the case of geometry, enabling more succinct and expressive code.

Here are some particular ways that a DSL may help geometry:

- Code that is clear and easy to read: a geometry DSL might utilize a syntax that is designed specifically for the description of geometric forms.
- Higher-level abstractions: a geometry DSL might offer more expressive and understandable code by providing higher-level abstractions for frequent geometric notions like points, lines, circles, and polygons.
- Greater type checking: a geometry DSL should include tighter type checking to avoid frequent errors like adding points to lines or computing the intersection of forms that are not intersecting.
- Code reuse: a geometry DSL might offer reusable parts for routine tasks like calculating a shape's area or locating the closest point on a line.
- Integration with additional tools: a geometry DSL may be used with additional tools, such as visualization libraries or computer-aided design (CAD) software, to provide smooth integration across various phases of a geometric modeling pipeline.

In conclusion, a geometry DSL may offer a more effective and expressive approach to interact with geometric forms, making it simpler to build, modify, and examine complicated models.

A DSL for geometry can help solve several problems related to expressing and manipulating geometric concepts in a more intuitive and efficient way. Some of these problems include:

- Expressing geometric concepts: a DSL for geometry can provide a more natural and intuitive way to express geometric concepts, such as points, lines, curves, and surfaces. This can make it easier for users to create and manipulate geometric shapes and models.
- Improving accuracy: geometry can be complex, and small errors in geometric calculations can lead to significant inaccuracies in the final results. A DSL for geometry can help improve the accuracy of geometric calculations by providing built-in functions and methods for common geometric operations.
- Increasing productivity: a DSL for geometry can help users be more productive by automating repetitive tasks and reducing the need for manual calculations. This can save time and reduce the risk of errors.
- Facilitating collaboration: a common language for expressing geometric concepts can facilitate collaboration between different stakeholders, such as engineers, designers, and architects. This can help ensure that everyone is on the same page and working towards the same goals.

Overall, a DSL for geometry can help solve several problems related to expressing and manipulating geometric concepts, improving accuracy, increasing productivity, and facilitating collaboration

DSL in geometry can be used by various users who need to express geometric concepts or perform geometric computations in a domain-specific context. Some potential users of DSL in geometry are:

- Mathematicians: mathematicians who work on geometry or related fields can use DSL in geometry to formalize geometric concepts and theorems.
- Engineers: engineers who work on fields like computer-aided design (CAD), computer graphics, or robotics can use DSL in geometry to define and manipulate 2D or 3D geometric models.
- Architects: architects can use DSL in geometry to specify and manipulate geometric models of buildings, including their shapes, dimensions, and orientations.
- Designers: product designers and industrial designers can use DSL in geometry to create and manipulate 3D models of their designs.
- Scientists: scientists who work on fields like physics or chemistry can use DSL in geometry to represent and simulate molecular structures and their interactions.
- Educators: educators can use DSL in geometry to teach geometry in a more interactive and engaging way, by allowing students to create and manipulate geometric models.

These are just some examples of potential users of DSL in geometry, but there could be many other users depending on the specific domain or application.

2. Grammar

For a better understanding, further is represented the grammar for this specific language according to a very simple and textual program. Through it, was shown in detail each feature of grammar.

The DSL design includes several stages. First of all, definition of the programming language grammar $G = (V_N, V_T, P, S)$:

V_N – is a finite set of non-terminal symbol;
V_T - is a finite set of terminal symbols.
P – is a finite set of production of rules;
S - is the start symbol;

In Table 1 are meta-notations used for specifying the grammar.

Table 1

Meta notation

Notation (symbol)	**Meaning**
<foo>	means foo is a nonterminal
foo	foo in bold means foo is a terminal
x*	zero or more occurrences of x
|	separates alternatives
→	derives
//	comment section

S = {<source code>}

V_T = {START, 0.9, A…Z, a...z, true, false, Point, Line, Segment, Triangle, Square, Rectangle, Parallelogram, Trapezoid, Rhombus, Circle, Ellipse, Cube, Sphere, Cylinder, Cone, Pyramid, length, angle, radius, diagonal, median, bisector, vertex_name, angle_name, area, perimeter, volume, ., , , :, (,), _, “, ”, /, +, -, *, ^, sin, cos, ctg, tg, END }

V_N = {<source code>, <method name>, <methods invocation>, <decimal numeral>, <floating-point>, <digits>, <non zero digit>, <boolean literal>, <characters>, <string>, <string characters>, <identifier>, <type>, <numeric type>, <variable declaration>, <variables declaration>, <method invocation>, <expression>, <comments>, <comment>}

P = {<source code> → **START***<variables declaration><methods invocation>*<comments>***END**

<variables declaration> → <variable declaration>|<variables declaration ><variable declaration>

<variable declaration> → <type><identifier>

<type> → **Point** | **Line**| **Segment** | **Triangle** | **Square** | **Rectangle** | **Parallelogram** | **Trapezoid** | **Rhombus** | **Circle** | **Ellipse** | **Cube** | **Sphere** | **Cylinder** | **Cone** | **Pyramid** | **…**

<identifier> → (<character> | _) (<character> | <digits> | _) *

<character> → **a** | **b** | **c** |**…**| **A** | **B** | **C** | **.** | **Z**

<digits> → <digit> | <digits> <digit>

<digit> → **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9**

<methods invocations> → <method invocation>|<methods invocation><method invocation>

<method invocation> → <identifier>.<method name>(<argument list>*)
<method name> → **length** | **angle** | **radius** | **diagonal** | **median** | **bisector** | **vertex_name** | **angle_name** | **area** | **perimeter** | **volume** | **…**
<argument list> → <expression>|<argument list>,<expression>
<expression> → <numeric type> | <string> | <boolean literal>
<numeric type> → <decimal numeral> | <floating-point>
<floating-point> → <decimal numeral>.<decimal numeral>
<decimal numeral> → <digits>*
<string> → "<string characters>*"
<string characters> → <characters>*<digit>*
<boolean literal> → **true** | **false**
<comments> → <comment>|<comments> <comment>
<comment> → // <string>
}

3. Semantic and lexicon

The program will be split into two sections. The user defines the name and type of the variables in the first step, the private variable declaration. The second section consists of a method invocation in which the user requests that specific parameters, such as an object's area or the volume of a 2D or 3D figure, be calculated based on other values that have already been input.

There must be an underscore (_) in place of white space between words, such as in vertex_A. This will be shown in the suggested grammar as a string or identifier. This rule only applies to those non-terminal and terminal symbols that include more than one string word. It also only applies to non-terminal symbols that stem from terminal ones.

In DSL, there are two different sorts of numbers: floating point and decimal. The software is used to distinguish between decimal and float by "." (dot symbol). Similar to linguistic scripts, instructions are carried out sequentially from top to bottom.

4. Parse Tree

A parsing tree or concrete syntax tree is an ordered, rooted tree that describes the syntactic structure of a string according to a context-free grammar. Computational linguistics is the main field in which the term "parse tree" is used.

The phrase "syntax tree" is more prevalent in theoretical syntax. The matching parse tree for the following sample of code was created (Fig. 1):

```
START
Square ABCD
ABCD.setParameters (4)
ABCD.perimeter()
ABCD.area()
END
```

Figure 1. Parse Tree

Conclusion

This article's goal was to demonstrate how to utilize a DSL for geometric calculations. The DSL will permit to make the easier measurement and calculations of area, perimeter, volume or other geometric numbers. Lines of code may be converted into geometric computations using DSL. The product is intended for students, professors, and engineers who may not be highly experienced with programming, in contrast to other languages of a similar nature. Just having variables and methods makes language sound as simple as possible.

As the measurement and computation of geometric figures and bodies is a significant issue, the benefit of the task must be stated last. Students begin to despise math and geometry as a result. Hence, geometry will be more simpler to learn and easy thanks to the straightforward language created for this aim.

References

1. MARTIN FOWLER, *Domain-Specific Languages Guide*, 28 Aug 2019 [accessed on 15.02.2023] Available: DSL Guide (martinfowler.com)
2. Domain-Specific Languages, [accessed on 15.02.2023] Available: What are Domain-Specific Languages (DSL) | MPS by JetBrains

NAVIGAREA LA INTERSECȚIA DINTRE TEHNOLOGIILE EMERGENTE, ANALIZA AVANSATĂ A DATELOR ȘI SECURITATEA CIBERNETICĂ: STRATEGII PENTRU ERA DIGITALĂ

Alexandr DIMOGLO

Departamentul de Inginerie Software și Automatică, grupa SI-221M, Facultatea de Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Alexandr Dimoglo, alexandr.dimoglo@isa.utm.md

Coordonator științific: BULAI Rodica, profesor universitar, Automatică și Tehnologii Informaționale

Rezumat. *Acest document prezintă o imagine de ansamblu a provocărilor și strategiilor de navigare la intersecția dintre tehnologiile emergente, analiza avansată a datelor și securitatea cibernetică. Discutăm despre amenințările cibernetice din ce în ce mai sofisticate și despre proliferarea dispozitivelor conectate care creează vulnerabilități în era digitală. Subliniem deficitul de profesioniști calificați în domeniul securității cibernetice și necesitatea unei guvernanțe și reglementări eficiente pentru a asigura protecția datelor și a vieții private. Pentru a aborda aceste provocări, propunem o abordare cu mai multe fațete care implică o combinație de soluții tehnice, politici și programe de educație și formare. Analiza noastră se bazează pe o gamă variată de surse din literatura de specialitate, inclusiv Ghidul practicianului în domeniul securității cibernetice, Extinderea speranței de sănătate, Cartografierea valului de digitalizare a industriei și Guvernanța globală a științei și tehnologiei, printre altele. Constatările noastre sugerează că strategiile eficiente în materie de securitate cibernetică necesită o abordare holistică care să cuprindă atât aspecte tehnice, cât și non-tehnice, și că colaborarea dintre mediul academic, industrie și guvern este crucială pentru a aborda provocările complexe ale erei digitale.*

Cuvinte cheie*: amenințări cibernetice, protecția datelor, blockchain, internetul lucrurilor, analiză predictivă.*

Introducere

Adoptarea din ce în ce mai frecventă a tehnologiei a adus numeroase beneficii în viața noastră de zi cu zi, dar a introdus, de asemenea, noi provocări în materie de securitate care trebuie abordate de către practicienii în domeniul securității cibernetice. Pentru a rămâne în fața amenințărilor emergente și pentru a atenua riscurile digitale în evoluție, este esențial să înțelegem și să dezvoltăm strategii eficiente pentru a atenua potențialele implicații de securitate ale tehnologiilor emergente, cum ar fi IoT și edge computing. Deși aceste tehnologii oferă niveluri de confort fără precedent, ele introduc, de asemenea, noi vulnerabilități și suprafețe de atac care necesită măsuri de securitate pentru a se proteja împotriva lor. În plus, utilizarea tehnicilor avansate de analiză a datelor în contextul securității cibernetice oferă un mare potențial de detectare și prevenire a atacurilor cibernetice, dar necesită competențe și instrumente sofisticate pentru a fi implementate în mod efficient [1]. Prin urmare, este esențial să se investească în formarea și dotarea profesioniștilor din domeniul securității cibernetice cu competențele necesare pentru a valorifica aceste tehnici. În plus, implicațiile etice ale dezvoltării și implementării inteligenței artificiale (AI) trebuie să fie luate în considerare în contextul securității cibernetice. Pe măsură ce IA continuă să evolueze și să devină tot mai omniprezentă în viața noastră de zi cu zi, este esențial să se evalueze impactul potențial al acesteia asupra securității cibernetice și să se abordeze orice preocupări de ordin etic care pot apărea. Acest lucru necesită o abordare cuprinzătoare a securității cibernetice, care să integreze considerentele etice în proiectarea și implementarea acesteia [2]. În concluzie, prin explorarea intersecției dintre tehnologiile emergente, analiza avansată a datelor și securitatea cibernetică și prin încorporarea considerentelor etice în abordarea noastră, putem dezvolta strategii eficiente pentru a naviga în peisajul complex și în continuă schimbare al securității digitale.

Provocări existente

Domeniul securității cibernetice se confruntă cu numeroase provocări care trebuie abordate. Una dintre provocările majore este natura din ce în ce mai sofisticată a amenințărilor cibernetice, care continuă să evolueze odată cu tehnologia. Atacatorii cibernetici devin din ce în ce mai pricepuți și mai plini de resurse, ceea ce face mai dificilă prevenirea atacurilor și protecția împotriva încălcărilor de date [3]. O altă provocare este reprezentată de proliferarea dispozitivelor conectate, cum ar fi cele din Internetul obiectelor (IoT), care reprezintă vulnerabilități potențiale ce pot fi exploatate de atacatori. În plus, există un deficit semnificativ de profesioniști calificați în domeniul securității cibernetice, ceea ce face dificilă gestionarea eficientă a riscurilor de securitate cibernetică de către organizații. De asemenea, este nevoie de o guvernanță și de o reglementare eficace în domeniul securității cibernetice pentru a gestiona aceste riscuri în mod corespunzător, ceea ce necesită cooperarea între diverse părți interesate. În cele din urmă, cantitatea tot mai mare de date colectate și stocate de organizații a făcut mai importantă ca niciodată prioritizarea confidențialității și a protecției datelor [4]. Acest lucru necesită punerea în aplicare a unor măsuri solide de protecție a datelor și a unor politici și reglementări eficiente pentru a se asigura că datele cu caracter personal nu sunt utilizate în mod abuziv sau gestionate în mod necorespunzător. Aceste provocări trebuie abordate cu o abordare cuprinzătoare a securității cibernetice care să țină cont de peisajul tehnologic în continuă schimbare și de natura evolutivă a amenințărilor cibernetice [5].

Răspândirea dispozitivelor conectate

"Răspândirea dispozitivelor conectate" sau a dispozitivelor IoT reprezintă o provocare majoră în privința securității cibernetice, după cum se menționează în diverse lucrări de cercetare. Se preconizează că numărul dispozitivelor IoT va crește până la 75 de miliarde până în 2025, ceea ce deschide oportunități semnificative de atac pentru infractorii cibernetici [6]. Una dintre provocările legate de securizarea dispozitivelor IoT este reprezentată de resursele și puterea de calcul limitate ale acestora, precum și de localizarea lor în locații îndepărtate sau greu accesibile. Botnet-urile, care sunt rețele de dispozitive compromise, reprezintă o amenințare deosebită pentru mediile IoT [7]. Detectarea și combaterea botnet-urilor în mediile IoT este dificilă din cauza numărului mare de dispozitive și a rețelelor complexe. Abordarea problemei proliferării dispozitivelor conectate în mediile IoT va necesita inovare și colaborare constantă între industrie, mediul academic și guvern. Pentru a aborda aceste provocări este necesară o abordare coordonată care să ia în considerare aspectele tehnice, sociale și etice ale securității cibernetice.

Deficitul de profesioniști calificați în domeniul securității cibernetice

Lipsa de profesioniști calificați în domeniul securității cibernetice este o provocare semnificativă cu care se confruntă în prezent domeniul securității cibernetice. Potrivit "Cybersecurity Practitioner's Guide", există un deficit de peste 3 milioane de profesioniști la nivel global, ceea ce reprezintă o problemă gravă, deoarece cererea de profesioniști în domeniul securității cibernetice a depășit oferta [1]. Acest deficit are implicații semnificative asupra capacității organizațiilor de a se apăra împotriva atacurilor cibernetice, după cum se arată în lucrarea "Mapping the Wave of Industry Digitalization by Co-Word Analysis: An Exploration of Four Disruptive Industries" [3]. Complexitatea tot mai mare a amenințărilor la adresa securității cibernetice înseamnă că organizațiile au nevoie de o serie de competențe foarte specializate pentru a se apăra eficient împotriva atacurilor. Rezolvarea problemei lipsei de profesioniști calificați în domeniul securității cibernetice va necesita un efort concertat din partea industriei, a mediului academic și a guvernelor pentru a forma și dezvolta următoarea generație de profesioniști în domeniul securității cibernetice [8].

O provocare majoră cu care se confruntă domeniul, cu bariere precum lipsa de conștientizare în rândul studenților cu privire la carierele în domeniul securității cibernetice, resursele limitate pentru educația în domeniul securității cibernetice și natura rapid schimbătoare a domeniului. Rezolvarea problemei lipsei de personal va necesita o colaborare între industrie, mediul academic și guvern, inclusiv inițiative de sensibilizare, investiții în programe de educație și formare și eforturi de promovare a diversității și incluziunii. Procedând astfel, putem construi un viitor digital mai rezistent și mai sigur.

Necesitatea reglementării eficiente în domeniul securității cibernetice

Guvernanța și reglementarea eficientă în domeniul securității cibernetice sunt vitale pentru protejarea infrastructurilor critice, informațiilor personale și a datelor sensibile. Guvernanța și reglementarea inadecvate pot duce la utilizarea necorespunzătoare a tehnologiei, la încălcări ale confidențialității și la atacuri cibernetice. Importanța guvernanței și a reglementării în materie de securitate cibernetică este evidențiată în bibliografie, unde studiile subliniază necesitatea guvernanței pentru a aborda provocările complexe care decurg din progresele tehnologice. Documentele "Global Science and Technology Governance" și "Toward a Transformed and Unequal World" subliniază în special necesitatea unor cadre de guvernanță și de reglementare pentru a gestiona riscurile și oportunitățile prezentate de progresele tehnologice, cum ar fi inteligența artificială[9-10].

În plus, lucrarea "BLOCKCHAIN CNN DEEP LEARNING EXPERT SYSTEM FOR HEALTHCARE EMERGENCY" explorează utilizarea blockchain și a învățării profunde pentru a îmbunătăți asistența medicală de urgență[11]. Acesta subliniază importanța cadrelor de reglementare pentru asigurarea securității și confidențialității datelor din domeniul sănătății. Lucrarea "INTERNET OF THINGS EXPERT SYSTEM FOR SMART CITIES USING THE BLOCKCHAIN TECHNOLOGY" evidențiază, de asemenea, potențialul blockchain pentru securizarea dispozitivelor IoT în orașele inteligente[12].

Importanța confidențialității și a protecției datelor

Confidențialitatea și protecția datelor sunt componente esențiale ale securității cibernetice, mai ales pe măsură ce utilizarea tehnologiei și generarea de date continuă să crească. Protejarea informațiilor sensibile împotriva furtului, a utilizării abuzive sau a accesului neautorizat este o prioritate absolută. Bibliografia oferă informații suplimentare despre acest subiect, cu lucrări precum "Cybersecurity Practitioner's Guide", care oferă orientări pentru protejarea datelor sensibile, și "Extending Health Expectancy", care subliniază importanța protejării informațiilor personale de sănătate. Lucrările "Global Science and Technology Governance", "Internet of Things Expert System for Smart Cities", "Authentication and Secure Communications for Internet of Vehicles" și "A Comprehensive Solution for Ensuring Information Security" evidențiază, de asemenea, importanța confidențialității și a protecției datelor în peisajul digital actual[9-16]. Aceste lucrări subliniază necesitatea unor măsuri eficiente de protejare a informațiilor sensibile, deoarece nerespectarea acestor măsuri poate avea consecințe grave pentru persoane și organizații.

Concluzie

În concluzie, analiza literaturii de specialitate privind tehnologiile emergente, analiza avansată a datelor și securitatea cibernetică a evidențiat oportunitățile și provocările pe care le prezintă era digitală. Pentru a ține pasul cu aceste schimbări, strategiile de securitate cibernetică trebuie să evolueze, adoptând o abordare holistică care să țină cont de interconectarea sistemelor tehnologice și de diversele amenințări care pot apărea. Sunt necesare colaborarea și un angajament față de învățarea și adaptarea continuă. Pentru a asigura un viitor digital mai sigur și mai prosper, sunt necesare investiții în profesioniști calificați în domeniul securității cibernetice, o guvernanță și o reglementare solide în materie de securitate cibernetică, precum și măsuri eficiente de protecție a vieții private și a datelor. Provocările sunt numeroase și complexe, dar cercetarea și colaborarea continuă pot duce la strategii și tehnologii inovatoare care să atenueze riscurile asociate tehnologiilor emergente, permițându-ne să le valorificăm întregul potențial.

Referinţe

1. A. G. HESSAMI, *Cyber security practitioner's guide*. Hackensack: World Scientific, 2020.
2. G. TERRY SHARRER, *Extending Health Expectancy*, Mol. Front. J., т. 03, ed. 02, cc. 147–165, dec. 2019, doi: 10.1142/S252973251940011X.
3. L. BZHALAVA, S. S. HASSAN, J. KAIVO-OJA, B. KÖPING OLSSON, AND J. IMRAN, *Mapping the Wave of Industry Digitalization by Co-Word Analysis: An Exploration of Four*

Disruptive Industries, Int. J. Innovation Technol. Management, т. 19, ed. 02, с. 2250001, apr. 2022, doi: 10.1142/S0219877022500018.
4. S. FENG, *Toward a Transformed and Unequal World: The AI Revolution and the New International System*, China Q of Int' l Strategic Stud, т. 05, ed. 02, сс. 267–287, jan. 2019, doi: 10.1142/S2377740019500118.
5. R. C. AGUILERA, M. P. ORTIZ, A. A. BANDA, AND L. E. C. AGUILERA, *Blockchain CNN Deep Learning Expert System For Healthcare Emergency*, Fractals, т. 29, ed. 06, с. 2150227, sep. 2021, doi: 10.1142/S0218348X21502273.
6. S. HAIYONG, *Global Science and Technology Governance: Impetus, Challenges, and Prospects*, China Q of Int' l Strategic Stud, т. 07, ed. 01, сс. 61–78, jan. 2021, doi: 10.1142/S2377740020500244.
7. R. C. AGUILERA, M. P. ORTIZ, J. P. ORTIZ, AND A. A. BANDA, *Internet of Things Expert System For Smart Cities Using The Blockchain Technology*, Fractals, т. 29, ed. 01, с. 2150036, feb. 2021, doi: 10.1142/S0218348X21500365.
8. I. AFANASIEVA, N. GOLIAN, O. HNATENKO, Y. DANIIEL, AND K. ONYSHCHENKO, *Data Exchange Model In The Internet Of Things Concept*, Telecom Rad Eng, т. 78, ed. 10, сс. 869–878, 2019, doi: 10.1615/TelecomRadEng.v78.i10.30.
9. A. ARAL AND T. OVATMAN, *A Decentralized Replica Placement Algorithm for Edge Computing*, IEEE Trans. Netw. Serv. Manage., т. 15, ed. 2, сс. 516–529, jun. 2018, doi: 10.1109/TNSM.2017.2788945.
10. XI. LIU, *Data Information Security of Communication Network Based on Edge Computing Technology and BP Neural Network*, Telecom Rad Eng, т. 78, ed. 20, сс. 1837–1845, 2019, doi: 10.1615/TelecomRadEng.v78.i20.60.
11. I. E. VILINOV, A. V. VOLODIN, AND V. V. DERGACHEV, *A Comprehensive Solution for Ensuring Information Security of an Industrial Facility Infrastructure*, Telecom Rad Eng, т. 72, ed. 3, сс. 181–193, 2013, doi: 10.1615/TelecomRadEng.v72.i3.10.
12. J. I. NASER, H. A. G. ALSALMAN, AND A. J. KADHIM, *Authentication and Secure Communications for Internet of Vehicles (IoV)-Assisted Fog Computing*, Telecom Rad Eng, т. 78, ed. 18, сс. 1659–1670, 2019, doi: 10.1615/TelecomRadEng.v78.i18.40.
13. S. K. PRASAD, J. RACHNA, O. I. KHALAF, AND D.-N. LE, *Map Matching Algorithm: Real Time Location Tracking for Smart Security Application*, Telecom Rad Eng, т. 79, ed. 13, сс. 1189–1203, 2020, doi: 10.1615/TelecomRadEng.v79.i13.80.
14. A. I. TSOPA, *Estimation of Signal Encoding and Scrambling Impact on Information Transmission System Security*, Telecom Rad Eng, т. 74, ed. 1, сс. 51–60, 2015, doi: 10.1615/TelecomRadEng.v74.i1.50.
15. X. MENG AND W. LU, *Container-Based Fast Service Migration Method for Mobile Edge Computing*, J CIRCUIT SYST COMP, т. 30, ed. 15, с. 2250117, dec. 2021, doi: 10.1142/S0218126622501171.
16. A. SUKHOV, A. SIHVONEN, L. E. OLSSON, AND P. R. MAGNUSSON, *That Makes Sense to Me: Openness to Change and Sensemaking in Idea Screening*, Int. J. Innov. Mgt., т. 22, ed. 08, с. 1840009, dec. 2018, doi: 10.1142/S1363919618400091.

ANALIZA EMPIRICĂ A REȚELELOR SERIE-PARALEL VERSUS PARALEL-SERIE CU METODA MONTE CARLO PENTRU DISTRIBUȚIA EXPONENȚIALĂ A ELEMENTELOR REȚELELOR

Veronica ANDRIEVSCHI-BAGRIN*

Departamentul ISA, Facultatea CIM, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

*Autorul corespondent: Veronica Andrievschi-Bagrin, e-mail: veronica.bagrin@ati.utm.md

Coordonator științific: Alexei LEAHU, dr., conf.univ., UTM

Rezumat. *Lucrarea dată prezintă o analiză empirică a fiabilității a două tipuri de rețele, din punct de vedere arhitectural, Serie-Paralel și Paralel-Serie. Rețelele date au același număr prestabilit de elemente pentru fiecare exemplu studiat și numărul de subrețele este egal pentru ambele modele, unica diferență fiind arhitectura de conectare. Analiza analitică a fost efectuată anterior, pentru care s-au dedus formule, s-au construit grafice și s-a făcut comparația lor. Urmează aceste rezultate să fie validate prin metoda Monte Carlo, ceea ce este descris și analizat în continuare. Așa cum unitățile rețelelor au durata vieții exprimate prin variabile aliatoare independente identic distribuite, va fi aleasă distribuția exponențială pentru care se vor valida datele analitice și se vor scrie concluzii. Pentru simularea Monte Carlo a fost dezvoltat un produs program în limbajul de programare Python și librăriile acestuia. Toate datele rezultate simulării sunt prezentate în tabele și funcțiile vizualizate cu ajutorul graficelor. Produsul program realizat are capacitatea de a fi extins pentru diferite distribuții și desigur pentru diferite arhitecturi de rețele ce va da posibilitatea de a analiza o gamă largă de modele dinamice mai complexe.*

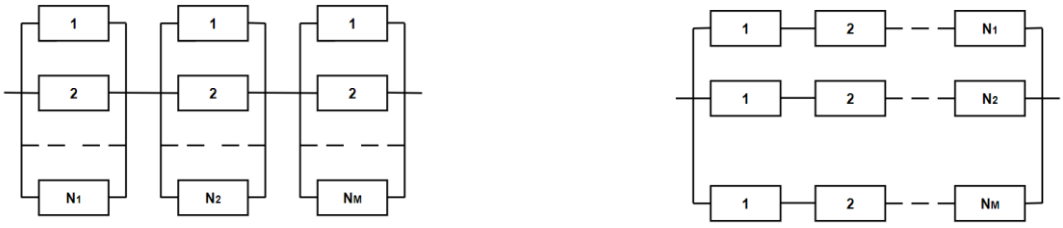
Cuvinte cheie*: distribuții, funcție de distribuție cumulativă, funcție de supraviețuire, .*

Introducere

Fiabilitatea rețelelor este un domenniu important pentru mulți proiectanți, cercetători și ingineri din domeniul tehnologiei informaționale și telecomunicațiilor. În acest sens, au fost deduse ecuații pentru calcularea fiabilității rețelelor de tip serie-paralel și paralel-serie, luând în considerare numărul constant de unități și sub-rețele și presupunând că durata de viață a acestora este distribuită aleatoriu și independent [1].

În mod particular, s-a demonstrat că, în cazul în care durata de viață a unităților este o variabilă aliatoare, independentă, identic distribuită, proprietatea unei rețele de a fi mai fiabilă decât alta nu depinde de funcția de distribuție a duratei de viață. Această concluzie a fost dedusă în lucrarea anterioară și este susținută de exemplele prezentate [2]. De asemenea, s-a arătat că, în cazul particular în care numărul de sub-rețele este mai mare decât 1 și sub-rețeaua cu cel mai mic număr de unități depășește acest număr, rețelele serie-paralel de tip A sunt întotdeauna mai fiabile decât rețelele paralel-serie de tip B.

Analiza fiabilității rețelelor, în lucrarea dată, ține de analiza arhitecturală a acestora. Va fi analizate două arhitecturi principale, care se găsesc în majoritatea cazurilor la proiectarea rețelelor și pot fi găsite sau reduse din rețele mai complexe. O rețea complexă în foarte multe cazuri, din viața reală, poate fi descompusă sau prezentată prin structuri mai simple precum structurile serie-paralel și/sau paralel-serie. În Fig. 1 sunt prezentate schematic arhitecturile de rețele studiate. Aici avem schemele A și B cu un număr M de subrețele în fiecare rețea și N_k elemente pentru fiecare subrețea, unde k = $1, 2, ..., M$



A - Serial-Parallel Network scheme. ***B*** - Parallel-Serial Network scheme.

Figure 1. Serial-parallel and parallel-serial schemes

În lucrările [2-4] s-au analizat arhitecturile A și B analitic și au fost deduse formule pentru funcțiile de distribuție cumulative pentru ambele arhitecturi. Aceste formule au servit drept bază pentru deducerea funcțiilor de supraviețuire a acestora. În continuare, în lucrarea curentă aceste formule au fost analizate analitic pentru diferite distribuții ale elementelor rețelei și anule distribuția uniformă și distribuția exponențială, și s-au comparat pentru arhitecturile A și B.

1. Analiza analitică și empirică a fiabilității rețelelor de tip Serie-Paralel și Paralel-Serie pentru distribuția uniformă a elementelor acestora

Luând la baza cercetării lucrările preventive la capitolul dat [2-5] avem drept punct de pornire formulele deduse analitic pentru modele matematice mai complicate, precum ar fi rețelele de tip A și B, dar totuși ne ținem la moment de cazul când numerele N_k, $k = 1, 2, ..., M$ și M sunt numere naturale, aceleași fiind pentru ambele modele. A fost luat drept timp de viață a tuturor elementelor componente rețelelor variabile aleatoare independente, cu valori non-negative (v.a.i.) X_{kj} cu funcția de distribuție cumulativă $F_{kj}(x)$ pentru unitatea de pe poziția j a subrețelei k, $j = \overline{1, N_k}$, $k = \overline{1, M}$. Respectiv durata vieții pentru fiecare subrețea pentru ambele modele va fi de asemenea o variabilă aleatoare independentă deoarece reprezintă un minimum sau un maximum din valorile elementelor subrețelei $Y_k = min\ (X_{kl}, X_{k2}, ..., X_{kNk})$, în cazul rețelei paralel-serie, sau $Y_k = max\ (X_{kl}, X_{k2}, ..., X_{kNk})$, în cazul rețelei serie-paralel, $k = \overline{1, M}$. Iar extinderea pentru întreaga rețea obținem $U = min\ (Y_l, Y_2, ..., Y_M)$, respective $V = max\ (Y_l, Y_2, ..., Y_M)$. Analogic au fost deduse formulele pentru funcțiile de distribuție cumulative prezentate în lucrarea [2] redate de formulele (1) și (2) .

$$F_U(x) = P(U \le x) = 1 - \prod_{k=1}^{M}\left[1 - \prod_{j=1}^{N_k} F_{kj}(x)\right] \quad (1)$$

$$F_V(x) = P(V \le x) = \prod_{k=1}^{M}\left[1 - \prod_{j=1}^{N_k}(1 - F_{kj}(x))\right] \quad (2)$$

În baza acestor formule s-au dedus ecuațiile pentru funcțiile de fiabilitate a rețelelor de tip *A* și *B*.

$$S_{S-P}(X) = P(U > x) = 1 - F_U(x) = \prod_{k=1}^{M}\left[1 - \prod_{j=1}^{N_k} F_{kj}(x)\right] \quad (3)$$

$$S_{P-S}(X) = P(V > x) = 1 - F_V(x) = 1 - \prod_{k=1}^{M}\left[1 - \prod_{j=1}^{N_k}(1 - F_{kj}(x))\right] \quad (4)$$

Pentru a determina care tip de rețea este mai fiabilă este suficient să comparăm funcțiile de fiabilitate $S_{U_M}(x)$, $S_{V_M}(x)$ pentru o distribuție concretă a timpului de viață $F_{kj}(x)$, $j = \overline{1, N_k}$, $k = \overline{1, M}$.

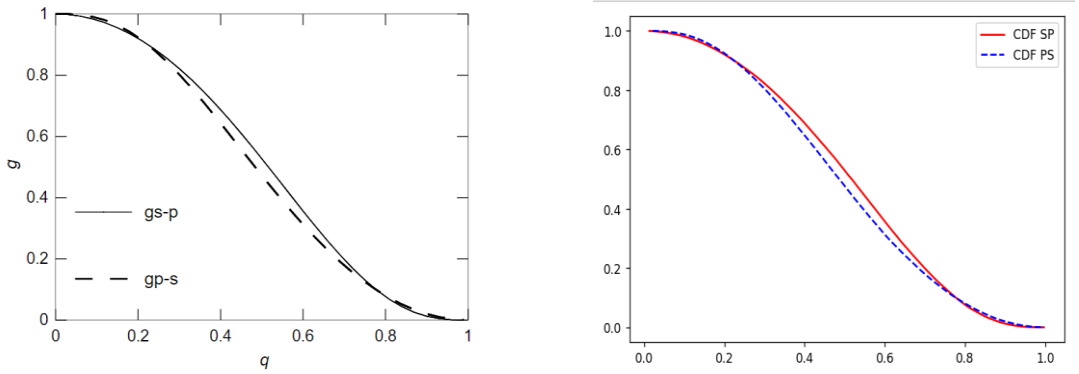
Dacă analizăm cazul în care toate elementele rețelei sunt variabile aleatoare identic distribuite $F_{kj}(x) = F(x)$, $j = \overline{1, N_k}$, $k = \overline{1, M}$ atunci vom avea respectivele formule de calcul a fiabilității redate prin formulele (5) și (6).

$$S_{S-P}(x) = P(U > x) = 1 - F_U(x) = \prod_{k=1}^{M}[1 - (F(x))^{N_k}] \quad (5)$$

$$S_{P-S}(x) = P(V > x) = 1 - F_V(x) = 1 - \prod_{k=1}^{M}[1 - (1 - F(x))^{N_k}] \quad (6)$$

Să analizăm câteva exemple.

Exemplul 1. Vom considera rețelele de tip serie-paralel și paralel-serie $M = 3$ subrețele fiecare și $N_1 = 4$, $N_2 = 2$, $N_3 = 2$ numărul de elemente în fiecare subrețea respective. Pentru distribuția uniformă a v.a.i.i.d a elementelor rețelelor vom compara din lucrările anterioare [2-4] graficul funcției de distribuție obținut analitic și graficul funcției de distribuție obținut empiric reprezentat în Fig. 2



Analiza analitică Analiza empirică

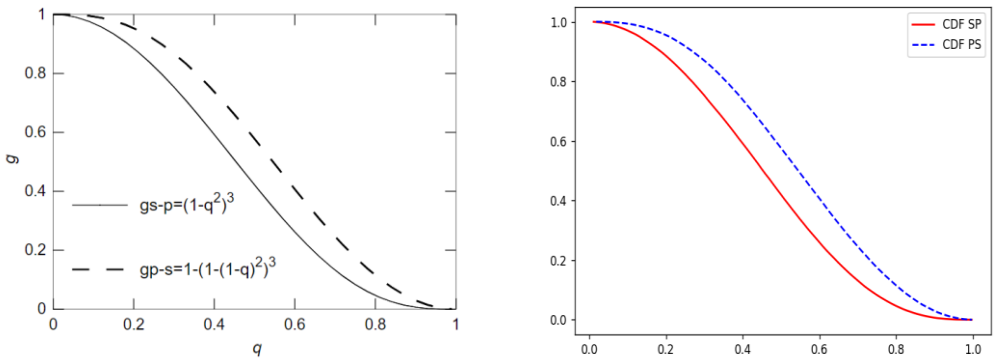
Figura 2. Analiza analitică și empirică pentru distribuția uniformă

După cum se observă rezultatele obținute prin metode Monte Carlo, cel empiric, este întrutotul identic celui obținut pe cale analitică, ceea ce validează corectitudinea formulelor deduse și comportamentul acestora.

Vom analiza următorul exemplu, de data aceasta cu numărul de elemente în fiecare subrețea mai mic decât numărul de subrețele.

Exemplul 2. Vom considera rețelele de tip serie-paralel și paralel-serie $M = 3$ subrețele fiecare și $N_1 = 2$, $N_2 = 2$, $N_3 = 2$ numărul de elemente în fiecare subrețea respective. Pentru distribuția uniformă a v.a.i.i.d a elementelor rețelelor vom compara din lucrările anterioare [2-4] graficul funcției de distribuție obținut analitic și graficul funcției de distribuție obținut empiric reprezentat în Fig. 3

De asemenea obținem rezultatul dorit de validare a formulelor obținute pe cale analitică și obținem un succes în lansarea produsului deprogram de simulare prin metode Monte Carlo a fiabilității rețelelor.



Analiza analitică Analiza empirică

Figura 3. Analiza analitică și empirică pentru distribuția uniformă

2. Analiza analitică a funcțiilor de distribuție deduse pentru fiabilitatea rețelelor de tip Serie-Paralel și Paralel-Serie pentru distribuția exponențială,

În continuare se va considera analiza comparativă a rețelelor la fiabilitate pentru cazul distribuției exponențiale pentru toate elementele rețelelor. Ca și în cazul uniform [2], am dedus formula funcției de distribuție pentru fiecare exemplu individual. Cu instrumentul Wolfram Alpha s-

au calculat funcțiile densității distribuției. Vom arăta că aceasta îndeplinește condițiile funcțiilor de densitate. Vom calcula media și varianța modelului dat, pentru fiecare exemplu. După aceea, vom compara aceste date cu rezultatele empirice obținute. În continuare analiza empirică se face în mediul de programare Python, folosind bibliotecile relevante pentru simularea metodelor Monte Carlo, iar eșantionul analizat va fi ales în număr de n_0=100000 de încercări.

Să analizăm primul exemplu.

Exemplul 1. Vom considera rețelele de tip serie-paralel și paralel-serie $M=3$ subrețele fiecare și $N_1 = 4$, $N_2 = 2$, $N_3 = 2$ numărul de elemente în fiecare subrețea respective. Pentru distribuția exponențială a v.a.i.i.d a elementelor rețelelor vom analiza funcțiile de distribuție obținute analitic pentru acest model.

Pentru respectivul model luând drept distribuție pentru fiecare v.a.i.i.d ale rețelelor distribuția exponențială cu parametrul **λ = 1,** obținem formulele pentru funcția de distribuție a rețelei, funcția densității de distribuție, media și varianța acestora cu ajutorul instrumentului recunoscut Wolfram Alpha. Rezultatele fiind prezentate în Fig. 4 pentru tipul de rețea Serie-Paralel și în Fig. 5 pentru tipul de rețea Paralel-Serie.

$$\frac{d}{dx}\left(1-\left(1-\left(1-e^{-x}\right)^4\right)\left(1-\left(1-e^{-x}\right)^2\right)^2\right)=$$
$$4e^{-8x}\left(1-2e^x\right)^2\left(e^x-1\right)\left(-4e^x+3e^{2x}+2\right)$$

$$\int_0^\infty 4e^{-8x}\left(1-2e^x\right)^2\left(e^x-1\right)\left(-4e^x+3e^{2x}+2\right)dx=1$$

$$\int_0^\infty x\left(4e^{-8x}\left(1-2e^x\right)^2\left(e^x-1\right)\left(-4e^x+3e^{2x}+2\right)\right)dx=\frac{229}{280}\approx 0.81786$$

$$\int_0^\infty (x-0.81786)^2\left(4e^{-8x}\left(1-2e^x\right)^2\left(e^x-1\right)\left(-4e^x+3e^{2x}+2\right)\right)dx=0.257501$$

Figura 4. Rezultatele obținute pentru modelul Serie-Paralel

$$\frac{d}{dx}\left(\left(1-e^{-4x}\right)\left(1-e^{-2x}\right)^2\right)=4e^{-8x}\left(-3e^{2x}+e^{6x}+2\right)$$

$$\int_0^\infty 4e^{-8x}\left(-3e^{2x}+e^{6x}+2\right)dx=1$$

$$\int_0^\infty x\left(4e^{-8x}\left(-3e^{2x}+e^{6x}+2\right)\right)dx=\frac{19}{24}\approx 0.79167$$

$$\int_0^\infty (x-0.79167)^2\left(4e^{-8x}\left(-3e^{2x}+e^{6x}+2\right)\right)dx=0.293403$$

Figura 5. Rezultatele obținute pentru modelul Paralel-Serie

Vom analiza următorul exemplu 2, de data aceasta cu numărul de elemente în fiecare subrețea mai mic decât numărul de subrețele.

Exemplul 2. Vom considera rețelele de tip serie-paralel și paralel-serie $M=3$ subrețele fiecare și $N_1 = 2$, $N_2 = 2$, $N_3 = 2$ numărul de elemente în fiecare subrețea respective. Pentru distribuția exponențială a v.a.i.i.d a elementelor rețelelor vom analiza funcțiile de distribuție obținute analitic pentru acest model.

La fel ca și în exemplu 1, modelul curent va fi analizat cu ajutorul instrumentului recunoscut Wolfram Alpha, luând drept distribuție pentru fiecare v.a.i.i.d ale rețelelor distribuția exponențială

cu parametrul **λ = 1.** Obținem formulele pentru funcția de distribuție a rețelei, funcția densității de distribuție, media și varianța acestora. Rezultatele fiind prezentate în Fig. 6 pentru tipul de rețea Serie-Paralel și în Fig. 7 pentru tipul de rețea Paralel-Serie.

$$\frac{d}{dx}\left(1-\left(1-\left(1-e^{-4x}\right)^4\right)\left(1-\left(1-e^{-4x}\right)^2\right)^2\right) = 16e^{-32x}\left(1-2e^{4x}\right)^2\left(e^{4x}-1\right)\left(-4e^{4x}+3e^{8x}+2\right)$$

$$\int_0^\infty 16e^{-32x}\left(1-2e^{4x}\right)^2\left(e^{4x}-1\right)\left(-4e^{4x}+3e^{8x}+2\right)dx = 1$$

$$\int_0^\infty x\left(16e^{-32x}\left(1-2e^{4x}\right)^2\left(e^{4x}-1\right)\left(-4e^{4x}+3e^{8x}+2\right)\right)dx = \frac{229}{1120} \approx 0.20446$$

$$\int_0^\infty (x-0.20446)^2\left(16e^{-32x}\left(1-2e^{4x}\right)^2\left(e^{4x}-1\right)\left(-4e^{4x}+3e^{8x}+2\right)\right)dx = 0.0160938$$

Figura 6. Rezultatele obținute pentru modelul Serie-Paralel

$$\frac{d}{dx}\left(\left(1-e^{-4\times 4x}\right)\left(1-e^{-2\times 4x}\right)^2\right) = 16e^{-32x}\left(e^{8x}-1\right)^2\left(e^{8x}+2\right)$$

$$\int_0^\infty 16e^{-32x}\left(e^{8x}-1\right)^2\left(e^{8x}+2\right)dx = 1$$

$$\int_0^\infty x\left(16e^{-32x}\left(e^{8x}-1\right)^2\left(e^{8x}+2\right)\right)dx = \frac{19}{96} \approx 0.19792$$

$$\int_0^\infty (x-0.19792)^2\left(16e^{-32x}\left(e^{8x}-1\right)^2\left(e^{8x}+2\right)\right)dx = 0.0183377$$

Figura 7. Rezultatele obținute pentru modelul Paralel-Serie

Concluzii

În concluzie, putem spune că analizând media duratei vieții rețelelor nu putem trage o concluzie cu privire la fiabilitatea unei rețele de un tip în comparație cu alta deoarece avem situații care confirm acest fapt. Funcția de fiabilitate în schimb conține mai multe informații decât doar media și poate ajuta la identificarea situațiilor în care un sistem este mai fiabil sau nu decât altul. De asemenea, fiabilitatea rețelelor nu se schimbă în funcție de distribuția duratei vieții a elementelor rețelei, ci numai arhitectura rețelelor analizate în raport cu numărul de elemente în subrețele și numărul de subrețele este importantă. Analiza noastră empirică a relevat de asemenea că proprietatea unei rețele de a fi mai fiabilă decât cealaltă nu depinde de funcția de distribuție a duratei de viață a elementelor. În final, concluziile noastre bazate pe modele dinamice generalizează concluziile altor autori bazate pe modele statice.

În concluzie, lucrarea noastră aduce cercetări în domeniul fiabilității rețelelor și ar putea fi utilizată în dezvoltarea de rețele mai fiabile și robuste, având drept scop asigurarea a o mai bună performanță și fiabilitate a rețelelor.

Mulțumiri

Aduc sincere mulțumiri dlui Dr., prof. univ. Alexei LEAHU, conducătorul meu științific, pentru suportul acordat și timpul dedicat lucrării. De asemenea aș vrea să menționez că această lucrare a fost susținută de proiectul 20.80009.5007.26 „Modele, algoritmi și tehnologii de conducere, optimizare și securizare a sistemelor Ciber-Fizice”.

Referințe

1. KAPUR, K.C., LAMBERSON, L.R. *Reliability in engineering design.* India: Wiley, 2009; pp. 75-94.

2. ANDRIEVSCHI-BAGRIN, V., LEAHU, A. Reliability of Serial-Paralel networks vs Reliability of Paralel-Serial networks with constant numbers of sub-networks and units. In: *The Journal of Engineering Science*, 2022, XXIX (4), UTM, Chișinău. https://jes.utm.md/category/vol-xxix-4-2022
3. LEAHU, A., ANDRIEVSCHI-BAGRIN, V. Lifetime Distributions and their Approximation in Reliability of Serial/Parallel Networks. In; *Analele Universitatii "Ovidius" Constanta - Seria Matematica*, 22.09.2020, 28 (2), pp. 161-172. https://sciendo.com/article/10.2478/auom-2020-0025
4. LEAHU, A., ANDRIEVSCHI-BAGRIN, V., CIORBĂ, D., FIODOROV. I. Min(max-psd) and max(min-psd) as lifetime distributions in network's reliability. In: *Analele științifice ale Universitatii "Ovidius" Constanta - Seria Matematic.* 08.10.2022, V30, pp. 173-184. https://sciendo.com/article/10.2478/auom-2022-0039
5. BARLOW, R. E., PROSCHAN, F. *Mathematical Theory of Reliability*. New York, USA: Wiley, 1965; pp. 5-9.

MODELING AND SIMULATION OF A K-MEANS CLUSTERING ALGORITHM IN MATLAB AND PYTHON

Ștefana DUȚĂ[1], Laura-Elena DOROBANȚU[2], Dan-Gabriel RADU[3]

[1]*Department of Information and Computer Systems Engineering, 411 IISC, Faculty of Electronics, Telecommunications and Information Technology, Politehnica University of Bucharest, Bucharest, Romania*
[2]*Department of Measurements, Electrical Devices and Static Converters, SIIM1, Faculty of Electrical Engineering, Politehnica University of Bucharest, Bucharest, Romania*
[3]*Department of Electrical Engineering, IEIA2, Faculty of Electrical Engineering, Politehnica University of Bucharest, Bucharest, Romania*

*Corresponding author: Ștefana DUȚĂ, stefana.duta@stud.fim.upb.ro

Scientific coordinator: Felix Constantin ADOCHIEI, PhD, DMEDSC

Abstract. *This paper presents an analysis of the K-means algorithm, a popular unsupervised learning technique used for clustering data based on similar features. For this, the implementation and performance of the K-means algorithm were compared in the Python and MATLAB programming languages. In addition, modeling and simulation of the algorithm were performed to compare the computation time and clustering quality in the two languages. The experimental results demonstrate that the K-means algorithm is a powerful tool for data clustering and can be used effectively in various applications. Although both Python and MATLAB are languages capable of efficiently implementing the K-means algorithm, MATLAB resulted in a shorter computation time.*

Keywords*: K-means, MATLAB, Python, modeling.*

Introduction

Supervised and unsupervised learning are the two main categories of machine learning. Supervised learning algorithms aim to learn a function that maps an input to its corresponding output using available labeled data. Obtaining labeled data is often a challenging task in supervised learning [1]. In situations where dataset labels are unavailable, unsupervised learning is typically used. Clustering algorithms are used in this framework to group data points that have similar characteristics, exploiting the underlying structure of the data distribution [2-3]. Unlike supervised learning, where the algorithm has access to labeled data, unsupervised learning algorithms do not have prior knowledge of the actual labels of the dataset and instead draw conclusions from the data itself [3-4]. The k-means algorithm is a widely used and popular clustering technique due to its ease of implementation and low computational complexity. It is often applied to image compression, where it can reduce the number of colors used to represent an image while maintaining its visual similarity [5-8].

Fig. 1 depicts a typical K-means clustering procedure, where the input dataset is initially represented in a 2D space, as shown in Fig. 1a. Following the K-means clustering process, which involves setting the number of centroids to K=3, the resulting clustered dataset is presented in Fig. 1b. The K-means algorithm requires users to set the number of clusters and initialize the centroids randomly, but this can lead to poor performance due to the dependence on the initial selection of clusters. This issue is more pronounced for large datasets, and determining the optimal number of clusters is a complex task [9]. Another significant limitation of the algorithm is the use of Euclidean distance as a similarity measure, which restricts its ability to detect clusters of non-standard shapes and makes detecting overlapping clusters challenging.

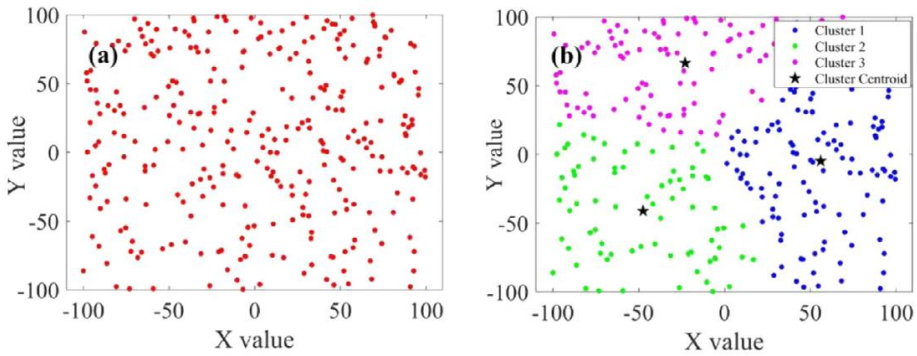


Fig. 1. The K-means clustering algorithm is depicted in (a) with randomly distributed datasets, followed by (b) the identification of the closest centroid for each of the three classes [8]

The K-means clustering algorithm is a heuristic iterative process that can be broken down into three main steps, as described by [7]:

a) Initialization involves setting the central points with a given number of K clusters.

b) The algorithm then assigns all data points to one of the K clusters based on the current centroids.

c) The centroids are then updated based on the newly formed clusters.

By repeating steps b) and c) over several iterations, the K-means algorithm eventually converges and can effectively detect ball-shaped or spherical clusters. However, this is limited using the Euclidean metric as a distance measure [8, 10].

Fig. 1.2 presents the steps of a K-means algorithm.

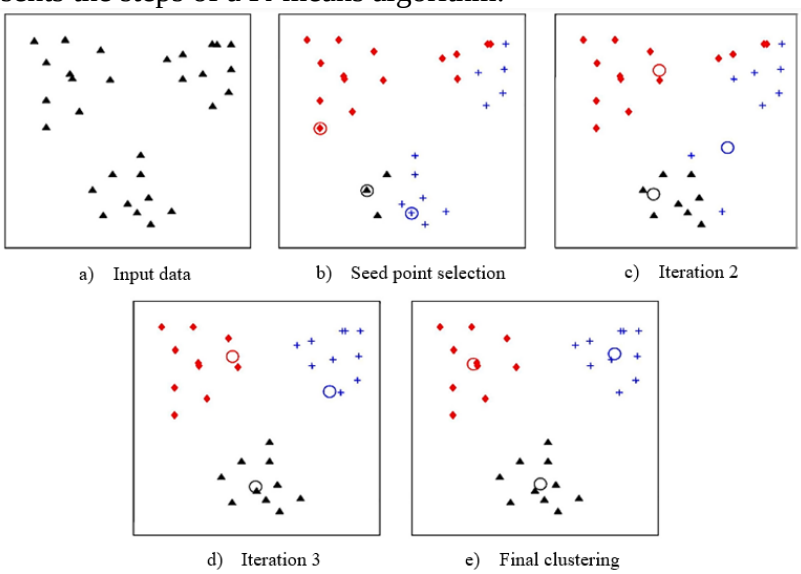


Fig. 2. The K-means algorithm is illustrated in the following steps: (a) A two-dimensional input dataset with three clusters. (b) Selection of three seed points as cluster centers, followed by the initial assignment of the data points to these clusters. (c) & (d) Intermediate iterations involving the updating of cluster labels and their centers. (e) The final clustering obtained by the K-means algorithm at convergence [11].

In the K-means algorithm, the goal is to group n multidimensional data points in a given dataset *X* into *K* categories. To achieve this, the Euclidean distance is utilized as the similarity index, and the algorithm seeks to minimize the sum of squares of different types, referred to as the "minimization" step. [10, 12]. This step is carried out according to Eq (1) [12]:

$$d = \sum_{k=1}^{k} \sum_{i=1}^{n} ||x_i - u_k||^2 \quad (1)$$

The variables used in the K-means algorithm are defined as follows: k represents the K cluster centers, u_k represents the k-th center, and xi represents the i-th point in the given dataset.

The solution of center u_k is shown in Eq (2):

$$\frac{\partial}{\partial u_k} = \frac{\partial}{\partial u_k} \sum_{k=1}^{k} \sum_{i=1}^{n} (\mathrm{x_i} - u_k)^2$$
$$= \sum_{i=1}^{n} 2(x_i - u_k) \tag{2}$$

Modeling and Simulation

This paper compares the implementation and performance of the K-means algorithm in Python and MATLAB d.

A. Data generating:

Random data are generated from three normal distributions with known parameters. First, each class used several data points to study the algorithm's computation time evolution. Next, the data is divided into three classes and randomly mixed.

B. The starting assumption

The three starting points for the algorithm are set.

C. Setting the algorithm parameters.

An expected precision of 0.001 and a substantial error starting value of 10000 are chosen.

D. Expectation

The expectation is the first function, which associates the closest point to each center with its cluster and handles updating the data labels according to the Euclidean distance.

E. Maximization

This function updates the center of the clusters according to the update of the labels in the previous step by averaging the coordinates of the points in each class on each axis.

F. Error Update

At each iteration, the desired and the calculated error are compared. If the calculated one is higher, it is updated with a new value calculated according to equation (1) presented in the previous chapter.

This study uses an algorithm based on the K-means implementation developed by Reza Ahmadzadeh, available in the GitHub repository at https://github.com/rezaahmadzadeh/K-means [13. The presented algorithm was trained using a Lenovo device with an ADM Ryzen 5 3500U processor and 8 GB RAM, using the Windows 11 64-bit operating system. The training process did not involve the GPU.

Results

CASE I – 50 values for each class

Actual values of the centroids of the three classes:

- (4, 0); (0, 4); (-3, 3).

Initial guess values:

- (0, 0); (2, 2); (-1, -1).

Tab. 1 shows the running times for the two programming languages.

Table 1

Experimental Results with 50 values/class – Running Time

Running Time (s)	**MATLAB (4 iterations)**	**Python (4 iterations)**	**Python (2 iterations)**
	0.0025	0.0128	0.0149

With the help of Python, a result can be reached with only two iterations, whereas in MATLAB, a minimum of 4 iterations are required to achieve a minimum error value (0). The result obtained after the same number of iterations for both programming languages, named after four iterations, is also observed.

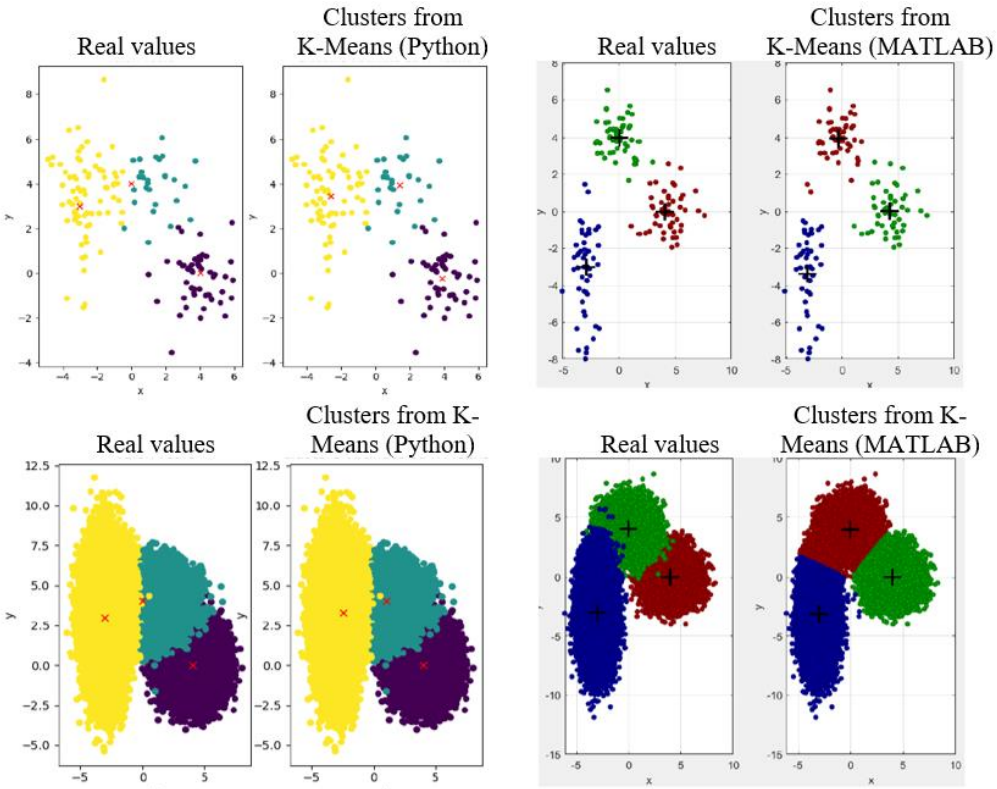


Fig. 3. Algorithm results: CASE I – up and CASE II- down. Python – left and MATLAB - right. Dividing the data into the three classes and determining the centroids

CASE II – 50000 values for each class

Actual values of the centroids of the three classes:

- (4, 0); (0, 4); (-3, 3).

Initial guess values:

- (0, 0); (2, 2); (-1, -1).

Tab. 2 shows the running times for case II. With Python, a result can be reached with six iterations, whereas in MATLAB, at least nine iterations are needed to get a minimum error value.

Table 2

Experimental Results with 50000 values/class – Running Time

Running Time (s)	**MATLAB (9 iterations)**	**Python (6 iterations)**
	0.1940	33.7897

Experimental results show that MATLAB is faster than Python when implementing the K-means algorithm. For example, the computation time in MATLAB was five times faster than in Python when using a set of 50 values. Furthermore, it is found that if the number of data is increased to 50000, MATLAB is 174 times faster than Python.

It was also found that the quality of the clustering results was similar in both languages, as can be seen in Fig. 3.

Conclusions

In conclusion, this paper compared the implementation and performance of the K-means algorithm in Python and MATLAB programming languages. By evaluating the algorithm on a generated data set, it was found that MATLAB is generally faster than Python in terms of computation time. This is probably due to the optimized built-in functions and toolboxes in MATLAB, designed explicitly for numerical computations and data analysis tasks.

However, it is worth noting that Python offers a vast ecosystem of libraries and tools that can be used to implement the K-means algorithm and is more flexible in integrating with other systems.

In summary, both Python and MATLAB can be used to implement the K-means algorithm and give accurate clustering results. However, MATLAB may be a better choice for tasks requiring faster computation time, while for jobs requiring more flexibility and integration with other tools, Python may be a better choice.

Bibliography

1. M. MOHRI, A. ROSTAMIZADEH, and A. TALWALKAR, "Foundations of machine learning," p. 486, 2012, Accessed: Jan. 18, 2023. [Online].
2. C. BISHOP, "Neural networks for pattern recognition," 1995.
3. A. K. JAIN, M. N. MURTY, and P. J. FLYNN, "Data clustering," *ACM Computing Surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, Sep. 1999, doi: 10.1145/331499.331504.
4. M. AHMED, R. SERAJ, and S. M. S. ISLAM, "The k-means Algorithm: A Comprehensive Survey and Performance Evaluation," *Electronics 2020, Vol. 9, Page 1295*, vol. 9, no. 8, p. 1295, Aug. 2020, doi: 10.3390/ELECTRONICS9081295.
5. X. WAN, "Application of K-means Algorithm in Image Compression," *IOP Conf Ser Mater Sci Eng*, vol. 563, no. 5, Aug. 2019, doi: 10.1088/1757-899X/563/5/052042.
6. Y. ZHAO and X. ZHOU, "K-means Clustering Algorithm and Its Improvement Research," *J Phys Conf Ser*, vol. 1873, no. 1, Apr. 2021, doi: 10.1088/1742-6596/1873/1/012074.
7. X. G. LI, M. F. YAO, and W. T. HUANG, "Speech recognition based on k-means clustering and neural network ensembles," *Proceedings - 2011 7th International Conference on Natural Computation, ICNC 2011*, vol. 2, pp. 614–617, 2011, doi: 10.1109/ICNC.2011.6022159.
8. H. LE MINH, T. SANG-TO, M. ABDEL WAHAB, and T. CUONG-LE, "A new metaheuristic optimization based on K-means clustering algorithm and its application to structural damage identification," *Knowl Based Syst*, vol. 251, p. 109189, Sep. 2022, doi: 10.1016/J.KNOSYS.2022.109189.
9. A. M. IKOTUN, A. E. EZUGWU, L. ABUALIGAH, B. ABUHAIJA, and J. HEMING, "K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data," *Inf Sci (N Y)*, vol. 622, pp. 178–210, Apr. 2023, doi: 10.1016/J.INS.2022.11.139.
10. A. SINGH, A. YADAV, and A. RANA, "K-means with Three different Distance Metrics," *Int J Comput Appl*, vol. 67, no. 10, pp. 13–17, Apr. 2013, doi: 10.5120/11430-6785.
11. A. K. JAIN, "Data clustering: 50 years beyond K-means," *Pattern Recognit Lett*, vol. 31, no. 8, pp. 651–666, Jun. 2010, doi: 10.1016/J.PATREC.2009.09.011.
12. C. YUAN and H. YANG, "Research on K-Value Selection Method of K-Means Clustering Algorithm," *J 2019, Vol. 2, Pages 226-235*, vol. 2, no. 2, pp. 226–235, Jun. 2019, doi: 10.3390/J2020016.
13. "K-means/k-means.py at master · rezaahmadzadeh/K-means · GitHub." https://github.com/rezaahmadzadeh/K-means/blob/master/Python/k-means.py (accessed Jan. 19, 2023).

DESIGN OF DOMAIN SPECIFIC LANGUAGE FOR ASTROLOGICAL CHARTS GENERATION - ALAKIRQL

Alexandru ANDRIEȘ[1], Georgeana GLOBA[1], Arteom KALAMAGHIN[1*], Dionisie SPATARU[1]

[1]*Departamentul Ingineria Software și Automatică; Facultatea Calculatoare, Informatică și Microelectronică; Universitatea Tehnică a Moldovei; o. Chișinău; Republica Moldova*

*Corresponding author: Arteom Kalamaghin, arteom.kalamaghin@isa.utm.md

Scientific coordinators: Mariana CATRUC, superior lecturer, UTM; **Irina COJUHARI**, assoc. prof. , UTM

Abstract. *This article represents an analysis of the definition of a* Domain-Specific Language *for work in the field of astrology - AlakirQL. The article's goal is to describe the functionality, specifications, semantic and grammatical features of the AlakirQL, as well as the lexer implementation progress, by analyzing both technical and non-technical subjects.*

Keywords: *Domain-Specific Language, Astrology, formal language, grammar, semantics.*

Introduction

DSLs (Domain-Specific Languages) are becoming increasingly popular tools for solving problems in areas where advanced scripting is not required. Programming provides a wide range of possibilities to create features, solve everyday problems, or routines. Since DSLs are widespread in this day and age, the ability to create automated alternatives for otherwise labored assignments has become effortless in any area of work [1].

In some cases, constructing a domain-specific language from scratch could be a better solution than using any other existent substitutes, since it might allow for a clearer expression of both algorithmic questions and answers in cases where they are regularly encountered. In addition, the improvement of tools has made this task even easier than before.

Domain description

Astrology represents the multitude of spiritual practices that claim to have the ability to discover details about people's lives and some global events based on the positions of different extraterrestrial objects [2].

Nowadays, the most popular application is composition of birth charts. Astrologers use the birth chart to make predictions about future events and to provide guidance for personal growth and development. The composition of the Birth Chart and the provision of tabular data related to this notion will be the primary problem with which the *AlakirQL* will be concerned. Some web resources provide solutions that implement this service, however, we believe that our approach has two big advantages over this form of service delivery. First of all, local use: a web-application requires personal data and contact information to provide results; when used locally your data does not leave the device. Automation: the implementation of control and data structures for repetitive flow and similar data storing will allow users to process large amounts of information more easily, speeding up routine work. It is also worth mentioning another approach to data formatting: special output and reading modules will allow users to save processed data to files in order to be able to return to them in the future.

Grammar

What follows is a definition of the grammar in a form that will help us easily construct a parse tree for a sample piece of code – $G = (S, P, V_N, V_T)$, where: S – the start token; P – a collection of production rules; V_N – lists non-terminals; V_T – a list of terminal symbols.

In Table I are listed meta-notations, understanding of which is essential while reading the grammar specified below.

Table 1.

Meta-notation

Notation	Denotation
<foo>	for non-terminals
foo	for terminal symbols (a token or a part of a token)
[x]	indicates zero or more occurrences of x (is optional; note that brackets in quotes are terminals)
x*	indicates zero or more occurrences of x
x^+,	separated by comma list of one or more x's
|	separates alternatives

S = { <program> }; V_T = { 0..9, A..Z, a..z, WHILE, DECLARE, CONST, (datatype names), (built-in function names), (operators) , _ ,) , (, [,] , { , } , deg, min, sec, year, month, day, hour, minute, second, tables, " , ; }; V_N = { ⟨block⟩, ⟨directive⟩, ⟨var_decl⟩, ⟨declarator⟩, ⟨type⟩, ⟨expr⟩, ⟨statement⟩, ⟨location⟩, ⟨method_call⟩, ⟨method_name⟩, ⟨bin_op⟩, ⟨literal⟩, ⟨id⟩, ⟨alpha_num⟩, ⟨alpha⟩, ⟨digit⟩, ⟨arith_op⟩, ⟨rel_op⟩, ⟨eq_op⟩, ⟨int_literal⟩, ⟨decimal_literal⟩, ⟨string_literal⟩, ⟨angle_literal⟩, ⟨dt_literal⟩ } ; P =

{ ⟨program⟩	→ ⟨directive⟩*
⟨block⟩	→ '{' ⟨directive⟩* '}'
⟨directive⟩	→ ⟨var_decl⟩ | ⟨statement⟩ **;**
⟨var_decl⟩	→ ⟨declarator⟩ ⟨id⟩+, [= ⟨expr⟩+,] | [= '[' ⟨expr⟩+, ']']
⟨declarator⟩	→ **CONST** | **DECLARE**
⟨type⟩	→ **INT** | **STRING** | **ANGLE** | **DATETIME** | **CHART**
⟨expr⟩	→ ⟨location⟩ | ⟨method_call⟩ | ⟨literal⟩ | '(' ⟨expr⟩ ')' | ⟨expr⟩ ⟨bin_op⟩ ⟨expr⟩
⟨statement⟩	→ ⟨location⟩+, = ⟨expr⟩+, | ⟨method_call⟩ | **WHILE** ⟨expr⟩ ⟨block⟩
⟨location⟩	→ ⟨id⟩ | ⟨id⟩ '(' ⟨int_literal⟩ ')' | ⟨id⟩.⟨location⟩
⟨method_call⟩	→ ⟨method_name⟩ ([⟨expr⟩+,])
⟨method_name⟩	→ **BUILD_CHART** | **HTABLE_DATA** | **WRITE_TO_FILE** | **PRINT**
⟨bin_op⟩	→ ⟨arith_op⟩ | ⟨rel_op⟩ | ⟨eq_op⟩
⟨literal⟩	→ ⟨int_literal⟩ | ⟨string_literal⟩ | ⟨angle_literal⟩ | ⟨dt_literal⟩
⟨id⟩	→ ⟨alpha⟩ ⟨alpha_num⟩*
⟨alpha_num⟩	→ ⟨alpha⟩ | ⟨digit⟩
⟨alpha⟩	→ a | b | ... | z | A | B | ... | Z | _
⟨digit⟩	→ 0 | 1 | 2 | ... | 9
⟨arith_op⟩	→ + | - | ∗ | /
⟨rel_op⟩	→ < | > | <= | >=
⟨eq_op⟩	→ == | !=
⟨int_literal⟩	→ ⟨decimal_literal⟩
⟨decimal_literal⟩	→ ⟨digit⟩ ⟨digit⟩*
⟨string_literal⟩	→ " ⟨alpha_num⟩* "
⟨angle_literal⟩	→ ' < ' ⟨int_literal⟩ ', ' ⟨int_literal⟩ ', ' ⟨int_literal⟩ ' > '
⟨dt_literal⟩	→ "D⟨int_literal⟩-⟨int_literal⟩-⟨int_literal⟩|⟨int_literal⟩:⟨int_literal⟩" }

Some specifications

An *AlakirQL* program is a sequence of directives that can be divided into two groups. Variables are initiated in the global scope (the only valid one) during field declarations and can be accessed by all methods in the program. Statements deal with processing of data stored in variables and represented by literals.

The *AlakirQL* keywords (PRINT, WHILE etc.) are case-insensitive, yet they are frequently expressed in all capitals; variable names are case-sensitive. White space must be used to separate keywords and identifiers. Any lexical tokens may be separated by white space. One or more spaces, tabs, page and line breaks, and comments are all examples of white space. Sequences starting with an alphabetic character or an underscore form a token with the sequence of characters following it.

Numbers in *AlakirQL* are 64 bit signed floating point num-s. An ⟨alpha_num⟩ is any printable ASCII character, as well as some special characters for zodiac signs. *AlakirQL* does not fully support data structures, but nevertheless uses a field specification access approach for ANGLE, DATE and CHART variables. ANGLE and DATE literals are regular expressions, so they have to be typed in a proper form to be successfully parsed.

Lexing

Lexical analysis in computer science refers to the process of transforming a series of characters into a sequence of lexical tokens (strings with an assigned and thus identified meaning). Almost always, parsing is divided into two smaller tasks. A lexer creates a potentially infinite sequence of tokens for use in the later stages of the parsing process [3]. The lexing module's operation is straightforward: when it traverses a file, it looks ahead. It attempts to assign meaning to the next word by checking in the following order: character, keyword, boolean, date literal, numeric literal, string literal, identifier, and so on. Further, all tokens are added to the list. The complexity of this process is LL(2). Due to space constraints, it is impossible to list all of the details; however, you can fully evaluate the progress by visiting our GitHub repository: https://github.com/prenaissance/alakirql.

Parse tree

A parse tree is a rooted tree that represents the syntactic structure of a string using some context-free grammar. The term parse tree is primarily used in computational linguistics. For the following simple valid code snippet a parse tree was generated: *DECLARE time = "D2001-01-01|00:00"; DECLARE data = HTABLE_DATA(time, "south", 11, "0N"); PRINT(data);*

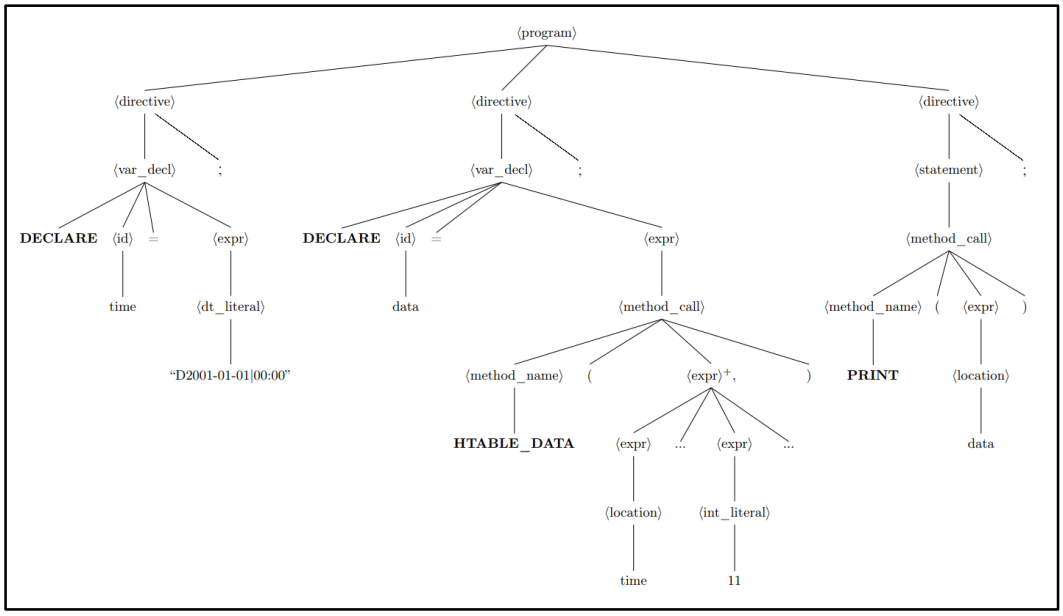


Figure 1. Parse Tree

Conclusion

Careful work at the stage of defining specifications and grammar allows to create a solid base for further development and expansion of the project. Stick to the REPL model and syntax simplification vector by introducing dynamic typing when declaring variables, for example; aim to win over the mass user. Hopefully, the product's final version will meet all expectations and become a useful tool in the field of astrology.

Sources

1. VOELTER, M. *DSL Engineering: Designing, Implementing and Using Domain-Specific Languages.* CreateSpace Independent Publishing Platform, 2013.
2. *Oxford Dictionary of English*. Oxford University Press.
3. DOMINUS, M.J. , *Higher-Order Perl: Transforming Programs with Programs*. Morgan Kaufmann, 2005.

THE STRUCTURE OF THE ISMS DOCUMENTATION IN ACCORDANCE WITH UPDATES TO ISO 27001:2022, 27002:2022

Dmitrii OBUH

Department of Software Engineering and Automation, Information Security, Faculty of computers, Informatics and Microelectronics, Technical University Of Moldova, Chisinau, Republic of Moldova

Corresponding author: Dmitrii OBUH, dmtrii.obuh@isa.utm.md

Tutor/coordinator: Rodica BULAI, university lecturer, in charge of the study program Information Security

Abstract. *Building information security in the modern world is one of the most important aspects of the stable development of any IT business. In order not to reinvent the wheel, it is most logical to use already proven solutions, namely to follow the international standards ISO 27001, ISO 27002. This paper will contain recommendations on the organization of the documentation structure according to the current updates of the above-mentioned standards.*

Keywords: *ISMS, documentation, ISO 27001, ISO 27002.*

Introduction

The ISO/IEC 27001:2022 is a global standard that outlines guidelines for managing information security in a company. This specification enables organizations of any kind to safeguard the security of various types of data, including employee information, financial records, intellectual property, and data shared with third-party entities [1].

The ISO 27001 standard outlines the necessary conditions for creating an Information Security Management System (ISMS) [2]. In contrast, the ISO 27002 provides advice on how to implement each of these conditions [3]. An ISMS involves a complete approach to handling confidential data within an organization, using risk management strategies to maintain security and protection.

An ISMS is designed to mitigate the risk of security breaches and maintain uninterrupted business operations by taking proactive measures to limit the impact of such incidents. It generally focuses not only on data and technology but also on employee behavior and processes [4].

These standards require specific documentation detailing policies and procedures. The disclosure of the topic and recommendations will be in the following chapters.

Contribution: In this paper, I develop an analysis of the most up-to-date standards to make recommendations on the organization of documentation to ensure information security. After studying many sources of information, I came to the conclusion that each organization needs its own set of documents, but nevertheless, it is possible to develop a versatile list that will be relevant and useful for many organizations. However, we should remember that an effective information security management system must be tailored to requirements of an organization [5].

A discussion about additional, but no less important documentation that is not included in the standards completes this paper.

Paper structure: This paper is structured in four sections, the first one being the introduction. In Section 2 I do an overview of the changes that have been made to the 2022 sample standard compared to 2013. Section 3 consists of recommendations on the structure of the documentation. We conclude with Section 4.

Overview of changes in the standards

The ISO 27001:2022 standard contains many useful optimizations. The number of sections in Appendix A has been reduced from 14 to 4 [6]. 57 security controls were merged, 11 new controls appeared. Here are new security controls:

- A.5.7 – Threat intelligence
- A.5.23 – Information security for use of cloud services
- A.5.30 – ICT readiness for business continuity
- A.7.4 – Physical security monitoring
- A.8.9 – Configuration management
- A.8.10 – Information deletion
- A.8.11 – Data masking
- A.8.12 – Data leakage prevention
- A.8.16 – Monitoring activities
- A.8.23 – Web filtering
- A.8.28 – Secure config

It's worth mentioning that the clause "6.2 Information security objectives and planning to achieve them" now states that the objectives should be monitored and also be available in a documented form [7]. Ultimately, if you have an ISMS and you want to update it in accordance with the latest standard, you need to do the following:

- Update the ISMS Management review procedure
- Update information security objectives and the Monitoring, measurement analysis and evaluation procedure
- Align the Risk Treatment Plan with the new structure of controls
- Update the Statement of Applicability
- Update the ISMS Communication Plan
- Update checklists and questionnaires used for audits
- Update other policies if needed

Recommendations on the structure of the documentation

The ISMS Documentation comprises several components, including the following levels:

- Level 0 - Corporate Information System Security Policy: This is the highest-level security policy of a company.
- Level 1 - ISMS Manual: This document outlines the ISO/IEC 27001 standard requirements and explains how the established ISMS satisfies those requirements.
- Level 2 - Supporting Policies and Guidelines: It consists of a complete collection of technical policies and guidelines that correspond to the ISMS scope.
- Level 3 - Procedures and Processes: This level contains the necessary processes and procedures for implementing and supporting the defined policies and guidelines.
- Level 4 - Templates and Forms: These templates and forms are utilized to streamline the operation of ISMS and serve as the foundation for records [8].

There are no specific rules regarding the format or quantity of documents required for an ISMS. Nevertheless, the information specified below must be recorded in written form [9]:

- Scope of the ISMS
- Risk assessment and risk treatment methodology
- Risk assessment and risk treatment report [10]
- Risk treatment plan
- Information security policy and objectives
- Statement of Applicability
- Acceptable use of assets
- Inventory of assets
- Definition of security roles and responsibilities
- Operating procedures for IT management
- Access control policy
- Supplier security policy

- Secure system engineering principles
- Incident management procedure
- Legal, regulatory and contractual requirements
- Business continuity procedures [11]

To satisfy the standards for continuous enhancement and offer substantiated proof of your ongoing initiatives, organizations should maintain written records of the following:

- Results of the management review
- Records of training, skills, experience and qualifications
- Internal audit program
- Results of internal audits [12]
- Monitoring and measurement results
- Logs of user activities, exceptions and security events
- Results of corrective actions

Presented below is a roster of guidelines that are necessary to adhere to the ISO 27001 Annex A. It is an essential addition to your documentations [13]:

- Mobile Device Policy
- Backup Policy
- Clear Screen and Clear Desk Policy
- Secure Development Policy
- Remote Work Policy
- Cryptographic Controls Policy
- Key Management Policy
- Information Transfer Policies (and Procedures)

The number and nature of extra documents required may vary depending on the organization's type and size. However, the subsequent records, which are beneficial for almost all organizations, should be kept:

- A policy outlining the classification of documented information
- A procedure for internal audit that is recorded
- A written procedure governing document control

A possible list of security policies that meet the requirements of ISO 27001 and ISO 27002 that will suit most companies may be the following:

- Human Resources Security Policy
- Incident Response Policy
- Project & Resource Management Policy
- Third-Party Management Policy
- Security & Privacy Governance Policy
- Vulnerability & Patch Management Policy
- Antivirus policy
- Secure Engineering & Architecture Policy
- Identification & Authentication Policy
- Network Security Policy
- Data Classification & Handling Policy
- Asset Management Policy
- Security Operations Policy
- Security Awareness & Training Policy
- Continuous Monitoring Policy
- Change Management Policy
- Compliance Policy
- Physical & Environmental Security Policy
- Information Exchange Security Management Policy

- Capacity & Performance Planning Policy
- Configuration Management Policy
- Business Continuity & Disaster Recovery Policy [14]
- Endpoint Security Policy
- Maintenance Policy
- Privacy Policy
- Technology Development & Acquisition Policy

Additional documents frequently used in the implementation of ISO 27001 [15]:

- Documentation management procedure
- Password Policy
- Rules for working in safe zones
- Business Impact Analysis
- Maintenance and inspection plan
- Disposal and Destruction Policy

The documentation for operations needs to comprise comprehensive directives pertaining to:

- Restarting and recovering the system
- Handling and processing of information
- Procedures for disaster recovery
- Management of changes, including scheduled maintenance and interrelationships between components
- Date of the last review and update
- Handling of audits and system logs
- Document classification
- Procedures for handling exceptions, along with a record of any exceptions
- Contacts for operations, technical support, emergency, and business purposes
- Managing output and media, including the secure disposal or destruction of sensitive data
- Back-up and recovery, encompassing on-site and off-site storage options
- Computer room management and safety protocols

It is also advisable to document other common activities, such as:

- The creation of a Risk Treatment Plan by the Risk Committee
- Scheduling and reports from Internal Audits
- Reviews of the Information Security Management System by Management
- Assessments of non-conformities or reported incidents by the Information Security Team

Conclusions

Following the ISO 27001, ISO 27002 industrial standards, it is possible to organize the structure of the ISMS documentation most effectively.

There are organizations that do not pay due attention to the elaboration of documents declaring the goals and objectives of information security, a set of measures to improve and modernize the information security of the enterprise. Negligent attitude to the creation of the ISMS documentation can subsequently lead to large time costs in the event of information security incidents, to an increase in cases of unintentional damage to the organization by employees due to insufficient awareness of the basic provisions and principles of information security.

In this paper, I talked about ISO standards, conducted a comparative analysis of the most current versions of 2022 with the versions of 2013, pointed out what you should pay attention to in order to update the company's documentation in accordance with the latest version of the standard, being certified according to the previous version, and also made recommendations on the organization of the structure of the ISMS documentation, which suitable for most organizations.

References

1. QUSEF, A., ALKILANI, H. The effect of ISO/IEC 27001 standard over open-source intelligence. In: *PeerJ. Ltd*, 2022.
2. BRENNER, J. ISO 27001 risk management and compliance. In: *Sabinet Online*, 2007.
3. EVANS, L. Protecting information assets using ISO/IEC security standards. In: *Association of Records Managers & Administrators (ARMA)*, 2016.
4. YASAR, K. What is Information Security Management (ISMS). [online]. 2022. [accessed 25.10.2022]. Available: https://www.techtarget.com/whatis/definition/information-security-management-system-ISMS#:~:text=The%20goal%20of%20an%20ISMS,well%20as%20data%20and%20technology
5. SALKIC, A. ISO 27001: What documentation is required for certification. [online]. 2021. [accessed 04.11.2022]. Available: https://sec-consult.com/blog/detail/iso-27001-what-documentation-is-required-for-certification/
6. KOSUTIC, D. ISO 27001 2013 vs. 2022 revision – What has changed. [online]. 2022. [accessed 29.10.2022]. Available: https://advisera.com/27001academy/blog/2022/02/09/iso-27001-iso-27002/
7. PROZOROV, A. ISO 27001:2022. What has changed. [online]. 2022. [accessed 01.11.2022]. Available: https://www.securitylab.ru/blog/personal/80na20/352401.php
8. BISWAS, P. ISO 27001:2013 ISMS Manual. [online]. 2020. [accessed 02.11.2022]. Available: https://preteshbiswas.com/2020/01/08/iso-270012013-isms-manual/
9. ISO 27001 and ISO 27002 Compliance, ImmuniWeb. [online]. [accessed 02.11.2022]. Available: https://www.immuniweb.com/compliance/iso27001-compliance-audit-checklist/
10. BISCOE, C. ISO 27001 Documentation Checklist. [online]. 2017. [accessed 03.11.2022]. Available: https://www.vigilantsoftware.co.uk/blog/iso-27001-documentation-checklist
11. TALIB, M.A., KHELEFI, A., BARACHI, M.E., ORMANDJIEVA, O. Guide to ISO 27001: UAE case study. In: *Informing Science Institute*, 2012.
12. MAUS, B. ISO 27001 Certification – What to Consider. [online]. 2022. [accessed 05.11.2022]. Available: https://otrs.com/otrsmag/iso-iec-27001-certification/
13. ISO 27001 Checklist: The Documentation Required, Risk Crew. [online]. [accessed 28.10.2022]. Available: https://www.riskcrew.com/2022/07/iso-27001-documentation-whats-required-and-whats-optional/
14. ISO 27001 & 27002 Based Security Documentation, ComplianceForge. [online]. [accessed 03.11.2022]. Available: https://www.complianceforge.com/product/iso-27001-27002-policies-standards-cdpp/
15. Documentatsiya pod ISO 27001:2013. Skolko yeyo dolzhno byt i kakoy? [Documentation for ISO 27001:2013. How much should it be and what?], Export center. [online]. [accessed 05.11.2022]. Available: https://export-center.by/blogpost/documents-27001

ПРИМЕНЕНИЕ 3D-МОДЕЛИРОВАНИЕ В НЕЙРОХИРУРГИИ ДЛЯ УЛУЧШЕНИЯ ПЛАНИРОВАНИЯ И ПРОВЕДЕНИЯ ОПЕРАЦИЙ НА ГОЛОВНОМ МОЗГЕ И СПИННОМ МОЗГЕ

Анатолий ГУМЕНЮК[1,2,*], Ростислав КЭЛИН[1]

[1]*Департ. Информатики, SOFT и автоматики, докторант DISA, Технический Университет Молдовы, г. Кишинев, Республика Молдова*
[2]*Институт неврологии и нейрохирургии Diomid Gherman, нейрохирург, заведующий отделения хирургии 1 дня, Кишинев, Республика Молдова*

[*]Автор кореспондент: Анатолий ГУМЕНЮК, e-mail gumeniucanatolie6@gmail.com

Научный руководитель: Балабанов Анатолий, *Технический Университет Молдовы, г. Кишинев РМ,* академик РАЕН, д-р.хаб., профессор**, / Научный консультант: Романенко Александр .** *Технический Университет Молдовы, г. Кишинев РМ,* д-р., доцент

Abstract - *Приведенный авторами обзор литературных источников и Интернет-данных может служить отправной точкой для исследований в области создания мобильного автономного автоматизированного рабочего места нейрохирурга (АРМНХ) на основе оригинального программного обеспечения, разрабатываемого авторами доклада.*

Ключевые слова: *цифровая визуализация и коммуникации в медицине, программное обеспечение (функционал) для 3D-моделирование он-лайн, головной и спиной мозг человека, хирургическое лечение в нейрохирургии. хроническая субдуральная гематома*

Введение

Современный этап развития естествознания характеризуется необходимостью исследований различных мульти дисциплинарных подходов. Так, например, сложные медико-биологические проблемы решаются уже не только с помощью инструментов медицины, но и методами биоинженерии, software-инженерии, а также путём модернизации известных и/или созданием новых прикладных информационных систем.

Действительно, при планировании хирургических операций на головном или спинном мозге могут быть использованы 3D-модели неблагоприятных очагов хирургии для более точного и детального их лечения. Хирурги могут использовать 3D-модели, чтобы лучше понимать сложные анатомические отношения внутри головного мозга и спинного мозга, а также разрабатывать стратегии для минимизации рисков и осложнений во время операции. Так в области практической медицины, хирурги всегда сталкивались с трудной задачей проведения операций на сложных анатомических структурах человеческого тела на основе лишь знаний анатомии человека и личного врачебного опыта. Благодаря новым технологиям медицинской визуализации в стандарты нейрохирургии сегодняшнего дня вошли такие инструменты, как мульти спиральная компьютерная томография (МСКТ) и магнитно-резонансная томография (МРТ), что сделавшие диагностику менее инвазивной и более информативной. (Все методы обследования можно разделить на неинвазивные и инвазивные – связанные с вмешательством в организм пациента. К неинвазивным относят УЗИ, часть лабораторных исследований и лучевых методик. Инвазивными являются эндоскопические вмешательства и лучевые методики, связанные с введением в сосудистое русло пациента катетеров и контрастных веществ.)

Рисунок 1. Мультиспиральная компьютерная томография (МСКТ)

Постановка задачи

У хирурга есть возможность, при подготовке к хирургическим операциям, использовать (или создать) программные процедуры сегментации (выделения) объектов хирургического вмешательства по DICOM-сериям. (DICOM-серия – пакет фото снимков МСКТ или МРТ в типовом формате их представления для визуализации на экране пользователя).

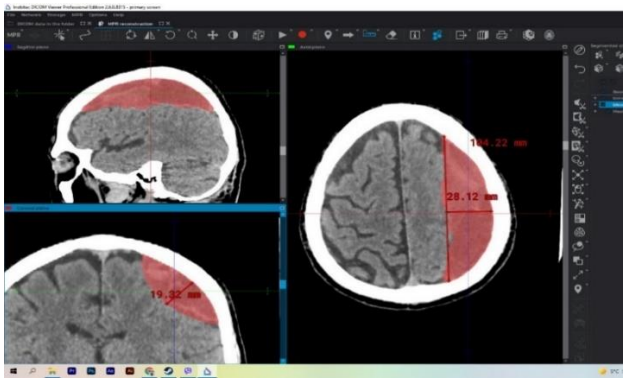


Рисунок 2 Двумерное изображение патологического очага в головном мозге в различных плоскостях

В настоящее время известны отдельные работы, связанные с методикой предварительной диагностики, предлагающие полуавтоматическую воксельную и полигональную аппроксимацию различных анатомических образований. Конечная объёмная модель (интерьер сцены) должна представлять собой набор выбранных на построенной модели мозга таких структур, которые позволят спланировать и реализовать решение поставленной хирургической задачи. Конечный результат сегментации существенно зависит, как от качества исходных данных, так и от точности и чистоты (тщательности) выделения диагностируемого объекта.

Выводы

Большим плюсом описанного подхода является возможность создавать модели только из необходимых для решения задачи объектов, избавляя визуализируемую сцену от незначимого с точки зрения хирургии «рентгенологического шума».

Особо ценной является и возникающая также возможность объединения образований, сегментированных из различных серий МРТ/МСКТ или даже ранее выполненных исследований.

3D-модели могут быть использованы также для обучения и тренировки хирургов перед операцией. Хирурги могут использовать 3D-модели для изучения анатомических особенностей конкретного пациента, планирования операции и тренировки процедур, которые могут быть трудными или рискованными [1,2].

Важное значение приобретают навигационные свойства управления 3D-моделью

Во время операции хирург может использовать 3D-модель для навигации в режиме реального времени. Это позволяет хирургу получать более точную информацию о местоположении инструментов и тканей во время операции, что помогает уменьшить риск повреждения важных структур мозга.

3D-модели могут быть использованы для визуализации сложных анатомических отношений и патологий в головном мозге и спинном мозге. Это позволяет хирургам более точно определять местоположение опухоли или других аномалий, а также предсказывать возможные последствия для пациента.

3D-модели могут быть использованы для создания индивидуальных имплантантов, таких как кости черепа или титановые пластины. Это позволяет более точно соответствовать индивидуальным потребностям конкретного пациента и уменьшить риск осложнений и реакций на имплантат [3-10].

Приведенный авторами обзор литературных и Интернет-данных является не полным и будут расширен, но уже может служить отправной точкой для исследований в области создания мобильного автономного автоматизированного рабочего места нейрохирурга (АРМНХ) на основе оригинального программного обеспечения, разрабатываемое авторами доклада.

Для конкретизации текущих и сопутствующих локальных задач наших исследований была выбрана довольно часто встречающаяся патология из раздела нейро травматологии - хроническая субдуральная гематома (ХСДГ) [11-15].

Наконец, важное значение приобретает вопрос защиты двоичной информации против помех, сбоев, хищения и/.или вредительского искажения персональных данных. При этом потребуются последние достижения в области криптографии, достигнутые в научной группе наших программистов и IT-специалистов

Библиография

1. КОКРО Р.А., СЕРРА С., ЦЕНГ С.Л . Нейрохирургические применения технологий трехмерной печати. В: *Неврология*. 2015 г.; 84(24): 241-250.
2. РИДЕРЕР Б.М., БОЛТ С., БРЕННЕР Э.Точность трехмерных печатных медицинских моделей. В: *Am J Нейрорадиология*. 2018; 39(3): 438-445.
3. СУГАВАРА Ю., ХИГАСИГУТИ Т., ЧИКУДА Х.Применение 3D-печати для предоперационного моделирования в нейрохирургии: *обзор. Сург Нейрол Инт*. 2016; 7 (Приложение 12): S347-S352.
4. YOON JW, CHEN RE, KIM DJ.Технология трехмерной печати и нейрохирургия: готовы к прайм-тайму? *В: Дж Нейрохирург*. 2017; 127(4): 718-726.
5. ZAIDI HA, MONTOURE A, NAKAJI P . Применение 3D-печати для предоперационного планирования в нейрохирургии. *В: Дж. Клин Нейроски*. 2016; 31: 220-226.
6. МАРТЕЛЛИ Н., СЕРРАНО С., ВАН ДЕН БРИНК Х.Преимущества и недостатки трехмерной печати в хирургии: систематический обзор. *В: Операция.* 2016; 159(6): 1485-1500.
7. ЯННОПУЛОС А.А., ШТАЙНЕР М.Л., ДЖОРДЖ Э.Кардиоторакальные применения 3-мерной печати. *В: J Торакальная визуализация*. 2016; 31(5): 253-272.
8. ЧАРРИ Д.Д., ДЕ АМОРИМ Р.Л., ГОНСАЛЕС-БЛОМ С.А Применение 3D-печати в нейрохирургии: *В:Дж. Клин Нейроски*. 2019; 63: 10-17.
9. WEI W, ZHANG X, LI Q. Применение технологии 3D-печати в хирургическом планировании черепно-мозговой травмы. *В: J Craniofacial Surg.* 2019; 30(2): e128-e130.

10. НИЗАМУДДИН Н., СМИТ ДЖ. Т., ХСУ Ф.П. Нейрохирургическая 3D-печать и интраоперационная нейронавигация. *В: J Нейрохирург Sci.* 2021; 65(2): 166-171.
11. HUANG Y, ZHANG Z, CHEN W. Application of three-dimensional computed tomography angiography in surgical treatment of chronic subdural hematoma. B*: World Neurosurgery*. 2020;138:e463-e470. doi: 10.1016/j.wneu.2020.02.096
12. WANG K, ZHANG Z, HUANG Y. Application of three-dimensional reconstruction in the surgical treatment of chronic subdural hematoma. *B: Experimental Therapeutic Medicine.* 2019;18(5):3279-3284. doi: 10.3892/etm.2019.7939
13. TAN H, ZHANG J, LIN.Application of 3D reconstruction technique in surgical treatment of chronic subdural hematoma. *B: International Journal of Clinical Experimental Medicine.* 2017;10(4):7007-7012.
14. YEOM JS, KIM SH, OH YH, SHIM JJ. Three-dimensional computed tomography for chronic subdural hematoma. *B: Jornal of Clinical Neuroscience.* ;14(4):327-330. doi: 10.1016/j.jocn.2006.02.027
15. CHAKRABORTY S, GHOSH SN, MUKHOPADHYAY A, DAS SK, BISWAS BK. Three-dimensional computed tomography in surgical management of chronic subdural hematoma: a comparative study with burr hole drainage. *B: Asian Journal of Neurosurgery*. 2018;13(1):78-83. doi: 10.4103/ajns.AJNS_91_16

UNLOCKING THE POWER OF QUERY LANGUAGE

Cristina ȚĂRNĂ[1], Tudor SCLIFOS[1 *],
Dan HARITON[1], Leonid CAMINSCHI[1],
Andrei CEBAN[1]

[1]*Software Engineering and Automatics, Technical University of Moldova, Group FAF-211, Faculty of Computers, Informatics and Microelectronics, Chișinău, Republic of Moldova*

**Corresponding author: Tudor Sclifos, tudor.sclifos@isa.utm.md*

Abstract. *This article explores the differences between Role-Based Access Control (RBAC), and Query language. The article delves into the technical aspects of each language, including its strengths and weaknesses. Additionally, the article highlights the developed Query language and its strong sides. The access is granted based on the role the user has and the accorded time. The base roles are select, delete, insert and update. By the end of this article, a comprehensive understanding of the access control policies will be established and how they can be combined with a Query language.*

Keywords*: authorization, privileges, security, enforcement, control*

Introduction

Access control policies are a cornerstone of information security, providing a means to restrict access to resources and data to only those users who are authorized to access them. Access control policies define the rules for granting and revoking access to resources and specify the conditions under which access is granted or denied. However, defining and enforcing access control policies can be a complex and challenging task, particularly in large and dynamic environments where access requirements are constantly changing.

Access control policy languages provide a formal framework for expressing access control policies in a machine-readable format. These languages enable organizations to define access control policies in a structured and consistent way, ensuring that policies are applied uniformly across the organization. Policy languages also enable automated enforcement and compliance monitoring, reducing the burden on security teams and improving the overall effectiveness of access control.

In this article, the key concepts and best practices of access control policy languages will explore. The various types of policy languages, including attribute-based access control (ABAC), role-based access control (RBAC), and mandatory access control (MAC) will be examined. The challenges of implementing access control policies and emerging trends and technologies that are shaping the future of access control policy languages will be explored. By the end of this article, a solid understanding of access control policy languages and the role they play in securing modern information systems will be established.

RBAC, frequently known as role-based security, is a technique for restricting system access to accredited users in computer system security. It is a strategy for enforcing either MAC or optional access control (DAC) [1].

A policy-neutral access-control method built around roles and privileges is known as role-based access control. It is simple to carry out user assignments thanks to RBAC's components, similar as role warrants, user roles, and role-role connections. To simplify security administration in big companies with thousands of users and hundreds of authorizations, RBAC can be a great tool. Even though RBAC is unlike MAC and DAC access control frameworks, it may effortlessly enforce these constraints.

1. Domain description and analysis

A domain-specific language (DSL) is a software language that's domain-specific. A software language is a language that is written and read by humans, but also automatically processed by computers. This section will touch upon the key aspects of a software language.

A domain is a focused area of knowledge. It comes with a group of domain experts owning the domain: people possessing, shaping, extending that area's knowledge, and sharing it with business stakeholders and other interested parties. Some could even claim that a domain frequently exists in the first place as such a group of experts in the field, with that group defining it ad hoc.
This pre-DSL is already domain-specific but needs some effort to become a software language. Looking at the key aspects of a software language, we can decide how to podcast the right DSL from the pre-DSL. With that information, starting to implement DSL as a software language is easy. That implementation takes the form of tools like the Domain IDE and the code generator. The Domain IDE allows experts in the field to write and read DSL content with DSL editors.

Adopting a DSL-based approach has many positive effects on software development. Here are some potential motivations for using a DSL:

- Abstraction: a DSL allows developers to abstract complex programming concepts and focus on a specific domain. This can make the code more readable and easier to maintain.
- Productivity: with a DSL, developers can create software quickly and efficiently. DSLs allow developers to express ideas in a concise and expressive way, which can speed up development time.
- Improved Communication: DSLs can improve communication between developers and non-technical stakeholders. With a DSL, business analysts or domain experts can more easily express their needs and requirements, without requiring a deep technical understanding.
- Safety: a DSL can help to catch errors early before they cause problems. By providing a language that is designed specifically for a domain, developers can write code that is less error prone.
- Customization: a DSL can be customized to a specific problem or domain, which can make it more powerful and easier to use.
- Domain-Specific Optimizations: a DSL can be perfected specifically for the domain it serves. This can lead to more efficient code and better performance.

The DSL chosen for this article is a Query language combined with RBAC.
A query language is a programming language designed to retrieve and manipulate data stored in a database. It is a specialized language that allows users to ask specific questions, or queries, about data in a structured way.

There are various query languages used for diverse types of databases, including SQL (Structured Query Language), which is used for relational databases, and NoSQL query languages like MongoDB query language and Cassandra Query Language, which are used for non-relational databases.

Query languages typically consist of statements or commands that are used to interact with a database, such as SELECT, INSERT, UPDATE, and DELETE. These statements allow users to retrieve specific data from a database, add new data, update existing data, and delete data.

Query languages are essential for working with databases, as they allow users to access and manipulate enormous amounts of data in a structured and efficient way.

2. Grammar

For greater comprehension, further the grammar for the Query language is represented according to an amazingly simple and textual program. The language allows the user to access, modify, insert and delete elements from a non-relational database. Next, each feature of grammar is shown in detail.

The DSL design includes a number of stages. Firstly, the definition of the programming language grammar is $G = (Vn, Vt, P, S)$:

Vn – a finite set of non-terminal symbols;
Vt - a finite set of terminal symbols.
P – a finite set of the production of rules;
S - the start symbol;

In Table 1 are meta-notations for specifying the grammar.

Table 1

Meta notation

Notation (symbol)	Meaning
<foo>	foo is a nonterminal.
"foo" | 'f'	means that foo and f are terminal i.e., a token or a part of a token.
x?	means zero or one occurrence of x, i.e., x is optional
x*	means zero or more occurrences of x
x+	one or more occurrences of x
{}	used for grouping
|	separates alternatives

```
<query> -> <selectquery> | <generatekeyquery> | <createuserquery> |
<loginquery> | <createquery> | <insertquery> | <updatequery> |
<deletequery>
| <help>
<selectquery> -> <select_clause> <from_clause> <where_clause>?
    <order_by_clause>?
<select_clause> -> "SELECT" <field_name> {"," <field_name>}* | "*"
<from_clause> -> "FROM" <collection_name>
<where_clause> -> "WHERE" <condition>
<order_by_clause> -> "ORDER BY" <field_name> ("ASC" | "DESC")?
<generatekeyquery> -> "generate_key" <privilege> <privilege> <privilege>
    <privilege> <time>?
<privilege> -> <boolean>
<time> -> <int>
<createuserquery> -> "create_user" <username> <password> <mail> <token>
<mail> -> <string> "@" <string> "." <string>
<token> -> <string>
<loginquery> -> "log_in" <username> <password>
<createquery> -> <create_clause> <codeblock>
<create_clause> -> "CREATE TABLE" <collection_name>
<codeblock> -> "(" <table_fields> ")"
<table_fields> -> <collection_collumn> {"," <collection_collumn>}*
<collection_collumn> -> <type> <field_name>
<insertquery> -> <insert_clause> <values_clause><insert_clause> -> "INSERT
INTO" <collection_name>
<values_clause> -> "VALUES" "(" <value> {"," <value>*} ")"
<updatequery> -> <update_clause> <where_clause> <set_clause>
<update_clause> -> "UPDATE" <collection_name>
<set_clause> -> "SET" <field_name> "=" <variable>
<deletequery> -> <delete_clause> <where_clause>?
<delete_clause> -> "DELETE FROM" <collection_name>
<help> -> "HELP"
<username> -> <string>
<password> -> <string>
<condition> -> <comparison> | <logical_op> | "(" <condition> ")" |
<comparison> -> <variable> <comparison_op> <variable>
<variable> -> <field_name> | <value>
```

```
<logical_op> -> <condition> <logical_op> <condition>
<comparison_op> -> "==" | "!=" | ">" | ">=" | "<" | "<="
<field_name> -> <string>
<collection_name> -> <string>
<value> -> <string> | <number> | <NULL>
<NULL> -> '0' | ''
<type> -> <string> | <int> | <float> | <char> | <digit> | <boolean>
<string> -> '"' <char>* '"'
<int> -> <digit>+
<float> -> <digit>+ "." <digit>+
<char> -> 'a' | 'b' | … | 'z' | 'A' | 'B' | … | 'Z'
<digit> -> '0' | '1' | ... | '9'
<boolean> -> 'True' | '1' | 'False' | '0'
```

Parse Tree

Statement example:

SELECT * FROM foo WHERE id=prev_id ORDER BY id

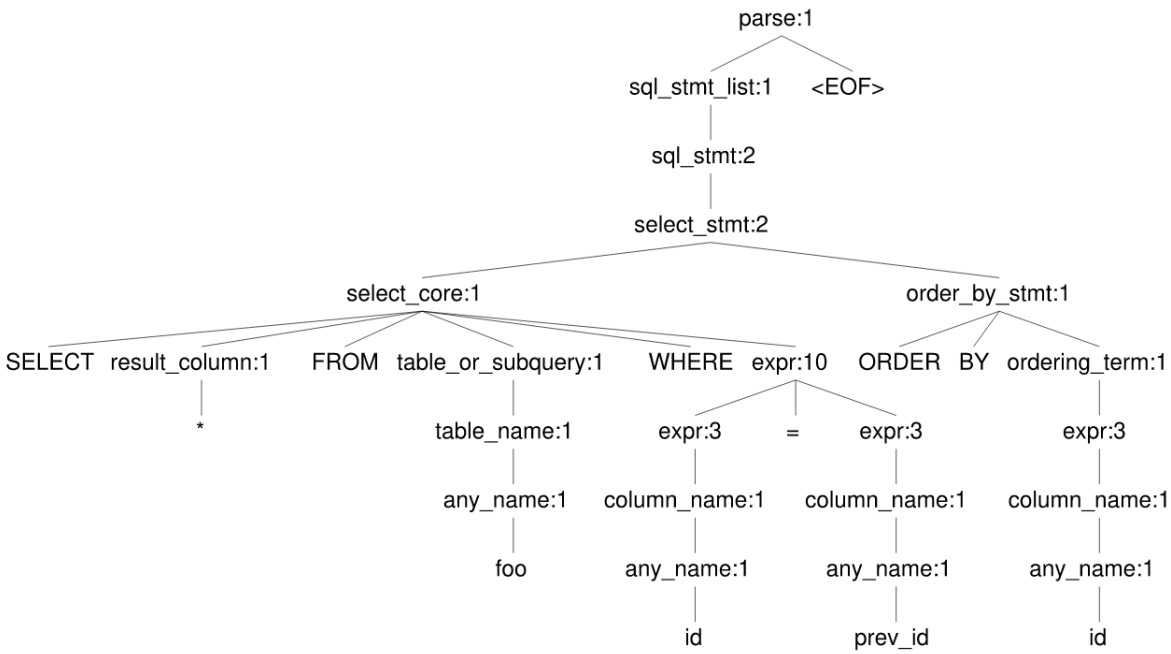


Figure 1. Parse tree for the statement example

Conclusions

To sum up, the query language presented is easy to understand and implement, as it is simpler than SQL. Overall, each language has its strengths and weaknesses, and the choice of which to use will depend on the specific needs and requirements of the organization. It is essential to carefully consider the advantages and limitations of each language before implementing an access control system.

References

1. FERRAIOLO, D.F. & KUHN, D.R. Creativity and Imagination: Role-Based Access. [online]. [accessed 5.03.2023]. Available at: https://csrc.nist.gov/publications/detail/conference-paper/1992/10/13/role-based-access-controls
2. MEINTE, B. Domain-Specific Languages Made Easy. [e-book] Available at: https://www.manning.com/books/business-friendly-dsls

SUBCONȘTIINȚA ARTIFICIALĂ

Vadim STRUNĂ[1,2]

[1]*Departamentul Informatică și Ingineria Sistemelor, Facultatea Calculatoare Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova*
[2]*Școala Doctorală, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova*

Autorul corespondent: Vadim Strună, e-mail: vadim.struna@iis.utm.md

Îndrumător/coordonator științific: Victor ABABII, conferențiar universitar, doctor, DIIS

Rezumat. *În această lucrare este prezentată o nouă parte din lumea computerelor. Mai exact, o nouă ramură a informaticii care va ajuta la calculul unui foarte mare volum de informații: Subconștiința Artificială (SA) și va fi întotdeauna foarte aproape de Inteligența Artificială. În zilele noastre vedem cât de mare e viteză dezvoltării generale a societății. Scopul este de a ajuta omenirea să construiască creierul artificial cu mai multă precizie și de a crea o nouă tehnologie în IT pentru a atinge pașii propuși. Algoritmul din această lucrare ne va arăta cum am putea obține un calcul mai dinamic pentru a primi decizii mai eficiente în sistemele informatice în dinamicitatea tehnologiilor din zilele noastre și din viitor.*

Cuvinte cheie*: Subconștiința Artificială (SA), Inteligență Artificială(IA), bază de cunoștințe, ADN, cluster, membran computing, programare celulară.*

Introducere

Ca prototip pentru SA alegem creierul uman, care pe lângă calcule și operații logice are și o mulțime de mistere inexplicabile pe care mereu omenirea se străduie să le explice și să le folosească constructiv. Nici până în vremea de astăzi știința nu știe exact toate capacitățile creierului și nu poate explica algoritmul său de funcționare.

Logica apariției SA este următoare: ca să trecem modelul creierului uman la calculatoare, atribuim Conștiinței umane Inteligența Artificală. Iar Subconștiinței umane îi atribuim Subconștiința Artificială. Deci ca să explicăm cum va lucra SA trebuie să înțelegem cum lucrează Subconștiința umană. Acest domeniu este foarte puțin studiat. Se poate afirma că medicina este abia la început de cale în studierea Subconștiinței umane. Cu toate acestea omenirea știe deja câteva lucruri importante despre Subconștiință. Sunt multe definiții care ne arată existența Subconștiinței. Dar noi, deja am început a implementa modelul funcționării creierului la modelul calculatoarelor. Noi știm că Subconștiința are niște procese ascunse de noi încă la moment, dar care știm că ele ajută la procesul de existență a omului. Omenirea are multe de descoperit până a explica cum totuși funcționează creierul nostru.

Trecând la modelul calculatoarelor, noi cu toții știm că procesorul calculatorlui nu este capabil să numere mai mult decât cifrele 0 și 1. Dar cu performanța procesoarelor moderne de astăzi este destul pentru a efectua calcule gigante. Vom folosi această logică și pentru SA în formarea deciziilor.

Vom propune Inteligenței Artificiale un ajutor prin apariția Subconștiinței Artificiale.

În Inteligența Artificială trebuie să fie mai multe celule inteligente [1], pe care să le putem folosi în diferite situații necesare nouă. De asemenea, aici ne vom folosi și de partea tehnologiei – Baza de cunoștințe. Aceste compartimente vor fi cei sunt trei piloni de bază a ideii SA.

Vor fi create noi celule inteligente [3] bazate ca mindmaping, care vor apărea când este necesar și vor funcționa cu baza de cunoștințe. Aceste celule vor fi ca obiecte, pot fi create cu clasele în limbajele de programare dinamică.

Acest sistem se bazează pe aceste celule care la rândul lor sunt implicit pasive, dar care ne-ar putea ajuta în calcule mult mai dificile, astfel am putea avea un creier artificial de o performanță foarte înaltă.

Cu acest creier trebuie să ajutăm omenirea să găsească răspunsul la multe întrebări, care ar putea fi aplicate [2] pentru a obține noi tehnologii în viitorul apropiat și a ajuta popoarele să ia decizii corecte, pentru a salva această planetă de ei înșiși.

Omenirea are nevoie de decizii importante și corecte.

Această idee a apărut în septembrie 2020 și va fi o lucrare de doctorat. Iată-ne la un început cu această nouă idee în informatică.

După cum știm, aproape fiecare din lucrările științifice sunt inspirate din natură. Prototipul meu este creierul uman cu toate procesele sale cunoscute de știința modernă din domeniiu.

Un moment important, de ce avem nevoie de acest sistem inteligent? Desigur, computerele sunt mai eficiente decât oamenii în capacitatea de stocare a informațiilor și mai buni la calculul unui număr de operațiuni într-o secundă, dar în logică și luarea deciziilor, sunt mult mai slabe.

De aceea am venit aici cu sistemul nou pentru a înțelege logica umană prin subconștiință.

De la început vom arăta aici ce este Conștiința pe modelul uman. Vom adapta acest concept la industria computerelor. Sper că acesta este un pas înainte pentru o lume a tehnologiilor noi

Trăim într-o lume uriașă a „pânzei de păianjen”, cu multe lucruri interesante în jur. În primul rând aici putem da definiții simple ale conștiinței și subconștiinței umane, fiindcă, după cum am menționat mai sus, atunci când am început această temă, ne-am inspirat din modelul viu de viață, modelul activității creierului uman. Am fost interesați cum funcționează, cum se întâmplă? Cum folosesc oamenii acest potențial energetic al creierului lor?

Conștiința – aceasta este logica, informația, gândurile, atenția, când citim sau vorbim... Cu alte cuvinte, toate acțiunile noastre conștiente. Vedem că aici nu este loc pentru sentimente și emoții.

Subconștiința – acest termen încă este ceva necunoscut exact în acest moment în lumea medicinei științifice. În cuvinte generale, acesta este un rezervor de procese nelogice de unde pot veni decizii interesante.

Subconștiința în computere - este bazat pe deciziile care sunt luate pe baza cunoștințelor ascunse și prelucrarea imaginilor.

Vom folosi toate calculele doar pe date matematice, logica matematică, unde desigur codul binar este la baza tuturor evenimentelor digitale și cu ajutorul logicii codului binar vom forma o parte a bazei de cunoștințe.

Subconștiința Artificială va lucra cu cifre de mărimi foarte mari.

1.Apariția *Subconștiinței Artificiale*

Ca să ne fie mai ușor să înțelegem Inteligența Artificială, va fi înțeleasă de noi drept fiind conștiința umană. Iar *Subconștiința Artificială* va fi va fi înțeleasă de noi drept fiind subconștiința umană Fig. 1.

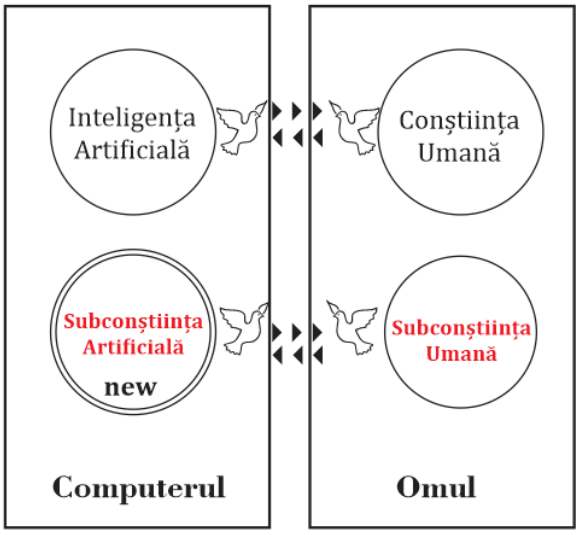


Figura 1. Prototipul apariției SA în informatică.

Necesitatea. - E nevoie de creat un algorim nou de calcule pentru un sistem inteligent informatic mai aproape de capacitatea creierului uman.

Știm cu toții că acest termen de inteligență artificială este încă totuși abstract. Dorim să credem că avem o definiție clară a inteligenței artificiale, dar nu am găsit-o în cele din urmă, o definiție frumoasă și explicită. Avem un *ocean de informații* pe care umanitatea trebuie s-o structureze după priorități și necesități. Știm că în subconștientul uman există multe informații ascunse de conștiiță, dar care cumva acționează asupra noastră. Deacea acest model îl putem trece în domeniul informaticii, care ne poate fi de mare folos.

Cum are loc prelucrarea imaginii cercetate. Pentru a înțelege mai bine, am luat ca exemplu celula de la om. După cum știm, în corpul nostru avem peste 10 miliarde de celule. Fiecare celulă Fig. 5 lucrează singură și are scopul, și drumul său. Celulă poate fi un obiect, care ar putea rezolva una sau mai multe probleme. În programarea OOP, un obiect sau o clasă va fi o celulă. Aceste celule ar putea face multe lucruri. Și scopul nostru este de a face ca aceste obiecte să funcționeze cât mai eficient și rapid la necesitate.

Informațiile din fiecare celulă vor putea fi salvate, aplicând programarea OOP unde obiectul poate funcționa cu un constructor, care în caz de necesitate, poate fi distrus prin crearea unui destructor.

Când primim ca material de prelucrare a informației o poză, o prelucrăm în felul următor:

La început va fi un obiect mare de bază (poza noastră). Împărțim imaginea respectivă în patru clustere. Deja un cluster va fi un obiect cu metodele, date și proprietățile sale. Proprietățile fiecărui cluster sunt asemănătoare cu ADN-ul omului. După cum se știe, fiecare celulă a ființei vii are un ADN. Acest ADN ar putea fi definit cu o singură adresă. Este la fel cum o celulă de memorie, acea lădiță, din fiecare limbaj de programare are un ID în care am putea plasa variabila noastră. De asemenea, ADN are proprietăți de multiplicare și de împărțire.

2.Explicarea *Subconștiinței*

După cercetările despre conștientul și subconștientul uman, am realizat un model inițial matematic despre cum funcționează. Și ce am primit? E foarte interesant!

De la sursa care analizează obiectul până la acest obiect sunt o mulțime de linii de diferite lungimi (direcțiile 1 și 2) Fig.3, până la fiecare punct a obiectului cercetat. Dimensiunea acestui punct va fi setată de noi, respectiv având posibilitatea de a modifica raza lui. Acest punct, va fi unitatea noastră de măsură a suprafeței obiectului cercetat. Pe măsură ce raza va fi mai mică, cu atât precizia informației despre obiectul studiat va fi mai bună.

Imaginea noastră va conține o multitudine de puncte B1 ... Bn, Fig.3.

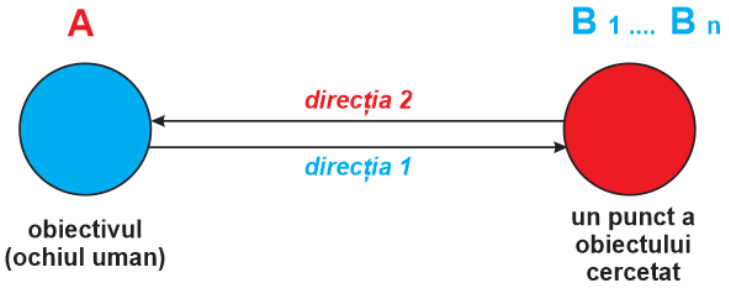


Figura 3. Drumul unei distanțe parcurs de la sursă la obiect

Sumând toate distanțele de la ochiul uman spre toate punctele obiectului văzut, prin intermediul informațiilor, care se află în subconștientul uman și care este primul rezultat al opiniei umane despre acest obiect, primim un număr foarte mare X. Fig. 4.

Litera majusculă A la noi este sursa de lumină (vedere) care analizează obiectul. Fig 3.

Literele B1...Bn, sunt punctele din care este compusă imaginea noastră procesată.

Noi mereu prelucrăm cu ajutorul creierului cifre gigante. Atunci cu siguranță putem trece acest model în domeniul computerelor, care cu calculele matematice stau perfect.

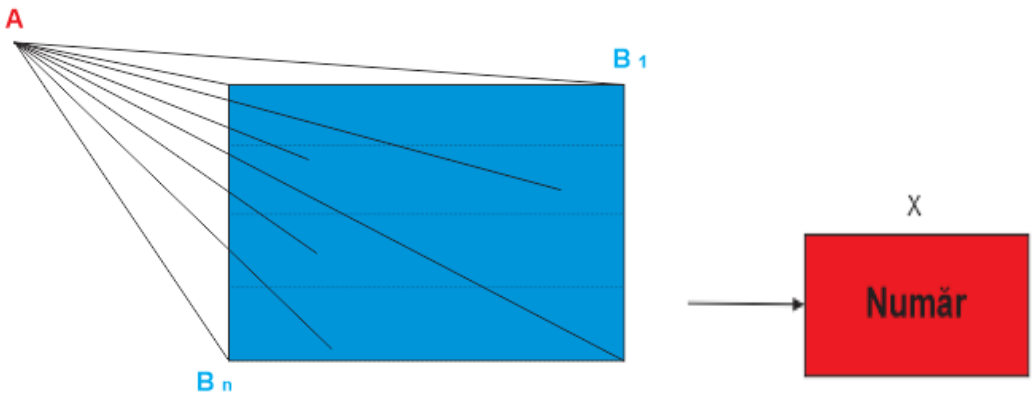


Figura 4. Modelul inițial grafic a Subconștiinței Artificiale (SA)

Cum funcționează Subconștiința la om?

Când întâlnim pe cineva pentru prima dată [1], vedem că momentan parcă avem deja de undeva o informație în creierul nostru despre această persoană. Ce informație este aceasta și de unde vine ea? Anume este informația ce vine din subconștiință.

După ce sumăm toate distanțele tuturor punctelor, vom primi un număr. Acest număr X va fi foarte mare Fig.4 și va interacționa cu baza de cunoștințe. Sistemul nostru va primi unele date și le va prelua prin programul nostru inteligent, care va folosi date din bază de cunoștințe.

Deci, din start o să avem un punct Fig. 4 de unde vom face calculul pentru obiectul studiat, care ar putea fi, lentile de cameră video sau cameră foto situate, de exemplu, pe un droid zburător.

Fiecare celulă poate fi împărțită în două, după aceea în patru, în opt și așa mai departe Fig. 5.

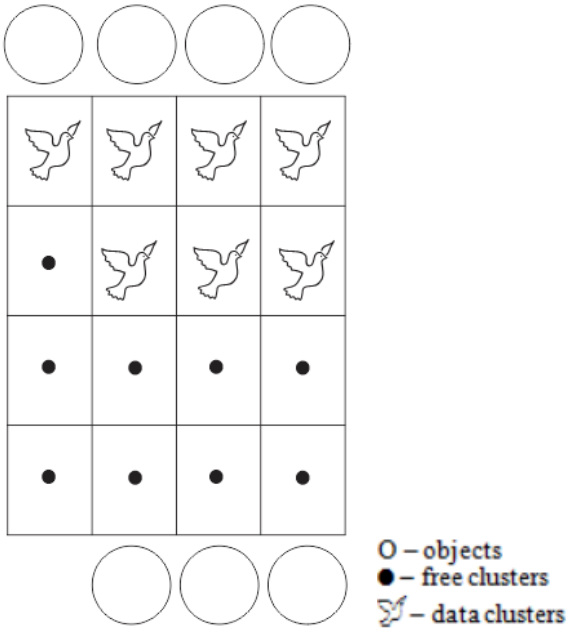


Figura 5. Aici este modelul unei imagini care este separată în multe obiecte cu date.

De exemplu, avem o poză. Vom împărți această imagine în mai multe părți. Fig. 5. Deci, fiecărei părți i se va atribui un singur obiect [4]. Și cum ne descurcăm cu alte părți ale imaginilor, care nu au obiecte?

În acest caz, am putea folosi o funcție din programarea OOP – moștenirea, unde fiecare obiect va procesa fiecare parte a imaginii. Moștenirea ne permite [5] să nu repetăm toate metodele din nou și din nou în fiecare obiect.

În Fig. 6 avem o schemă generală a sistemului nostru de subconștiință artificială, care este compusă din interacționarea bazei de cunoștințe, inteligenței artificiale, POO, programării procedurale, funcționale și celulare.

Figura 6. Sistem de Subconștiință Artificială

3. Mecanismul de luare a deciziilor

Toate lucrurile și cuvintele bune, frumoase, explicite din baza de cunoștințe sunt asociate cu numere pozitive. Cele negative sunt acsociate cu numerele negative. Cu cât numărul [6] va fi mai mic, decizia va tinde spre a nu fi efectuată Fig.7. Baza de cunoștință va avea indexarea fiecărui cuvânt anume cu un număr respectiv și specificator de semn.

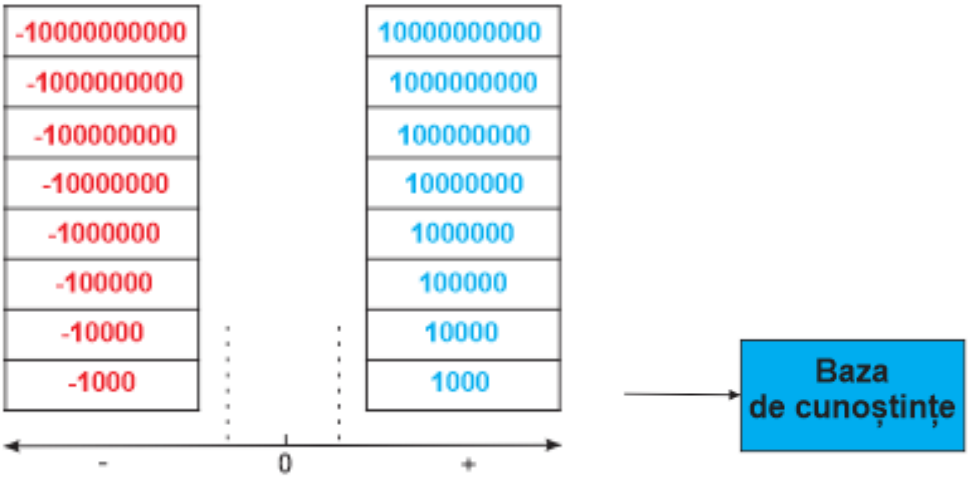


Figura 7 . Algoritmul de învățare a Subconștiinței Artificiale

Concluzii

Aici am adus aceste două definiții importante care au definit *Subconștiința Artificială*:

Subconștiința în computere - este atunci când deciziile sunt luate pe baza cunoștințelor ascunse și care intervin la necesitate.

Subconștiința în computere - suma tuturor distanțelor de la ochiul (sursa) artificială [1] până la toate punctele obiectului nostru analizat, care este trecută prin setul de informații care se află în baza de cunoștințe, formează decizia mai corectă a rezultatului cu privire la acest obiect.

Acesta este primul pas pentru a ajuta sistemul artificial [7] să ia decizii mai corecte și să găsească cea mai bună soluție eficientă a diferitelor sisteme tehnologice. Sper că *Subconștiința Artificială* se va folosi doar în scopuri pașnice.

Mulţumiri. Rezultatele prezentate în lucrare au fost realizate cu supotul tehnologic și metodologic oferit de DIIS UTM.

Referinţe:

1. ДЖОН КЕХО. Подсознание может всё. Попурри, 2005, 27-30p. ISBN: 985-483-447-6.
2. ТОНИ БЬЮЗЕН. Супер-память. Попурри, 2004, 47-52p. ISBN: 985-483-137-X.
3. ТОНИ БЬЮЗЕН И БАРРИ БЬЮЗЕН. Супер-мышление. Попурри, 2003, 27-39p. ISBN: 985-438-994-4.
4. PAPAZOGLOU, M.P., GEORGAKOPOULOS, D. Service-Orineted Computing. *In: Communications of the ACM, 2003/Vol. 46, No. 10*, pp. 25-28.
5. PAPAZOGLOU, M.P., HEUVEL, W-J. Service Orineted Architectures: Approaches, Technologies and Research isuues. *In: The VLDB Journal (2007) 16: 389-415*. DOI: 10.1007/s00778-007-0044-3.
6. ROSEN, M., LUBLINSKY, B., SMITH, K.T., BALCER, M.J. *Applied SOA: Aervice-Orineted Architecture and Design Strategies.* Wiley, 2008, 660p. ISBN: 978-0-470-22365-9.
7. DAUMEINGTS, G., MULLER, J., MOREL, G., VALLESPIR, B. *Enterprise Interoperability: New Challenges and Approaches*. Springer, 2007, 585p. ISBN: 978-1-84628-713-8.

WHAT IS SOCIAL ENGINEERING AND HOW TO PREVENT IT?

Alexandru TODOS

Departament of Software Engineering and Automatics, SI-221M, Faculty of Computers, Informatics and Microelectronics, Technical University of Moldova, Chisinau, Republic of Moldova

Corresponding author: Todos Alexandru, todos.alexandru1@gmail.com

Scientific coordinator: Rodica BULAI, university lector, Technical University of Moldova

Abstract. *When we think about cyber-security, most of us think about defending ourselves against hackers who use technological weaknesses to attack data networks. But there's another way into organizations that's taking advantage of human weakness. This is known as social engineering, which involves tricking someone into divulging information or enabling access to data networks.*

Keywords*: information security, privacy, security management, phising, vulnerability*

Introduction

Social engineering [1] is a cybersecurity attack that uses deception and manipulation to convince unsuspecting users to reveal confidential information about themselves such as social account credentials, banking credentials, etc.

The effectiveness of this attack does not necessarily need a high level of technical ability. Instead, social engineering preys on universal human psychological traits including civility, curiosity, credulity, avarice, carelessness, shyness, and apathy.

Social engineering can occur over the phone, through direct contact, or via the Internet. Attackers may seek to install malicious software on a victim's device, gaining full control over their computer and network, to achieve their goals of sabotage or theft.

Cybersecurity experts are concerned about social engineering attacks because no matter how robust the security stack is, users can still be tricked into providing their credentials to a hacker.

How does social engineering work?

Social engineering attacks rely on communication between attackers and victims. Attackers motivate users to compromise themselves, rather than using brute force methods to breach data.

The attack cycle consists of **preparation**, **infiltration**, **exploitation** and **disengagement**. Attackers collect information about victims through social media, telephone calls, email, text messages or other sources [2]. After that, they approach as reliable sources or authorities, use the information they have learned about victim to gain trust and persuade to give them sensitive information [3].

This process can take place in a single email or over months in a series of social media chats [4]. By studying the behaviour of the victim and playing on his weaknesses, the attacker can get the information he needs Thus, by studying the behaviour of the victim and playing on weaknesses, the attacker can get the information without writing a line of virus code.

Types of social engineering

Almost every type of cybersecurity attack has some kind of social engineering. Attackers that use social engineering often employ the following strategies [5]:

1. Phishing Attacks: attackers pretend to be a trusted institution or individual to persuade victim to expose personal data. Phishing scams are taking advantage of user trust. Methods used in phishing each have unique modes of delivery [6], including but not limited to:

Voice phishing – a sophisticated form of phishing attack. In this attack, a phone number is usually spoofed to appear legitimate. Attacker might disguise themself as someone trusted to trick victims into giving up sensitive information.

SMS phishing – a type of phishing attack that comes in the form of text messages. These types of attacks solicit immediate action from a victim, by including malicious links to click [7].

Email phishing – the most traditional type of phishing, using an email urging you to reply or follow-up by other means.

Search engine phishing – attempt to place links to fake websites at the top of search results. These may be paid ads or use legitimate optimization methods to manipulate search rankings. These links tempt you to travel to phishing websites.

2. Baiting Attacks: attacker leaves a malware-infected device, such as a USB drive, where someone is likely to find it. If a curious person picks up the device and plugs it into their own computer, they may infect their device with malware.
3. Physical Breach Attacks: attackers physically show up and pretend to be someone they are not to access restricted locations or information. These attacks take place in corporate settings, such as in enterprises, governments, or other organizations. Attackers pose as a representative of a reputable, well-known vendor for the business. Some attackers might even be ex-employees with grudges against their former employers. To avoid questioning, they provide an unremarkable but plausible identity.
4. Scareware Attacks: a form of malware used to frighten victim into taking an action. This malware uses alarming warnings that report fake infections or claim one of your accounts has been compromised. Usually, it appears in the form of pop-ups, claiming to help you remove a computer virus. As a result, scareware pushes victim to do thoughtless actions, like buying fraudulent cybersecurity software.

Why are people exposed to social engineering attacks?

Did you know that majority of social engineering attacks are successful because of human mistakes? [8] That is because of human emotions and psychological aspects that social engineers can use to their advantage. This includes emotions such as:

- Carelessness: most of us have unintentionally clicked a few links or opened an untrustworthy email attachment. The harm done may be trivial to severe, depending on how quickly we were able to stop such an act.
- Curiosity: victim mistakenly receives an email with intriguing content not meant for them. Despite potential risks, curiosity drives victim to download the attached file, only to discover it has a virus.
- Fear: according to Charles E. Lively, Jr. in the paper “Psychological-Based Social Engineering,” [9] attacks that play on fear are usually the most aggressive form of social engineering because it pressures the target to the point of making them feel anxious, stressed, and frightened. Participants in such attacks are more likely to provide money, intellectual property, or other information.
- Empathy and sympathy: when crises or natural disasters occur, it is impossible to escape the urge to provide aid. Nowadays, it's far simpler to go online, type your card information into a website accepting donations, and hit "Enter" because many of us are unlikely to be able to quickly board a plane and fly to damaged areas to help. Not all those websites are legitimate. The associated feelings of sympathy and empathy are used by social engineers to steal money from individuals who are truly in need and put it directly into their own pockets.

Prevention techniques

For all mobile and computer users, understanding how to avoid social engineering assaults is crucial. Here are some important ways to protect against all types of cyberattacks:

1. Avoid clicking on links in any emails or messages**.** Type URLs manually and ensure that they are official and legitimate.

2. Use multi-factor authentication. Protecting your accounts with more than simply a password is safer.
3. Use strong passwords. Your passwords should be complicated employing a variety of character kinds, such as symbols, numerals, and uppercase letters [10]. Use a password manager to store and remember all your passwords.
4. Avoid sharing personal information such as name of your school, pet or other personal details. You could be unknowingly exposing answers to your security questions or parts of your password.
5. Never let strangers connect to your primary Wi-Fi network, make access to a guest Wi-Fi connection available. This allows your main connection to remain secure and interception-free.
6. Use a VPN (Virtual Private Network). A virtual private network are services that give you a private, encrypted "tunnel" on any internet connection you use. Your connection is not only guarded from unwanted eyes, but your data is anonymized so it cannot be traced back to you via cookies or other means.
7. Keep all network-connected devices and services secure. Be sure to protect commonly overlooked devices like home network routers. Data breaches on these devices could fuel personalization for a social engineering scam.

Most businesses are aware of cyber-attacks and have invested heavily in security measures to reduce security threats. Though, with all that in place, in the digital world, there is still an element called human. To prevent successful social engineering attacks in companies, consider implementing the following measures:

1. Conduct regular security trainings and simulate social engineering attempts [11].
2. Increase spam filtering and implement security policies. For example, require multi-factor authentication for employees.
3. Frequently check critical parts of the company.
4. Periodically execute penetration tests and do internal audits [12].

Conclusion

Social engineering is one of the most effective ways of gaining access to secure systems and obtaining sensitive information yet requires minimal technical knowledge. Social engineering works against technical barriers and exploits human vulnerabilities to gain access. The best defence is to educate users on the techniques used by social engineers. Even so, a determined attacker with sufficient skill, resources and ultimately, luck, will be able to retrieve the information they are looking for. For this reason, organisations and individuals should have measures in place to respond to, and recover from, a successful attack.

Acknowledgment. I express my sincere gratitude to professor Bulai Rodica from the Technical University of Moldova for assistance and guidance provided during the preparation of this scientific report.

References

1. CASTELLUCCIO, MICHAEL. *Social Engineering 101*. Strategic Finance, vol. 84, no. 6, Dec. 2002, pp. 57+.
2. *Investigators at State Key Laboratory of Mathematical Engineering and Advanced Computing Describe Findings in Social Engineering (A Risk Analysis Framework for Social Engineering Attack Based on User Profiling).* Obesity, Fitness & Wellness Week, 13 Mar. 2021, p. 278.
3. JACKSON, RUSSELL A. *PULLING STRINGS: High-level hackers are using social engineering tactics to manipulate employees into giving up vital information.* Internal Auditor, vol. 75, no. 4, Aug. 2018

4. ALBLADI, S.M., WEIR, G.R.S. *Predicting individuals' vulnerability to social engineering in social networks.* Cybersecurity 3, 7 (2020).
5. *WHAT IS SOCIAL ENGINEERING?* Internal Auditor, vol. 75, no. 4, Aug. 2018
6. ALABDAN, RANA. *Phishing Attacks Survey: Types, Vectors, and Technical Approaches.* Future Internet, vol. 12, no. 10, Oct. 2020, pp. 1h+.
7. ALAZAB, MAMOUN, and RODERIC BROADHURST. *Spam and criminal activity.* Trends & Issues in Crime and Criminal Justice, no. 526, 21 Dec. 2016
8. ADAMS, ANNE, and MARTINA ANGELA SASSE. *Users are not the enemy.* Communications of the ACM, vol. 42, no. 12, Dec. 1999, p. 40.
9. CHARLES E. LIVELY, JR, GSEC Option 1 version 1.4, *Psychological-Based Social Engineering* (2003).
10. DEALVARE, A.M. *How crackers crack passwords or what passwords to avoid.* In Proceedings of Unix Security Workshop II. (Portland, 1990).
11. WANG, ZUOGUANG. *Social engineering in cybersecurity: a domain ontology and knowledge graph application examples.* Cybersecurity, vol. 4, no. 1, 2 Aug. 2021
12. PYZI., KEN. *Shutting the door on social engineering: internal audit can help organizations thwart efforts to manipulate employees to gain system access.* Internal Auditor, vol. 72, no. 5, Oct. 2015

SUBSECŢIA MICROELECTRONICĂ ŞI NANOTEHNOLOGII

MICROELECTRONICS AND NANOTECHNOLOGIES SUBSECTION

REALIZATION OF H2S GAS SENSOR USING SOL-GEL PROCESSED ZnFe2O4 THIN FILM

Rajat[1], Amit Kumar SHRINGI[2], Mahesh KUMAR[2,*]

[1]*Department of Physics, Indian Institute of Technology Jodhpur, Jodhpur-342037, (India)*
[2]*Department of Electrical Engineering, Indian Institute of Technology Jodhpur, Jodhpur-342037, (India)*

*Corresponding authors: Mahesh Kumar, mkumar@iitj.ac.in

Abstract: *The rapid growth in technology has led the convenience for society but it also inevitably brought many problems to the environment and different lives. Hydrogen Sulfide (H_2S) is one of the major pollutants considered toxic gas. Significant sources of H_2S gas are fossil fuels, natural gas production, and other chemical industries. The higher concentration of H_2S (>500 ppm) in breathing can cause unconsciousness or even death. Thus, selective detection of H_2S traces in the ambient atmosphere is highly desirable. In this study, we demonstrated low-cost and highly sensitive $ZnFe_2O_4$ nanostructured thin film with spinel structure using the Sol-gel method. The $ZnFe_2O_4$-based sensor device shows the highest sensitivity of 81% for 100ppm of H_2S concentration at 250°C. It also exhibits fast response and good selectivity. The work implies the potential application of sol-gel prepared $ZnFe_2O_4$ in H_2S gas detection at 250°C.*

Keywords: *$ZnFe_2O_4$, spinel ferrite, H_2S, operating temperature, selectivity, response.*

Introduction

$ZnFe_2O_4$ has a spinel ferrite structure with a unit cell consisting of an array of FCC O^{2-} anions with Zn^{2+} at tetrahedral interstices and Fe^{3+} at octahedral interstices formed of O^{2-}. $ZnFe_2O_4$ is a narrow band gap around 1.9 eV ternary metal oxide semiconductor. $ZnFe_2O_4$ detects target gas via a complex interaction of the surface gas-solid phase process. $ZnFe_2O_4$ is multifunctional material used in the gas sensing field, photocatalyst, Li-ion batteries, solar cells, and magnets. In the gas sensing field, an excellent exploration of properties has been demonstrated by $ZnFe_2O_4$ with its co-substances such as porous $ZnFe_2O_4$ NRs to acetone, RGO- $ZnFe_2O_4$ to ethanol, $ZnFe_2O_4$-NRs to Formaldehyde, porous $ZnFe_2O_4$ NSs to H_2S and so on.

The mechanism for gas sensing of $ZnFe_2O_4$ gas sensors is as follows: At the time of the surface gas-solid phase reaction process, O_2 molecules adsorbed on the surface of $ZnFe_2O_4$ and the sensor captures free electrons from the conduction band and the formation of O^{2-} anions occur shown in E. (1). Depending on operating temperature, oxygen is present in molecular (O_2^-) or atomic (O^- and O^{2-}) form. The dependency of oxygen forms on temperature is shown in Eq. (2) and Eq. (3) as follows:

$$O_2(gas) + e^- \rightleftharpoons O^{2-}(ads), T < 147℃ \quad (1)$$

$$O^{2-}(ads) + e^- \rightleftharpoons 2O^-(ads)\ 147°C < T < 397°C \quad (2)$$

$$O^-(ads) + e^- \rightleftharpoons O^{2-}(ads)\ T > 397°C \quad (3)$$

Due to unique ferrite crystal structure and high surface activation energy and by lowering the limit of detection with the coexistence of familiar substances makes $ZnFe_2O_4$ a good sensing material. ZnFe2O4 due to its low-cost synthesis, small size, low power consumption, and portability explores for the application of industrial problems and diagnostic clinical events. The redistribution of cations in the $ZnFe_2O_4$ structure changes its various properties which gives a new direction to gas sensing research. Doping suitable metal on $ZnFe_2O_4$ with restriction $MxZn_{1-x}Fe_2O_4$ can improve the sensing performance of $ZnFe_2O_4$ due to their good surface activity (M=Ni, Cu, Ca, etc.) Various methods have been used to improve the gas sensing properties and reduce the operating temperature of Metal Oxide Semiconductor nanomaterials like functionalization of surface and doping of noble gaseous atoms [1-2].

Wie Zhang et al [3]. reported, In Cu- $ZnFe_2O_4$, generally there is no effect of Copper doping on the growth process but it increased the dispersibility of products. Fen Liu et al [4]. reported, the sensor based on 0.125% G- $ZnFe_2O_4$ (180℃,10h) exhibits good selectivity and reproducibility to 10ppm acetone vapour at 275℃. X.Chu et al [5] reported ZnFe2O4- GQDs nanocomposite with 15 ml GQD suspension exhibited good response and good selectivity to acetone vapor at room temperature.

Methodology

$ZnFe_2O_4$ solution is prepared by dissolving 0.2 M and 0.4 M of $Zn(NO_3)_2.6H_2O$ and $Fe(NO_3)_3.9H_2O$ respectively in 5ml water. This mixed solution was magnetically stirred for 2 hours and then 0.6 M of Citric acid was added to the prepared solution and then the final solution was stirred for 20 hours at room temperature. Next, 150 µL Acetylacetone and 150 µL Triton X- 100 were added. Then, cleaned Si was spin-coated at 1500 rpm for 20 seconds five times with the above-prepared solution. Afterward, it was annealed at 700°C under a tubular furnace for 3 hours.

Then, an IDE (Interdigitated Electrode) pattern with Au-Cr with a thickness of 200 nm-20 nm was created on a thin film. Gas sensor performance was measured by a sensing system. By measuring the electrical resistance or current of a gas sensor by applying 3 V of voltage between the two electrodes. The gas sensor resistance is first measured in air, and then a certain amount of H_2S gas injected into the test chamber by injection or any other controlled apparatus. All gas sensing tests were performed at about 250 °C. The response to H_2S was defined as (Ig-Ia)/Ig, where I_a and I_g were the currents of the gas sensors in the air and H_2S respectively. The response and recovery time of a sensor is defined as the times taken to achieve up to 90% change of the total resistance in the case of adsorption and desorption of injected gas, respectively.

Results and Discussion

XRD and SEM images tell us about the crystal structure and morphology of the sample as shown in Fig. 1(a) and Fig. 1(b) respectively.

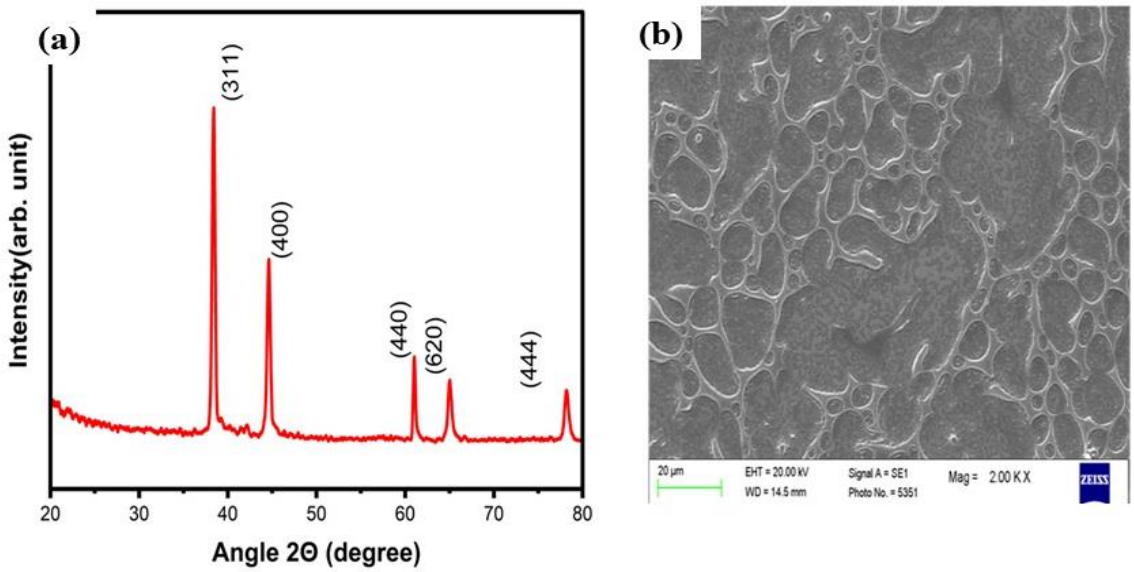


Figure 1. (a) XRD of $ZnFe_2O_4$ thin film (b) SEM image of $ZnFe_2O_4$ thin film

UV visible study is also being done to determine the band gap of the samples as shown in Fig. 2(a) and IV characteristics as shown in Fig. 2(b).

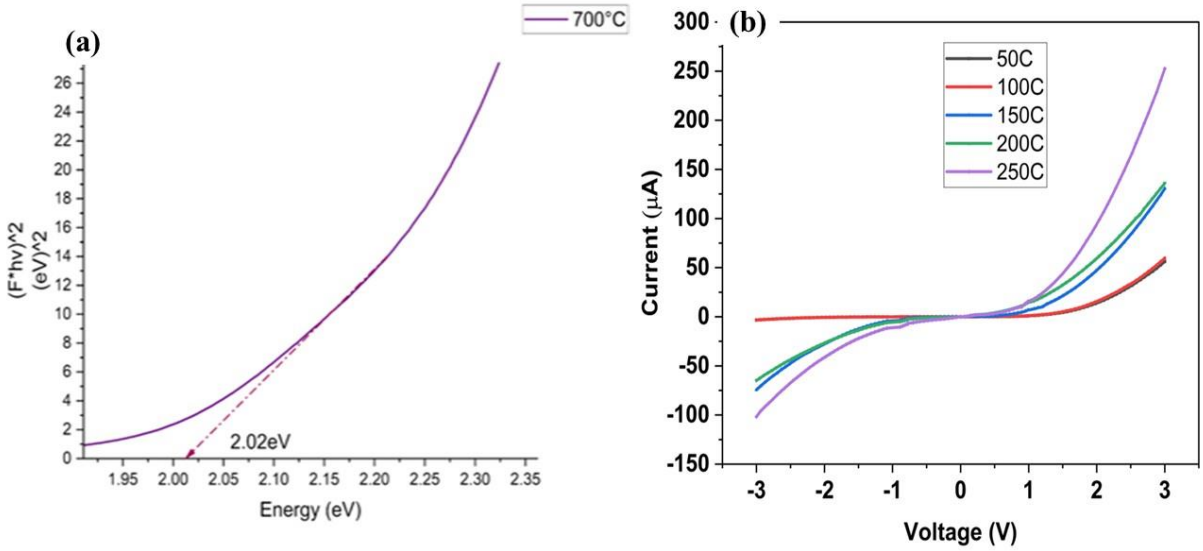


Figure 2. (a) UV Vis graph (b) IV Characteristics.

Gas Sensing Characteristics

Gas sensing selectivity can be figured out by the above-mentioned graph of bare ZFO which shows good selectivity for H_2S gas as for 100ppm gas at 250 ° C temperature. CO, NO_2, NH_3, SO_2, and H_2S show relative responses as 31%, 35%, 36%, 37%, and 57% respectively. Sensing response and selectivity as shown in Fig. 5 and Fig. 6 respectively.

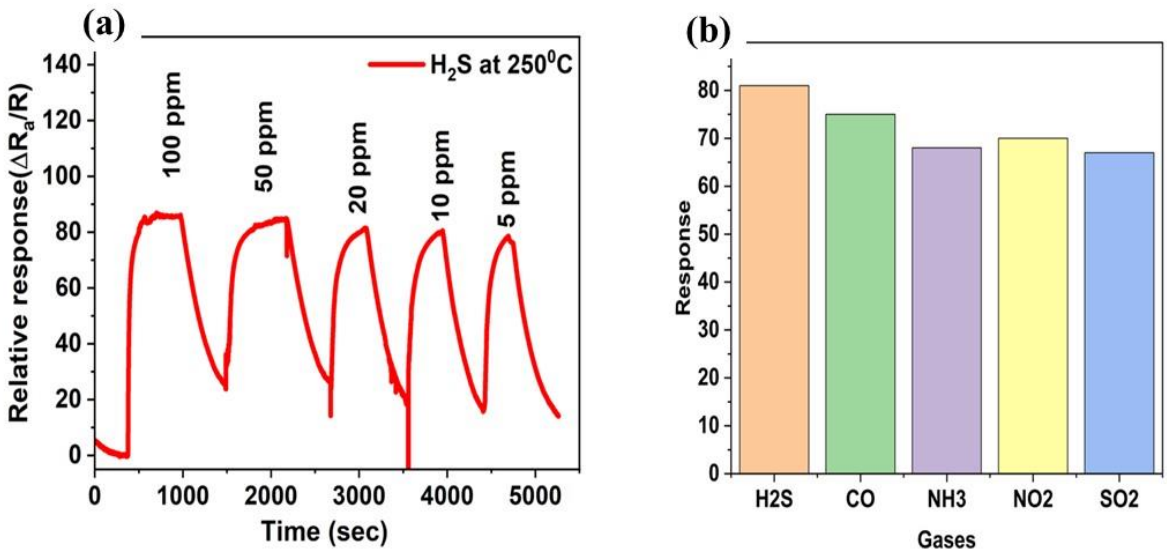


Figure 3. (a) Relative response (b) selectivity of different gases.

Conclusions

$ZnFe_2O_4$ is an n-type spinel ferrite structure having an energy band gap of around 1.9 eV confirmed by UV-visible spectroscopy. The above-studied thin film sample of $ZnFe_2O_4$ has the morphology of nanofibres confirmed by SEM characterization technique with good elemental composition confirmed by EDAX. Material that is pure without any other impurity is also confirmed by XRD characterization having all major peaks of $ZnFe_2O_4$ with a signature peak of it as (311) also gives firm evidence of $ZnFe_2O_4$ presence. Electrical properties and chemical properties are also characterized by gas sensing characteristics as discussed above with good results of sensing selectively 100ppm H_2S gas at 250° C.

References

1. Y. CAO, D. JIA, P. HU, and R. WANG, One-step room-temperature solid-phase synthesis of $ZnFe_2O_4$ nanomaterials and its excellent gas-sensing property. *Ceramics International*, 2013, 39(3), pp. 2989–2994.
2. H. MEI, S. ZHOU, M. LU, and L. CHENG, Tetrapod-like ZnO/$ZnFe_2O_4$ based heterostructure for enhanced ethanol detection. *Journal of Alloys and Compounds*, 2020, 840, pp. 155583, 2020.
3. WEI ZHANG, YANBAI SHEN, JIN ZHANG, HONGSHAN BI, SIKAI ZHAO, PENGFEI Z HOU, CONG HA, DEZHOU WEI, NA CHENG, Low-temperature H_2S sensing performance of Cu-doped $ZnFe_2O_4$ nanoparticles with spinel structure, *Applied Surface Science*, 2019, 470, pp. 580-590.
4. F. LIU, X. CHU, Y. DONG, W. ZHANG, W. SUN, L. SHEN, Acetone gas sensors based on graphene-$ZnFe_2O_4$ composite prepared by solvothermal method. *Sensors and Actuators, B: Chemical*, 2013, 188, pp. 469–474.
5. X. CHU, P. DAI, S. LIANG, A. BHATTACHARYA, Y. DONG, M. EPIFANI, The acetone sensing properties of $ZnFe_2O_4$-graphene quantum dots (GQDs) nanocomposites at room temperature. *Physica E: Low-Dimensional Systems and Nanostructures*, 2019, 106, pp. 326–333.

PROPRIETĂȚILE NANOBAGHETELOR DE ZnO:Sn

Tudor ZADOROJNEAC[1], Dinu LITRA[1], Philipp SCHADTE[2], Cristian LUPAN[1]

[1]*Universitatea Tehnică a Moldovei, Facultatea Calculatoare Informatică și Microelectronică, Departamentul Microelectronică și Inginerie Biomedicală,, Centrul de Nanotehnologii și Nanosenzori, Chişinau, Republica Moldova,*
[2]*Catedra pentru Materiale Avansate, Departamentul pentru Ştiința Materialelor, Universitatea Kiel, Kaiserstr. 2, D-24143, Kiel, Germania,*

*Autorul corespondent: Cristian Lupan, cristian.lupan@mib.utm.md

Rezumat. *În această lucrare, sunt prezentate rezultatele obținute în urma investigațiilor efectuate asupra nanobaghetelor de ZnO dopate cu Sn obținute prin metodă hidrotermală. A fost utilizată microscopia electronică cu scanare cu scopul analizei morfologice a nanostructurilor de oxid de Zn dopat cu Sn, spectroscopia cu raze X cu dispersie de energie cu scopul determinării componenței chimice și difracția cu raze X pentru analiza structurală. Dopajul cu Sn este considerat a fi factorul cheie care modifică proprietățile morfologice și senzoriale pentru utilizarea în aplicații în fotodetectoare miniaturizate, diode luminiscente, sursă de lumină laser și senzori de gaze.*

Cuvinte cheie*: oxid de zinc, Sn, morfologie, dopare, hidrotermală.*

Introducere

În ultimul deceniu nanotehnologia a generat multă atenție în domeniul științei și ingineriei. Una dintre aplicațiile interesante au fost senzorii de oxid de metal nanostructurat care acționează ca nas electronic pentru a detecta compuși organici volatili toxici (VOC). Dopajul și funcționalizarea cu metale nobile s-a dovedit experimental a fi modalități eficiente de îmbunătățire a proprietăților de detectare, în principal prin adaptarea energiei de activare, adsorbție și desorbție a ionilor de oxigen și viteze de reacție. În majoritatea cazurilor, modificarea suprafeței poate fi benefică și pentru transportul de electroni. Prin urmare, combinația dintre structura ierarhică și suprafața modificată ar putea îmbunătăți și mai mult proprietățile de detectare ale senzorilor în baza oxizilor metalici [1]. Rezultatele obținute au demonstrat că senzorii în baza oxizilor metalici ZnO au o sensibilitate înaltă față de gazele cercetate. În plus, Sn s-a dovedit a fi un dopant eficient care a îmbunătățit proprietățile de detectare ale senzorilor de gaz ZnO [2]. Au fost investigate proprietățile morfologice utilizând microscopia electronică cu scanare (SEM), difracția cu raze X (XRD) și spectroscopia cu dispersie de energie (EDX).

Proprietățile excelente a oxizului de Zn, cum ar fi o bandă interzisă largă (3,37 eV la 300 K), o energie mare de legare a excitonului (60 meV) și un punct de topire mare de 1975 °C asigură aplicații în fotodetectoare miniaturizate, diode luminiscente, sursă de lumină laser și nanosenzori în medii dure. Un alt domeniu important în care sunt utilizate aceste nanostructuri este celulele solare organice care au fost propuse ca alternative cu costuri reduse față de panourile anorganice, în ciuda eficienței lor mai scăzute [3]. Pentru ZnO, adăugarea de impurități induce adesea modificări dramatice ale proprietăților lor electrice și optice, ceea ce sugerează noi aplicații. Sb și Ag acești dopanți complecși, introduși prin tehnici de sinteză, pot defini tipul de conductivitate și proprietățile optice și electrice ale ZnO. Metoda de creștere a soluției, pe de altă parte, este o metodă ușoară, cu temperatură scăzută și cu cost redus, care poate produce material ZnO stoichiometric și de înaltă calitate [4].

Partea experimentală

Nanobaghetele date au fost obținute prin metoda hidrotermală. În calitate de substrat s-a folosit plăcuțe de sticlă sau de Si. Reactorul a fost montat pe o placă fierbinte la o temperatură fixă în intervalul 60-95 °C, iar reacția a fost lăsată să se desfășoare pentru diferite durate de timp între 10 și 60 de minute. Nanocristalele de ZnO s-au format la o valoare a pH-ului de 10-11. După un interval de timp

predeterminat la 60–95 °C, alimentarea plitei de încălzire a fost oprită. Reactorul a fost lăsat pe placa fierbinte timp de 30 de minute pentru a se răci la 40°C. În cele din urmă, substraturile au fost scufundate și clătite în apă deionizată și apoi probele au fost uscate în aer la 150 °C timp de 5 minute [5].

Rezultate și discuții

Tehnica microscopiei electronice este una dintre cele mai utilizate tehnici utilizate în domeniul nanotehnologiilor, care permit vizualizarea suprafeței specimenului. Această tehnică utilizează interacțiunea dintre electroni cu specimenul pentru generarea de imagini topologice.

În figura 1 sunt reprezentate imaginile SEM a nanostructurilor de ZnO:Sn, unde se observă că pe suprafața substratului nanostructurile sunt distribuite uniform și nu sunt prezente suprafețe unde lipesc acestea (figura 1a). La mărire se observă că majoritatea nanostructurilor au o forma hexagonală (figura 1b), care nu este hexagonală perfectă dar modificată, datorită dopării cu Sn, mai îndetaliat acest lucru poate fi observat (figura 1c,d) la scara de 100 și 20 nm. În acest caz forma hexagonală imperfectă a nanostructurilor care se obține datorită dopării cu Sn este mai evidențiată în comparație cu asemenea nanostructuri raportate de alți autori [6,7].

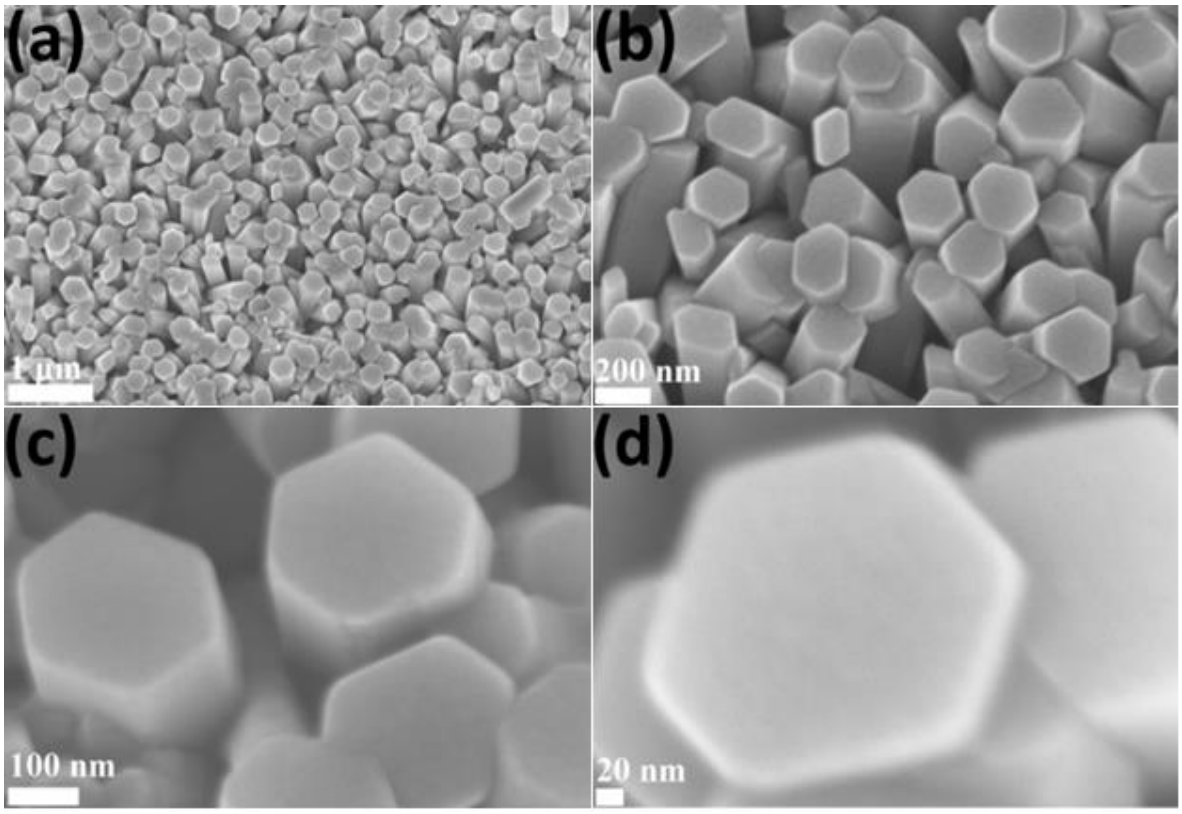


Figura 1. Imagini SEM a nanobaghetelor de ZnO:Sn obținute prin metoda hidrotermală: (a), (b) – vedere generală a nanorozilor de ZnO:Sn la scara de 1 µm și 200 nm. (c), (d) – prezentarea nanorozilor individuali de ZnO:Sn la scara de 100 și 20 nm.

Structura cristalului a fost investigată prin difracție de raze X (XRD). În figura 2 este prezentat modelul XRD al ZnO:Sn în intervalul de 20-80° 2θ, observând vârfurile ZnO și un singur vârf Zn_2SnO_4. Nu au fost detectate vârfuri de SnO_2 conform, posibil din cauza concentrației scăzute de dopaj, observate în studiul EDX. Se observă o cristalinitate ridicată a nanostructurilor ZnO:Sn datorită raportului semnal/zgomot bun al vârfurilor ZnO, cu cel mai înalt vârf la 34,3° pentru vârful (0002) ZnO, ceea ce înseamnă că nanorodurile sunt orientate preferenţial de-a lungul axei *c* [8].

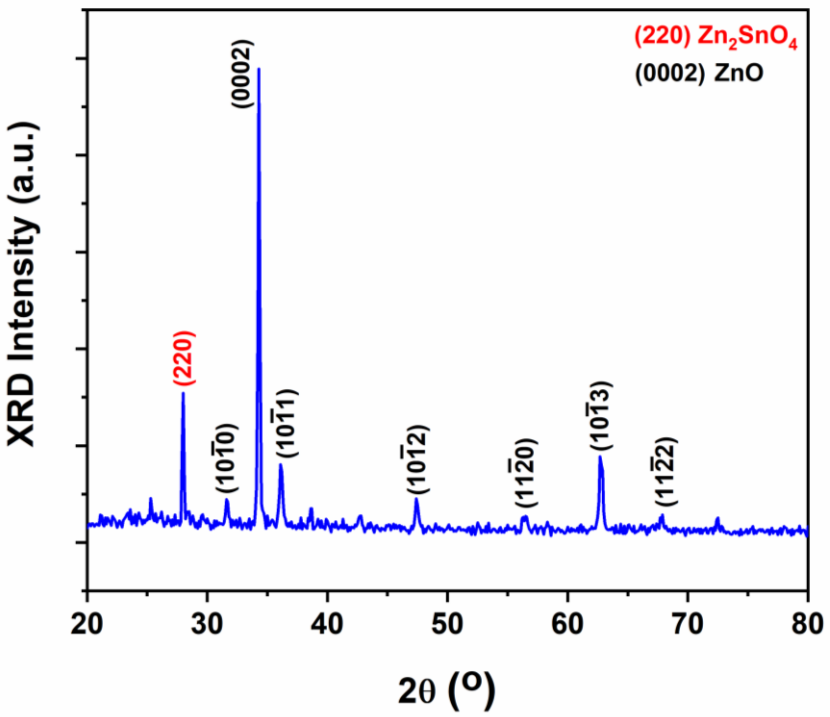


Figura 2. Difractograma XRD a ZnO:Sn crescut prin metoda hidrotermală

Proprietățile chimice au fost studiate folosind EDX. Spectroscopia EDX este implicată în detectarea compoziției elementare a substanței prin utilizarea microscopului electronic cu scanare. EDX este capabil să detecteze elemente care au un număr atomic mai mare decât borul și aceste elemente pot fi detectate la o concentrație de cel puțin 0,01%. Aplicarea EDX include evaluarea și identificarea materialelor, identificarea contaminării [9].

În tabelul 1 sunt prezentate elementele și at% ale acestora care au fost detectate în timpul studiului EDX, observându-se o concentrație de dopaj de 0,24% în aceste nanostructuri.

Tabelul 1.

Elemente detectate în timpul mapării EDX a nanobaghetelor ZnO:Sn

Element	Atomic %
O	49.79
Zn	49.97
Sn	0.24
Total	100.00

Maparea EDX este prezentată în figura 2, unde s-au observat elemente chimice de pe suprafața filmelor. Elementele individuale Zn, O și Sn sunt prezentate în figurile 3b,3c și 3d, observând că Sn este bine distribuit la suprafață, ceea ce înseamnă că nanorodurile ZnO:Sn sunt dopate uniform.

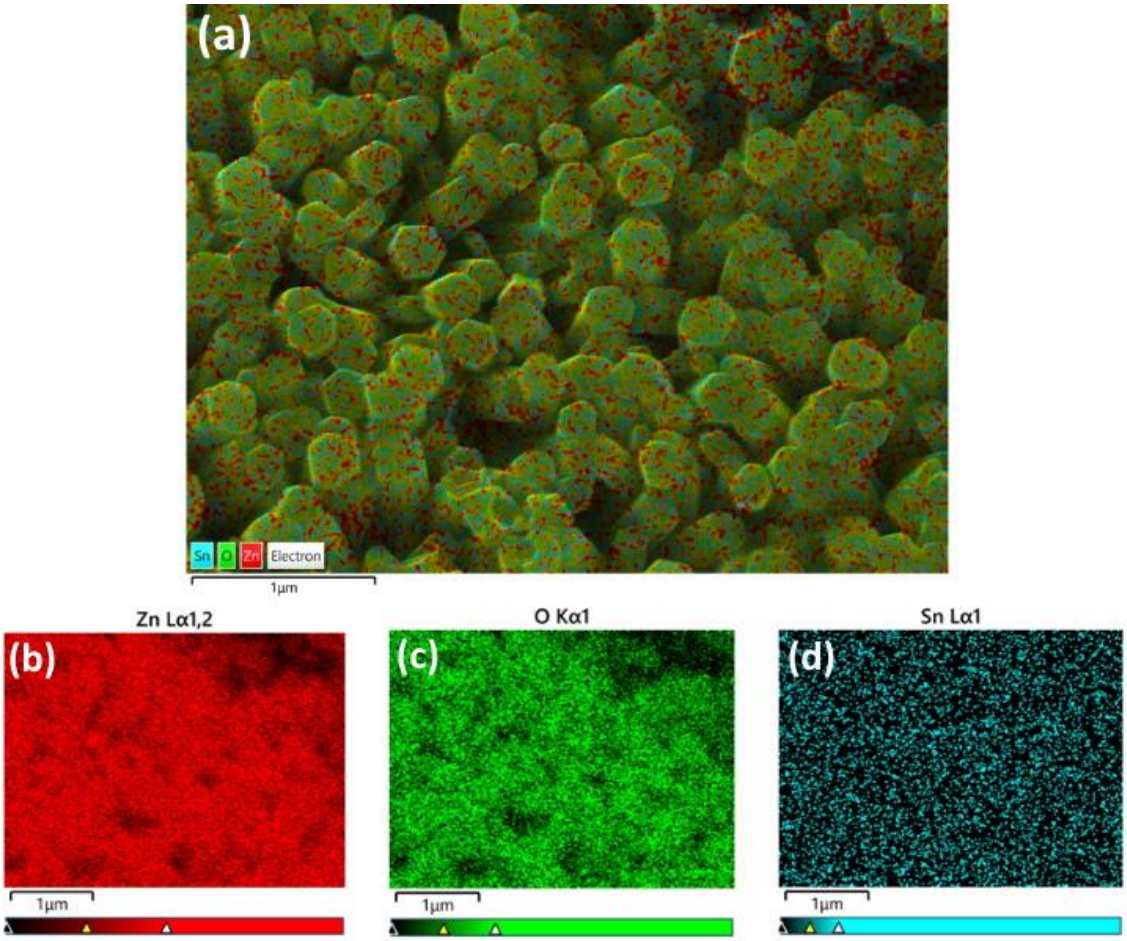


Figura 3. (a) Maparea EDX a tuturor elementelor din nanostructuri. Maparea EDX pentru (b) Zn, (c) O și (d) Sn.

Concluzii

În această lucrare sunt prezentate rezultatele obținute în urma cercetărilor asupra nanobaghetelor de ZnO dopate cu Sn obținute prin metoda hidrotermală. Obținerea imaginilor SEM a facut posibil analiza suprafeței nanobaghetelor la o scară de 20 nm astfel în același timp și compararea rezultatelor cu alți autori, de unde sa evidențiat cateva aspecte pozitive. Analiza difracției cu raze X a arătat o cristalinitate ridicată a nanostructurilor ZnO:Sn cu cel mai înalt vârf la 34,3° pentru vârful (0002) ZnO. Analiza a arătat o concentrație de dopaj de 0,24% de Sn în aceste nanostructuri ceia ce poate fi observat în tabelul 1. Din maparea EDX s-a observat repartizarea uniformă a dopajului de Sn.

Mulţumiri. Autorii sunt recunoscători Universității Tehnice din Moldova, Centrul de Nanotehnologii și Nanosenzori, în special Prof. Univ. Dr. Hab. O. Lupan pentru suport. This paper was partially supported by the Technical University of Moldova and the ANCD-NARD Grant No. 20.80009.5007.09 at TUM.

Referinţe

1. Mingshui YAO, Fei DING, Yuebin CAO, Peng HU, Junmei FAN, Chen LU, Fangli YUAN, Changyong SHI, Yunfa CHEN. "Sn doped ZnO layered porous nanocrystals with hierarchical structures and modified surfaces for gas sensors." *Sensors and Actuators B: Chemical* 201 (2014): 255-265.
2. F. PARAGUAY D, M. MIKI-YOSHIDA, J. MORALES, J. SOLIS, W. ESTRADA L. Influence of Al, In, Cu, Fe and Sn dopants on the response of thin film ZnO gas sensor to ethanol vapour. Thin Solid Films, 373 (2000) 137-140.

3. F.Z. BEDIA, A. BEDIA, M. AILLERIE, N. MALOUFI, B. BENYOUCEF. "Structural, optical and electrical properties of Sn-doped zinc oxide transparent films interesting for organic solar cells (OSCs)." *Energy procedia* 74 (2015): 539-546.
4. Oleg LUPAN, Lee CHOW, Luis K. ONO, Beatriz Roldan CUENYA, Guangyu CHAI, Hani KHALLAF, Sanghoon PARK, and Alfons SCHULTE. "Synthesis and characterization of Ag- or Sb-doped ZnO nanorods by a facile hydrothermal route." *The Journal of Physical Chemistry C* 114.29 (2010): 12401-12408.
5. D. POLSONGKRAM, P. CHAMNINOK, S. PUKIRD, L. CHOW, O. LUPAN, G. CHAI, H. KHALLAF, S. PARK, A. SCHULTE. "Effect of synthesis conditions on the growth of ZnO nanorods via hydrothermal method." *Physica B: Condensed Matter* 403.19-20 (2008): 3713-3717.
6. Jian-Hui SUN, Shu-Ying DONG, Jing-Lan FENG, Xiao-Jing YIN, Xiao-Chuan ZHAO. "Enhanced sunlight photocatalytic performance of Sn-doped ZnO for Methylene Blue degradation." *Journal of Molecular Catalysis A: Chemical* 335.1-2 (2011): 145-150.
7. Changle WU, Li SHEN, Huaguang YU, Qingli HUANG, Yong Cai ZHANG. "Synthesis of Sn-doped ZnO nanorods and their photocatalytic properties." *Materials Research Bulletin* 46.7 (2011): 1107-1112.
8. Cristian LUPAN, Abhishek Kumar MISHRA, Niklas WOLFF, Jonas DREWES, Helge KRUGER, Alexander VAHL, Oleg LUPAN, Thierry PAUPORTÉ, Bruno VIANA, Lorenz KIENLE, Rainer ADELUNG, Nora H de LEEUW, and Sandra HANSEN. Nanosensors Based on a Single ZnO: Eu Nanowire for Hydrogen Gas Sensing. ACS Applied Materials & Interfaces, (2022) 14(36), 41196-41207.
9. M. Abd MUTALIB, M.A. RAHMAN, M.H.D. OTHMAN, A.F. ISMAIL, J. JAAFAR. "Scanning electron microscopy (SEM) and energy-dispersive X-ray (EDX) spectroscopy." *Membrane characterization*. Elsevier, 2017. 161-179.

EFECTUL DE PROXIMITATE ÎN MICROELECTRONICA SUPRACONDUCTOARE

Vladimir BOIAN

Institutul de Inginerie Electronică și Nanotehnologie "D. Gitu", UTM, Chisinau, Republica Moldova

Rezumat.– *La momentul actual a minituarizării dispozitivelor microelectronice se înregistrează noi rezultate datorită necesităților în acest domeniu, al supraconductibilității și al nanotehnologiei. Un dispozitiv principal a microelectronicii de mare viteză și de mică putere este supapa de spin, care este alcătuită din straturi supraconductoare nanometrice, cum ar fi niobiu și cobaltul. Aceste două ordine – supraconductibilitatea și feromagnetismul, care la prima vedere au tangente diametral opuse și în stare naturală nu se află practic niciodată unul lângă celălalt - datorită stării: Larkin - Ovchinnikov - Fulde - Ferrell (LOFF), demonstrează fenomene cuantice destul de captivante pentru dezvoltarea lor ulterioară cu aplicații în: tehnologia informatică, chimie, biologie, farmacologie, intelectul artificial etc. În acest context de idei prezentăm cercetările privind structurile hibride Nb/Co cu proprietăți supraconductoare și determinarea grosimii critice a stratului supraconductor de Nb în contact cu Co.*

Cuvinte chee: *efectul de proximitate supraconductor/ferromagnet, împerecherea FFLO, grosimea critică*.

Introducere

De la lucrările lui Ginzburg [1], problema coexistenței a două ordine cu rază lungă de acțiune - supraconductibilitatea (S) și feromagnetismul (F) - a fost intens discutată. Ginzburg a ajuns la concluzia că cele două ordine antagoniste nu pot coexista într – un material omogen, deoarece supraconductibilitatea necesită ca electronii de conducție să formeze perechi Cooper, adică perechi de electroni cu spini antiparaleli, în timp ce feromagnetismul forțează spinii electronilor să se alinieze paralel. În cazul în care supraconductibilitatea și feromagnetismul nu pot exista într – un material omogen, acestea pot a fi separate spațial la scară nanometrică, formând un material stratificat artificial. Acest scenariu a fost realizat în supraconductori magnetici multicomponenți [2] și în bistraturi și materiale stratificate S/F create artificial [3]. Acest din urmă sistem are avantajul că grosimea și/sau succesiunea straturilor S și F pot fi modificate în timpul fabricării. Garifianov și alț [4]. a sugerat utilizarea unor perechi de metale imiscibile, cum ar fi Pb și Fe, pentru a evita problema difuziei. Cu toate acestea, o analiză detaliată a arătat [5] că, în acest caz, transparența interfeței S/F este redusă datorită barierei de potențial electrostatic (deplasare de bandă) creată prin ajustarea potențialului electrochimic al metalelor aflate în contact.

Rezultate și discuții

Perechile S/F nu ar trebui să fie formate din metale imiscibile pentru a evita formarea de insule din cauza lipsei de umezire la interfața lor. Mai degrabă, ar trebui utilizate metale cu solubilitate limitată și cu intervale înguste de formare a compușilor intermetalici.

Tipul de substrat, calitatea suprafeței și regimurile de creștere a peliculei trebuie să asigure că rugozitatea interfeței stratului F este cât mai mică posibil în comparație cu grosimea stratului F. În acest sens au fost efectuate o serie de modelări matematice pe calculator cu scop de a scoate în evidență parametri tehnologici ce asigură calitatea interfețelor dintre supraconductori și fieromagnetici în cazul nostru.

Procesele de formare și structura sistemelor multistrat pentru spintronică au fost studiate prin metoda dinamicii moleculare [6,7]. Dinamica moleculară descrie mișcarea fiecărui atom de nanosistem la un anumit moment de timp, prin urmare este posibilă reproducerea evoluției detaliate

a nanoparticulelor și a proprietăților acestora. Baza metodei este ecuațiile mișcării tuturor atomilor, completate de condițiile inițiale sub forma coordonatelor și vitezei atomilor:

$$m_i \frac{d^2\mathbf{r}}{dt^2} = -\frac{\P U(\mathbf{r})}{\P \mathbf{r}_i} + \mathbf{F}_{ex}, \quad \mathbf{r}_i(t_0) = \mathbf{r}_{i0}, \quad \frac{d\mathbf{r}_i(t_0)}{dt} = \mathbf{V}_{i0}, \quad i = 1,\text{K } N \qquad \textbf{(1)}$$

unde N - este numărul de atomi care au format nanosistemul, m_i - este masa atomului - i; $\mathbf{r}_{i0}, \mathbf{r}_i(t)$ - sunt vectorul de rază inițială și respectiv al atomului – i , respectiv; $U(\mathbf{r})$ - este potențialul sistemului, care depinde de poziția relativă a tuturor atomilor; $\mathbf{V}_{i0},, \mathbf{V}_i(t)$ - sunt viteza inițială și, respectiv, cea a celui de – al i - atom; $\mathbf{r}(t) = \{\mathbf{r}_1(t), \mathbf{r}_2(t), .., \mathbf{r}_K(t)\}$ - arată dependența de amplasare a întregului sistem de atomi; $\mathbf{F}_{ex}$ - este forța externă, reflectă interacțiunea nanosistemului cu mediul extern, inclusiv responsabilă pentru ajustarea energiei pentru a menține o temperatură constantă.

Metoda dinamicii moleculare se bazează pe conceptul de potențial, care este responsabil pentru natura și caracterul interacțiunilor atomilor nanosistemului. Există multe opțiuni posibile pentru tipul de potențial, dar recent datorită acurateței, câmpurile de forță cu multe particule au câștigat o mare popularitate. În această lucrare, am folosit potențialul metodei modificate a atomului încorporat - **MEAM** (**M**odified **E**mbedded **A**tom **M**ethod). Metoda modificată a atomului scufundat se bazează pe teoria funcțională a densității - DFT (Density Functional Theory). În această metodă, potențialul rezultat al nanosistemului este reprezentat ca suma contribuțiilor energiei atomilor individuali, iar contribuțiile interacțiunilor de perechi și cu mai multe elemente sunt considerate separat.

Ca modul de calcul al programului pentru cercetare teoretică, a fost utilizat complexul de calcul **LAMMPS** (**L**arge-scale **A**tomic / **M**olecular **M**assively **P**arallel **S**imulator) [8].

O serie de experimente pe calculator în care a fost studiată formarea de nanosisteme multistrat Nb - Co în intervalul de temperatură 300 - 800 K, pentru temperaturi ale substratului de 300, 500 și respectiv 800 K. Rezultatele simulării sunt prezentate în Figura 2 sub forma unui grafic compozițional procentual al acestui nanosistem. Calculele au arătat că temperatura substratului afectează semnificativ formarea structurii nanosistemului. O creștere a temperaturii duce la o creștere a grosimii totale a nanosistemului (la 800 K, această valoare a crescut cu 0,3 nm comparativ cu o temperatură de 300 K). Regiunea penetrării reciproce a atomilor de Nb în straturile sistemului constând din Co, și invers, este de asemenea în creștere, ceea ce se vede clar în graficele din figura 1:

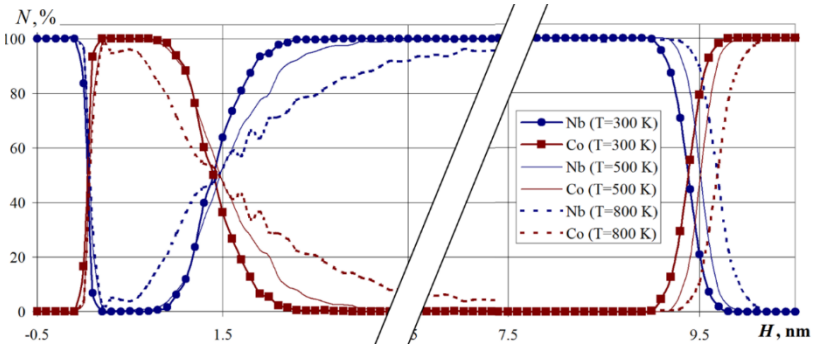


Figura 1. Compoziția procentuală a nanosistemului multistrat Nb - Co format la o temperatură a substratului de 300, 500 și, respectiv, 800 K. Parametrul N este procentul de cobalt și niobiu din straturile orizontale de nanofilme.

Măsurarea grosimii și rugozității straturilor F ar trebui să ofere date corecte și fiabile într – un interval de grosime de aproximativ (1 nm).

Pentru structurile hibride S/F (Nb/Co), persistă următoarele aspecte:

a) în procesul de pulverizare magnetronică trebuie exclusă posibilitatea formării insulare a materialului,
b) energia nivelelor Fermi trebuie să aibă aproximativ aceeași valoare,
c) condiții de vid înalt.

Pentru a determina grosimea critică a Niobiului în contact cu Cobaltul, a fost acoperită o probă de 80mm * 10mm, care a fost plasată asimetric față de ținta magnetronului, iar stratul supraconductor (Nb) a fost depus sub forma unei pene ca în reprezentarea de mai jos:

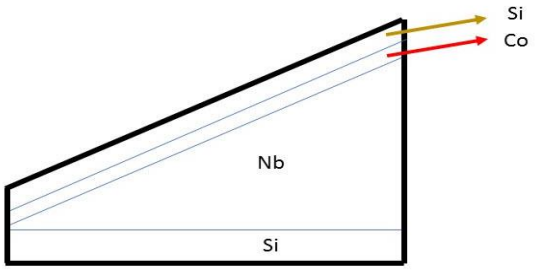


Figura 2. Diagrama de profil a structurii pulverizate cu ajutorul magnetronului Z – 400

Structura de dimensiunile: 80 mm * 10 mm este tăiată în 20 de bucăți. Fiecare dintre aceste 20 de bucăți este măsurată prin metoda celor 4 sonde și lipită pe placa de circuit imprimat pentru a fi introdusă în criostat. În urma experimentului s – a observat că, la o grosime constantă de 2,5 nm a Cobaltului, cuplat în proximitate cu Niobiu, odată cu scăderea grosimii supraconductorului – scade și temperatura critică a structurii hibride. La o grosime mai mică de 17,5 nm de Niobiu cuplat în proximitate cu Cobalt, a cărui temperatură critică este de 1,28 K, supraconductibilitatea dispare. Rezultatele sunt prezentate în figura de mai jos:

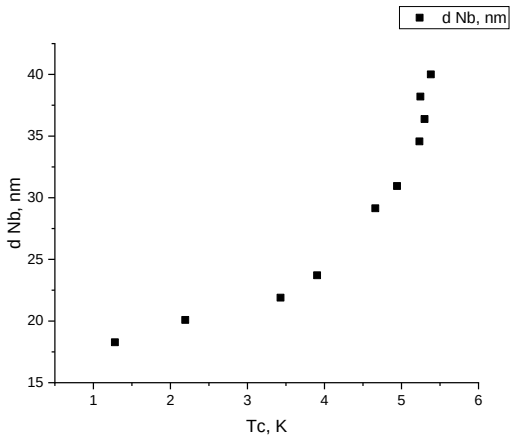


Figura 3. Dependența temperaturii critice de grosimea Niobiului

Concluzie:

Am investigat proprietățile supraconductoare ale bistraturilor de Nb/Co preparate prin pulverizare magnetronică în curent continuu pe substraturi de Si. Calitatea filmelor a fost caracterizată prin analiza difracției de raze X cu unghi mic. Parametrii fizici obținuți prin adaptarea teoriei proximității S/F la experiment se corelează bine cu proprietățile materiale ale perechii S/F, precum și cu datele din literatura de specialitate privind efectul de proximitate S/F în studiile altor perechi de materiale în contact. Rezultatele sunt interpretate cu succes în termenii Fulde – Ferrell – Larkin – Ovchinnikov (FFLO), cum ar fi cuplarea supraconductoare neomogenă într – un strat feromagnetic de Co.

Referinţe:

1. V. GINZBURG, Ferromagnetic Superconductors. *Teoret. Fiz.* 31, 202 (1956) [Sov. Phys. JETP 4, 153 (1956)].
2. M. B. MAPLE and Ø. FISCHER (eds.), Superconductivity in Ternary Compounds I and II (*Springer-Verlag*, Berlin, 1982). DOI: 10.1007/978-1-4899-3768-1_1
3. CHIEN, C. L. & REICH, D. H. Proximity effects in superconducting/magnetic multilayers. *J. Magn. Magn. Mater.* 200, 83-94 (1999)
4. N.N. GARIFYANOV, YU.V. GORYUNOV, TH. M. UHGE, L. LAZAR, G.G. KHALIULLIN, K. WESTERHOLT, I.A. GARIFULLIN, H. ZABEL, Proximity effect in Fe/Pb/Fe trilayers, Eur. Phys. J. B 1, 405 (1998).
5. L. LAZAR, K.WESTERHOLT, H. ZABEL, L. R. TAGIROV, YU.V. GORYUNOV, N.N. GARIFYANOV, I.A. GARIFULLIN, Superconductor/ferromagnet proximity effect in Fe/Pb/Fe trilayers, *Phys. Rev. B* 61, 3711 (2000).
6. BASKES, M.I. Modified embedded-atom potentials for cubic materials and impurities, *Physical Review B* **1992,** *46*, 2727-2742. doi: 10.1103/PhysRevB.46.2727
7. BASKES, M.I.; SRINIVASAN, S.G.; VALONE, S.M.; HOAGLAND, R.G. Multistate modified embedded atom method, *Physical Review B.* **2007,** *75*, 094113. doi: 10.1103/PhysRevB.75.094113
8. PLIMPTON, S. Fast Parallel Algorithms for Short-Range Molecular Dynamics, *Journal of Computational Physics* **1995,** *117*, Pp. 1-19. doi: 10.1006/jcph.1995.1039

CONTROLUL INTELIGENT A TEMPERATURII DE LA DISTANȚĂ ELABORAT PENTRU TRATAMENTUL TERMIC AL FOLIILOR DE ZINC

Cătălin CRECIUNEL[1,2*]

[1]*Centrul Național de Studiu și Testare a Materialelor (CNSTM), UTM, Chișinău, Republica Moldova*
[2] *Departamentul Microelectronică și Inginerie Biomedicală (DMIB), grupa MN-191, Facultatea Calculatoare, Informatică și Microelectronică (FCIM), UTM, Chișinău, Republica Moldova*

*Autorul corespondent: Cătălin CRECIUNEL, e-mail creciunel.catalin@mib.utm.md

Coordonator științific: Dr. Conf. Eduard MONAICO, *Centrul Național de Studiu și Testare a Materialelor, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova*

Rezumat. *A fost propusă obținerea oxidului de zinc utilizând tratamentul termic. Pentru a putea seta datele cât mai corect a fost propusă realizarea unui dispozitiv ce permite setarea și vizualizarea temperaturii. Dispozitivul poate seta temperatura cu ajutorul telefonului și să ofere starea actuală în timp real.*

Cuvinte cheie*: PID, microcontroler, Bluetooth, tratament termic, oxid de zinc.*

Introducere

Anterior a fost demonstrată transformarea straturilor de ZnSe în oxid de zinc (ZnO) cu ajutorul tratamentului termic la temperaturi ridicate folosite pentru lasere aliatoare [1]. Un reactor termic permite realizarea unui tratament termic, însă unul dintre neajunsurile sale este faptul că nu permite setarea unei temperaturi și măsurarea timpului. Nevoia de a controla aceste variabile, precum și de a determina momentul exact în care s-a ajuns la temperatura setată ne-a determinat să căutăm soluții smart. Oxidul de zinc are o perspectivă mare și la moment se utilizează în calitate de straturi transparente în regiunea vizibilă și infraroșie a spectrului, contacte electrice și straturi buffer în elementele solare și în dispozitivele de înregistrare a informației. Oxidul de zinc este un semiconductor din grupa A2B6 cu banda interzisă de 3,37 eV la temperatura camerei.

Anterior au fost raportate formarea straturilor de ZnO prin tratamentul termic al straturilor subțiri de Zn depuse pe substrat de sticlă [2]. Autorii au investigat tratamentul termic în diapazonul de temperaturi de la 450 °C la 650 °C. Temperatura de 600 °C fiind optimală pentru formarea nanofirelor e ZnO de o calitate înaltă.

Tratamentul termic de asemenea este folosit și pentru îmbunătățirea proprietăților straturilor nanostructurate formate. Spre exemplu, Guillaume și colegii au folosit post tratamentul termic la 973–1373 K în flux continuu de azot pentru recoacerea straturilor de ZnO depuse prim pulverizare magnetron, pentru a investiga transformarea caracteristicilor microstructurale și optice prin difracție de raze X, microscopie electronică de transmisie, fotoluminiscență și electroluminiscență [3]. În altă metodă de depunere [4], straturile crescute prezintă o singură fază hexagonală de Zn fără orientare preferenţială şi conțin doar 28% oxigen. Odată cu creșterea temperaturii de recoacere conținutul de oxigen este sporit, probele fiind total oxidate la 300 °C și transformate în material stoichiometric caracteristic pentru ZnO.

Soluția implementată a fost realizarea unui termostat ce are la bază algoritmul PID și permite setarea temperaturii utilizând interfața Bluetooth. Algoritmul PID se folosește peste tot unde este nevoie de controlul automat al diferitelor procese, cum ar fi: controlul vitezei, controlul temperaturii în frigidere, incubatoare, sobe termice, printarea 3D.

Controlorul PID este compus din trei componente: componenta proporțională (P); componenta integrală (I); componenta derivată (D). Semnalul de ieșire reprezintă suma tuturor componentelor, fiecare din ele având un coeficient propriu. Această sumă, după calcul, devine semnal de comandă ce se transmite dispozitivului de control:

$$out = P*kP + I*kI + D*kD \quad (1)$$

Componenta P reprezintă diferența dintre valoarea actuală de pe senzor și punctul de referință – eroarea proporțională arată cât de departe este parametrul sistemului de punctul de referință:

P = punctSetat - valActual (2)

Componenta I reprezintă suma tuturor erorilor înmulțită la perioada de discretizare (dt) a sistemului:

I = I + (punctSetat - valActual)*dt (3)

Componenta D reprezintă diferența dintre eroarea prezentă (err) și cea anterioară (prev_err), împărțită la timpul dintre calcule. Aceasta reacționează la schimbările semnalului venite dinspre senzor, ceea ce permite compensarea zgomotului în sistem și micșorarea oscilațiilor [5]: err = punctSetat - valActual

D = (err - prev_err)/dt (4)

prev_err = err.

Cea mai mare dificultate întâlnită în utilizarea algoritmului PID este identificarea valorii coeficienților kP, kI, kD. Drept soluție a fost propusă utilizarea metodei, gradului de stabilitate maximă (maximum stability degree, GMS). Pentru acest scop este necesar construirea unui grafic ce ne permite vizualizarea în timp a valorilor setate și reale [6].

Materiale și metode

Pentru a putea realiza comunicarea telefonului cu dispozitivul realizat a fost selectată cea mai ieftină metoda actuală, și anume un microcontroler ce ar conține deja acest modul integrat. Microcontrolerul selectat este ESP32, ce are la bază un sistem pe 32 biți, are modul WIFI, Bluetooth, periferie GPIO (periferie de intrare și ieșire a pinilor), UART, SPI, I_2C.

Măsurarea temperaturii luând în calcul spectrul mare de valori a temperaturilor ce pot fi setate a fost destinat unui termocuplu de tip K care poate măsura valori intre 0 și 1200°C [4].

Conversia datelor analogice obținute de termocuplu în digitale pentru a putea fi utilizate ulterior este realizat cu ajutorul modulului MAX6675, cu un interval de măsurare între 0°C și 1024 °C, o precizie de măsurare de ± 1,5 °C, rezoluția temperaturii de 0,25 °C, și modul de ieșire, semnal digital SPI [5].

Asamblând elementele de colectare a adatelor a fost necesar realizarea modulului de control a semnalului, în acest scop a fost asamblat în baza unui triac conectat la un optocuplor ce ne asigură decuplarea galvanică a semnalului de comanda de 3 V de cel de 220 V, comutarea optocuplorului fiind realizată prin intermediul unui tranzistor.

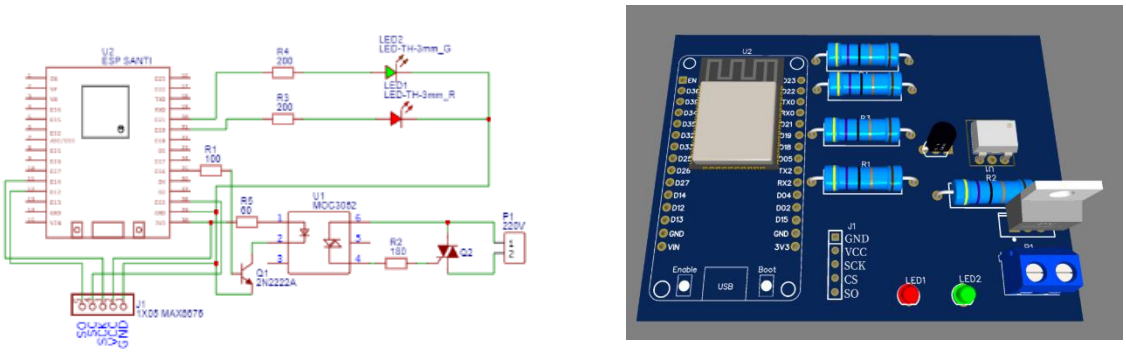


Figura 1. Schema principială a circuitului realizat (stânga) și modelul 3D a dispozitivului (dreapta)

În scopul înștiințării persoanelor ce nu sunt conectate la dispozitiv cu ajutorul aplicației ce permite vizualizarea și setarea temperaturii, au fost conectate două LED-uri: cel roșu destinat informării că temperatura nu corespunde celei setate, iar cel verde înștiințează despre atingerea temperaturii setate.

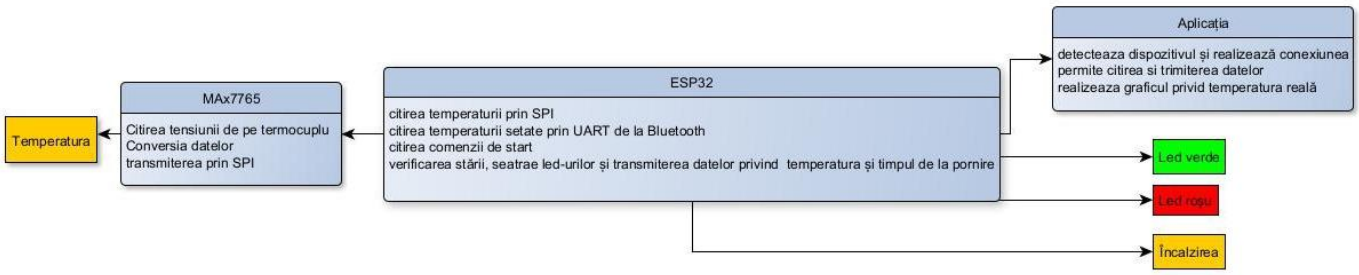


Figura 2. Schema conceptuală a dispozitivului

Codul pentru acest dispozitiv a fost realizat cu ajutorul limbajului de programare C, utilizând PlatformIO, conformitate Figura 3 în care este descrisă schema bloc. De asemenea, din schema bloc poate fi observată care este frecvența de citire și transmitere a datelor.

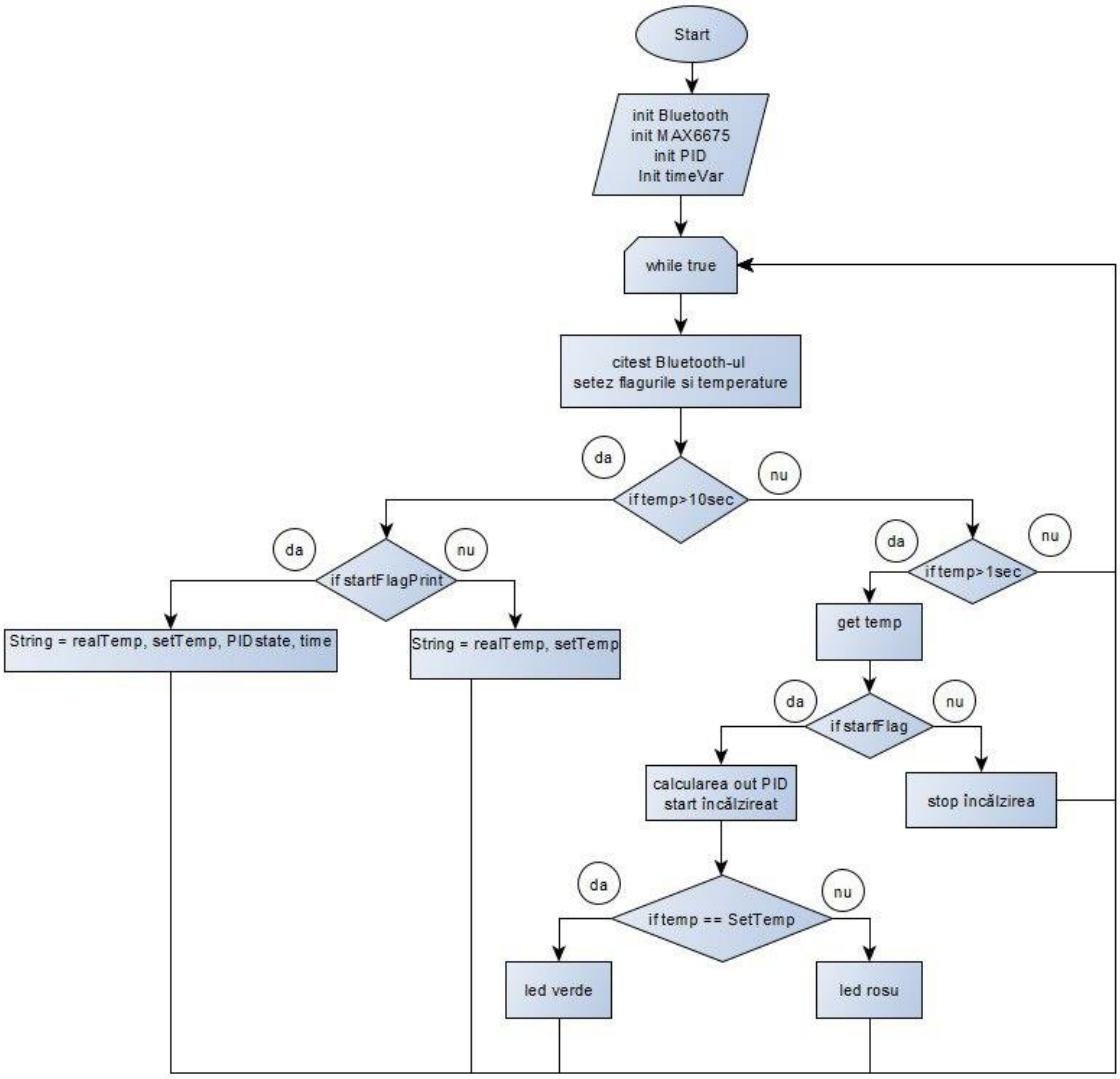


Figura 3. Schema bloc a dispozitivului realizat

Rezultate și discuții

Cu ajutorul dispozitivului realizat fost obținute probe de ZnO folosind două abordări diferite și anume, tratamentul termic al foliilor de Zn și tratarea foliilor de Zn cu strat e aur depus pe suprafață. Parametrii tehnici ai experimentelor sunt sumarizați în Tabelul 1.

Tabelul 1

Probele tratate termic pentru a obține ZnO

Nr.	Timp, min	Temperatura, °C	Comentariu
1	10	800	Conține aur
2	45	200	Conține aur
3	45	400	
4	15	600	Conține aur
5	15	800	Conține aur
6	30	400	Conține aur
7	30	300	Conține aur
8	30	200	Conține aur

În experimente au fost utilizate folii de Zn cu puritatea de 99,9 %, grosimea de 0,25 mm procurate de la Sigma-Aldrich. Înaintea procesului de tratare termică, probele au fost funcționalizate cu un strat de aur depuse cu ajurul plasmei in vid la 0,05 bari cu un curent de 40 mA la o tensiune automata, timp de 13 secunde pentru a obține grosimea de 13 nm. În timpul tratamentului termic, stratul de Au se conglomerează în dote de Au care ar trebui să introducă schimbări în morfologia obținută.

Studiile de microscopie electronică cu scanare au demonstrat formarea de nanofire de ZnO cu lungimea de câțiva micrometri și diametrul mediu mai mic de 200 nm. Structuri similare au fost fabricate prim metoda similară, dar fără depunerea stratului de aur și folosite în calitate de senzori de gaze [7].

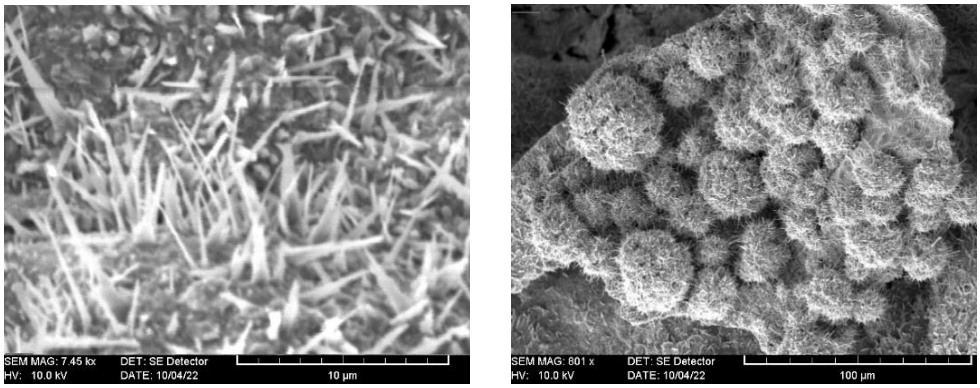


Figura 4. Imaginile SEM a straturilor de ZnO obținute la tratarea termică la temperatura de 800 °C timp de 10 min a foliei de zinc fără depunerea Au (stânga) și a foliei de zinc cu aur (dreapta)

Din Figura 4, pot fi observate structuri de tip con pentru probele fără stratul de aur. Se observă că pentru probele cu aur depus pe suprafață înaintea procesului de tratare termică, morfologia constă în formarea structurilor de tip arici (Figura 4 dreapta). Se constată că densitatea pe suprafață a firelor este relativ mai mare, atunci când tratamentul termic este efectuat la temperatură mai înalte și durate scurte. Investigarea influenței stratului de aur asupra morfologiei necesită investigații mai minuțioase, precum și investigarea proprietăților fotoluminescente care for fi efectuate în continuare în cadrul tezei de licență.

Concluzii

Dispozitivul realizat își îndeplinește sarcina inițial propusă, permite controlul temperaturii la distanță utilizând telefonul mobil, iar utilizarea Bluetooth-ului prezintă un avantaj de securitate, deoarece va putea fi setat doar de cei ce sunt în apropiere și vor putea dirija procesul. Experimental a fost demonstrată influența stratului de aur depus pe suprafața foliilor de zinc, care ulterior au fost supuse tratamentului termic.

Mulțumiri

Autorul aduce mulțumiri conducătorului științific, domnului Dr. conf. Eduard MONAICO și domnului Dr. Vladimir CIOBANU pentru acel bagaj vast de sfaturi, cunoștințe, abilități și aptitudini de lucru acumulate în laborator. Această lucrare a fost elaborată în cadrul proiectului program de stat cu cifrul #20.80009.5007.20 și proiectul bilateral cu cifrul #22.80013.5007.4BL.

Referințe

1. URSAKI, V.V. ZALAMAI, V.V. BURLACU, A.; KLINGSHIRN, C. MONAICO, E. TIGINYANU, I.M. Random Lasing in Nanostructured ZnO Produced from Bulk ZnSe. *Semicond. Sci. Technol*, **2009**, *24*, 085017, doi:10.1088/0268-1242/24/8/085017.
2. KHANLARY, M.R. VAHEDI, V. REYHANI, A. Synthesis and Characterization of ZnO Nanowires by Thermal Oxidation of Zn Thin Films at Various Temperatures. *Molecules*, **2012**, *17*, pp.5021–5029, doi:10.3390/molecules17055021.
3. GUILLAUME, C. FRIEIRO, J.L. BLÁZQUEZ, O. LABBÉ, C. LÓPEZ-VIDRIER, J. GARRIDO, B. HERNÁNDEZ, S. LIU, B. KHOMENKOVA, L. FRILAY, C. LEMARIÉ, F. LEROUX, C. PELLOQUIN, D. PORTIER, X. Influence of Post Annealing Treatments on the Luminescence of Rare Earth Ions in ZnO:Tb,Eu/Si Heterojunction. *Applied Surface Science*, **2021**, *556*, 149754, doi:10.1016/j.apsusc.2021.149754.
4. AIDA, M.S. TOMASELLA, E. CELLIER, J. JACQUET, M.; BOUHSSIRA, N. ABED, S.; MOSBAH, A. Annealing and Oxidation Mechanism of Evaporated Zinc Thin Films from Zinc Oxide Powder. *Thin Solid Films*, **2006**, *515*, pp.1494–1499, doi:10.1016/j.tsf.2006.04.011.
5. BARBOS, O. Sistem de monitorizare și dirijare a temperaturii în baza controlorului pid. In *Proceedings of the Sesiune națională cu participare internațională de comunicări științifice studențești, ediția XXV – 2021*, Chișinău: CEP USM, April 22 2021; pp. 60–63.
6. COJUHARI, I. IZVOREANU, B; FIODOROV, I. MORARU, D. LISNIC, I. Synthesis of the pid controller to the system with maximum stability degree based on the genetic algorithm. In *Proceedings of the International Conference on Electronics, Communications and Computing – 2019*; Chisinau, Moldova, October 23 2019; p. 41.
7. CAMPOS, A.C. PAES, S.C. CORREA, B.S. CABRERA-PASCA, G.A. COSTA, M.S. COSTA, C.S.; OTUBO, L. CARBONARI, A.W. Growth of Long ZnO Nanowires with High Density on the ZnO Surface for Gas Sensors. *ACS Appl. Nano Mater*. **2020**, *3*, pp.175–185, doi:10.1021/acsanm.9b01888.

OBȚINEREA ȘI STUDIUL STRATURILOR DE OXID DE ZINC PRIN CORODAREA ELECTROCHIMICĂ A FOLIILOR DE ZINC

Cătălin CRECIUNEL[1,2*]

[1]*Centrul Național de Studiu și Testare a Materialelor (CNSTM), Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova*
[2] *Departamentul Microelectronică și Inginerie Biomedicală (DMIB), grupa MN-191, Facultatea Calculatoare, Informatică și Microelectronică (FCIM), Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova*

*Autorul corespondent: Cătălin CRECIUNEL, e-mail creciunel.catalin@mib.utm.md

Coordonator științific: Dr. Conf. Eduard MONAICO, *Centrul Național de Studiu și Testare a Materialelor, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova*

Rezumat. *A fost obținute straturi subțiri de oxid de zinc utilizând corodarea electrochimică fiind o metodă simplă, rapidă și eficientă prin care se poate obține diferite structuri. A fost studiată influența naturii și concentrației electrolitului, tensiunii de anodizare asupra morfologiei straturilor de ZnO. După procesul de anodizare, a fost stabilit că cea mai eficientă soluție sa demonstrat a fi anodizarea în NaOH. A fost optimizat potențialul de anodizare aplicat prin studiul morfologiei cu ajutorul microscopului electronic.*

Cuvinte cheie*: Anodizare, oxid de zinc, fotoluminescență, nanostructurare, strat poros.*

Introducere

În prezent oxidul de zinc (ZnO) este un material utilizat la scară largă în electronică datorită proprietăților sale unice [1–3]. În dispozitivele electronice, este adesea folosit ca material semiconductor pentru diverse aplicații, inclusiv:

- Celule solare: ZnO este un absorbant eficient de lumină și are o bandă interzisă largă, ceea ce îl face potrivit pentru utilizarea în dispozitive fotovoltaice [4–6].
- Tranzistoare: ZnO poate fi folosit ca material în calitate de canal pentru tranzistoarele cu efect de câmp (TEC), care sunt utilizate pe scară largă în circuitele digitale și analogice [7,8].
- Diode emițătoare de lumină (LED): ZnO este adesea folosit ca substrat pentru fabricarea LED-urilor, oferind stabilitate mecanică și conductivitate electrică structurii LED [9].
- Senzori de gaz: ZnO are o sensibilitate ridicată la diferite gaze, ceea ce îl face un material popular pentru dispozitivele de detectare a gazelor [10].

Metodele de obținere a oxidului de zinc la scară nanometrică sunt sistematizate în articolele de sinteză [1,5,11,12]. Majoritatea metodelor necesită temperaturi ridicate pentru a obține structuri de o calitate înaltă. Un exemplu potrivit ar fi transformarea ZnSe poros în ZnO la tratament termic la 900 °C [13]. A fost raportată recent nanostructurarea ZnO prin corodarea cristalelor masive de ZnO ce rezultă în morfologii diferite [14].

O metodă alternativă, la temperatura camerei poate servi corodarea electrochimică (CE) a foliilor de Zn [15]. CE a materialelor semiconductoare a demonstrat că se pot obține diferite morfologii, în dependență de materialul utilizat, natura electrolitului, tensiunea aplicată, etc [16].

Materiale și metode utilizate

Pentru procesul de corodare electrochimică au fost folosite folii de Zn cu puritatea de 99,9 %, grosimea de 0,25 mm procurate de la Sigma-Aldrich. Pentru a obține oxidul de zinc am utilizat mai multe concentrați și soluții. Ca electroliți de studiu au fost alese două soluții în bază de NaOH și alta în bază de $NaHCO_3$ dizolvate în apă distilată, concentrația fiind variată de la 0,1 M până la 0,5 M. Procesul de anodizare poate decurge în două forme, potențiostatic sau galvanostatic (la tensiune constantă sau la curenți constanți).

Standul de anodizare constă din vasul în care se află soluția, un electrod din platină scufundat în soluție, un alt electrod conectat la probă conform Figurii 1, dirijate la o sursă de tensiune reglabilă Keithley 2400 conectată la calculator. Electrolitul a fost agitat cu ajutorul mesei cu magneți, rotind bagheta din teflon cu miez metalic ce se află în vasul cu soluție.

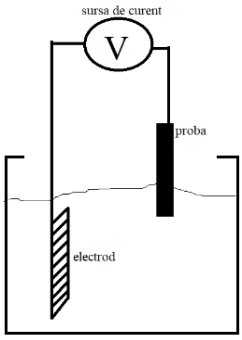


Figura 1. Schema procesului de anodizare

Rezultate și discuții

Pentru a optimiza procesul de corodare în soluțiile pregătire, probele au fost supuse măsurării curbelor de polarizare, prin măsurarea curentului în dependență de tensiunea aplicată prezentate în Figura 2. Concentrația electrolitului introduce schimbări majore în procesul de nanostructurare după cum se observă din compararea imaginilor din Figura 2a și 2b. La concentrația de 0,1 M NaOH curba de polarizare la 20 V nu diferă mult după formă de Figura 2a, spre deosebire de cazul măririi concentrației la 0,2 M NaOH. La așa concentrație a electrolitului se observă că la mărirea tensiunii de la 0 până la aproximativ 15 V are loc mărirea exponențială a curentului, la parcurgerea inversă, adică micșorarea tensiunii se observă că curentul are valori mult mai mici, indicând o oxidare puternică a foliei de Zn (vezi Figura 2b). În continuare se va examina concentrația de 0,1 M NaOH rezultând în formarea mai uniformă a oxidului.

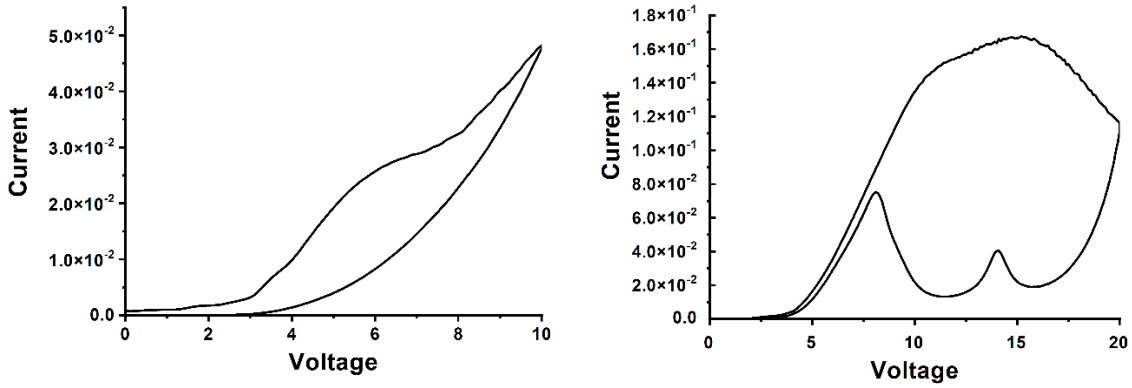


Figura 2. Curba de polarizare cu mărirea tensiunii și revenire la zero pentru folia de Zn în NaOH cu concentrația de 0,1 M (a) și 0,2 M (b)

Un alt parametru important studiat reprezintă tensiunea de anodizare aplicată. În Figura 3 sunt prezentate rezultatele investigațiilor cu ajutorul microscopului electronic cu scanare (SEM) VEGA Tescan 5130 MM a straturilor de ZnO formate la tensiuni diferite. Un interes prezintă probele obținute la tensiuni mai mici (10V - 20V) reprezentând o structură mai uniformă și nanostructurată similar imaginii prezentate în Figura 3c. Cu mărirea tensiunii de anodizare se observă formarea stratului cu multe fisuri și fără o uniformitate. De menționat că foliile inițiale prezintă unele regiuni cu adâncituri probabil de la procesul e poleire mecanică a foliilor.

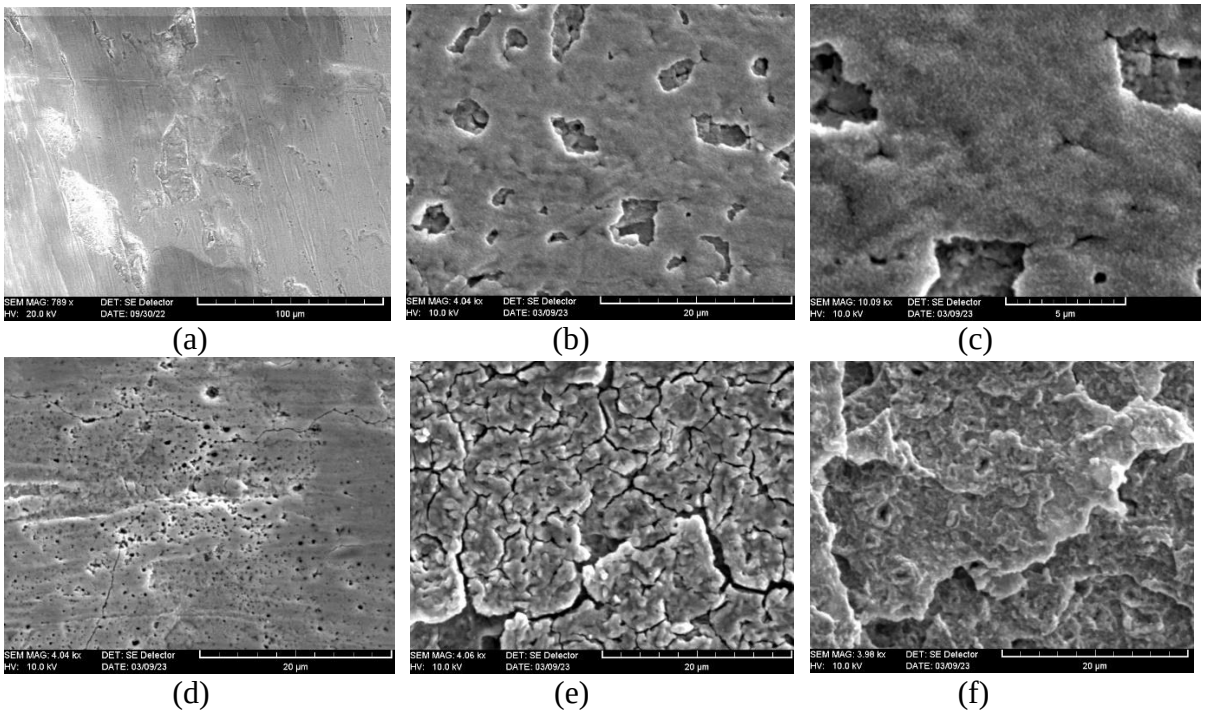


Figura 3. Imaginile SEM a foliei inițiale de Zn (a) și a foliilor supuse corodării electrochimice în 0,1 M NaOH timp e 30 min la tensiunea de 10 V (b,c), 20 V (d), 30 V (e) și 40 V (f)

Spectrele fotoluminescenței au fost măsurate la temperatura camerei cu ajutorul monocromatorului MDR2 cu rețea de difracție 1200 linii/mm. Pentru excitare a fost folosit laserul de He-Cd cu linia laser 325 nm. Rezoluția alcătuiește 0,5 meV. În spectrele de luminiscență ale probelor de oxid de zinc obținute prin corodarea foliei de zinc la tensiunea de 10 V și 20 V, se observă două caracteristici: o linie verde la aproximativ 2.4 eV și o linie ultravioletă la aproximativ 3.3 eV (Figura 4).

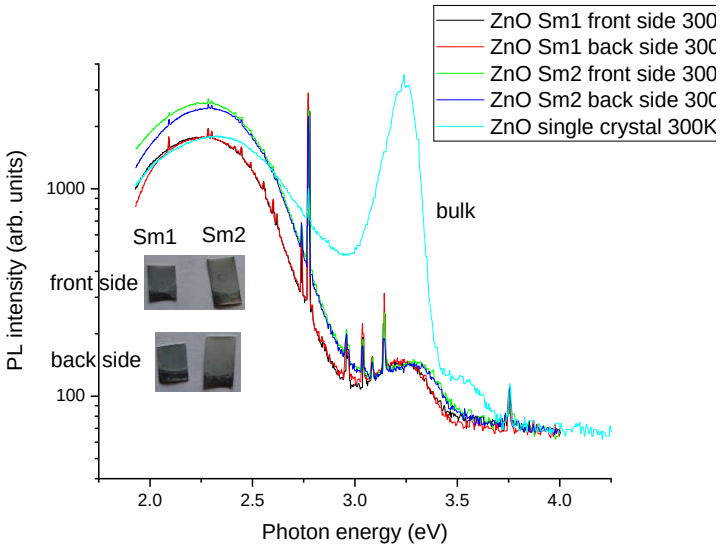


Figura 4. Spectre de fotoluminescență măsurate la temperatura camerei a probelor corodate (folie de zinc) și monocristal de oxid de zinc (curbă albastră). Insert prezintă imaginile foto ale probelor

Linia verde este legată de defecte intrinseci ale materialului (vacanțe de oxigen și zinc interstițial [2,17]), în timp ce linia ultravioletă este legată de luminescența fundamentală. Pentru comparație, în Figura 4 este prezintat spectrul de oxid de zinc monocristal măsurat în aceleași condiții

ca și probele corodate. Se poate observa că luminescența fundamentală este mai pronunțată în spectrele monocristalului (curba bulk) decât în probele corodate. Acest lucru poate fi explicat prin formarea unui strat foarte subțire timp de 30 min de corodare. La moment au fost formate straturi la corodare timp de 2 ore ce vor fi supuse investigațiilor ulterioare.

Concluzii

A fost optimizată concentrația electrolitului și tensiunea de anodizare fiind e 0,1 M NaOH și respectiv 10 V – 20 V. Din soluțiile utilizate cel mai stabil este NaOH formând un strat solid cu o adeziune bună, soluția de $NaHCO_3$, s-a obținut cu o structură mult mai poroasă însă în lipsa unei adeziuni bune de substrat, nu a putut fi cercetat. Spectrele de fotoluminescență măsurate demonstrează posibilitatea obținerii oxidului de zinc prin metoda anodizării, fiind necesară mărirea duratei anodizării pentru producerea straturilor mai groase. Spectrele prelevate pe diferite părți ale aceleiași probe sunt identice, ceea ce indică omogenitatea nanostructurilor de oxid de zinc rezultate pe suprafața foliei de zinc.

Mulțumiri

Autorul aduce mulțumiri conducătorului științific, domnului Dr. conf. Eduard MONAICO și domnului Dr. Victor ZALAMI. Această lucrare a fost elaborată în cadrul proiectului bilateral cu cifrul 22.80013.5007.4BL.

Referințe

1. KOŁODZIEJCZAK-RADZIMSKA, A. JESIONOWSKI, T. Zinc Oxide—From Synthesis to Application: A Review. *Materials*, **2014**, *7*, pp.2833–2881, doi:10.3390/ma7042833.
2. KLINGSHIRN, C.F. MEYER, B.K. WAAG, A. HOFFMANN, A. GEURTS, J. *Zinc Oxide: From Fundamental Properties Towards Novel Applications*, Springer Series in Materials Science Springer: Berlin, Heidelberg, 2010, Vol. 120, ISBN 978-3-642-10576-0.
3. SHARMA, D.K. SHUKLA, S. SHARMA, K. K. KUMAR, V. A. Review on ZnO: Fundamental Properties and Applications. *Materials Today: Proceedings*, **2022**, *49*, pp.3028–3035, doi:10.1016/j.matpr.2020.10.238.
4. WIBOWO, A. AGUNG MARSUDI, M. IKHLASUL AMAL, M. BAGAS ANANDA, M. STEPHANIE, R. ARDY, H.; JAYA DIGUNA, L. ZnO Nanostructured Materials for Emerging Solar Cell Applications. *RSC Advances*, **2020**, *10*, pp.42838–42859, doi:10.1039/D0RA07689A.
5. VITTAL, R.; HO, K.-C. Zinc Oxide Based Dye-Sensitized Solar Cells: A Review. *Renewable and Sustainable Energy Reviews* **2017**, *70*, pp.920–935, doi:10.1016/j.rser.2016.11.273.
6. PIETRUSZKA, R. WITKOWSKI, B.S. OZGA, M. GWOZDZ, K. PLACZEK-POPKO, E. GODLEWSKI, M. 9.1% Efficient Zinc Oxide/Silicon Solar Cells on a 50 Mm Thick Si Absorber. *Beilstein J Nanotechnol*, **2021**, *12*, pp.766–774, doi:10.3762/bjnano.12.60.
7. JIANG, L. LI, J. HUANG, K.; LI, S. WANG, Q. SUN, Z. MEI, T. WANG, J. ZHANG, L. WANG, N. WANG, X. Low-Temperature and Solution-Processable Zinc Oxide Transistors for Transparent Electronics. *ACS Omega*, **2017**, *2*, pp.8990–8996, doi:10.1021/acsomega.7b01420.
8. YOU, H.C. Transistor Characteristics of Zinc Oxide Active Layers at Various Zinc Acetate Dihydrate Solution Concentrations of Zinc Oxide Thin-Film. *Journal of Applied Research and Technology. JART* **2015**, *13*, pp.291–296, doi:10.1016/j.jart.2015.06.003.
9. RAHMAN, F. Zinc Oxide Light-Emitting Diodes: A Review. *OE* **2019**, *58*, 010901, doi:10.1117/1.OE.58.1.010901.
10. LIU, J. GUO, Z. MENG, F. LUO, T. LI, M. LIU, J. Novel Porous Single-Crystalline ZnO Nanosheets Fabricated by Annealing ZnS(En) 0.5 (En = Ethylenediamine) Precursor. Application in a Gas Sensor for Indoor Air Contaminant Detection. *Nanotechnology*, **2009**, *20*, 125501, doi:10.1088/0957-4484/20/12/125501.

11. ASPOUKEH, P.K. BARZINJY, A.A. HAMAD, S.M. Synthesis, Properties and Uses of ZnO Nanorods: A Mini Review. *Int Nano Lett*, **2022**, *12*, pp.153–168, doi:10.1007/s40089-021-00349-7.
12. WANG, J. CAO, J. FANG, B. LU, P. DENG, S.; WANG, H. Synthesis and Characterization of Multipod, Flower-like, and Shuttle-like ZnO Frameworks in Ionic Liquids. *Materials Letters*, **2005**, *59*, pp.1405–1408, doi:10.1016/j.matlet.2004.11.062.
13. URSAKI, V.V. ZALAMAI, V.V. BURLACU, A. KLINGSHIRN, C. MONAICO, E. TIGINYANU, I.M. Random Lasing in Nanostructured ZnO Produced from Bulk ZnSe. *Semicond. Sci. Technol*, **2009**, *24*, 085017, doi:10.1088/0268-1242/24/8/085017.
14. ZALAMAI, V.V. COLIBABA, G.V. MONAICO, E.I. MONAICO, E.V. Enhanced Emission Properties of Anodized Polar ZnO Crystals. *Surf. Engin. Appl.Electrochem*, **2021**, *57*, pp.117–123, doi:10.3103/S1068375521010166.
15. CHING, C.G. OOI, P.K. NG, S.S. AHMAD, M.A. HASSAN, Z. ABU HASSAN, H. ABDULLAH, M.J. Fabrication of Porous ZnO via Electrochemical Etching Using 10wt% Potassium Hydroxide Solution. *Materials Science in Semiconductor Processing*, **2013**, *16*, pp.70–76, doi:10.1016/j.mssp.2012.06.017.
16. MONAICO, E. TIGINYANU, I. URSAKI, V. Porous Semiconductor Compounds. *Semicond. Sci. Technol*, **2020**, *35*, pp.103001, doi:10.1088/1361-6641/ab9477.
17. ÖZGÜR, Ü. ALIVOV, YA.I. LIU, C. TEKE, A. RESHCHIKOV, M.A. DOĞAN, S. AVRUTIN, V. CHO, S.-J.; MORKOÇ, H. A Comprehensive Review of ZnO Materials and Devices. *Journal of Applied Physics*, **2005**, *98*, 041301, doi:10.1063/1.1992666.

IMPLEMENTATION OF A SOURCE METER CONTROL SYSTEM WITH PYTHON USING SCPI AND VISA

Alexandr SEREACOV[1*], **Adrian BIRNAZ**[1]

[1]*Department of Microelectronics and Biomedical Engineering, Faculty of Computers Informatics and Microelectronics, Technical University of Moldova, Chisinau, Republic of Moldova*

*Corresponding author: Alexandr Sereacov, alexandr.sereacov@mib.utm.md

Tutor/coordinator: Oleg LUPAN, doctor habilitatus, *Department of Microelectronics and Biomedical Engineering, Faculty of Computers Informatics and Microelectronics, Technical University of Moldova, Chisinau, Republic of Moldova*

Abstract. *This paper describes existing communication with electronic devices open standards and experience of their integration in measurement system software. The presented measurement system software is designed to use SCPI over VXI bus to communicate with measurement device. Also, there is described a modular software architecture to make possible to interchange communication logic without touching other modules.*

Keywords: *SCPI, VISA, clean architecture, Python, modular software design*

Introduction

Nowadays laboratories are equipped with a very large variety of measurement devices which require unique software to work with. Building of a complex measurement system based on multiple different devices produces by different vendors requires usage of expensive software products. It this paper we describe our approach to implement measurement system using open standards and Python language to implement a user interface and data acquisition software for a third-party measurement device.

Communication layer

SCPI (Standard Commands for Programmable Instruments) is a standard language that defines a common syntax for communicating with test and measurement instruments, such as oscilloscopes, signal generators, and multimeters. SCPI provides a standardized set of commands and parameters that can be used to control and query instrument settings, as well as to read measurement data and status information. These program commands and parameters are sent from a controller to an instrument using IEEE 488.1, VXIbus, RS-232C, etc., interfaces [1].

One of the key benefits of using SCPI is that it provides a consistent programming interface across different types of instruments from different manufacturers. This means that a programmer can write a single program that can control a wide range of instruments without having to learn different command sets for each one.

SCPI commands and parameters are designed to be flexible, allowing instruments to accept a range of formats for programming. For example, a command to set the frequency of a signal generator might be accepted in multiple formats, such as "FREQ 1kHz", "FREQ:FIXED 1000", or "FREQ:CW 1000MHz" [1-3].

When an instrument responds to a SCPI command, it can send back both data and status information. Data information can be formatted in a device- and measurement-independent way, making it easier for the programmer to work with the data regardless of the specific instrument being used.

Overall, SCPI provides a powerful and flexible standard for controlling and querying test and measurement instruments, reducing the programming effort required to work with a variety of instruments and improving the consistency and reliability of instrument control.

SCPI command set for generic source meter:

- :ABORt
- :FETCh?
- :MEASure:<function>?
- :READ?
- :CALCulate
- :DIGital
- :DISPlay
- :FORMat
- :ROUTe
- :SENSe1
- :SOURce
- :STATus
- :SYSTem
- :TRACe
- :TRIGger

Horizontal and vertical consistency are two important concepts in SCPI programming that help to ensure a consistent and standardized programming interface across different instruments and functions [1].

Horizontal consistency refers to the consistency of SCPI commands and parameters across different instruments of the same type. For example, all oscilloscopes from different manufacturers should accept the same SCPI commands and parameters for setting the time base, triggering, and acquiring waveform data. This makes it easier for programmers to write generic code that can be used with different instruments, and also reduces the learning curve when switching between instruments.

Vertical consistency, on the other hand, refers to the consistency of SCPI commands and parameters across different functions within the same instrument. For example, the same command and parameter format should be used to set the trigger level on a scope, whether the trigger is for a single-shot acquisition or a repetitive waveform capture. This ensures that programmers can reuse code and commands across different functions within the same instrument, and also reduces the likelihood of errors due to differences in command formats [1].

A key to consistent programming is the reduction of multiple ways to control similar instrument functions. The philosophy of SCPI is for the same instrument functions to be controlled by the same SCPI commands. To simplify learning, SCPI uses industry-standard names and terms that are manufacturer and customer supported.

SCPI provides several different levels of instrument control. Simple Measure commands provide users easy and quick control of SCPI instrumentation, while more detailed commands provide traditional instrument control [1].

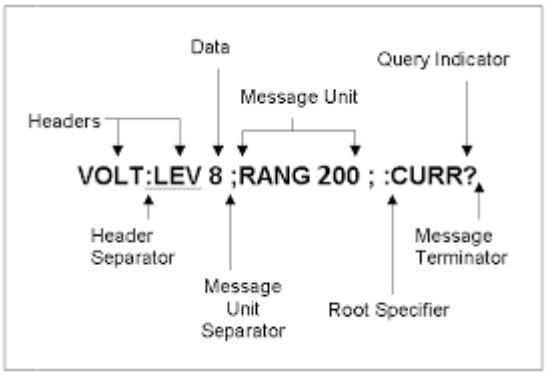


Figure 1. Example of SCPI commands syntax

One of the key design goals of SCPI is to allow for future expansion with new commands and parameters, without breaking compatibility with existing instruments and programs. This is achieved through a standardized command structure and syntax, which allows new commands to be added without conflicting with existing ones.

However, while SCPI commands and parameters can be expanded in the future to support new instruments, test programs designed for newer instruments may not be fully compatible with older instruments. This is because the new commands may not be recognized by the older instruments, or the parameters may be interpreted differently.

Additionally, SCPI provides a standardized mechanism for querying an instrument's capabilities, allowing a program to check whether a specific command or parameter is supported before sending the command.

However, SCPI does not provide complete instrument interchangeability, as it only defines the programming commands and responses, and not the instrument functionality, accuracy, resolution, connectors, etc. These factors are critical for ensuring that a replacement instrument can be seamlessly integrated into an ATE system without requiring hardware or software modifications.

In practice, replacing an SCPI instrument with another SCPI instrument typically requires less effort than replacing a non-SCPI instrument, since the programming commands and responses are standardized. However, even with SCPI instruments, there may be some differences in the specific commands and parameters used by different manufacturers or models, which can require some adjustments to the ATE system software.

To achieve true instrument interchangeability, additional standards are needed to define the physical and functional characteristics of instruments, such as the mechanical and electrical interfaces, accuracy specifications, and performance characteristics. Standards such as VXIbus, PXI, and LXI provide additional levels of standardization beyond SCPI, defining physical and electrical interfaces that can help to ensure that instruments from different manufacturers can be easily integrated into an ATE system.

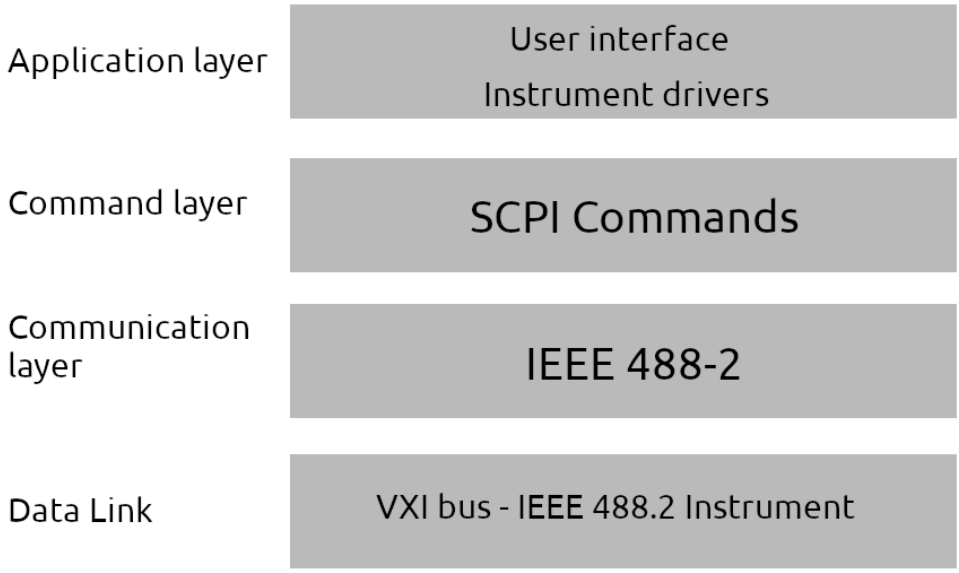


Figure 2. Protocol stack used for system implementation

Implementing clean architecture-based software

When designing software systems that involve user interfaces, it is often beneficial to combine a user interface implementation pattern with the clean architecture pattern. This can help to ensure that the system is easy to use and maintain, while also providing a high degree of modularity and flexibility.

The user interface implementation pattern typically involves separating the user interface code from the business logic and data access code. This is achieved by creating a set of classes and objects that are responsible for presenting information to the user and responding to user input, while delegating the actual processing of that input to separate layers of the system.

The clean architecture pattern, on the other hand, emphasizes the separation of concerns and the creation of independent modules that can be easily swapped in and out as needed. It achieves this by organizing the system into layers, each with a specific responsibility and purpose.

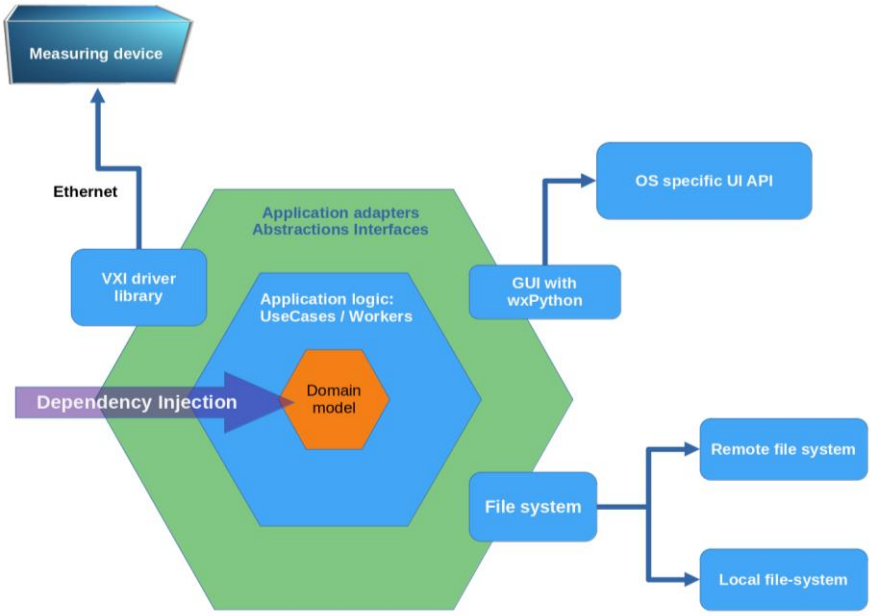


Figure 3. Clean architecture layers diagram

Combining these two patterns involves creating a user interface layer that interacts with the business logic layer in a clean and modular way. This can be achieved by creating interfaces or protocols that define the communication between the two layers, and implementing those interfaces in separate classes or modules.

For example, the user interface layer might define a set of protocols or interfaces that describe how user input should be handled, and the business logic layer might implement those protocols in separate classes or modules. This allows the two layers to communicate with each other in a way that is flexible and modular, and makes it easy to modify or replace individual components of the system without affecting the rest of the codebase.

Overall, combining a user interface implementation pattern with the clean architecture pattern can help to create a software system that is easy to use, maintain, and modify over time. By separating concerns and organizing the system into independent modules, it is possible to achieve a high degree of flexibility and modularity, while still providing a seamless and intuitive user experience.

User interface structure and implementation

This program provides the possibility to perform real time measurements with data saving and visualization features. Two work modes are supported:

- Volt-Ampere characteristics
- Real time transient analysis

Real time transient analysis has several adjustable parameters:

- Source voltage
- Maximum current limit
- Experiment temperature
- Agent to test sensitivity on. (Ex: gas, humidity level, light wavelength)

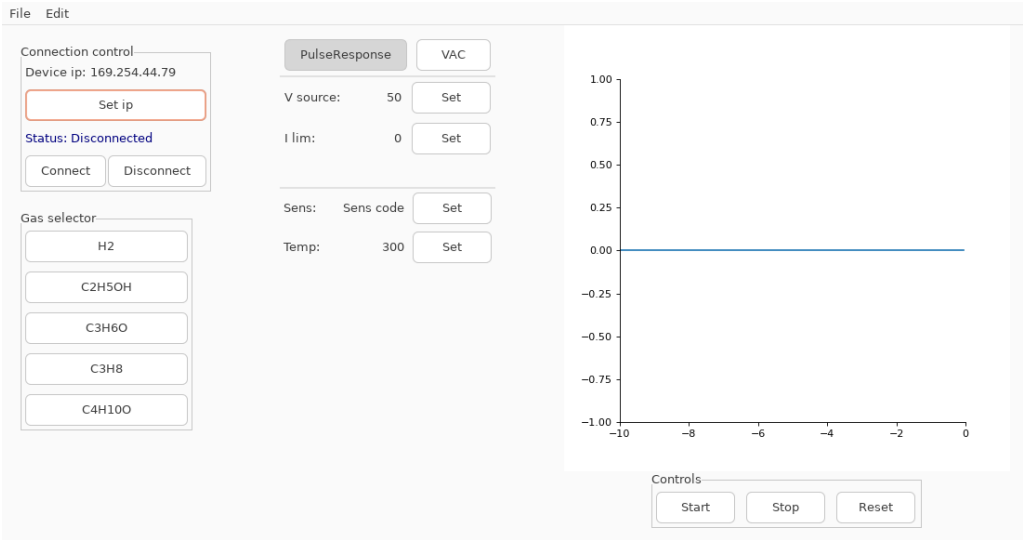


Figure 4. Graphic user interface for real time transient analysis.

Volt-Ampere characteristics has 3 main parameters:

- Start voltage level
- End voltage level
- Number of measurement points (how many measurements of electrical current level will be performed during voltage change from it's start level to it's end level).

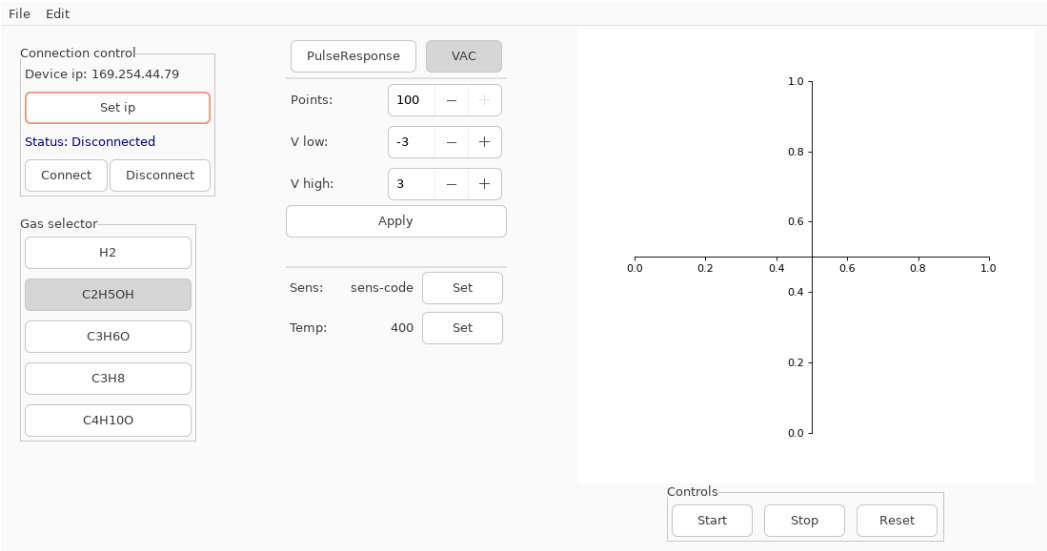


Figure 5. Graphic user interface for volt-ampere analysis

The source voltage can be set within the limits of millivolts up to 250 volts. Should be mentioned, that this limit is based on the measurement devices capabilities and on the software ones.

Graphic user interface structure is presented on Fig. 6. We knowingly left the GUI structure without implementation details, to keep it is structure cleaner. User interface handling is implemented in reactive manner. Then some event occurs, visual element responsible for this kind of event is changing it's properties.

User interface logic works on it's separate thread to keep it independent from the communication and data capture logic.

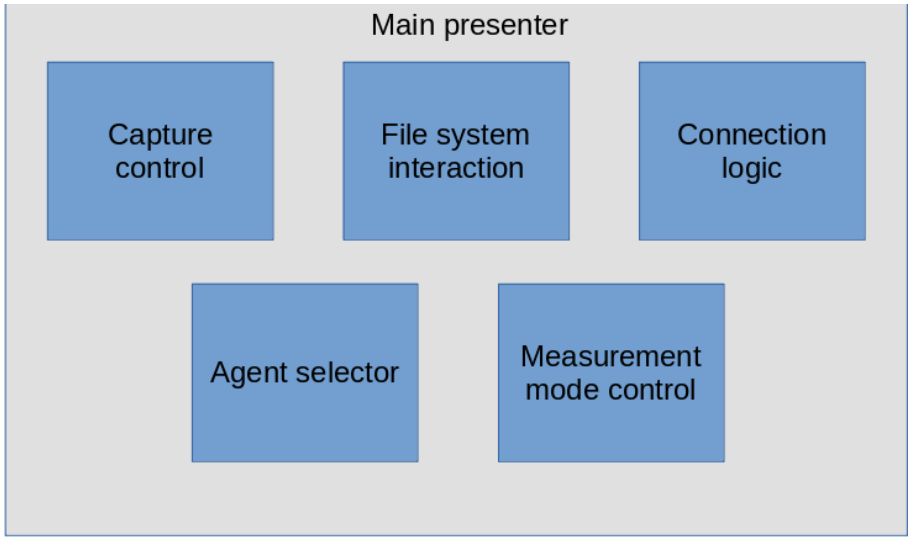


Figure 6. Graphic user interface main modules

Conclusions

Presented measuring system architecture has as a main goal to be standardized and opened for modifications and evolution. This lead us to the system with a communication layer based on open standards like: SCPI command standard, VXI network protocol, Ethernet protocol. And also, software part was based on the free licensed solutions: Python programming language, wxPython cross-platform framework for user interface implementation, matplotlib for graph building and data visualization in real time.

The proposed architecture has flexibility to be modified in accordance with actual individual for every laboratory requirement. The multi-modular approach enables us to integrate new devices with minimal efforts of new driver and communication protocol implementation.

This system is flexible enough to be extended to fully automatic measuring system.

References

1. SCPI Consortium, Standard Commands for Programmable Instruments (SCPI). USA, 1999
2. *IEEE,* International Standard EEE 488.1, *2004*
3. AARØEN, O, KIÆR, H, RICCARDI, E. PYVISA: Visualization and Analysis of path sampling trajectories. *J Comput Chem*. 2021; 42: 435– 446. https://doi.org/10.1002/jcc.26467
4. JOHN M. PIEPER.: SCPI and VISA a valuable combination. (2002)

LUMINISCENȚA SOLUȚIEI COLOIDALE ALE PUNCTELOR CUANTICE DE SnS_2

Ecaterina CRISTEA[1,2]*

[1] *Laboratorul de spectroscopie optică și comunicații optice, Centrul Național de Studiu și Testare a Materialelor, Universitatea Tehnică a Moldovei, doctorandă an. 3*
[2] *Departamentul Telecomunicații și Sisteme Electronice, Facultatea Electronică și Telecomunicații, Universitatea Tehnică a Moldovei, lect. univ.*

*Autorul corespondent: Ecaterina CRISTEA, e-mail ecaterina.cristea@srco.utm.md

Coordonator științific: Victor ZALAMAI, dr., conf. șt., *Laboratorul de spectroscopie optică și comunicații optice, Centrul Național de Studiu și Testare a Materialelor*

Rezumat. *Studiile luminiscenței cristalelor de SnS_2 au rafinat structura benzilor la decalajul minim între zone. Intervalul minim de bandă este determinat de tranziții directe de la punctul L la punctul Γ al zonei Brillouin (Eg = 2.324 eV la 300 K). O soluție coloidală din punctele cuantice de SnS_2 luminescent la 300 K pe o gamă largă de energie. Schimbând dimensiunea punctelor cuantice, este posibilă modificarea regiunii maximumului luminiscenței. Principala contribuție la luminiscența punctelor cuantice de SnS_2 se datorează tranzițiilor directe în centrul zonei Brillouin.*

Cuvinte cheie*: Sulfura de staniu, punctele cuantice, fotoluminescența, soluția coloidală.*

Introducere

De la descoperirea grafenului în 2004 [1], materialele monoatomice și materialele cu o grosime de mai mulți atomi au atras atenția din ce în ce mai mult datorită proprietăților lor mecanice și electronice excelente. Absența legăturilor atârnate pe suprafețe mari specifice asigură o disipare scăzută a sarcinii în grafen și alte materiale bidimensionale (2D) non-grafen, ceea ce le face potrivite pentru aplicații în electronică de viteză mare, optoelectronică, senzori, generarea și stocarea energiei [2]. Cu toate acestea, utilizarea grafenului în electronică este limitată din cauza absenței de bandă interzisă [1]. În acest sens, în ultimii ani, au fost propuse și studiate intens alternative de grafen, care au o structură bidimensională similară, dar cu bandă interzisă [3]. Printre astfel de materiale, calcogenurile și dicalcogenurile din metale de tranziție prezintă o structură stratificată cu atomi legați unul de celălalt formând straturi 2D fără legături suspendate, straturile fiind legate între ele prin forțe slabe de van der Waals.

Semiconductori din grupa IV-VI cum sunt desulfura de staniu (SnS_2) și diseleniura de staniu ($SnSe_2$) sunt exemple tipice de s-p calcogenuri metalice stratificate care au fost recent propuse pentru a fi utilizate în următoarele aplicații: memorie de fază [4], tranzistoare cu efect de câmp [4], diode cu peliculă subțire [6], fotodetecție rapide [7], baterii litiu și sodiu-ion [8], supercondensatori [9], separare a apei [10], și detectarea gazelor [11]. De menționat că acești compuși sunt caracterizați printr-o distanță mare între straturi (d = 0,5899 nm în SnS_2), ceea ce face ușoară intercalarea atomilor și moleculelor.

Pe lângă structurile bidimensionale ca grafenul, structurile zero-dimensionale ca punctele cuantice (QD) sunt, de asemenea, studiate în mod activ. În special, se propune utilizarea punctelor cuantice bazate pe SnS_2 în celulele solare hibride [12]. Efectele de dimensiune cuantică joacă un rol important în proprietățile optoelectronice ale sistemelor optice bazate pe puncte cuantice. De exemplu, prin schimbarea dimensiunii unui punct cuantic, se poate controla lungimea de undă a radiației emise de un punct dat. Astfel, punctele cuantice SnS_2 are interes practic ca materiale luminiscente.

Metodele experimentale

Cristalele de SnS_2 au fost crescute prin metoda transportului de gaz în fiole de cuarț în forma plăcii de diferite dimensiuni, care au fost ușor secționate, ceea ce a făcut posibilă obținerea de suprafețe oglindă cu dimensiuni până la 5x7 mm și plăci de diferite grosimi 10 μm - 3 mm. Suprafețele majorității plăcilor erau paralele cu axa cristalografica *b*. Punctele cuantice au fost obținute mai întâi prin măcinarea mecanică a cristalelor obținute la dimensiunea de sute de micrometri, iar apoi au fost măcinate cu ajutorul ultrasunetului deja la dimensiuni de zeci de nanometri. Structurile bidimensionale obținute pe bază de SnS_2 au fost dizolvate în apă. Soluția coloidală rezultată a fost studiată prin metode luminiscente.

Spectrele de luminescență au fost măsurate cu ajutorul spectrometrului dublul cu apertură optică înaltă СДЛ-1 cu rețele de difracție 1200 linii/mm. Puterea luminii alcătuiește 1:3, iar dispersia liniară este de ordinul 10 Å/mm. Pentru excitare a fost folosit laserul He-Cd cu lungimea de undă 325 nm. Rezoluția alcătuiește 0.5 meV. Probele sunt fixate pe elementul de răcire a sistemei criogenice de ciclu închis LTS-22-C-330.

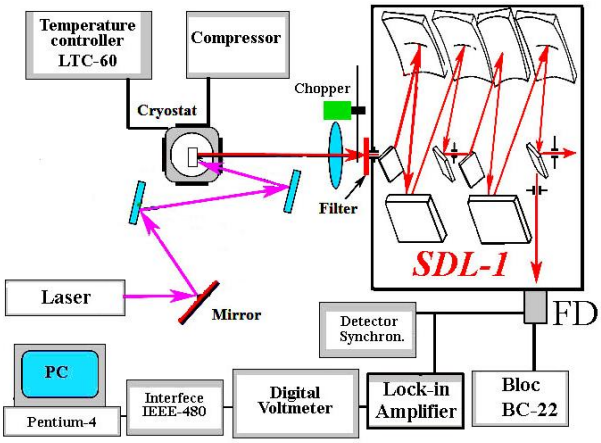


Figura 1. Schema instalației pentru măsurarea spectrelor fotoluminescenței

În Figura 1 este reprezentată schema instalației pentru măsurarea fotoluminescenței. Fluxul de lumină de la laserul He-Cd (linia de excitare 325 nm) cu ajutorul sistemei de oglinzi se focusează pe proba. Lumina laser excită în probă luminescența, care este colectată cu ajutorul condensorului şi se focusează la fanta spectrometrului. Semnalul de la fotomultiplicatorul ФЭУ-106 cu fotocatodul SbKNaCs, care lucrează în regimul de numărare a fotonilor, se transmite la calculatorul prin interfaţa IEEE-480.

Rezultate experimentale

Se știe că în proprietățile optice efectul de mărime cuantică se manifestă atât în spectre de absorbție, cât și în spectre de luminiscență, ducând la o modificare a lungimii de undă a luminiscenței, care corespunde energiei de recombinare a unei perechi electron-gol. Spectrul de luminiscență este întotdeauna deplasat în raport cu spectrul de absorbție către lungimi de undă lungi. Această schimbare se numește schimbarea Stokes. Această regulă se explică prin doi factori: - în nanocristalele cu structură hexagonală, diferite sub-benzi de goluri participă la tranzițiile optice în timpul excitației și emisiei; - pierderea unei părți din energia absorbită care se duce la mișcarea termică a moleculei. Pentru a obține QD-uri din SnS_2 s-a ales metoda pulberii, care are o serie de avantaje. Una dintre proprietățile fundamentale care distinge punctele cuantice coloidale de materialele semiconductoare tradiționale este posibilitatea existenței sub formă de soluții, sau mai precis, sub formă de soluri. Această proprietate oferă o gamă largă de posibilități de manipulare a unor astfel de obiecte și le face atractive pentru tehnologie.

Este cunoscut faptul că spectrul energetic al unui punct cuantic depinde de dimensiunea acestuia. Similar cu tranziția între nivelurile de energie dintr-un atom, atunci când purtătorii de sarcină

trec între nivelurile de energie într-un punct cuantic, un foton poate fi emis sau absorbit. Frecvențele de tranziție, de exemplu lungimea de undă de absorbție sau luminiscența poate fi controlată prin schimbarea dimensiunii punctului cuantic. Dependența spectrului de energie de dimensiune oferă un potențial uriaș pentru aplicarea practică a punctelor cuantice.

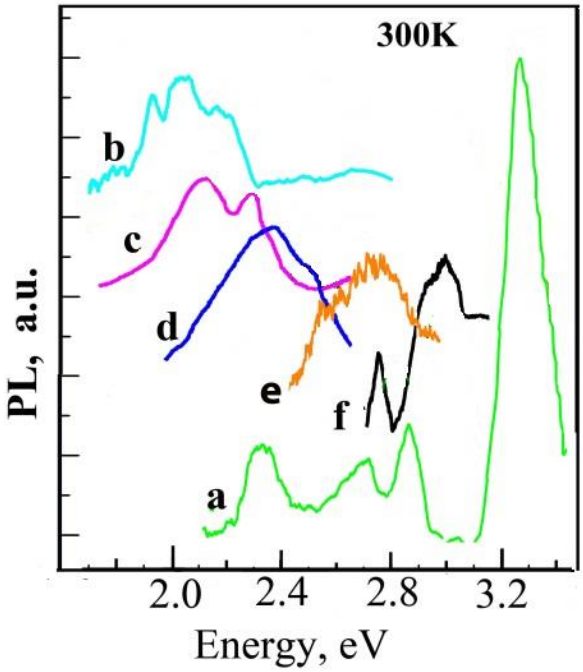


Figura 2. Spectrele de luminescență ale monocristalelor SnS_2 (curba a) și soluției coloidale de QD-uri de SnS_2 (curbe b-f) măsurate la temperatura camerei

În Figura 2 se prezintă spectrul de luminiscență ale monocristalul de SnS_2 excitat de o linie laser de 325 nm la 300 K (curba a). Liniile observate sunt asociate cu tranziții directe și indirecte de bandă-bandă. Luminescența unei soluții coloidale cu puncte cuantice obținute din acest cristal de SnS_2 și măsurate în diferite puncte ale soluției atunci când fasciculul laser se mișcă de sus în jos este prezentată în Fig. 2 curbe b-f. În acest caz, a fost îndeplinită condiția ca pulberea din cristalele de SnS_2 să fie mărunțită de un dispozitiv cu ultrasunete pentru a obține puncte cuantice nanodimensionale. Puncte cuantice mari prezintă luminescență în regiunea de 2 eV (spectre b și c), punctul de măsurarea cât mai aproape de fundul cuvei. Pe măsură ce fasciculul laser s-a deplasat în partea superioară a celulei cu soluția coloidală, maximele de luminescență s-au deplasat către energii mai mari (spectre d, e, f). Considerăm că spectrele de luminescență ale soluției coloidale cu QD de SnS_2 arată cel mai intens maxim de luminiscență la 300 K datorită tranzițiilor directe de la banda de conducție la banda de valență din centrul zonei Brillouin. Presupunem că tranzițiile directe au contribuția principală la luminescența punctelor cuantice bazate pe SnS_2.

Concluzii

A fost obţinută soluţie coloidală din QD de SnS_2 dizolvate în apă. Punctele cuantice au fost obținute prin măcinarea cu ultrasunete a cristalelor de SnS_2 sub formă de pulbere. Soluția rezultată a prezentat luminiscență într-o gamă largă de energie. În funcție de dimensiunea punctelor cuantice, lungimea de undă a luminii emise variată. În spectrele de luminescență ale soluției din QD-uri de SnS_2, apar maxime intense de luminiscență datorită tranzițiilor directe în centrul zonei Brillouin. Presupunem că tranzițiile directe au contribuția principală la luminescența punctelor cuantice.

Mulţumiri. Această lucrare a fost elaborată în cadrul proiectului bilateral cu cifrul 22.80013.5007.4BL.

Referinţe

1. NOVOSELOV, K.S., GEIM, A.K., MOROZOV, S.V., JIANG, D., ZHANG, Y., DUBONOS, S.V., GRIGORIEVA, I.V., FIRSOV, A.A. Electric field effect in atomically thin carbon films. In: *Science*, 2004, 306, pp. 666-669.
2. CASTRO NETO, A.H., GUINEA, F., PERES, N.M.R., NOVOSELOV, K.S., GEIM, A.K. The electronic properties of graphene. In: *Review in Modern Physics*, 2009, 81, pp. 109-162.
3. WANG, F., WANG, Z., WANG, Q., WANG, F., YIN, L., XU, K., HUANG, Y., HE, J. Synthesis, properties and applications of 2D non-graphene materials. In: *Nanotechnology*, 2015, 26, 292001 (26pp).
4. SUN, M., HU, Y., SHEN, B., ZHAI, J., SONG, S., SONG, Z. Si/$SnSe_2$ multilayer films for phase change memory applications. In: *Integrated Ferroelectrics*, 2012, 140 pp. 1-7.
5. PAN, T.S., DE, D., MANONGDO, J., GULOY, A.M., HADJIEV, V.G., LIN, Y., PENG, H.B. Field effect transistors with layered two-dimensional SnS_2xSex conduction channels: effects of selenium substitution. In: *Applied Physics Letters*, 2013, 103, 093108(pp.5).
6. SANCHEZ-JUAREZ, A., TIBURCIO-SILVER, A., ORTIZ, A. Fabrication of SnS_2/SnS heterojunction thin film diodes by plasma-enhanced chemical vapor deposition. In: *Thin Solid Films*, 2005, 480-481, pp. 452-456.
7. FAN, C., LI, Y., LU, F., DENG, H.-X., WEI, Z., LI, J. Wavelength dependent UV-Vis photodetectors from SnS_2 flakes. In: *RSC Advances*, 2016, 6, pp. 422-427.
8. WANGA, L., ZHUO, L., YUA, Y., ZHAOA, F. High-rate performance of SnS_2 nanoplates without carbon-coating as anode material for lithium ion batteries. In: *Electrochimica Acta*, 2013, 112, pp. 439-447.
9. CHAUHAN, H., SINGH, M.K., KUMAR, P., HASHMI, S.A., DEKA, S. Development of SnS_2/RGO nanosheet composite for cost-effective aqueous hybrid supercapacitors. In: *Nanotechnology*, 2017, 28, pp. 025401.
10. SUN, Y., CHENG, H., GAO, S., SUN, Z., LIU, Q., LIU, Q., LEI, F., YAO, T., HE, J., WEI, S. Freestanding tin disulfide single-layers realizing efficient visible-light water splitting. In: *Angewandte Chemie International Edition*, 2012, 51, pp. 8727-8731.
11. OU, J.Z., GE, W., CAREY, B., DAENEKE, T., ROTBART, A., SHAN, W., WANG, Y., FU, Z., CHRIMES, A.F., WLODARSKI, W., RUSSO, S.P., LI, Y.X., KALANTAR-ZADEH, K. Physisorption-Based charge transfer in two-dimensional SnS_2 for selective and reversible NO_2 gas sensing. In: *ACS Nano*, 2015, 9, pp. 10313-10323.
12. TAN, F., QU, S., WU, J. Preparation of SnS_2 colloidal quantum dots and their application in organic/inorganic hybrid solar cells. In: Nanoscale Research Letters, 2011, 6, pp, 298-304.

РЕАЛИЗАЦИЯ ДОЗАТОРА ДЛЯ ПРОХЛАДИТЕЛЬНЫХ НАПИТКОВ НА МИКРОКОНТРОЛЛЕРЕ STM32 С ПОМОЩЬЮ КОНЕЧНОГО АВТОМАТА

Татьяна МАСЛОВА

Департамент Микроэлектроники и Биомедицинской Инженерии, группа MN-191, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор-корреспондент: Татьяна Маслова, tatiana.maslova@mib.utm.md

Координатор/ научный руководитель: Адриан БЫРНАЗ, докторант,
Центр Нанотехнологий и Наносенсоров, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Аннотация. *В данной работе описан и реализован аппарат для дозирования жидкостей: в данном случае прохладительных напитков. Данный аппарат дозирует напитки: в зависимости от фиксированной дозы, заданной собственником аппарата (в миллилитрах), и желаемой дозы, задающейся клиентом - пользователем.*

Ключевые слова*: Дозатор, диспенсер, Embedded system, STM32.*

Введение

Диспенсер (дозатор) – универсальное устройство и его можно использовать для дозирования разных жидкостей: воды, напитков, мыла, медицинских растворов; и не обязательно жидкостей: драже, снеки, бумажные полотенца, салфетками и. т. п. Если провести анализ похожих аппаратов – можно сделать вывод, что они уже активно используются повсеместно: кофе машины, снековые вендинговые аппараты, почтоматы, банкоматы. В данной работе будет описан аппарат для дозирования прохладительных напитков. На Рис. 1 представлена общая блок схема устройства:

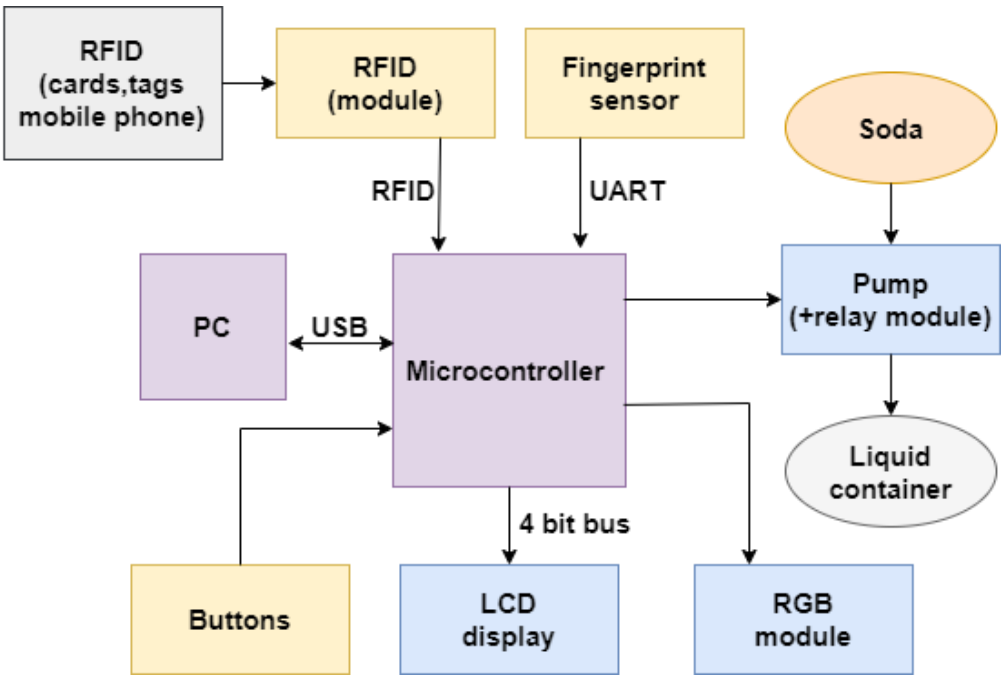


Рисунок 1. Блок схема устройства

Основная часть

На Рис. 2 представлена блок схема устройства. Части схемы, которые выполняют общие функции объединены в блоки. Всего 5 блоков, каждый из которых выполняет следующие функции:

Блок 1 – “Microcontroller” - главный блок устройства. Он осуществляет взаимодействие между периферийными устройствами, управляет всеми процессами устройства, осуществляет контроль данных и расчет необходимых значений. В данном проекте используется микропроцессор stm32G071 [1] из семейства 32-битных микроконтроллеров STM32G0 с процессорным ядром Cortex-M0+ производства компании STMicroelectronics. Обладает высокой рабочей частотой ядра до 64 МГц и значительной производительностью 0,8 - 1,27 DMIPS/МГц. Имеет два SPI, два I2C, четыре USART, 12-ти битный аналогово-цифровой преобразователь, два 12-ти битных цифро-аналоговых преобразователя, 14 таймеров (из которых 5 общего назначения).

Блок 2 - периферийные устройства для авторизации пользователя. Для авторизации можно использовать RFID [2] карты, метки, а также датчик отпечатка пальца [3]. Протокол общения RFID и микроконтроллера: SPI; датчика отпечатка пальца и микроконтроллера: USART. У карты и метки один общий ID пользователя. При этом каждый пользователь может привязать к своему аккаунту несколько отпечатков пальцев.

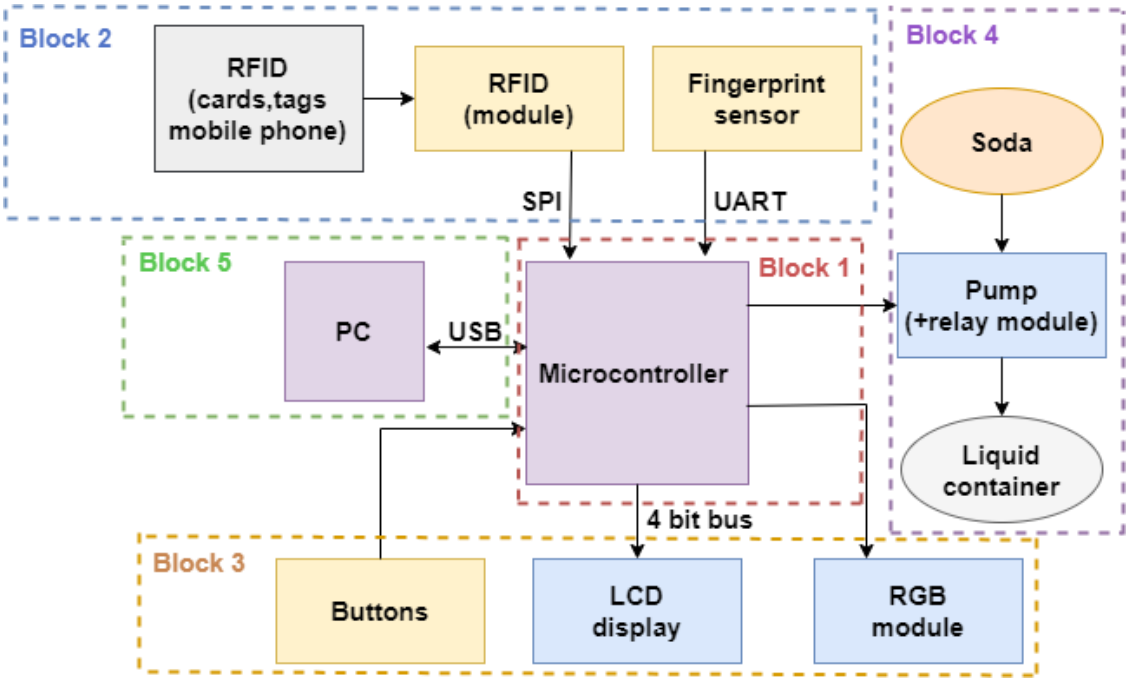


Рисунок 2. Блок схема устройства

Блок 3 – операторская панель или HMI (Humane Machine Interface). Состоит из 3-х кнопок [4], LCD-дисплея 20x4[5], RGB-модуля [6]. С помощью кнопок (“вправо”,“ОК”,”влево”) и экрана можно задавать количество и выбирать тип напитка. При обработке операций RGB-модуль подсвечивается определённым светом RGB-модуль, например: зелёным (успешное выполнение операции), красным (ошибка, например при неправильной попытке аутентификации)

Блок 4 – распределение жидкости(содовой). Наливает напиток в зависимости от выбора пользователя. Состоит из насосов [7] (подключаются через Реле- модуль [8]), ёмкости с напитком и ёмкости для налитого напитка (например стаканчик).

Блок 5 – блок распределения прав пользователей. Подключением через usb происходит устранения неполадок и изменения настроек с компьютера.

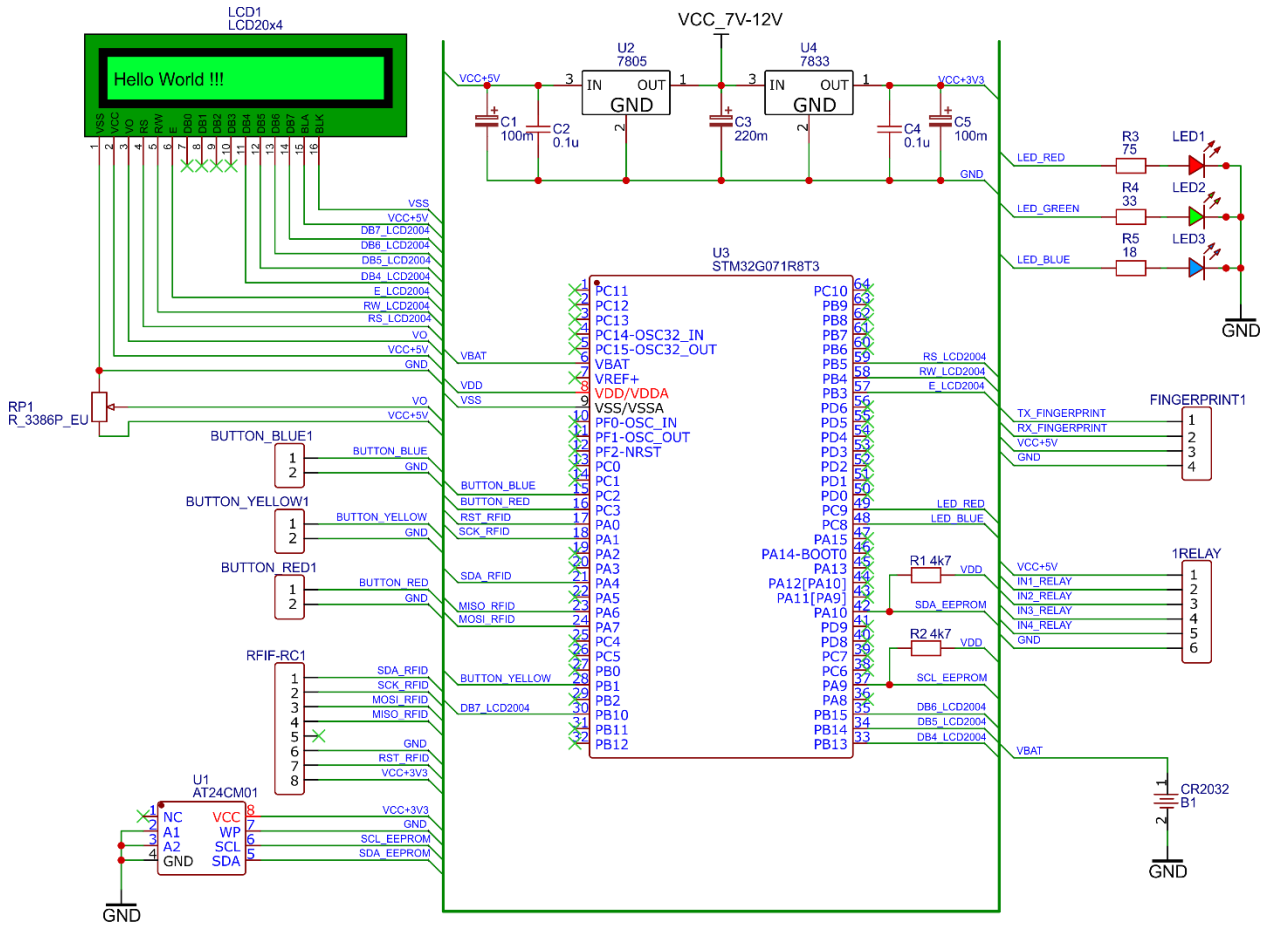


Рисунок 3. Электрическая принципиальная схема

На основе блок-семы была разработана электрическая принципиальная схема устройства, она представлена на Рис. 3.

Устройство работает следующим образом:

1. При включении устройства на экране отображается приветствие.
2. Далее идёт режим ожидания авторизации пользователя.
3. После прохождения авторизации с помощью кнопок пользователь может выбрать тип желаемого напитка, затем необходимое количество. Если это окончательный выбор пользователь нажимает кнопку “ОК” и происходит наполнение стаканчика выбранным напитком – успешное выполнение операции. Далее переход в режим ожидания авторизации следующего пользователя. Если же выбор не окончателен, и пользователь решил изменить тип напитка и задать другое количество – он может отменить текущую операцию. При одновременном нажатии двух крайних кнопок – происходит отмена и возвращение в режим ожидания авторизации пользователя. Затем необходимо авторизоваться повторно и сделать новый выбор. Если лимит желаемого напитка исчерпан – кнопка “ОК” для выбора данного напитка будет не доступна. Тогда пользователь может выбрать другой тип напитка, лимит которого не исчерпан. При условии исчерпания всех напитков при авторизации пользователь увидит на экране сообщение о том, что лимит всех напитков на сегодня исчерпан, затем устройство снова перейдёт в режим ожидания авторизации следующего пользователя.

На Рис. 4. изображена схема алгоритма для данного устройства.

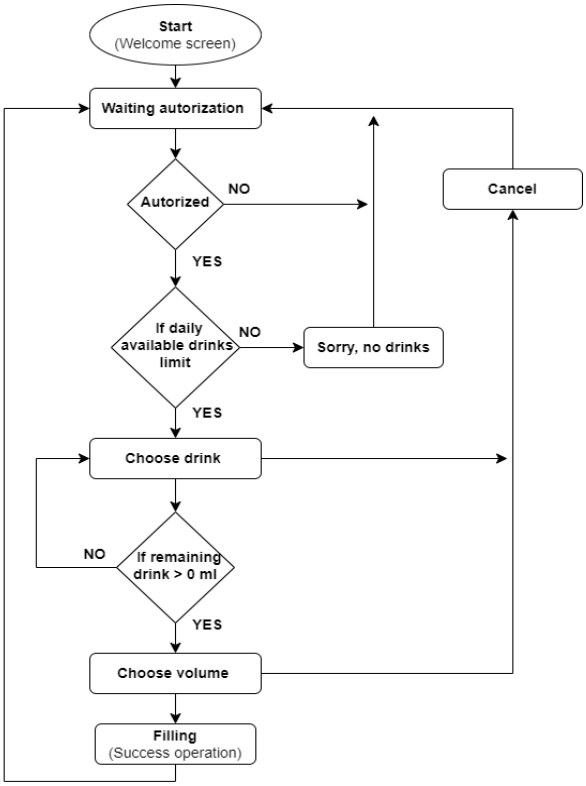


Рисунок 4. Схема алгоритма работы прибора

Выводы

Данный прибор может быть реализован для разных целей (в зависимости от продукта, который будет дозироваться) например: дозировать прохладительные напитки в кафе или торговом центре; или дозировать сок для больного диабетом ребёнка в домашних условиях (родительский контроль). Разумеется, в зависимости от целей могут отличаться используемые насосы и трубки для жидкости. Данный прибор позволяет дозировать несколько типов напитков. Количество каждого напитка так же может быть индивидуальным и может быть отредактировано с компьютера при подключении устройства через USB кабель. Устройство работает в реальном времени по модели конечного автомата. и может быть реализовано по приведённому выше алгоритму.

Благодарности. Автор благодарен AROBS Software за оказанную поддержку при реализации данного проекта.

Список используемой литературы

1. STM32G071.[online].[accessed:04/03/2023] Accessible: https://www.compel.ru/lib
2. МодульRFID.[online].[accessed:04/03/2023] Accessible:https://3d-diy.ru/wiki/arduino-moduli/rfid-modul-rc522/
3. Датчик отпечатка пальца.[online].[accessed:04/03/2023] Accessible: https://iarduino.ru/shop/Expansion-payments/skaner-otpechakov-palcev.html
4. Кнопки.[online].[accessed:04/03/2023] Accessible:https://all-arduino.ru/product/moduli-knopok-dlya-arduino/
5. LCD-дисплей 20x4.[online].[accessed:04/03/2023] Accessible: https://iarduino.ru/shop/displays/lcd2004-simvolnyy-displey-golubaya-podsvetka-20x4.html
6. RGB-модуль.[online].[accessed:04/03/2023] Accessible:https://robotica.md/ru/modul-rgb-led
7. Насос.[online].[accessed:04/03/2023] Accessible:https://robotica.md/ru/pompa-apa-3-6V
8. Реле -модуль.[online].[accessed:04/03/2023] Accessible:https://robotica.md/ru/releu-4-canale

SISTEM DEPARTAMENTAL DE MANAGEMENT A COMPONENTELOR ELECTRONICE

Rodica POPA[1*], Maxim CHIRIAC[2]

[1]*Departamentul Microelectronică şi Inginerie Biomedicală, Facultatea Calculatoare Informatică şi Microelectronică, Universitatea Tehnică a Moldovei, Chişinau, Republica Moldova*
[2]*Centrul de Nanotehnologii şi Nanosenzori, Departamentul Microelectronică şi Inginerie Biomedicală, Facultatea Calculatoare Informatică şi Microelectronică, Universitatea Tehnică a Moldovei, Chişinau, Republica Moldova*

*Autorul corespondent: Popa Rodica, popa.rodica@mib.utm.md

Coordonator: Nicolai ABABII, doctor, Centrul de Nanotehnologii și Nanosenzori, Departamentul Microelectronică și Inginerie Biomedicală, Facultatea Calculatoare Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chişinau, Republica Moldova

Rezumat. *În lucrare este descris conceptul de creare și funcționare a unei baze de date, care va gestiona componentele electronice în cadrul instituțiilor publice. Sunt expuși pașii de creare a interfeței grafice pentru gestionarea bazei de date, utilizând limbajul de programare SQL (limbajul structurat de interogare – Structured Query Language). Sunt descrise câmpurile de bază necesare pentru buna funcționare a bazei de date conform conceptului. La finalizarea creării bazei de date, a fost elaborată interfața grafică pentru comoditatea operării acesteia de către utilizatorii finali.*

Cuvinte cheie: *Bază de date, utilizator, componente, interfață, electronică*

Introducere

O bază de date este o colecție ordonată de informații sau date structurate ce sunt stocate electronic pe un sistem informațional. Informațiile în acest sistem sunt stocate sub formă de rânduri și coloane ce formează un tabel, fiind ușor gestionate, modificate, actualizate, monitorizate și organizate. Părțile componente a unei baze de date sunt: hardware, software, protocolul de comunicare (legătura dintre componentele mașinii și utilizator), proceduri (instrucțiunile utilizate), limbajul de acces la baza de date [1].

Obiectivele principale pentru apariția, evoluția și stocarea datelor într-o bază de date sunt:

1. *Evitarea redundanței și inconsistenței datelor* – capacitățile enorme ale calculatoarelor se folosesc nu numai pentru stocarea volumelor mari de date, dar și pentru controlul verificării dacă datele existente sunt consistente și nu sunt date multiplicate;
2. *Facilitarea accesului la date* – presupune ca baza de date să aibă componente specializate pentru diferite operații de utilizare;
3. *Asigurarea accesului securizat la date* – protecția datelor se implementează prin două aspecte: securitate și integritate;
4. *Partajabilitatea datelor* - se referă nu numai la aspectul asigurării accesului mai multor utilizatori la aceleaşi date, ci şi la posibilitatea dezvoltării unor aplicaţii fără a se modifica structura bazei de date [2].

Pentru asigurarea securității informațiilor, bazele de date dispun de o serie de nivele de acces, ce pot fi accesate numai de utilizatorii cu privilegiu superior sau egal nivelului de importanță respective, restricționând accesul la informații și evitându-se modificările neautorizate. Utilizatorii se clasifică în patru grupe mari: administratori, programatori, utilizatori cu pregătire specială, utilizatori obișnuiți. Administratorii vor gestiona baza de date, aceștia nu necesită pregătire specială pentru a o scrie, însă ei sunt persoanele de bază ce dispun de idea cum va fi această bază de date proiectată. Programatorii în urma discuției și preluarea cerințelor de la administrator creează acel schelet a bazei de date și gestionează în câmpuri și coloane informația necesară spre introducere. Utilizatorii cu pregătire specială sunt utilizatorii care dispun de anumite privilegii de acționare a bazelor de date în

comparație cu utilizatorii simpli, spre exemplu utilizatorii experimentați pot vizualiza informații mai detaliate despre cele stocate în baza respectivă. Utilizatorii simpli sunt utilizatorii ce au posibilitatea de a vizualiza informațiile reduse despre datele introduse în acel sistem. Aplicațiile de baze de date sunt mult mai complexe decât alte categorii de aplicații, deoarece trebuie să fie realizată atât interfața cu sistemele de gestiune a bazelor de date, pentru introducerea și extragerea datelor, cât și interfața cu utilizatorii, astfel încât utilizatorii să poată efectua diferite operațiuni [3].

Partea experimentală

Planificarea unei baze de date reprezintă elaborarea unor activități ce permit realizarea efectivă și eficientă a pașilor de proiectare a unei baze de date. Pașii proiectării unei aplicații de bază de date sunt: etapa de analiză, etapa de proiectare și etapa de realizare.

Etapa de analiză reprezintă activitățile întreprinse pentru rezolvarea problemei, stabilirea responsabilităților fiecărui utilizator [4].

Etapa de proiectare definește proprietățile obiectivelor, evenimentele de care este responsabilă baza de date.

Etapa de realizare constă în crearea efectivă a obiectivelor cu structura și proprietățile proiectate, fiind urmată de programarea, codificarea și testarea funcționalității aplicației pe baza unor date.

După finalizarea analizei, se proiectează obiectivele bazei de date, începând cu tabelele și corelațiile dintre acestea. Tabelele constituie obiectele bazei de date destinate păstrării informației. Pentru gruparea datelor în tabele, sunt respectate unele reguli: pentru fiecare entitate a sistemului se creează câte un tabel, structura tabelului cuprinde atribuțiile sale reprezentate prin câmpuri.

Un scop major al sistemelor bazelor de date este oferirea utilizatorilor o viziune abstractă asupra informațiilor, această operațiune fiind numită *abstractizarea datelor* reprezentată conceptual în fig. 1.

Abstractizarea datelor reprezintă posibilitatea sistemului de a ascunde anumite detalii asupra modului în care sunt păstrate și memorate datele, susținând utilizatorul de a cunoaște întreaga structură a bazei de date. Structura bazei de date cuprinde trei nivele de abstractizare:

- Nivelul intern;
- Nivelul conceptual;
- Nivelul extern.

Nivelul intern, sau fizic, reprezintă modalitatea efectivă în care datele sunt scrise pe suportul de stocare: disc optic, disc magnetic, bandă magnetică. Structura datelor este descrisă foarte detaliat, fiind accesibilă numai administratorilor și inginerilor de sistem. Nivelul intern cuprinde două părți principale: un set de programe ce interacționează cu sistemul de operare pentru îmbunătățirea managementului bazei de date; fișierele stocate în memoria internă a calculatorului.

Nivelul conceptual, sau global, este superior celui fizic, descriind datele ce sunt memorate, precum și relațiile care există între acestea. Toți utilizatorii își exprimă nevoile de date la nivel conceptual, prezentându-le administratorului bazei de date, acesta fiind cel care are o viziune globală necesară satisfacerii tuturor cerințelor informaționale [5].

Nivelul extern este ultimul nivel de abstractizare, unde poate fi descrisă o bază de date. Descrie acea parte a bazei de date care este relevantă pentru fiecare utilizator. Nivelul extern include numeroase viziuni externe. Fiecare viziune descrie o parte a schemei logice de care este interesat utilizatorul în particular. Concomitent, viziunea ascunde de utilizator toate celelalte elemente ale schemei logice, astfel aplicațiile propun utilizatorilor viziuni parțiale și personalizate de date. O interogare exprimată de un utilizator, într-un limbaj acceptabil de sistem, este analizată din punct de vedere sintactic și semantic. Obiectele citate trebuie să fie cunoscute în schema externă a utilizatorului.

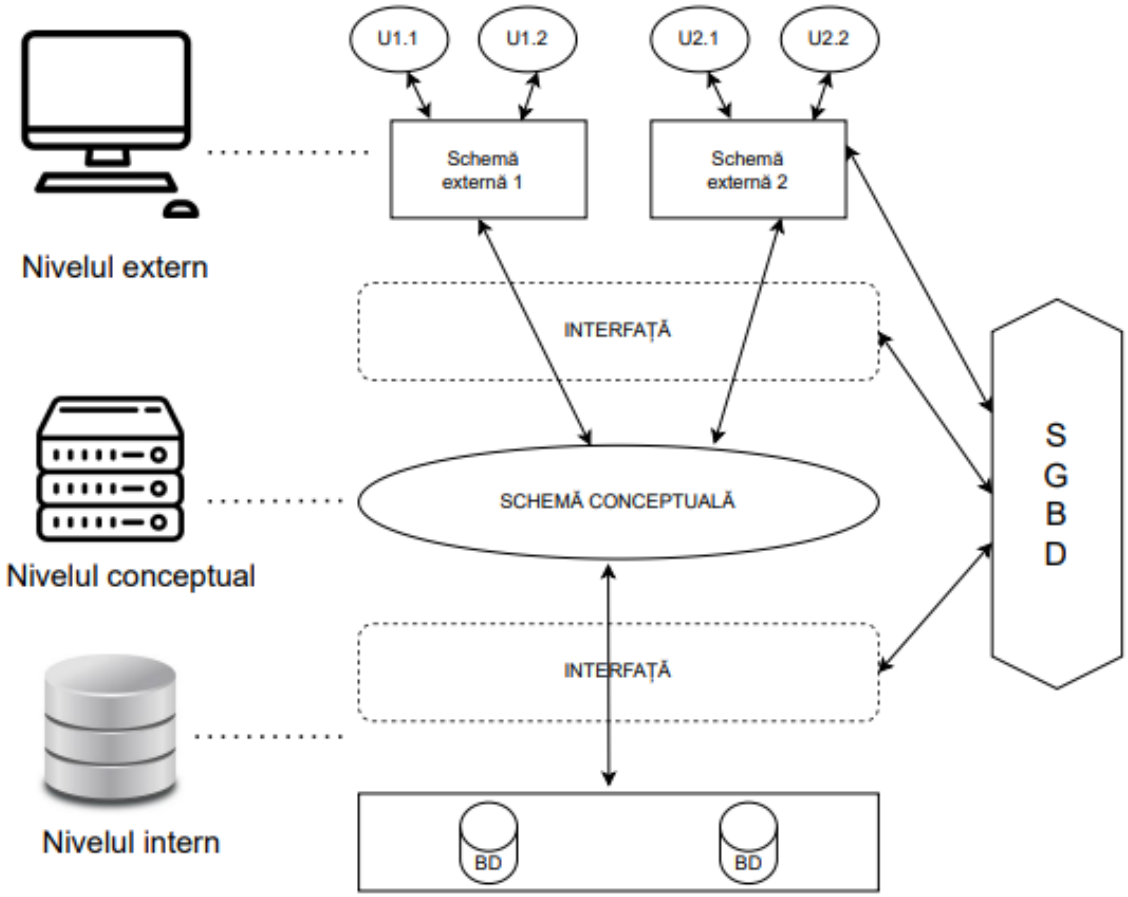


Figura 1. Reprezentarea nivelelor de abstractizare

Rezultate

În fig. 2 este reprezentat diagrama bazei de date implementate în cadrul tezei de licență. Aranjarea și construcția acestei diagrame este foarte importantă, deoarece de aceasta depinde întreaga calitate a bazei de date [6].

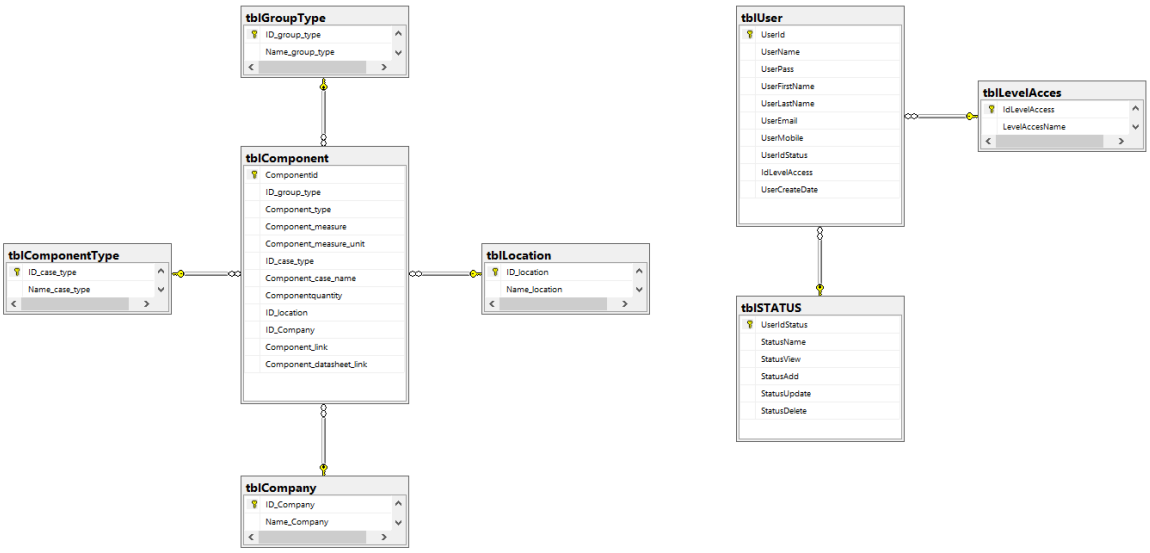


Figura 2. Diagrama Bazei de Date

Tabelul ***tblComponent*** are următoarele câmpuri: *Componentid; ID_group_type; Component_type; Component_measure_unit; ID_case_type; Component_case_name; Componentquantity; ID_location; ID_Company; Component_link; Component_datasheet_link.* Fiecare câmp are semnificația și valoarea sa, dar toate împreună formează o totalitate de elemente caracteristice componentelor. Câmpurile ce conțin parametri unici pentru informațiile din tabel, formează tabele separate, dar reiese din tabelul principal. De la câmpul *ID_case_type*, se formează un nou tabel ***tblComponentType***, ce este cu referință la tabelul ***tblComponent***, ce separă componentele după tehnologia de montare. În cazul ID-ului „1" componenta va avea tehnologia de montare de tip THT (Through Hole Technology- Tehnologia de montare prin găurire), iar când ID-ul este „2'' tehnologia de montare va fi SMD (Surface Mount Device- Tehnologia de montare la suprafață). Câmpul *ID_location* determină crearea unui nou tabel, ***tblLocation***, ce conține informații despre locația componentei interesate. *ID_Company* este câmpul ce duce la formarea unui nou tabel, **tblCompany**, ce conțin denumirile companiilor ce au livrat componenta anumită. Câmpul *ID_group_type* necesită formarea tabelului ***tblGroupType***, ce împarte componentele în *active* sau *pasive.* Denumirea componentei este descrisă în câmpul *Component_type,* unitatea de măsură este exprimată în cîmpul *Component_measure_unit,* carcasa este notată în *Component_case_name,* numărul componentelor respective se notează în *Componentquantity,* link-ul paginii procurării și datasheet-ului se înscriu în câmpul *Component_link* și *Component_datasheet_link.*

Tabelul **tblUser** a fost creat folosind același principiu ca în tabelul componentelor. Acest tabel are următoarele câmpuri: *UserID, UserName, UserPass, UserFirstName, UserLastName, UserEmail, UserMobile, UseridStatus,idLevelAccess, UserCreateDate.* Câmpul *idLevelAccess* este exprimat prin tabelul ***tblLevelAcces***, unde utilizatorul se rezumă la informațiile și efectuarea unor operațiuni specifice ierarhiei nivelului de acces. Statutul utilizatorului este exprimat în tabelul ***tblSTATUS,*** ce provine din câmpul *UseridStatus* al tabelului principal ***tblUser***.

Tabelul tblSTATUS este format din câmpurile: UserIdStatus, StatusName, StatusView, StatusAdd, Statusupdate, StatusDelete. Fiecare câmp permite efectuarea unor operațiuni: vizualizare, adăugare, modificare, ștergere. Acest tabel va avea trei tipuri de utilizatori: practicant; inginer; inginer șef. Practicant va fi persoana care va avea acces la baza de date doar pentru vizualizarea informațiilor despre compontente din tabelul tblComponent. Inginerul va deține posibilitatea de a adăuga și modifica informațiile referitoare la componente, iar inginerul șef va fi în stare să creeze utilizatori noi cu drepturi de toate cele trei tipuri specificate mai sus. Deasemenea acesta va fi unicul nivel în stare de a șterge informațiile despre componente și utilizatori în cazul în care va fi nevoie.

Interfața grafică reprezentată în figura 3 este formată din partea de control poziționată în partea dreaptă și partea de afișare a informației înregistrată în baza de date.

Partea de control fiind formată din forma de logare, forma de căutare, forma pentru adăugare, numită **InginerForm** și forma administratorului numită **AdminForm.** Forma de logare determină accesul persoanei logate ce duce la activarea și accesarea formelor de căutare, InginerForm, AdminForm. Forma de căutare este activă pentru orice utilizator autorizat de sistem, iar formele InginerForm și AdminForm sunt accesate de utilizatori cu privilegii mai avansate. Căutarea componentelor este posibilă după criteriile:Tip componentă; valoare; locație. Utilizatorii cu drepturi de ingineri au posibilitatea de adăuga componente sau modificarea acestora, iar utilizatorii cu drepturi de administrator au posibilitatea de a șterge componente sau utilizatori, vizualizarea informațiilor despre o componentă mai detaliată, vizualizarea tuturor utilizatorilor, adăugarea sau modificarea acestora.

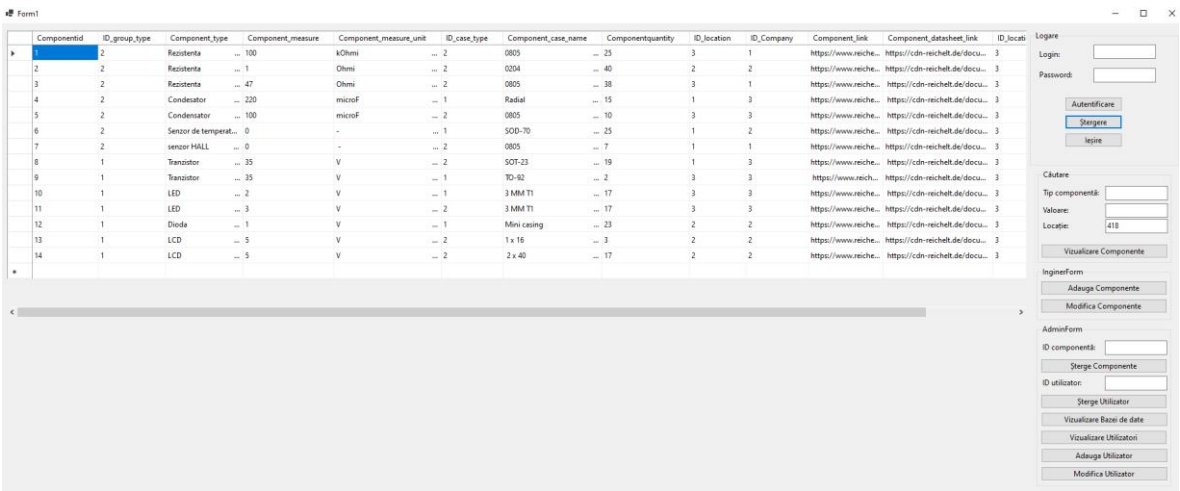


Figura 3. Interfaţa grafică

Concluzii

În acest articol este descris conceptul bazei de date propus spre elaborare în cadrul tezei de licență. Baza de date, este preconizată pentru gestionarea componentelor electronice, în instituții publice și private, cum ar fi centre medicale, spitale, companii private din domeniul electronicii. Sunt demonstrate rezultatele prealabile și etapa la care se află elaborarea acesteia la momentul actual.

Pentru o înțelegere mai adecvată a structurii bazei de date a fost creată diagrama bazei de date reprezentată în fig.3, fiind construită din tabele cu diferite cîmpuri ce conțin informații despre caracteristicile de bază a componentelor și a datelor despre utilizator.

Interfața grafică ghidează utilizatorii să introducă numai datele necesare, în dependență de nivelul de acces a persoanei logate.

În urma realizării lucrării au fost obținut o bază de date pentru gestionarea componentelor electronice într-un departament cu trei nivele de acces și interfață grafică.

Mulţumiri. Rodica Popa este recunoscătoare Departamentului Microelectronică și Inginerie Biomedicală a Universității Tehnice a Moldovei, pentru experiența oferită în cadrul studiilor la ciclul I (Licență).

Această lucrare a fost susținută parțial de Universitatea Tehnică din Moldova și Grantul ANCD-NARD Nr. 20.80009.5007.09 la UTM.

Referinţe

1. ORACLE, Что такое база данных?, [online]. [accesat: 26.01.2023]. Disponibil: https://www.oracle.com/cis/database/what-is-database/#nosql
2. К. Дж. Дейт ,Введение в системы баз данных , Москва ,Санкт-Петербург , Киев 2005
3. SUMOLOGIC, What is database security?, [online]. [accesat: 06.02.2023]. Disponibil: https://www.sumologic.com/blog/what-is-database-security/
4. SUPPORT.MICROSOFT, Noțiuni de bază despre proiectarea bazelor de date, [online]. [accesat: 06.02.2023]. Disponibil: https://support.microsoft.com/ro-ro/office/no%C8%9Biuni-de-baz%C4%83-despre-proiectarea-bazelor-de-date-eb2159cf-1e30-401a-8084-bd4f9c9ca1f5#:~:text=O%20bun%C4%83%20proiectare%20a%20bazei%20de%20date%20este%2C%20prin%20urmare,acurate%C8%9Bea%20%C8%99i%20integritatea%20informa%C8%9Biilor%20dvs.
5. SCALER, Data Abstraction in DBMS, [online]. [accesat: 09.02.2023]. Disponibil: https://www.scaler.com/topics/data-abstractions-in-dbms/
6. LEARN, Tables, [online]. [accesat: 09.02.2023]. Disponibil: https://learn.microsoft.com/en-us/sql/relational-databases/tables/tables?view=sql-server-ver16

SISTEM HOLOGRAFIC DE AFIȘARE A INFORMAȚIEI

Maxim CHIRIAC

Centrul de Nanotehnologii și Nanosenzori, Departamentul Microelectronică și Inginerie Biomedicală, Facultatea Calculatoare Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chişinau, Republica Moldova

Autorul corespondent: Maxim Chiriac, maxim.chiriac1@mib.utm.md

Coordonator: Nicolai ABABII, doctor, Centrul de Nanotehnologii și Nanosenzori, Departamentul Microelectronică și Inginerie Biomedicală, Facultatea Calculatoare Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chişinau, Republica Moldova

Rezumat. *Această lucrare prezintă o nouă modalitate de afișare a informației în scopuri educative și comerciale la un nivel mai înalt, neutilizat în larg consum în Republica Moldova. În articol este descrisă importanța utilizării* utilizării afișoarelor de informații *și ulterior etapele necesare pentru realizarea acestui dispozitiv și rezultatele obținute.*

Cuvinte cheie: *display, hologramă, 3D, afișor*

Introducere

O nouă tendință de utilizare a televizoarelor în scopuri de promovare a produselor sunt hologramele 3D [1]. Ecran holografic este o bucată de plexiglas complet transparentă, pe care se aplică o peliculă de retroproiecție, invizibilă pentru ochiul uman. Ecranul holografic vă permite să vedeți atât o imagine strălucitoare și contrastantă din fluxul de lumină, cât și tot ceea ce se întâmplă în spatele acestuia, creând astfel efectul unei „pseudo-holografii" - o imagine care plutește în aer. Concentrându-ne ochii pe imagine, vedem o imagine clară, contrastantă, iar dacă relaxăm focalizarea, putem privi prin imagine. Ecranul transparent a fost utilizat pe scară largă la expoziții și prezentări, deoarece această soluție este o modalitate inovatoare de demonstrare a informațiilor. Utilizarea tehnologiilor de proiecție holografică face posibilă realizarea unei iluzii realiste a unei imagini în mișcare, inscripție, obiect sau personaj animat care apare în fața vizitatorului ca de nicăieri, atrăgând atenția, fiind amintit și permițând evidențierea fundalului general. În plus, există posibilitatea de interactivitate. Instalarea senzorului de mișcare permite de a reda conținutul în două moduri - static și dinamic. Conținutul static este difuzat continuu de îndată ce senzorul observă schimbări în mediu, de ex. fie persoana trece pe lângă holoecran, fie merge până la el. În acest moment, începe redarea altui conținut, care conține de obicei informații mai detaliate despre ceva. Următoarea caracteristică interactivă este Kinect. Acest sistem transmite nu doar un semnal de mișcare, ci și volum, în urma căruia se poate schimba grafica cu gesturi, dacă este o prezentare. De asemenea, ca opțiune interactivă, este posibilă atingerea ecranului holografic. Acest lucru este posibil datorită unei pelicule tactile special aplicată pe partea din față a ecranului. În acest caz, ecranul se transformă într-o tabletă mare transparentă, care este potrivită nu numai pentru difuzarea videoclipurilor, vizualizarea fotografiilor, ci și pentru lucrul cu aplicații, colectarea informațiilor de contact și feedback. Domeniul principal de aplicare a ecranelor sunt vitrine transparente interactive de proiecție, figuri și logo-uri transparente, publicitate interactivă în aer, demonstrație de reclame, vizualizarea materialelor la conferințe, prezentări, întâlniri de afaceri, expoziții, informarea într-un mod inovator și multe altele, atragerea atenției asupra produsului. Datorită proprietăților unice ale tehnologiei holografice, aceste ecrane permit decorarea unui punct de vânzare, poate acționa ca un element de design luminos pentru o cafenea, restaurant, club de noapte, salon și chiar un birou de lucru și poate fi utilizat pentru a crea efecte optice în show-business. Tehnologia se bazează pe un astfel de fenomen fizic precum difracția luminii. Cu ajutorul acestuia, s-a reușit să se creeze imagini tridimensionale.

Partea experimentală

În cadrul acestui sistem pentru realizare a fost nevoie de un televizor cu rezoluție înaltă, consumabile necesare pentru carcasa sistemului, obiecte imprimate 3D și plexiglas.
Televizorul utilizat în cadrul acestui sistem holografic are o diagonală de 32 inch, rezoluție FHD (1920x1080).

În fig. 1 este reprezentat modelul 3D al sistemului holografic proiectat din diferite vederi. Carcasa acestuia este realizat din țeavă metalică cu dimensiunea de 40 x 20 mm, alunecător reglabil, placaj șlefuit și elementele de fixare. Carcasa are dimensiunea de 740 x 440 mm cu înălțimea de 500 mm. Pentru o vizualizare mai prounțată a imaginii de către afișorul holografic placajul șlefuit a fost vopsit în culoare neagră mat.

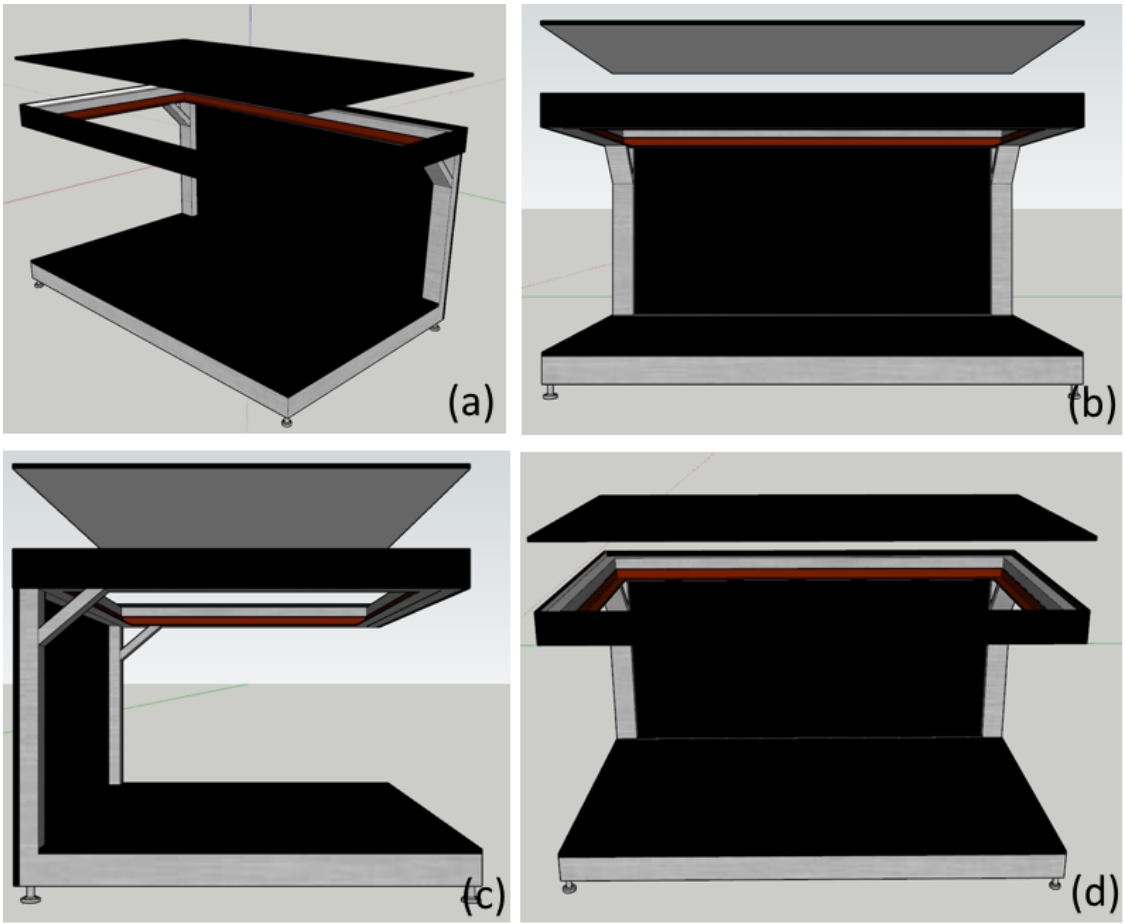


Figura 1. Modelul 3D a dispozitivului elaborat:
(a) vederea din unghi; (b) vederea din față;
(b) vederea din dreapta; (d) vederea de sus.

Obiectele imprimate 3D prezentate în fig. 2, au fost proiectate în softul gratuit SketchUp Free și imprimat la imprimanta 3D Creality. Modelul colțarului de fixare are o dimensiune exterioară de 20x20 mm, iar fixatorul central un diametru de 50x50 mm. Imprimarea acestor două obiecte a fost realizată la temperatura de 175 °C a filamentului, iar suportul inferior la temperatura de 50 °C. Calitatea obiectelor realizate a fost selectată standard cu factorul de umplere 100 %. Astfel timpul total de imprimare a acestor modele a durat 3 ore 45 minute.

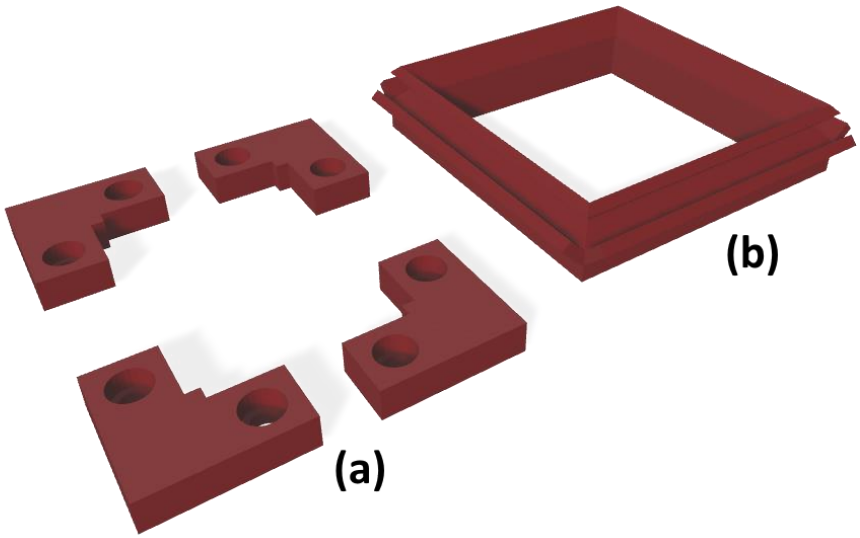


Figura 2. Modelele proiectate 3D: (a) colțare de fixare; (b) fixator central.

Rezultate

Hologramele 3D [2] sunt o soluție nouă și impresionantă pentru a explica produsele într-un mod inovator și într-un mod ușor de înțeles. Spre deosebire de formele convenționale de prezentare - cum ar fi pliantele, revistele informative, posturile radio, posturile TV, filmele 2D – utilizarea hologramelor permite transmiterea impresiilor mult mai realiste. Prin utilizarea dispozitivului holografic, imaginea digitală 2D devine o imagine virtuală vie a unui obiect real prezentat în 3D.

În fig. 3 este reprezentat dispozitivul real integral asamblat utilizând toate componentele descrise mai sus, și prelucrarea părții vizuale a placajului șlefuit cu culoare neagră. În fig. 3a este reprezentat dispozitivul cu vederea din față unde se observă fixatoarele reale a modelelor proiectate 3D reprezentate în fig. 2. În fig. 3b se observă colțare pe fixare din toate cele 4 părți ce sunt utilizate pentru a elimina destabilizarea carcasei la momentul deplasării acesteia. Cele 4 fețe a dispozitivului realizate din plexiglas permite redarea imaginii 2D sub formă de 3D cu un unghi de vizualizare 360º. Vizualizarea funcționării sistemului holografic este reprezentat în fig. 4.

Figura 3. Sistemul holografic asamblat real: (a) vederea din față; (b)vederea din colț.

Sistemul holografic [3] de afișare a informației prezintă produsele într-un mod unic. Produsele din beton pot fi combinate impresionant cu efecte vizuale precum focul, apa sau fumul. Totul depinde de video-ul ce va fi proiectat pe această hologramă și de creatorul acestui videou. Se poate crea un nou limbaj vizual care este atrăgător din punct de vedere vizual și se potrivește perfect cu produsul respectiv.

Acestea rezolvă problema prezentării corelației complexe într-un mod clar și ușor de înțeles. De exemplu, procesele de filtrare, fluxurile de fluide sau procesele chimice pot fi reprezentate realist pe o expoziție reală.

S-ar putea surprinde clientul într-un mod impresionant cu prezentarea 3D. Conținutul rămâne ancorat în memoria lui. Prin utilizarea tehnicii hologramei, compania asociată este, de asemenea, percepută de către privitor ca fiind inovatoare, ceea ce înseamnă un avantaj competitiv inestimabil.

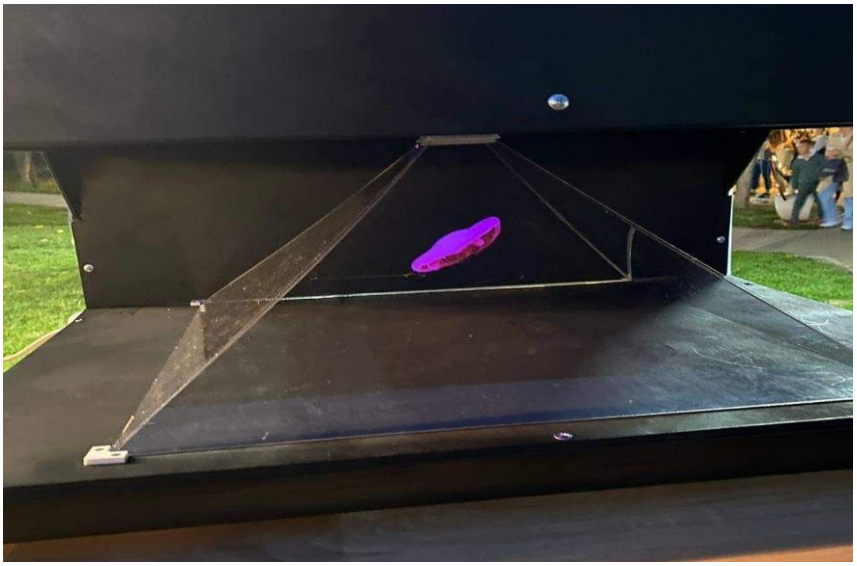


Figura 4. Redarea imaginii pe sistemul holografic.

Sistemele holografice [4] de afișare a informației 3D au o gamă diversă și foarte impresionantă de utilizări. Printre altele, ele pot fi folosite pentru: Târguri; Evenimente; Muzee; Expoziții; Prezentări de produse, etc. Pentru fiecare loc, ocazie sau scop există un sistem holografic adecvat. Hologramele 3D sunt potrivite pentru fiecare sector. Fie că este industrie, inginerie mecanică, farmaceutică, alimentație, automobile sau alte sectoare, gama de posibilități este practic nelimitată. Sistemul elaborat permite redarea imaginilor și informațiilor printr-un mod mai atractiv pentru vizualizatori.

Concluzii

În acest articol este descris dispozitivul deja expus spre vizualizare la evenimentul „NOAPTEA CERCETĂTORILOR EUROPENI” și expoziția „CREAȚIA DESCHIDE UNIVERSUL” care a captivat vizualizatorii cu informațiile difuzate pe acesta. Deci pentru a utiliza un sistem holografic este nevoie de un proiector de hologramă, care permite proiecția holografică 3D, conținutul de hologramă, care permite apariții libere în spațiu și publicul care va vizualiza conținutul propus.

Mulțumiri. Chiriac Maxim este recunoscător Centrului de Nanotehnologii și Nanosenzori din cadrul Departamentului Microelectronică și Inginerie Biomedicală a Universității Tehnice a Moldovei, pentru experiența oferită în cadrul studiilor la ciclul I (Licență) și ciclul II (Master), în special profesorului universitar, doctor habilitat, Lupan Oleg.

Referințe

1. HOLOGRAPHIC DISPLAYS [online]. [accesat: 13.02.2023]. Disponibil: https://www.realfiction.com/holographic-displays
2. What is a 3D hologram? [online]. [accesat: 13.02.2023]. Disponibil: https://magic-holo.com/en/what-is-a-3d-hologram/
3. Holographic Technology: Applications and How it Works [online]. [accesat: 13.02.2023]. Disponibil: https://www.analyticssteps.com/blogs/holographic-technology-applications-and-how-it-works
4. Holograms in Real Life: How the Technology Works and Industry Use Cases [online]. [accesat: 13.02.2023]. Disponibil: https://www.respeecher.com/blog/holograms-real-life-technology-works-industry-use-cases

EXFOLIEREA MATERIALELOR 2D SnS, SnS_2 ȘI SnSe PRIN METODE ȘI TEHNICI COST-EFECTIVE

Mihail BATIRI

Centrul Național de Studiu și Testare a Materialelor, Universitatea Tehnică a Moldovei, Chișinău, Moldova

Autorul corespondent: Mihail Batiri, e-mail michaelbatiri@gmail.com

Coordonator științific: Ion TIGHINEANU, Acad., Dr.hab., prof.univ.

Rezumat. *Materialele bidimensionale, cum ar fi sulfura de staniu (SnS), disulfura de staniu (SnS_2) și selenura de staniu (SnSe), prezintă proprietăți unice și potențiale aplicații în domenii precum electronica, optoelectronica, cataliza și energia. Exfolierea este o metodă eficientă de obținere a acestor materiale, prin separarea straturilor subțiri dintr-un substrat masiv. În acest articol, sunt examinate diverse metode și tehnici cost-eficiente de exfoliere a acestor materiale bidimensionale, precum și caracteristicile acestora. Scopul general este de a evalua potențialele aplicații ale acestor materiale în domeniul dispozitivelor optoelectronice și de energie și de a oferi o perspectivă generală asupra evoluției acestor materiale și tehnicilor de exfoliere în contextul cost-eficienței.*

Cuvinte cheie*: exfoliere, materialele bidimensionale, metode, sulfura de staniu.*

Introducere

Materialele bidimensionale (2D) sunt în prezent un subiect de interes major în cercetarea științifică și tehnologică [1]. Acestea sunt materiale cu grosimea de ordinul nanometric, cu proprietăți fizice și chimice unice și potențial pentru o gamă largă de aplicații, de la electronică și optoelectronică până la energie și medicină. Printre aceste materiale, SnS, SnS_2 și SnSe sunt sulfuri de staniu și seleniu, respectiv, care prezintă proprietăți interesante pentru aplicații în celule solare, fotodetectoare [2] și alte dispozitive optoelectronice [3]. Exfolierea este o tehnică importantă de producere a materialelor 2D, care implică desprinderea de straturi subțiri de materiale dintr-un substrat masiv. În cazul materialelor 2D, exfolierea poate fi realizată prin mai multe metode și tehnici cost-eficiente, cum ar fi exfolierea mecanică, exfolierea chimică și exfolierea electrochimică.

Metode cost-efective de exfoliere a materialelor 2D SnS, SnS_2 și SnSe

Exfolierea materialelor 2D este esențială pentru caracterizarea proprietăților lor fizice și optice și pentru dezvoltarea aplicațiilor lor practice.

Exfolierea mecanică este o metodă de obținere a materialelor 2D prin îndepărtarea unui strat subțire de material dintr-o suprafață mai mare. În cazul exfolierii cristalelor, o bucată mică este plasată pe o bandă adezivă și apoi banda este îndepărtată rapid pentru a lăsa un strat subțire de material pe suprafața benzii adezive. Acest proces este repetat de mai multe ori pentru a obține o cantitate suficientă de material 2D pentru caracterizare și aplicații ulterioare [4].

Exfolierea cu solvenți implică utilizarea unui solvent pentru a separa straturile de material 2D. O metodă comună este utilizarea solvenților organici, cum ar fi N-metilpirrolidonă (NMP) sau dimetilformamidă (DMF). Un studiu recent a arătat că exfolierea cu NMP a permis obținerea cu succes a materialelor 2D SnS, SnS_2 și SnSe [5]. În acest caz, cristalele au fost pulverizate în NMP și s-au obținut particule mici de material care au fost apoi curățate și dispersate în soluție.

Exfolierea cu ajutorul laserului este o metodă prin care se utilizează un fascicul de laser pentru a separa straturile de material 2D dintr-un cristal tridimensional. Această metodă a fost dezvoltată pentru prima dată pentru a exfolia grafenul [6] și constă în expunerea unei suprafețe de material la un fascicul de laser cu o anumită putere și durată.

În plus, există și alte metode de exfoliere, cum ar fi exfolierea cu abur, exfolierea cu ultrasunete și exfolierea chimică.

Caracterizarea materialelor exfoliate

Materialele SnS, SnS_2 și SnSe sunt materiale semiconductoare, cu banda interzisă în intervalul 0,8-2,2 eV, ceea ce le face potrivite pentru aplicații optoelectronice [7]. Materialele SnS, SnS_2 și SnSe prezintă o structură cristalină similară, cu un sistem de cristalizare ortorombic. Aceste materiale au de asemenea, o anizotropie puternică a proprietăților fizice, ceea ce înseamnă că proprietățile lor variază în funcție de direcția de creștere a cristalului. Cristalele masive fost crescute prin metoda transportului de gaz în fiole, acest proces a permis obținerea de suprafețe de oglindă cu dimensiuni aproximative de 5 × 7 mm și grosimi variabile cuprinse între 78 μm și 3 mm [8]. În plus, aceste materiale au o conductivitate termică scăzută și o conductivitate electrică moderată, ceea ce le face să fie atractive pentru aplicații termoelectrice [9]. Mai mult decât atât, materialele SnS, SnS_2 și SnSe prezintă o rezistență chimică bună la mediile corozive și stabilitate termică ridicată, ceea ce le face să fie potrivite pentru aplicații în medii extreme.

Pentru a determina proprietățile și caracteristicile materialelor exfoliate, sunt utilizate mai multe metode și tehnici de caracterizare [10]. Printre acestea se numără: microscopia electronică cu baleiaj (SEM), microscopia electronică de transmisie (TEM), microscopia cu forță atomică (AFM), fotoluminescența (PL), difracția de raze X (XRD), spectroscopia Raman, spectrometria de masă cu ionizare secundară (SIMS) și analiza termică (TGA), microscopia optică și alte metode de caracterizare.

Microscopia electronică cu baleiaj (SEM) este utilizată pentru a observa suprafața materialului exfoliat la o scară mare de magnificare, ceea ce permite observarea detaliată a morfologiei și topografiei suprafeței. Figura 1 prezintă două imagini SEM ale materialului SnS, unde (a) reprezintă imaginea SEM a materialului SnS masiv, iar (b) reprezintă imaginea SEM a materialului SnS_2 exfoliat în strat subțire prin metoda mecanică. Pentru a obține straturi subțiri de SnS, s-a folosit o metodă de exfoliere mecanică. În această metodă, o bucată de material SnS a fost lipită de o bandă adezivă cunoscută sub numele de "blue tape" [11]. Acest lucru a permis îndepărtarea cu ușurință a unui strat subțire de SnS de pe suprafața materialului. Pentru a transfera acest strat subțire pe un alt substrat, s-a folosit o altă bandă adezivă numită "TRT".

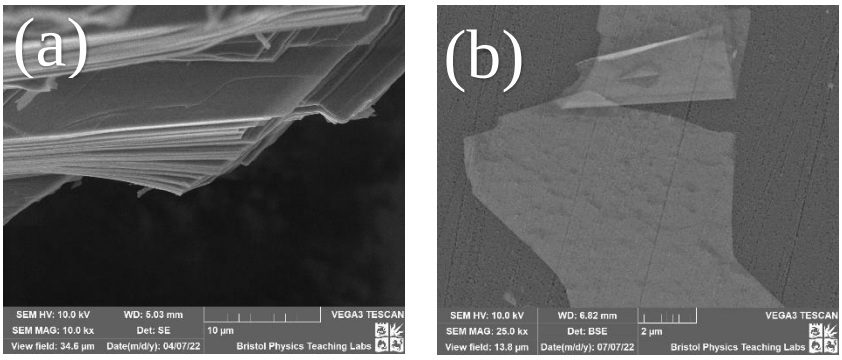


Figura 1. Imagine SEM al materialului SnS: (a) SnS masiv, (b) SnS exfoliat în strat subțire prin metoda mecanică.

Microscopia cu forță atomică (AFM) este o metodă de caracterizare non-destructivă care permite măsurarea topografiei suprafeței materialului exfoliat și caracterizarea proprietăților fizice ale suprafeței, cum ar fi aderența și rigiditatea. Caracterizarea structurii materialului stratificat SnS_2 și SnSe poate fi realizată prin utilizarea microscopiei cu forță atomică. Această metodă este una dintre cele mai utilizate metode pentru caracterizarea materialelor la scară nanometrică și poate fi folosită pentru a determina grosimea și topografia materialului. În figura 2 este prezentată caracterizarea AFM a straturilor subțiri de SnSe și SnS_2, prin care s-a analizat suprafața și grosimea acestuia [12]. Studiul a demonstrat că prin metoda mecanică se pot obține straturi foarte subțiri cu grosimea de până la 7 nm.

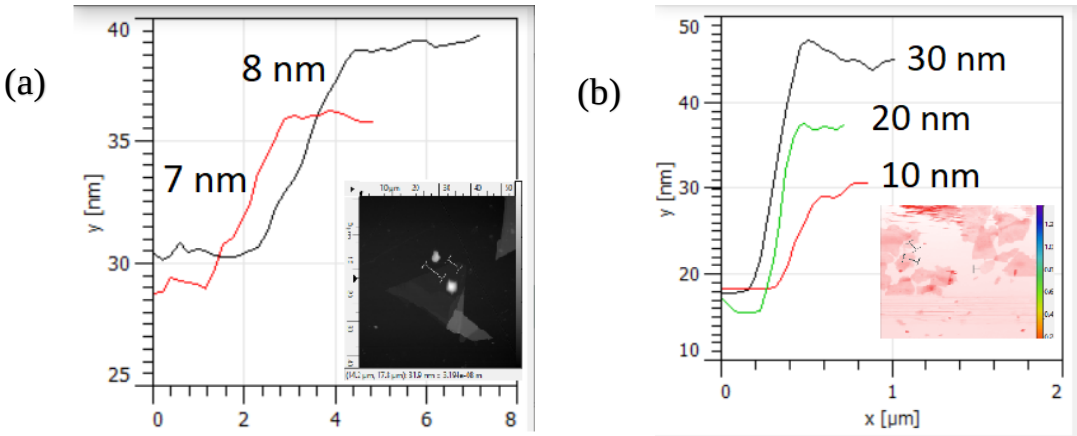


Figura 2. Caracterizarea AFM a straturilor subțiri de (a) SnS_2 (b)SnSe

Pe lângă determinarea grosimii, AFM poate fi utilizat și pentru a examina topografia suprafeței materialului SnSe. De exemplu, prin scanarea suprafeței, se pot detecta defectele și imperfecțiunile structurii materiale, iar grosimea stratului poate fi măsurată prin tehnicile AFM cu scanare în modul de contact sau non-contact.

Spectroscopia Raman este o tehnică de analiză a materialelor care se bazează pe interacțiunea dintre radiația luminoasă și legăturile moleculare ale materialului și poate fi folosită pentru a investiga compoziția și proprietățile structurale ale materialelor în diverse domenii științifice, de la fizica materialelor și ingineria mecanică până la biochimie și farmacologie.

Microscopia electronică de transmisie (TEM) este o metodă avansată de caracterizare a materialului care implică interacțiunea unui fascicul de electroni cu un material foarte subțire de obicei pregătit prin tehnici speciale de obicei cu grosimea sub 200 nm, pentru a produce o imagine cu o rezoluție extrem de mare. În timpul analizei, electronii trec prin materialul subțire și sunt dispersați și difractați de structura internă a materialului. Prin colectarea și procesarea acestor electroni dispersați, se poate obține o imagine a structurii cristaline interne a materialului. TEM este, de asemenea, utilizat în caracterizarea materialelor la nivel atomic și molecular, precum și în studiul fenomenelor fizice și chimice la scară nanometrică.

Concluzii

În această lucrare s-a demonstrat eficiența metodei exfolierii mecanice pentru obținerea materialelor 2D. Au fost obținute straturi subțiri cu o grosime cuprinsă între 7-30 nm și suprafețe mai mari de 3 micrometri. Aceste materiale exfoliate pot fi utilizate în cercetări ulterioare pentru aprofundarea studiilor în domeniul lor și le fac potențiale candidați pentru o gamă largă de aplicații în nanotehnologie, electronică, celule fotovoltaice, dispozitive fotoconductoare, catalizatoare, detectoare de gaze, dispozitive optoelectronice. Prin urmare, concluzionăm că metoda mecanică de exfoliere este o metodă eficientă, cost-efectivă și promițătoare pentru producerea de materiale 2D cu proprietăți unice.

Mulţumiri

Autorul exprimă recunoștința sa față de conducătorul științific, Acad. Ion Tighineanu și față de Dr. Andrei Sarua, pentru îndrumarea lor valoroasă, cunoștințele și experiența dobândită în laborator. Această lucrare a fost realizată în cadrul programului H2020 grant #810652 „Nano-MedTwin” și al proiectului din Programul de Stat #20.80009.5007.20.

Referinţe:

1. NOVOSELOV, K.S., GEIM, A.K., MOROZOV, S.V., JIANG, D., ZHANG, Y., DUBONOS, S.V., GRIGORIEVA, I.V., AND FIRSOV, A.A. Electric field effect in atomically thin carbon films. In: *Science*, 2004, 306, pp. 666–669.
2. DRAGOMAN, M., BATIRI, M., DINESCU, A., CIOBANU ,V., RUSU, E., DRAGOMAN, D., TIGINYANU, I. A SnS2-based photomemristor driven by sun. In: *Journal of Applied Physics*, 2018, 123, 024506.
3. WANG, H., CHEN, L., GUO, Y., YU, H., & ZHANG, S. Recent advances in two-dimensional SnS and SnSe materials for optoelectronics and energy conversion. In: *Materials Horizons*, 2020, 7(10), pp. 2487-2518.
4. RAZA, W., ALI, Z., & KHALID, N. R. An overview of recent advances in exfoliation of two-dimensional materials for energy storage applications. Journal of Materials Science: In: *Materials in Electronics*, 2019, 30(18), pp. 16707-16723.
5. VELICKÝ, M., DONNELLY, G. E., HENDREN, W. R., MCFARLAND, S., SCULLION, D., DEBENEDETTI, W. J. I., CORREA, G. C., HAN, Y., WAIN, A. J., HINES, M. A., MULLER, D. A., NOVOSELOV, K. S., ABRUÑA, H. D., BOWMAN, R. M., SANTOS, E. J. G., & HUANG, F. Mechanism of Gold-Assisted Exfoliation of Centimeter-Sized Transition-Metal Dichalcogenide Monolayers. In: *ACS Nano*, 2018, 12, 10, pp. 10463–10472.
6. ZHANG, X., ZHAO, C., LIU, Y., HE, C., & XIE, Y. Synthesis of SnS and SnS2 nanosheets by liquid-phase exfoliation and their potential application in photodetection. In: *Nanotechnology*, 2017, 28(50), pp. 505705.
   AN, SJ., KIM, YH., LEE, C., PARK, DY., JEONG, MS. Exfoliation of Transition Metal Dichalcogenides by a High-Power Femtosecond Laser. In: *Scientific Reports*, 2018,volume 8, pp. 12957.
7. RUSU, E., SYRBU, N., TIRON, A., ZALAMAI, V. Band structure and optical constants of SnS2 single crystals. In: *Mater. Res. Express, 2019*, 6 046203.
8. WANG, G., PANDEY, R., & KARNA, S. P. Emerging 2D semiconducting materials beyond graphene and TMDCs: properties, synthesis, and applications. In: *Journal of Materials Chemistry C*, 2016, 4(31), pp. 7239-7254.
9. Shanmugam, V., Mensah, R.A., Babu, K., Gawusu, S., Chanda, A., Tu, Y., Neisiany, R.E., Försth, M., Sas, G., Das, O. A Review of the Synthesis, Properties, and Applications of 2D Materials. In: *Part. Part. Syst. Charact.*, 2022, 39, 2200031.
10. ZHANG, J., GUO, S., & JIANG, J. Recent advances in thermoelectric materials based on SnS. In: *Journal of Materials Chemistry C*, 2018, 6(27), 7101-7116.
11. CHHOWALLA, M., & JENA, D. Heterostructures based on two-dimensional materials for advanced device applications. In: *Nature Materials*, 2016, 15(3), pp. 290-301.

ELECTROLYTE MATTERS: UNDERSTANDING ITS IMPACT ON SUPERCAPACITOR PERFORMANCE

Barnika CHAKRABORTY[1], Mangesh DESAI [2*], Kirandeep SINGH [3]

[1]*Institute for Materials Science, Faculty of Engineering, Christian- Albrechts- University, Kiel, Germany*
[2]*Department of Physics, Savitribai Phule Pune University, Pune, India*
[3]*CSIR- National Chemical Laboratory, Dr. Homi Bhabha Road, Pune-411008, India*

*Corresponding author: Kirandeep Singh, kp.singh@ncl.res.in; bach@tf.uni-kiel.de

Abstract. *The article examines how sodium and lithium ions can be inserted into thin films of MoS_2 synthesized by DC magnetron sputtering, with the aim of using these films for energy storage. These films possess unique properties and structures that make them promising candidates for energy storage. The insertion of sodium and lithium ions has substantially boosted their energy storage capacity and electrical conductivity. The research also demonstrated that the insertion process is reversible, suggesting that the thin films could potentially be employed in rechargeable batteries. However, additional research is necessary to refine the insertion procedure, enhance the stability and performance of the films, and investigate other possible uses. Overall, this study provides valuable insights into the potential use of sputtered MoS_2 thin films for energy storage applications.*

Keywords: *capacitance, carbon cloth (CC), intercalation, lithium, molybdenum disulfide.*

Introduction

The need for energy in daily life, both at industrial and household levels, has existed for ages. Fossil fuels have traditionally supplied this energy [1], but they are harmful to society and nearly depleted. Thus, there is an increasing demand for non-conventional or renewable energy sources such as wind, hydro, thermal, tidal, and solar [2]. However, the irregular availability of these sources highlights the importance of energy storage devices such as fuel cells, batteries, and supercapacitors [3].

Supercapacitors can be categorized into three main types: electrochemical double layer capacitors (EDLCs), pseudocapacitors, and hybrid capacitors. EDLCs are based on the principle of charge separation at the electrode-electrolyte interface, while pseudocapacitors involve surface redox reactions at the electrode-electrolyte interface. Hybrid capacitors combine the characteristics of both EDLCs and pseudocapacitors, offering high energy and power densities [4].

Generally, a supercapacitor comprises various parts, including the electrode, current collectors, electrolyte, and separators, that work together to store and release electrical energy efficiently. We would dig out our focus on electrolyte and their affection towards super capacitive performance. The electrolyte ions diffuse into the layers of electrodes, especially a 2D working electrode to cause intercalation. Intercalation of ions into the electrode material can improve the supercapacitive performance by modifying charge storage and diffusion of ions [5].

Experimental Section

Materials and chemicals. The molybdenum disulfide sputtering target with 99.999% purity was acquired from Testbourne Ltd in the UK. Flexible carbon cloth was purchased from Nanoshel India Pvt. Ltd. The chemicals used in this study, including sodium sulfate (Na_2SO_4) and lithium perchlorate ($LiClO_4$), were obtained from Merck Sigma-Aldrich and used without additional purification.

Synthesis Method. To create a flexible supercapacitor electrode, a low-cost carbon cloth mesh was first cut into 2 cm × 1 cm slices and cleaned with a solution of acetone and isopropyl alcohol. The substrate was then mounted on a resistive heating stage within the sputtering chamber, and a commercially available MoS_2 sputtering target was utilized to deposit thin films. The deposition process was carried out using 4N purity Argon gas at room temperature for 15 minutes under a constant DC power of 45 W, substrate-target distance of 60 mm, and working pressure of 10 mTorr, while the temperature was increased to promote thin film growth [6]. Prior to the deposition, a base vacuum of 1×10^{-6} was maintained in the sputtering chamber.

Results and Discussion

Structural and Morphological Characterization

Fig. 1 (A, B, C) shows the FE-SEM of MoS_2 thin films at different magnifications. At higher magnification it has been found that the MoS2 exhibits a sheet like structure on carbon cloth. Additionally, the porous nature of carbon cloth and deposited MoS_2 on top of it (Figure 1 A & B) can prove to be significant for the diffusion and absorption of electrolytes.

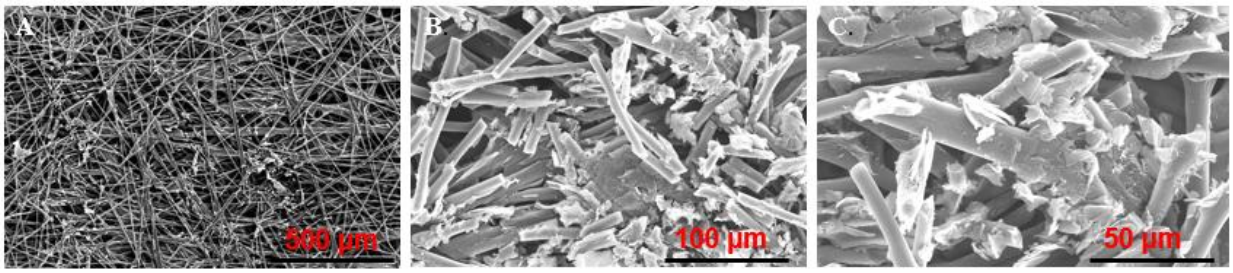


Figure 1. (A, B, C) FESEM image of MoS_2 thin film sputtered on CC

Raman spectra (Fig 2) show the characteristics signal of MoS_2 with the peaks from carbon cloth (1318 cm^{-1}). The peaks at positions 294 cm^{-1}, 226 cm^{-1}, 411 cm^{-1} corresponds to the E^1_{2g} and A^1_g peaks of the MoS_2 films. These E^1_{2g} and A^1_g peaks are related to in plane and out-of-plane vibrational modes of MoS_2, respectively.

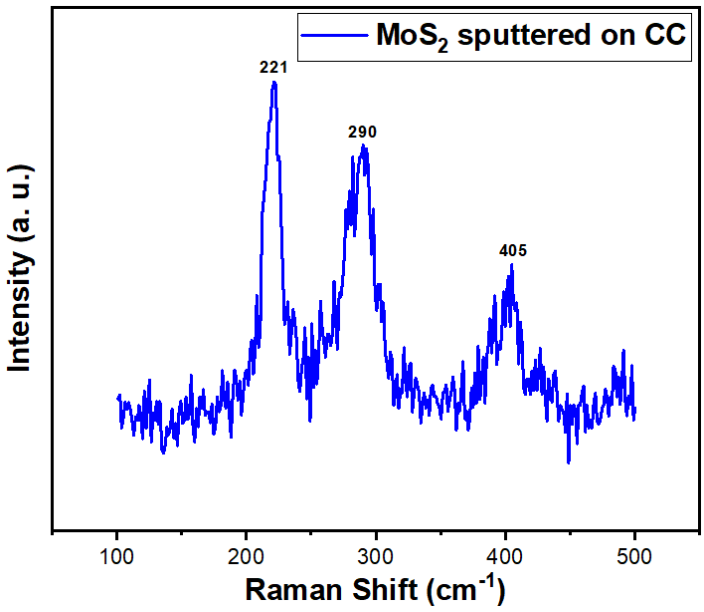


Figure 2. Raman Spectra of MoS_2 thin film sputtered on CC

Electrochemical Characterization

Figure 3 demonstrates cyclic voltammetry (CV) studies of MoS_2 thin films in sodium sulphate, Na_2SO_4 and lithium perchlorate, $LiClO_4$ electrolytes. In both the electrolytes, the area under curve was found to be increased with an increasing number of potential scans due to intercalation of the alkali ion like Na or Li. The process of intercalation is possible since MoS_2 is layered material. The MoS_2 electrode provides electrochemical active sites and/or voids, facilitating the charge storage mechanism via intercalation and EDLC. With increase in the scan rate the specific capacitance values were decreased because electrolyte ions do not get enough time to interact with MoS_2 electrode. The electrode provided edge-active sites and high effective surface area, facilitating the charge storage mechanism. The Sp. capacitance values for Na_2SO_4 was 9.5238 F/g, 3.8095 F/g, 1.9047 F/g at 20 mV/s. 50 mV/s, 100 mV/s scan rates respectively. The Sp. capacitance values for $LiClO_4$ was 13.4249 F/g, 8.6727 F/g, 5.9060 F/g at 20 mV/s. 50 mV/s, 100 mV/s scan rate respectively. The values stood higher for $LiClO_4$ than Na_2SO_4.

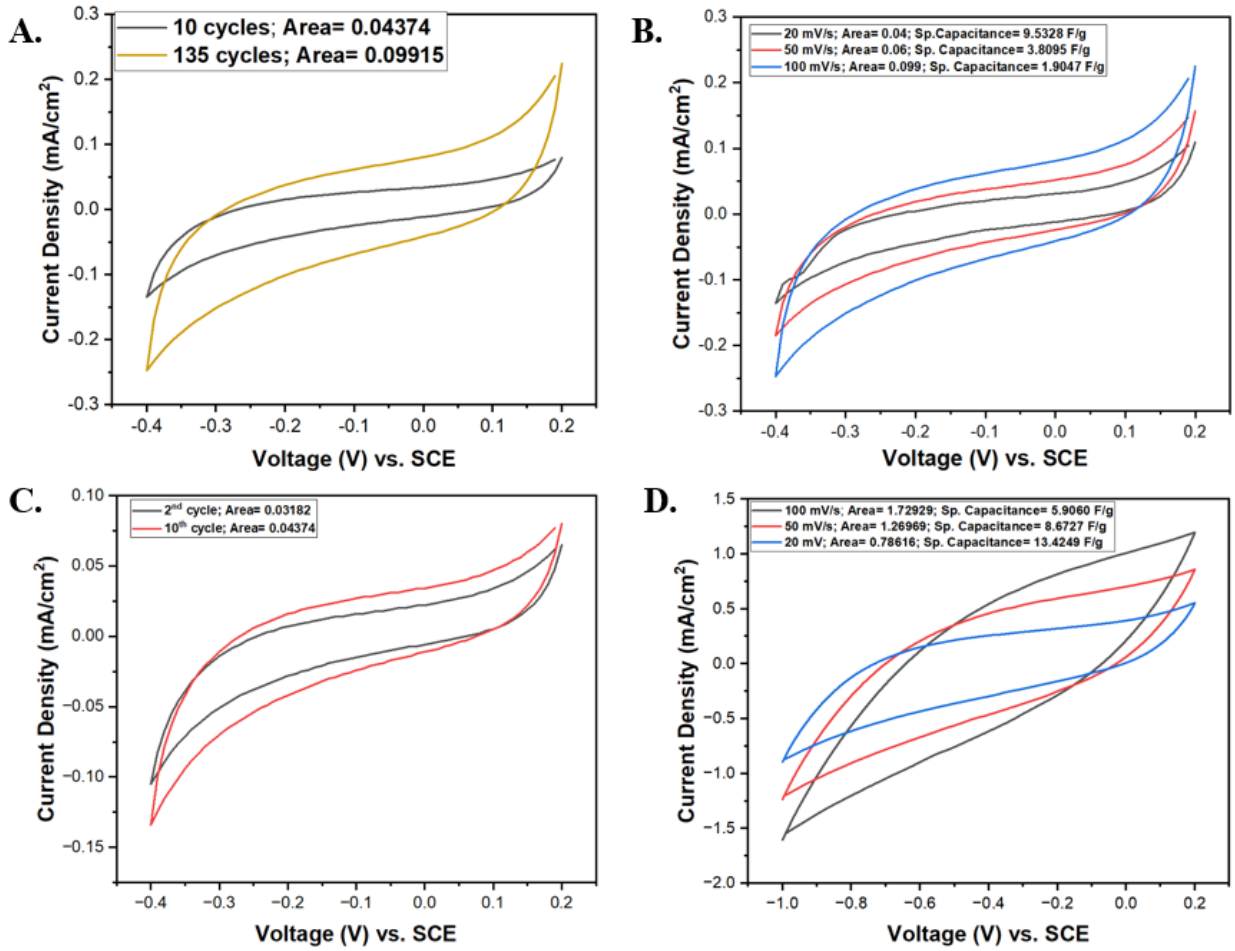


Figure 3. CV curves for (A, B) with Na_2SO_4 as the electrolyte as a comparison of different cycles and different scan rates. CV curves for (C, D) with $LiClO_4$ as the electrolyte as a comparison of different cycles and different scan rates.

Conclusion

In conclusion, sputtered MoS_2 thin films have shown promising results for sodium and lithium ion intercalation. The thin films' unique structure and properties have made them suitable candidates for energy storage applications. The intercalation of sodium and lithium ions has been found to significantly enhance the thin film's electrical conductivity and energy storage capacity. Moreover, the intercalation process has been found to be reversible, indicating the potential for these thin films to be used in rechargeable batteries.

Further research is required to optimize the intercalation process, improve the performance and stability of the thin films, and explore other potential applications. Overall, the results obtained from this study provide valuable insights into the use of sputtered MoS_2 thin films for energy storage applications [7].

Acknowledgements

CSIR- National Chemical Laboratory and SERB Grant No. EEQ/2021/000242.

References

1. WINTER, M.; BRODD, R. J. What are batteries, fuel cells, and supercapacitors? *Chem. Rev*. 2004, 104 (10), pp. 4245−4270. https://pubs.acs.org/doi/10.1021/cr020730k
2. THEERTHAGIRI, J.; SENTHIL, R. A.; NITHYADHARSENI, P.; LEE, S. J.; DURAI, G.; KUPPUSAMI, P.; MADHAVAN, J.; CHOI, M. Y. Recent progress and emerging challenges of transition metal sulfides based composite electrodes for electrochemical supercapacitive energy storage. *Ceram. Int.* 2020, 46 (10), pp. 14317−14345. https://researchspace.csir.co.za/dspace/handle/10204/11949
3. ZHANG, X.; ZHANG, H.; LI, C.; WANG, K.; SUN, X.; MA, Y. Recent advances in porous graphene materials for supercapacitor applications. *RSC Adv*. 2014, 4 (86), pp. 45862−45884. https://pubs.rsc.org/en/content/articlepdf/2014/ra/c4ra07869a
4. LIU, C.; YU, Z.; NEFF, D.; & ZHAMU, A. Graphene-based supercapacitor with an ultrahigh energy density. 2010 *Nano Letters,* 10(12), pp. 4863-4868. https://doi.org/10.1021/nl102661q
5. CHMIOLA, J.; YUSHIN, G.; GOGOTSI, Y.; PORTET, C.; SIMON, P.; & TABERNA, P. L. Anomalous increase in carbon capacitance at pore sizes less than 1 nanometer. *Science*, 2006 313(5794), pp. 1760-1763. https://doi.org/10.1126/science.1132195
6. SHARMA, G. K.; RANJAN, B.; KAUR, D. Electrochemical kinetics of 2D-MoS_2 sputtered over stainless-steel mesh. *Ceram. Int.* 2022 48, pp. 23404- 23414. https://doi.org/10.1016/j.ceramint.2022.04.332
7. JIN, Y.; LI, H.; LIANG, C.; LI, M.; LI, Y.; & ZHANG, X. (2021). Sodium and lithium ions intercalation on sputtered MoS_2 thin film for energy storage applications. *Journal of Power Sources*, 498, pp. 229970. https://doi.org/10.1016/j.jpowsour.2021.229970

STUDIEREA SUPRACONDUCTIBILITĂȚII ÎN NANOSTRUCTURI DE NIOBIU-COBALT

Cezar-Casian MALCOCI

Departamentul Microelectronică și Inginerie Biomedicală, gr. MN-211, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Moldova

Autorul corespondent: Cezar-Casian Malcoci, sharovasea@gmail.com

Îndrumător/coordonator științific: Anatolie SIDORENKO, acad., dr. hab., prof. univ., Universitatea Tehnică a Moldovei

Rezumat. *În lucrare se descriu etapele de pregătire a unei serii de probe și depunere de Nobiu – Cobalt la aceași temperatură, utilizând un magnetron (Z400). Apoi etapele de selectare a unei probe pentru tăiere în mai multe bucăți cu scopul de a măsura supraconductibilitatea în aceași probă la diferite grosimi ale straturilor. Cercetarea a avut loc în două etape, prima etapa fiind efectuată în laboratorul "Structuri cu Corp Solid" a Institutului de Inginerie Electronică și Nanotehnologii "D. Ghițu" din Chișinău, departamentul "Criogenie". A doua etapă a cercetării a fost efectuată la Universitatea din Stockholm în centrul AlbaNova.*

Cuvinte cheie*: materiale supraconductoare, temperatura critică, probe, LHZ400, Criocooler-ul 17T.*

Introducere

În ultimii ani, interesul pentru studierea supraconductivității în nanostructuri metalice a crescut semnificativ, datorită potențialului lor de a furniza proprietăți unice, în special datorită dimensiunilor lor reduse. În acest context, nanostructurile de niobiu-cobalt au atras atenția cercetătorilor datorită proprietăților lor interesante, cum ar fi supraconductivitatea la temperaturi relativ ridicate și durata mare de viață a electronilor supraconductori.

Scopul acestui studiu este de a investiga proprietățile supraconductoare ale nanostructurilor de niobiu-cobalt, în vederea identificării factorilor care influențează comportamentul supraconductiv al acestor materiale și a dezvoltării de aplicații practice bazate pe acestea. Cercetarea se va concentra pe sinteza, caracterizarea structurală și electrică, precum și pe evaluarea proprietăților termice ale nanostructurilor de niobiu-cobalt, cu accent pe comportamentul supraconductiv și potențialul lor de utilizare în tehnologii avansate.

1. Supraconductori: definiție și evoluție

Un supraconductor este un element sau aliaj metalic, atunci când este răcit sub o anumită temperatură prag, materialul pierde toată rezistența electrică. Pragul de temperatură sub care un material trece într-o stare supraconductoare este desemnat ca *temperatura critică* (T_c). S-a demonstrat că nu toate materialele se transformă în supraconductori, deoarece materialele au fiecare propria lor valoare de T_c [1].

Fenomenul a fost observat pentru prima dată de către specialistul Heike Kamerlingh Onnes în anul 1911, care i-a adus Premiul Nobel pentru fizică în anul 1913 [1-3]. Acesta studiind dependența de temperatură a rezistivității electrice a mercurului a observat că sub o anumită temperatură, apropiată de temperatura heliului lichid (4,2 K), rezistivitatea scade brusc către zero. Ulterior cercetările a permis să se determine temperatura critică pentru diferite elemente chimice, atât simple, cât și compuse.

Pe parcursul anilor acest domeniu s-a extins foarte mult și au fost descoperite multe alte forme de supraconductori, inclusiv și supraconductorii de tip 2 în anul 1930. Specialiștii Meissner W. și Ochsenfeld R. în anul 1933 descoperă că un material supraconductor expulzează un câmp magnetic. Fenomenul dat este astăzi cunoscut ca *efectul Meissner* (-Ochsenfeld). Acesta este atât de puternic, încât un magnet poate fi levitat deasupra unui material supraconductor.

În anul 1941 a fost identificată supraconductibilitatea în nitrura de niobiu la 16 K. Iar în anul 1953 este descoperit Vanadium-3 Siliciu ca supraconductor la temperatura de 17,5 K. Primul cablu supraconductor comercial de niobiu-titan este dezvoltat de Westinghouse scientists în 1962 [3]. Pentru prima dată acest fir de înaltă energie a fost utilizat pentru electromagneții unui accelerator de particule.

Teoria de bază a supraconductivității (BSD) a fost prezentată în anul 1957 de către fizicienii americani, aceasta lea adus oamenilor de știință - John Bardeen, Leon Cooper și John Schrieffer - Premiul Nobel pentru fizică din anul 1972. Pentru munca cu superconductivitatea o parte din Premiul Nobel pentru fizică în anul 1973, ia revenit lui Brian Josephson [1].

Domnul Brian D. Josephson, în anul 1962, fiind un student din ciclul trei al universității din Cambridge, a prezis că un curent electric va circula între două materiale supraconductoare, chiar și atunci când ele sunt separate de un material ne-supraconductor sau un izolant. Acest fenomen de tunelare este astăzi cunoscut sub numele de *efect Josephson*. Fiind aplicat la fabricarea unor dispozitive electronice, ca de exemplu CALMAR, un instrument capabil să detecteze câmpuri magnetice de intensitate foarte reduse.

Primul supraconductor organic (pe bază de carbon) a fost sintetizat cu succes în anul 1979 de către cercetătorul danez Klaus Bechgaard de la Universitatea din Copenhaga împreună cu alți trei membri ai unei echipe franceze. Materialul abreviat $(TMTSF)_2PF_6$ a fost răcit la o temperatură de numai 1,2 K și apoi supus unei presiuni înalte pentru a deveni supraconductor.

Specialiștii Karl Muller și Johannes Bednorz, în anul 1986, au făcut o descoperire care a revoluționat modul în care oamenii de știință utilizau și apreciau superconductorii. Se considera că superconductivitatea se manifesta numai atunci când este răcită până aproape de zero absolut, dar folosind un oxid de bariu, lantan și cupru, au descoperit că a devenit un supraconductor la aproximativ 40 de grade Kelvin.

Cercetătorii de la University of Houston şi University of Alabama din Huntsville (SUA) în anul 1987 substituie lantan cu ytriu şi obţin un compus ceramic supraconduor la 92 K, aducând supraconductibilitate în domeniul temperaturii azotului lichid, YBCO. În anul 1988 specialistul Allen Hermann de la University of Arkansas realizează un supraconductor ceramic conţinând calciu şi taliu la temperatura de 120 K. Apoi, cercetătorii de la IBM şi AT&T Bell Labs produc un supraconductor la 125 K. Peste ceva timp în anul 1993 specialiştii A. Schilling, M. Cantoni, J. D. Guo, şi H. R. Ott din ETH Zurich, din Elveția, propun un supraconductor pe bază de mercur, bariu, calciu şi cupru oxigen, (HgBa2Ca2Cu3O8) la care temperatura critică este 133K [4].

În prezent supraconductorii sunt folosiți în: cabluri supraconductoare, biomagnetism, levitație magnetică, motoare și generatoare, magneți, transformatori, calculatoare cuantice etc.

2. Partea experimentală

Scopul lucrării constă în realizarea unor serii de probe prin depunerea straturilor nanometrice de cobalt și nobiu și cercetarea fenomenului de supraconductibilitate, utilizând criostatul. Cercetarea a avut loc în două etape, prima etapa fiind efectuată în cadrul laboratorului ”Structuri cu Corp Solid” a Institutului de Inginerie Electronică și Nanotehnologii ”D. Ghițu” din Chișinău, departamentul ”Criogenie”.

În perioada ianuarie 2022 au fost realizate 5 probe, utilizând instalația Z400 din laborator.

Pentru realizarea experimentului s-au parcurs etapele:

1. *Pregătirea probelor*. Începem prin selectarea plăcuței de siliciu. Orice acțiune cu placheta se va desfășura utilizând penșeta și un șervețel curat pentru evitarea depunerilor particulelor nedorite pe suprafața acesteia. Tăierea plachetei se realizează utilizând pixul cu vârf de diamant sau utilizând utilajul special de tăiere.

După obținerea probelor cu dimensiunile și forma dorită urmează curățirea acestora:

- Probele se introduc în curățătorul cu microunde (unde urmează de a fi spălate de 4-5 ori câte 15 minute).
- Extragem probele cu penșeta și le clătim cu izopropanol.

- În containerul oval le clătim cu apă distilată.
- Le uscăm cu uscătorul cu gaz.
- Curățim preventiv capacul magnetronului și locul lui de aplicație cu un șervețel umed în izopropanol.

2. *Introducerea probelor în aparatul Z400*. Înaintea introducerii probelor curățim utilizând un șervețel umed în izopropanol capacul magnetronului și locul lui de contact. După curățire amplasăm probele pe ținta magnetronului.

3. *Depunerea straturilor*. După amplasarea probelor se închide magnetronul și efectuăm următorii pași pentru crearea vidului (presiunii necesare): pornim sistemul de răcire cu apă; verificăm indicatorul pentru a confirma circulația apei în sistem, în caz contrar revenim la pasul 1; pornim blocul central; pornim pompa; deschidem ”robinetul”; așteptăm scăderea presiunii până la 10^{-2}; pornim al 2-lea indicator; apăsăm butonul Test 1; pornim pompa turbomoleculară. Când utilajul ajunge la presiunea dorită efectuăm depunerea straturilor prin apăsarea unor butoane respectând parametrii doriți.

La final s-au obținut 5 probe, astfel s-a încheiat prima etapă de cercetare. Rezultatele obținute sunt centralizate în tabelul 1.

Tabelul 1

Rezultatele obținute pentru cele 5 probe

Nr. probei	Data	Componenta	Temperatura substratului
1	22.01.2022	Si – substrat/Nb(150)/Co(10)/Nb(150)/Si-cap	200 C
2	20.01.2022	Si – substrat/Nb(150)/Co(10)/Nb(150)/Si-cap	200 C
3	21.01.2022	Si – substrat/Nb(50)/Co(1,5)/Nb(8)/Co(2,5)/Nb(8)/Si-cap	200 C
4	26.01.2022	Si – substrat/Nb(150)/Co(4)/Nb(150)/Si-cap	200 C
5	31.01.2022	Si – substrat/Nb(150)/Co(1,5)/Nb(8)/Co(2,5)/Nb(150)/Si-cap	200 C

A doua etapă a cercetării a fost efectuată la Universitatea din Stockholm în centrul AlbaNova, unde s-au verificat probele utilizând Criocooler-ul 17T. Sistemul utilizează o pereche de criocooler în două trepte pentru a produce temperaturi de aproximativ 4,2 Kelvin la magnet și permit funcționarea VTI-ului între 1,6K și temperatura camerei. În continuare se prezintă etapele parcurse în laboratorul din centrul AlbaNova.

Pregătirea probei nr. 3:

1. Zgârierea probei pentru amplasarea în Fascicul de ioni focalizat (FIB).
2. Amplasarea în FIB.
3. Fotografierea probei nr. 3.
4. Amplasarea probei nr. 3 în microscop pentru analiza zgârieturilor.

Pregătirea probei nr. 1:

1. Selectarea și tăierea probei nr. 1 în 22 de bucăți.
2. După tăiere urmează lipirea probelor pe PCB (uscarea durează 12 ore).
3. După ce proba sa uscat pe PCB, urmează lipirea (bondarea) în camera curată pentru crearea contactelor de măsurare a temperaturii critice Tc.
4. Urmează fixarea probei în undița criostatului.
5. Introducerea undiței cu proba în criostat.
6. Răcirea criostatului pâna la temperatura de 4 K pentru a verifica supraconductibilitatea probei. Răcirea criostatului se datorește cuptorului din cadrul Criocooler-ul 17T.
7. În cazul în care nu s-a găsit supraconductibilitate până la 4 K urmează răcirea până la valori mai joase utilizând soft-ul LabVIEW.
8. Când atingem starea de supraconductibilitate menținem temperatura stabilă și obținem Temperatura critică a probei. Aceasta se determină datorită soft-ului LabVIEW.
9. După ce s-a obținut temperatura critică pornim cuptorul pentru a încălzi criostatul până la temperatura camerei ca să putem schimba proba și repeta procesul.

În urma efectuării pașilor descriși mai sus s-a obținut pentru proba nr. 1 temperatura critică. Rezultatele obținute se prezintă în tabelul 2.

Analizând tabelul 2 se poate spune că subproba 2 și 7 nu au temperatură critică, de aceea au fost eliminate din studiu. Totodată se observă că odată cu micșorarea grosimii are loc o scădere a temperaturii critice.

Tabelul 2

Rezultatele obținute pentru proba 1

Nr. subprobei	X, mm de la capătul gros și grosimea de Nb, dNb nm	Tc, K	Nr. subprobei	X, mm de la capătul gros și grosimea de Nb, dNb nm	Tc, K
2	-	-	16	Aprox = 29.14	4.660
7	-	-	17	Aprox = 30.95	4.942
10	Aprox = 18.28	1.28	19	Aprox = 34.57	5.233
11	Aprox = 20.09	2.193	20	Aprox = 36.38	5.3
12	Aprox = 21.9	3.43	21	Aprox = 38.2	5.245
13	Aprox = 23.71	3.905	22	Aprox = 40	5.381

Criostatul permite înregistrarea tuturor datelor în timpul celor trei faze: răcire, menținerea temperaturii și încălzirii. Aceste date sunt necesare pentru a lua decizii privind starea probei pe parcursul experimentului și utilizarea ulterioară a ei.

Concluzii

Lucrarea descrie fenomenele fizice și influența acestora asupra supraconductorilor. Probele au fost testate în studiul proceselor de formare a unei nanostructuri bazate pe Niobiu și Cobalt. A fost investigată influența diferiților parametri tehnologici: grosime; temperatura depunerii; rezistență etc.

În urma efectuării pașilor descriși în lucrare s-a obținut pentru proba nr. 1 temperatura critică. Analizând rezultatele obținute se poate spune că subproba 2 și 7 nu au temperatură critică, de aceea au fost eliminate din studiu. Totodată se observă că odată cu micșorarea grosimii are loc o scădere a temperaturii critice.

Temperatura substratului este un factor principal care afectează formarea nanosistemelor multistrat, structura atomică a zonelor de contact ale interfeței, precum și compoziția și structura nanosistemului multistrat în ansamblu.

Referinţe

1. Ce este un supraconductor? Definiție și utilizări. [online]. [accesat 10.10.2022]. Disponibil: https://www.greelane.com/ro/știință-tehnologie-math
2. Supraconductibilitate. [online]. [citat 13.10.2022]. Disponibil: https://ro.wikipedia.org/wki/Supraconductibilitate.
3. The History of Superconductors. [online].[citat 27.11.2022]. Disponibil: www.superconductors.org/History.htm
4. Introducere în supraconductibilitate. [online].[citat 20.11.2022]. Disponibil: www.phys.ubbcluj.ro/~iosif.deac/courses/FCS/c...

TESTER DE RESPIRAȚIE CU HIDROGEN

Mihai BRÎNZĂ[1,2,3]

[1]*Departamentul Microelectronică și Inginerie Biomedicală; IBM-211M; Facultatea Calculatoare, Informatică și Microelectronică; Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova*
[2]*Centrul de Nanotehnologii și Nanosenzori; Departamentul Microelectronică și Inginerie Biomedicală; Facultatea Calculatoare, Informatică și Microelectronică; Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova*
[3]*Centrul Național de Inginerei Biomedicală; Departamentul Microelectronică și Inginerie Biomedicală; Facultatea Calculatoare, Informatică și Microelectronică; Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova*

*Autorul corespondent: Mihai Brînză, mihai.brinza.98@gmail.com

Îndrumător/coordonator științific: Oleg LUPAN, Dr. hab., prof. univ., UTM, FCIM, DMIB, CNN.

Rezumat. *Lucrarea constă în expunerea rezultatelor cercetărilor senzorilor de hidrogen pe bază de oxizi metalici ca alternativă dispozitivelor actuale pentru testarea gazului expirat în scopuri de diagnosticar. Au fost cercetate cîteva probe din TiO_2 cu scopul de a obține un senzor selectiv la H_2 care va permite în viitor determinarea concentrațiilor gazului dat la diferite limite de la 0 pînă la 100 ppm pentru a detecta sau a prezice careva boli, pentru a face un diagnostic corect dereglărilor gastro-intestinale și pentru a îmbunătăți viața pacienților.*

Cuvinte cheie*: respirație cu hidrogen, nanosenzori, polimer PV, diagnosticare, biomarkeri.*

Introducere

Tendința actuală a sectorului medical este de a dezvolta metode practice de diagnosticare neinvazivă a pacienților, întrucît acestea prezintă un risc aproape nul pentru securitatea și bunăstarea omului. Cu toate că majoritatea informației de regulă, legată de bunăstarea unei persoane este colectată în mare parte din aspectul vizual și calitatea sanguină a unui posibil pacient, deja de cîțiva ani este cercetată legătura între anumite gaze volatile și anumite boli care influențează, sau mai bine spus, au drept rezultat eliminarea din organism a unei anumite concentrații de diferite gaze(cum ar fi H2, CO2, NH4).

Diferite articole pot fi citite astăzi despre legătura dintre gazele expirate și diferite incomodități/neplăceri/disconforturi ale tractului digesitv dar și a stării organismului la general. Este demonstrat că excesul bacterian al intestinului subțire este caracterizat prin creșterea numărului și/sau tipului de bacterii colonice în tractul gastrointestinal superior. Pacienții afectați pot fi asimptomatici sau pot reprezenta simptome precum: plenitudine, disconfort abdominal, diaree, steatoree, flatulență, dispepsie, malabsorbție a nutrienților, scădere în greutate. Metoda considerată standardul de aur pentru așa tip de diagnosticări este considerată aspiratul jejunal(jejun - Parte a intestinului subțire cuprinsă între duoden și ileon.) dar pentru colectarea acesteia sunt necesare procese invazive. Alternativa neinvazivă în cazul dat se adeverește a fi testul respirator de H2 care are un cost redus, simplu și cu sensibilitate și specificate bună [1].

Senzorii produși pentru detectarea hidrogenului pot fi produsi pe bază de silicon poros dopat cu grafen [2] sau pot fi produși senzori nanostructurați și dopați sub formă de pelicule [3]. Tehnologiile de producere diferă, ceea ce afectează costul unui senzor, proprietățile acestuia, dar și sfera de utilizare. Cel mai des senzorii se testează pentru o anumită concentrație care corepsunde sferei de utilizare. În industrie deseori vor fi întîlnite concentrații fixe, corespunzătoare utilizări gazului în dependeță de sfera acestuia(i.g. combustibili) strict legate de securitatea și bunăstarea omului. În partea medicală a senzorilor de gaze concentrațiile date pot fi de referință mult mai mici,dar nu mai puțin exacte. Dacă să luăm drept gazul cercetat o componentă a expirului uman, atunci concentrațiile cercetate pot fi în jur de 20 ppm.

Bacteriile din intestin produc în general hidrogen gazos prin fermentarea carbohidraților. În intestin, bacteriile pot face acest lucru numai atunci când carbohidrații din dietă nu sunt absorbiți în intestinul subțire și rămân la fel de material nedigerat pe măsură ce călătorește de-a lungul tractului digestiv în intestinul gros. Deși o parte din hidrogenul gazos produs de bacterii este expulzat sub formă de flatus sau în formarea altor molecule, cum ar fi sulfurile, acetatul și acizii grași cu lanț scurt, majoritatea gazului este absorbit prin mucoasa intestinului gros în fluxul sanguin. Gazul este apoi transportat la plămâni prin fluxul sanguin și din sânge este schimbat în căile respiratorii ale plămânilor și expirat. Singura sursă de hidrogen gazos din respirație poate fi din fermentația bacteriană din intestin. Același lucru este valabil și pentru gazul numit metan care este expirat de unii oameni și nu de toți. Bacteriile din intestinul lor gros produc metan din hidrogen. Cantitatea de gaze de hidrogen și metan expirate din plămâni poate fi măsurată cu ușurință prin prelevarea unei probe de respirație și măsurarea cu o mașină de testare a respirației [4].

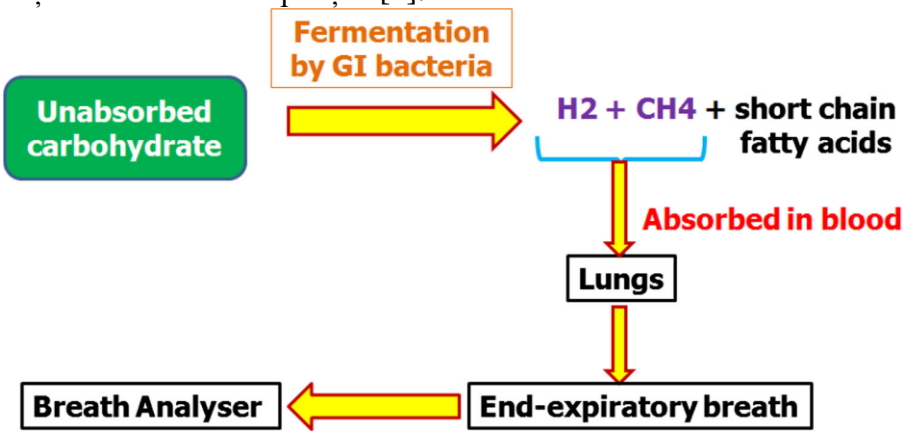


Figura 1. Principiul unui test de respirație cu hidrogen [4].

Influența H_2 în organismul uman

Tehnicile care utilizează infuzii intestinale de gaz au fost folosite pentru a studia producția de H2 în intestinul uman. Volumul de H2 în intestinul a 10 subiecți normali a variat de la 0,06 la 29 ml. Producția de H2, care a fost în medie de 0,24 ml pe minut în stare de post, a crescut brusc după instilarea intestinală de lactoză până la o rată medie de vârf de 1,6 ml pe minut. Ingestia de alimente a crescut, de asemenea, producția de H2 de șapte ori până la 30 de ori. În intestinul normal, mai mult de 99% din producția de H2 a fost colonică, dar producția de intestin subțire a crescut la un pacient cu un număr excesiv de bacterii ale intestinului subțire. Producția de H2 la om depinde în primul rând de livrarea substraturilor ingerate, fermentabile, către o floră intestinală abundentă, care în mod normal este prezentă numai în colon [5].

Un exemplu bun de producător de H2 în organismul uman ar fi bacteria intestinală anaerobă gram-negativă Bilophila Wadsworthia care este a 3 cea mai frecvent izolată în apendicita perforată și gangrenoasă, dar întîlnită și într-o varietate de alte infecții. Acest organism efectuează un tip unic de respirație anaerobă în care taurina, un dizolvat organic major la mamifere, este folosită ca sursă de sulfit care servește ca acceptor terminal pentru lanțul de transport de electroni. Arătăm aici că hidrogenul molecular, unul dintre produsele majore ale bacteriilor fermentative din colon, este un substrat excelent de creștere pentru B. wadsworthia. S-au cuantificat activitățile enzimatice asociate cu oxidarea H2, formiat și piruvat pentru celulele obținute în diferite condiții de creștere [6].

Tipuri de teste de H_2

Testele de respirație sunt cel mai frecvent utilizate pentru diagnosticarea malabsorbției lactozei, sorbitolului și fructozei, testul de respirație cu glucoză (GBT) pentru creșterea bacteriană intestinală subțire (SIBO) și testul de respirație cu lactuloză pentru timpul de tranzit orocecal [7].

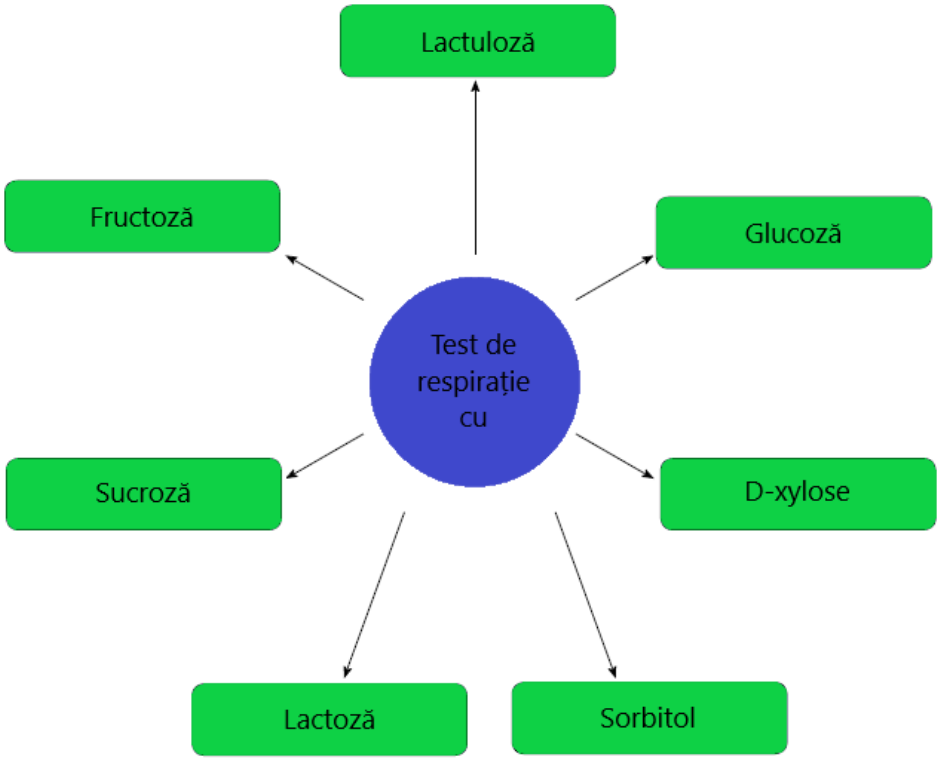


Figura 2. Tipuri de teste cu respirație

În condiții fiziologice, glucoza este imediat absorbită în intestinul subțire[8]. Cu toate acestea, dacă există o creștere bacteriană excesivă în intestinul subțire, fermentația bacteriană a glucozei care duce la producerea de H2 poate avea loc înainte de absorbția glucozei, care este măsurată prin creșterea concentrației de H2/CH4. Astfel, orice creștere ≥ 10 ppm a concentrației H2/CH4 în două citiri consecutive peste valoarea bazală trebuie considerată ca fiind semnificativă și indică despre SIBO [7].

Intoleranța la lactoză predomină în întreaga lume. Subiecții evită în general laptele și alte produse lactate pentru a-și îmbunătăți simptomele. Pentru o utilizare eficientă, lactoza necesită hidroliza de către enzima lactază. O creștere a concentrației H2/CH4 ≥ 20 ppm în două citiri consecutive peste valoarea bazală este considerată intoleranță la lactoză. Testul de respirație este acum considerat a fi cea mai practică și de încredere metodă de a diagnostica malabsorbția lactozei [7].

Tehnici pentru analiza respirației

În analiza respirației, respirația poate fi colectată și analizată ca condensat de respirație expirată sau ca gaz de respirație expirat. Prelevarea de probe de gaz prin colectare într-o pungă de gaz este mult mai populară decât metoda de condensare a respirației, în care respirația este condensată folosind SPME. Multe tehnici au fost utilizate pentru detectarea și măsurarea gazelor respiratorii, iar cele mai multe dintre ele pot fi clasificate în una din trei grupuri majore:

(1) analizoare convenționale care utilizează GC, de obicei cuplate cu spectrometrie de masă (MS) sau flacără. detectarea ionizării (FID) și noi tehnologii care utilizează noi metode de spectroscopie, cum ar fi spectrometria de mobilitate ionică (IMS);
(2) GC cuplat cu senzori de oxid de metal semiconductor (MOX);
(3) senzori noi, întâlniți în mod obișnuit sub formă de senzori unici sau matrice.

Pe lângă un rol în separarea gazelor, GC joacă un rol în preconcentrare deoarece spațiul de injectare a analitului instalat în capul coloanei cromatografice Sistemul GC are o funcție de concentrare a gazului [9].

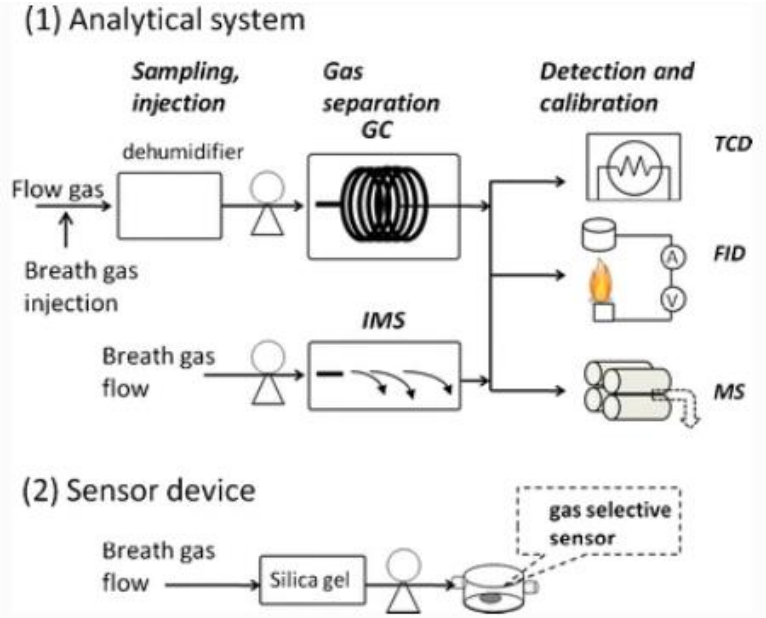


Figura 3. Un sistem de analiză a respirației constând din eșantionarea, separarea și detectarea gazelor respiratorii (sus) și sistemul mai simplu al unui dispozitiv cu senzor selectiv pentru gaz (jos). Detector de ionizare cu flacără FID, cromatografie gazoasă GC, spectrometrie de mobilitate ionică IMS, spectrometru de masă MS, detector de conductivitate termică TCD [9]

Senzorii MOX cercetați

Senzorii de gaze sunt dispozitive care permit detectarea anumitor gaze în dependență selectivitatea acestora. În figura 4 este reprezentată structura simplificată a unui senzor de gaze de unde se poate observa că acesta este format dintr-un receptor și traductor. Receptorul este materialul care intră în contact direct cu moleculele de gaz, în urma acestei interacțiuni receptorul își schimba proprietățile sale electrice. Traductorul are funcția de a detecta aceste schimbări și de a transforma rezultatul în semnal electric.

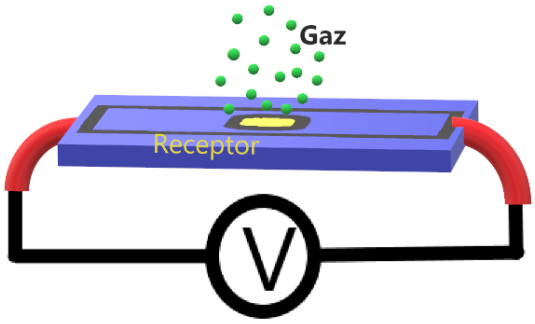


Figura 4. Schemă simplificată a unui senzor de gaze.

Senzorii de gaze posedă următoarele proprietăți:

Răspunsul – rata cu care se schimbă rezistența senzorului la aplicarea gazului. Acesta este un factor foarte important, valoarea ridicata a sensibilității la un gaz particular indică faptul că acest senzor este destul de bun pentru detectarea acestui gaz. Se determină prin formula:

$$S=\frac{I_{gaz}}{I_{aer}} \quad (1)$$

Unde:

S – Răspunsul;

$\mathbf{I_{gaz}}$ – Valoarea curentului la expunerea în gaz;

$\mathbf{I_{aer}}$ – Valoarea curentului la expunerea în aer.

Selectivitatea – abilitatea senzorului de a-și schimba proprietățile în dependență de tipul gazului aplicat. Acest factor este foarte important la detectarea gazelor deoarece în mediul înconjurător persistă diferite tipuri de gaze cu diferite concentrații care ar putea afecta rezultatele măsurării. Pentru a crește selectivitatea senzorilor de gaz, modificarea suprafeței sau dopajul cu diverși aditivi catalitici este utilizat pentru o mai bună adsorbție a componentelor țintă.

Timpii de răspuns și recuperare – timpul în care valoarea rezistenței se modifică de la 10% până la 90% se numește timpul de răspuns și respectiv de la 90% până la 10% se numește timpul de recuperare. În figura 5 este reprezentată metoda de determinare a timpului de răspuns și recuperare.

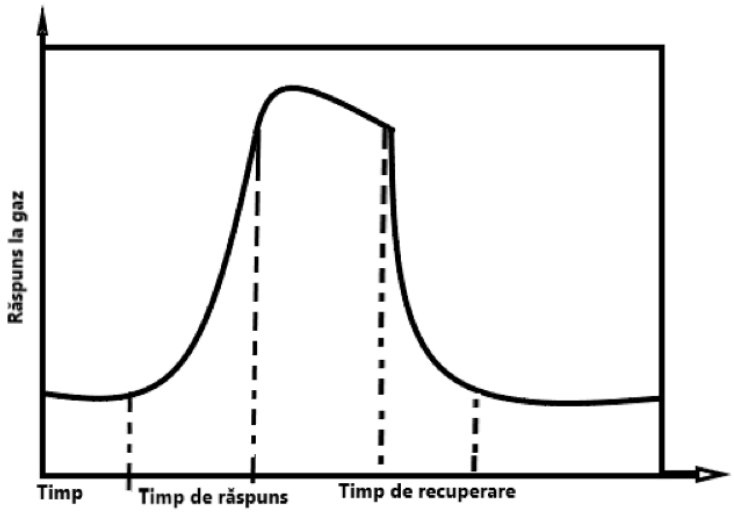


Figura 5. Schemă pentru determinarea timpului de răspuns și timpului de recuperare

Oxizii metalici sunt foarte des utilizați in fabricarea senzorilor de gaze din cauza proprietăților sale deosebite. Aceștea au demonstrat o capacitate foarte bună de detectare a gazelor cu o sensibilitate înaltă și cost de producție redus [66] comparativ cu alte metode de cercetare a gazelor, cum ar fi cromatografia de gaze – un echipament costisitor dar și cu întrebuințări limitate.

Trebuie remarcat faptul că o mare parte a senzorilor conductometrici de gaz (chimirezistentori) reprezintă dispozitive proiectate pe bază de nanocompozite cu oxid de metal, deoarece de obicei au în componența sa nanoparticulele de metal nobil încorporate în matricea de detectare a gazului prin oxidul de metal. Nanoclusterele sau nanoparticulele de metale nobile precum Pd, Pt, Au, Rh și Ag încorporate în matricea oxidului metalic, demonstrează și oferă proprietățile catalitice care pot produce modificări în interacțiunile chimice analit-oxid de metal, au posibilitatea de a crește activitatea catalitică a materialelor de detectare a gazului, pot modifica parametrii de adsorbție/desorbție și îmbunătățesc procesul de detectare [10].

Fabricarea senzorului

Pentru a obține structurile senzorilor utilizați în acest studiu au fost propuse mai multe metode și prin urmare sa creat o partidă de senzori pentru a compara rezultatele obținute și pentru a evidenția direcția de dezvoltare a acestora. Structurile CuO au fost obținute prin pulverizarea cupru metalic în condiții de vid pe substraturi de sticlă la 25 ° C folosind un sistem RF-magnetron personalizat. Apoi, structurile au fost tratate termic în condiții atmosferice normale conform unei lucrări anterioare [11]. Pentru structurile TiO2, a fost utilizată o combinație de două metode. Filmele de CuO au fost obținute prin sinteza soluțiilor chimice (abordarea SCS) pe un substrat de sticlă [12,13] urmată de un tratament termic în diferite regimuri. Ulterior, un film ultra-subțire de TiO2 a fost depus prin metoda de depunere a stratului atomic (ALD) la o temperatură de depunere de 75 °C. În final, probele au fost recoapte conform proceselor descrise într-o lucrare anterioară [14]. Detalii suplimentare despre parametrii de depunere sunt descrise într-o lucrare anterioară [15].

După fabricarea celor trei structuri diferite de detectare a gazelor, o peliculă subțire POLIMER PV de 25 nm a fost depusă pe fiecare dintre ele într-o configurație iCVD personalizată raportată în altă parte [16,17]. O pompă rotativă cu palete (Pfeiffer Vacuum Duo 10) a fost utilizată pentru evacuarea reactorului. Monomerul V4D4 (Abcr, 97 %) și inițiatorul PFBSF (Chempur, 95 %) au fost livrate prin supape de măsurare a debitului scăzut la debite de 0,2 sccm și, respectiv, 0,1 sccm, către reactor. Sistemul a fost operat în regim de debit continuu și o presiune de proces de 40 Pa a fost controlată de o supapă fluture (VAT 615) cuplată la un manometru capacitiv (MKS Baratron). Reactorul și toate liniile de monomeri și evacuare au fost încălzite la 110 °C pentru a preveni condensarea monomerului. Etapa de substrat a fost răcită cu un termostat (Huber CC-K6) la 32 °C. O matrice de filamente constând din sârmă NiCr (Goodfellow) este situată la 30 mm deasupra etajului substratului și a fost încălzită rezistiv folosind 64,05 W în timpul depunerii cu o sursă de alimentare.

Senzorul final poate fi închipuit în linii generale după figura 6. care demonstrează atît structura pe straturi cît și polimerul adăugat pentru îmbunătățirea acesteia.

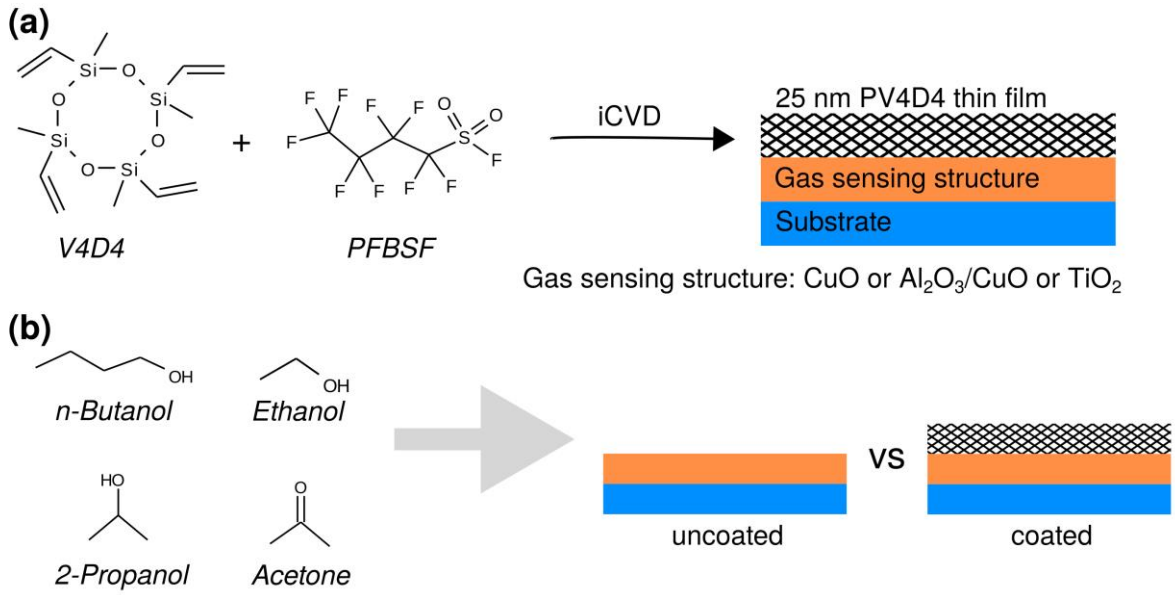


Figura 6. Ilustrarea schematică a studiului: (a) Senzorii de gaz CuO, Al2O3/CuO și TiO2 sunt acoperiți fiecare cu o peliculă subțire POLIMER PV de 25 nm crescută din monomer V4D4 și inițiator PFBSF utilizând procesul iCVD (b) Senzorii de gaz acoperiți și neacoperiți sunt apoi expuși la diferiți vapori organici și comparați pe baza răspunsurilor lor individuale ale gazelor [18].

Instrumentele utilizate

Pentru obținerea senzorilor s-au utilizat metode descrise anterior în această lucrare. Pentru obținerea rezultatelor, s-a utilizat o instalație specială făcută de Centrul de Nanotehnologii și Nanosenzori din cadrul UTM. În linii generale instalația în sine prezintă o sistemă cu tuburi și motoare pentru transmiterea unui tip de gaz de o anumită concentrație la senzor și un sistem de încălzire pentru modificarea temperaturii senzorului. Se adaugă la acest sistem un dispozitiv de măsurare a curenților – Keithley 2400 care lucrează după principiul schemei din figura 7.

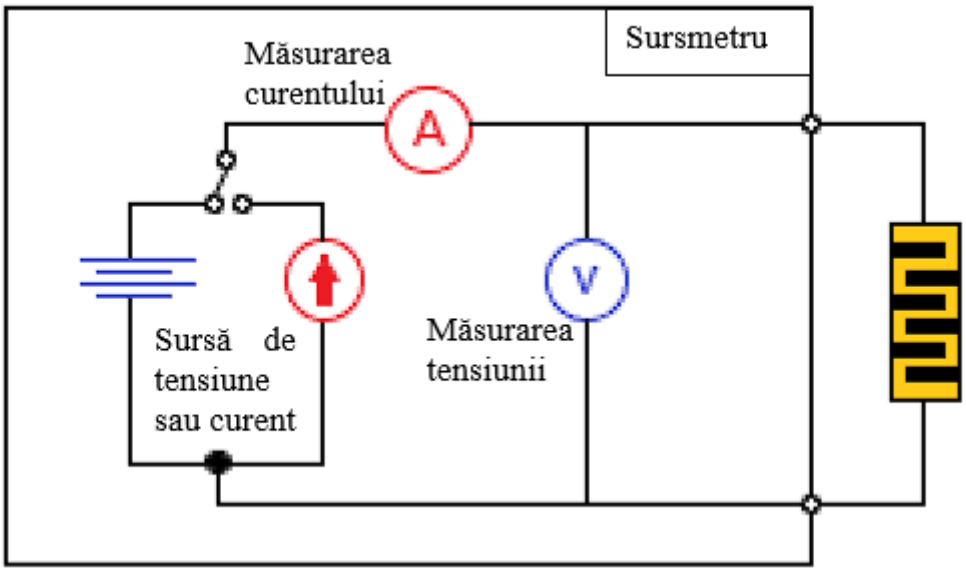


Figura 7. Schema principială de culegere a datelor de pe senzori MOX

Datele obținute s-au salvat într-un fișier care poate fi importat în softul de prelucrare OriginLab care are în linii generale aceleași componente ce se pot observa în figura 8. Cu ajutorul soft-ului dat prelucrarea rezultatelor a fost posibilă sub formă de grafice. Fiecare grafic este destinat unei temperaturi de testare, pe axa x este timpul iar pe axa y răspunsul senzorului la gaz.

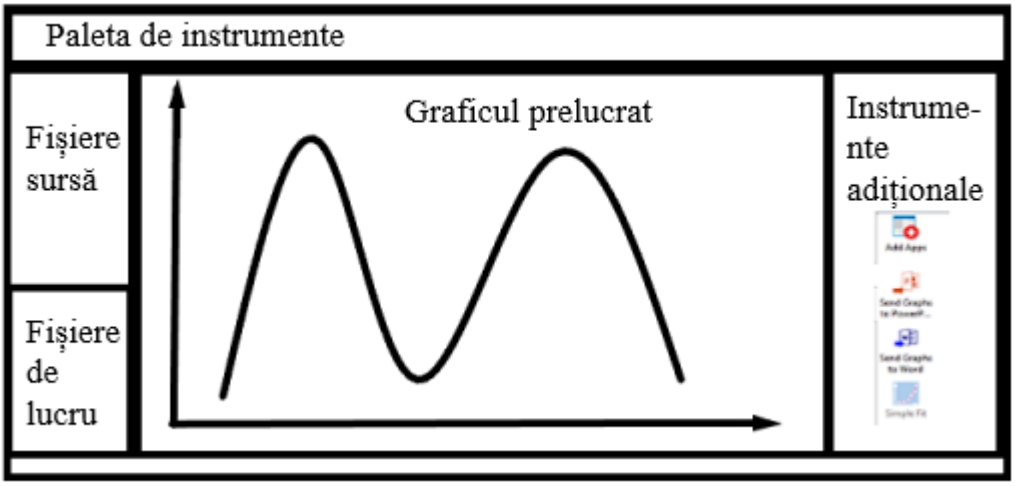


Figura 8. Interfața conceptuală de prelucrare a datelor colecatate de la senzori

Metode de cercetare

Procesul de măsurare a fost efectuat în timp real. Pentru aceasta a fost nevoie să se stabilizeze semnalul și să se aplice gaz pe senzor la anumite intervale de timp pentru a defini foarte bine impulsurile. După cum a fost menționat anterior, datele au fost colectate cu ajutorul sursmetrului Keithley 2400 conectat la calculator.

Transferul de date între calculator și sursmetru a fost efectuat cu ajutorul unui program scris în limbajul de programare grafic, adică procesul de măsurare a fost efectuat în timp real. Probele obținute au fost instalate pe o platformă formată dintr-un metal termoconductor și o sursă de căldură pe bază de o lampă halogenă, acestea fiind conectate la rîndul său la un stand. Sursmetrul se conecteza la probă cu ajutorul unui contact cu 12 pini. O altă componentă a stăndului este pompa de gaze care

realizează transportul de gaze: Hidrogen, Metanol, Etanol, Metan, n-Butanol, 2-Propanol, Acetonă și Amoniac din baloane cu diferite gaze și anumite concentrații pregătite din timp. Procesul de măsurare constă în aplicarea unei tensiuni de la sursmetru și monitorizarea variției curentului în timpul aplicării diferitor gaze la diferite temperaturi, începând cu temperatura camerei pâna la 150 -175 oC.

Rezultatele obținute se salveză în calculator sub formă de fișiere cu extensia .dat, cu ajutorul softului menționat anterior. Mai târziu aceste fișiere se introduc într-un alt soft care permite transformarea acestora în tabele sau grafice datorită caror este posibil de analizat rezultatele obținute.

Rezultatele obținute

Urmând o serie de procese, au fost obținuți 2 senzori de hidrogen în baza tehnologiilor care au fost deja expuse în nenumărate articole și chiar menționate în această lucrare.

Cererile masive de aplicații pentru senzori de înaltă performanță indică importanța controlului precis al caracteristicilor oxidului semiconductor. Longevitatea senzorului necesară pentru zonele îndepărtate din medii dure este obligatorie și poate beneficia foarte mult de abilitățile de autocurățare. Pentru a satisface această cerere, în studiul dat au fost se vor prezenta filme ultra-subțiri de TiO2 depuse cu grosimi diferite până la 15 nm pe substraturi de sticlă folosind depunerea stratului atomic (ALD). [19]

Filmele mai subțiri (15 nm) au prezentat performanțe mai mari de detectare a UV și a gazelor decât filmele mai groase (45 nm), ceea ce a fost legat de grosimea filmului comparabilă cu lungimea Debye [20]. Probele cu grosimea de 15-40 nm au prezentat un răspuns de ~ 600-650% la 100 ppm de H2 la temperatura de funcționare de 250°C. [19,21]

Deși oxizii metalici sunt bine cunoscuți pentru detectarea gazelor, umiditatea continuă să fie o problemă. Penntru îmbunătățirea proprietăților senzorului și protejarea la umiditatea a acestuia, în lucrarea [22] a fost utilizat un înveliș din monomerul 1,3,5-trimetil-1,3,5-trivinilciclotrisiloxan(PV3D3) este combinat cu inițiatorul fluorură de perfluorbutansulfonil pentru a obține proprietăți hidrofobe. Cu aceleași intenții a fost utilizat POLIMER PV, un înveliș superior care urmează a fi cercetat și a cărui performanțe necesită demonstrate.

Prima probă a cărei rezultate vor fi expuse prezintă un senzor de TiO2 de 15nm cu o acoperire de POLIMER PV care are o performanță destul de bună la hidrogen.

A doua probă reprezinăt un senzor de TiO2 de 25nm cu o acoperire de POLIMER PV care permite a face diferențierea proprietăților în dependență de grosimea stratului(cum a fost menționat anterior) și proprietățile stratului adăugat la detectarea gazelor.

Proba care urmează a fi cercetată a prezentat rezultate interesante, cu răspunsuri la gaz destul de clare și cu o selectivitate bună. Experiențele asupra probei au fost făcute la temperatura camerei, 250-350 °C cu umeditatea atmosferică relativ normală, de 20-21%.

În figura 9.a) este reprezentat răspunsul dinamic al oxidului de titan cu înveliș de POLIMER PV la H2 cu o concentrație de 100 ppm la o temperatură de lucru de 250 °C. Întru-cît răspunsul a fost convertit din formula inițială S=Igaz/Iaer în procente, din graficul dat reiese că răspunsul senzorului la hidrogen la datele menționate anterior a fost de 170%. Timpul de răspuns adică intervalul de timp în care are loc creșterea valorii de la 10% până la 90% este de aproximativ 3.53 sec, iar cel de recuperare adică intervalul de timp în care are loc descreșterea valorii de la 90% până la 10% este de aproximativ 12.56 sec. Tensiunea de 3,9V a fost aplicată timp de 18 secunde după care următoarele 20-30 secunde se aplică gazul în acest interval de timp se poate observa creșterea curentului. La deconectarea gazului curentul treptat revine la valoarea inițială. Pentru a obține mai multe pulsuri procedura sa repetat de 3 ori.

În figura 9.b) este reprezentat răspunsul dinamic al oxidului de titan cu înveliș de POLIMER PV la H2 cu o concentrație de 100 ppm la o temperatură de lucru de 300 °C. Întru-cît răspunsul a fost convertit din formula inițială S=Igaz/Iaer în procente, din graficul dat reiese că răspunsul senzorului la hidrogen la datele menționate anterior a fost de 228%. Timpul de răspuns adică intervalul de timp în care are loc creșterea valorii de la 10% până la 90% este de aproximativ 2.01 sec, iar cel de recuperare adică intervalul de timp în care are loc descreșterea valorii de la 90% până la 10% este de

aproximativ 26.51 sec. Tensiunea de 2,4V a fost aplicată timp de 20 secunde după care următoarele 20-30 secunde se aplică gazul în acest interval de timp se poate observa creșterea curentului. La deconectarea gazului curentul treptat revine la valoarea inițială. Pentru a obține mai multe pulsuri procedura sa repetat de 3 ori.

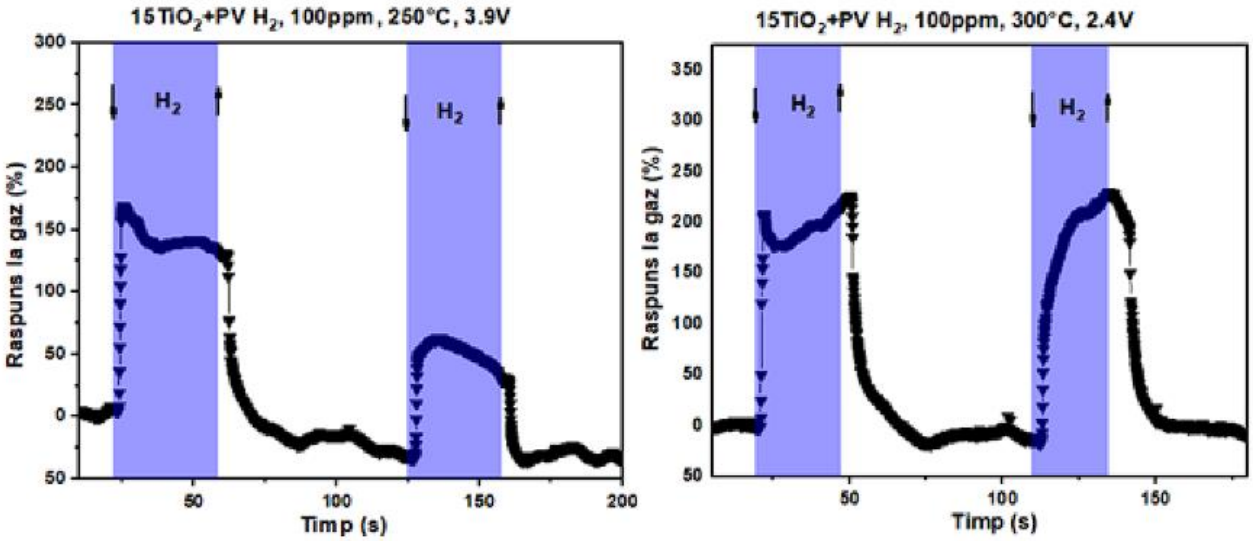


Figura 9.a) Răspunsul H2 la temperatura de 250°C. b) Răspunsul H2 la temperatura de 300°C

În figura 10. este reprezentat răspunsul dinamic al oxidului de titan cu înveliș de POLIMER PV la H2 cu o concentrație de 100 ppm la o temperatură de lucru de 350 °C. Întru-cît răspunsul a fost convertit din formula inițială S=Igaz/Iaer în procente, din graficul dat reiese că răspunsul senzorului la hidrogen la datele menționate anterior a fost de 595%. Timpul de răspuns adică intervalul de timp în care are loc creșterea valorii de la 10% până la 90% este de aproximativ 2.71 sec, iar cel de recuperare adică intervalul de timp în care are loc descreșterea valorii de la 90% până la 10% este de aproximativ 2.11 sec. Tensiunea de 0,9V a fost aplicată timp de 20 secunde după care următoarele 20-30 secunde se aplică gazul în acest interval de timp se poate observa creșterea curentului. La deconectarea gazului curentul treptat revine la valoarea inițială. Pentru a obține mai multe pulsuri procedura sa repetat de 3 ori.

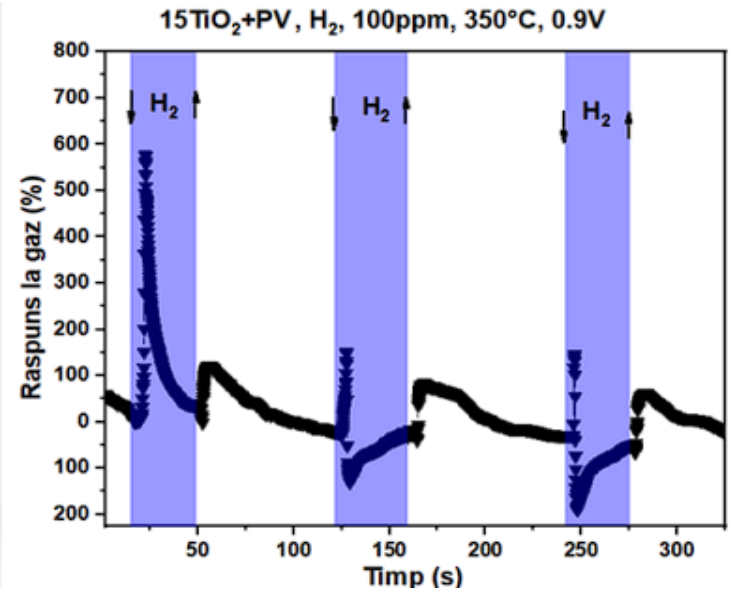


Figura 10. Răspunsul H2 la temperatura de 350°C

În cazul dat, semnalul se prezintă într-o formă care indică că tipul senzorului tinde să își schimbe prorpietățile la temperaturi mari. Deși impulsul inițial se prezintă ca a fi unul mărit, următoarele sunt reduse, cu 2 vîrfuri în loc de unul. Așa tip de senzor se va studia în altă lucrare.

Ca o concluzie asupra probei date, răspunsurile la H2 sunt foarte dependente de temperatură. De la 250 °C pînă la temperatura camerei proba nu a dat răspunsul așteptat.

În ciuda faptului că destinația și scopul măsurărilor efectuate asupra acestei probe, era obținerea rezultatelor răspunsurilor la H2, este imposibil a nu menționa faptul că POLIMER PV a avut un impact asupra proprietăților senzorului întru-cît datele diferă de cele din alte lucrări.

Pentru a rezuma mai bine rezultatele primei probe, este necesar să se compare rezultatele într-o diagramă care va arăta toate gazele testate și răspunsurile acestora. După cum se observă în figura 11, proba data în condițiile de mediu uscate la temperaturile deja menționate a obținut răspunsuri stabile doar la hidrogen.

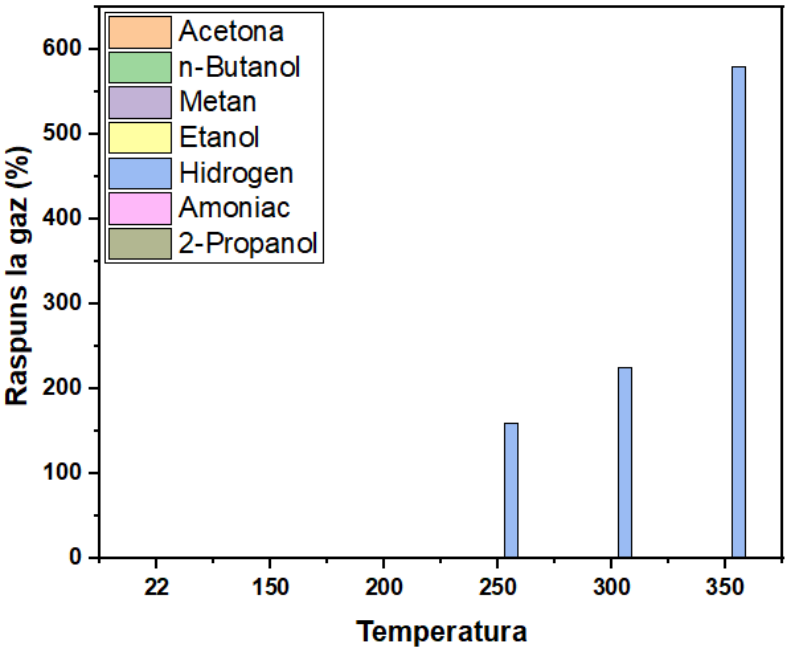


Figura 11. Comparația răspunsurilor gazelor

Concluzii

Direcția testelor neinvazive este una foarte atrăgătoare pentru sfera medicală. Procesele de diagnosticare ce permit siguranța pacientului după testul efectuat devin din ce în ce mai solicitate pentru domeniul dat. Corpul uman deși a fost studiat pînă la mici detalii, încă oferă o mulțime de enigme și soluții ulterioare pentru diagnosticarea și tratarea nenumăratelor boli care afectează pacientul începînd cu viața cotidiană și finisînd cu unele cazuri letale.

În studiul dat s-a determinat că există o legătură între concentrațiile gazelor din tractul digestiv și unele sindroame și boli. Diferite sindroame cum ar fi cel al colonui iritabil, creșterea excesivă a bacteriilor în intestin, intoleranța la lactoză și alte zaharide au o legătură definită cu hidrogenul, dar și alte gaze. Acestea ajung în expir și oferă informații zilnic în dependență de concentrațiile determinate.

Desigur că această tehnologie nu este nouă întru-cît testele de respirație pot fi recunoscute în determinarea alcoolemiei. Testele de respirație cu hidrogen există și ele după cum a fost menționat anterior, cu destinația determinării și diagnosticării intoleranței la lactoză. Totuși aportul lucrării date este că se cercetează un alt tip de senzori.

Datele obținute oferă o temelie bună pentru cercetarea în continuare potențialului succes al senzorilor pe bază de oxizi metalici în sfera medicală și anume partea de diagnosticare neivazivă a pacienților cu un risc inexistent pentru sănătatea acestora din motivele utilizării unui potențial dispoztiv de tipul dat.

Health Breath Test(HBT) – Test de sănătate a respirației poate fi subînțeles orice test care presupune studierea expirului și diagnosticare componenței acestuia. Chiar dacă la moment studiul dat oferă date obținute doar pe partea de hidrogen, ulterior se pot dezvolta testere mult mai avansate ce vor studia expirul în baza mai multor gaze expirate. Pentru aceasta este nevoie de a obține mai multe date clinice legate de sănătatea pacienților și legătura acestor stări cu anumite concentrații de gaze.

Ajustarea perioadei de aplicare a gazului și anume cu intervalele menționate de 18-30 secunde pentru aplicare a permis determinarea formării unui răspuns clar care urmează a fi convertit spre un diagnostic clar. Totuși chiar și în forma expusă în graficile prelucrate, rezultate de 575% și 550% ca răspuns al senzorului la H2, deja permit dezvoltarea testerului de respirație cu H2.

Mulţumiri. Autorul lucrării oferă sincere mulțumiri coordonatorului științific dr.hab., prof. univ. Oleg Lupan pentru tot suportul, îndrumarea și resursele necesare în crearea cecetării date. De asemenea sincere mulțumiri se acordă departamentului de Microelectronică și Inginerie Biomedicală care oferă mereu suport studenților, masteranzilo și motivează tinerii cercetători să continue la ciclul III – doctorat să vină și să continue dezvoltarea științei în Republica Moldova.

Referinţe

1. MICHELLE B. G.; COSTA, I. L.; AZEREDO Jr., R. D.; MARCIANO, L. M.; CALDEIRA, M. B. „*Evaluation of small intestine bacterial overgrowth in patients with functional dyspepsia through H_2 breath test*” Arq Gastroenterol. 2012 Dec;49(4):279-83
2. NSA, E.; CHO, H-B.; SONG, Y.; LEE W., SEKINO, T.; CHOA Y-H. *Room-Temperature H2 Gas Sensing Characterization of Graphene-Doped Porous Silicon via a Facile Solution Dropping Method*. Sensors. 2017; 17(12):2750
3. CREŢU, V.; LUPAN, C.; MANCIU, N.; ABABII, N.; POSTICA, V.; RAILEAN, S.; GALSTYAN, V.; CHOW, L.; PAUPORTE, T. *Thin films of copper oxide nanostructured via rapid thermal processing*. In: Electronics, Communications and Computing IC|ECCO-2021. Editia a 11-a, 21-22 octombrie 2021, Chişinău. Chișinău, Republica Moldova: Technical University of Moldova, 2021, p. 29. ISBN 978-9975-45-776-7.
4. RANA, SV.; MALIK, A.; *Hydrogen breath tests in gastrointestinal diseases*. Indian J Clin Biochem. 2014 Oct;29(4):398-405. doi: 10.1007/s12291-014-0426-4. Epub 2014 Mar 21. PMID: 25298621; PMCID: PMC4175689.
5. MICHAEL, D.; LEVITT, M.D.; „*Production and excretion of hydrogen gas in man*”; The New England Journal, july 17, 1969 DOI: 10.1056/NEJM196907172810303
6. da SILVA, S.M.; VENCESLAU, S.S., FERNANDES, C.L.V. et al. *Hydrogen as an energy source for the human pathogen Bilophila wadsworthia* . Antonie van Leeuwenhoek 93, 381–390 (2008). https://doi.org/10.1007/s10482-007-9215-x
7. RANA, SV.; MALIK, A; *Breath tests and irritable bowel syndrome*. In: World J Gastroenterol. 2014 Jun 28;20(24):7587-601. doi: 10.3748/wjg.v20.i24.7587. PMID: 24976698; PMCID: PMC4069289.
8. BOND, JH.; LEVITT, MD. *Use of breath hydrogen (H2) to quantitate small bowel transit time following partial gastrectomy.* In: J Lab Clin Med 1977; 90: 30-36 [PMID: 874370]
9. SHIN, W. *Medical applications of breath hydrogen measurements*. In: Anal Bioanal Chem 406, 3931–3939 (2014). https://doi.org/10.1007/s00216-013-7606-6
10. KOROTCENKOV, G., B. K. CHO. "*Metal oxide composites in conductometric gas sensors: Achievements and challenges*." *Sensors and Actuators B: Chemical* 244 (2017): 182-210.
11. WANG, Y., LIU, L., XU, L., CAO, X., LI, X., HUANG, Y., & ZHU, W. "Ag 2 O/TiO 2/V 2 O 5 one-dimensional nanoheterostructures for superior solar light photocatalytic activity." *Nanoscale* 6.12 (2014): 6790-6797.
12. WANG, Y.; CHU, W.; WANG, S.; LI, Z.; ZENG, Y.; YAN, S.; SUN, Y. "*Simple synthesis and photoelectrochemical characterizations of polythiophene/Pd/TiO2 composite microspheres*." *ACS applied materials & interfaces* 6.22 (2014): 20197-20204.

13. EPIFANI, M.; DÍAZ, R.; FORCE, C.; COMINI, E.; ANDREU, T.; R. ZAMANI, R.; ARBIOL, J.; SICILIANO, P.; FAGLIA, G.; MORANTE, JR.; "*Colloidal counterpart of the TiO2-supported V2O5 system: a case study of oxide-on-oxide deposition by wet chemical techniques. Synthesis, vanadium speciation, and gas-sensing enhancement.*" *The Journal of Physical Chemistry C* 117.40 (2013): 20697-20705.
14. ZHENYU, Li., HONGNAN, Zhang., WEI, Zheng., WEI, Wang., HUIMIN, Huang., CE, Wang., ALAN, G. MacDiarmid., YEN, Wei. "Highly sensitive and stable humidity nanosensors based on LiCl doped TiO2 electrospun nanofibers." *Journal of the American Chemical Society* 130.15 (2008): 5036-5037.
15. ABDEL-KARIM, R., REDA, Y., & ABDEL-FATTAH, A.. *Nanostructured materials-based nanosensors*. In: *Journal of The Electrochemical Society*, *167*(3) (2020), 037554.
16. LUPAN, O.; CRETU, V.; POSTICA, V.; AHMADI, M.; ROLDAN CUENYA, B.; CHOW, L.; TIGINYANU, I.; VIANA, B.; PAUPORTÉ, T.; ADELUNG, R. Sensors and Actuators B 223 (2016) 893-903.
17. LOU Z., LI, F., DENG, J., WANG L., & ZHANg T. (2013). *Branch-like hierarchical heterostructure (α-Fe2O3/TiO2): a novel sensing material for trimethylamine gas sensor. ACS applied materials & interfaces*, 5(23), 12310-12316.
18. SCHRÖDER, S.; ABABII, N.; BRÎNZĂ, M.; MAGARIU, N.; ZIMOCH, L.; BODDULURI, M.T.; STRUNSKUS, T.; ADELUNG, R.; FAUPEL, F.; LUPAN, O. *Tuning the Selectivity of Metal Oxide Gas Sensors with Vapor Phase Deposited Ultrathin Polymer Thin Films*. In: *Polymers* 2023, *15*, 524. https://doi.org/10.3390/polym15030524
19. LUPAN, O.; POSTICA, V.; ABABII, N.; REIMER, T.; SHREE, S.; HOPPE, M.; POLONSKYI, O.; SONTEA, V.; CHEMNITZ, S.; FAUPEL, F.; ADELUNG, R.; *Ultra-thin TiO2 films by atomic layer deposition and surface functionalization with Au nanodots for sensing applications, Materials Science in Semiconductor Processing*, Volume 87, 2018, Pages 44-53, ISSN 1369-8001, https://doi.org/10.1016/j.mssp.2018.06.031
20. JING, B.; ZHOU, B. "*Titanium dioxide nanomaterials for sensor applications.*" *Chemical reviews* 114.19 (2014): 10131-10176.
21. ABABII, N.; HOPPE, M.; SHREE, S.; VAHL, A.; ULFA, M.; PAUPORTÉ, T.; VIANA, B.; CRETU, V.; MAGARIU, N.; POSTICA, V.; SONTEA, V.; TERASA, MI.; POLONSKYI, O.; FAUPEL, F.; ADELUNG, R.; LUPAN, O. *Effect of noble metal functionalization and film thickness on sensing properties of sprayed TiO2 ultra-thin films.* In: Sensors and Actuators A: Physical, Volume 293, 2019, Pages 242-258, ISSN 0924-4247, https://doi.org/10.1016/j.sna.2019.04.017.
22. S. SCHRÖDER, N. ABABII, O. LUPAN, J. DREWES, N. MAGARIU, H. KRÜGER, T. STRUNSKUS, ADELUNG, R.; HANSEN, S.; FAUPEL, F.; *Sensing performance of CuO/Cu2O/ZnO:Fe heterostructure coated with thermally stable ultrathin hydrophobic PV3D3 polymer layer for battery application, Materials Today Chemistry*, Volume 23, 2022, 100642, ISSN 2468-5194, https://doi.org/10.1016/j.mtchem.2021.100642

PROCESOR DIGITAL DE SUNET PE BAZA MICROCONTROLERULUI STM32F446RE

Andrei SPATARI[1*], Adrian BÎRNAZ[2]

[1]*Departamentul Informatică și Ingineria Sistemelor, grupa CR-202, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău. Republica Moldova*
[2]*Departamentul Microelectronică și Ingineria Biomedicală, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău. Republica Moldova*

*Autorul corespondent: Spatari Andrei, andrei.spatari@iis.utm.md

Îndrumător/coordonator științific: Baran Igor, programator-analist, „AROBS – Software" SRL [1], Chișinău, Republica Moldova.

Rezumat. *În cadrul acestui articol se prezintă un dispozitiv care are posibilitatea de achiziționare a semnalului audio prin intermediul unui convertor analog-digital și procesarea ulterioară a acestuia de microcontroler din familia STM32 prin adăugarea efectului de ecou cu aplicarea semnalului rezultant la ieșirea convertorului digital-analog.*

Cuvinte cheie*: NUCLEO-F446RE, semnal audio, amplificator operațional, procesare numerică.*

Introducere

În ultimul timp procesarea semnalelor, în special cel sonor, reprezintă unul dintre cele mai studiate domenii ale ingineriei [2], care necesită cunoștințe destul de vaste atât a ce reprezintă acest semnal (ce proprietăți posedă), cât și variațiunile de procesare [3]. În acest articol se descrie un dispozitiv care permite procesarea semnalului audio și care în viitor ar putea deveni un DJ Controller complex.

Scopul acestui proiect a fost crearea unui dispozitiv, care ar permite modificarea semnalului la ieșire (printr-un efect de ecou) față de cel de intrare. Pentru monitorizarea acestui efect a fost proiectat un „*Control panel*" ce constă din 3 potențiometre și 24 de LED-uri. În dependență de valorile obținute de la aceste potențiometre și supuse unor formule, s-a putut observa diferența dintre semnalul de ieșire și cel de intrare. Acesta este unul dintre cele 2 blocuri de bază ale acestui dispozitiv: „*Control panel*" și „*Input signal*". Cel din urmă permite ridicarea semnalului deaspura axei Ox, pentru a obține, doar valori pozitive, fiindcă convertorul analog-digital al plăcii NUCLEO-F446RE nu lucrează cu valori negative.

Componentele sistemului

Elementele de bază ale dispozitivului:

- 3 amplificatoare TL071;
- placa de dezvoltare NUCLEO-F446RE;
- 2 baterii de 9 V fiecare;
- 3 potențiometre D103 cu capacitatea de 10 kΩ fiecare;
- 7 condensatoare electrolitice: 6 de 1 μF la 50 V, 1 de 10 μF la 50 V;
- 1 diodă Zenner din seria BTZ55C3V6;
- 1 diodă Schottky din seria 1N5822;
- 24 de LED-uri de 5mm (12 verzi, 6 galbene și 6 roșii);
- 29 de rezistențe: 24 de 470 Ω, 1 de 4.7 kΩ, 1 de 24 kΩ, 1 de 1 kΩ, 1 de 3 kΩ, 1 de 36 kΩ.
- 2 porturi audio-jack de tip mamă

Structura și principiul de funcționare al blocului „Control panel"

În figura Fig. 1 este reprezentată schema de structură a dispozitivului dat, în care pot fi evidențiate blocurile din care constă.

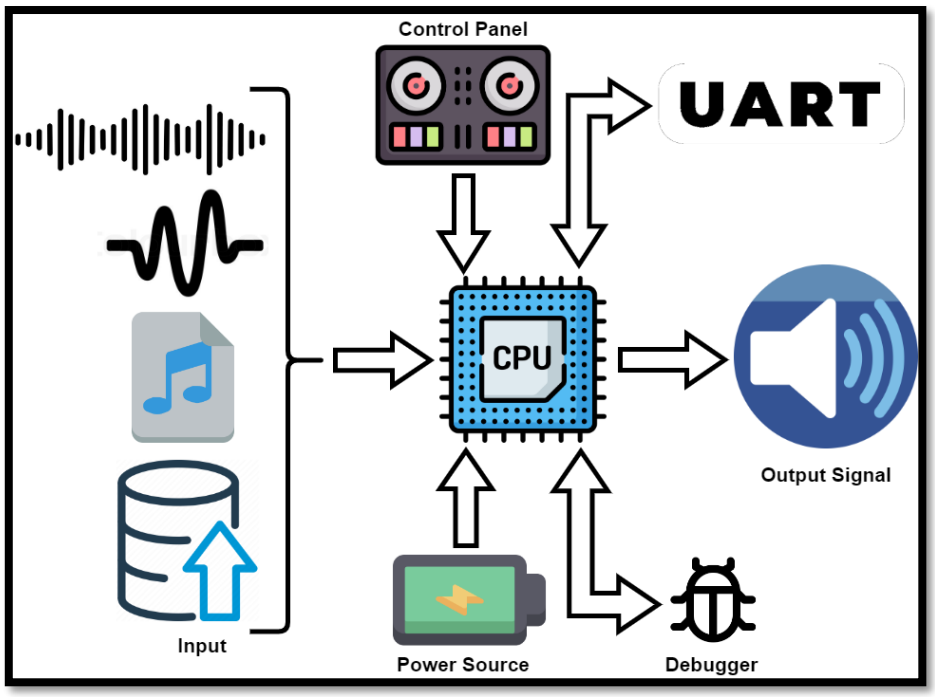


Figura 1. Schema de structură a dispozitivului „Sound Processor”

Blocul „Control panel” este format din 3 potențiometre din seria D103 cu nominalul de 10 kΩ fiecare. Acestea sunt conectate la 3 pini ai plăcii NUCLEO-F446RE [4], iar colectarea datelor se face cu ajutorul convertorului analog-digital (ADC)[5], setat în modul multicanal. Odată colectate, aceste valori sunt supuse unor formule matematice, care ulterior sunt utilizate în algoritmul de procesare. Spectrul de valori este cuprins între 0 și 4095, deoarece convertorul analog-digital este pe 12 biți.

Valoarea primului potențiometru este introdusă în Ec. (1). Rezultatul dat reprezintă delay-ul care la rândul său arată numărul de poziții cu care se va deplasa semnalul din vectorul cu dimensiunea maximală. Intervalul delay-ului este cuprins între 0 și 39 de unități.

Valoarea celui de-al doilea potențiometru este introdusă în Ec. (2). Rezultatul obținut este volumul porțiunii de semnal audio care a fost suprapus (adunat) la semnalul de la intrare. Intervalul de volum este între 0 și 100 de unități.

Valoarea celui de-al treilea potențiometru este introdusă în Ec. (3). Rezultatul obținut este volumul semnalului total care a fost scos în exterior prin intermediul unui convertor digital-analog (DAC). Intervalul acestui rezultat este de 20 și 100 de unități.

$$delay = 39 - \frac{valuePotentiometer1}{105} + 1 \quad (1)$$

$$delayVolume = \frac{valuePotentiometer2}{41} \quad (2)$$

$$volume = \frac{valuePotentiometer3}{51} + 20 \quad (3)$$

Pentru fiecare potențiometru au fost plasate câte 8 LED-uri de 5 mm (4 verzi, 2 galbene și 2 roșii), care au rolul de indicator. Astfel, intervalul de valori ai potențiometrului este împărțit în 8 segmente egale, iar fiecărui LED îi corespunde unul dintre aceste segmente.

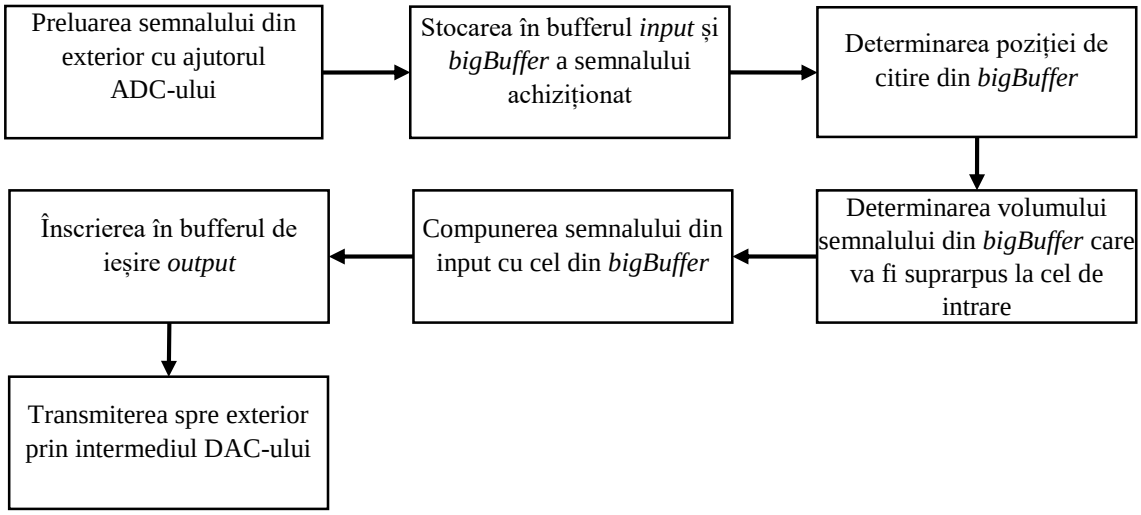


Figura 2. Algoritmul de lucru al procesării semnalului

Algoritmul de funcționare al dispozitivului este prezentat în Fig. 2 și constă în preluarea datelor de la potențiometre (printr-un ADC multicanal), semnalul audio de la celălalt convertor analog – digital, stocarea acestuia în 2 vectori, unul de lungimea de 2000 de elemente (*input*), iar celălalt de 40000 de elemente (*bigBuffer*) în care se înscrie în continuu. Inițial este determinată poziția elementului din vectorul *bigBuffer*, cu ajutorul valorii *delay*, care va fi suprapus semnalului de intrare. După, acestui element îi este dat volumul (cât de tare să fie auzit) prin valoarea *delayVolume*. Următorul pas este adunarea semnalului din *input* cu cel obținut și rezultatului dat îi este atribuit valoarea *volume*, cu care să fie auzit deja în exterior.

Convertorul analog-digital al plăcii NUCLEO-F446RE poate opera cu tensiuni cuprinse între 0 – 3.3V, pe când semnalul audio, generat de majoritatea plăcilor de sunet, are valoarea de amplitudine, la volum maxim, egală cu ~500mV, Fig. 3 (a) semnalul *V(V1:+)*. Respectiv valoarea minimă a amplitudinii este de -250mV. Pentru a ajusta semnalul de ieșire a plăcii audio la valorile de lucru a ADC-ului este necesar de adăugat o tensiune de deplasare pentru a poziționa semnalul deasupra axei Ox, astfel eliminând valorile negative a tensiuni de intrare. Cu toate astea amplitudinea semnalului va fi prea mică pentru prelucrarea ulterioară din acest motiv este necesară amplificare lui. Pentru a satisface cerințele expuse mai sus a fost concepută schema din Fig. 3, realizată în programul Schematics OrCAD [6].

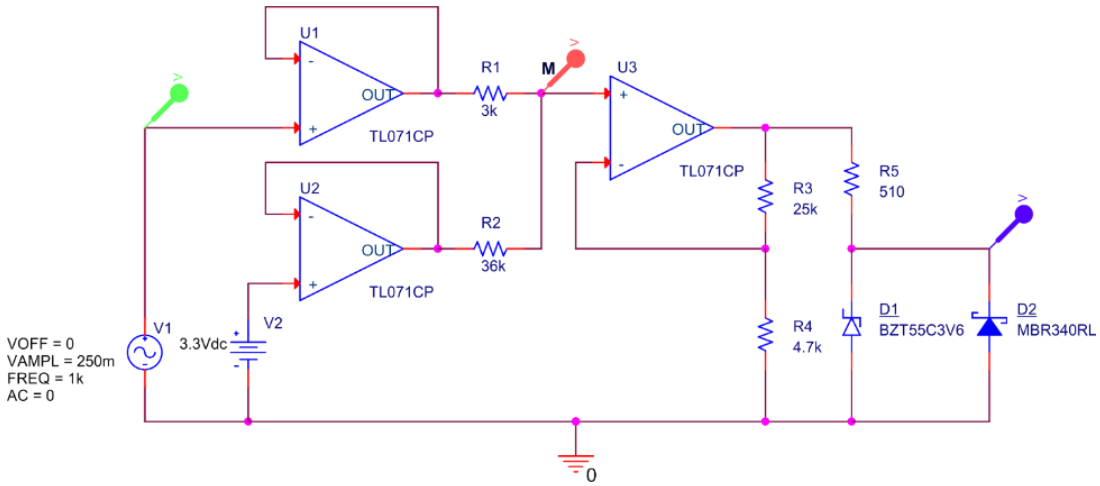


Figura 3. Schema electrică a circuitului de amplificare și sumare a semnalului util cu tensiunea de deplasare

Amplificatoarele operaționale [7] *U1* și *U2* sunt utilizate în calitate de tampon pentru a nu permite trecerea semnalului de deplasare la intrarea plăcii de sunet, dar și obținerea unei impedanțe mici necesare pentru sumarea semnului de intrare cu semnalul de deplasare. Placa NUCLEO-F446RE are o sursă stabilizată de tensiune egală cu 3.3V. Această tensiune poate fi utilizată ca tensiune de deplasare, *V2*. Semnalele sumate vor avea diferite ponderi. Ponderea semnalului de la sursa *V2* trebuie să fie mai mică întru cât avem nevoie de a deplasa semnalul cu +250mV. Pentru a seta ponderea semnalelor se vor alege rezistențele *R1* și *R2* conform Ec. (4) și Ec. (5).

$$P_{V1} = \frac{R2}{R1 + R2} \tag{4}$$

$$P_{V2} = \frac{R1}{R1 + R2} \tag{5}$$

Tensiunea rezultantă la sumare semnalelor de la sursele *V1* și *V2* o primim în punctul M de pe schemă, Fig. 3(a) semnalul *V(R1:2)*. Tensiunea din punctul *M*, V_M, poate fi calculată după formula din Ec. 6.

$$V_M = V1 \cdot P_{V1} + V2 \cdot P_{V2} \tag{6}$$

Tensiunea V_M se aplică la intrarea neinversoare a amplificatorului operațional *U3*. Coeficientul de amplificare *k* se setează prin intermediul rezistoarelor *R3* și *R4* conform Ec. (7) și în rezultat vom obține semnalul amplificat în limitele necesare, prezentat în Fig. 3(b).

$$k = \frac{R3}{R4} + 1 \tag{7}$$

Pentru a proteja intrarea ADC-ului de la suprasarcini sau de potențial negativ, care în unele cazuri poate apărea la ieșire amplificatorului operațional *U3*, este conceput circuitul format din rezistorul *R5* dioda stabilizatoare *D1* și dioda Schottky *D2*. În cazul în care tensiunea de ieșire va fi mai mare decât tensiunea de stabilizare, dioda Zenner polarizată invers va fi străpunsă, astfel surplusul de tensiune va cădea pe rezistorul *R5*. Valoarea rezistorului poate fi calculat conform Ec. (8)

$$R5 = \frac{V_{U3} - V_Z}{I_Z \cdot I_{ADC}} \tag{8}$$

Dacă la ieșirea amplificatorului *U3* se va forma potențial negativ dioda *D2*, se va deschide rapid și datorită căderii mici de tensiune pe joncțiune, semnalul nu va avea valori negative mai mici de -200mV, protejând astfel convertorul analog-digital. Simularea acestui caz este prezentat în Fig. 3(c).

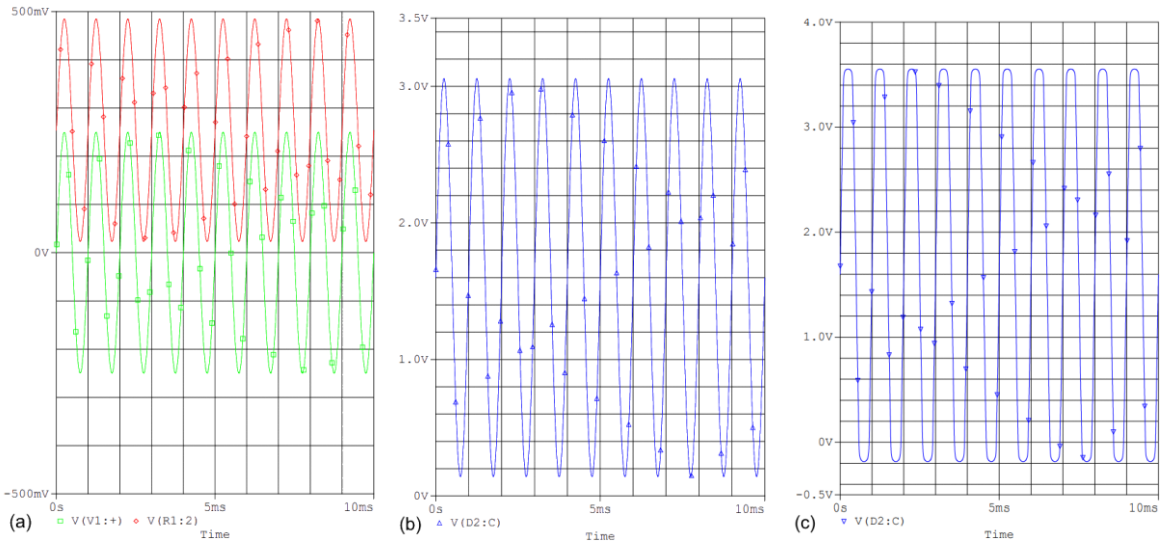


Figura 3. Semnale obținute în urma simulării circuitului. (a) Semnalul de intrare și semnalul de intrare deplasat. (b) Semnalul de ieșire. (c) Semnalul de ieșire limitat

Concluzii

Dispozitivul elaborat reprezintă un sistem de achiziție a semnalului auditiv care oferă posibilitatea procesării acestuia și adăugarea efectului de ecou cu transmiterea spre dispozitivele de redare a semnalului sonor. Funcționalitatea dispozitivului proiectat poate fi extinsă până la funcționalitatea unui DJ Controller profesionist. În rezultatul cercetării domeniului și proiectării dispozitivului prezentat a fost implementat circuitul de amplificare și translare a semnalului de intrare deasupra axei Ox, fiind format din 3 amplificatoare operaționale TL071[8], rezistențe și diode. De asemenea au fost efectuate și diferite testări, atât separate, pentru fiecare bloc, cât și asupra produsului final pentru verificarea corectitudinii algoritmului. Dispozitivul proiectat poate fi utilizat de către ingineri sau studenți în diferite domenii ca: generator sau analizor de semnal, filtrare și pentru reproducerea sunetelor. Astfel, în procesul de proiectare, au fost aplicate cunoștințele teoretice și abilitățile practice obținute în cadrul procesului de studiu la disciplinele de specialitate.

Mulțumiri. Aducem sincere mulțumiri companiei „AROBS Software” SRL pentru suportul acordat în realizarea acestui proiect.

Referințe

1. AROBS Software [online]. [Accesat 24.02.2023]. Disponibil: https://acem.md/en/teams/arobs-software-2/
2. Importance of audio signal processing [online]. [Accesat 24.02.2023]. Disponibil: https://www.pathpartnertech.com/audio-signal-processing-understanding-digital-analog-audio-signal-processing/
3. ROTAR D. Capitolul 1. In *PROCESAREA DIGITALĂ A SEMNALELOR*. Chișinău: TEHNICA INFO, 2005, pp 7 – 9.
4. Characteristics of NUCLEO-F446RE [online]. [Accesat 25.02.2023]. Disponibil: https://os.mbed.com/platforms/ST-Nucleo-F446RE/
5. UNIVERSITATEA TEHNICĂ A MOLDOVEI. CIRCUITE PENTRU CONVERSIA DATELOR: CONVERTOARE DIGITAL-ANALOGICE ȘI ANALOG-DIGITALE. In *CIRCUITE ANALOGICE ȘI DE CONVERSIE*. Chișinău: U.T.M, 2008, pp 77,78
6. LUPAN O. Studiul amplificatoarelor cu ajutorul programului Schematics OrCAD. In *INSTRUMENTE SOFTWARE ÎN ELECTRONICĂ*. Chișinău: U.T.M, 2006, pp 27 – 37
7. ДОРОНИН Д., САНДУ Ю., БЫРНАЗ А., Проектирование и расчет измерительной схемы для резистивного датчика температуры PT-100. In *Conferinţa Tehnico-Ştiinţifică a Studenţilor, Masteranzilor și Doctoranzilor, Universitatea Tehnică a Moldovei, Ediția 2022,* Chișinău, 29-31 martie 2022, Vol. I, pp 220-224
8. *Amplificator operațional TL071* [online]. [Accesat 28.02.2023]. Disponibil https://www.st.com/resource/en/datasheet/tl071.pdf

SUBSECŢIA BAZE DE DATE

DATABASE SUBSECTION

MEMCACHED – РАСПРЕДЕЛЕННАЯ СИСТЕМА КЭШИРОВАНИЯ ПАМЯТИ

Николай СКУРТУ

Департамент Программной Инженерии и Автоматики, Группа TI-195, Факультет Вычислительной Техники и Микроэлектроники, Технический Университет Молдовы, Кишинёв, Республика Молдова

Автор корреспондент: Николай СКУРТУ, e-mail: scurtu.nicolai@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация. *В данной статье представлены общие представления о методике кэширования, применяемой для баз данных, на примере одной из систем – Memcached. Поскольку Memcached – это относительно простая, долго существующая и устоявшаяся система, на её примере легко показать основаные достоинства, и что немаловажно, недостатки данного подхода к кэшированию в оперативной памяти.*

Ключевые слова: *Memcached, кэш, LRU, клиент-серверная архитекура, оперативная память, база данных*

Введение

При работе с базами данных неизбежно возникают ситуации, когда некоторые из запросов к серверу обслуживаются особенно долго по сравнению с остальными. Это может происходить из-за того, что запрос написан неэффективно, индексы используются неудачно, либо по другим причинам. Все эти ситуации не являются предметом интереса для данной статьи, но бывают ситуации, где даже, если учесть все вышеперечисленные аспекты, все равно время выполнения запроса оставляет желать лучшего.

Например, в следующем SQL-запросе обрабатывается относительно большой объём данных [1]:

```
1 SELECT *
2 FROM hugetable
3 WHERE timestamp > lastweek
4 ORDER BY timestamp ASC LIMIT 50000;
```

Конечно, для начала необходимо подумать, нельзя ли снизить время выполнения таких длительных запросов, учитывая особенность конкретной предметной области. Если, например, выяснится, что в 90% случаях пользователю нашей базы данных требуется всего 10 первых записей, но не в коем-случае не 50 000, то, очевидно, можно создать нужные индексы, понизить лимит запрашиваемых записей до 10 и сделать дополнительную опцию для запроса бóльшего количества записей.

Тем не менее, остается вопрос, а что же делать, если количество запрашиваемых данных в запросе оптимизировано, но из-за большого количества пользователей один и тот же запрос повторяется очень часто. Зачастую для таких ситуаций целесообразно использовать кэш.

Кэш – это промежуточный буфер с быстрым доступом к нему, содержащий информацию, которая может быть запрошена с наибольшей вероятностью. Доступ к данным в кэше осуществляется быстрее, чем выборка исходных данных из более медленной памяти или удалённого источника, однако её объём существенно ограничен по сравнению с хранилищем исходных данных [2].

Memcached – это система, которая позволяет осуществлять кэширование результатов уже выполненых базой данных запросов в оперативной памяти.

Принцип работы системы Memcached

Memcached - это распределенная система кэширования памяти общего назначения. Memcached основана на клиент-серверной архитектуре. Memcached-сервер предоставляет услуги по кэшированию данных, а клиент предоставляет эти данные в форме пар ключ-значение. Важно то, что значение в данном случае – просто строка, то есть Memcached ничего не знает о формате используемых вами данных: он просто хранит строки, их интерпретация – задача клиента. Теперь, перед тем как в очередной раз потребуется сделать запрос к базе данных, сначала нужно обратиться к Memcached-серверу и узнать, не хранятся ли эти данные в памяти Memcached-сервера, и если хранятся, то получить их оттуда, нежели запрашивать их у базы данных. В результате такого двуступенчатого подхода (сначала заглянуть в кэш, и если данных там не окажется, заглянуть в базу данных) можно добиться прироста произоводительности и снижения нагрузки на базу данных. Действительно, извлечь данные, которые хранятся в оперативной памяти в виде пар ключ-значение, намного проще, чем анализировать SQL-запрос и, в худшем случае, загружать эти данные с диска.

Рисунок 1. Успешный запрос к Memcached-серверу

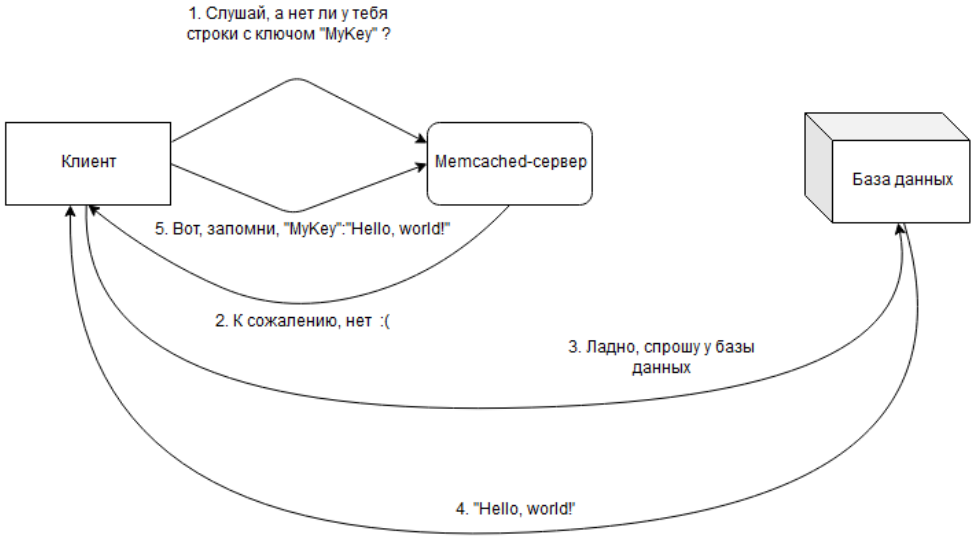


Рисунок 2. Неудачный запрос к Memcached-серверу

Важно также понимать, что Memcached – это инструмент программиста: именно программист решает где, когда, и как будут запрашиваться данные и сохраняться. Memcached – это не автоматическая прослойка, которая магическим образом ускоряет выполнение всех запросов к БД. Чтобы Memcached действительно ускорил работу какой-то программы, нужно

чтобы эта программа сама взаимодействовала с Memcached-сервером. Это отчётливо видно на Рис.2. Клиент, после того как получил отрицательный ответ на шаге 2, сам, явно, решил сохранить пару ключ-значение в памяти Memcached-сервера (шаг 5). Это говорит о том, что программа вообще-то не обязана использовать кэш. Если из тех или иных соображений часть данных в любом случае должна быть запрошена напрямую из базы данных, то, благодаря ненавязчивости Memcached, у программиста будет такая возможность. Это демонстрирует определенную гибкость при использовании данного инструмента. Тем не менее, гибкость такого рода присутствует не во всех аспектах Memcached.

Архитектура Memcached

В прошлом параграфе подразумивалось, что в системе присутствует всего один Memcached-сервер. К слову, это совершенно не обязан быть отдельный физический узел: это можеть быть отдельный процесс, запущенный на той же машине, что и веб-сервер и\или база данных. В данном случае лишь указывалось логическое расположения частей системы, а не физическое. В данном параграфе будет раскрыто слово «распределенная» в описании системы Memcached. «Распределенная» в данном контексте означает, что Memcached-сервер не обязан быть один – их может быть несколько. Самое приятное тут, что от этого особенно ничего не меняется. Дело состоит в том, что Memcached-серверы друг о друге ничего не знают. Это достигается за счет того, что клиент по ключу определяет к какому серверу стоит обратиться и каждый ключ обслуживается всего одним сервером. Информация не реплицируется, то есть серверы не хранят одну и ту же информацию, а, наоборот, каждый сервер ответственен за какой-то уникальный набор ключей. В таком подходе требуется всего лишь перечислить адреса всех серверов в клиенте, чтобы он мог обращаться к ним по необходимости.

Самое время вспомнить, что мы живем не в идеальном мире и компьютеры как и все хоть сколько-нибудь сложные механизмы иногда выходят из строя. Нужно задуматься, а что же будет, если такое случится с одним из (или если вам черная кошка перешла дорогу, то сразу с несколькими) Memcached-серверов. Несомненно, весь кэш, который хранился на этом сервере будет недоступен, следовательно, любые обращения к этим данным со стороны клиента будут неудачными. Получается, что обращения к кэшу, в данном случае, будут только торомзить систему. Дело усугубляется тем, что раз один из серверов недоступен, то алгоритм, по которому клиент выбирает сервер для ключа, следует бы подкорректировать, чтобы он не порождал запросы к недоступному серверу, а перераспределил нагрузку между оставшимися серверами. Вопрос теперь в том, а насколько хорошим будет это перераспределение: перераспределятся ли только ключи, за которые отвечал ныне недоступный сервер или сразу все ключи. Очевидно, нет смысла «передвигать» и так хорошо работающий кэш. Здесь определяющую роль играет выбор Memcached-клиента, в качестве которого может выступать, например, библиотека для языка C. Хорошие клиенты реализуют так называемый Consistent Hashing. Это алгоритм, который как раз устойчив к убавлению или добавлению серверов Memcached.

Остается теперь ответить на вопрос, что же будет делать Memcached, когда оперативная память, выделенная ему, будет заканчиваться. Если требуется сохранить какую-то информацию, но для неё нет свободной памяти, то Memcached будет искать «жертву», чтобы вместо нее записать новую информацию. Эту «жертву» Memcached ищет по алгоритму LRU (Least Recently Used). Суть этого алгоритма в том, чтобы найти участок памяти, который дольше всего не использовался (является наименее востребованным), и заменить его содержимое новыми данными. Данный алгоритм полагается на то, что клиент запрашвает какую-то часть данных с большей частотой, а остальную часть запрашивает гораздо реже. Таким образом, исключение менее востребованных данных не повлечет за собой рост частоты кэш-промахов (или, по крайней мере, будет её минимизировать).

Основные недостатки системы Memcached

Как уже было сказано, Memcached – это инструмент программиста, при этом довольно простой. Это с одной стороны достоинство, а с другой стороны, во многих ситуциях, недостаток. Memcached не предоставляет никаких типов данных, тем самым не позволяя получать или обновлять данные частично. Например, профиль пользователя может состоять из большого набора полей, и для обновления всего одного поля не нужно перезаписывать всю структуру данных целиком. С другой стороны, если использовать Memcached для статического веб-сайта, на котором информация обновляется крайне редко, то такой подход позволяет хранить HTML-код в виде одной строки, целиком. В данном случае Memcached позволяет избежать накладных расходов с хранением и анализом информации, связанной с типами данных.

Другой недостаток заключается в жестком ограничении на политику замещения памяти, а именно такая политика ровно одна – LRU, о которой было рассказано выше. LRU – это не единственно возможный подход по замещению памяти. Например, можно исключать случайно попавшиеся ключи, если у нас есть дополнительные предположения относительно нашей системы (равномерное распределение запросов по ключам), и сэкономить на накладных расходах, связанных с алгоритмом LRU. Даже сам алгоритм LRU допускает несколько разных подходов в реализации: можно использовать приближение к этому алгоритму, которое дает достойный результат, но требует меньше ресурсов.

Еще один недостаток связан с тем, что сохранять данные на диске Memcached без сторонних инструментов не позволяет. Хотя, учитывая то, что Memcached предназначен для одной конкретной цели – кэширования, то можно сказать, что он и не должен предоставлять лишних функций, которые напрямую не связаны с самим кэшированием.

Заключение

Memcached – это специализированный инструмент. Он выполняет свою задачу по эффективному кэшированию и делает это быстро и просто [3]. Главный его недостаток в контектсе одной задачи оказывается его главным достоинством в контексте другой задачи. Если задача достаточно хорошо оптимизируется тем набором услуг, который предлагает Memcached, то нет необходимости его заменять на более сложный инструмент. Ведь, более сложный инструмент не только предлагает больше опций, но также и больше багов. Известны случаи, когда лишний функционал библиотеки, о котором не подозревало большинство ее пользователей, оказывался мишенью для атак из-за некоторых уязвимостей. Напротив, если задача сама по себе предполагает использование тех опций, которыми не располагает Memcached, то стоит поискать альтернативы, например, довольно популярное на данный момент решение это Redis [4]. Redis – это намного больше, чем просто кэш в оперативной памяти, соответственно там есть и выбор из несколько политик замещения, множество типов данных и т.д [5].

Тем не менее, мир не ограничен только уже существующими системами. Иногда требуется реализовать свою собственную систему, а для этого очень полезно бывает взглянуть на опыт написания схожих систем.

Библиография

1. Memcached Github Wiki. – [Электронный ресурс] [дата обращения 02.03.2023], Доступно: https://github.com/memcached/memcached/wiki
2. Memcached Wikipedia. – [Электронный ресурс] [дата обращения 02.03.2023], Доступно: https://en.wikipedia.org/wiki/Memcached
3. VIJAY CHIDAMBARAM, DEEPAK RAMAMURTHI Performance Analysis of Memcached. [Электронный ресурс] [дата обращения 02.03.2023], Доступно: http://pages.cs.wisc.edu/~vijayc/papers/memcached.pdf
4. Using Redis as an LRU cache. – [Электронный ресурс] [дата обращения 02.03.2023], Доступно: https://redis.io/topics/lru-cache
5. Stackoverflow Memcached vs Redis. – [Электронный ресурс] [дата обращения 02.03.2023], Доступно: https://stackoverflow.com/questions/10558465/memcached-vs-redis

ТЕХНОЛОГИЯ ENTITY FRAMEWORK CORE

Алексей КОЗЬМА

Департамент Программной Инженерии и Автоматики, группа TI-195, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Алексей КОЗЬМА, e-mail: cozma.alexei@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Данная статья посвящена технологии Entity Framework Core. Из нее можно будет узнать, для чего была придумана данная технология, и какие проблемы в сфере Баз Данных она решает. В статье будут рассмотрены отличия EF Core от его предшественника, а также преимущества использования именно этой технологии, и в каких сферах и почему она наиболее широко применяется.*

Ключевые слова: *ORM-инструмент (Object-Relational Mapping), Code-First, Database-First, LINQ(Language-Integrated Query), .NET Core.*

Введение

Entity Framework Core – это технология компании Microsoft, пришедшая на смену технологии Entity Framework, представленная миру 27 июня 2016 года вместе с такими технологиями как NET Core и ASP.NET Core. Ключевыми свойствами новой серии Core стали открытый исходный код, уменьшенный размер, расширяемость и кросс-платформенность [1].

ORM-технология

Entity Framework Core – это ORM-инструмент, то есть он позволяет работать с базами данных на более высоком уровне абстракции, взаимодействуя с таблицами и находящимися в них данными, как с объектами.

ORM – это механизм, который позволяет создать связь между объектами, написанными при помощи объектно-ориентированных языков, таких как, например, Java или C#, и реляционной базой данных. Благодаря этому механизму у разработчиков имеется возможность взаимодействовать с объектами, не задумываясь, где и каким образом хранятся данные. Такой уровень абстракции позволяет не изменять приложение каждый раз, когда меняется способ хранения данных и существенно сократить время при разработке программного обеспечения [2, 3].

С какими СУБД можно использовать EF Core

Entity Framework Core поддерживает любую СУБД(систему управления базами данных), для которой существует нужный провайдер. По умолчанию Microsoft предоставляет встроенные провайдеры для работы с MS SQL Server, PostgreSQL, SQLite, однако можно найти провайдеры также для MySQL, OracleDB и т. д..

Какие методы разработки поддерживает EF Core

Entity Framework Core поддерживает 2 метода разработки:

1. Code-First(основной способ)
2. Database-First

В первом случае EF Core создает базу данных с отношениями, используя миграции, основывающиеся на конфигурации предоставленной в классах.

Во втором случае EF Core, исходя из базы данных, создает необходимые классы, однако данный метод не является предпочтительным, так как EF Core не предоставляет удобного дизайнера для создания базы данных.

Различия между EF Core и EF 6.x

В первую очередь стоит отметить, что новый EF Core стал более компактным и расширяемым по сравнению со своим предшественником и получил возможность применяться на любых системах (т.н. кросс-платформенность). Однако это не единственные их различия. Кроме этого можно выделить:

- поддержку баз данных, хранящихся в оперативной памяти (In-memory databases), что значительно облегчает тестирование приложений, за счет того, что обращения к ней происходят намного быстрее и вероятность нечаянно допустить потерю каких-нибудь данных из рабочей БД сводится к нулю;
- удаление EDM (Entity Data Model) дизайнера, вследствие чего EF Core не поддерживает подход Model-First в отличие от EF;
- поддержка IoC (Inversion-of-Control);
- поддержка unique-ограничения;
- поддержка альтернативных ключей [4].

Преимущества работы с EF Core:

Entity Framework Core обладает рядом преимуществ для разработки программного обеспечения. Среди них:

- применение подхода Code-First;
- использование одного языка программирования для разработки приложения и для взаимодействия с БД;
- возможность получения информации из БД с помощью технологии LINQ (Language-Integrated Query);
- кросс-платформенность;
- возможность переключаться между различными базами данных, внося минимальные изменения в само приложение;
- удобство в тестировании взаимодействия приложения с БД [5].

Области применения EF Core

Благодаря своим преимуществам Entity Framework Core получил широкое распространение в различных типах приложений (Рис. 1).

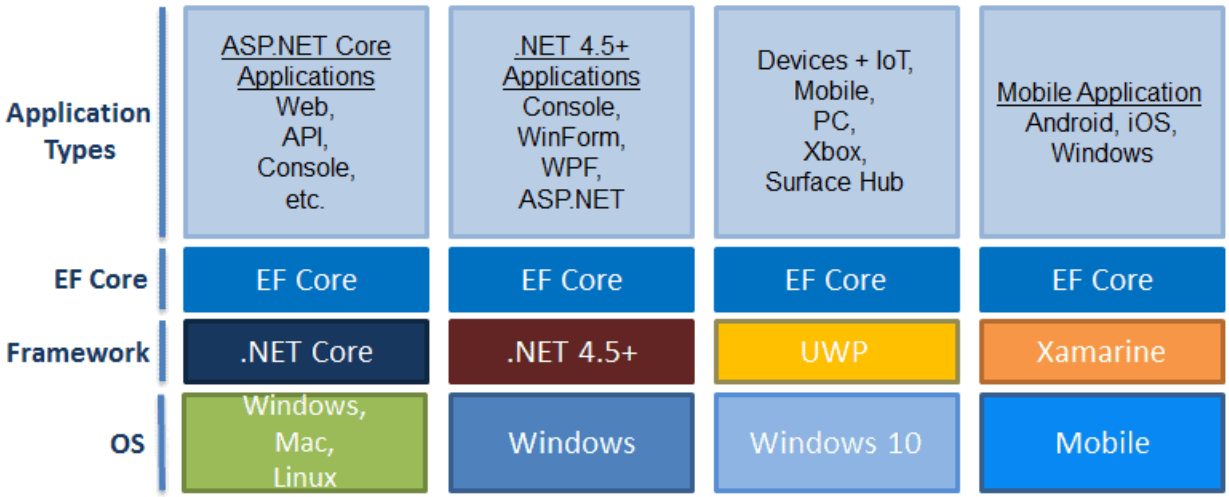


Рисунок 1. Области применения EF Core

Миграции в EF Core

Благодаря миграциям Entity Framework Core синхронизирует модели, используемые в коде, с базой данных. Таким образом при изменении класса-модели изменится и соответствующий ему объект в БД. Для этого EF Core предоставляет ряд специальных команд для работы с миграциями, исполняемых в NuGet Package Manager Console либо в CLI(Command Line Interface):

- Add-migration <migration name> — создает снэпшот миграции
- Remove-migration — удаляет последний снэпшот миграции
- Update-database — обновляет базу данных, используя последний снэпшот
- Script-migration – генерирует SQL код, используя все снэпшоты миграции

Добавление миграции. После создания новой миграции командой Add-migration в папке проекта появятся следующие классы(Рис. 2):

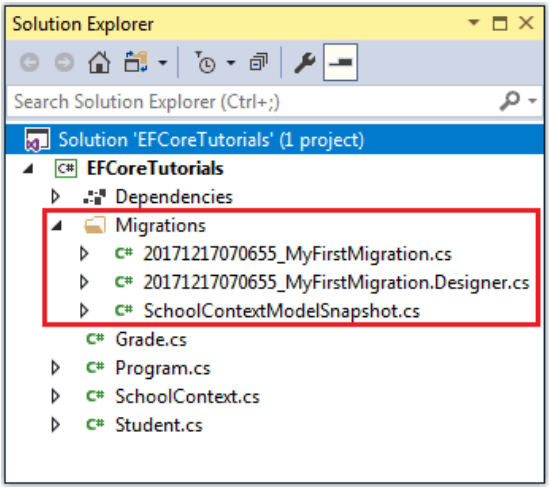


Рисунок 2. Классы, созданные добавлением миграции

1. **<timestamp>_<migration name>.cs — главный файл миграции, в котором содержится информация о добавляемых и удаляемых из БД объектов;**
2. **<timestamp>_<migration name>.Designer.cs — файл с метаданными для работы EF Core;**
3. **<contextclassname>ModelSnapshot.cs — снэпшот действующей БД, использующийся для определения изменений.**

Удаление миграции. При использовании команды Remove-migration удаляется последний сделанный снэпшот, если он не был применен к базе данных. Попытка удалить снэпшот после обновления базы данных на его основе приведет к ошибке Entity Framework Core.

Обновление базы данных. При использовании команды Update-database будет использован последний снэпшот для создания или обновления базы данных. Кроме этого Entity Framework Core добавит запись в таблицу _EFMigrationHistory с информацией о миграции нужной для того, чтобы прослеживать изменения БД и совершить откат к предыдущим миграциям при необходимости. Важной особенностью команды Update-database является то, что, указав после самой команды название миграции, пользователь может вернуться к необходимой миграции.

Получения SQL-скрипта. При помощи команды script-migration можно получить все изменения в виде SQL-кода в случае, когда необходимо выполнить миграцию вручную [6].

Заключение

В заключении можно сказать, что технология Entity Framework Core является прекрасным решением для C#-разработчиков, когда появляется необходимость создать взаимодействие приложения с базой данных. Принимая во внимание частоту использования баз данных в различных проектах и область применения Entity Framework Core, можно утверждать, что данная технология будет полезна к изучению любому программисту, работающему на языке C#.

Библиография

1. Введение в Entity Framework Core. [online]. [просмотрен 23.12.2021]. Доступно: https://metanit.com/sharp/entityframeworkcore/1.1.php
2. object-relational mapping (ORM). [online]. [просмотрен 23.12.2021]. Доступно: https://www.theserverside.com/definition/object-relational-mapping-ORM
3. Object-Relational Mapping (ORM). [online]. [просмотрен 23.12.2021]. Доступно: https://www.techopedia.com/definition/24200/object-relational-mapping--orm
4. Compare EF Core & EF6. [online]. [просмотрен 23.12.2021]. Доступно: https://docs.microsoft.com/en-us/ef/efcore-and-ef6/
5. What Is Entity Framework And How Entity Framework Core Is Different. [online]. [просмотрен 23.12.2021]. Доступно: https://www.c-sharpcorner.com/article/what-is-entity-framework-how-entity-framework-core-is-different/
6. Entity Framework Core. [online]. [просмотрен 23.12.2021]. Доступно: https://www.entityframeworktutorial.net/efcore/entity-framework-core.aspx

ОПТИМИЗАЦИЯ ЗАПРОСОВ SQL (MySQL)

Эдуард ЦУРКАН

Департамент Программная Инженерия и Автоматика, группа TI-195, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Эдуард ЦУРКАН, e-mail: eduard.turcan@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Статья посвящена оптимизации запросов sql (MySQL). Целью является дать ответ на вопрос - как увеличить производительность системы, которая использует базу данных с (медленными) не оптимизированными запросами. Как, благодаря простым правилам уменьшить время ожидания и ускорить работу всей системы, чтобы работа с ней приносила удовольствие конечным пользователям.*

Ключевые слова: *оптимизация, индексы, SELECT, WHERE, LIKE, LIMIT, PRIMARY KEY, снижение нагрузки, DISTINCT.*

Введение

Оптимизация – это изменение системы с целью повышения ее быстродействия. Она состоит из 3-х типов: оптимизация запросов, оптимизация структуры и оптимизация сервера. В данной статья будет рассмотрен 1-ый тип оптимизаций БД. К примеру, если рассматривать обычный сайт, у которого посещаемость пользователей мала и который не так нагружен информацией, то не будет понятно, оптимизированы запросы к БД или нет. А вот для популярных, больший рабочих сайтов, серверов под большой нагрузкой (большая посещаемость пользователей), то разница между оптимизированным и неоптимизированным SQL является огромной, и во время выполнения они могут значительно влиять на поведение и надежность сервисов (медленно обрабатываются, большая задержка, видная пользователям).

1. Индексы

Индексы используются в MySQL для поиска кортежей с указанными значениями атрибутов, в паре с WHERE, где в качестве параметра задается условие поиска. Без индексов, MySQL начинает проходится по отношению с первого кортежа, в поиске указанных ранее значений. Чем больше отношение, тем больше понадобится времени на работу запроса [1]. Если у отношения, атрибуты имеют индексы, то они будут использоваться в запросе для поиска, тем самым MySQL быстрее найдет нужный кортеж, без просмотра всего отношения.

2. SELECT

2.1 SELECT* - вывод всех атрибутов из отношения. Как известно, чем больше данных используется в запросе для считывается из отношения, тем медленнее данный запрос будет выполнятся. SELECT* - увеличивает время работы с хранилищем данных [2]. Использования этого метода может привести к большой задержке при передаче данных по сети, если веб-приложение использует БД с такими запросами. Самое простое решение этой проблемы – это прописать вручную, какие именно столбцы из запроса нужны.

2.2 Оптимизация WHERE

Используя несколько условий в WHERE, разделенные ключевым словом AND, необходимо расставить их в правильном порядке. Первыми условиями должны идти те, которые с большей вероятностью могут вернуть FALSE. Это делается для того, чтобы запрос меньше обрабатывался в случае ошибки, как следствие скорость выполнения увеличивается. Что касается той же ситуации, но с использованием ключевого слова OR, то тут немного иначе. Условия должны располагаться в порядке уменьшения вероятности истинности

данного условия. Это означает, что чем быстрее одно из условий вернет TRUE, тем меньше условий еще будет обработано, тем самым увеличься производительность запроса.

Если необходим запрос, в котором проверяется несколько конкретных условий, по типу: нужна информация о пассажирах 1 и 3-ого авиарейсов, то лучше использовать операцию IN, взамен OR. В данном случае скорость выполнения запроса возрастет, так как операция IN изначально работает быстрее, чем операция OR.

Прошлое замечание уместно и для следующих операций. Если нужно узнать, существует ли кортеж, который соответствует каким-то условиям, то используйте EXISTS вместо COUNT> 0 в своих подзапросах. Это непременно уменьшит время выполнения всего запроса. Операцию LIKE, также, лучше использовать реже, взамен предлагается использовать поиск, основанный на FULL-TEXT индексах. Аргумент – прост, банально быстрее.

2.3 Группировка с помощью SELECT

Чтобы оптимизировать операцию группировки следует использовать как можно меньше атрибутов для группировки. Лучше использовать оператор WHERE вместо HAVING, потому что в этом случае уменьшается количество строк для группировки на ранней стадии [3]. Использование команды DISTINCT ускорит работу группировки, если не используются функций (COUNT (), MIN (), MAX и т.д.). Для более быстрого выполнения запросов с использованием DISTINCT стоит добавить команду LIMIT. Эта делается из-за того, что команда DISTINCT требует повышенного времени обработки.

2.4 Сортировка в SELECT

Одной из самых ресурсоемких задач в MY SQL является сортировка строк(кортежей). Чтобы снизить нагрузку этой операции необходимо, при инициализации атрибутов устанавливать точный размер, столько сколько нужно, и не выделять лишние байты про запас. Как пример, для атрибута “фамилия” не выделять больше места (VARCHAR (100)), а выделить минимально возможный размер (VARCHAR (30)).

3. Создание атрибута ID для каждого отношения

Для нормальной работы БД, в каждой таблице желательно иметь поле ID, которое будет PRIMARY KEY или одним из атрибутов составного ключа с типом INT [4]. Так же неплохо сделать это поле - UNSIGNED, потому что отрицательные значения у него никогда не будут. К примеру, если существует отношение, в котором есть уникальное поле username (имя пользователя), это еще не значит, что оно может являться ключем. Использование поля с типом VARCHAR, как PRIMARY KEY - очень медлительно. Читабельнее будет, если каждый пользователь, помимо имени будет обладать уникальным номером (ID), так как в этом отношении могут быть кортежи с одинаковыми значениями в поле username(Тески).

4. Использование NOT NULL

Если есть необходимость в использовании NULL, то его можно без особых проблем для запроса использовать, но, если надобности нет, то лучше использовать NOT NULL. В сравнении с NOT NULL, NULL занимает больше места в память и усложняет работу запроса, использующего этот атрибут. В подтверждении этого - Документация MySQL: «Столбцы NULL занимают больше места в записи, из-за необходимости отмечать, что это NULL значение [5]. К примеру, для таблиц MyISAM (одна из основных систем хранения данных в СУБД MySQL.), каждое поле с NULL занимает 1 дополнительный бит, который округляется до ближайшего байта».

5. Использование LIMIT 1, при поиске одного-единственного кортежа

Создавая запрос, где результатом будет один-единственный кортеж желательно использовать LIMIT 1. Это могут быть примеры, по типу: необходимо найти один уникальный кортеж или же, просто проверить существование записей, которые удовлетворяют запросу WHERE.

Во всех этих случаях, использование LIMIT 1 в запросе будет ускорять его работу. Следовательно, база данных остановит выборку кортежей, после нахождения первого соответствия, вместо того, чтобы выбирать всю таблицу или индекс какого-то атрибута.

6. Использование статичных отношений.

Статичным отношением, можно назвать таблицу, которая имеет атрибуты фиксированного размера, их еще называют «таблицы фиксированного размера». VARCHAR (50) и CHAR (50) с первого взгляда кажутся похожими, но дело в том, что запись (CHAR (50)) будет занимать 50 байтов, вне зависимости от ее реального содержания, а (VARCHAR (50)) не обязательно все 50 байтов (это замечание очень помогает при восстановление данных, при их потере). К примеру, VARCHAR, TEXT, BLOB являются типами данных для которых размер не известен заранее и может меняться (не фиксированный размер). Если добавить в отношение такой атрибут, то это отношение уже не будет статичным, соответственно, оно будет обрабатываться MySQL по-другому.

Использование таблиц с фиксированным размером увеличит эффективность базы данных. Зная размер кортежа, MySQL сможет очень быстро вычислить его позицию в базе данных. Если же, размер кортежа окажется не фиксирован, тогда MySQL будет обрабатывать это действие по-другому, через индексы. К преимуществам использования статичных отношений еще можно отнести: простое кэширование данных, и соответственно восстановление после различных ошибок (“падение БД”).

7. Заключение

В данной статье были затронуты лишь некоторые возможные варианты оптимизации запросов. Если придерживаться данным советам, то ваши SQL запросы будут выполняться быстрее, тем самым вы увеличите производительность своей БД [6]. Все советы, данные в этой статье были основаны исходя из синтаксиса языка запросов SQL и при помощи официальной документации MySQL, которая настоятельно рекомендует придерживаться всем вышеперечисленным стандартам работы с запросами в БД MySQL.

Библиография

1. Оптимизация запросов в MySQL – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: http://www.php.su/articles/?cat=phpdb&page=005
2. Оптимизация MySQL запросов – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://habr.com/ru/post/41968/
3. Оптимизация SQL – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://falcon.web-automation.ru/list/sqlserver/opt
4. Оптимизация MySQL запросов – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://handyhost.ru/help/poleznyie-stati/optimizacziya-mysql-zaprosov.html
5. MySQL 8.0 Reference Manual / Optimization – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://dev.mysql.com/doc/refman/8.0/en/optimization.html
6. MySQL Documentation – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://dev.mysql.com/doc/

РАСПРЕДЕЛЕННАЯ СУБД APACHE CASSANDRA

Яна ЗАБОЛОТНЫЙ

Департамент Программной Инженерии и Автоматики, группа TI-195, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корресспондент: Яна ЗАБОЛОТНЫЙ, e-mail: zabolotnii.iana@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Данная работа посвящена СУБД Apache Cassandra. Она освящает историю этой популярной базы данных, рассказывает о ее преимуществах, недостатках и принципе работы. Статья даёт ответы на множество вопросов, которыми задаются без исключения все пользователи этой базы данных.*

Ключевые слова*: Apache Cassandra, архитектура, СУБД, база данных*

Введение

Apache Cassandra - это распределенная и децентрализованная СУБД, используемая для управления огромными объемами данных, разбросанных по всему миру. Cassandra обеспечивает высокодоступный сервис без единой точки отказа. Проект был разработан в 2008 году компанией Facebook на языке Java и передан в фонд Apache Software Foundation в 2009 году. На рисунке 1 изображен логотип Apache Cassandra.

Эта СУБД классифицируется как гибридное решение NoSQL, потому что она сочетает в себе модель сохранения данных, основанную на типе столбцов, с концепцией «ключ-значение» (key-value). Apache Cassandra используется многими крупными компаниями, такими как Cloudkick, Reddit, Digg, Huawei, Netflix, Apple, Instagram, GitHub, Twitter, Spotify [1].

Рисунок 1 – Логотип Apache Cassandra.

1. Модель данных Apache Cassandra

Модель данных Apache Cassandra (рис.2) состоит из таких элементов:

- Столбец (column) – представляет собой ячейку с данными, состоящую из 3 частей: имени (column name), метки времени (timestamp) и значения (value). Метка времени связана с каждым значением и является 64 - битным числом, которое задает пользователь.
- Семейство столбцов (column family) – набор именованных строк.
- Строка (row) – набор именованных столбцов.
- Пространство ключей (keyspace) – группа, состоящая из семейств столбцов.

Пространство ключей логически группирует семейства столбцов [1].

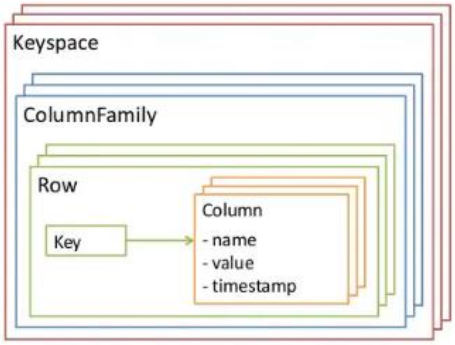


Рисунок 2 – Модель данных Apache Cassandra.

2. Архитектура Apache Cassandra

Apache Cassandra состоит из нескольких узлов, по которым размещаются данные. Данная СУБД не поддерживает концепцию master/slave, когда один из серверов является сервером управления для других кластерных компонентов, что отличает ее от других решений, используемых для больших данных.

Для того, чтобы распределить элементы данных между узлами Cassandra использует поочередное хеширование, применяя алгоритм хеширования для вычисления хеш-значений ключей каждого элемента данных. Спектр всех вероятных значений хеш-функции, то есть пространство ключей, распределяется между узлами кластера таким образом, что каждому элементу данных должен присваиваться узел, который отвечает за сохранение и регулирование этого элемента данных [1].

Таким образом, тот факт, что в Cassandra отсутствует центральный узел лишает ее недостатка, который присущ многим системам master / slave, в которых при сбое основного сервера кластер полностью выходит из строя. В кластере Cassandra все узлы эквивалентны друг другу, и, если один из них выйдет из строя, некоторые из оставшихся возьмут на себя его функции. Благодаря такой функции Apache Cassandra прекрасно подходит для распределенных систем с большим доступом, расположенных в разных центрах обработки данных. Но даже при всех достоинствах этой гибкой адаптивной архитектуры она определяет особенности операций чтения и записи, а также возлагает ограничения на использование СУБД в реальных проектах. На рисунке 3 изображена архитектура кластера Apache Cassandra[2].

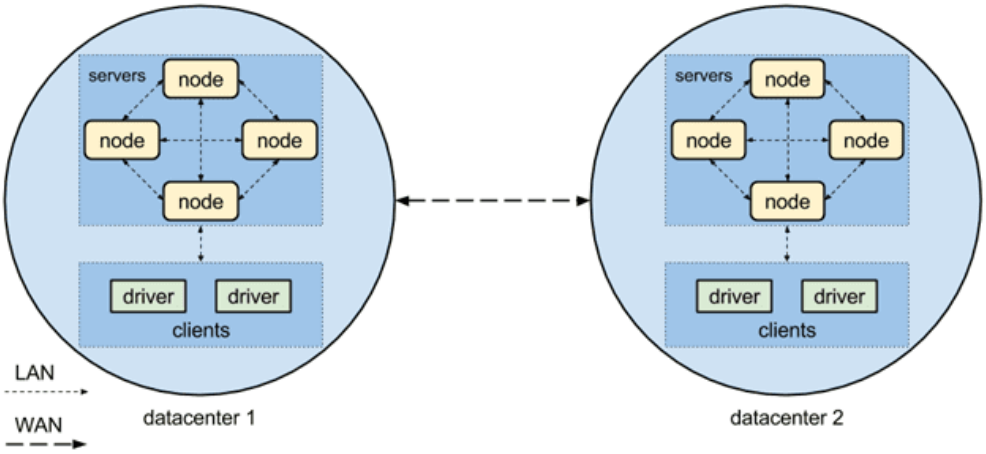


Рисунок 3 – Кластерная архитектура Apache Cassandra.

3. Особенности и преимущества Apache Cassandra

Apache Cassandra стала популярной из-за своих технической исключительности. Вот некоторые особенности Apache Cassandra:

- Простое распределение данных - гарантирует гибкость для распределения данных там, где это нужно, путем репликации данных в нескольких центрах обработки данных [2].
- Гибкое хранение данных - поддерживает все возможные форматы данных, в том числе структурированные, полуструктурированные и неструктурированные. Cassandra вносит динамические изменения в структуры данных [2].
- Эластичная масштабируемость - обладает высокой масштабируемостью и позволяет добавлять дополнительное оборудование для поддержки большого количества клиентов и данных [3].
- Быстрая запись - была разработана для работы на недорогом типовом оборудовании. Выполняет невообразимо быструю запись и может сохранять сотни терабайт данных без потерь [3].
- Always on Architecture - Cassandra не имеет единой точки отказа и постоянно доступна для критически важных бизнес-приложений, которые не могут позволить себе выйти из строя.
- Быстрая линейно-масштабируемая производительность - линейно масштабируемая, поэтому она повышает пропускную способность, когда увеличивается численность узлов в кластере. Cassandra поддерживает быстрое время отклика.
- Поддержка транзакций и репликации (рис.4) - поддерживает такие свойства транзакций, как дробность, согласованность, изоляция и долговечность.

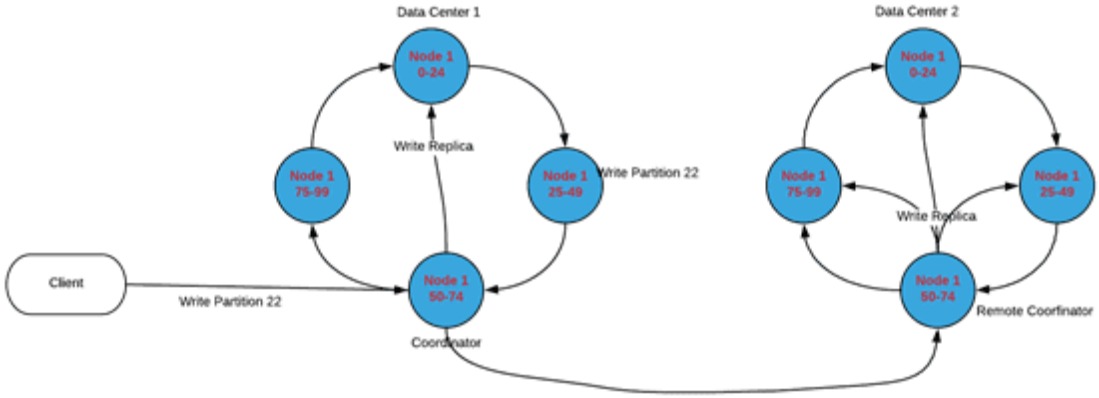


Рисунок 4 – Пример репликации данных в Apache Cassandra

4. Недостатки Apache Cassandra

Ключевые недостатки Cassandra, как и ее преимущества, связаны с особенностями архитектуры этой распределенной СУБД. В частности, наиболее значительными недостатками данной системы, основными с точки зрения применения в проектах больших данных, являются такие:

- Особенности внутреннего языка запросов – язык CQL существенно отличается от SQL, несмотря на небольшую схожесть. В основном, операции INSERT и UPDATE - это одно и то же. Кроме того, Cassandra не использует операции объединения. Объединение двух семейств столбцов потребует программного извлечения и объединения данных, что является дорогостоящим и трудозатратным для больших наборов данных[4].
- Высокие дополнительные расходы при работе с большими объемами данных - из-за подхода к проектированию данных, на основе SELECT, требуется очень сильная денормализация, что подразумевает фактическое создание отдельной таблицы для каждого запроса [4].

- Требования к уникальности ключа – каждый ключ обязательно должен быть уникальным. Данные перезаписываются, если один и тот же ключ используется дважды. Хотя это решается путем добавления рандомного значения или метки времени к ключу, а также благодаря составному ключу (рис.3).
- Сложность с поисковой операцией, которая не предусмотрена в архитектуре Cassandra, и поисковые системы построены поверх нее путем сортировки данных. Cassandra поддерживает вторичные индексы с довольно узким функционалом, который создает система[4].

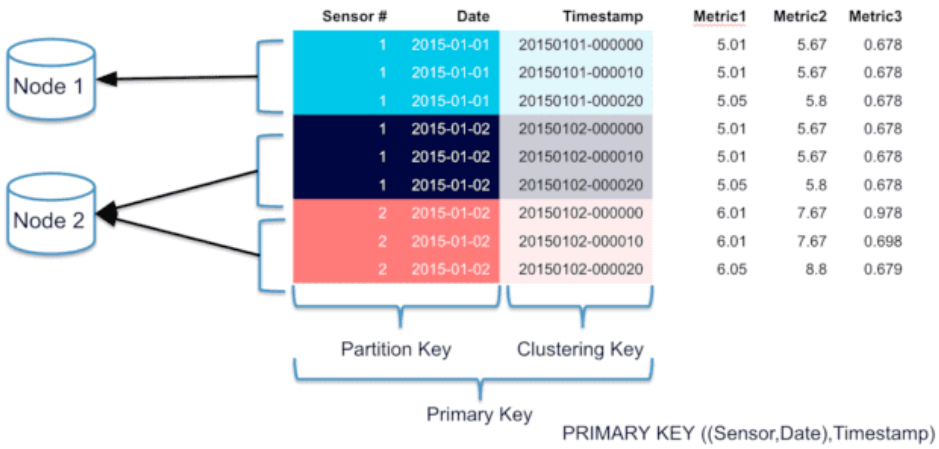


Рисунок 4 – Пример формирования ключей в Apache Cassandra

Заключение

Apache Cassandra - это СУБД, которой доверяют тысячи компаний благодаря своей масштабируемости и высокой доступности без вреда для производительности. Линейная масштабируемость и проверенная отказоустойчивость на стандартном оборудовании или облачной инфраструктуре делают ее идеальной платформой для критически важных данных. Как и у любой другой СУБД, у Apache Cassandra есть свои преимущества и недостатки, именно поэтому так важно изучить каждые из них.

Библиография

1. Wikipedia. Apache Cassandra. – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://ru.wikipedia.org/wiki/Apache_Cassandra
2. Что такое Apache Cassandra: архитектура, история и принципы. – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://www.bigdataschool.ru/wiki/cassandra
3. Плюсы и минусы Apache Cassandra. – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://www.bigdataschool.ru/blog/cassandra-key-features.html
4. Apache Cassandra. СУБД с распределенной системой хранения данных. – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://blog.bissquit.com/dbms/apache-cassandra-subd-s-raspredelennoj-sistemoj-hr/

INTERACȚIUNEA DINTRE ENTITY FRAMEWORK ȘI BAZE DE DATE

Victor COVRIG

Departamentul Ingineria Software și Automatică, grupa TI-192, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Victor COVRIG, e-mail: covrig.victor@isa.utm.md

Conducător științific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat: *În aceasta lucrare se definește noțiunea de ORM și Entity Framework și se enumeră modalitățile de interacțiune dintre bazele de date si Entity Framework, cu descrierea amanunțită a fiecăruia din ele, dar și se aduce exemple de utilizare în dependență de necesitățile dezvoltatorului. Aceste metode de interacțiune sunt: Code First, Database First și Model First.*

Cuvinte-cheie: *ORM, Database First, Model First, Code First, Entity Data Model*

Introducere

Object Relational Mapper-ul reprezintă un instrument ce realizeaza interactiunea cu baza de date prin intermediul paradigmei obiect-orientată. Conceptul de ORM este implementat utilizand librării (framework-uri) în limbaje ce suportă paradigma obiect-orientată. Exemple de librării ce implementează ORM-ul sunt: Entity Framework (limbajul C#) Sequelize (limbajul JavaScript), Laravel Eloquent (limbajul PHP). Unul din cele mai cunoscute și des utilizate librării ORM ce utilizează tehnologia .Net este Entity Framework [1].

Entiy Framework permite de a abstractiza baza de date și de a lucra cu datele independent de tipul de păstrare lor. În cadrul lucrului direct cu o bază de date se operează cu tabele, indecși, chei primare și secundare. La nivelul de concept ce propune Entity Framework, lucrul cu bazele de date se efectuează prin intermediul obiectelor. Concepția centrală a Entity Framework este entitatea. Entitatea reprezintă un set de date, ce sunt asociate unui anumit obiect [2].

Un alt concept de bază este Entity Data Model (EDM). Acest model suprapune clasele entităților cu tabelele reale. Exista 3 abordări ale EDM: Database First, Model First și Code First.

1. Code First

Code First este modalitatea în care EDM generează baza de date utilizând codul claselor .Net și utilizează Entity Framework pentru a "migra" baza de date atunci când atributele sunt modificate [3]. Această modalitate este potrivită atunci când nu există încă o bază de date a aplicației. Printre beneficiile abordării Code First se numără:

- Codul corespunde întodeauna cu schema tabelelor, astfel necorespondența în tipurile atributelor, definiția tabelelor sunt minime;
- Oricine poate genera tabele dacă cunoaște C#, chiar dacă nu are prea multe cunoștințe in C# (pe de altă parte, este și un dezavantaj, deoarece pot fi stabilite incorect contrângerile de cheie, de tuplu etc.);
- Este o modalitate universală, întrucât lucrează cu diferite sisteme de gestiune a bazelor de date, precum PostgresSQL, MySQL, SQL Server.

Urmatoarea diagramă (Fig. 1) afișează modul în care funcționeaza Code First.

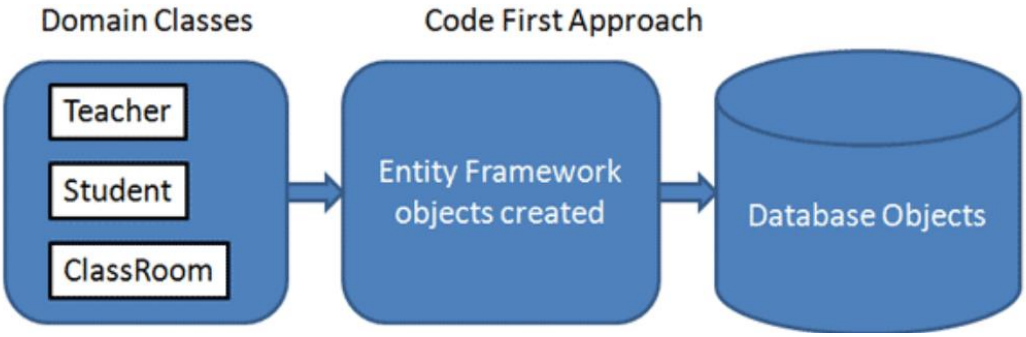


Figura 1. Modul de funcționare Code First

Pe partea de aplicație sunt prezente clasele de domeniu – clasele ce definesc atributele și contrângerile de atribute și de domeniu al fiecărui tabel al bazei de date. Aceste clase interacționează cu clasa de context (DbContext) ce reprezintă punctul principal de interacțiune al bazei de date cu clasele de domeniu. Prin intermediul mecanismului Code First, inclus in cadrul Entity Framework este definit codul de adaugare a tuplurilor noi în tabel și salvarea schimbărilor. După executarea acestui cod este creată baza de date împreună cu toate tabelele definite în cod.

2. Database First

Database First este prima abordare a EDM ce a aparut în Entity Framework. Această abordare este potrivită pentru cazul în care baza de date este deja creată. Acesta abordare prezintă avantaje, comparativ cu Code First în cazul în care avem deja un număr mare de tabele (peste 50):

- Evitarea introducerii manuale a fiecarei definiții a tabelelor, ca în cazul Code First;
- Nu putem utiliza migrațiile deoarece baza de date deja există, iar schimbarile în atribute trebuie gestionate în cod SQL.

Nu este recomandat de a schimba clasele pe care le generează Entity Framework, deoarece acestea vor fi regenerate dupa ce este efectuată urmatoarea rulare a comenzii. Dacă pentru crearea claselor este utilizat IDE-ul Visual Studio, clasele sunt generate pe baza unui fișier edmx, care este afisat cu ajutorul unui „designer”, care este creat odata cu accesarea obțiunii: „Generate from database”.

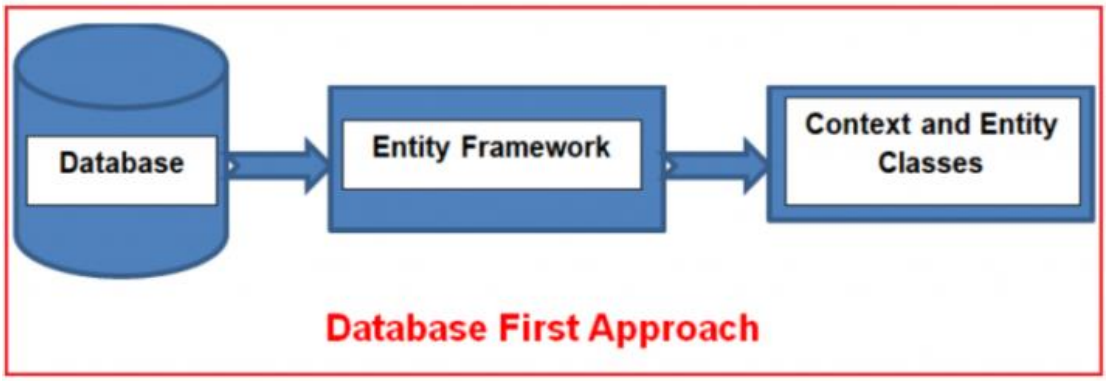


Figura 2. Schema de funcționare Database First

3. Model First

Model First poate fi comparat cu Code First, deoarece ambele abordări contribuie la crearea bazei de date utilizând modele. Însa, Model First nu doar genereaza baza de date, dar și codul claselor de domeniu și contextul bazei de date. Code First presupune crearea manuală a claselor, însă Model First genereaza în mod automat toate datele necsare, chiar și adaugă suportul Entity Framework în aplicație, fără a utiliza configurarea manuală cu ajutorul pachetelor NuGet, așa cum este necesar la Code First [5].

Model First utilizează designer-ul EF, ce prezintă modele din fișiere edmx. Acest deginer este prezent in Visual Studio pentru a genera baza de date si clasele. In figura ce urmeaza este prezentat un exemplu de modele contruite in interfața grafică (Fig. 3). Trebuie menționat că constrângerile de domeniu și de atribut, dar și relațiile între tabele sunt setate tot în interfața grafică.

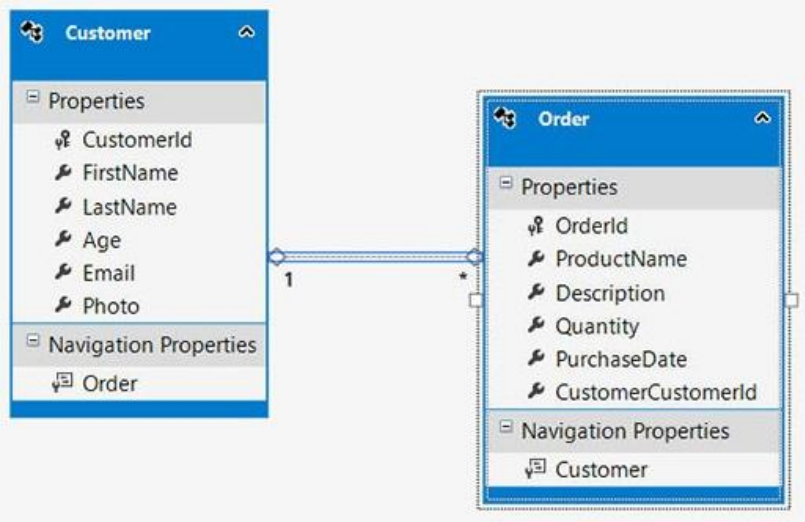


Figura 3. Modele create in EF designer

Următoarea diagramă prezintă schema de funcționare a Model First în generarea claselor de domeniu si bazei de date.

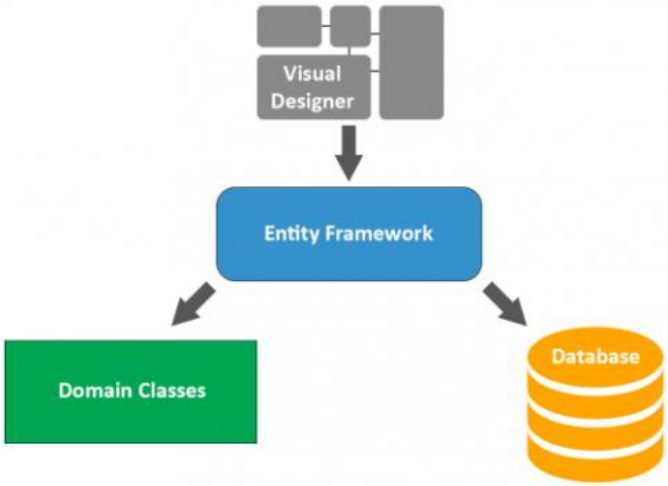


Figura 4. Schema de funcționare Model First

Concluzii

În lucrare au fost expuse toate cele 3 abordări ale Entity Data Model: Code First, Database First și Model First. Toate cele 3 abordări au avantajele și dezavantajele proprii și sunt utilizate în pentru situații diferite. Astfel, Code First este o soluție atunci când nu exista o bază de date, dar există dezvoltatori ce pot crea codul necesar generării bazei de date. Database Model este utilizat când avem deja baza de date creată și este necesar de a genera clasele de domeniu și clasele de context. Model First este o soluție mai universala ce se utilizează pentru a genera în mod automat baza de date și clasele de domeniu și context având la dispoziție doar modelul conceptual al tabelelor bazei de date (atributele tabelelor, contrângeri, relații între tabele).

Bibliografie

1. What is an ORM and Why You Should Use it. [online]. [accesat18.12.2021]. Disponibil: https://blog.bitsrc.io/what-is-an-orm-and-why-you-should-use-it-b2b6f75f5e2a
2. Введение в Entity Framework. [online]. [accesat18.12.2021]. Disponibil: https://metanit.com/sharp/entityframework/1.1.php
3. Code First vs Database First vs Model First – EntityFramework Approaches Explained. [online]. [accesat18.12.2021]. Disponibil: https://dotnetcoretutorials.com/2021/06/26/code-first-vs-database-first-vs-model-first-entityframework-approaches-explained/
4. Reverse Engineering [online]. [accesat 18.12.2021]. Disponibil: https://docs.microsoft.com/en-us/ef/core/managing-schemas/scaffolding?tabs=dotnet-core-cli
5. Использование Model-First. [online]. [accesat 18.12.2021]. Disponibil: https://professorweb.ru/my/entity-framework/6/level1/1_5.php

ТРАДИЦИОННАЯ И ОБЛАЧНАЯ АРХИТЕКТУРА ХРАНИЛИЩ ДАННЫХ

Лучия НИГАЙ

Департамент Программной Инженерии и Автоматики, группа TI-196, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический университет Молдовы, Кишинев. Молдова

Автор Корреспондент: Лучия НИГАЙ, e-mail: nigai.lucia@isa.utm.md

Научный руководитель: Саранчук Дориан, DISA, FCIM, UTM

Аннотация: *В данной статье рассматриваются традиционная архитектура хранилищ данных и компоненты из которых она состоит, а также облачная архитектура хранилищ данных, ее компоненты и функционирование на примере Amazon Redshift.*

Ключевые слова: *хранилище данных, традиционная архитектура, облачные хранилища, Amazon Redshift*

Введение

Хранилище данных – это разновидность системы хранения и управления данными, предоставляющая инструменты онлайн-аналитической обработки (OLAP), и позволяющая проводить интерактивный анализ данных в многомерном пространстве, который приводит к обобщению данных и их интеллектуальному анализу.

Хранилища данных содержат в себе исторические и коммутативные данные из различных источников, поэтому их архитектура довольно непроста. В этой статье мы рассмотрим традиционную и облачную архитектуру хранилищ данных, широко используемые в различных областях, таких как обработка транзакций, аналитическая обработка, прогнозная аналитика и многое другое.

Традиционная архитектура хранилищ данных

Традиционные хранилища данных имеют три подхода к построению: виртуальное хранилище данных (одноуровневая архитектура), киоски или витрины данных (двухуровневая архитектура) и корпоративные хранилища данных (трехуровневая архитектура). Далее рассмотрим каждый из подходов [1]:

Виртуальное хранилище данных, его целью является минимизация объема данных за счет удаления их избыточности. Обычно оно создается путем создания ряда представлений к базе данных или использованием специальных средств доступа.

Киоски или витрины данных разделяют физические источники и хранилище данных, путем построения витрин данных без создания центрального хранилища. Они способны хранить лишь ограниченное количество данных, являются нерасширяемыми, а также поддерживают лишь небольшое количество пользователей. Выделяют два типа киосков данных: независимый (данные поступают напрямую из оперативных источников данных) и зависимый (информация является подмножеством основного хранилища) [1].

Корпоративное хранилище данных – наиболее используемый вид хранилищ данных, состоящий из трех уровней (Рисунок 1):

- Сервер базы данных, который используется для извлечения данных из транзакционных систем и других источников данных.
- Сервер OLAP, представляющий собой абстрактное структурированное представление базы данных, подходящее для анализа и создания сложных запросов.

– Клиентский интерфейс, а именно инструменты API, которые используются для извлечения и анализа данных из хранилища.

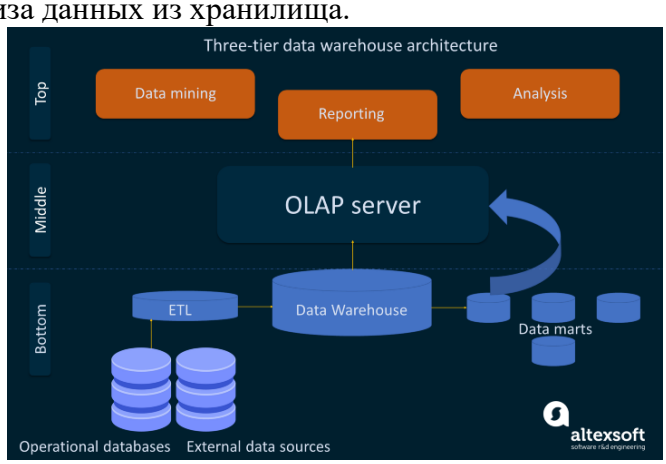


Рисунок 1. Трехуровневая архитектура

В большинстве своем хранилища данных состоят из 5 основных компонентов[2]:

– Программное обеспечение промежуточного слоя, к которому относится сетевые и коммуникационные протоколы, драйверы и т.д.
– Транзакционные базы данных и внешние источники информации, такие как отчеты.
– Уровень доступа к данным, обеспечивающий взаимодействие с пользователем и загрузку данных из транзакционных систем.

Загрузка и обработка

Существует два подхода: ETL и ELT.

При использовании первого подхода (ETL) данные сначала извлекаются из пула источников данных и хранятся во временной промежуточной базе данных. Затем для структурирования и преобразования данных в подходящую форму выполняются операции преобразования и лишь после этого уже структурированные данные загружаются в хранилище.

При использовании второго подхода (ELT) отсутствует промежуточная база данных и данные немедленно загружаются в единый репозиторий сразу после извлечения из исходных пулов данных. Затем они преобразуются в системе хранилища данных [3].

– Информационное хранилище, являющееся ядром всей системы и представляющее собой один или несколько серверов БД.
– Метаданные, содержащие сведения о источнике, использовании, значении и особенности хранилища данных; а также то, как данные могут быть изменены и обработаны.
– Уровень информационного доступа, обеспечивающий взаимодействие пользователя с хранилищем посредством стандартных систем анализа и манипулирования данными.
– Уровень управления, где происходит отслеживание и программирование различных процедур.

Мы рассмотрели традиционную архитектуру хранилищ данных, а также основные компоненты, из которых состоит хранилище данных.

Облачная архитектура

Сама концепция облачных хранилищ относительно нова, и разработана на основе концепции облачных вычислений, которые являются развитием параллельной обработки, распределенных вычислений и грид-вычислений. Благодаря технологии облачных вычислений пользователи могут обрабатывать огромные объемы информации в течении нескольких секунд, за счет разбиения программы обработки на меньшие программы, которые затем обрабатываются и анализируются системой, состоящей из нескольких серверов.

Концепция облачного хранения аналогична облачным вычислениям, каждое облако состоит из нескольких узлов, которые соединены между собой быстрыми каналами передачи и управления большими объемами информации. В зависимости от требований к облачным хранениям и вычислениям существуют различные типы серверов, сетей и технологий хранения. В основе всех облачных сервисов, продуктов и решений лежат программные инструменты с тремя основными функциональными элементами: инструменты для обработки данных и запуска приложений (вычислительные серверы), перемещения данных (сеть) и хранения данных (хранилище).

Каждое хранилище имеет свою уникальную архитектуру, но несмотря на это выделяют четыре модели облачной архитектуры: общедоступная, частная, гибридная, и мультиоблачная архитектуры [4].

- Общедоступная - данная модель характеризуется тем, что вычислительные ресурсы принадлежат и управляются поставщиком и совместно используются, и перераспределяются между несколькими арендаторами через интернет.
- Частная - данная модель облака принадлежит и управляется частным образом, и может включать в себя несколько серверов. Среди ее преимуществ можно выделить настраиваемость и строгие параметры безопасности.
- Гибридная - данная модель совмещает в себе эффективность работы общедоступного облака и возможности защиты данных частного за счет использования публичных и частных облачных архитектур.
- Мультиоблачная - использует несколько общедоступных облачных сервисов. В ее преимущества входит большая гибкость в выборе и развертывании облачных сервисов, снижение зависимости от одного поставщика облачных услуг[4].

Архитектура облачного хранилища включает в себя следующие основные компоненты:

- Виртуализация: облака создаются на основе виртуализации серверов, хранилищ и сетей. Виртуализированные ресурсы - это программное или виртуальное представление физического ресурса, такого как серверы или хранилище, что позволяет нескольким приложениям использовать одни и те же физические ресурсы [5].
- Инфраструктура: Облачная инфраструктура включает в себя все компоненты традиционных центров обработки данных, включая серверы, постоянное хранилище и сетевое оборудование, включая маршрутизаторы и коммутаторы.
- Промежуточное ПО: программные компоненты, такие как базы данных и коммуникационные приложения, позволяют подключенным к сети компьютерам, приложениям и программному обеспечению взаимодействовать друг с другом.
- Управление: эти инструменты обеспечивают непрерывный мониторинг производительности и емкости облачной среды.
- Программное обеспечение для автоматизации: в облачной архитектуре автоматизация используется для простого масштабирования системных ресурсов в соответствии с резким увеличением спроса на вычислительную мощность, развертывания приложений для обеспечения управления в облачной среде [5].

Рассмотрим архитектуру хранилища Amazon Redshift (Рисунок 2.). Оно использует архитектуру массивно-параллельной обработки, в которой узлы хранят данные в срезах в столбчатом формате. Иерархия архитектуры Amazon состоит из множества уровней, таких как клиентские приложения, ведущий узел, вычислительный узел, срез узла. Давайте вкратце рассмотрим функции каждого из этих слоев. Клиентские приложения используются для подключения к кластеру Amazon Redshift через драйверы JDBC или ODBC. Ведущий узел отвечает за связь с клиентским приложением и вычислительными узлами. Вычислительный узел выполняет такие функции, как загрузка данных, резервное копирование и восстановление данных. Срез узла используется для распределения данных внутри узла. Каждый раз, когда ведущий узел назначает операции срезам узла, они начинают работать параллельно, чтобы завершить операцию.

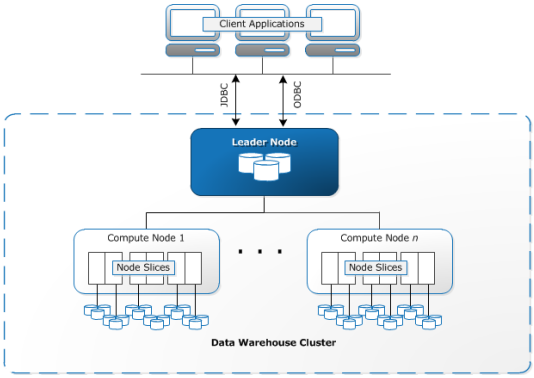


Рисунок 2. Архитектура Amazon Redshift

Заключение

Традиционная архитектура данных все еще широко используется в основном при работе с схожими структурированными типами данных и требует больших затрат на создание и обслуживание, а также уступает в гибкости и скорости обработки данных облачным хранилищам. В свою очередь облачная архитектура отличается масштабируемостью, высокой скоростью обработки данных и гибкостью для различных форматов данных, что дает значительные преимущества в интеграции и производительности. А также благодаря массово параллельной обработке облачные хранилища способны выполнять сложные аналитические запросы в гораздо более быстром темпе.

Библиография

1. Архитектура хранилищ данных: традиционная и облачная–[Online]- [обращения 02.03.2023], Доступно: https://habr.com/ru/post/441538/
2. Data warehouse system architecture – [Online], [обращения 02.03.2023] Доступно: https://docs.aws.amazon.com/redshift/latest/dg/c_high_level_system_architecture.html
3. Data Warehouse Architecture: Types, Components, & Concepts –[Online]- [обращения 02.03.2023], Доступно: https://www.astera.com/type/blog/data-warehouse-architecture/
4. What is Cloud Architecture?–[Online]- [обращения 02.03.2023], Доступно: https://www.vmware.com/topics/glossary/content/cloud-architecture.html
5. Современная архитектура хранилища данных: традиционное или облачное хранилище данных–[Online]- [обращения 02.03.2023], Доступно: https://biconsult.ru/services/sovremennaya-arhitektura-hranilishcha-dannyh-tradicionnoe-ili-oblachnoe-hranilishche-dannyh

POSTGRESQL

Анатолий ГАРБУЗОВ

Департамент Программной Инженерии и Автоматики, группа TI-196, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Анатолий ГАРБУЗОВ, e-mail: garbuzov.anatolie@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *В данной статье рассматривается СУБД PostgreSQL. Описываются структуры и типы данных, модель данных, масштабируемость, расширения и преимущества данной СУБД.*

Ключевые слова: *объектно-реляционная база данных, PostgreSQL, JSON, Postgre95*

Введение

PostgreSQL создана на основе некоммерческой СУБД Postgres, разработанной как open-source проект в Калифорнийском Университете Беркли. Главным разработчиком данной СУБД является Майкл Стоунбрейкер. Начало разработки СУБД пришлось на 1986 год. PostgreSQL разрабатывалась Майклом Стоунбрейкером и его студентами долгие 8 лет, начиная с 1986 года и заканчивая 1994 годом.

Рисунок 1. Объектно-реляционная система управления базами данных, основанная на языке SQL [1]

За весь период разработки синтаксис СУБД пополнился различными процедурами, правилами и другими различными компонентами. В 95-ом году разработка СУБД снова разделилась. Главный разработчик, а именно Майкл Стоунбрейкер, использовал полученный опыт в создании коммерческой СУБД Illustra, а его студенты продолжили заниматься PostgreSQL, в следствии чего создали новую версию СУБД Postgre, получившую название — Postgres95, где POSTQUEL – язык запросов, изменили на SQL. Далее разработку Postgres95 передали одной команде энтузиастов, которые не имели отношение к Университету Беркли [2].

После чего название Postgre95 перестало существовать и ему на смену пришло новое, существующее и по сей день — PostgreSQL.

Особенности PostgreSQL

Надежность. Помимо безопасного хранения данных и предоставления пользователю возможности извлекать данные при обработке запроса, он поддерживается сообществом участников, которые регулярно находят ошибки и пытаются улучшить программное обеспечение, делая PostgreSQL надежным [2].

Производительность. Операции записи в PostgreSQL могут выполняться одновременно без необходимости блокировки чтения/записи. Индексы используются для ускорения запросов при работе с большими объемами данных, что позволяет базам данных находить определенную строку без необходимости циклического просмотра всех данных [2].

В PostgreSQL вы даже можете создать индекс выражения, который работает с результатом выражения или функции, а не только со значением столбца. Также поддерживается частичная индексация, при которой индексируется только часть таблицы. Он также поддерживает распараллеливание запросов на чтение, JIT-компиляцию выражений и вложенные транзакции (через точки сохранения), что обеспечивает высокую производительность и эффективность.

Соответствие стандартам SQL

Еще одна область, в которой PostgreSQL выделяется среди других систем реляционных баз данных, — это ее соответствие стандартам SQL. Стандарты SQL были разработаны группами ANSI и ISO для определения минимальных требований к функциональности и совместимости для реализаций SQL. Хотя спецификации, предоставляемые этими органами, предназначены для определения функций, которые должны предоставлять системы SQL, из-за сложности и длительного развития языка строгое соблюдение не всегда возможно. Согласно документации PostgreSQL, в настоящее время ни одна база данных не удовлетворяет всем требованиям, изложенным в спецификации. Учитывая это предостережение, PostgreSQL удовлетворяет большему количеству спецификаций SQL, чем другие варианты. По их собственным подсчетам, они соответствуют как минимум 160 из 179 основных требований, изложенных в SQL:2016.

Расширения. PostgreSQL может похвастаться несколькими надежными наборами функций, включая восстановление на определенный момент времени, управление параллельным доступом к нескольким версиям (MVCC), табличные пространства, детализированный контроль доступа, асинхронную репликацию, усовершенствованный планировщик/оптимизатор запросов и ведение журнала с опережающей записью. Multi-Version Concurrency Control позволяет одновременное чтение и запись таблиц, блокируя только одновременные обновления одной и той же строки. Таким образом удается избежать столкновений [3].

Масштабируемость. PostgreSQL поддерживает Unicode, международные наборы символов, многобайтовые кодировки символов и учитывает локаль для сортировки, учета регистра и форматирования. PostgreSQL обладает высокой масштабируемостью — он может работать как с количеством одновременных пользователей, так и с объемом данных, которыми он может управлять. Кроме того, PostgreSQL является кроссплатформенным и может работать во многих операционных системах, включая Linux, Microsoft Windows, OS X, FreeBSD и Solaris [3].

Динамическая загрузка. Сервер PostgreSQL также может включать в себя пользовательский код посредством динамической загрузки. Пользователь может указать файл объектного кода; например, общая библиотека, которая реализует новую функцию или тип, и PostgreSQL загрузит ее по мере необходимости. Возможность изменять его работу «на лету» делает его уникально подходящим для быстрой реализации новых структур хранения и приложений.

Преимущества. Поддержка огромного количества типов данных, включая: [4]

- общие примитивы базы данных, такие как числовые, строковые, логические типы и типы даты и времени: они встречаются в большинстве баз данных и отлично поддерживаются в PostgreSQL.

- сетевые адреса: PostgreSQL поддерживает различные типы, связанные с сетью, такие как адреса CIDR, адреса с масками подсети и MAC-адреса, как для IPv4, так и для IPv6.
- геометрические типы: доступно множество типов, которые помогут вам определить двухмерные объекты. К ним относятся точки, линии и сегменты, а также полные фигуры, такие как прямоугольники, многоугольники и круги.
- денежные типы: тип PostgreSQL moneyхранит валюту с фиксированной дробной точностью. Тип учитывает локаль и включает автоматическое форматирование вывода.
- диапазоны: диапазоны позволяют вам работать с диапазонами значений, например, с диапазонами дат и времени для планирования.
- JSONB: встроенная поддержка PostgreSQL для хранения и работы с объектами JSON может помочь вам работать как с реляционными, так и с нереляционными данными в одной системе.
- hstore: этот hstoreтип позволяет вам работать с парами ключ-значение изначально.
- многомерные массивы: массивы очень полезны для хранения значений, состоящих из нескольких частей, которые не имеют большого значения вне своего конкретного контекста.

Возможность определять свои собственные сложные типы. Определение собственных типов помогает вам более точно согласовать вашу базу данных с тем, как ваши данные представлены в ваших приложениях. 4

Полнотекстовый поиск. Полнотекстовый поиск предоставляет мощные методы поиска и работы с данными в полуструктурированном и неструктурированном тексте. Поиск можно настроить так, чтобы он соответствовал вашим ожиданиям в отношении релевантности и соответствия.

Надежные системы аутентификации, контроля доступа и управления привилегиями, подходящие для организаций любого размера: PostgreSQL имеет зрелые функции аутентификации и авторизации пользователей, позволяющие определять, кто может использовать систему и что каждому пользователю разрешено видеть или делать.

Оболочки внешних данных: Обертки внешних данных позволяют представлять и получать доступ к таблицам и данным на удаленных серверах. Представления и материализованные представления.

Поддержка представлений и материализованных представлений обеспечивает удобный и упрощенный доступ к данным за счет абстрагирования исходных структур таблиц для информации, которая часто запрашивается вместе.

Комментарии к объектам базы данных. Возможность прикреплять комментарии к таблицам, базам данных, столбцам и другим отдельным объектам базы данных позволяет документировать решения или детали реализации.

Ведение журнала с опережающей записью для обеспечения восстановления на определенный момент времени, отработки отказа и потоковой репликации. Эти технологии помогают гарантировать, что ваша база данных остается согласованной даже в случае сбоя программного обеспечения, а также помогают копировать данные между системами для масштабирования и резервного копирования данных.

Поддержка поведения, подобного NoSQL, такого как хранение документов с использованием JSONB и пар ключ-значение с помощью hstore . Возможность использования нескольких парадигм базы данных в одной системе может помочь минимизировать ваши административные издержки и улучшить совместимость между различными представлениями.

Заключение

PostgreSQL — это бесплатная система управления реляционными базами данных с открытым исходным кодом, ориентированная на совместимость и расширяемость SQL. Благодаря более чем 30-летнему активному развитию PostgreSQL является одним из наиболее широко используемых инструментов баз данных с открытым исходным кодом во всем мире.

В этой статье мы рассмотрели некоторые важные особенности PostgreSQL, архитектуру PostgreSQL, варианты ее использования, преимущества, операционные проблемы и ключевые альтернативы. Мы завершили его несколькими рекомендуемыми практиками, чтобы поддерживать вашу базу данных WordPress в оптимальном состоянии, пока вы продолжаете масштабирование.

Библиография

1. PostgreSQL — объектно-реляционная система управления базами данных [Электронный ресурс]. [дата обращения 02.03.2023], Доступно:– https://web-creator.ru/articles/postgresql
2. PostgreSQL Википедия[Электронный ресурс]. [дата обращения 02.03.2023], Доступно:–https://ru.wikipedia.org/wiki/PostgreSQL
3. Чем PostgreSQL лучше других SQL баз данных с открытым исходным кодом. [Электронный ресурс]. [дата обращения 02.03.2023], Доступно:– https://habr.com/ru/post/282764/
4. Основные преимущества PostgreSQL для разработчиков [Электронный ресурс]. [дата обращения 02.03.2023], Доступно:– https://www.cischool.ru/osnovnye-preimushhestva-postgresql-dlya-razrabotchikov/

СУБД «DB2» IBM

Дмитрий РОМАНЕНКО

Департамент Программной Инженерии и Автоматики, группа TI-196, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор кореспондент: Дмитрий РОМАНЕНКО, e-mail: romanenco.dmitrii@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация. *Статья посвящена базам данных DB2 IBM: о том, что из себя представляет; достоинства над другими СУБД, а также недостатки; классификация различных версий; функциональные возможности, предоставляемые IBM, а также недостатки подобной СУБД; зачем нужна такая база данных и где применяется*

Ключевые слова: *СУБД, IBM, реляционная, база данных.*

Введение

DB2 — продукт, созданный корпорацией IBM в качестве семейства систем управления реляционными базами данных. Компания IBM постоянно демонстрирует выдающиеся результаты в области производительности, технической реализации и масштабируемости, что подтверждается ее лидерством в отрасли, а также является одной из самых «зрелых» СУБД.

DB2 — система, зародившаяся в начале 1970-х благодаря разработчику Эдгару Кодду, члену команды IBM. Кодд создал теорию реляционных баз данных, а после опубликовал модель использования данных. Чтобы реализовать модель, он разработал язык реляционных баз данных, после названный Alpha. После разработка была переданы группе других программистов, который впоследствие нарушили некоторые принципы модели, сделав ее «структурированным языком запросов», или как она известна сегодня — SQL.

В 1970-х система имела названия System R(или System Relational), в которой впервые был реализован язык SQL. В 1979 IBM выпустила System R в качестве коммерческого продукта, включающего SQL.

В 1982 году модель получила всем известное название DB2. Однако также долгое время до названия DB2, СУБД была известна как “Database 2”, также являющийся торговой маркой IBM[1].

Классификация редакций СУБД IBM DB2

DB2 Everywhere - это DBMS, который можно установить на смартфоны и PDA. Основной его целью является связь с удаленными базами данных. Базу данных можно установить непосредственно на мобильное устройство, однако удаленный доступ невозможен.

Сервер DB2 Express может использоваться с устройствами с до двух процессоров и четырьмя гигабайтами оперативной памяти. Для платформ, работающих на Windows, Linux и Solaris, есть версия Express.

Тот же «экспресс», но исключительно бесплатный, это DB2 Express-C. работает, к сожалению, только на Linux и Windows (хотя есть поддержка для Mac OS Leopard, которая не упоминается в книгах DB2). Кроме того, экстрасы, которые присутствуют в простом экспрессе, здесь отсутствуют. С другой стороны, DB2 Express стоит денег и не имеет pureXML, что позволяет взаимодействовать с XML-данными.

Десктопная версия DBMS называется DB2 Personal Edition, совместимая как с Linux, так и с Windows.

Многопользовательский DB2 Workgroup Server Edition может работать только на 4 процессорах и 16 ГБ оперативной памяти.

Многопользовательский DB2 Enterprise Server Edition не имеет ограничений в обработке или оперативной памяти.

Эта версия DB2 особенно хорошо подходит для «хранилищ данных», или баз данных, где данные постоянно накапливаются и в первую очередь исследуются.

Разработчик приложений баз данных требует от DBMS большего, чем просто набор инструментов и программного обеспечения, что обеспечивает DB2 Personal Developer's Edition. Содержит DB2 Personal Edition для Linux и Windows, DB2 Connect Personal Edition (необходимо подключиться к DB2 для баз данных z/OS и DB2 в базах данных i5/OS), DB2 Extenders (которые расширяют функциональность DB2, чтобы выполнять более сложные поиски и рассчитывать географическое расстояние) и SDK (который содержит примеры и названия в C/C++, PHP, Java и Python).

Кроме того, DB2 Everyplace, Express, Personal, Workgroup Server и Enterprise Server Editions (вместе с любыми дополнениями, включенными в эти версии) все включены в DB2 Universal Developer's Edition. Используя «полную мощность» практически всех технологий DB2, этот набор позволяет создавать приложения.

DB2 Enterprise Developer's Edition поставляется с различными версиями DB2 enterprise Server Edition, а также рядом драйверов и клиентов для подключения к Cloudscape, Informix и DB2 для z/OS.

Эксклюзивная версия операционной системы i5/OS, известная как DB2 для i5 / OS, работает на системах IBM iSeries.

DB2 для z/OS - это система управления базами данных (DBMS), которая работает на мейнфремах IBM класса zSeries, которые являются высокопроизводительными, высококачественными машинами[2].

Достоинства и недостатки DB2 IBM

DB2 имеет ряд преимуществ над другими СУБД, а именно:

- наличие хорошей бесплатной версии;
- отсутствие ограничителя запросв в бесплатной версии;
- хорошая бесплатная техподдержка;
- есть возможность получить платную поддержку производителя, что позволяет применять в Entrprise секторе бизнеса;
- достойная производительность;
- исправленные проблемы с неуникальностью индексов;
- возможность широко работать с RLS, т.к. нет ограничения на 256 таблиц.

Несмотря на большой список преимуществ, DB2 также имеет и серьезные недостатки:

- мало специалистов и высокая стоимость квалифицированных специалистов;
- маленькая распространенность;
- в отличии от MS SQL Server, для новых версий 1С выпускает "адаптированные" версии (в прочем, тоже самое верно и для постгресса);
- размер баз больше, чем в других СУБД;
- долгая загрузка dt-файла;
- сложность в конфигурировании парметровт СУБД;
- некоторые сообщения платформой могут не верно обрабатываться, для решения приходиться "понижать уровень" логирования ошибок;
- не собирает взаимные блокировки[3].

Функциональные возможности DB2

– Кроссоверная платформа. Некоторые из самых популярных операционных систем поддерживаются даже самой базовой версией «IBM DB2 Express - C»: Windows (включая x64), Linux (POWER и x86 серверы), Mac OS X и Solaris x64.

– ACS или Advanced Copy Services. Вы можете использовать технологию быстрого хранения копирования с DB2 ACS для выполнения операций копирования данных для резервного копирования и восстановления. Операции по резервному копированию и восстановлению значительно ускоряются возможностью копировать данные с помощью устройства хранения данных. Snapshot backup - это термин для резервного копирования, выполняемого с использованием DB2 ACS.

– Онлайновая реорганизация таблиц, также известная как перестройка на месте, позволяет пользователям перестроить таблицы, не жертвуя своим доступом к ним полностью.

– Высокая доступность аварийного восстановления (High Availability Disaster Recovery – HADR). Для баз данных DB2 поддержка высокой доступности и перезагрузки обеспечивается через возможности DB2 HADR.

– Поддержка 64-битных инстанций.

– Поддержка многомерных группировок таблиц (MDC, Multidimensional Table Clustering), материализованных таблиц запросов (MDT) и параллелизма запросов (QP)

– Поддержка SQL-репликации и сжатия данных при резервном копировании.

– Помощь в разделении баз данных (distributing databases). Кластеры масштабирования: первоначально известные как DB2 EEE. Вы можете развернуть одно изображение базы данных на многочисленных физических серверах с помощью этой функциональности.

– Поиск текста в DB2. Вы можете искать текстовые столбцы таблиц DB2 с помощью инструмента поиска текста. До тех пор, пока вы используете предоставленные вами поисковые аргументы, поддержка текстового поиска позволяет искать текстовые индексы с помощью встроенных функций DB2 CONTAINS, SCORE и xmlcolumn-contains.

– Имея потенциал для дальнейшего расширения поддержки, IBM Data Studio является инструментальной платформой, которая охватывает полный жизненный цикл приложений (проектирование, разработка, развертывание, поддержка и администрирование) для всех IBM реляционных DBMS [1]..

Типы данных DB2

Таблица 1.

Числовые типы данных [4]

Тип данных	Размерность типа	Характеристики типа данных
BIGINT	от -9223372036854775808 до 9223372036854775807	Целое число. Использует для хранения данных 63 бита (+ 1 бит для хранения знака).
INT	от -2147483648 до 2147483647	Целое число. Использует для хранения данных 31 бит (+ 1 бит для хранения знака).
SMALLINT	от -32768 до 32767	Целое число. Использует для хранения данных 15 бит (+1 бит для хранения знака).
DECIMAL / NUMERIC	от 1 - 10^{31} до 10^{31} - 1	Десятичное число. Положение запятой может быть указано при инициализации.
FLOAT	от -7,2E 75 до 7,2E 75	Дробное число одинарной точности. Использует для хранения данных 32 бита.
DOUBLE	от -7,2E 75 до 7,2E 75	Дробное число двойной точности. Использует для хранения данных 64 бита.

Таблица 2.

Строковые типы данных

Тип данных	Характеристики типа данных
CHAR(X)	Фиксированная строка их X символов. По умолчанию создается строка длиной 1 символ. X должен быть больше нуля, максимальное значение - 255.
VARCHAR(X)	Строка переменной длины. Максимальная длина строки не должна превышать X символов. Минимальное значение X - 1, максимальное значение - 32704.
CLOB(X)	Длинная строка переменной длины, максимальная длина не должна превышать X символов. Максимальное значение X - 2 147 483 647. Значение по умолчанию - 1M.

Таблица 3.

Типы данных для представления времени [3]

Тип данных	Характеристики типа данных
DATE	Дата. Служит для хранения даты, представленной в виде года, месяца и дня.
TIME	Время. Служит для хранения времени, представленного в виде часов, минут и секунд. Часы представлены в 24-часовом формате. Если значение часов равно 24, тогда минуты и секунды равны нулю.
DATETIME	Дата и время. Служит для хренения даты и времени. Содержит год, месяц, число, часы, минуты, секунды и доли секунд.

Заключение

Несомненно, мощный DB2 предлагает широкий спектр функций, которые очень полезны, но не поддерживаются MySQL. С такими функциями DB2 является надежным вариантом для вашей основной базы данных. Работа в DB2 проста и полезна для всех профилей благодаря различным функциям.

Библиография

1. DB2 Корпорации IBM. – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: http://bourabai.kz/dbt/servers/DB2.htm
2. База данных IBM BD2. – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://revolution.allbest.ru/programming/00246855_0.html
3. DB2 – Википедия. – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://ru.wikipedia.org/wiki/DB2
4. Краткое руководство по DB2. – [Электронный ресурс]. – [дата обращения 02.03.2023], Доступно: https://coderlessons.com/tutorials/bazy-dannykh/izuchite-db2/kratkoe-rukovodstvo-po-db2

GRAPHQL – ЯЗЫК ЗАПРОСОВ И МАНИПУЛИРОВАНИЯ ДАННЫМИ С ОТКРЫТЫМ ИСХОДНЫМ КОДОМ

Роман ПОГОНЕЦ

Департамент Программная Инженерия и Автоматика, группа TI-196, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Молдова

Автор Корреспондент: Роман ПОГОНЕЦ, e-mail: roman.pogonet@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Данная работа посвящена GraphQL. Она освещает историю этого популярного языка запросов, рассказывает о его преимуществах, недостатках и принципе работы. В статье будет рассмотрен пример, показывающий принцип работы GraphQL, а так же будет сделан вывод на основе рассмотренного материала.*
Ключевые слова: *GraphQL, язык запросов, данные, запрос, таблицы.*

Введение

GraphQL – это язык запросов и обработки данных, открытый для API и среда исполнения, предназначенный для выполнение запросов к существующим данным. GraphQL был разработан внутри компании Facebook в 2012 году, а затем выпущен в открытый доступ в 2015 году [1].

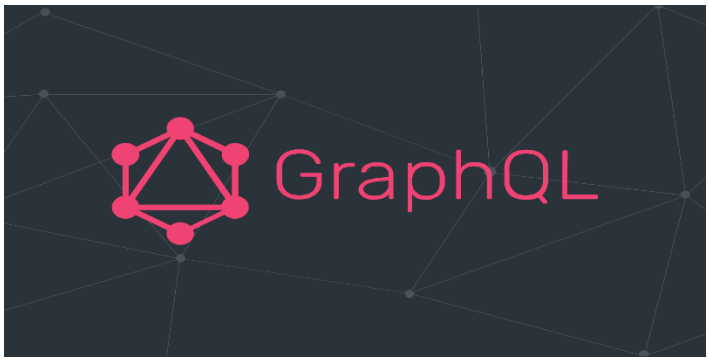


Рисунок 1. GraphQL

GraphQL был разработан с учётом гибкости, скорости и пожеланий разработчиков. GraphQL позволяет разработчикам создавать API по своему усмотрению и предоставлять функциональность API клиентам, использую спецификацию GraphQL.

Одна из ключевых концепций запросов GraphQL заключается в том, что возвращаемые данные предсказуемые. Вместо дополнительных строк кода возвращается, что было запрошено. Такая декларативная выборка данных особенно полезная на мобильных устройствах с ограниченной пропускной способностью: приложения, использующие GraphQL, также стабильны и быстры, поскольку они контролируют запрашиваемые и получаемые данные, а не сервер. Они также быстро работают на медленных сетевых соединениях, поскольку один запрос может одновременно возвращать несколько запрашиваемых данных [2].

Основываясь на упрощении данных GraphQL, API организован по типам и полям, а не по конечным точкам. Это означает, что все данные доступны из одной конечной точки, что позволяет приложениями запрашивать только то, что им нужно. Снижая эксплуатационную сложность, GraphQL присоединяется к рабочему процессу оптимизированных решений для подключения API.

Как работает GraphQL?

GraphQL поддерживает подробные запросы данных, предоставляя клиентам больше контроля над отправляемой информацией. Клиенты отправляют запросы GraphQL в виде запросов, а сервер возвращает ответы в формате JSON. GraphQL не требует специальной архитектуры приложения, может использоваться в различных средах (включая интегрированные среды разработки [IDE]) и может применяться с существующими инструментами управление API или поверх существующих RESP API.

Важные термины в GraphQL включает:

Схема – Созданная разработчиком API для указания того, какие данные могут быть запрошены клиентом, схема состоит из типов объектов, которые определяют, что может быть запрошено клиентом, и полей, которые представляют атрибуты объекта.

Запрос – запросы проверяются и зачем выполняется на основе схемы: GraphQL не позволяет выполнять запросы без определения типов объектов.

Преобразователь – это преобразователя, связанная с каждым полем в схеме, вызывается для генерации значения при выполнении API. Преобразователи являются важным архитектурным элементом GraphQL.

Одной из уникальных особенностей GraphQL являются то, что ответ отражает структуру запроса. Это упрощает анализ клиента, поскольку формат ответа сервера полностью предсказуем [3].

Основные различие между REST API и GraphQL

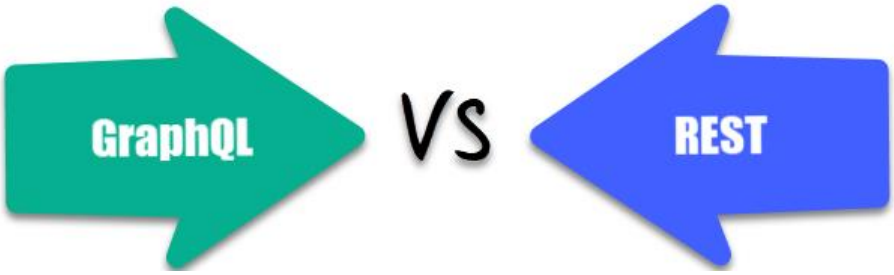


Рисунок 2. GraphQL vs Rest

GraphQL

GraphQL – это серверная технология прикладного уровня, разработанная компанией Facebook которая выполняет запросы к существующим данным он следует архитектуре, управляемой клиентом, может быть организовать в виде схемы, имеет растущее сообщество, высокая скорость разработки, кривое обучение сложное для понимания, строго типизирован, имеет единственную конечную точку, использует метаданные для проверки запросов [4].

REST

Rest – это стиль архитектурный программного обеспечения, определяющий набор ограничений для построение веб-сервисов. Используя серверно-управляемую архитектуру, Rest может быть организован с точки зрения конечных точек, большое сообщество, медленная скорость разработки, умеренная для понимание, слабо типизирован, несколько конечных точке, нет возможности кэширование машиночитаемых метаданных [5].

Преимущества GraphQL

- Предоставляет язык запросов, который является декларативным, а не императивным.
- Он иерархичен и ориентирован на продукт.
- GraphQL строго типизирован. Это означает, что запросы выполняются в определенной системе контексте.
- Запросы в GraphQL кодируются на клиенте, а не на сервере.
- Он обладает всеми возможностями прикладного уровня модели OSI.

- GraphQL обеспечивает читабельные запросы.
- В GraphQL легко интегрирует со многими базами данных.
- Данные могут быть получены с помощью одного вызова API.
- Помогает в пакетной обработке запросов и кэширование.
- Настраивает запросы в соответствии с вашими потребностями.
- Помогает находить формы в нужном формате
- GraphQL автоматически синхронизирует документацию с изменениями API.
- Эволюция API возможна без управления версиями.
- Его можно использовать для быстрого прототипирования приложений.
- Поля GraphQL могут быть переданы на более высокий уровень компонента для повторного использования.
- Это позволяет вам выбирать, какие функции выставлять и как они работают [6].

Недостатки GraphQL

- Недостаток ресурсов на бэкэнд части.
- Отсутствует шаблонов проектирования для сложного приложения.
- Проблемы с производительностью при сложных запросах.
- Слишком много для небольших приложений.
- Не полагается на метод кэширования HTTP, позволяющих сохранять содержимое запроса.
- Вам необходимо изучить язык определения схемы GraphQL, прежде чем внедрять стратегии GraphQL.
- GraphQL использует одну конечную точку вместо кэширования HTTP.
- Не лучшее решение для простых приложений, поскольку оно может добавить сложности [7].

Заключение

Реализация GraphQL может быть сложной, но это впечатляющий шаг вперед в обработке дынных. GraphQL не решает все проблемы, но это определенно технология, которую стоит попробовать. Например можно начать с размышлений о наиболее запутанных и неудобных подсистем обработки данных в вашем проекте и попытаться реализовать эти подсистемы на GraphQL.

Библиография

1. Подробности о GraphQL [online]. [дата обращения 10.03.2023], Доступно: https://habr.com/ru/company/ruvds/blog/445268/
2. Введение в GraphQL [online]. [дата обращения 11.02.2023], Доступно: https://graphql.org/learn/
3. Что такое GraphQL [online]. [дата обращения 11.02.2023], Доступно: https://www.redhat.com/en/topics/api/what-is-graphql
4. GraphQL vs Rest [online]. [дата обращения 12.02.2023], Доступно: https://www.guru99.com/graphql-vs-rest-apis.html#:~:text
5. GraphQL против Rest [online]. [дата обращения 12.02.2023], Доступно: https://www.rubrik.com/blog/technology/19/11/graphql-vs-rest-apis
6. GraphQL [online]. [дата обращения 12.02.2023], Доступно: https://hygraph.com/academy/what-is-graphql
7. Что такое GraphQL [online]. [дата обращения 12.02.2023], Доступно: https://www.nginx.com/resources/glossary/graphql/

SISTEME DE STOCARE A DATELOR: INNODB VS MYISAM

Valentina ASTAFI

Departamentul Ingineria Software și Automatică, grupa FI-191, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Moldova

Autor corespondent: Astafi Valentina, astafi.valentina@isa.utm.md

Rezumat. *Sistemele de stocare, odată cu evoluția timpului, au devenit mai complexe și dinamice, iar nevoia de stocare a datelor a crescut la fel și alegerea unei baze de date care să corespundă tuturor cerințelor, a devenit cam dificilă. MyISAM si InnoDB sunt două motoare de stocare din MySQL cel mai utilizate. Pentru îmbunătățirea performanțelor bazelor de date, este necesar de a cunoaște diferențele dintre diverse motoare de căutare și de stocare a datelor. MyISAM este un motor de stocare implicit, care mai târziu a fost înlocuit cu InnoDB. MyISAM se bazează pe algoritmul ISAM care afișează mai rapid rezultatul din seturi de date mai mari. Fiecare sistem de stocare a datelor are diverse utilizări: MyISAM este mai potrivit pentru depozitarea de date cu volume mici și aplicațiile web, iar InnoDB este mai potrivit pentru o bază de date mai mare, deoarece acceptă tranzacții de volum.*

Cuvinte cheie*: sisteme de stocare a datelor, performanță, fiabilitate, ACID (atomicitate, coerență, izolare, durabilitate), ISAM (Indexed Sequential Access Method), baze de date, SQL (Structured Query Language).*

Introducere

MySQL este un sistem de management al bazelor de date relaționale, care înseamnă că este un sistem de gestionare a bazelor de date bazat pe model relațional. Este folosit pentru diverse aplicații și este construit în jurul SQL (Structured Query Language).

Caracteristicile cheie ale MySQL

- *Fiabilitate și ușurință în utilizare:* MySQL este ușor de descărcat, instalat și ușor de utilizat;
- *Scalabilitate:* MySQL are o arhitectură scalabilă și oferă utilități de încărcare rapidă de înaltă performanță, cu diferite memorii cache;
- *Compatibilitate:* MySQL este compatibil și poate fi instalat pe toate platformele moderne precum Windows, Linux, Unix;
- *Securitatea datelor:* MySQL are toate caracteristicile care asigură securitatea bazelor de date și le permite doar utilizatorilor autorizați să le acceseze;
- *Susținere de multe limbaje de programare* (PHP, Java, C ++, PERL etc.);
- *Licența open-source* MySQL este personalizabilă. Este gratuit de utilizat pentru uz personal, un dezvoltator îl poate modifica în funcție de cerințele aplicației [1].

Pentru a putea alege cel mai convenabil motor de stocare, este foarte important și tipul motorului de stocare. Astfel deosebim :

- *Motoarele tranzacționale* - ating performanțe bune de citire simultană și dacă în timpul unei operațiuni apare o problemă hardware sau electrică și operațiunea nu poate fi finalizată, datele vechi pot fi recuperate și nu sunt corupte cu fragmente de date noi;
- *Motoarele non tranzacționale* - oferă performanțe mai bune, deoarece nu sunt supuse numeroaselor verificări, sunt rapide, folosesc puțin spațiu pe disc și consumă mai puțină memorie [2].

Sistemele de stocare MyISAM și InnoDB

MyISAM și InnoDB sunt două motoare de baze de date, care pot fi aplicate la nivelul fiecărei tabele dintr-o bază de date. MyISAM este un tip de stocare non tranzacțional și orice opțiune de scriere trebuie să fie anulată manual, este utilizat în cazul versiunilor mai vechi ale MySQL, în timp ce în cazul versiunilor mai noi, motorul InnoDB este definit în mod implicit.

InnoDB este un tip de stocare a tranzacțiilor care anulează automat scrierile dacă nu sunt finalizate. Pentru îmbunătățirea performanțelor bazelor de date, este necesar de a cunoaște diferențele dintre diverse motoare de căutare.

MyISAM este cel mai răspândit tip, stabil și dovedit, acest lucru se datorează în mare măsură faptului că nu gestionează relațiile sau tranzacțiile și, prin urmare, evită controalele consumatoare de resurse, dar pierde în siguranță ceea ce câștigă în viteză, dar își pierde din popularitate din cauza lipsei sale de funcționalitate: fără tranzacții, fără recuperare în caz de accident, capabil să lucreze cu date mari și foarte rapid pentru SELECT-uri lungi. Aceste tabele ocupă mai puțin spațiu decât omologii lor, recunoscând indexurile full-text și tabelele comprimate.

InnoDB este un motor relațional, se asigură că relațiile dintre datele mai multor tabele sunt consistente și că, dacă modificăm anumite date, aceste modificări se reflectă în tabelele legate, este obligatoriu atunci când se lucrează cu tranzacții (tip ACID) și chei străine. Pe de altă parte, toate actualizările trebuie să fie efectuate prin tranzacție ceea ce poate încetini sistemul [3].

De ce să trecem de la MyISAM la InnoDB?

InnoDB este mai eficient (timpul de execuție al anumitor interogări poate fi redus considerabil), mai puțin predispus la corupția datelor datorită gestionării tranzacțiilor, iar la nivel de blocare, InnoDB este capabil să blocheze un singur rând al unei tabele, în timp ce MyISAM blochează întreaga tabelă (ceea ce poate duce la disfuncționalități). Cu toate acestea, motorul InnoDB consumă mai mult spațiu pe disc. Doar înainte, de a face orice operațiune, este necesar de a face o copie de rezerva a bazei de date.

Pentru a vedea care sunt totuși caracteristicile principale ale acestor sisteme de stocare, o să facem o paralelă între MyISAM și InnoDB [4].

Avantaje/Dezavantaje

Motoarele de stocare administrează maniera în care informația este organizată, stocată și accesată, dar ca și oricare alt subiect au avantaje și dezavantaje, dar ele trebuie analiza din punctul de vedere al nevoile aplicației și nevoile utilizatorului [5].

MyISAM

Principalele avantaje ale utilizării MyISAM sunt:

- Administrare mai ușoară;
- Posibilitatea de a bloca la nivel de tabel (table lock);
- Flexibilitate în ceea ce privește integritatea datelor;
- Posibilitatea de a copia direct fișierele de pe un server pe altul, suport pentru tabele comprimate.
- Foarte rapid pentru interogări SELECT sau INSERT;
- Suportă indexuri fulltext, ceea ce permite căutări pe cuvinte bazate pe un index specific, accelerând astfel căutările;

Dezavantaje:

- Nu acceptă tranzacții;
- Nu acceptă chei străine;
- Flexibilitatea conduce la unele aspecte negative la menținerea unei bune integrității datelor [6].

Exemplu: *ALTER TABLE `table` ENGINE=MYISAM*;

InnoDB

Principalele avantaje ale utilizării InnoDB sunt:

- Posibilitatea de a gestiona bine integritatea datelor, chiar și în cazul unei căderi a serverului;
- Gestionează tranzacțiile (set de cereri grupate), cum ar fi BEGIN, COMMIT, ROLLBACK;

- Gestionarea cheile străine și a constrângerile de integritate;
- Sistem de recuperare automată în caz de eroare, accident, cu sistem de reluare a jurnalului;
- Suport ACID pentru a se asigura că toate înregistrările trec sau nu trec;
- Minimizarea maximă a riscurilor cauzat de erori, accidente;
- Permite indexări full text de la lansarea MySQL 5.6.

Dezavantaje:

- Administrarea bazei de date devine mai complicată;
- Motor de stocare mare, necesită multe resurse și din acest motiv este foarte lent;
- Strictețe în legătură cu integritatea datelor;
- Complexitate din punct de vedere al administrării;
- Posibilitatea de blocare rând, pentru a introduce sau actualiza o înregistrare [7].

Compararea și optimizarea performanței a celor două sisteme de stocare

Pentru a înțelege mai bine diferențele specifice de performanță ale motoarelor de stocare trebuie să efectuăm câteva teste asupra bazelor de date.

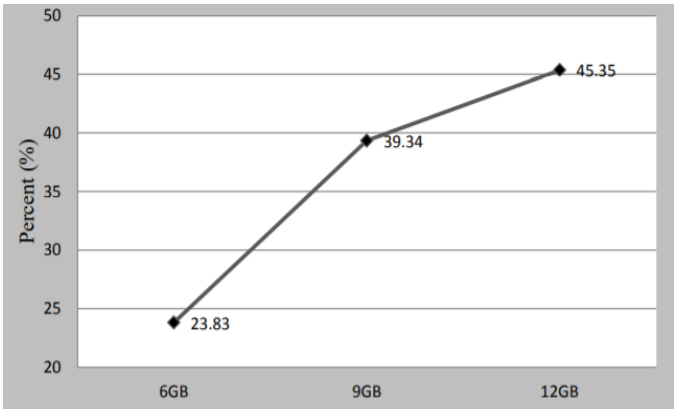


Figura 1. Creșterea procentuală a performanței MyISAM la InnoDB

A fost efectuat un test complet pentru a pre completa memoria cache de interogări MySQL și a fost înregistrată o rulare ulterioară. Aceste date obișnuite au fost comparate cu prima rulare a testului de debit al setului de date existent de 12 GB. Au fost analizate doar interogări legate de tabelele identice.

Datele din Figura 1 arată că MyISAM depășește InnoDB cu douăzeci și patru la sută utilizând setul de date de 6 GB, treizeci și nouă la sută la 9 GB și patruzeci și cinci la sută la 12 GB [8].

Modificarea procentuală se calculează împărțind diferența de timp de execuție la timpul de execuție. Această ecuație ilustrează creșterea performanței de la MyISAM la InnoDB.

Optimizarea performanței

Strategiile de testare de obicei se bazează pe rezultatele testelor. Unele aspecte de optimizare pentru MyISAM sunt:

- *key_buffer_size:* numărul de buffer-uri care sunt distribuite în cache-ul indexului MyISAM;
- *query_cache_size:* numărul de cache distribuite în cache-ul de interogări;
- *long_query_time:* setați timpul de interogare lentă;
- *external-locking:* interziceți utilizarea blocării externe și preveniți blocarea;
- *back_log:* numărul de solicitări poate fi stivuit înainte ca temporar să nu mai răspundă la solicitări noi;
- *table_cache:* numărul de tabel de date din cache pentru a evita cheltuielile pentru deschiderea tabelului în mod repetat;
- *thread_cache_size:* numărul de fire de execuție pe care memoria cache le poate folosi pentru a reduce cheltuielile pentru crearea de noi fire [9].

Pentru InnoDB unele aspecte de optimizare sunt:

- *innodb_buffer_pool_size:* 70-80% din memorie;
- *innodb_log_file_size:* aceasta depinde de nevoile dvs. de viteză de recuperare (în mediu 256 M, un echilibru bun între timpul rezonabil de recuperare și performanță bună);
- *innodb_log_buffer_size:* 4M(mediu);
- *innodb_thread_concurrency:* 8 (implicit) [10, 11].

Motorul de stocare a bazei de date este o parte foarte importantă a bazei de date și asigurarea condițiilor fiecărui motor de stocare este o precondiție importantă pentru optimizarea motoarelor de stocare a bazei de date. Mai sus sunt enunțate caracteristicile celor două motoare și strategiile de optimizare prin studierea lor.

Concluzii

Tehnologiile au schimbat și vor continua să schimbe modul de organizare, desfășurare și funcționare a vieții din toate punctele de vedere. Astfel, dezvoltarea progresivă a tehnologiilor informaționale ne face să ne adaptam gândirea și resursele conform cerințelor. Motoarele de stocare influențează în mod direct performanțele unei baze de date, ele se pot îmbunătăți, doar că trebuie să definim clar caracteristicile și să analizăm ce sistem informatic utilizăm.

Atât InnoDB, cât și MyISAM au cele mai bune performanțe, alegerea unui motor de stocare se face în dependență de cerințele utilizatorului. MyISAM este preferată atunci când este necesar în mod special să se efectueze cereri de citire sau de inserție de date, iar InnoDB pentru sistemele care nu trebuie să conțină erori și care necesită chei străine. Ca de exemplu pentru o aplicație care gestionează cantități mari de date, cum ar fi o aplicație bancară, acesta este motorul de stocare recomandat.

Referințe

1. MICHAEL KOFLER, The Definitive Guide to MySQL. *Editura Apress*, 2005, 816 p., ISBN: 978-1-59059-144-4.
2. MyISAM, InnoDB...: les différents types de stockage SQL [accesat 13.12.2022]. Disponibil: http://www.journaldunet.com/developpeur/tutoriel/out/040511-sql-types-stockage-myisam-innodb.shtml.
3. InnoDB vs. MyISAM – which one should you choose? [accesat 18.12.2022]. Disponibil: https://breachdirectory.com/blog/innodb-vs-myisam-which-one-should-you-choose/.
4. Mysql: principalele diferențe între MyISAM și InnoDB [accesat 18.12.2022]. Disponibil: http://www.tux-planet.fr/mysql-les-principales-differences-entre-myisam-et-innodb/.
5. Storage Engine Performance Benchmark for MyISAM and InnoDB [accesat 09.12.2022]. Disponibil: https://t.ly/dek8.
6. Study and Optimization Based on MySQL Storage Engine [accesat 18.12.2022]. Disponibil: https://t.ly/dIJ-y.
7. A Comparison of Leading Database Storage Engines in Support of Online Analytical Processing in an Open Source Environment [accesat 18.12.2022]. Disponibil: https://dc.etsu.edu/cgi/viewcontent.cgi?article=2271&context=etd.
8. InnoDB [accesat 13.12.2021]. Disponibil: https://mariadb.com/kb/en/innodb/.
9. MyISAM Overview [accesat 13.12.2022]. Disponibil: https://mariadb.com/kb/en/myisam-overview/.
10. Sintaxa SQL și structura datelor →Motoare de stocare [accesat 18.12.2022]. Disponibil: https://mysql.developpez.com/faq/?page=Moteurs-de-stockage.
11. Mysql: principalele diferențe între MyISAM și InnoDB [accesat 18.12.2022]. Disponibil: http://www.tux-planet.fr/mysql-les-principales-differences-entre-myisam-et-innodb/.

ВЗАИМОДЕЙСТВИЕ R И SQL SERVER

Рувим БОЗАЖИ

Департамент Программной Инженерии и Автоматики, группа TI-195, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Рувим БОЗАЖИ, e-mail: bozaji.ruvim@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *в данной работе проводится описание взаимодействия R и SQL server. В статье представлены история интеграции, а также описаны базовые возможности, принцип работы и основные инструменты.*

Ключевые слова: *SQL Server, R Services, T-SQL, база данных*

Введение

R — это интерпретируемый язык программирования с минимальным набором примитивных типов данных для статистической обработки данных с открытым исходным кодом. Основной способ работы с R – командный интерпретатор. Используется как ПО для анализа данных, статистической графики и является одним из самых популярных языков науки о данных [1].

Для удобной работы языка, были созданы графические интерфейсы, такие как RStudio, а также в некоторых редакторах кодов предусмотрены специальные режимы для R. Со временем этот язык, с огромным количеством библиотек для машинного обучения, практически стал стандартом для статистических программ. Большие возможности R стали средством расширения функционала SQL.

1. Интеграция R в SQL

В 2016 году, преследуя цель обеспечить расширенную аналитику на локальных платформах данных, Microsoft приобрела R, после чего интегрировала этот язык с SQL Server и с другими программными обеспечениями [2].

Таким образом Revolution R Enterprise, самое быстрое и экономичное решение для масштабной аналитики и доступа к информации, скрытой в данных, был переименован в SQL Server R Services.

R Services - это функция, которая интегрирует язык R с SQL Server дает возможность запускать скрипты R с реляционными данными, выполнять аналитику и устранять риски безопасности, связанные с перемещением данных. Архитектура спроектирована таким образом, что внешние скрипты, использующие R, выполняются в отдельном процессе от SQL Server [3].

2. Взаимодействие с SQL

R, как язык программирования с открытым исходным кодом, имеет все программные конструкции, такие как переменные, циклы, операторы. Однако его ключевым отличием является то, что он состоит из огромного количества библиотек для статистических вычислений, визуализации, обработки естественного языка, машинного обучения и т. Д.

Когда возникает необходимость вычислить определенную статистику в обычном коде T-SQL, появляется проблема с тем что это требует очень сложной логики в своей реализации. Поэтому используется R для этой цели, поскольку есть готовые к использованию библиотеки, которые могут вычислять эти значения без необходимости изучения или реализации каких-либо сложных статистических формул.

Так как для выполнения скриптов в SQL используется SQL Server Management Studio, то его же можно использовать чтобы вписать R в экосистему SQL.

Запуск команд на R внутри SQL Server создаётся с помощью отдельной хранимой процедуры *sp_execute_external_script*, которая принимает несколько параметров. Основными обязательными параметрами являются *@language* и *@script***.** *@language* принимает значение – R. А в *@script* передается код на языке R, который необходимо выполнить. При запуске R из T-SQL с помощью служб машинного обучения, все скрипты R выполняются вне процесса SQL Server, это обеспечивает безопасность и большую управляемость. А пользователь получает к себе результаты в виде таблиц, графиков [4].

Второй способ применить R в SQL - с помощью специального API посылать команды на SQL Server, где выполняются все вычисления, а возвращаются только результаты. Таким образом есть возможность избежать долгого процесса загрузки данных на свою машину.

Одна из главных особенностей — это возможности R по графическому представлению данных. R позволяет генерировать эти графики внутри SQL Server 2016 и отображать в Reporting Services.

3. Проблемы

R обрабатывает только те данные, которые на данный расположены в оперативной памяти что создает проблемы, когда данные выходят за пределы оперативной памяти. Поэтому подобные задачи возможно решить если разделить их по частям. R Server содержит расширенные пакеты и функции, которые позволяют обрабатывать данные многопоточно, загружая в память необходимые отрывки по мере необходимости. Все эти технологии были встроены и в SQL Server.

Т.к. T-SQL и R разные языки, каждый со своими типами данных, то сразу возникает много вопросов о совместимости между ними. R поддерживает только ограниченный набор типов: numeric, integer, complex, logical, character, date/time и raw. Все типы T-SQL будут неявно преобразованы в их соответствующие аналоги в R.

Заключение

R Services значительно расширяет возможности SQL Server особенно в области анализа данных и предоставляет те функции, которые невозможно выполнить в T-SQL. SQL Server R Services предоставляют большой дополнительный функционал и ряд библиотек которые значительно упрощают работу. Задачи обработки больших объемов данных обычно предпочтительнее выполнять в T-SQL, а с помощью R возможно выполнять специальные задачи, такие как статистические вычисления, машинное обучение, обработка больших объемов данных.

Библиография

1. Getting started with R in SQL Server. - [online]. – [дата обращения 02.03.2023], Доступно: https://www.mssqltips.com/sqlservertutorial/9040/getting-started-with-r-in-sql-server/
2. R development on SQL Server - [online]. – [дата обращения 02.03.2023], Доступно: https://docs.microsoft.com/en-us/sql/machine-learning/r/sql-server-r-services?view=sql-server-2016
3. SQL Server 2016: R Services. - [online]. – [дата обращения 02.03.2023], Доступно: https://olontsev.ru/tag/r-services/
4. Служба SQL Server R – для чего? - [online]. – [дата обращения 02.03.2023], Доступно: http://wisedata.ru/2017/01/12/rec-sql-server-r-service/#respond

НОВОВВЕДЕНИЯ MICROSOFT SQL SERVER

Владимир ВЕРЕТКО

Департамент Программной Инженерии и Автоматики, группа TI-195, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Владимир ВЕРЕТКО, e-mail: veretco.vladimir@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Статья посвящена дополнительным возможностям и функциям, появившимся в Microsoft SQL Server 2019 - новой и самой актуальной на сегодняшний день версии системы, разработанной компанией Microsoft для управления реляционными базами данных.*

Ключевые слова: *кластеры данных, приблизительная обработка запросов, интерфейс, пакетный режим, скалярные функции, компиляция переменных.*

Введение

В течение последних 10 лет Microsoft SQL Server значительно усовершенствовался, пройдя путь от небольшой СУБД до мощной платформы, используемой на крупных проектах с высоким уровнем отказоустойчивости и надежности. С очередным выпуском SQL Server все больше является единым центром управления данными, поскольку учитывает все современные требования по работе с различными форматами данных и широко применяется при построении платформы интеграции, хранении, обработке и анализе любых данных [1].

Версия SQL Server 2019 (15.x), выпущенная в октябре 2019 г., основана на предыдущих выпусках и расширяет SQL Server как платформу, предоставляя клиенту больший выбор типов данных, языков разработки, локальных или облачных сред и операционных систем [2].

1. Редакции Microsoft SQL Server 2019

Microsoft SQL Server 2019 был выпущен в четырех редакциях. Наиболее полная из них - Enterprise - имеет все параметры SQL Server и применяется при работе с крупными БД, требующими максимально высокой производительности, масштабируемости, доступности и надежности.

Редакция Standard, будучи самой распространенной, располагает ключевыми возможностями управления БД. В отличие от версии Enterprise, у Standard имеются ограничения как в объеме использования ресурсов, так и в функциональности.

Редакция Developer предназначена для разработчиков ПО и включает весь функционал SQL Server, позволяя разрабатывать и проводить тестирование различных приложений на основе SQL Server. Она является бесплатной, однако ею могут пользоваться только программисты при работе над приложениями, в то же время в качестве сервера БД ее использовать нельзя.

Последняя версия – Express – является бесплатной и используется для обучения, а также при создании приложений по обработке данных на небольших персональных компьютерах и серверах. У этой версии также имеются ограничения. Так, например, можно задействовать только 4 ядра, а максимально возможный размер базы данных не превышает 10 ГБ. Опции данного выпуска значительно ниже функционала платных версий.

2. Нововведения в Microsoft SQL Server 2019

На сегодняшний день предприятия часто полагаются на большие и непрерывно увеличивающиеся объемы данных, расположенных в неконтролируемых для компании источниках. В новой версии данной СУБД можно практически в реальном времени получить ценную информацию из всех данных, используя т.н. кластеры больших данных,

обеспечивающие многофункциональную среду для обработки больших наборов данных, в том числе с применением машинного обучения и искусственного интеллекта [3].

Другое нововведение в SQL 2019 - применение функции APPROX_COUNT_DISTINCT для приблизительной обработки запросов. Данная функция выполняет статистическое вычисление COUNT(DISTINCT()), однако для крупных наборов данных она была усовершенствована. Таким образом, она потребляет меньше ресурсов ОС и обеспечивает лучший параллелизм. Ее обычно используют в случаях, когда главную роль играет не точность выполнения запроса, а время его работы.

Параметр LIGHTWEIGHT_QUERY_PROFILING, добавленный в SQL 2019, является параметром области базы данных, который позволяет клиенту активировать или отключать инфраструктуру профилирования запросов. Упрощенная инфраструктура профилирования запросов (LWP) позволяет эффективнее обрабатывать запросы в отличие от обычных механизмов профилирования. По умолчанию в версии 2019 г. она активирована.

SDK для языка Java – интерфейс, расширяющий язык Java и использующийся во время выполнения кода Java из SQL Server и обмена с ней данными. Если раньше в базе данных SQL сервера можно было выполнять код только на языках Python и R, то теперь это стало возможным и на Java [4].

Улучшенная поддержка Linux – теперь на Linux поддерживаются такие службы, как координатор распределенных транзакций (MSDTC), репликация, OpenLDAP для сторонних поставщиков, а также система отслеживания изменений данных (CDC) и службы машинного обучения.

Пакетный режим для данных rowstore – данная опция позволяет клиенту обрабатывать запросы к базе данных в пакетном режиме, а значит, без использования индексов columnstore. Она эффективнее использует ресурсы процессора при исполнении аналитических запросов, а потому ее рекомендуют использовать в случаях, когда большая часть работы состоит из аналитических запросов, или когда рабочая нагрузка зависит от процессора. Стоит отметить, что такая возможность также была и раньше, но только в случае, когда запрос включал операции с индексами columnstore. Однако приложения могли применять функции, которые не поддерживают индексы columnstore, и поэтому могли не работать в пакетном режиме. Начиная с версии 2019 г. пакетный режим доступен для запросов, включающих операции как с индексами rowstore, так и с columnstore.

Встраивание скалярных пользовательских функций – это функция, появившаяся в последней версии Microsoft SQL, в запросе автоматически преобразует скалярные функции, определяемые пользователем, в реляционные выражения, что способствует значительному увеличению производительности запросов, использующие скалярные пользовательские функции.

Отложенная компиляция табличных переменных – данная опция позволяет оптимизировать план и повысить производительность запросов, использующих табличные переменные. Этот эффект достигается тем, что при оптимизации и первичной компиляции инструкций используются оценки кратности, вычисляемые исходя из фактического количества строк табличной переменной.

Заключение

Таким образом, версия SQL Server 2019 (15.x), выпущенная в октябре 2019 г., основана на предыдущих версиях и расширяет SQL Server как платформу, предоставляя клиенту больший выбор типов данных, языков разработки, локальных или облачных сред и операционных систем. Она также является более гибкой, поскольку поддерживает новые языки программирования, и кроссплатформенной, широко используемой не только под Windows, но и под Linux.

Стоит признать, что за последнее время Microsoft SQL Server значительно изменился, приобретя новый функционал, предоставив клиенту больше возможностей, а также пройдя

путь от небольшой СУБД до мощной платформы, используемой на всех крупных проектах. С очередным выпуском SQL Server становится все больше единым центром управления данными, учитывая все современные требования по работе с различными форматами данных и будучи широко используемым при построении платформы интеграции, хранении, обработке и анализе любых данных.

Библиография

1. Новые возможности SQL Server 2019 (15.x) [online]. – [дата обращения 02.03.2023], Доступно: https://docs.microsoft.com/ru-ru/sql/sql-server/what-s-new-in-sql-server-ver15?view=sql-server-ver15
2. Что нового в Microsoft SQL Server 2019 – обзор новых возможностей [online]. – [дата обращения 02.03.2023], Доступно: https://info-comp.ru/overview-new-features-ms-sql-server-2019
3. Первой CTP редакция SQL Server 2019 [online]. – [дата обращения 02.03.2023], Доступно: https://habr.com/ru/post/424441/
4. 10 причин перейти на Microsoft SQL Server 2019 [online]. – [дата обращения 02.03.2023], Доступно: https://softline.ru/about/blog/10-prichin-pereyti-na-microsoft-sql-server-2019

КЛАССИФИКАЦИЯ БАЗ ДАННЫХ

Олег БОГАЧ

Департамент Программной Инженерии и Автоматики, группа TI-195, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Олег БОГАЧ, e-mail: bogaci.oleg@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *в данной статье описывается классификация баз данных, а также области их применения*

Ключевые слова: *базы данных, классификация, фактографические, документальные, реляционные, иерархические, сетевые*

Введение

БД — представляет собой большие объемы информации, над которыми пользователь может производить необходимые операции: дополнять, удалять, обрабатывать и т.д [1]. Также информация, находящаяся в базе , имеет одно важное свойство – ее можно сгруппировать. К примеру, данные о населении можно организовать по полу, возрасту или даже цвету глаз. Такая группировка предоставляет нам возможность автоматической работы с данными с помощью СУБД. Перейдем к классификациям баз данных [2]

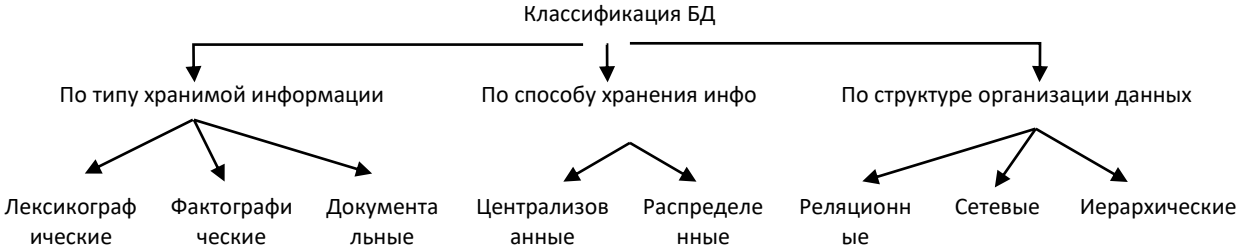


Рисунок 1. Классификации баз данных

Фактографические БД – это БД, которые содержат информацию, относящуюся непосредственно к предметной области. Фактографические БД в основном используют в экономике и управлении: социально-демографические БД; жилищно-коммунальные БД; финансово-экономические БД.

Лексикографические БД – это разнообразные словари (толковые, многоязычные), глоссарии, кодификаторы, своды, рубрикаторы, которые применяются как справочники вместе с фактографическими БД.

БД, хранящие документы, сгруппированные (организованные) по конкретным, заданным особенностям, определяются, как **документальные БД**.

Документальные БД включают с себя различные документы, классифицированные по определенным свойствам.

Говоря о документе, мы имеем в виду либо его адрес в памяти, либо сам текстовый документ. Самым ярким примером документальных баз данных стали базы содержащие в себе нормативные документы различных структур, например, справка по форме №xx.

Интегрированные или централизованные БД дают право на доступ к информации определенной группе пользователей. Данный вид доступа может предоставляться, как группе пользователей одновременно, так и параллельно.

Распределительные БД полностью копируют интегрированные БД, но в отличии от них имеют возможность находиться на разных устройствах физически, но считаться одним целым с точки зрения логики.

Рисунок 2. БД по методу хранения данных

Структурированные БД – заранее проектируется и создается прототип, в соответствии с которым будут хранится находящиеся там данные.

Реляционная база данных – база данных основанная на “relations”, то есть на связях, отношениях и зависимостях. Самая популярная и самая приближенная к алгебре модель баз данных. Если есть информация, которую можно сгруппировать то выбор реляционных БД самый очевидный. Ее основой служат таблицы, а связи происходят по «доменам», «атрибутам», «кортежам».

Иерархическая модель данных – все данные представлены с помощью иерархической структуры, «родителей» и «наследников», данные распределены по разным уровням.

Сетевая модель данных – так же как. распределительные БД полностью копируют интегрированные, **сетевые модели данных** копируют **иерархические.** Разница заключается лишь в том, что в иерархических структурах «потомок» должен иметь только одного единственного предка, а в сетевой структуре данных у потомка одновременно имеется любое количество предков.

Достоин упоминания еще один вид баз данных который не классифицируется.

Сверхбольшая база данных — это база данных, которая занимает чрезвычайно большой объём на устройстве физического хранения.

Количественное определение понятия «чрезвычайно большой объём» меняется во времени [3].

1997 год - Knight Ridder’s DIALOG 7 терабайт
2001 год – 10,5 терабайт
2003 год – 25 терабайт
2005 год ~ 100 терабайт
2006 год – GOOGLE 850 терабайт

2010 год – петабайты

2014 год – Facebook 300 петабайт / Google 10-15 экзабайт

В целом, по оценкам компании IDC, суммарный объём данных «цифровой вселенной» удваивается каждые два года и изменится до ~70 зеттабайт в 2023 году.

Заключение:

Рассмотрев различные типы баз данных, можно сделать вывод о том, что их разнообразие продолжает расти. На основе текущих тенденций можно предположить, что в будущем могут появится новые типы баз данных, такие как:

a) Базы данных на основе блокчейн-технологий. Блокчейн делает невозможным вмешательство злоумышленника в данные в сети без разрыва цепи.
b) Квантовые базы данных: такие базы данных могут использоваться для решения сложных задач, которые невозможно решить с помощью классических баз данных. Квантовые базы данных могут обеспечивать более быструю обработку данных и анализ, что может быть полезно в различных областях, включая научные исследования, бизнес-аналитику, медицину и другие. В этот момент такая концепция может показаться очень расплывчатой и ярко образной. Однако это вполне выполнимо, поскольку исследователи уже реализовали квантовые фильтры Блума и квантовые калибровочные графы.
c) Базы данных на основе искусственного интеллекта: Одним из направлений развития баз данных может стать использование искусственного интеллекта для оптимизации процессов хранения и обработки данных. Такие базы данных могут использоваться для анализа больших объемов информации, определения тенденций и паттернов, прогнозирования будущих событий и др.

Однако, несмотря на то, что различные типы баз данных предоставляют различные возможности и преимущества, выбор подходящего типа базы данных все еще остается важным вопросом для бизнеса и организаций. Правильно выбранная база данных может обеспечить более эффективную и точную обработку данных, а также обеспечить лучшие результаты при анализе данных.

Таким образом, при выборе типа базы данных следует учитывать конкретные потребности и требования организации, а также будущие возможности для масштабирования и расширения базы данных в соответствии с ростом бизнеса и изменением рыночных условий.

Библиография

1. Студенческий научный форум. - [online]. – [дата обращения 02.03.2023], Доступно: https://scienceforum.ru/2021/article/2018024159
2. Классификация баз данных - [online]. – [дата обращения 02.03.2023], Доступно: https://webonto.ru/klassifikatsiya-baz-dannyih/
3. БД - [online]. – [дата обращения 02.03.2023], Доступно: https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85

SQL SERVER BUSINESS INTELLIGENCE

Максим ГОДУНОВ

Департамент Программной Инженерии и Автоматики, Группа TI-196, Факультет Вычислительной Техники и Микроэлектроники, Технический Университет Молдовы, Кишинёв, Республика Молдова

Автор корреспондент: Максим ГОДУНОВ, e-mail: godunov.maxim@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Статья посвящена рассмотрению SQL Server Business Intelligence. Данная статья представляет обзор функциональных возможностей, которые позволяют пользователям собирать, анализировать и визуализировать данные для принятия более осознанных бизнес-решений.*

Ключевые слова*: SQL Server, Business Intelligence, данные, Analysis Service*

Введение

Выбор подходящего средства бизнес-аналитики может оказывать огромное влияние. Вы можете ознакомиться с дополнительными сведениями о различных предложениях Майкрософт и выбрать то из них, которое наилучшим образом соответствует вашим потребностям.

SQL Server Business Intelligence - программное обеспечение, созданное для помощи менеджеру в анализе информации о своей компании и её окружении. Большинство инструментов Business intelligence применяются конечными пользователями для доступа, анализа и генерации отчетов по данным, которые чаще всего располагаются в хранилище, витринах данных или оперативных складах данных. Разработчики приложений используют BI-платформы для создания и внедрения BI-приложений, которые не рассматриваются как BI-инструменты [1].

С помощью бизнес-аналитики (BI) от Майкрософт любая компания может перевести свои данные в умные и интуитивно понятные визуализации, которые помогут принимать более обоснованные решения.

SQL Server

SQL Server - это мощная система управления реляционными базами данных, разработанная компанией Microsoft. Эта платформа предоставляет широкий спектр инструментов для создания, управления и анализа данных, которые могут быть использованы в различных областях бизнеса и технологий.

Существует несколько инструментов, которые могут использоваться для управления базами данных SQL Server, включая:[2]

- SQL Server Management Studio (SSMS) - это интегрированная среда разработки, которая позволяет управлять базами данных, создавать запросы, настраивать безопасность и многое другое;
- SQL Server Data Tools (SSDT) - это набор инструментов для разработки баз данных SQL Server, который включает в себя функции для создания, изменения, тестирования и развертывания баз данных;
- SQL Server Configuration Manager - это инструмент для настройки и управления различными службами SQL Server, включая настройку сетевых протоколов и параметров безопасности;

— SQLCMD - это утилита командной строки, которая позволяет пользователям выполнить скрипты SQL и управлять базами данных.

SQL Server предоставляется в нескольких версиях с различными функциональными возможностями и ценами, чтобы удовлетворить потребности различных пользователей. Некоторые из доступных версий включают: [1]

— **Enterprise**: предназначен для больших компаний с высокими требованиями к данным, хранилищам и интернет-доступным базам данных. Включает все возможности SQL Server и обладает наибольшей стоимостью лицензии;
— **Standard**: разработан для малых и средних предприятий, поддерживает электронную торговлю и хранение данных;
— **Workgroup**: подходит для малых компаний, не имеет ограничений по размерам и числу пользователей, может использоваться как внутренняя база данных для небольших веб-серверов или отделений;
— **Express**: бесплатный вариант с минимальным набором функций, ограничениями по размеру базы данных и числу пользователей. Может служить заменой базы данных Access.

Business Intelligence

Бизнес-аналитика (BI) – это комплекс подходов, методов и инструментов, используемых для анализа, обработки и преобразования больших объемов данных, полученных из различных источников внутри и вне компании. Целью бизнес-аналитики является выявление скрытых закономерностей, трендов и возможностей, чтобы помочь руководителям и принимающим решения сотрудникам осуществлять обоснованный выбор в повседневной деятельности и стратегическом планировании [1]

Технологии бизнес-аналитики обеспечивают текущие, исторические и прогнозные представления внутренне структурированных данных для продуктов и отделов, обеспечивая более эффективное принятие решений и стратегическое оперативное понимание с помощью таких функций, как онлайн-аналитическая обработка (OLAP), отчетность, прогнозная аналитика, интеллектуальный анализ данных текста, сравнительный анализ и Управление эффективностью бизнеса (BPM).

BI также включает в себя определенные методологии и процедуры для реализации таких интерактивных методов сбора информации, в том числе:

— Определение команд интервью;
— Исследовательские организации;
— Отбор и подготовка интервьюируемых;
— Разработка вопросов для собеседования;
— Расписание и последовательность собеседований.

SQL Server Business Intelligence

SQL Server Business Intelligence (BI) - это серия инструментов для преобразования необработанных данных в информацию, которую компании могут использовать для принятия решений [3]. Ранее она включала IDE Business Intelligence Development Studio (BIDS) в предыдущих версиях Microsoft SQL Server, но после выпуска Visual Studio 2010 была заменена на SQL Server Data Tools - Business Intelligence.

Раньше SQL Server использовал Business Intelligence Development Studio (BIDS), чтобы позволить разработчикам преобразовывать данные, но с Visual Studio 2010 его заменила SQL Server Data Tools - Business Intelligence. Это позволяет пользователям Visual Studio быстро создавать программы бизнес-аналитики и тестировать их [4].

Инструменты бизнес-аналитики позволяют разработчикам выполнять операции извлечения, преобразования и загрузки (ETL) данных из хранилищ данных. Приложения

включают показатели производительности, аналитику, корпоративную отчетность и управление знаниями.

Одним из главных преимуществ SQL Server BI является его способность предоставлять полноценные решения по управлению и анализу данных. Используя интегрированные компоненты, организации могут оптимизировать свои процессы управления данными, повысить качество данных и получить более глубокие понимание своих данных. Кроме того, SQL Server BI очень масштабируем и может использоваться организациями любого размера и отрасли.

Возможности Microsoft SQL Server Business Intelligence

- Управление данными - SQL Server BI предоставляет инструменты для загрузки, интеграции и управления данными из различных источников. Это позволяет организациям получить единую и консолидированную версию своих данных, которую можно использовать для принятия более обоснованных решений;
- Аналитика данных - SQL Server BI предоставляет инструменты для анализа данных, которые помогают пользователям получать ценные инсайты и понимать тенденции в своих данных. Эти инструменты включают в себя SSAS для создания многомерных моделей данных и Power BI для создания интерактивных отчетов и визуализаций;
- Отчетность - SQL Server BI предоставляет инструменты для создания интерактивных отчетов, панелей мониторинга и карточек показателей, которые позволяют пользователям легко отслеживать ключевые показатели производительности и принимать осознанные решения на основе данных;
- Управление мастер-данными - SQL Server BI включает Master Data Services (MDS), инструмент для управления мастер-данными, который позволяет организациям управлять, согласовывать и синхронизировать свои мастер-данные;
- Обеспечение качества данных - SQL Server BI включает Data Quality Services (DQS), инструмент для управления качеством данных, который помогает организациям повысить качество своих данных и оптимизировать свои процессы управления данными.

Эти и другие возможности SQL Server Business Intelligence позволяют организациям получить максимальную выгоду из своих данных, повысить эффективность своих бизнес-процессов и принимать более осознанные решения на основе данных.

Microsoft SQL Server Business Intelligence (MSBI) представляет собой платформу бизнес-аналитики, которая обеспечивает инструменты для извлечения, трансформации и загрузки данных (ETL), анализа данных и создания отчетов. Некоторые из инструменты MSBI включают:

- Integration Services (SSIS) - Инструмент для ETL-процессов, позволяющий импортировать, трансформировать и экспортировать данные из различных источников;
- Analysis Services (SSAS) - Сервер аналитической обработки, позволяющий создавать кубы данных и многомерные модели для анализа больших объемов информации;
- Reporting Services (SSRS) - Инструмент для создания, публикации и распространения отчетов в различных форматах;
- Power BI - Инструмент для визуализации данных, позволяющий создавать дашборды, отчеты и интерактивные графики;
- Master Data Services (MDS) - Инструмент для управления мастер-данными, обеспечивающий одну версию правды для ключевых бизнес-показателей;
- Data Quality Services (DQS) - Инструмент для управления качеством данных, позволяющий оценивать, улучшать и обеспечивать точность данных в ETL-процессах;
- StreamInsight - Инструмент для обработки потоков данных в реальном времени, позволяющий анализировать и принимать решения на основе быстро меняющейся информации.

Преимущества пользования SQL Server Business Intelligence указаны в таблице 1.

Таблица 1

Преобразование сложных данных	**Модернизация отчетности**	**Использование надежной платформы Бизнес-аналитики**
Создание и работа с эффективными моделями данных, которые легко можно понять и анализировать	Создание и работа с интерактивными отчетами Power BI, которые иллюстрируют главные показатели производительности.	Воспользуйтесь преимуществами масштабируемости, доступа к защищенным данным и знакомых инструментов.

Сервис Microsoft Azure SQL Databases

Microsoft Azure SQL Databases — это облачное решение для хранения и управления данными, основанное на платформе Microsoft SQL Server. С помощью Azure SQL Databases, пользователи могут создавать, масштабировать и управлять базами данных на облачной платформе Azure [5].

Azure SQL Databases предоставляет множество возможностей для управления данными, включая:

— Высокая доступность - Azure SQL Databases обеспечивает высокую доступность и гарантирует непрерывность работы приложений, даже в случае отказа аппаратного обеспечения или сетевых проблем.
— Масштабируемость - Azure SQL Databases позволяет масштабировать базы данных в зависимости от потребностей организации, увеличивая или уменьшая количество ресурсов в режиме реального времени.
— Безопасность - Azure SQL Databases предоставляет множество функций для защиты данных, включая шифрование данных в покое и в движении, аутентификацию и авторизацию на основе ролей, а также функции отслеживания и аудита.
— Интеграция - Azure SQL Databases интегрируется с другими облачными сервисами Azure, такими как Azure Data Factory и Power BI, что позволяет пользователям создавать более эффективные бизнес-процессы.

С помощью Azure SQL Databases, организации могут существенно снизить свои затраты на инфраструктуру и управление базами данных, обеспечить безопасность и надежность своих данных, а также получить ценные инсайты и аналитику из своих данных.

Принцип работы Azure SQL указан на рисунке 1.

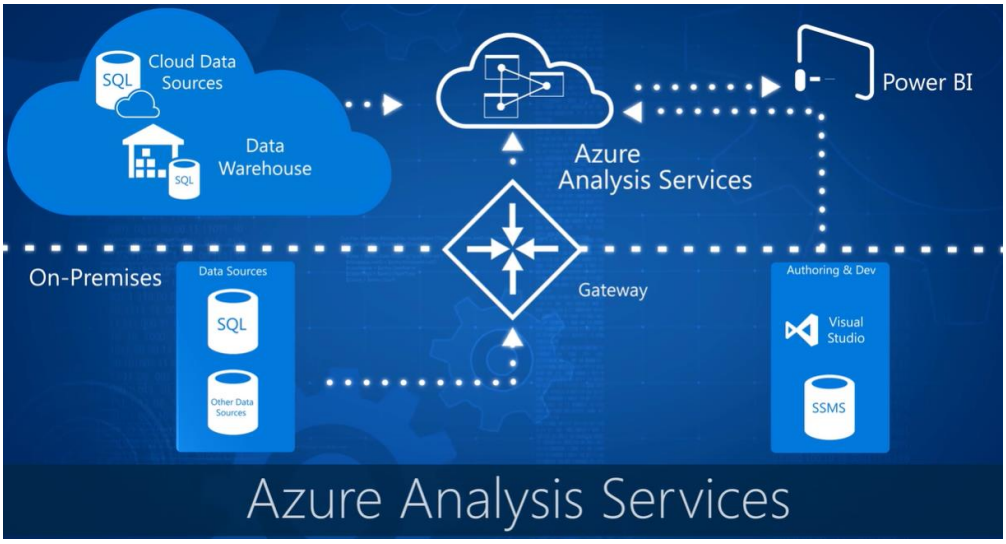


Рисунок 1 – Принцип работы Azure SQL Databases.

Заключение

В статье было рассмотренно основные инструменты, входящие в пакет Microsoft SQL Business Intelligence, а также привели пример облачного решения для хранения и управления данными - Azure SQL Databases. Эти инструменты не только помогают организациям управлять и анализировать данные, но и повышают эффективность и конкурентоспособность компании в целом.

В современном мире все больше растет конкуренция в сфере информационных технологий. Каждый устоявший на рынке или же новый сервис стремится привлекать своих пользователей надежным функционалом. В целом, MSBI предоставляет комплексные возможности для управления данными и анализа информации, что позволяет компаниям принимать обоснованные решения и повышать эффективность бизнес-процессов.

Компания Microsoft ежегодно выпускает различного рода дополнения и модификации, которые поддерживают конкурентоспособность и функциональность своих продуктов на рынке SQL Server со своими интегрированными в него сервисами, такими как Azure Analysis Service, Business Intelligence технически удовлетворяет потребности всех пользователей информационных технологий.

Для будущего поколения инструменты SQL Business Intelligence и Azure SQL Databases будут иметь большое значение. С увеличением объема данных, производительность и доступность этих инструментов станут еще более критическими. Кроме того, развитие технологий и появление новых возможностей в области искусственного интеллекта, машинного обучения и аналитики данных будет продолжаться, что позволит использовать данные еще более эффективно.

Библиография

1. Что такое Business Intelligence? Обзор BI систем. – [online] [дата обращения 02.03.2023], Доступно: https://www.clouderp.ru/tags/business_intelligence/
2. Why a SQL Server BI platform?– [online] [дата обращения 02.03.2023], Доступно: https://www.microsoft.com/en-us/sql-server/sql-business-intelligence
3. SQL Server Business Intelligence – [online] [дата обращения 02.03.2023], Доступно: st-martin.ru
4. Monitor Azure SQL Database using Azure SQL Analytics. – [online] [дата обращения 02.03.2023], Доступно: https://docs.microsoft.com/en-us/azure/azure-monitor/insights/azure-sql
5. SQL Server 2012 Business Intelligence Edition– [online] [дата обращения 02.03.2023], Доступно: https://www.mssoft.ru/Makers/Microsoft/SQL_Server_2012_Business_Intelligence/ https://st-martin.ru/

INTERBASE — РЕЛЯЦИОННАЯ СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

Вадим ПОТРЫМБА

Департамент Программной Инженерии и Автоматики, TI-196, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Молдова

Автор корреспондент: Вадим ПОТРЫМБА, e-mail: potrimba.vadim@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *InterBase - это полнофункциональная, высокопроизводительная, сверхбыстрая, шифруемая, масштабируемая, реляционная и встраиваемая многоплатформенная база данных SQL с многомерной защитой данных, аварийным восстановлением и синхронизацией изменений для разработчиков, которые хотят внедрить недорогую защищенную базу данных с нулевым администрированием в свои кросс-платформенные приложения. В этой статье будет рассказано о структуре базы данных InterBase, её возможностях и особенностях.*

Ключевые слова: *база данных, реляционная СУБД, безопасность, лёгкость использования.*

Описание InterBase

InterBase - одна из первых когда-либо созданных реляционных баз данных корпоративного уровня. С момента своего создания он основан на превосходной архитектуре управления версиями, которая до сих пор не имеет аналогов во многих популярных базах данных. Это также была первая СУБД, реализовавшая управление параллелизмом нескольких версий (MVCC). MVCC упоминается как архитектура нескольких поколений в InterBase.

InterBase разрабатывается и поддерживается компанией Embarcadero Technologies, которая разрабатывает и продает RAD Studio, Delphi, C++Builder и InterBase. Сервер RAD работает на базе InterBase. Embarcadero Technologies является подразделением Idera, Inc[1].

InterBase является лауреатом премии Total Telco (Рисунок 1) в области интернета вещей за “самое инновационное использование данных” благодаря своей запатентованной технологии масштабируемых представлений изменений, которая обеспечивает самый быстрый, простой и безопасный способ отслеживания изменений в сохраненных данных. Переподключайтесь в любое время, чтобы получать персонализированные изменения на уровне поля для любого пользователя / устройства, используя стандартный SQL, и быстро обновляйте удаленные кэшированные данные с помощью небольших контекстно-зависимых дельт!

Рисунок 1. Награда Total Telco

Почему InterBase?

– Нулевое администрирование с крошечным размером

InterBase идеально подходит для ISV, OEM-производителей и VAR, которым требуется база данных с компактными размерами памяти и диска, но без ущерба для сложных функций для поддержки критически важных бизнес-приложений. В отличие от многих баз данных,

требующих дорогостоящей поддержки и обслуживания, Интерфейс не требует администрирования. Простая установка и небольшие размеры делают InterBase невероятной базой данных для встраивания в ваши приложения.

– Внедрить Один Раз – Развернуть Везде

Внедряйте InterBase в свое приложение, и по мере перемещения приложения между платформами InterBase будет перемещаться вместе с вами. InterBase доступен для Android, iOS, Windows, OS X, Linux и macOS, что обеспечивает невероятную гибкость развертывания. InterBase поддерживает подключение между версиями InterBase и между операционными системами.

– Сверхбыстрая производительность

InterBase - одна из самых быстрых доступных собственных баз данных. Используя встроенную симметричную многопроцессорную обработку (SMP), InterBase может использовать преимущества современных многоядерных процессоров и многопроцессорных систем без увеличения затрат. В ряде областей были добавлены улучшения и оптимизации производительности, такие как восстановление баз данных, пакетные обновления и запросы. InterBase использует архитектуру нескольких поколений, обеспечивающую уникальные возможности управления версиями, которые обеспечивают высокую доступность данных для обработки транзакций и поддержки принятия решений одновременно.

– Интеграция с лучшими инструментами разработки приложений

InterBase поддерживает все основные парадигмы разработки, включая Java, C, C++, .NET, Delphi и PHP. Он тесно интегрируется с C++Builder, RAD Studio и Delphi, так что при развертывании ваших проектов RAD на любой платформе InterBase автоматически развернется вместе с ним. Если вы являетесь разработчиком Visual Studio, вы можете работать с InterBase через ADO. Сетевые или ODBC-разъемы для легкого доступа к вашим данным [2].

1. Файл базы данных

В InterBase база данных может состоять не только из одного файла, но и из нескольких. Программист присваивает имя файлу базы данных, а расширение у файла — gdb. В ранних версиях InterBase размер файла базы данных не превышал 2—4 Гбайт, начиная с версии 6.5 размер файла не должен превышать 32 Гбайт.

Файл базы данных представляет собой контейнер, внутри которого хранятся страницы различного типа и назначения. Типовой состав и количество страниц определяются сервером InterBase. Все страницы имеют одинаковый размер. Размер страницы определяется один раз при создании базы данных. В процессе эксплуатации базы данных размер страницы изменить нельзя. Единственная возможность изменить размер страницы базы данных — восстановление базы данных из резервной копии (подробнее см. гл. 12). Размер страницы фиксированный — 1024, 2048, 4096 и 8192 байта. Начиная с версии InterBase 6.5 и выше допускается размер страницы 16 384 байта. За одно обращение к базе данных сервер InterBase читает одну страницу целиком. Во время сеанса связи пользователя базы данных с базой банных обмен данными выполняется через буфер (Database cache). Размер буфера определяется при создании базы данных, по умолчанию размер буфера установлен в 75 страниц.

Всего предусмотрено 11 типов страниц (Таблица 1). Каждый тип имеет свой идентификатор (порядковый номер) [3].

Страница любого типа начинается со строки заголовка, которая содержит информацию о типе страницы и номере следующей страницы того же типа. Далее следует специфическая информация.

Таблица 1.

Назначение страниц.

Идентификатор	Определение в ods.h	Обозначение	Назначение
0	pagundefined	VP	Неопределенный тип
1	pagheader	HP	Заголовок (общие сведения) базы данных
2	pagjpages	SIP	Учет распределения страниц базы данных
3	pagtransactions	TIP	Учет транзакций
4	pag_pointer	PP	Учет указателей на другие страницы
5	pag_data	DP	Записи таблицы
6	pag_root	IRP	Учет вершин индексов
7	Pagjndex	IP	Страница индексов
8	pag_blob	BP	Массивы данных в двоичном коде
9	Pagids	GP	Учет генераторов
10	Pagjog	LP	Не используется

https://studref.com/329182/informatika/fizicheskaya_struktura_bazy_dannyh_interbase

Заключение

Система управления базами данных (СУБД), и InterBase в частности, предназначена для хранения данных и управления этими данными. По запросу пользователя СУБД размешает данные в базе данных, преобразуя их в свой внутренний формат. Понятно, что каждая СУБД имеет свой собственный формат представления данных. Также по запросу пользователя СУБД извлекает данные, преобразует их к виду, удобному для пользователя, и предъявляет их на экран [4].

InterBase – это:

– безопасность:
– кроссплатформенное шифрование;
– отдельный вход в систему безопасности;
– снижение подверженности риску;
– безопасность пользователей на основе ролей;
– зашифрованные резервные копии.
– уверенность в себе:
– независимая работа;
– почти нулевое техническое обслуживание;
– быстрое восстановление после сбоя;
– аварийное восстановление;
– лёгкость:
– небольшая площадь занимаемой площади;
– быстрая установка;
– быстрое восстановление;
– более быстрые данные.

Библиография

1. Электронная библиотека им. Н.Э. Баумана. InterBase. [online]. [27.12.2021] – доступна: https://ru.bmstu.wiki/InterBase
2. InterBase. About InterBase. [online]. [22.12.2021] –доступна: https://interbase.com/
3. Студенческие реферативные статьи и материалы. Физическая структура базы данных InterBase. [online]. [24.12.2021] – доступна: https://studref.com/329182/informatika/fizicheskaya_struktura_bazy_dannyh_interbase
4. База данных InterBase для начинающих. [online]. [07.01.2022] – доступна: https://www.ibprovider.com/rus/documentation/interbase.html

INFLUENȚA VOLUMULUI DE DATE ASUPRA TIMPULUI DE EXECUȚIE A DIFERITOR TIPURI DE INSTRUCȚIUNI ÎN SQL SERVER

Irina GRIGORAȘ

Departamentul Ingineria Software şi Automatică, gr. TI-205, Universitatea Tehnică a Moldovei, Chişinău, Republica Moldova

Autorul corespondent: Grigoraș Irina, irina.grigoras@isa.utm.md

Coordonator științific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *În articol este descris un experiment asupra categoriilor de instrucțiuni cel mai des utilizate în limbajul SQL. A fost creat un cod de generare a unei baze de date cu 3 tabele. Un tabel a fost populat cu date de 10.000, 100.000, 1.000.000 și 5.000.000, după care secvențial au fost executate interogări simple, fără index, cu index cluster și non-cluster, și s-a verificat timpul de execuție pentru fiecare dintre acestea. Conform rezultatelor s-au demonstrat afirmații din diferite surse despre comenzile SELECT, UPDATE, DELETE.*

Cuvinte cheie: *DDL, DML, DQL, index, cluster, non-cluster, timp de execuție.*

Introducere

În această lucrarea voi încerca să demonstrez informații de pe internet despre timpul de execuție a anumitor comenzi în SQL efectuat în SQL Server. Au fost create tabele pentru proiect cărora le-au fost alocate diferite volume de date, unde a fost măsurat timpul de execuție la cazuri anumite.

În special m-am concentrat pe comenzile DELETE și UPDATE utilizate pentru tabele care conțin sau nu index cluster/non-cluster. Conform datelor obținute am efectuat concluzii și le-am comparat cu surse de pe internet.

1. Instruncțiuni DDL

Limbajul de definire a datelor (DDL, Data Definition Language) este un limbaj orientat la structura bazei de date. DDL permite crearea, modificarea şi suprimarea obiectelor bazei de date (relaţiilor, atributelor, constrângerilor, viziunilor, indecşilor,...). El permite, de asemenea, definirea domeniilor de date (numere, secvenţe de caractere, date temporale.) şi adăugarea constrângerilor asupra valorilor datelor [1].

În cadrul experimentului au fost create 3 tabele conectate între ele pentru o școală auto, un tabel reprezentând contractele clienților ce include id-ul grupelor din care fac parte, iar grupurile conțin id-ul cursului pe care îl urmează grupul dat, exemplu categoria A, B etc.

În tabelul 1 este prezentat timpul de execuție față de numărul de clienți a efectuării instrucțiunilor DROP – pentru ștergerea bazei dacă există, CREATE – crearea tabelelor, ALTER – adăugarea constrângerilor și execuția unor funcții cu WHILE, IF pentru generarea unor date.

Tabelul 1

Crearea tabelelor, dependențelor și generarea datelor cu INSERT

Timp	Numărul de clienți
4 sec	10.000
31 sec	100.000
4 min 33 sec	1.000.000
18 min 46 sec	5.000.000

Conform datelor din Tab. 1 am constatat ca există o legitate de creștere a timpului în dependență de numărul de clienți. Aceasta este că timpul crește cu 8n la numărul clienților înmulțit cu 10, unde n este timpul obținut la calculul anterior.

Tabelul 2

Demonstrații

Nr d/o	Calcul teoretic	Calcul experimental	Concluzie
1	la 10.000 clienți avem timp n = 4 sec, deci la 100.000 timpul ar trebui sa fie 8n = 8*4 = 32 (sec)	31 sec	31 ~ 32 Trecut
2	100.000 clienți => n = 31 sec, la 1.000.000 timpul ar trebui sa fie 8n = 8*31 = 248 (sec)	4 min 33 sec = 273 sec	248 ~ 273 Trecut
3	1.000.000 clienți => n = 273 sec, la 1.000.000 timpul ar trebui sa fie 8n = 8*273 = 2184 (sec)	Consideram că 18 min 46 sec e jumătate și în total va fi 37 min 52 sec = 2272 sec	2184 ~ 2272 Trecut

Deși după cum se vede în Tab. 2 există puțină neconcordanță a datelor, dar asta poate fi provocată de starea calculatorului și capacitatea de executare a interogării. Trebuie de menționat că în cazul dat legitatea de creștere s-a dovedit a fi 8n, în alte cazuri ea poate să diferă acest lucru fiind influnțați de o mulțime de factori.

2. Instruncțiuni DQL

Limbajul de interogare a datelor (DQL, Data Query Language) include instrucţiuni SQL care permit extragerea datelor din baza de date. Este limbajul în care utilizatorii pot formula cereri la baza de date. Deşi reprezintă o parte foarte importantă a limbajului SQL, DQL este format din instrucţiuni care folosesc o singură comandă: SELECT [1].

Timpul de execuție pentru o instrucțiune SELECT cu sau fără JOIN în SQL Server poate varia în funcție de o serie de factori, cum ar fi dimensiunea tabelelor, numărul de rânduri care trebuie preluate, complexitatea interogării și cât de încărcat e serverul.

În Tab. 3 se poate observa timpul de execuție pentru comanda SELECT cu și fără join și clauza WHERE.

Tabelul 3

Comanda SELECT

Timp de execuție (sec)	Cu join și clauza where (sec)	Numărul de clienți
0,35	0,1	10.000
0,44	0,15	100.000
0,50	0,35	1.000.000
59	18	5.000.000

Un motiv de ce datele diferă atât de mult constă în faptul că i-a timp pentru ca fiecare rând să fie afișat, deci întru-cât SELECT-ul simplu afișează toate liniile oricât de multe ar fi, iar cea cu JOIN și clauza WHERE datorită filtrării datelor, economisându-se timp la afișare.

O instrucțiune SELECT fără JOIN se va executa de obicei mai rapid decât o instrucțiune SELECT cu JOIN, mai ales dacă tabelul care este interogat este mare și condiția JOIN implică mai multe tabele. Acest lucru se datorează faptului că atunci când se utilizează un JOIN, SQL Server trebuie să combine rânduri din mai multe tabele, ceea ce poate fi o operație mai complexă și mai consumatoare de timp. Datorită faptului că s-a utilizat și clauza WHERE care a filtrat datele și astfel mai puține date fiind necesar de afișat am obținut acești timpi.

3. Instruncțiuni DML cu indecși cluster și non-cluster

Limbajul de manipulare a datelor (DML, Data Manipulation Language) constituie setul de comenzi referitor la actualizarea conţinutului bazei de date. DML permite introducerea a noi tupluri, modificarea tuplurilor, suprimarea tuplurilor nedorite din baza de date şi blocarea relaţiilor. Instrucţiunile principale sunt INSERT, UPDATE, DELETE, LOCK [1].

În SQL Server, index cluster pe o coloana este creat automat de constrângerea cheii primare. Protocolul specifică că fiecare tabel va avea un singur index cluster. Similar cu modul în care este utilizat un dicționar, un index cluster este utilizat pentru a defini ordinea, sortarea tabelului sau organizarea datelor în ordine alfabetică. Când un tabel are un asemenea index, tabelul este numit tabel cluster. Dacă un tabel nu are index cluster, rândurile sale de date sunt stocate într-o structură neordonată numită heap [2].

Indicii non-cluster au o structură separată de rândurile de date. Un index non-cluster conține valorile cheii de index non-cluster și fiecare intrare de valoare cheie are un pointer către rândul de date care conține valoarea cheie [3].

Tabelul 4

Timpului de execuție a diferitor comenzi SQL pentru diferițe tipuri de indecși

Tip	Update (sec)	Delete (sec)	Numărul de clienți
1. Simplu	0,35	0,25	10.000
	0,94	0,34	100.000
	2,54	2,45	1.000.000
	20,23	15,11	5.000.000
2. Clustered	0,45	0,33	10.000
	0,87	0,47	100.000
	2,20	2,00	1.000.000
	14,46	20,03	5.000.000
3. Nonclustered	0,37	0,27	10.000
	0,40	3,00	100.000
	3,04	9,23	1.000.000
	20,34	47,68	5.000.000

Din Tabelul 4 se poate observa diferențele de comportament a instrucțiunilor UPDATE și DELETE la utilizarea lor în tabel ce nu conține un anumit tip de index și cel ce conține index cluster și non-cluster. După cum se vede la in tabelul ce conține index cluster întrodus prim Primary KEY, trimpul de execuție la ambele comenzi a fost mai rapid decât la acela fără nici un anumit tip de index.

Timpul necesar pentru executarea unei instrucțiuni UPDATE sau DELETE poate varia în funcție de un număr de factori, cum ar fi dimensiunea tabelului, numărul de rânduri afectate de instrucțiune și complexitatea interogării. Referitor la cazul relatat despre execuția mai rapida a comenzii UPDATE față de DELETE de mai sus la primele 2 tipuri, la al treilea tip, non-cluster index, se observă complet opusul, aici execuția la actualizare a luat practic ca și celelalte 2, dar în schimb la ștergere aceasta a luat mai mult timp.

Conform unor surse, executarea instrucțiunii UPDATE poate dura mai mult decât instrucțiunea DELETE dacă tabelul are un număr mare de rânduri și interogarea este complexă, lucru demonstrat în partea din tipul 1 și 2. Acest lucru se datorează faptului că instrucțiunea UPDATE trebuie să citească și să modifice fiecare rând afectat, în timp ce instrucțiunea DELETE trebuie doar să localizeze și să elimine rândurile.

Când se utilizează instrucțiunea de actualizare, SQL Server trebuie să modifice datele din rând și să actualizeze orice indici, ceea ce poate dura mai mult, mai ales dacă tabelul are un număr mare de rânduri sau dacă tabelul are indecși care trebuie actualizați. Pe de altă parte, atunci când se utilizează instrucțiunea de ștergere, SQL Server trebuie să elimine rândurile și să actualizeze indecșii, ceea ce poate fi mai rapid, mai ales dacă tabelul are un număr mare de rânduri, iar instrucțiunea de actualizare trebuie să actualizeze multe rânduri.

Concluzii

În lucrarea dată s-a experimentat timpul de execuție a diferitor comenzi în SQL Server efectuate asupra tabelelor cu diferite volume de date. S-a constatat că la crearea tabelelor s-a conturat o legitate care permite de a prognoza timpul de execuție la mărirea numărului de date inserate. Comanda SELECT, în general, oferă o execuție mai rapidă efectuându-se filtrări a datelor, datorită micșorării numărului de date necesare de a fi afișate. Cel mai lung timp la această comanda este afișarea datelor pentru utilizator.

Altă concluzie care s-a confirmat este că comanda UPDATE cu creșterea datelor incetinește mai rapid decât DELETE, astfel că aceasta are o execuție mai avantajoasă pentru aceleași date. În general, timpul de execuție obținut de SQL SERVER diferă de la un calculator la altul, și în timpul experimentării s-au obținut timpi diferiți. Am demonstrat infomații obținute din internet despre indecși cluster și non-cluster, și anume că la DELETE indexul non-cluster se execută mai greu decât cel cluster sau chiar fără. Deci, trebuie de decis cum avem de gând să utilizăm un tabel ca mai apoi să decidem ce instrucțiune sau index să utilizăm.

Referințe

1. V. COTELEA, *Algebra relațională și limbajul SQL*. Chișinău: Vizual Design, 2013.
2. Diference between Clustered and Non-clustered Index [online]. [accesat 26.02.2023]. Disponibil: https://byjus.com/gate/difference-between-clustered-and-non-clustered-index/
3. Clustered and nonclustered indexes described [online]. [accesat 01.02.2023]. Disponibil: https://learn.microsoft.com/en-us/sql/relational-databases/indexes/clustered-and-nonclustered-indexes-described?view=sql-server-ver16

БЛОКЧЕЙН

Николай ШАРАФУДИНОВ

Департамент Программная Инженерия и Автоматика, группа TI-196, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Молдова

Автор Корреспондент: Николай ШАРАФУДИНОВ, e-mail: sarafudinov.nicolae@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *цель статьи – определить, что такое блокчейн, из чего он состоит и где используется. Для этого в статье рассмотрены такие понятия как узел, хеширование и шифрование, технология DLT, консенсус и алгоритм консенсуса. А также в данной статье показано почему данная технология становится популярной в наше время.*

Ключевые слова: *блокчейн, хеш-дерево, хеширование, распределенный реестр, консенсус.*

Введение

Последние несколько лет технология распределенного реестра (distributed ledger technology, DLT), в частности блокчейн (blockchain), не только активно обсуждается, но и плотно внедряется в большинстве развитых стран во многих областях экономики. Многие организации ставят своей целью разобраться в возможном применении технологии. В данной статье будет описано что такое блокчейн, чем является алгоритм консенсуса, а также в каких сферах можно применить технологию блокчейн.

Что такое блокчейн?

Блокчейн — это система записи информации таким образом, чтобы ее было невозможно изменить, взломать или обмануть. Блокчейн, как следует из самого названия, представляет собой цепочку блоков. Каждый блок содержит в себе набор из совершенных в течение определенного периода времени (в биткоине это в среднем 10 минут) транзакций между пользователями.

Каждый последующий блок имеет временную метку и ссылку на предыдущий блок. Так же в блок заносится не сама транзакция, а 32-битное значение, обозначенное, как корень Меркла (merkle_root) рис. 1. Также блок содержит уникальный заголовок (block header) уникальность которого достигается за счет того, что каждый заголовок является значением хеш-функции SHA256, от информации, хранящейся в нем: список транзакций (корня Меркла), временная метка создания блока, версия текущего алгоритма, текущая сложности вычисления блока, nonce и заголовок предыдущего блока [1].

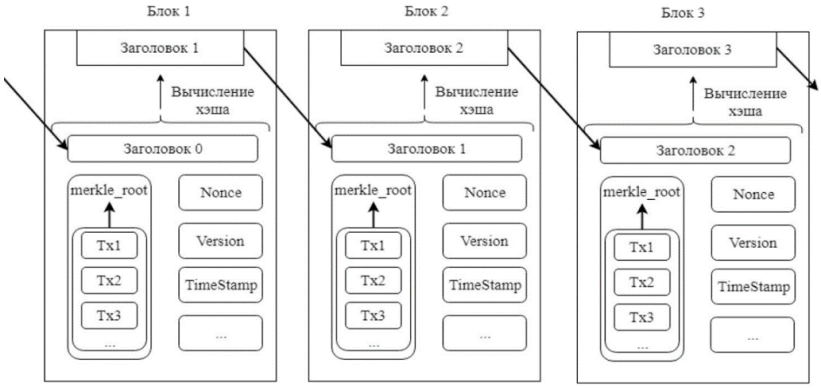


Рисунок 1. Blockchain

Узел в блокчейне

Узел — это любой компьютер, который подключен к блокчейну, проверяющий и подтверждающий транзакции, а также хранит копию блокчейна. Узлы обеспечивают безопасность блокчейна, даже при условии, когда узел отключен от сети «offline node», или же подключена к сети непостоянно то при подключении, данные узлы должны загрузить обновлённые копии реестра для синхронизации с сетью. Синхронизируя свои хранилища с данными о последних транзакциях, большее количество узлов делает внесение изменений невозможным, а также не оставляет хакеру шансов остаться незамеченным. Хакер не сможет удалить данные с множества различных узлов, что значит — вся информация в безопасности. Более того, узлы сортируются по их доступности и в соответствии с их состоянием, в сети или же нет, принимается решение непрерывно отправляет данные в сеть или же игнорировать узлы, не подключенные к сети.

Хеширование и Шифрование

Шифрование и временные метки в совокупности позволяют технологии блокчейн автоматически проверять неизменность этой постоянно увеличивающейся последовательности хеш-кодов. Эта операция не позволяет вставлять новые блоки не по порядку, делая невозможными изменение или фальсификацию данных транзакций.

Отличительной чертой распределенного реестра является активное использование криптографических методов. В первую очередь, речь идет о криптографически стойкой хеш-функции (hash-function SHA-256). Если процесс хеширования повторяется с точно такими же транзакциями, будут созданы точно такие же хеши. Это позволяет любому, кто использует блокчейн, проверить, что данные не были подделаны, потому что любое изменение в любой части данных приведет к совершенно другому хешу, влияющему на каждую итерацию хешей вплоть до корня в Merkle tree рис. 2.

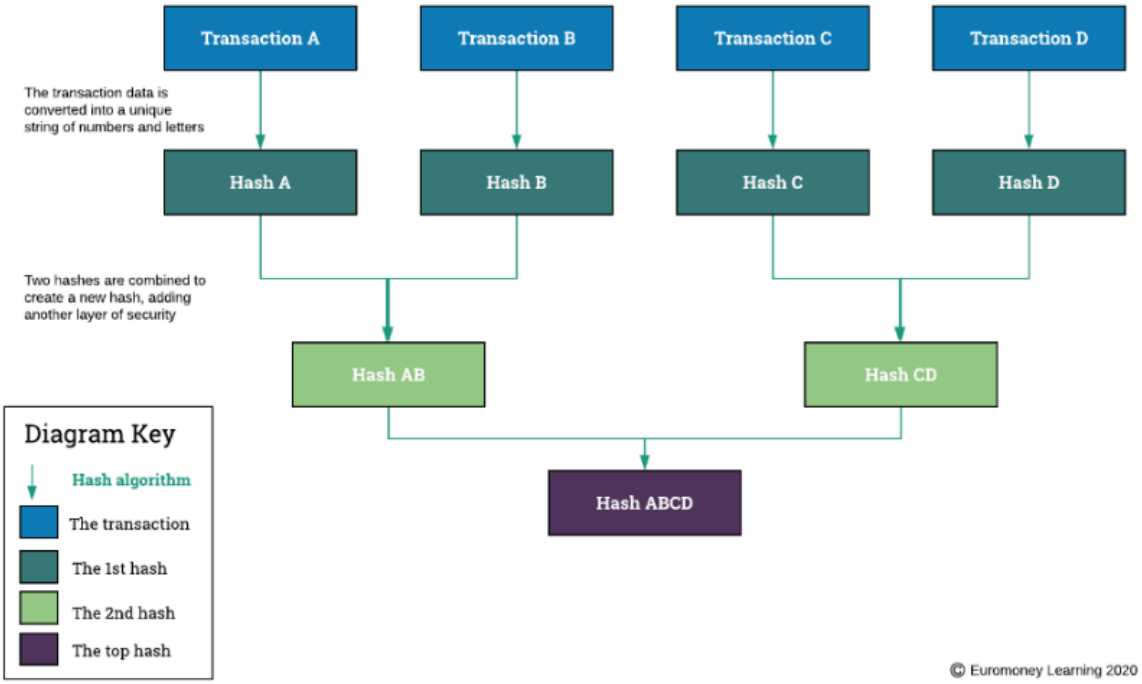


Рисунок 2. Хеш-дерево, Merkle tree

Merkle tree служат для значительного сокращения объема данных, необходимых для хранения, передачи по сети, путем объединения наборов хешированных транзакций в один корневой хеш. Поскольку каждая транзакция хешируется, затем объединяется и снова хешируется, окончательный корневой хеш по-прежнему будет иметь стандартный размер.

Технология распределенного реестра DLT (distributed ledger technology)

Несмотря на то, что на рынке существует большое количество DLT, все они состоят из одних и тех же строительных блоков: распределенного реестра, алгоритма консенсуса (для обеспечения идентичности всех копий реестра) и вознаграждения за участие в сети. DLT — это протокол, который обеспечивает безопасное функционирование децентрализованной цифровой базы данных [2]. DLT позволяет безопасно и точно хранить всю информацию с помощью криптографии. Как только информация сохраняется, она становится неизменной в базе данных и регулируется правилами блокчейна. Главными особенностями DLT являются:

- Хранение данных в одноранговой сети — у каждого узла равные права.
- Удаленность узлов — узлы с копиями баз данных удалены друг от друга географически и связаны друг с другом через интернет.
- Механизм синхронизации — узлы сообщают друг другу о любых изменениях, внесенных в реестр, проверяют эти изменения и вносят их в свою копию реестра.

Консенсус

Механизм консенсуса представляет собой некий компьютерный алгоритм, который лежит в основе распределенного реестра. Благодаря данному алгоритму децентрализованные узлы сети достигают согласия о текущем состоянии данных во всех блоках [3]. Алгоритм консенсуса необходим для проверки корректна ли транзакция. Другой важной задачей механизма консенсуса является разрешение конфликтов между некоторыми противоречащими транзакциями, проводимыми одновременно. Ситуация, когда два майнера одновременно сгенерировали подходящие блоки, вызывает раздвоение цепи блоков. Тогда главной цепочкой будет считаться та, ответвление которой будет быстрее продолжено. Множество алгоритмов консенсуса находятся еще в процессе создания. Наибольшую популярность приобрел механизм консенсуса Proof-of-Work (доказательство выполнения работы).

Механизм консенсуса Proof-of-Work (PoW)

Чтобы участвовать в проверке транзакции, участникам необходимо публично доказать проведенную работу. А именно решение криптографической задачи [4]. Решение криптографической задачи сводится к простейшему перебору миллионов комбинаций кода путем изменения динамического числа nonce, создающих доказательство «работы». В момент, когда майнер находит хеш, он отправляет хеш другим компьютерам сети на проверку. Другие участники не могут использовать его для создания блока так как ключ разгадки хеша принадлежит майнеру, что решил криптографическую задачу первым. Считается, что транзакция является окончательно проведенной, если от блока, в котором она учтена, цепь блоков продлена хотя бы на 6 блоков. В противном случае, если транзакция по передаче некоторого количества биткоинов оказалась в отвергнутой цепочке, она становится недействительной и автоматически вновь становится в очередь на подтверждение при генерации текущего блока. Особенность полученного хеша в его асимметрии — он достаточно сложен для нахождения майнеру, но при этом данный хеш легко проверить [5].

Формирование цены

Криптовалюты имеют ценность, потому что, они являются уникальными цифровыми активами, которые могут быть использованы для проведения безопасных, быстрых и анонимных транзакций без участия посредников.

Для сдерживания инфляции и для регулирования сложности работы по генерированию новых блоков, система автоматически анализируется время, затраченное на нахождения хеш кода, и раз в две недели настраивает сложность для обеспечения среднего показателя 6 блоков в час. Кроме того примерно раз в четыре года уменьшается в два раза вознаграждение за нахождение нового блока что приводит к росту цены криптовалюты.

Сферы применения блокчейн

На сегодняшний день экосистема, складывающаяся вокруг блокчейна, увеличивается в геометрической прогрессии и становится все более сложной. Одной из сфер использования децентрализованного распределенного реестра является криптовалюта которая позволяет быстро и безопасно осуществлять платежи.

Помимо финансовой сферы блокчейн используется там, где каждый блок представляет собой токен. Токен — это единица учёта, не являющаяся криптовалютой, выполняющая функцию «заменителя ценных бумаг» в цифровом мире. Токены представляют собой запись в регистре, распределенную в блокчейн-цепочке.

В медицине технологию блокчейн можно использовать для картотеки с медицинскими данными пациентов. В сфере сделок по купле-продаже и передача прав на собственность. Также Блокчейн может использоваться в регистрации транспортных средств и проверке истории владения ими. А также Блокчейн избавит производителей и потребителей от подделок и облегчит сертификацию отслеживаемой продукции.

Не меньше востребована и технология NFT – невзаимозаменяемых токенов. Она позволяет закрепить и передавать право собственности на цифровые произведения искусства. Развитие блокчейн-экосистемы не ограничивается этими технологиями, и в будущем мы увидим новые решения.

Заключение

Данная технология все еще развивается и в скором будущем вытеснит устоявшиеся порядки изменив вид отношений между поставщиком и получателем. Система прозрачна так как ничего не скрывается, возможность защиты и неизменности данных на сегодняшний день очень важны, именно поэтому технология в ближайшем будущем не утратит, а на оборот приобретёт еще большую популярность и востребованность.

Библиография

1. К. Д. ШИЛОВ, А. В. ЗУБАРЕВ. Блокчейн и распределенные реестры как виды баз данных [online]. [20.12.2022]. Доступно: https://cyberleninka.ru/article/n/blokcheyn-i-raspredelennye-reestry-kak-vidy-baz-dannyh
2. ДАРЬЯ КЛЕЙН. Технология распределенного реестра DLT за рамками блокчейна. [online]. [25.12.2022]. Доступно: https://crypto-fox.ru/faq/distributed-ledger-technology/
3. ЕВГЕНИЙ ХАТА. Что Блокчейн для бабушки за 60 минут. [online]. [30.12.2022]. Доступно: http://loveread.ec/read_book.php?id=72457&p=1
4. Euromoney. How does a transaction get into the blockchain? [online]. [05.01.2022]. Доступно: https://www.euromoney.com/learning/blockchain-explained/how-transactions-get-into-the-blockchain
5. ForkLog. Что такое алгоритм Proof-of-Work (PoW)?. [online]. [10.01.2022]. Доступно: https://forklog.com/cryptorium/chto-takoe-proof-of-work-i-proof-of-stake

FIREBASE

Руслан ЦЫПУШТАНОВ

Департамент Программная Инженерия и Автоматика, группа TI-196, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Молдова

Автор Корреспондент: Руслан ЦЫПУШТАНОВ, e-mail: tipustanov.ruslan@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *данная работа посвящена платформе Firebase. Она освещает историю этой популярной платформы, рассказывает о его преимуществах, недостатках и принципе работы.*

Ключевые слова: *база данных, firebase, NoSql, json, realtime database, authentication.*

Введение

Firebase –американская компания, платформа для разработки программного обеспечения, разработанная в 2011 году и приобретённая компанией Google в 2014 году. В 2012 году в мае, Firebase получила 1,4 млн долларов от компаний Flybridge Capital Partners, Greylock Partners в 2013 году от компаний Union Square Ventures и Flybridge Capital Partners привлекла 5,6 млн долларов [1]. Firebase представляет решение Backend Service для мобильных и для web-приложений которое включает службы для создания, тестирования и управления приложениями. С помощью Google Analytics можно просматривать статистику о своём приложении, здесь можно узнать количество пользователей приложения, количество трафика, который использует аудитория. Firebase используют популярные приложения такие как The New York Times, Todoist и др.

Authentication

Одной из функций, позволяющих сэкономить время, является служба аутентификации. Эта функция разработана для приложений, которые используют Firebase. Firebase authentication имеет множество вариантов регистрации: Email, номер телефона, Google аккаунт, Github аккаунт, Facebook. Что особенного в Firebase Authentication, так это то, что он упрощает безопасный вход в систему, что невероятно сложно реализовать самостоятельно.

Realtime database

Realtime database это база данных NoSql, которая размещена в облаке. Данные синхронизируются с каждым новым подключённым клиентом, данные содержаться в формате JSON. Когда вы подключаете свое приложение к Firebase, вы не подключаетесь через обычный HTTP. Вы подключаетесь через WebSocket. WebSockets намного, быстрее, чем HTTP [2] Одним из превосходства считается то, собственно, что база данных имеет возможность работать в автономном режиме, кэшируя данные в памяти устройства, а впоследствии повторного подключения к интернету синхронизируя их. API-интерфейс realtime databse разрешает исполнять лишь только те операции, которые имеют все шансы быть исполнены проворно. Это позволяет проводить обработку данных в реальном времени, обслуживая миллионы юзеров без вреда для скорости отклика. Можно думать о базе данных как о дереве JSON, размещенном в облаке. В различие от базы данных SQL, тут нет таблиц или же записей. Когда добавляются данные в дерево JSON, оно делается узлом в имеющейся структуре JSON со связанным ключом. База данных Realtime Data-base была оригинальной «Firebase» до того, как она присоединилась к Google в 2014 году. На рисунке 1 изображён пользователь в базе данных.

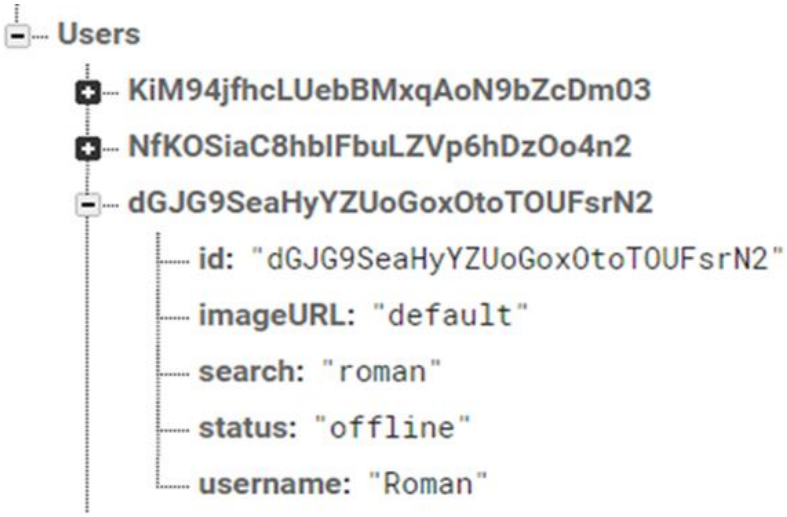


Рисунок 3. Пользователь в базе данных

Cloud firestore

Cloud Firestore — это более новое предложение, которое предлагает большую масштабируемость и более быстрый доступ, чем базы данных в реальном времени. Например, одно изменение состоит в том, что когда база данных реального времени захватывает коллекцию элементов из базы данных, она также захватывает все вложенные коллекции. В Cloud Firestore запросы неглубокие, поскольку они не захватывают вложенные коллекции. Google Firestore, также известный как Cloud Firestore, считается частью платформы разработки Google Firebase. По сути, это облачная база данных NoSQL для хранения и синхронизации данных. К Firestore можно получить прямой доступ мобильными и веб-приложениями через собственные SDK. Клиентская SDK Firestone управляют сетью и аутентификацией, снижая необходимость во программировании. Он позволяет пользователям использовать варианты SDK Unity, Java, C++, Go и Node.js, а также предлагает поддержку REST и RPC API. В то время как база данных реального времени представляет собой просто гигантское дерево JSON, Cloud Firestore немного более структурирован. Все данные состоят из документов (которые в основном предполагает собою хранилища ключей и значений) и коллекций (которые, представляют собой коллекции документов).

Firestore от Google предлагает своим пользователям мощные функции безопасности. Он обеспечивает удобное аварийное восстановление для постоянной защиты данных. Firestore также имеет функции как для веб-платформ, так и для мобильных платформ, включая не каскадные правила и автоматическую проверку данных.

Подключение Firebase к проекту на примере android

Для того чтобы использовать регистрацию и realtime database в проекте, нужно прежде всего добавить Firebase в проект. Существует два варианта: использовать рабочий процесс установки консоли Firebase и с помощью Android Studio Firebase Assistant. Для первого варианта нужно сначала создать проект Firebase. Затем добавить compile зависимость к файлу build.gradle модуля app [3].

Ценообразование

У Firebase есть два тарифных плана: Spark и Blaze. Blaze (платный) — это тарифный план с оплатой по мере. использования: 0,18 доллара США / ГБ за хранилище базы данных, 0,026 доллара США за ГБ за хранилище приложений, а также дополнительные расходы за операции с базой данных, передачу данных и т. д [4]. Realtime database бесплатно поддерживает 100 одновременных подключений, чтобы увеличить одновременных подключений до 200 тысяч, придётся заплатить.

Преимущества Firebase

Главными преимущества fierebase являются:

- Хранится в облаке, поэтому легко доступен везде.
- Бесплатное начало работы на платформе.
- Широкий спектр функций и услуг.
- Предоставляет realtime database.

Недостатки firebase

Главными недостатками fierebase являются:

- Не имеет открытый исходный код.
- Доступны только базы данных nosql.
- Медленные запросы.

Заключение

Firebase, это платформа для разработки мобильных и web приложений от компании Google. Это платформа с закрытым исходным кодом, которая привязана к определенным поставщикам и предоставляет только базы данных NoSQL. Главным минусом можно назвать то, что безвозмездно можно пользоваться только на начальном этапе приложения, а по мере увеличения пользователей придется перейти на платный тариф.

Библиография

1. Firebase. [online]. [20.12.2022]. Доступно: https://ru.wikipedia.org/wiki/Firebase
2. Что такое Firebase. [online]. [20.12.2022]. Доступно: https://howtofirebase.com/what-is-firebase-fcb8614ba442
3. Добавьте Firebase в свой проект Android. [online]. [20.12.2022]. Доступно: https://firebase.google.com/docs/android/setup?hl=ru
4. Стоимость услуги Firebase. [online]. [20.12.2022]. Доступно: http://surl.li/ftdgh

SQLite – КОМПАКТНАЯ ВСТРАИВАЕМАЯ СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

Светлана СОЛТЫК

Департамент Программной Инженерии и Автоматики, TI-196, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Светлана СОЛТЫК, e-mail: soltic.svetlana@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Данная работа посвящена СУБД SQLite. SQLite - это программное обеспечение, которое позволяет пользователям взаимодействовать с реляционной базой данных. В SQLite база данных хранится в одном файле — черта, которая отличает ее от других движков баз данных. Этот факт обеспечивает большую доступность: копирование базы данных не сложнее, чем копирование файла, в котором хранятся данные, совместное использование базы данных может означать отправку вложения электронной почты. В статье рассказывается о преимуществах, особенностях и принципе работы. Так же в статье будет рассмотрено использование SQLite, и сделан вывод на основе рассмотренного материала.*

Ключевые слова: *SQLite, СУБД, SQL, библиотека, таблица, один файл.*

1. Введение

SQLite - это встроенный механизм базы данных SQL. Отличается от многих других баз данных SQL тем, что SQLite не имеет отдельного серверного процесса. SQLite считывает и записывает непосредственно в обычные дисковые файлы. Полная база данных SQL с несколькими таблицами, индексами, триггерами и представлениями содержится в одном файле на диске. Формат файла базы данных является кроссплатформенным - вы можете свободно копировать базу данных между 32-разрядными и 64-разрядными системами или между архитектурой big-end и little-end. Эти функции делают SQLite популярным выбором в качестве формата файла приложения. Файлы базы данных SQLite являются рекомендуемым форматом хранения Библиотекой Конгресса США. Следует думать о SQLite не как о замене Oracle, а как о замене fopen().

SQLite - это компактная библиотека. При включенных всех функциях размер библиотеки может составлять менее 750 КБ, в зависимости от целевой платформы и настроек оптимизации компилятора. Существует компромисс между использованием памяти и скоростью. SQLite обычно работает быстрее, чем больше памяти вы ему предоставляете. Тем не менее, производительность обычно довольно высока даже в средах с низким объемом памяти. Зависит от использования SQLite, SQLite может быть быстрее, чем прямой ввод-вывод файловой системы [1].

SQLite очень внимательно тщательно тестируется перед каждым выпуском и имеет репутацию очень надежного механизма. Большая часть исходного кода SQLite посвящена исключительно тестированию и проверке. Автоматизированный набор тестов запускает миллионы и миллионы тестовых примеров, включающих сотни миллионов отдельных операторов SQL, и обеспечивает 100%-ный охват тестированием отрасли. SQLite изящно реагирует на сбои в распределении памяти и ошибки дискового ввода-вывода. Транзакции выполняются автоматически, даже если они прерваны системными сбоями или сбоями питания. Все это проверяется с помощью автоматизированных тестов с использованием специальных тестовых жгутов, которые имитируют системные сбои. Конечно, даже при всем этом тестировании все еще есть ошибки. Но в отличие от некоторых подобных проектов (особенно коммерческих конкурентов) SQLite открыт и честен в отношении всех ошибок и предоставляет списки ошибок и поминутную хронологию изменений кода.

База кода SQLite поддерживается международной командой разработчиков, которые работают над SQLite полный рабочий день. Разработчики продолжают расширять возможности SQLite и повышать его надежность и производительность, сохраняя при этом обратную совместимость с опубликованными спецификациями интерфейса, синтаксисом SQL и форматом файла базы данных. Исходный код абсолютно бесплатен для всех, кто этого хочет, но также доступна профессиональная поддержка.

Проект SQLite был начат 9 сентября 2000 года. Будущее всегда трудно предсказать, но разработчики намерены поддерживать SQLite до 2050 года. Проектные решения принимаются с учетом этой цели.

2. Особенности SQLite

SQLite обладает следующими заметными особенностями: *бессерверный, автономный, с нулевой конфигурацией, транзакционный* [2].

Бессерверный

Обычно для работы СУБД, такой как MySQL, PostgreSQL и т.д., требуется отдельный серверный процесс. Приложения, которые хотят получить доступ к серверу базы данных, используют протокол TCP/IP для отправки и получения запросов. Это называется архитектурой клиент/сервер.

Следующая диаграмма, показанная на рисунке 1, иллюстрирует архитектуру клиент/сервер СУБД:

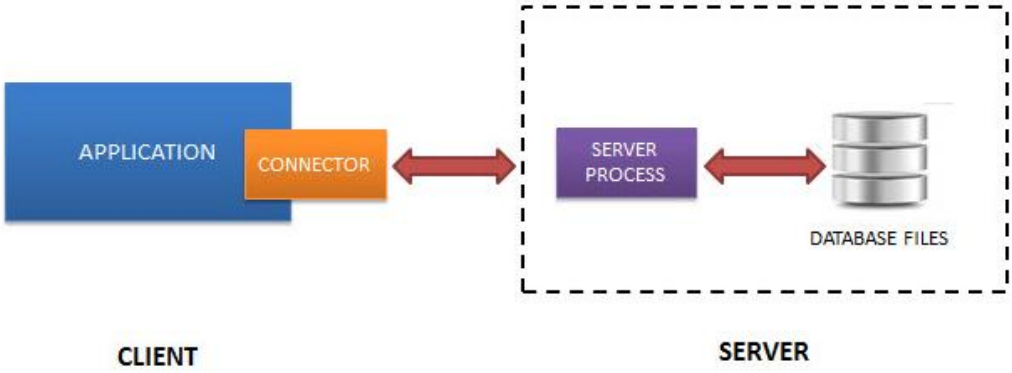


Рисунок 1. Архитектура клиент/сервер СУБД.
Источник: https://tableplus.com/blog/2018/08/sqlite-vs-mysql-which-database-should-you-use.html

SQLite не работает таким образом. Для запуска SQLite НЕ требуется сервер.

База данных SQLite интегрирована с приложением, которое обращается к базе данных. Приложения взаимодействуют с базой данных SQLite для чтения и записи непосредственно из файлов базы данных, хранящихся на диске.

Следующая диаграмма, показанная на рисунке 2, иллюстрирует архитектуру без сервера SQLite:

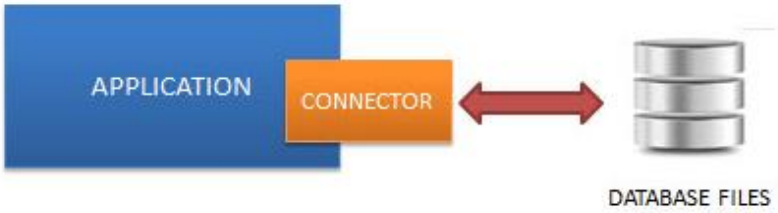


Рисунок 2. Архитектура без сервера SQLite.
Источник: https://tableplus.com/blog/2018/08/sqlite-vs-mysql-which-database-should-you-use.html

Автономный

SQLite является автономным, что означает, что он требует минимальной поддержки со стороны операционной системы или внешней библиотеки. Это делает SQLite пригодным для использования в любой среде, особенно во встроенных устройствах, таких как iPhone, телефоны Android, игровые консоли, портативные медиаплееры и т. д.

SQLite разработан с использованием ANSI-C. Исходный код доступен в виде большого файла sqlite3.c и его заголовочного файла sqlite3.h. Если вы хотите разработать приложение, использующее SQLite, вам просто нужно добавить эти файлы в свой проект и скомпилировать его со своим кодом.

Нулевая конфигурация

Так как SQLite имеет бессерверную архитектуру нет необходимости «устанавливать» SQLite перед его использованием. Нет серверного процесса, который нужно настраивать, запускать и останавливать.

Кроме того, SQLite не использует никаких файлов конфигурации.

Транзакционный

Все транзакции в SQLite полностью совместимы с ACID. Это значит, что запросы и изменения являются последовательными, атомарными, изолированными и надежными.

Иными словами, внутри транзакции все изменения происходят полностью или не происходят вообще, и даже если возникает непредвиденная ситуация, такая как сбой приложения, сбой питания или сбой операционной системы.

3. Недостатки SQLite

Переносимость сигнатур SQLite, к сожалению, делает его плохим выбором, когда множество разных пользователей обновляют таблицу одновременно (для сохранения целостности данных только один пользователь может одновременно записывать в файл). Также может потребоваться дополнительная работа по обеспечению безопасности личных данных благодаря тем же функциям, которые делают SQLite доступным. Кроме того, SQLite не проверяет типы данных. Там, где многие другие программы для баз данных отвергают данные, которые не соответствуют схеме таблицы, SQLite позволяет пользователям хранить данные любого типа в любом столбце [3].

SQLite создает схемы, которые ограничивают тип данных в каждом столбце, но не обеспечивают их соблюдение. В приведенном ниже примере на рисунке 3 показано, что в столбце id предполагается хранить целые числа, в столбце имя предполагается хранить текст, а в столбце возраст предполагается хранить целые числа [4]:

```
CREATE TABLE celebs (
    id INTEGER,
    name TEXT,
    age INTEGER
);
```

Рисунок 3. Создание таблицы и переменных

Однако SQLite не будет отклонять значения неправильного типа. Мы могли случайно вставить неправильные типы данных в столбцы. Хранение разных типов данных в одном столбце - плохая привычка, которая может привести к ошибкам, которые трудно исправить, поэтому важно строго относиться к вашей схеме, даже если SQLite не будет ее применять.

4. Использование SQLite

Даже учитывая недостатки, преимущества возможности доступа к базе данных и управления ею без использования серверного приложения огромны. SQLite используется во всем мире для тестирования, разработки и в любых других сценариях, когда имеет смысл, чтобы база данных находилась на том же диске, что и код приложения. Разработчики SQLite считают его одним из наиболее тиражируемых программных продуктов в мире.

Поскольку SQLite широко используется в каждом смартфоне, и активно используется более 4,0 миллиарда (4,0e9) смартфонов, каждый из которых содержит сотни файлов базы данных SQLite, представляется вероятным, что в активном использовании находится более одного триллиона (1e12) баз данных SQLite.

Заключение

В заключение, следует указать важный плюс в использовании SQLite.

SQLite использует динамические типы для таблиц. Это означает, что вы можете хранить любое значение в любом столбце, независимо от типа данных.

SQLite позволяет использовать одно подключение к базе данных для одновременного доступа к нескольким файлам базы данных. Это обеспечивает множество приятных функций, таких как объединение таблиц в разных базах данных или копирование данных между базами данных одной командой.

SQLite способен создавать базы данных в памяти, с которыми очень быстро работать.

Библиотека SQLite написана на C. существует большое количество привязок на других языках программирования, включая Apple Swift, Delphi, C++, Java, C#, VB.NET , Python, Perl, узел.js, PHP и многие другие [5].

Простота и удобство интеграции SQLite привели к тому, что библиотека используется в браузерах, музыкальных проигрывателях и многих других программах.

SQLite используется в:

- Adobe Integrated Runtime — среда для запуска приложений (частично);
- Gears;
- Autoit;
- Фреймворк Qt;
- Платформа XUL на движке Gecko 1.9+, XULRunner 1.9+;
- Mozilla Firefox (начиная с версии 3.0);
- Mozilla Thunderbird (начиная с версии 3.0);
- Songbird;
- SQLite Manager;
- Skype;
- Viber;
- Некоторые модели GPS-навигаторов Garmin.

Библиография

1. SQLite homepage. What is SQLite? - [online]. [21.12.2022] – Доступно: https://www.sqlite.org/index.html
2. SQLite. SQLite features. - [online]. [22.12.2022] – Доступно: https://www.sqlite.org/features.html
3. Codecademy Team. What is SQLite? - [online]. [04.01.2022] – Доступно: https://www.codecademy.com/article/what-is-sqlite
4. SQLite Tutorial. Basic SQLite Tutorial. - [online]. [25.12.2022] – Доступно: https://www.sqlitetutorial.net/
5. Sibsankar Haldar. - SQLite Database System Design and Implementation (Second Edition, Version 1) – 2015.

OLAP СИСТЕМА АНАЛИЗА

Никита МАКЕЕВ

Департамент Программная Инженерия и Автоматика, группа TI-196, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Молдова

Автор Корреспондент: Никита МАКЕЕВ, e-mail: macheev.nikita@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Статья посвящена OLAP системе анализа обработки данных. Описана общая информация о системе OLAP. Типы OLAP систем, пример с кубом OLAP, различные операции над кубом и разобрано понятие тест FASMI .*

Ключевые слова*: СУБД, OLAP, Тест FASMI, Типы OLAP.*

Введение

В нашем мире все больше и больше фирм, предприятий, корпораций работают с большими объемами данных, имеют своего рода приложения, которые нужны для того, чтобы комплексно проанализировать данные и провести различного рода манипуляции. Данный анализ необходим из-за того, что в итоге он должен поспособствовать принятию различных решений. Для этого служащие должны обладать данной информацией и зачастую, количественной информацией. Именно поэтому такой набор данных следует собрать из всех источниках систем предприятия, а после чего проанализировать. Именно для этого делаются хранилища данных и в свою очередь система OLAP помогает анализировать и обрабатывать большие массивы данных.

OLAP определение

OLAP – это интерактивная аналитическая обработка, своего рода технология обработки данных, основная суть которой заключается в подготовке общей информации на основе больших данных, которые структурированы по многомерному принципу [1].

Основоположник термина OLAP — Эдгар Кодд, в основе его концепции лежит куб. OLAP куб – это структура данных, оптимизированная для очень быстрого анализа данных.

Тест FASMI

С течением времени определение Кодда переросло в тест FASMI, в свою очередь который требует, чтобы OLAP-приложение представляло своеобразную возможность быстро проанализировать информацию.

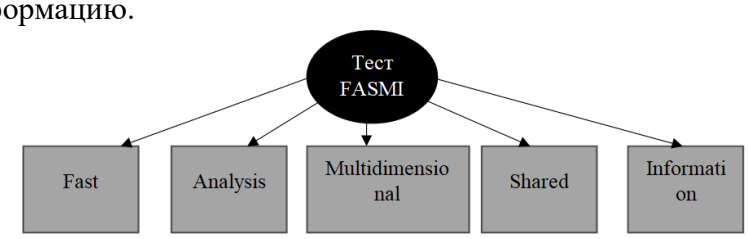


Рисунок 1. Тест FASMI.

Fast (Быстрый) – анализ должен происходить быстро по всем источникам информации. Примерно - 5 с или менее.

Analysis (Анализ) – должна быть возможность осуществлять своего рода анализ например, числовой и статистический.

Shared (Разделяемой) – большое количество пользователей может иметь доступ к данным, при этом необходимо контролировать доступ к конфиденциальной информации.

Multidimensional (Многомерной) – любая информация должна быть многомерной. Это основная, наиболее существенная характеристика OLAP [2].

Information (Информации) – приложение должно иметь возможность обращаться к любой нужной информации, независимо от ее объема и места хранения.

Типы OLAP

Существует 3 типа OLAP

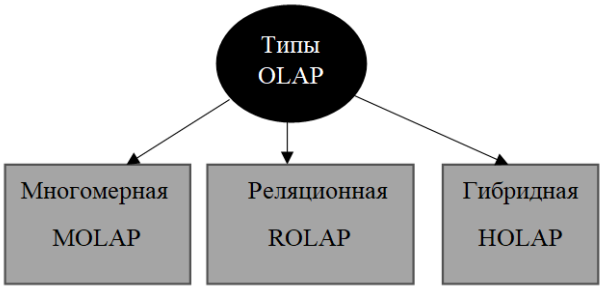


Рисунок 2. Типы OLAP.

MOLAP – Одна из форм OLAP. Основная суть которой заключается в суммировании базы данных и создании многомерной схемы данных с возможностью сохранения, как базовых данных, так и агрегатов [3].

ROLAP – также один из типов OLAP, который работает с реляционной базой данных по итогу все измерения, факты, агрегаты хранятся в реляционных таблицах.

HOLAP – тип OLAP, который использует реляционные таблицы для хранения базовых данных и многомерные таблицы для агрегатов.

Куб OLAP

Куб OLAP – это структура данных с гарантированной возможностью обеспечения быстрого анализа данных. Кубы могут отображать и производить различные операции с объемными данными, в свою очередь предоставляя пользователям доступ к данным и возможность поиска, благодаря чему все данные могут быть сведены в фрагменты и обработаны по мере нужды для поиска решений в различных вопросах [4].

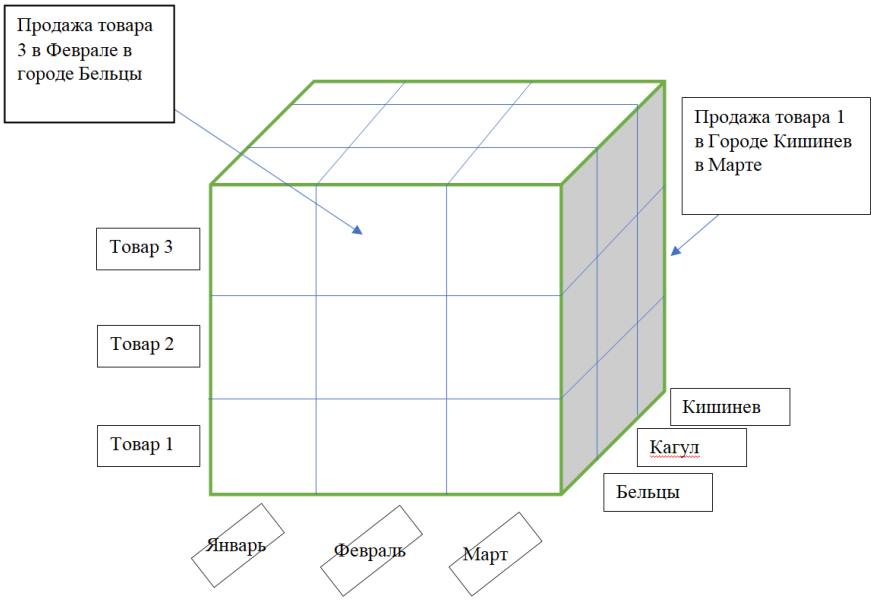


Рисунок 3. Пример Гиперкуба для продаж в Кишиневе

Операции над Гиперкубом

1) Свернуть

Свернуть – это когда данные, которые разделены, объединяются, а затем используются в дальнейшем. Это можно сделать благодаря:

- Уменьшению размеров.
- Используя иерархию понятий, где система группирования вещей делается на основе определенного порядка или уровня.

Пример

Предположим, что наш магазин развивается и выходит на рынок СНГ

Таблица 1

Пример Свертывания

Прибыль за Январь	Кишинев
Прибыль за Январь	Кагул
Прибыль за Январь	Бельцы
Прибыль за Март	Киев
Прибыль за Март	Одесса

Таблица 2

Таблица после свертывания

Прибыль за Январь	Молдова
Прибыль за Март	Украина

2) Развертывание

Развертывание – это своего рода разделение данных на мелкие части, которое также применяется к кубу данных. В примере указанном в таблице 3 применяется расширение измерений, расширение это не что иное, как добавление новых измерений к тому что имеем.Также есть условие, что если данные существуют то мы можем их расширить.Также расширение можно применять к любой оси куба данных [5].

Процесс может быть сделан:

- Переход вниз по ожидаемой иерархии при фрагментации.
- Увеличение размерности текущих наборов данных.

Пример в Таблице 2 мы получили прибыль за месяц в определенном магазине в Кишиневею

Таблица 3

Таблица после Развертывания

Прибыль товара 1	Магазин Linella
Прибыль товара 1	Магазин Nr1
Прибыль товара 1	Магазин Metro

3) Разрез

Разрез используется для того, чтобы получить двумерное представление куба

Так как у нас размерность куба больше 3 мы можем получить различное множество срезов [6]

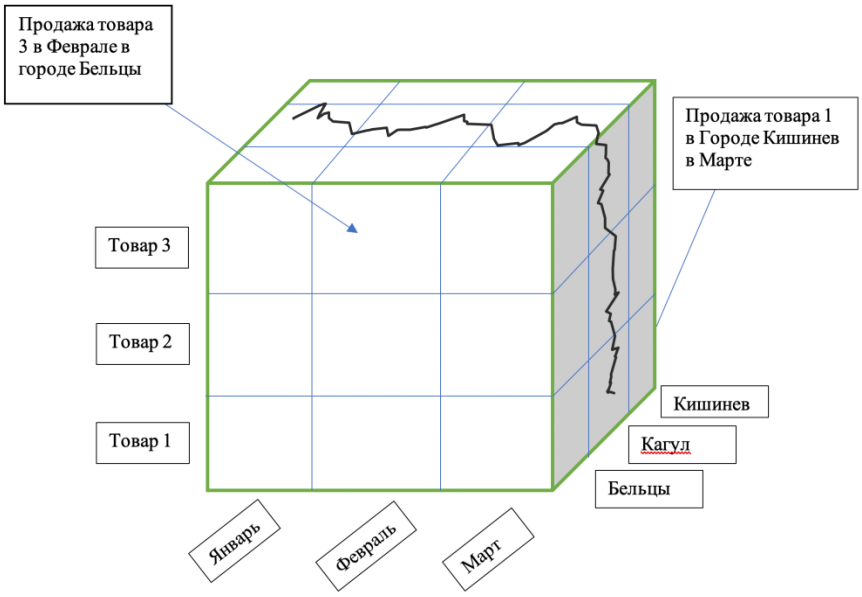


Рисунок 4. Двумерный срез куба для одной меры

Таблица 4

Таблица после Развертывания

	Январь	Февраль	Март
Товар 1	10	3	5
Товар 2	4	5	7
Товар 3	6	5	4

В приведенном примере мы сделали двумерный срез куба для одной меры а именно ‘Кагул’ и получлии таблицу 4, где видно сколько и какого товара было продано в каждый месяц [7]

4) Вращение

Вращение – это своего рода ориентации куба различных данных для проверки других представлений, которые могут иметь данные. С помощью поворачивания одного измерения куба данных, мы получаем реализацию операции вращения. Так для нашего примера мы изменили область товара с оси X на ось Y и месяц с оси Y на ось X, таким образом, предоставив ей другое представление.

Таблица 5

Таблица после Развертывания

	Товар 1	Товар 2	Товар 3
Январь	10	4	6
Февраль	3	5	5
Март	5	7	4

Заключение

В информационном мире, который с каждым годом все больше и больше развивается и процесс информатизации набирает колоссальный рост, всю информацию нужно хранить в различных базах данных. Миру просто необходимо решение в виде OLAP системы, у которой в свою очередь очень значимые плюсы такие как, предметная ориентированность, прямой доступ к данным, удобная навигация по данным, визуализация данных, неизменность данных, Высокая скорость формирования отчетов и др. И это лишь часть из большого количества преимуществ использования OLAP системы, все это в совокупности дает нам отличную систему благодаря которой, можно очень быстро проанализировать различные данные и принять оперативные решения.

Библиография

1. Microsoft Docs. Куб OLAP- [online]. [20.12.2022]. – Доступно: https://docs.microsoft.com/ru-ru/system-center/scsm/olap-cubes-overview?view=sc-sm-2019 .
2. Kpms . Структура OLAP Системы- [online]. [20.12.2022]. – Доступно: https://www.kpms.ru/Automatization/OLAP_system.htm .
3. Olap . Введение в OLAP и многомерных баз дынных- [online]. [20.12.2022]. – Доступно: http://www.olap.ru/basic/alpero2i.asp .
4. Coderlessons . Введение в OLAP и многомерных баз дынных- [online]. [20.12.2022]. – Доступно: https://coderlessons.com/tutorials/bolshie-dannye-i-analitika/teoriia-khraneniia-dannykh/7-chto-takoe-olap .
5. Wikipedia. OLAP - [online]. [20.12.2022]. – Доступно: https://ru.wikipedia.org/wiki/OLAP.
6. Habrahabr. Введение в многомерный анализ. - [online]. [20.12.2022]. – Доступно: https://habr.com/ru/post/126810/.
7. Education-wiki. Операции в OLAP. - [online]. [20.12.2022]. – Доступно: https://ru.education-wiki.com/1424365-operations-in-olap .

POWER BI: ВИЗУАЛИЗАЦИЯ И ИСТОЧНИКИ ДАННЫХ

Елисавета ДМИТРИЕВА

Департамент Программной Инженерии и Автоматики, Группа TI-209, Факультет Вычислительной Техники и Микроэлектроники, Технический Университет Молдовы, Кишинёв, Республика Молдова

Автор корреспондент: Елисавета ДМИТРИЕВА, e-mail: elisaveta.dmitrieva@isa.utm.md

Научный координатор: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *в данной статье рассматривается программное обеспечение Power BI, приводятся некоторые характеристики и компоненты Power BI. Также в статье раскрываются темы источников данных для Power BI, а также их визуализация. т часто используемые технологии, относящиеся к Big Data, и приводится их классификация. В работе перечислены примеры использования Big Data крупными компаниями.*

Ключевые слова: *база данных, Power BI, визуализация данных*

Введение

Многие организации по всему миру ежедневно накапливают данные о доходах, эффективности маркетинга, взаимодействии с клиентами, уровне запасов, темпах производства, количестве сотрудников, затратах и других ключевых показателях деятельности. Однако из-за огромного объема данных, которые необходимо обрабатывать, сотрудникам сложно получить полную картину, что, в свою очередь, может привести к серьезным ошибкам [1].

Визуализация данных может превратить такие подробные данные в полезную бизнес-информацию, которую легко понять, и которая визуально привлекательна. Данные скрывают ценные сведения, которые могут стимулировать рост бизнеса. Проблема в том, что голые цифры не всегда рассказывают всю историю. Однако, когда данные представлены в визуальном формате, они выявляют связи, закономерности и аналитические данные, которые, несмотря на их важность, обычно остаются незамеченными.

Визуализации оживляют данные и блестяще создают истории, основанные на ценной информации, скрывающейся за цифрами. С помощью динамических приборных панелей, интерактивных отчетов, диаграмм, графиков и других визуальных элементов визуализация данных помогает быстро и эффективно раскрыть ключевые аналитические данные [2].

Для упрощения создания отчётов и визуализации данных в 2011 году компания Microsoft создала комплексное программное обеспечение бизнес-анализа Power BI [3].

Power BI — это унифицированная и масштабируемая платформа для самостоятельной и корпоративной бизнес-аналитики. Это пакет облачных сервисов бизнес-аналитики от Microsoft. Он используется для преобразования необработанных данных в значимую информацию с помощью интуитивно понятных визуализаций и таблиц. Так что, теперь можно легко анализировать данные и принимать на их основе важные бизнес-решения.

Power BI

Power BI – это набор программных сервисов Microsoft, которые работают вместе, превращая несвязанные источники данных компании в целостные интерактивные отчеты. При этом источником могут быть базы данных, файлы Excel, данные из облачных источников и интернета, текстовые файлы, [4] база данных SQL Azure, Azure Synapse Analytics, раннее хранилище данных sql Data Warehouse, Spark для Azure HDInsight и т.д.

Power BI можно использовать не только для индивидуальных отчетов компании, но и для презентаций компании и бизнес-планов для инвесторов и потенциальных партнеров. Его также можно использовать для работы над групповыми проектами.

Power BI включает в себя:

– приложение Power BI Desktop для создания моделей данных и разработки отчетов — панелей управления (Dashboard);

– Power BI Online Service (SaaS — Software as a Service) для публикации отчетов;

– мобильное приложение Power BI Mobile для просмотра отчетов на мобильных устройствах и планшетах под управлением Windows, iOS и Android.

Визуализация данных

Визуализация данных – самая интересная и эффектная часть Power BI.

Преимущества:

– возможность напрямую подключаться к различным источникам данных;

– все данные объединяются в одном отчете - нет необходимости просматривать разные отчеты, чтобы получить полную картину о компании;

– запись последовательности процессов и расчетов - нет необходимости повторять расчеты, освобождая вас от рутинной монотонности, а расчеты производятся автоматически;

– возможность работы в режиме реального времени. Технология оперативных больших данных относится ко всем обычным, повседневным данным, которые мы генерируем (онлайн-транзакции, социальные сети и т.д.).

Визуализация данных – это конечный результат работы в Power BI, который позволит наглядно и интерактивно отследить основные тенденции, зависимости и отклонения показателей [4]. Это возможность эффективно представить компанию и наглядно продемонстрировать результаты своей работы собственникам и руководителям. А понятные отчеты — это то, что нужно для принятия правильных бизнес-решений.

Создание содержимого в Power BI

В Power BI есть два способа создания или импорта собственного содержимого: файлы и базы данных.

Файлы:

– Книги Excel или файлы.xlsx и XLSM могут содержать различные типы данных. Например, книги могут включать данные, которые вводите на листы самостоятельно, или данные, запрашиваемые и загружаемые из внешних источников данных с помощью Power Query. Power Query можно получить & данные преобразования на вкладке Данные Excel или получить внешние данные в Power Pivot. Возможность импортировать данные из таблиц на листах или из модели данных. Дополнительные сведения см. в статье Получение данных из файлов в Power BI.

– Power BI Desktop или PBIX-файлы отчетов запрашивают и загружают данные из внешних источников данных для создания отчетов. В Power BI Desktop можно расширить модель данных с помощью мер и связей, а также опубликовать PBIX-файлы в служба Power BI. Power BI Desktop предназначено для опытных пользователей, которые хорошо разобрываются в источниках данных, запросах и преобразовании данных, а также о моделировании данных. Дополнительные сведения см. в статье Подключение к данным в Power BI Desktop.

– Разделенные запятыми файлы значений или .csv — это простые текстовые файлы со строками данных, содержащими значения, разделенные запятыми. Например, файл.csv, содержащий данные имени и адреса, может содержать много строк, каждая из которых содержит значения для имени, фамилии, адреса улицы, города и штата. Невозможно

импортировать данные в файл.csv, но многие приложения, например Excel, могут сохранять простые табличные данные в виде файлов.csv.

– Для других типов файлов, таких как XML (.xml) или текст (.txt), можно использовать Excel Get & Transform Data или Power BI Desktop Get data для запроса, преобразования и загрузки данных. Затем можно импортировать файл Excel или Power BI Desktop в служба Power BI.

Место хранения файлов имеет существенное значение. OneDrive обеспечивает максимальную гибкость и интеграцию с Power BI. Можно хранить данные на локальном диске, но если данные необходимо обновлять, следует предпринять некоторые дополнительные шаги.

Базы данных:

В службе Power BI можно подключиться в реальном времени к следующим базам данных в облаке:

– База данных SQL Azure
– Azure Synapse Analytics, ранее sql Data Warehouse

Spark в Azure HDInsight

Подключения из Power BI к предоставленным выше базам данных являются активными. Например, можно подключиться к базе данных Azure SQL и начать изучать ее данные, создавая отчеты в Power BI. Всякий раз, когда срезаются данные или, добавляется поле в визуализацию, Power BI напрямую запрашивает базу данных.

Для других типов баз данных необходимо использовать Power BI Desktop или Excel для подключения, запроса и загрузки данных в модель данных. Затем этот файл можно импортировать в Power BI, где находится набор данных. При настройке запланированного обновления Power BI использует сведения о конфигурации и подключении из файла для прямого подключения к источнику данных. Power BI запрашивает обновления и загружает их в набор данных.

Наборы данных и источники данных

В Power BI часто используются термины набор данных и источник данных, а также их синонимы. Но наборы данных и источники данных — это две разные вещи, хотя они и связаны друг с другом.

Power BI автоматически создает набор данных при использовании команды Получение данных для подключения к приложению-шаблону, файлу или источнику динамических данных и их импорта из нее. Наборы данных содержат информацию об источнике данных и учетные данные источника данных. Наборы данных также часто содержат подмножество данных, скопированных из источника данных. При создании визуализаций в отчетах и информационных панелях часто отображаются данные из набора данных.

Данные в наборе данных могут поступать из источников данных. Например, данные могут поступать из следующих источников данных:

— веб-служба, такая как Google Analytics или QuickBooks;
— облачная база данных, такая как база данных SQL Azure;
— база данных или файл на локальном компьютере или сервере в организации.

Обновление данных

Если файл сохраняется на локальном диске или на диске в организации, может потребоваться шлюз Power BI, чтобы обновить набор данных в Power BI. Компьютер, на котором хранится файл, должен работать во время обновления. Также можно повторно импортировать файл или использовать команду Опубликовать из Excel или Power BI Desktop, но эти параметры не являются автоматизированными процессами.

Если сохранять файлы в OneDrive для работы, учебы или на сайтах групп SharePoint, набор данных, отчеты и панель мониторинга всегда актуальны. Так как OneDrive и Power BI

находятся в облаке, Power BI может напрямую подключаться к файлам или импортировать файлы в Power BI. Power BI подключается примерно раз в час и проверяет наличие обновлений. При их наличии все наборы данных и визуализации обновляются автоматически.

Приложения-шаблоны из служб также обновляются автоматически, в большинстве случаев один раз в день. Обновление можно выполнить вручную, но отображаются ли обновленные данные, зависит от поставщика услуг. Обновления приложений-шаблонов от сотрудников организации зависят от используемых ими источников данных и от того, как создатель приложения настроит обновление.

Azure SQL База данных, Azure Synapse Analytics (ранее SQL Data Warehouse) и Spark в Azure HDInsight являются облачными источниками данных. Служба Power BI также находится в облаке, поэтому Power BI может подключаться к этим источникам данных в реальном времени с помощью DirectQuery. При использовании DirectQuery Power BI всегда синхронизирована, и не нужно настраивать запланированное обновление.

SQL Server Analysis Services — это динамическое подключение к Power BI, как и к облачной базе данных Azure. Разница заключается в том, что база данных находится на сервере в организации. Для этого типа подключения требуется шлюз Power BI, который может настроить ИТ-отдел.

Заключение

Power BI открывает новые возможности в процессе визуализации данных. Данное ПО упрощает редактирование базы данных, путём преобразования её во всем привычную Excel таблицу. Также Power BI гарантирует сохранность и конфиденциальность данных, что не маловажно для организаций, которые решили использовать данное ПО для создания своих отчётов, систем показателей, панелей мониторинга и потоковых наборов данных. В Power BI удобно хранить свои таблицы с базами данных в Центре данных и в любое время их редактировать или использовать их для визуализации. Привлекательность Power BI заключается в том, что его технология настолько совершенна, что вам больше не нужно быть экспертом по базам данных, чтобы работать с ними, писать SQL-запросы и т.д. Приложение Microsoft Power BI включает инструменты, уже используемые в Excel для бизнес-анализа.

Библиография

1. finalytics.pro, Power BI: бизнес анализ и визуализация данных – [online]. [20.12.2022]. – Доступно: https://finalytics.pro/inform/power-bi-analyz-vizualizatsiya/
2. learn.microsoft.com, Визуализации в отчетах Power BI. – [online]. [20.12.2022]. – Доступно: https://learn.microsoft.com/ru-ru/power-bi/visuals/power-bi-report-visualizations
3. learn.microsoft.com, Источники данных для службы Power BI. – [online]. [20.12.2022]. – Доступно. – Режим доступа: https://learn.microsoft.com/ru-ru/power-bi/connect-data/service-get-data
4. Wikipedia, Power BI. – [online]. [20.12.2022]. – Доступно: https://ru.wikipedia.org/wiki/Power_BI

MICROSOFT SQL SERVER DATA TOOLS - BUSINESS INTELLIGENCE (SSDT-BI)

Светлана КРАЧУН

Департамент Программной Инженерии и Автоматики, группа TI-209, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Светлана КРАЧУН, e-mail: svetlana.craciun@isa.utm.md

Научный координатор: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *В рамках данной статьи произведён краткий анализ набора инструментов Microsoft SQL Server Data Tools - Business Intelligence (SSDT-BI), в ходе которого будут разобраны его принцип работы и польза для разработчика.*

Ключевые слова: *данные, Business Intelligence (SSDT-BI), SSIS (SQL Server Integration Service), SSAS (SQL Server Analysis Service), SSRS (SQL Server Reporting Service).*

Введение

Работа с данными является важной частью человеческой жизни. Она позволяет поддерживать порядок и структурировать информацию. Однако, может ли это занятие быть простым? В прошлом, существовали архивы, а информация записывалась на бумаге, что определённо делало обработку данных трудоёмким и малоэффекивным.

В наше время, с помощью современных технологий, обработка данных стала более простой и эффективной, что позволяет быстро и точно получать необходимую информацию. Это критично важно в различных областях, таких как бизнес, наука, медицина и многих других.

Спрос и технологии являются двумя ключевыми элементами, которые доводят конкуренцию до максимума, если не до предела, размывая отраслевые границы и заставляя предприятия меняться. Прогресс в области информационных технологий также позволил компаниям накапливать большие объемы данных из многочисленных источников посредством своих бизнес-процессов. Чтобы оставаться конкурентоспособными, организациям необходим инструмент, который может обеспечить целостное представление об операционной среде организации, используя преимущества огромного объема собранных данных и позволяя корпоративным конечным пользователям лучше анализировать состояние своего бизнеса и предоставлять точную информацию, которая способствует улучшенному принятию решений.

Одним из главных помощников в работе с данными является Microsoft SQL Server Data Tools - Business Intelligence (SSDT-BI).

1. Что это такое?

Microsoft SQL Server Data Tools - Business Intelligence (SSDT-BI) является набором инструментов для разработки интеллектуальных бизнес-приложений на основе SQL Server. Это широко используемая система управления реляционными базами данных (RDBMS), предназначенная для предоставления пользователям возможности хранения и поиска данных в соответствии с их требованиями. Этот набор инструментов содержит в себе мощные средства для создания, деплоя и отладки интеллектуальных бизнес-приложений, включая инструменты для создания отчетов, анализа данных и т.д [1].

Рис. 1. Логотип Business Intelligence (SSDT-BI)

Business Intelligence (BI) — это быстрорастущий сектор программного обеспечения, и поставщики программного обеспечения быстро разрабатывают различные инструменты бизнес-аналитики SQL Server для удовлетворения растущих требований организаций к анализу данных [2].

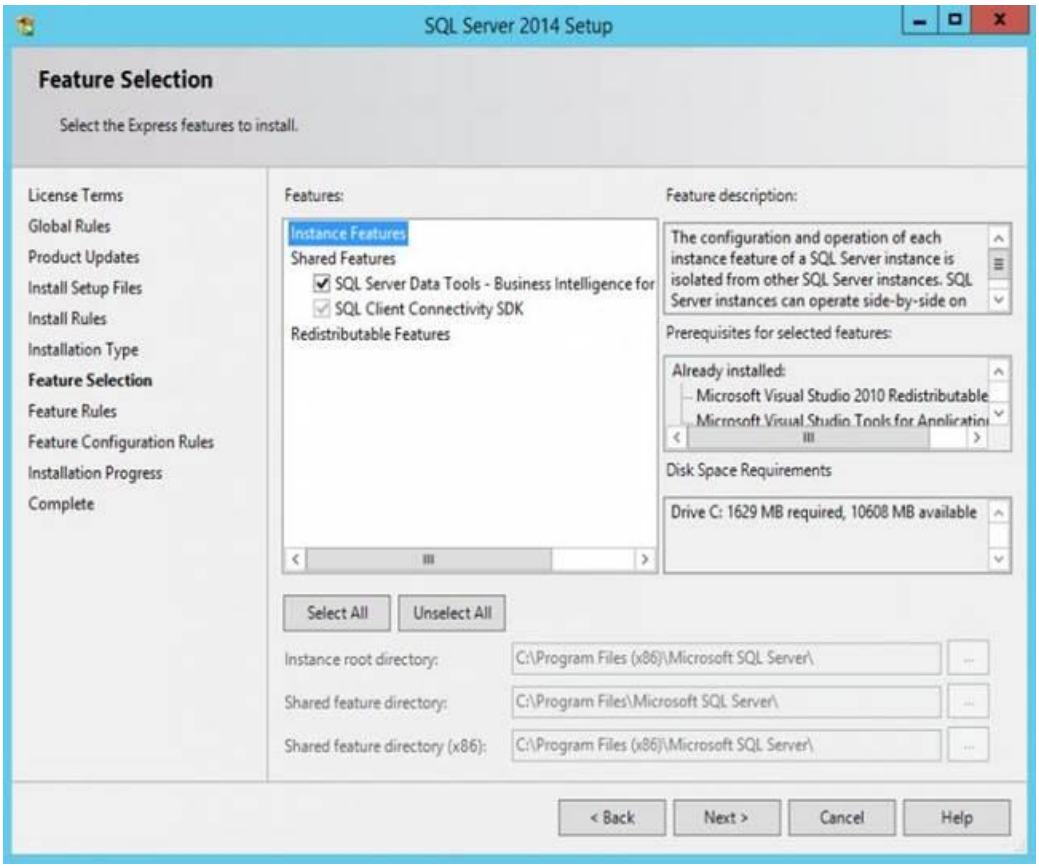


Рис. 2. Установка Business Intelligence (SSDT-BI)

2. Компоненты SSDT-BI

Стек Business Intelligence состоит из широкого набора инструментов, повышающих ценность базы данных SQL-сервера. Выделим самые основные:

- SSIS (SQL Server Integration Service)
- SSAS (SQL Server Analysis Service)
- SSRS (SQL Server Reporting Service)

Все вышеописанные компоненты, в свою очередь, являются компонентами SQL Server Data Tools (SSDT), который предоставляет инструменты для разработки и деплоя баз данных. Этот инструмент позволяет разработчикам создавать и изменять базы данных, используя визуальный интерфейс и встроенные шаблоны.

Рис. 3. Установка наборов инструментов Business Intelligence (SSDT-BI)

SQL Server Integration Service, также известная как SSIS, представляет собой компонент программного обеспечения базы данных Microsoft SQL Server, который можно использовать при передаче данных из рабочей среды в базу данных анализа или хранилище данных. SSIS может извлекать данные из различных источников, таких как базы данных SQL, базы данных Oracle, базы данных DB2 и файлы Microsoft Excel [2].

В дополнение к перемещению данных SSIS поставляется с несколькими основными компонентами, которые позволяют планировать запуски или сообщать об ошибках. Это гарантирует, что автоматизированные процессы, заполняющие ваши инструменты Business Intelligence, будут снабжены самыми последними данными [3].

SQL Server Analysis Services (SSAS) — это функция онлайн-аналитической обработки данных и создания отчетов, используемая в SQL Server Business Intelligence для обеспечения работы данных с конечными пользователями. SSAS создает многомерные кубы OLAP (технология обработки данных, заключающаяся в подготовке суммарной информации на основе больших массивов данных, структурированных по многомерному принципу), используя данные из киосков/хранилищ данных, чтобы анализировать данные быстрее и глубже [2].

SSAS предоставляет огромное количество встроенных средств агрегирования и детализации. Служба SSAS предварительно вычисляет и физически сохраняет данные, полученные из других источников, таким образом, чтобы по ним можно было быстро получать отчеты и вычислять их, что делает время ответа на запрос очень коротким [3].

Рис. 4 Логотип SQL Server Analysis Services (SSAS)

SQL Server Reporting Services, сокращенно называемые SSRS, представляют собой средство создания отчетов, которое позволяет создавать табличные отчеты и различные типы отчетов в виде диаграмм, отчетов в виде графиков, карт или географических отчетов. Кроме того, SSRS поддерживает интеграцию с SharePoint и Power BI, а также отчеты для мобильных устройств [2].

SSRS поставляется с широким спектром функций графического интерфейса и широким набором встроенных виджетов отчетов для отображения исключительно удобных визуализаций данных, помимо обширной коллекции простых графиков.

SSRS соединяет различные внешние источники данных и компилирует необходимые отчеты в легко читаемый формат, который предоставляет конечным пользователям правильную информацию на основе разрешений пользователя и группового доступа [3].

Рис. 5. SQL Server Reporting Services

3. Принцип работы Business Intelligence

Все решения SQL Server Business Intelligence имеют стандартизированный подход к подготовке данных, чтобы сделать их готовыми к использованию. Этот метод преобразует данные, оптимизированные для системы с трудоемкими транзакциями, и передает их в систему, более подходящую для аналитических запросов.

Производственный сервер обычно ориентирован на OLTP (онлайн-обработку транзакций). В результате он имеет реляционную конфигурацию, индексы и типы данных, облегчающие операции INSERT, UPDATE и DELETE. Тот же шаблон не применяется к данным, оптимизированным для бизнес-аналитики [3].

4. Преимущества, предоставляемые SQL Server Business Intelligence

Преобразование: SQL-серверы могут преобразовывать сложные данные и преобразовывать их в полезные идеи, которые обеспечивают долгосрочное развитие и расширение бизнеса.

Модернизация: инструмент бизнес-аналитики SQL Server выходит за рамки создания начального отчета. Он может предоставлять полностью интегрированные отчеты Power BI, ключевые показатели эффективности и отчеты с разбивкой на страницы с единого доступного веб-портала.

Соединение: SQL-серверы предоставляют гибридные отчеты, которые соединяют вас с вашими локальными данными из облака. Нет требований, связанных с перемещением данных.

Масштаб: платформа SQL Server Business Intelligence полностью масштабируема, поэтому ее можно расширять по мере расширения информации.

Бизнес-аналитика SQL Server

Существует два основных типа систем, одна из которых еще более оптимизирована, чем другая, и поэтому является конечной точкой в потоке данных SQL Server Business Intelligence:

1. SQL Server Business Intelligence, этап 1: хранилище данных
2. SQL Server Business Intelligence, этап 2: онлайн-куб аналитической обработки (OLAP)

Этап 1:

Первым этапом вне системы OLTP является хранилище данных. Обычно он размещается в стандартном экземпляре SQL Server как обычная база данных. Однако он включает денормализованную модель базы данных вместо реляционной. Возможно размещение экземпляра OLTP и хранилища данных в одном экземпляре SQL Server. Мы также можем создавать отчеты, которые напрямую запрашивают хранилище данных. В результате небольшие компании обычно останавливаются на этом этапе, тем более что базовые уровни лицензии SQL Server обычно не включают инструменты служб Analysis Services, необходимые для следующего шага к аналитической обработке.

Этап 2:

На втором этапе разрабатывается куб оперативной аналитической обработки (OLAP). Куб OLAP — это структура данных, которая преодолевает ограничения реляционных баз данных за счет использования многомерного представления. Кубы OLAP могут эффективно обрабатывать совокупные данные, поступающие из обширных наборов данных. Кроме того, они могут содержать сложные ключевые показатели эффективности бизнеса и сложные показатели производных данных.

Клиенты могут просматривать кубы OLAP с помощью таких инструментов, как сводные таблицы Excel, или их можно использовать в качестве источника данных для служб отчетов SQL Server, где информация должна отображаться в более структурированном стиле. Мы можем запрашивать кубы OLAP, используя язык запросов MDX.

Когда вы дойдете до этого этапа, вы сможете запросить оптимизированную базу данных, доступную только для чтения, и вернуть результаты с богатыми бизнес-аналитиками для обширных наборов данных с малым временем отклика (по сравнению с аналогичными запросами в традиционной базе данных OLTP). В результате своей многомерной формы и природы куба, доступного только для чтения, куб занимает меньше места на диске и полностью обходит проблемы параллелизма, не требуя блокировок [3].

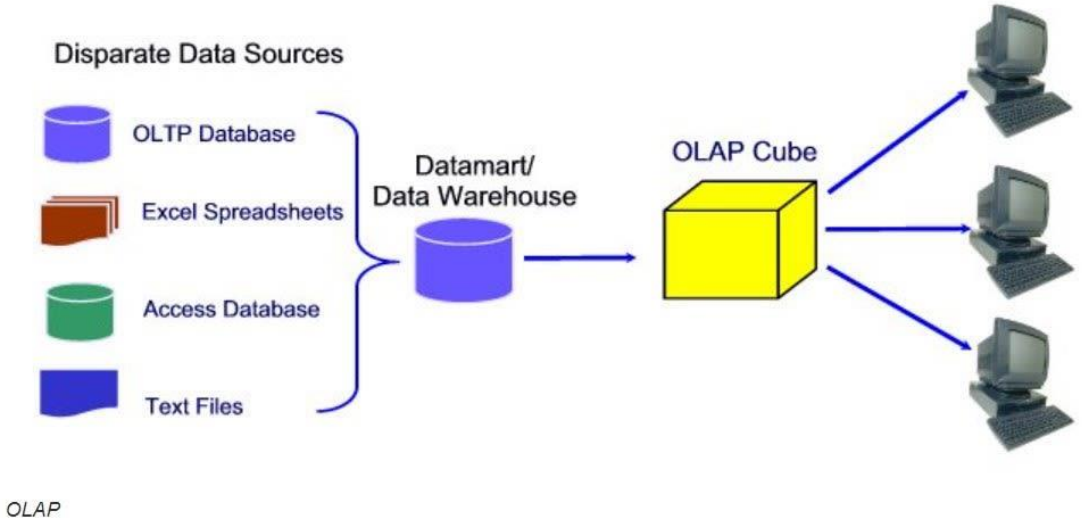


Рис. 6. Кубы OLAP

Неотъемлемые компоненты бизнес-аналитики SQL Server

А) Конвейер интеграции данных

SQL Server Business Intelligence нуждается как в базовой схеме, которая поддерживает эти базы данных, так и в системах ETL (извлечение, преобразование, загрузка), которые обеспечивают передачу данных между базой данных OLTP и хранилищем данных. Инструменты SQL Server Data Tools можно использовать для любых операций, требующих загрузки, передачи, улучшения и отправки данных. Передача между хранилищем данных и кубом OLAP выполняется службами SQL Server Analysis Services (SSAS).

Б) Представление данных

Третий этап начинается после того, как разработан базовый макет и реализован поток данных между различными уровнями. После выполнения этих операций нам нужно найти способ доставить эти данные конечному пользователю. Отчеты — это клиентская часть Business Intelligence. В бизнес-аналитике SQL Server это можно сделать несколькими способами. Один из наиболее распространенных способов — через службы отчетов SQL Server (SSRS). Эта функция может создавать, размещать и представлять отчеты на основе данных BI [4].

Заключение

В целом, SSDT-BI является мощным инструментом для разработки интеллектуальных бизнес-приложений на основе SQL Server, который позволяет разработчикам эффективно создавать и деплоировать базы данных, отчеты, анализы и интеллектуальные приложения. Использование SSDT-BI позволяет снизить время и стоимость разработки, а также улучшить качество результатов.

Этот инструмент может использоваться как для создания новых приложений, так и для модернизации существующих, и его можно интегрировать с другими средствами разработки, такими как Visual Studio. SSDT-BI является важным инструментом для разработки интеллектуальных бизнес-приложений и помогает разработчикам достичь желаемых результатов.

Литература

1. Wikipedia – электронная энциклопедия – [online]. [20.12.2022]. – Доступно - https://ru.wikipedia.org/
2. Документация Microsoft – [online]. [20.12.2022]. – Доступно - https://learn.microsoft.com/
3. SQL Server Business Intelligence 101: подробное руководство – [online]. [20.12.2022]. – Доступно - https://hevodata.com/
4. Business Intelligence (BI): полное руководство – [online]. [20.12.2022]. – Доступно https://www.microstrategy.com/

EXECUTAREA UNEI CERERI (QUERY) în MS SQL SERVER

Ion MORARU

Departamentul Ingineria Software şi Automatică, grupa TI-181 F/R, Facultatea Calculatoare, Informatică şi Microelectronică, Universitatea Tehnică a Moldovei, Chişinău, Republica Moldova

Autorul corespondent: Ion MORARU, e-mail: ion.moraru@isa.utm.md

Coordonator ştiinţific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *În articolul dat este reprezentat cum funcţionează SQL Server şi cum să-i valorificam puterea. O cerere de date poate fi trimisa spre execuţie in mai multe moduri, dar SQL Server are un regim definit cum sa le execute si sa le returneze clientului. Fluxul de cererii si modalitatea de execuţie este descrisa in acest document din perspectiva clientului si modalitate de lucru a serverului. Sistemul de Gestiune a bazelor de date SQBD este un sistem complex de lucru cu datele in procesul de prelucrare si păstrare a datelor.*

Cuvinte cheie: *SGBD, TDS, CLR, RPC.*

1. Query

SQL Server este o platformă client-server. Singura modalitate de a interacţiona cu baza de date back-end este prin trimiterea de query, care conţin comenzi pentru baza de date. Protocolul folosit pentru a comunica între aplicaţia dumneavoastră şi baza de date se numeşte TDS (Tabular Data Sream). Aplicaţia poate folosi una dintre cele mai multe implementări la nivelul clientului ale protocolului: *SqlClient*, OleDB, ODBC, JDBC, PHP Driver pentru SQL Server gestionat CLR(Common Language Runtime) sau implementarea FreeTDS cu sursă deschisă. Esenţa este că atunci când aplicaţia funcţionează pe o baza de date pentru funcţionare, va trimite o solicitare prin protocolul TDS. Cererea în sine poate lua mai multe forme: [1]

a) Solicitare date pe loturi (Batch Request)

Acest tip de cerere conţine doar text T-SQL pentru ca un lot să fie executat. Acesta este tipul de solicitare pe care *SqlClient* o trimite oricare dintre *SqlCommand*: *ExecuteReader(), ExecuteNonQuery(), ExecuteScalar(), ExecuteXmlReader()* (sau echivalentele asincrone respective) pe un obiect *SqlCommand* cu o listă de parametri goală. Monitorizarea cu SQL Profiler, o clasă de evenimente *SQL:BatchStarting*.

b) Solicitare date prin procedură (RPC Request)

Acest tip de cerere conţine un identificator de procedură, care trebuie de executat împreună cu de parametri care se despart prin virgula. Un interes deosebit este atunci când id-ul procedurii va fi 12, adică. *sp_execute*. În acest caz, primul parametru este textul T-SQL de executat. Cererea o va trimite spre executare *SqlCommand* cu cel putin un parametru. Pentru monitorizare folosim SQL Profiler RPC:StartingEvent.

c) Solicitare prin bulk (Bulk Load Request)

Bulk Load este o solicitare de tip special utilizată de operaţiunile de inserare în bloc, cum ar fi utilitarul *bcp.exe*, interfaţa *IRowsetFastLoad* din OleDB sau de clasa *SqlBulkcopy*. Încărcarea în bloc este diferită de celelalte solicitări, deoarece este singura solicitare care începe execuţia înainte ca cererea să fie finalizată pe protocolul TDS. Acest lucru îi permite să înceapă execuţia şi apoi să înceapă să consume fluxul de date de inserat.

După ce o solicitare TDS completă ajunge la motorul bazei de date, SQL Server va crea o sarcină pentru a gestiona cererea. Lista cererilor de pe server poate fi interogată din *sys.dm_exec_requests* [2].

2. Sarcini (Task)

Sarcinile menționate mai sus. sunt create pentru a gestiona cererea, va reprezenta cererea de la început până la finalizare. De exemplu, dacă cererea este o solicitare de tip lot (SQL Batch), sarcina va reprezenta întregul lot, nu instrucțiuni individuale. Declarațiile individuale din lotul SQL nu vor crea sarcini noi. Anumite instrucțiuni individuale din cadrul lotului se pot executa cu paralelism (denumite adesea DOP, Degree of Parallelism), în cazul lor, sarcina va genera noi sub-sarcini pentru executarea în paralel. Dacă cererea returnează un rezultat, lotul este complet când rezultatul este consumat complet de client (de exemplu, când eliminați SqlDataReader). Puteți vedea lista sarcinilor de pe server interogând *sys.dm_os_tasks*.

Când o nouă solicitare ajunge pe server și sarcina este creată pentru a gestiona acea cerere, în starea *PENDING*. În această etapă, serverul nu are încă idee care este de fapt cererea. Sarcina trebuie să înceapă să se execute mai întâi, iar pentru aceasta motorul trebuie să îi atribuie un worker.

3. Muncitorii (Workers)

Workers sunt grupul de fire (*threads*) al SQL Server. Un număr de workers este creat inițial la pornirea serverului și mai mulți pot fi creați la cerere până la fire (*threads*) maxime de lucru configurate. Numai *workers*-rii execută cod. *Workers* așteaptă ca sarcinile cu statut *PENDING* să devină disponibile (din cererile care vin pe server) și apoi fiecare *worker* preia exact o sarcină și o execută. *Worker*-ul este ocupat (ocupat) până când sarcina se termină complet. Sarcinile care sunt în așteptare atunci când nu mai sunt *workers* disponibili vor trebui să aștepte până când una dintre sarcinile de execuție (în rulare) se finalizează și *worker*-ul care a executat acea sarcină devine disponibil pentru a executa o altă sarcină în așteptare. Pentru o solicitare de lot (batch) SQL, *worker*-ul care preia sarcina respectivă va executa întregul lot SQL (fiecare instrucțiune). Acest lucru ar trebui să rezolve întrebarea adesea pusă dacă instrucțiunile dintr-un lot SQL (=> cerere => sarcină => worker) se pot executa în paralel: nu, deoarece sunt executate pe un singur fir (=> *worker*), atunci fiecare instrucțiune din lot nu va incepe sa se execute pina nu se finalizeaza precedenta. Pentru instrucțiunile care folosesc intern paralelism (DOP > 1) și creează sub-sarcini, fiecare sub-sarcină trece exact prin același ciclu: este creată ca *PENDING* și un *worker* trebuie să o ridice și să o execute (un worker diferit de de cel ce executa sarcina parinte!). Listele și starea *workers* din SQL Server pot fi văzute interogând *sys.dm_os_workers*.

4. Analiză și compilare

Odată ce o sarcină a început să execute o solicitare, primul lucru pe care trebuie să-l facă este să înțeleagă conținutul cererii. În această etapă, SQL Server se va comporta ca un limbaj VM (virtual machina) pentru interpretarea limbajului: text T-SQL va fi analizată și va fi creat un arbore de sintaxă abstractă pentru a reprezenta cererea. Întreaga cerere (lotul) este analizată și compilată. Dacă apare o eroare în această etapă, cererile se încheie cu o eroare de compilare (cererea este apoi finalizată, sarcina este finalizată și *worker*-ul este liber să preia o altă sarcină în așteptare). SQL și T-SQL este un limbaj declarativ de vârf cu instrucțiuni extrem de complexe. Compilarea loturilor T-SQL nu are ca rezultat un cod executabil similar cu instrucțiunile native ale procesorului și nici măcar codul octet JVM, dar are ca rezultat planuri de acces la date (sau planuri de interogare). Aceste planuri descriu modalitatea de deschidere a tabelelor și indexurilor, de a căuta și de a localiza rândurile de interes și de a efectua orice manipulare a datelor, așa cum este solicitat în lotul SQL. De exemplu, un plan de interogare va descrie o cale de acces precum „deschide indexul idx1 pe tabelul t, localizează rândul cu cheia „k” și returnează coloanele a și b”. Ca o notă secundară: o greșeală comună făcută de dezvoltatori este încercarea de a veni cu o singură interogare T-SQL care să acopere multe alternative, de obicei prin utilizarea expresiilor inteligente în clauza WHERE, având adesea multe alternative SAU (de ex. (COLUMN = @ parametrul SAU @parametrul ESTE NULL). Pentru dezvoltatorii care încearcă să mențină lucrurile DRY și evitarea repetării sunt bune practici, pentru interogările SQL sunt pur și simplu proaste. Compilarea trebuie să vină cu o cale de acces care să funcționeze pentru orice valoare a parametrilor de intrare și rezultatul este cel mai adesea sub-optim [3].

5. Optimizare

Vorbind despre alegerea unei căi optime de acces la date, aceasta este următoarea etapă din durata de viață a cererii: optimizarea. În SQL și în T-SQL, optimizarea înseamnă a alege cea mai bună cale de acces la date dintre toate alternativele posibile. Luați în considerare că dacă avem o interogare simplă cu îmbinare între două tabele și fiecare tabel are un index suplimentar, există deja 4 moduri posibile de a accesa datele și numărul de posibilități crește exponențial pe măsură ce complexitatea interogării crește și sunt disponibile mai multe căi de acces alternative (practic, mai mulți indici). Adăugați la aceasta că JOIN-ul se poate face folosind diverse strategii (neted loop, hash, merge) și vom vedea de ce optimizarea este un concept atât de important în SQL. SQL Server folosește un optimizator bazat pe costuri, ceea ce înseamnă că va lua în considerare toate (sau cel puțin multe) alternative posibile, va încerca să facă o presupunere despre costul fiecărei alternative și apoi o va alege pe cea cu cel mai mic cost. Costul este calculat în primul rând luând în considerare dimensiunea datelor care ar trebui să fie citite de fiecare alternativă. Pentru a veni cu aceste costuri, SQL Server trebuie să cunoască dimensiunea fiecărui tabel și distribuția valorilor coloanelor, care sunt disponibile din statisticile asociate datelor. Alți factori luați în considerare sunt consumul CPU și memoria necesară pentru fiecare alternativă de plan. Folosind formule reglate de-a lungul a mulți ani de experiență, toți acești factori sunt sintetizați într-o valoare unică a costului pentru fiecare alternativă și apoi alternativa cu cel mai mic cost este aleasă ca plan de interogare care trebuie utilizat.

Explorarea tuturor acestor alternative poate fi consumatoare de timp și de aceea, odată ce un plan de interogare este creat, este de asemenea stocat în cache pentru reutilizare ulterioară. Solicitările similare viitoare pot sări peste faza de optimizare dacă pot găsi un plan de interogare deja compilat și optimizat în memoria cache internă a SQL Server.

6. Execuţie

Odată ce un plan de interogare este ales, cererea poate începe să se execute. Planul de interogare este tradus într-un arbore de execuție real. Fiecare nod din acest arbore este un operator. Toți operatorii implementează o interfață abstractă cu 3 metode:

- *open(),*
- *next(),*
- *close().*

Bucla de execuție constă în apelarea *open()* pe operatorul care se află la rădăcina arborelui, apoi apelarea *next()* în mod repetat până când returnează false și în final apelarea *close()*. Operatorul de la rădăcina arborelui va apela la rândul său aceeași operație pe fiecare dintre operatorii săi copii, iar aceștia la rândul lor apelează aceleași operații pe operatorii lor copii și așa mai departe. La frunză arborilor există de obicei operatori de acces fizic care preiau efectiv date din tabele și indici. La niveluri intermediare există operatori care implementează diverse operațiuni de date precum filtrarea datelor, efectuarea de JOIN-uri sau sortarea rândurilor. Interogările care utilizează paralelismul folosesc un operator special numit operator de schimb *Exchange*. Operatorul *Exchange* lansează mai multe fire de execuție (sarcini => workers) în execuție și cere fiecărui fir de execuție să execute un subarbore al planului de interogare. Apoi agreghează producția de la acești operatori, folosind un model tipic de mai mulți producători - un singur consumator [4].

Acest model de execuție se aplică nu numai la interogări, dar și la modificarea datelor (inserare, ștergere, actualizare). Există operatori care se ocupă de inserarea unui rând, operatori care se ocupă de ștergerea unui rând și operatori care se ocupă de actualizarea unui rând. Unele solicitări creează planuri banale (de exemplu, *INSERT INTO... VALUES* ...), în timp ce altele creează planuri extrem de complexe, dar execuția este identică pentru toate și are loc exact așa cum am descris: arborele de execuție este repetat apelând *next( )* până când este gata.

Unii operatori sunt foarte simpli, luați în considerare, de exemplu, operatorul *TOP*(N): atunci când *next()* este apelat, tot ce trebuie să apeleze *next()* pe arbore și să țină un numărin contorul intern. După apelarea de N ori, pur și simplu returnează false fără a mai apela pe arbore, terminând astfel iterația acelui sub-arbore.

Alți operatori au un comportament mai complex, luați în considerare ce trebuie să facă un operator de Nested Loops: trebuie să țină evidența poziției iterației buclei atât pe copilul exterior, cât și pe copilul interior, să apeleze *next()* pe copilul exterior, să deruleze copilul interior și apelați *next()* pe copilul interior până când predicatul join este satisfăcut.

Anumiți operatori au un comportament *stop-and-go*, ceea ce înseamnă că nu pot produce nicio ieșire până când nu consumă toate intrările de la operatorii proprii. Exemple de astfel de operatori este SORT: primul apel la *next()* nu revine până când toate rândurile create de operatorii copii sunt preluate și sortate.

Un operator precum *HASH JOIN* va fi atât un comportament complex, cât și de tip stop-and-go: pentru a construi tabelul *hash*, trebuie să apeleze *next()* pe elementul secundar de construcție până când acel operator returnează false. Apoi apelează *next()* pe operatorul arbore din partea sondei până când se găsește o potrivire în tabelul *hash*, apoi revine. Apelurile ulterioare continuă să apeleze *next()* pe operatorul copil din partea probei și să revină la potrivirea tabelului *hash*, până când operatorul secundar *next()* returnează ***false***.

7. Rezultate

Rezultatele sunt returnate în programul client pe măsură ce execuția continuă. Pe măsură ce rândurile „se returneaza" în arborele de execuție, operatorul de top are de obicei sarcina de a scrie aceste rânduri în containerele de rețea și le trimite înapoi către client. Rezultatul nu este creat mai întâi într-un spațiu de stocare intermediar (memorie sau disc) și apoi trimis înapoi către client, ci este trimis înapoi așa cum este creat (pe măsură ce interogarea se execută). Trimiterea rezultatului înapoi către client este, desigur, supusă protocolului de control al fluxului de rețea. Dacă clientul nu consumă în mod activ rezultatul (de exemplu, apelând *SqlDataReader.Read()*), atunci în cele din urmă controlul fluxului va trebui să blocheze partea de trimitere (interogarea care este în curs de executare) și aceasta, la rândul său, va suspenda execuția interogare. Interogarea se reia și produce mai multe rezultate (continua să repete planul de execuție) de îndată ce controlul fluxului de rețea eliberează resursele de rețea necesare.

Un caz interesant este parametrii OUTPUT asociați cu cererea. Pentru a returna valoarea de ieșire înapoi către client, valoarea trebuie să fie inserată în fluxul de date din rețea care curge de la executarea interogării înapoi către client. Valoarea poate fi scrisă înapoi clientului doar la sfârșitul execuției, pe măsură ce cererea se termină. Acesta este motivul pentru care valorile parametrilor de ieșire pot fi verificate numai după ce au fost consumate toate rezultatele.

Concluzii

SQL, T-SQL este un limbaj script nativ pentru comunicarea cu bazele de date. Înțelegerea corecta a formarii si executării unui script pe baza de date ne formează o practica optima de formare a interogărilor, in plus putem înțelege fiecare etapa in executarea si interveni in cazul unor abateri.

Bibliografie

1. SQL Server Protocols [online]. [accesat 20.12.2022]– Disponibil: https://docs.microsoft.com/en-us/openspecs/sql_server_protocols/ms-sqlprotlp/f16558b2-4561-45be-89c9-6f9114514c97
2. Abstract syntax tree [online]. [accesat 20.12.2022]– Disponibil: https://en.wikipedia.org/wiki/Abstract_syntax_tree
3. Transact-SQL [online]. [accesat 20.12.2022]– Disponibil: https://docs.microsoft.com/en-us/sql/t-sql/language-reference?view=sql-server-ver15
4. Execution Plan Caching and Reuse [online]. [accesat 20.12.2022]– Disponibil: https://docs.microsoft.com/en-us/previous-versions/sql/sql-server-2008-r2/ms181055(v=sql.105)?redirectedfrom=MSDN

RISCURILE ASOCIATE CU UTILIZAREA SQL. SQL INJECTION

Cristian DICOL

Departamentul Ingineria Software și Automatică, grupa TI-191 F/R, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Cristian DICOL, e-mail: dicol.cristian@isa.utm.md

Coordonator științific: Dorian SARANCIUC, lector universitar, FCIM, DISA

Rezumat. *În acest articol se spune despre atacurile cibernetice prin injectare SQL. Articolul explică modalitatea în care acest tip de atac este realizat și de ce sunt atacurile prin injectare SQL o amenințare majoră pentru organizațiile de toate dimensiunile. De asemenea, sunt oferite sfaturi pentru organizațiile care doresc să se protejeze împotriva acestor atacuri, precum utilizarea interogărilor parametrizate, validarea și igienizarea intrărilor în aplicații, implementarea măsurilor de protecție la nivel de aplicație și de rețea și antrenarea personalului pentru a recunoaște semnele unui atac posibil.*

Cuvinte cheie: *atac cibernetic, coduri, declarație sql, bază de date, formular web, url, interogări parametrizate, validare, igienizare*

Introducere

Un atac cibernetic cunoscut sub numele de injecție SQL implică inserarea de cod rău intenționat într-o instrucțiune SQL, de obicei prin câmpuri de introducere a utilizatorului, pentru a manipula sau a fura date dintr-o bază de date. Acest lucru se poate face prin completarea unui formular web cu informații eronate sau prin crearea unei adrese URL rău intenționate care conține o instrucțiune SQL rău intenționată [1].

Un script SQL (Structured Query Language) este inserat într-o anumită aplicație printr-o casetă de introducere în acest tip de atac. Altfel spus, un hacker va încerca să insereze o comandă SQL care poate citi datele din baza de date într-un formular de înregistrare dacă găsește unul pe un site web [2].

Un atac reușit va putea insera, actualiza și șterge date din baza de date. Poate îndeplini sarcini administrative.

Cât de dese sunt SQL injections?

Datorită faptului că aceste atacuri trec frecvent neobservate, este dificil să spunem cu certitudine cât de răspândite sunt. Deși sunt un tip de atac bine-cunoscut și folosit frecvent, ele reprezintă totuși o amenințare serioasă pentru afaceri de toate tipurile. Organizațiile trebuie să ia măsuri de precauție pentru a se apăra împotriva aceste amenințări prin protejarea adecvată a bazelor de date și prin implementarea controalelor de validare și dezinfectare a intrărilor în aplicațiile lor [3].

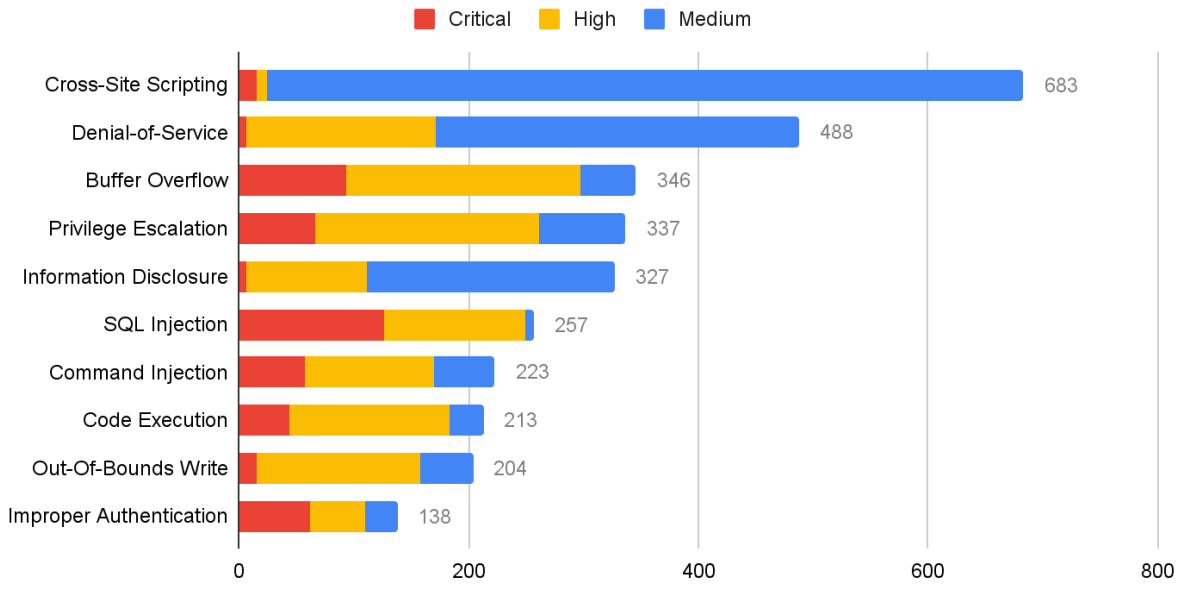


Figura 1 – Tendințe privind securitatea rețelei: noiembrie 2021 până în ianuarie 2022

Cum organizațiile trebuie să se protejeze împotriva lor?

Sunt mai mulți pași pe care organizațiile pot întreprinde pentru a se proteja împotriva atacurilor prin injectare SQL: [4]

a. Utilizați interogări parametrizate: interogările parametrizate sunt o modalitate sigură și eficientă de a preveni atacurile prin injecție SQL. Acestea vă permit să specificați substituenți pentru introducerea utilizatorului în instrucțiunile dvs. SQL, care sunt apoi igienizate și validate automat de serverul bazei de date.
b. Validarea și igienizarea intrărilor: este important să validați și să igienizați toate intrările de utilizator înainte de a le utiliza într-o instrucțiune SQL. Aceasta poate include verificarea caracterelor ilegale, aplicarea limitelor de lungime de intrare și conversia caracterelor speciale în omologii lor escape.
c. Utilizați cel mai mic privilegiu: acordarea utilizatorilor doar a permisiunilor minime de care au nevoie pentru a-și face treaba poate ajuta la prevenirea accesului neautorizat la datele sensibile.
d. Actualizați și corecționați în mod regulat: menținerea la zi a serverului de baze de date și a aplicațiilor cu cele mai recente corecții de securitate poate ajuta la prevenirea exploatării vulnerabilităților de către atacatori
e. Utilizați un paravan de protecție pentru aplicații web (WAF- web application firewall): un WAF este un instrument de securitate care vă poate ajuta să vă protejați site-ul web de atacurile web obișnuite, inclusiv atacurile de injecție SQL.
f. Testați și monitorizați în mod regulat: testarea regulată a aplicațiilor pentru vulnerabilități și monitorizarea activităților suspecte vă poate ajuta să identificați și să răspundeți la potențialele atacuri de injecție SQL în timp util.
g. Prin implementarea acestor și a altor măsuri de securitate, organizațiile își pot reduce semnificativ riscul de a deveni victime ale atacurilor cu injecție SQL.

Exemplul unui atac

De exemplu, să presupunem că există un formular de autentificare pe un site web care cere un nume de utilizator și o parolă. Site-ul web ar putea utiliza o interogare SQL de genul acesta pentru a verifica dacă numele de utilizator și parola introduse sunt valide:

```
SELECT * FROM users WHERE username='$username' AND password='$password';
```

Figura 2 – Valorile introduse de utilizator

Unde, $username și $password sunt variabile ce conțin valorile introduse de utilizator. În cazul în care un atacator introduce un nume de utilizator de ' OR '1'='1 și o parolă de ' OR '1'='1, interogarea SQL rezultată va arăta astfel:

```
SELECT * FROM users WHERE username='' OR '1'='1' AND password='' OR '1'='1';
```

Figura 3 – Declarație evaluată ca adevărată

Deoarece „1"="1" este întotdeauna adevărat, această interogare va fi întotdeauna evaluată drept adevărat, permițând atacatorului să se conecteze fără să cunoască numele de utilizator și parola adecvate.

Deoarece oferă unui atacator acces la date sensibile, capacitatea de a modifica sau elimina date sau chiar de a prelua întreaga bază de date, atacurile cu injecție SQL pot fi extrem de periculoase. Pentru a opri atacurile cu injecție SQL, este esențial să validați și să igienizați corect intrarea utilizatorului.

Figura 4 – SQL injection pe numerele unui automobil

Cele mai importante domenii pentru cercetările privind atacurile cu injecție SQL

Există diverse arii de cercetare care ar putea fi considerate cruciale pentru a aborda atacurile prin injectare SQL în viitor:[5]

a. Tehnici avansate de detectare și prevenire: Experții lucrează în mod constant la dezvoltarea de metode noi și îmbunătățite de detectare și prevenire a atacurilor prin injectare SQL. Acestea ar putea include dezvoltarea de noi algoritmi, modele de învățare automată sau alte tehnologii care pot identifica și bloca mai bine acest tip de atacuri.

b. Sanitizare eficientă a datelor: Pe măsură ce hackerii continuă să găsească noi modalități de a ocoli măsurile de sanitizare existente, cercetarea în metode mai eficiente de sanitizare a intrărilor utilizatorilor ar putea fi un domeniu important de concentrare.

c. Îmbunătățirea educației în securitate: În timp ce soluțiile tehnice sunt importante, este crucial ca organizațiile să înțeleagă în profunzime cum să se protejeze împotriva atacurilor prin injectare SQL. Cercetările viitoare ar putea concentra dezvoltarea de metode mai eficiente de educare și formare a angajaților în ceea ce privește modul de a preveni și de a răspunde acestui tip de atacuri.

d. Înțelegerea mai profundă a motivațiilor și tacticilor atacatorilor: Prin înțelegerea motivațiilor pentru care atacatorii vizează anumite organizații și a tacticilor pe care le folosesc pentru a efectua atacuri prin injectare SQL, cercetătorii pot informa mai bine dezvoltarea de strategii de prevenire și răspuns.

Concluzii

Injectarea SQL este un tip de atac cibernetic prin care un atacator introduce coduri cu intenții rele într-o declarație SQL pentru a manipula sau a fura date dintr-o bază de date. Acest lucru poate fi realizat prin intermediul formularelor web sau prin URL-uri malițioase care conțin declarații SQL cu intenții rele. Atacurile cu injectare SQL sunt frecvente și pot rămâne adesea nedetectate, fiind o amenințare majoră pentru organizațiile de toate dimensiunile. Organizațiile pot lua măsuri pentru a se proteja împotriva acestor atacuri prin securizarea bazelor de date, utilizarea interogărilor parametrizate, validarea și igienizarea intrărilor în aplicații, implementarea măsurilor de protecție la nivel de aplicație și de rețea și antrenarea personalului pentru a recunoaște semnele unui atac posibil.

În general, este nevoie de cercetări continue pentru a ține pasul cu amenințarea crescândă a atacurilor prin injectare SQL și pentru a proteja mai bine organizațiile și indivizii de acest tip

Referințe

1. SQL Injection - Ghidul celei mai utilizate metode de atac. [online]. [accesat 20.12.2022]– Disponibil: httpswww.link-academy.comblogsql-injection-ghidul-celei-mai-utilizate-metode-de-atac#~text=Ce%20este%20SQL%20Injection%3F,intermediul%20unei%20căsuțe%20de%20input.
2. SQL Injection [online]. [accesat 20.12.2022]– Disponibil: https://owasp.org/www-community/attacks/SQL_Injection
3. How to prevent SQL injection with prepared statements [online]. [accesat 20.12.2022]– Disponibil: https://searchsecurity.techtarget.com/definition/SQL-injection
4. What is SQL Injection (SQLi) and How to Prevent It [online]. [accesat 20.12.2022]– Disponibil: https://www.acunetix.com/websitesecurity/sql-injection/
5. SQL Injection: Vulnerabilities & SQL Injection Prevention [online]. [accesat 20.12.2022]– Disponibil: https://www.veracode.com/security/sql-injection

ИСПОЛЬЗОВАНИЕ МАШИННОГО ОБУЧЕНИЯ В УПРАВЛЕНИИ БАЗАМИ ДАННЫХ

Максим СТРОПША

Департамент Программной Инженерии и Автоматики, группа TI-192 F/R, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Максим СТРОПША, e-mail: stropsa.maxim@iis.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Резюме: *Доклад посвящен применению машинного обучения для управления базами данных. Традиционные методы управления данными (например, разделение данных, настройка ручек, переписывание запросов, оценка стоимости) носят эмпирический характер и в значительной степени зависят от вмешательства человека. Эти методы не способны удовлетворить требования высокой производительности для различных приложений и масштабируемости. Объем корпоративных данных увеличивается, и очень важно найти эффективные способы их обработки. Машинное обучение может автоматизировать и оптимизировать процессы управления базами данных.*

Ключевые слова: машинное обучение, управление базами данных, большие данные, оптимизация производительности.

Введение

Традиционные методы управления данными (например, разделение данных, настройка ручек, переписывание запросов, оценка стоимости) носят эмпирический характер и в значительной степени зависят от вмешательства человека для настройки и обслуживания баз данных. Однако существующие эмпирические методы не способны удовлетворить требования высокой производительности для различных приложений, крупномасштабных экземпляров баз данных и различных пользователей [1].

Ожидается, что общий объем корпоративных данных, генерируемых каждый год, увеличится с 18,2 ЗБ в 2015 г. до 175,8 ЗБ к 2025 г. Согласно опросу Seagate, только 32% этих данных доступны в состоянии готовности к анализу. Поскольку объемы данных растут быстрее, чем предприятия могут за ними угнаться, очень важно найти эффективные способы хранения, управления и анализа данных. Один из подходов заключается в использовании машинного обучения (ML) для улучшения структуры базы данных и оптимизации запросов [2].

Цель статьи - понять, как машинное обучение может влиять на стратегию проектирования хороших баз данных и оптимизировать операции с базами данных. Понимание современного состояния этих областей может помочь бизнесу управлять и извлекать полезную информацию из собранных данных.

Проект адресует три вопроса, связанные с базами данных: создание эффективного индекса, неэффективные запросы JOIN и возможность выполнять моделирование и анализ внутри базы данных, а не платить за передачу данных в внешний инструмент.

Связанная работа

В статье 2000 года Xindong Wu [3] идентифицировал будущие вызовы в комбинации машинного обучения и технологии баз данных для создания систем Интеллектуального обучающего баз данных (ILDB). Wu [3] идентифицировал две основные области, которые фокусируются на эффективном представлении данных и методах машинного обучения, способных обрабатывать системы баз данных предприятий. Он также отметил, что реальные

базы данных обычно шумны и содержат отсутствующие или неверные данные, что представляет собой вызов для систем обучения. Хотя статья Wu [3] не современная, она предоставила основное представление возможных проблем для будущего.

В обзоре литературы 2015 года Najafabadi [4] идентифицировали возможные проблемы при адаптации текущих методов машинного обучения к современным сценариям с базами данных. Авторы определили следующие области: потоковые данные(streaming data), многомерные данные(high-dimensional data), масштабируемость моделей и распределенных вычислений. Эти области связаны с проблемами алгоритмов машинного обучения, а не с проблемами использования машинного обучения для создания более совершенных систем баз данных.

Индексация

Разработчики баз данных создают структуру данных, называемую индексом, для упорядочения данных, чтобы к ним можно было быстро получить доступ для эффективного выполнения запросов. Тип используемого индекса определяется как структурой данных, так и типами запросов, которые будут выполняться чаще всего. Наиболее распространенными индексами являются BTree-Index для поиска диапазонов, Hash-Index для одиночного поиска и Index для определения наличия ключа.

Все эти индексы создаются с фиксированными структурами данных, которые не меняются в ответ на основные шаблоны данных. Бейтель и др. утверждали, что эти индексы не используют шаблоны данных для дальнейшей оптимизации работы базы данных.

Бейтель и др. продемонстрировали повышение скорости выполнения запросов на 44% и сокращение памяти на 75%, необходимой для хранения индекса с использованием индексов нейронной сети, по сравнению с BTree-Index в базе данных только для чтения. Использование индексов на основе ML имеет некоторые недостатки, такие как необходимость планировать ложноотрицательные результаты (т. е. индекс ML сообщает, что ключ не существует, когда он существует), которые не встречаются с традиционными индексами, и индексы ML возвращают приблизительные местоположения данных, а не точные местоположения записей.

Для решения этих проблем Бейтель и др. объединять изученные индексы с традиционными индексами, позволяя более быстрому индексу ML выполнять большую часть поиска, в то время как традиционный индекс гарантирует правильность поиска. Хотя в этом исследовании рассматривались только базы данных только для чтения, авторы утверждают, что индекс онлайн-обучения ML может обрабатывать доступные для записи базы данных.

Рабочие нагрузки на системы управления базами данных (СУБД) обычно меняются со временем. Например, приложение для отслеживания автобусов может иметь высокую рабочую нагрузку в течение дня, когда автобусы ходят, и мало запросов ночью, когда автобусы не ходят.

Согласно Ma[4] и соавт., СУБД, которая может прогнозировать свою рабочую нагрузку, может оптимизировать свои индексы, чтобы работать лучше по мере изменения рабочей нагрузки и работать автономно. Администраторы-люди несут ответственность за ручную настройку оптимизации в текущей СУБД. Ma [4] и др. представить QueryBot 5000, структуру прогнозирования рабочей нагрузки, которая предсказывает будущие модели рабочей нагрузки с использованием ансамблевой модели линейной регрессии, рекуррентной нейронной сети и кластеризации.

Кластеризация запросов имеет решающее значение для снижения сложности моделирования рабочей нагрузки и обеспечения эффективного прогнозирования. Ma и др. обнаружил, что пять крупнейших кластеров могут моделировать 95% рабочей нагрузки экспериментальных систем. Эта структура прогнозирует рабочую нагрузку как в краткосрочной, так и в долгосрочной перспективе, позволяя автоматизированной СУБД выбирать наилучшие оптимизации.

В отличие от оптимизирующей платформы Бейтель и др., платформа QueryBot 5000 использует онлайн-модели машинного обучения, которые адаптируются к изменениям

поступающих рабочих нагрузок в режиме реального времени. Одним из ограничений этой структуры является то, что по-прежнему требуется некоторая ручная настройка порога, при котором рабочая нагрузка изменилась достаточно, чтобы вызвать необходимость повторной кластеризации.

Индексация в базах данных - исследования включают использование машинного обучения для создания эффективных индексов. Цель состоит в удалении зависимости DBMS от экспертов в домене для выбора индексов, и вместо этого дизайн индексов основан на схеме базы данных и нагрузке, которую она получает. Архитектура, названная SmartIX, абстрагирует задачу анализа и принятия решения о том, будут ли кандидатские столбцы для индекса улучшать производительность, задачу, которую обычно выполняют люди. Вместо этого SmartIX содержит агента обучения по награде, который исследует возможные кандидатские столбцы для индекса. Он также содержит среду, где агент получает награды за свои решения и в которой его следующие действия передаются в третью компоненту - интерфейс DBMS, который записывает команды для изменения базы данных и также читает статистику о текущей производительности индексов. Обучение агента было проведено с использованием двухслойной нейронной сети перцептрона за 64 000 шагов времени, где статистика обучения собиралась каждые 128 шагов и использовалась для предоставления графического представления результатов. При сравнении производительности конфигурации индекса, созданной SmartIX, с конфигурациями, созданными генетическими алгоритмами и др

Оптимизация запросов на соединение

Объединение нескольких таблиц вместе для построения результата запроса — одна из самых затратных операций с базой данных. Операции соединения являются дорогостоящими, поскольку база данных должна определить наиболее эффективный порядок таблиц, участвующих в объединении. Это решение обычно принимается с использованием эвристики, чтобы сузить область поиска.

) был разработан Марклом и его коллегами в 2003 году как первый пример оптимизатора запросов, который использует машинное обучение для повышения производительности за счет изучения исторических данных.

Чтобы исправить ошибки моделирования в предыдущих планах выполнения запросов, постепенно корректирует оценки кардинальности и избирательности для предикатов. LEO может обнаруживать ошибки оценки во время выполнения запроса и автоматически повторно оптимизировать запрос. LEO работает, регистрируя фактические мощности и учась уменьшать разницу между фактическими и прогнозируемыми мощностями. Проблема сходимости, в которой скорость обучения модели сильно зависит от данных и предположений о данных, была одной из проблем, с которыми столкнулись Маркл и его коллеги, это все еще распространено в современном машинном обучении.

Авторы решили эту проблему, позволив LEO работать в двух режимах: исследовательском, который выбирает рискованные, но недорогие планы на основе предположений, и конкретном, который отдает предпочтение планам, основанным на опыте и реальных фактах. LEO также решает проблему коррелированных соединений, обнаруживая разницу между предсказанным и фактическим числом элементов. Еще одна проблема, поднятая авторами, заключается в том, чтобы определить, когда стоит повторно оптимизировать запрос в середине выполнения из-за нелинейности в модели затрат, проблема, позже решенная Кришнаном и др.

Лю и др. использовали трехслойную нейронную сеть для оптимизации запросов на соединение. Чтобы сгенерировать начальные обучающие запросы для своей нейронной сети, они разработали собственный алгоритм.

Нейронная сеть может считывать из запросов только нестрогие операторы диапазона, такие как "=" или ">=", но они разработали методы для перезаписи всех других реляционных операторов, таких как "=" и "" или ">". Они сравнили нейронную сеть с оптимизатором

запросов, встроенным в IBM DB2, систему управления базами данных. Результаты показали, что нейронные сети превзошли методы оценки, основанные на статистике, при оценке кардинальности запросов.

Однако Лю и соавт. заявили, что хотели бы проделать больше работы, например, иметь возможность сравнивать столбцы с операторами отношения, потому что на данный момент можно сравнивать значения столбцов только с константами (c1 >= 3, но не c1 >= c2). Во-вторых, они хотели поэкспериментировать с различными типами нейронных сетей, такими как RNN и глубокие нейронные сети. В более поздних разработках в этой области использовались более сложные модели.

Было показано, что оптимизатор соединения RL создает лучшие планы оптимизации, а также работает более чем в десять раз быстрее, чем традиционные методы оптимизации, такие как левосторонняя эвристика. Одним из ограничений этого исследования является то, что авторы рассматривают только один тип операции соединения, который составляет большинство протестированных тестов.

Моделирование в базе данных

Другая интересная область исследования - разработка новых языков программирования для создания моделей машинного обучения в базе данных. Эти модели могут использовать близость данных и избежать необходимости длительных конвейеров для перемещения данных в другое хранилище для анализа.

MADlib — это аналитическая библиотека для Greenplum и SQL разработал Хеллерштейн. MADlib поддерживает реализация традиционных алгоритмов машинного обучения такие как SVM и K-means. Он также поддерживает множество контролируемых и неконтролируемых алгоритмов машинного обучения, является открытым исходным кодом. Его кодовая база получает вклад как от промышленности, так и от научных кругов. Цель авторов заключалась в том, чтобы «ускорит инновации и передачу технологий в Сообщество Data Science через общую библиотеку масштабируемой аналитики базы данных». Машинное обучение и другие продвинутые методы анализа с MADlib реализованы с использованием простые SQL-скрипты. Чтобы запустить последовательность этих операторов SQL или выполнить итеративный анализ, можно также использовать файл драйвера на основе Python для запуска MADlib. Пользователи также имеют доступ к шаблонным запросам, которые помогают, когда схема для входные таблицы модуля не обязательно фиксированы. MADlib нашла применение в сотрудничестве с исследователями из Университета Висконсина, а также из Университета Флориды, и в Беркли. MADlib еще не нашел коммерческого применения. Другой похожий язык — MLog, разработанный Ли, декларативный язык программирования, который взаимодействует непосредственно с базой данных. MLog улучшает MADlib на позволяя пользователям создавать более сложные модели. MLog позволяет пользователям создавать сложные модели глубокого обучения в базы данных таким же образом, как пользователи могут открывать сеанс SQL и выполнять простые запросы к данным на месте. Глубокий модели обучения обычно требуют большого объема обучения данные должны быть точными. MLog позволяет создать модель где находятся данные, экономя время и упрощая процесс построения модели. Программное обеспечение работает, просматривая данные как тензоры, и включает в себя функциональные возможности для автоматической оптимизации процесса построения модели. Модели машинного обучения, созданные с помощью Декларативный язык запросов MLog может быть скомпилирован в нативные программы TensorFlow с сопоставимой производительностью до оптимизированных вручную моделей. Авторы обнаружили, что в некоторых случаях программы TensorFlow, созданные с помощью MLog, все еще могут быть до двух раз медленнее, чем код, настроенный вручную. Другое ограничение заключается в том, что язык MLog не был полностью интегрирован с SciDB.

Вызовы

Ван и др. обсуждает, как обе системы глубокого обучения может быть улучшена с точки зрения базы данных и как можно улучшить приложения баз данных с помощью глубокого обучения методы. Они обнаружили, что глубокое обучение может использовать преимущества методов оптимизации, используемых в области баз данных. в обучении своих моделей. Такие методы, как операция планирование, управление памятью, параллелизм и параллелизм/непротиворечивость. Точно так же поле базы данных может использовать обработка естественного языка, отличительная черта глубокого обучения, интерпретировать неформальные запросы к базе данных и переводить их в формальные запросы. Кроме того, планирование запросов, традиционно сделать с использованием сложных моделей затрат в базе данных, вместо этого может быть сделать более эффективно с помощью модели RNN. Наконец, глубокое обучение может помочь в анализе пространственно-временных данные, хранящиеся в базах данных, путем сопоставления проблемы с образную структуру, а затем с помощью сверточной нейронной сеть для использования пространственной локализации данных. В общем, обмен методами и представлениями между сообщества баз данных и глубокого обучения принесут большую пользу обеим областям. Guo и Daudjee описывают ограничения недавнего использования глубокое обучение для оптимизации запроса на соединение. Они обнаружили, что в худшем случае эти модели создают планы выполнения, которые на порядки дороже, чем самый дешевый план и может даже привести к сбою СУБД, требуя слишком много памяти.

Они утверждают, что дополнительное время, необходимое для выполнения худшего планирует, что результаты модели могут перевесить сэкономленное время с помощью глубокого обучения. Это указывает на необходимость уравновешивания высокая производительность с последовательностью. Авторы считают, что необходимы дополнительные исследования для повышения надежности модели для это приложение. Авторы также утверждают, что лучшая функция кодирование необходимо для расширения запроса на соединение с глубоким обучением оптимизаторы к корпоративным базам данных со многими таблицами, которые могут часто меняются, что требует переобучения модели. Они утверждают эта функция кодирования, устойчивая к изменениям таблицы, должна быть требованием для будущих исследований.

Заключение

Хотя машинное обучение может значительно повысить эффективность и автономию базы данных, все еще существуют проблемы с коммерчески жизнеспособностью этих гибридных систем. Применение машинного обучения для улучшения проектирования и эксплуатации баз данных — это новая и быстро развивающаяся область исследований.

Библиография

1. Machine Learning for Data Management: A System View – [online]. [20.12.2022]– Доступен: http://dbgroup.cs.tsinghua.edu.cn/ligl/papers/icde22-tutorial-paper.pdf, http://www.ijecs.in/index.php/ijecs/article/view/4520
2. 2020. Rethink Data: Put More of Your Business Data to Work—From Edge to Cloud. Technical Report. Seagate. 56 pages. [online]. [20.12.2022]– Доступен: https://www.seagate.com/files/www-content/our-story/ rethink-data/files/Rethink_Data_Report_2020.pdf
3. Xindong Wu. 2000. Building Intelligent Learning Database Systems. AI Magazine 21, 3 (Sept. 2000), 61–68. https://doi.org/10.1609/aimag. v21i3.1524 Number: 3.
4. Maryam M. Najafabadi, Flavio Villanustre, Taghi M. Khoshgoftaar, Naeem Seliya, Randall Wald, and Edin Muharemagic. 2015. Deep learning applications and challenges in big data analytics. Journal of Big Data 2, 1 (Feb. 2015), 1. https://doi.org/10.1186/s40537-014-0007-7

SQL SERVER INTEGRATION SERVICES VS AZURE DATA FACTORY

Xenia DAMANCIUC

Departamentul Ingineria Software și Automatică, grupa TI-191 F/R, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Xenia DAMANCIUC, e-mail: damanciuc.xenia@gmail.com; damanciuc.xenia@isa.utm.md

Conducător științific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *Pentru dezvoltarea unei aplicații de succes este necesar să avem pus la bază un sistem care este considerat sursa centralizată de adevăr pentru informația predispusă. Întru realizarea acestui scop se folosește așa numitul proces ETL, care semnifică „extrage, transformă și încarcă”. În cadrul acestui articol veți avea posibilitatea să înțelegeți care sunt avantajele și dezavantajele prin implementarea unui ETL utilizând soluția SQL Server Integration Services în comparație cu Azure Data factory.*

Cuvinte cheie*: date, serviciu cloud, serviciu on-premise, ETL*

Introducere

Fie că stăm la baza dezvoltării unei aplicații scopul căreia va fi să colecteze date despre angajații unei companii și să le ofere altor aplicații interne sau externe. Este important de menționat faptul că este necesar de integrat o multitudine de aplicații, care oferă seturile de date în diferite formate. Pentru realizarea acestei solicitări se va implementa o soluție, scopul căreia va fi să extragă datele de la aplicațiile-producător, ulterior să poată procesa toate formatele predispuse, iar ca rezultat va încărca rezultatul final într-un depozit de date. Acest proces este numit ETL sau „extrage, transformă, încarcă”. Astfel, există diferite tipuri instrumente de implementare a acestuia:[1]

- *open-source* – instrumente ce sunt gata pentru a fi integrate cu alte sisteme. Sunt potrivite pentru organizațiile cu buget restrâns.
- *de procesare în serie* – instrumente ce vor realiza procesele în situațiile în care puterea de calcul este mai liberă.
- *bazate pe tehnologii cloud* – instrumente care pot colecta date de la orice serviciu cloud și le pot încărca în depozitul de date solicitate prin optimizarea eficientă conform cerințelor sistemului Dvs.
- *în timp real* – instrumente care permit obținerea datelor actualizate în timp real. Indiferent de volum, înregistrările relevante pot fi mutate instantaneu în destinațiile potrivite.

SQL Server Integration Services (SSIS) este o platformă care vă permite să creați soluții de integrare și transformare a datelor la nivel de întreprindere. SSIS este o tehnologie de depozitare a datelor care poate fi utilizată pentru extragerea, încărcarea și transformările de date, cum ar fi curățarea, agregarea și combinarea datelor [2].

Datele pot fi extrase și transformate din mai multe surse, inclusiv fișiere de date XML, fișiere plate și surse de date relaționale, și apoi încărcate în una sau mai multe destinații folosind Integration Services.

Azure Data Factory este serviciul ETL cloud al Azure pentru integrarea datelor fără server și transformarea datelor. Oferă o interfață de utilizare fără cod pentru crearea intuitivă și monitorizarea și gestionarea pe un singur panou.

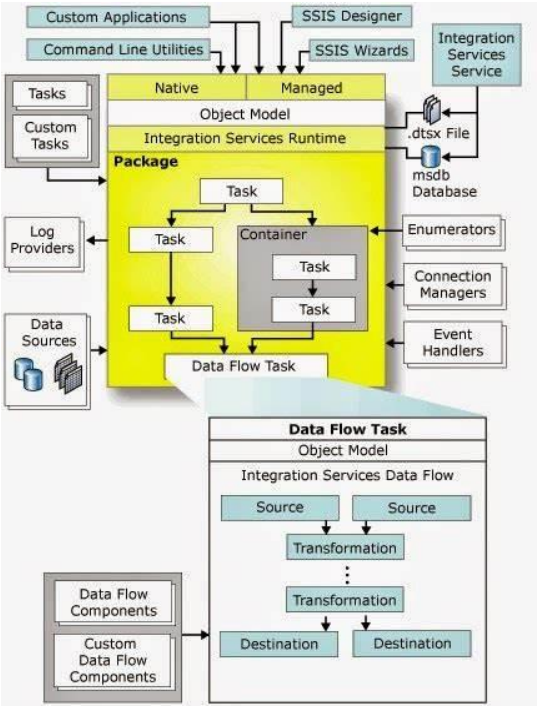
Prin urmare, vom studia avantajele și dezavantajele instrumentului de procesare în serie – SQL Server Integration Services (SSIS) în comparație cu instrumentul bazat pe tehnologie cloud – Azure Data Factory (ADF).

Părțile componente ale SSIS și ADF

SSIS este un instrument ETL (extract-transform-load). Este conceput pentru a extrage date din una sau mai multe surse, a transforma datele din memorie - în fluxul de date - și apoi a scrie rezultatele la o destinație.

SSIS are patru componente importante:

- *Flux de control* – Este partea de bază a pachetului SSIS. Aceasta stabilește ordinea tuturor componentelor, cum ar fi containerele, sarcinile etc.
- *Flux de date* – această componentă vă permite să extrageți, să transformați și apoi să încărcați datele într-o altă destinație.
- *Pachete* – colecție de control și flux de date. Containerele și sarcinile fluxului de date din fluxul de control și sursele, și destinațiile din fluxul de date sunt cunoscute în totalitate ca pachet.
- *Parametri* – Acestea sunt tipuri speciale de variabile. Ele ajută la ușurarea procesului de transmitere a valorilor de rulare către pachetele SSIS.



Figură 1 - Părțile componente ale unui SSIS

ADF, pe de altă parte, este mai mult un instrument ELT (extract-load-transform), adică pentru extragerea datelor dintr-o sursă și scrierea ulterioară într-o altă sursă. Există posibile transformări de date în timpul acestui transfer, cum ar fi conversia dintr-un format de fișier în altul, dar acestea sunt destul de limitate. Acesta poate fi comparat cu un flux de control în SSIS.

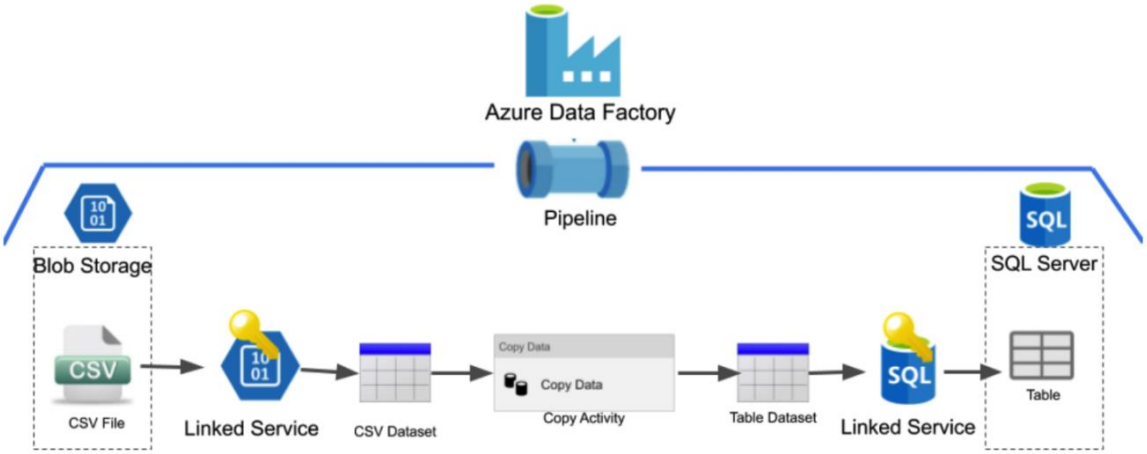
ADF la rândul său, de asemenea, constă din patru componente:

Pipeline – Conține sarcinile pe care doriți să le executați. Acesta definește fluxul de lucru complet, cum ar fi ce sarcini trebuie efectuate și în ce ordine.

Activitate – Aceștia sunt pașii individuali din interiorul unei pipeline, în care fiecare activitate îndeplinește o singură sarcină. Ele pot fi fie legate cu lanțuri, fie rulate în mod paralel și controlează fluxul în interiorul unui pipeline.

Seturi de date – Datele colectate sunt necesare ca intrare pentru procesul ETL. Acestea sunt vizualizări care reprezintă baze de date, fișiere sau mape.

Servicii legate – Acestea sunt practic șirurile de conexiune utilizate pentru a conecta sursele și serviciile de date și pentru a le autentifica.



Figură 2 - Părțile componente ale unui ADF

ADF vs SSIS

Ambele instrumente au descrieri similare, deoarece au fost create în același scop. Dar pentru a înțelege diferențele dintre ele, să le comparăm.

Curbă de învățare

Ambele instrumente sunt ușor de învățat și oferă posibilitatea îndeplinirii sarcinilor simple. Dar pentru a stăpâni orice software nou, este nevoie de mult timp și practică. SSIS este un software matur cu puține modificări majore în ultimii ani. De asemenea, acesta are la bază o documentație bine stabilită, precum probleme și răspunsuri comune deja analizate și discutate. Prin urmare, este ușor de învățat în comparație cu Azure Data Factory.

ADF, pe de altă parte, este încă în evoluție. Multe caracteristici și capabilități trebuie încă să fie lansate. În cazul apariției unei probleme este mai greu să fie găsită o soluție rapidă. Cu toate acestea, dacă aveți deja cunoștințe în SSIS, este mult mai ușor să-l învățați și să aplicați lucrurile deja cunoscute.

Viteza datelor

SSIS este un instrument ETL de procesare în serii. Acest lucru se realizează prin gruparea rândurilor care urmează să fie procesate în serii, apoi procesează fiecare serie sau pachet și actualizează fiecare grup așa cum este procesat.

ADF acceptă atât procesarea în serie, cât și procesarea fluxului (stream processing). Poate procesa date pe baza evenimentelor care au loc în conturile de stocare, cum ar fi ștergerea sau sosirea fișierelor. De asemenea, acceptă declanșatoare, unde puteți trece ora de început și de sfârșit pentru fiecare fereastră de timp din interogarea dvs.. Astfel, procesarea și returnarea datelor se va întâmpla între acel interval. În plus față de acestea, ADF acceptă și declanșatoare în serie.

Programabilitate

SSIS are un SDK de programare. Astfel, permite dezvoltatorilor să scrie propriul cod pentru definirea obiectelor de conexiune, sarcinilor, furnizorilor de jurnal și transformărilor. Dispune de un model de obiect programabil care permite dezvoltatorilor să creeze, să stocheze și să încarce pachete folosind BIML, precum și să creeze, să distrugă și să modifice oricare dintre obiectele conținute.

Pe de altă parte, ADF nu are un SDK de programare nativ, dar are automatizare folosind PowerShell fără a implica componente terțe. Pentru a rula conductele manual, puteți utiliza și metode precum .NET SDK, RestAPI-uri, Python SDK-uri.

Prețuri

SSIS vine ca parte a licenței SQL Server. Prețul este gratuit pentru edițiile Express și Developer, dar pentru Enterprise, costă 14.256 USD per nucleu.

Azure Data Factory oferă servicii la prețuri cu plata pe măsură. Acesta este calculat pe baza numărului de rulări de orchestrare a conductelor, a execuției fluxului de date și a depanării, precum și a numărului de operațiuni din fabrica de date, cum ar fi monitorizarea pipeline-ului.

Tabelul 1

Argumente PRO și CONTRA

Argumente/Serviciu	SSIS	ADF
PRO	• Capabil să gestioneze date dintr-o varietate de surse de date. • Oferă funcționalitate de transformare • C# sau VBA pot fi folosite extinderea funcționalului • Ușor de învățat și ușor de utilizat datorită documentației închegate • Capacitățile de depanare sunt excelente, în special în timpul execuției fluxului.	• Oferă o soluție fără server care elimină sarcinile banale de întreținere și actualizare a software-ului. • Suporta integrarea cu mai mulți conectori terți. • Acceptă interogări lungi și consumatoare de timp. • Foarte scalabil și rentabil. • Crearea pipeline-urilor este mult mai ușoară
CONTRA	• Raportul de execuție a pachetului poate fi văzut numai prin Management Studio. • Rularea mai multor pachete în paralel este dificilă. Evoluție lentă • Lucrul cu seturi de date nestructurate poate fi dificil.	• Mai puține funcții de transformare native în comparație cu SSIS. • Nu are instrumente de depanare. • Necesita menținerea unei strategii de facturare perfectă, altfel va duce la costuri excesive. • Lipsa flexibilității în comparație cu alte instrumente ETL, de ex. Componenta de script C#.

Concluzii

SSIS și ADF sunt ambele instrumente ETL foarte capabile. Fiecare poate reuși atunci când este utilizat corespunzător. Dacă volumul de lucru este în mare parte on-premises sau procesele ETL rulează în mod constant pe tot parcursul zilei, atunci SSIS este soluția potrivită. Dar dacă doriți să plătiți doar pentru resursele utilizate și cea mai mare parte a volumului de muncă este în cloud, atunci ADF este o alegere potrivită pentru dvs. Având în vedere factorii discutați mai sus și ceea ce este cerut de proiectul dvs., puteți decide care este instrumentul potrivit pentru lucrare, dacă SSIS, ADF sau un hibrid dintre cele două.

Bibliografie

1. Choosing Between SQL Server Integration Services and Azure Data Factory. [online] [accesat 22.11.2022]. Disponibil: https://www.mssqltips.com/sqlservertip/7094/azure-data-factory-vs-ssis-similarities-differences/
2. Comparing SSIS and Azure Data Factory. [online] [accesat 22.11.2022]. Disponibil: https://www.timmitchell.net/post/2020/07/16/comparing-ssis-and-azure-data-factory/

TRANZACȚIILE ÎN SQL

Mariana TROFIN

Departamentul Ingineria Software și Automatică, grupa TI-191 F/R, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Mariana TROFIN, e-mail: frunze.mariana@isa.utm.md

Coordonator științific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *În articolul științific este descrisă utilizarea tranzacțiilor în SQL ca un instrument esențial pentru asigurarea integrității și fiabilității bazei de date. Tranzacțiile permit utilizatorilor să efectueze mai multe operații asupra bazei de date într-o singură unitate logică, cum ar fi adăugarea, modificarea sau ștergerea datelor. Cele trei comenzi de bază în tranzacții sunt COMMIT, ROLLBACK și SAVEPOINT, fiecare având rolul lor în gestionarea tranzacțiilor. Nivelele de tranzacții sunt: READ UNCOMMITTED, READ COMMITTED, REPEATABLE READ și SERIALIZABLE, fiecare oferind un anumit nivel de protecție împotriva modificărilor concurente și a pierderii datelor.*

Cuvinte cheie*: tranzacții, commit, rollback, savepoint, nivelurile de izolare*

Tranzacțiile în SQL

Tranzacțiile în SQL sunt un mecanism esențial care permite efectuarea mai multor operații asupra bazei de date într-o singură unitate logică. Acestea sunt utilizate pentru a gestiona și a controla modificările efectuate asupra bazei de date, astfel încât să poată fi anulate orice modificări în cazul unei erori sau a unei probleme.

Tranzacțiile sunt gestionate prin intermediul unor comenzi specifice, cum ar fi COMMIT, care confirmă toate modificările efectuate în cadrul tranzacției, ROLLBACK, care anulează toate modificările efectuate în cadrul tranzacției, și SAVEPOINT, care permite crearea unui punct de salvare în cadrul tranzacției pentru a putea anula modificări specifice [1].

Utilizarea tranzacțiilor în SQL este esențială pentru orice aplicație care se bazează pe o bază de date, deoarece acestea asigură că orice modificare efectuată este validată și că baza de date rămâne întotdeauna în starea sa corectă.

Cum ajută la gestionarea și controlul modificărilor bazei de date

Tranzacțiile sunt un mecanism important pentru a gestiona și controla modificările bazei de date. Ele oferă următoarele beneficii: **atomicitate**, **consistență, izolare**, **durabilitate** [2].

Atomicitate: O tranzacție este o unitate atomică de lucru, care este executată în întregime sau nu este executată deloc. Aceasta înseamnă că, dacă o tranzacție conține mai multe modificări, toate acestea vor fi aplicate împreună sau niciuna dintre ele nu va fi aplicată.

Consistență: O tranzacție va menține baza de date într-o stare consistentă, indiferent dacă a fost executată sau nu.

Izolare: O tranzacție este izolată de alte tranzacții care sunt în curs de executare. Aceasta înseamnă că o tranzacție nu va afecta alte tranzacții sau va fi afectată de alte tranzacții.

Durabilitate: O tranzacție care a fost executată cu succes va fi permanent aplicată bazei de date.

Aceste caracteristici ale tranzacțiilor ajută la gestionarea și controlul modificărilor bazei de date, asigurând că baza de date este întotdeauna într-o stare consistentă, izolată și durabilă.

Exemple reale: COMMIT, ROLLBACK și SAVEPOINT

COMMIT, ROLLBACK și SAVEPOINT sunt operațiuni utilizate pentru a gestiona tranzacțiile în bazele de date [3].

COMMIT

COMMIT este utilizat pentru a confirma modificările efectuate într-o tranzacție și pentru a le face permanente în baza de date. Acest lucru înseamnă că modificările devin vizibile pentru toate tranzacțiile care vin după aceasta și nu mai pot fi revocate.

Iată un exemplu de instrucțiune COMMIT în SQL:

```
BEGIN TRANSACTION;
UPDATE employees SET salary = salary + 1000 WHERE department = 'sales';
COMMIT;
```

În acest exemplu, o tranzacție este inițiată cu instrucțiunea "BEGIN TRANSACTION". Apoi, salariul tuturor angajaților din departamentul "vânzări" este majorat cu 1000. Tranzacția este apoi confirmată cu instrucțiunea "COMMIT", ceea ce face ca actualizarea să fie permanentă în baza de date.

ROLLBACK

ROLLBACK este utilizat pentru a anula modificările efectuate într-o tranzacție. Acest lucru înseamnă că toate modificările efectuate în tranzacție sunt șterse, iar baza de date este readusă la starea sa anterioară.

Iată un exemplu de instrucțiune ROLLBACK în SQL:

BEGIN TRANSACTION;

UPDATE employees SET salary = salary + 1000 WHERE department = 'sales';

ROLLBACK;

În acest exemplu, o tranzacție este inițiată cu instrucțiunea "BEGIN TRANSACTION". Apoi, salariul tuturor angajaților din departamentul "vânzări" este majorat cu 1000. Cu toate acestea, dacă dintr-un motiv oarecare decidem că această actualizare nu ar trebui să fie confirmată, putem utiliza instrucțiunea "ROLLBACK" pentru a anula modificările efectuate în cadrul tranzacției [4].

Trebuie remarcat faptul că, instrucțiunea "rollback" va anula toate modificările efectuate de la ultima instrucțiune de confirmare, prin urmare, dacă aveți mai multe instrucțiuni în cadrul unei tranzacții și doriți să anulați doar una dintre ele, trebuie să utilizați puncte de salvare.

SAVEPOINT

SAVEPOINT este utilizat pentru a crea un punct de salvare în cadrul unei tranzacții. Acest lucru permite anularea doar a unui subset de modificări efectuate în tranzacție, în loc să anulați toată tranzacția.

Iată un exemplu de utilizare a unei instrucțiuni SAVEPOINT în SQL:

```
BEGIN TRANSACTION;
UPDATE employees SET salary = salary + 1000 WHERE department = 'sales';
SAVEPOINT sp1;
UPDATE employees SET department = 'marketing' WHERE department = 'sales';
SAVEPOINT sp2;
UPDATE employees SET address = 'New York' WHERE department = 'marketing';
ROLLBACK TO sp1;
```

În acest exemplu, o tranzacție este inițiată cu instrucțiunea "BEGIN TRANSACTION". Apoi, salariul tuturor angajaților din departamentul "vânzări" este majorat cu 1000. După aceea, se creează un punct de salvare "sp1". Apoi, departamentul tuturor angajaților din departamentul "vânzări" este schimbat în "marketing" și se creează un alt punct de salvare "sp2". În cele din urmă, adresa tuturor angajaților din departamentul "marketing" este schimbată în "New York". Cu toate acestea, decidem că nu dorim ultima actualizare, așa că folosim instrucțiunea "ROLLBACK TO" pentru a anula modificările efectuate după ce a fost creat punctul de salvare "sp1".

Prin urmare, efectul final va fi următorul:

Salariul tuturor angajaților din departamentul "vânzări" este mărit cu 1000.

Departamentul tuturor angajaților din departamentul "vânzări" se modifică în "marketing.

Trebuie remarcat faptul că, dacă declarația de revenire la anulare va anula toate modificările efectuate după punctul de salvare, nu este posibil să se revină la o singură declarație.

Nivelurile de izolare

În general, există patru niveluri de izolare, fiecare oferind un anumit nivel de protecție împotriva modificărilor concurente și a pierderii datelor.

Nivelul Read Uncommitted (CITIRE neconfirmată)

Acest nivel permite ca o tranzacție să citească date care sunt în curs de modificare de către o altă tranzacție, ceea ce poate duce la date inconsistente și poate afecta integritatea datelor. Acest nivel nu asigură nicio protecție împotriva pierderii datelor și poate duce la o serie de probleme precum anomalii de citire murdară și citiri fantomă. Acest nivel de izolare nu este recomandat pentru tranzacții importante.

De exemplu, în aplicații de monitorizare a performanței unde datele pot fi pierdute fără a afecta rezultatele finale.

Nivelul Read Committed (CITIRE confirmată)

Acest nivel permite ca o tranzacție să citească datele modificate de alte tranzacții numai după ce acestea au fost confirmate. Acest nivel asigură o protecție mai bună decât nivelul Read Uncommitted împotriva pierderii datelor, însă nu oferă nicio protecție împotriva citirilor fantomă. Acest nivel este adecvat pentru tranzacții care nu necesită o protecție strică împotriva modificărilor concurente și pierderii datelor.

Un exemplu ar fi o aplicație de gestiune a inventarului, unde este important să se asigure că stocurile sunt actualizate corespunzător, dar nu este necesar să se evite întotdeauna modificările concurente.

Nivelul Repeatable Read (CITIRE repetabilă)

Acest nivel asigură ca o tranzacție să poată citi aceleași date de mai multe ori, chiar și în cazul în care acestea sunt modificate de alte tranzacții în timpul execuției tranzacției actuale. Această izolare previne anomalii de citire murdară și citiri fantomă. Cu toate acestea, acest nivel nu oferă o protecție completă împotriva pierderii datelor și poate afecta performanța în cazul în care sunt implicate multe tranzacții concurente.

Un exemplu ar fi un sistem de plăți online, unde se dorește să se asigure că tranzacțiile sunt efectuate într-un mod sigur și nu pot fi modificate de alte tranzacții concurente.

Nivelul Serializable (SERIALIZABIL)

Acest nivel oferă cel mai înalt nivel de izolare și asigură că tranzacțiile concurente nu pot modifica sau citi aceleași date simultan. Acest nivel previne anomalii de citire murdară, citiri fantomă și pierderi de date. Acest nivel poate afecta performanța în cazul în care sunt implicate multe tranzacții concurente, dar este esențial pentru tranzacțiile care necesită o protecție strictă împotriva modificărilor concurente și pierderii datelor.

Un exemplu ar fi un sistem bancar, unde este crucial ca toate tranzacțiile să fie efectuate corect și să nu existe erori de sincronizare a datelor.

Concluzii

În concluzie, importanța tranzacțiilor în SQL și beneficiile lor în gestionarea și controlul modificărilor bazei de date sunt explicite în acest articol. Prin intermediul comenzilor specifice, cum ar fi COMMIT, ROLLBACK și SAVEPOINT, tranzacțiile permit efectuarea mai multor operații asupra bazei de date într-o singură unitate logică, ceea ce le face esențiale pentru orice aplicație care se bazează pe o bază de date. Utilizarea lor asigură că orice modificare efectuată este validată și că baza de date rămâne întotdeauna în starea sa corectă. Beneficiile precum atomicitate, consistență, izolare și durabilitate oferite de tranzacții asigură că baza de date este întotdeauna într-o stare consistentă, izolată și durabilă.

Referințe

1. V.Cotelea. M.Cotelea, Baze de date. Chișinău, 2016
2. James R. Groff. Paul N. Weinberg, "SQL: The Complete Reference"
3. C.J. Date, "SQL and Relational Theory: How to Write Accurate SQL Code"
4. John L. Viescas și Michael J. Hernandez, "SQL Queries for Mere Mortals: A Hands-On Guide to Data Manipulation in SQL"

ПОСТРОЕНИЕ И ОПТИМИЗАЦИЯ ЗАПРОСОВ SQL

Денис ЛУПАШКО

Департамент Программной Инженерии и Автоматики, группа TI-192 F/R, Факультет Вычислительной Техники, Информатики и Микроэлектроники,
Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Денис ЛУПАШКО, e-mail: denis.lupasco@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Данная статья посвящена решению проблемы высокопроизводительных и времязатратным запросов к реляционным базам данных SQL, а также инструментам для анализа sql запросов.*

Ключевые слова: *sql, оптимизация sql запросов, построение запросов sql.EXPLAIN.*

Введение:

Работая программным инженером я часто сталкиваюсь с высокопроизводительными и неэффективными запросами к базе данных и необходимостью повысить их эффективность и время исполнения. В этой статье я хочу поделиться способами поиска проблем и их решением.

1. Построения эффективных sql запросов

Для того чтобы написанные вами запросы к базе данных были производительными надо понимать что же делает их не производительными и строить свои запросы так чтобы избегать бутылочных горлышек замедляющих ваш запрос [1].

- **Извлекайте только необходимые данные -** избегайте ‘**SELECT ***’ , если в своем запросе используете ‘EXISTS’ используйте константу в операторе ‘**SELECT**’ этого подзапроса вместо и если возможно избавьтесь от подзапросов используя оператор **‘INNER JOIN’**
- **Избегайте оператора ‘DISTINCT’-** оператор используется для возврата только различных значений, хорошо обдумайте есть ли в нем необходимость чтобы получить результат который вам необходим.
- **Ограничьте результаты -** используйте ‘**TOP’, ‘LIMIT’** чтобы ограничить результат запроса и тем самым увеличить его скорость.
- **Замените оператор OR вашего запроса на IN** если вы не используете индексы в вашей базе данных на сравниваем ом столбце.
- **Не используйте UNION -** нужно понимать что используя этот оператор вы просматриваете одну и ту же таблицу дважды всегда старайтесь вместо **‘UNION’** использовать ‘**OUTER JOIN’**
- **Оператор ‘NOT’ -** также не использует индексы лучше вместо него использовать ‘<, <>, !>’
- **Оператор ‘AND’** - так же как ‘NOT’ и ‘OR’ не использует индексы и является ресурсоемким, рекомендуется вместо него использовать оператор **‘BETWEEN’.**
- **Операторы ‘ANY’, ‘ALL’ -** с этими операторами нужно быть осторожным так как при включении их в запрос индексы не будут использоваться и если вам необходимы индексы лучше использовать альтернативу ‘MIN’ или **‘MAX’** но функции агрегации над многими строками могут занимать много времени так что в данном случае все зависит от желаемого результата.
- В тех случаях когда столбец используется в вычислениях или скалярных функциях индекс не используется например вместо **‘where birth_year + 2 = 2002’** лучше написать запрос без вычислений **‘where birth_year = 2000’**

- **Предпочитайте обходиться без оператора ‘HAVING’**- этот оператор используется вместе с оператором **‘GROUP BY’** чтобы ограничить результат запроса но является более ресурсозатратным чем **‘WHERE’**

Выше были представлены базовые антипаттерны но даже их достаточно чтобы справиться с низкой производительностью и большим временем исполнения SQL запросов в большинстве случаев.

2. Инструменты для оптимизация SQL запросов

На практике часто встречаются SQL запросы в которых иногда бывает крайне сложно найти проблему медленного исполнения, а порой и не возможно, здесь в игру вступают различные инструменты для анализа времени и плана выполнения всех частей запроса которые помогают в поиске самых медленных и неэффективных частей запроса. На моей практике самый большой запрос был около 700 строк и время выполнения составляло минуту, мне удалось снизить время выполнения до 20 секунд используя встроенный инструмент PostgreSQL ‘EXPLAIN’ команда возвращает план выполнения запроса, похожие команды или функции есть во многих базах данных но я буду пользоваться именно данной командой так как ее я применял для решения большинства проблем связанных с производительностью [2].

Чтобы использовать команду ‘EXPLAIN’ просто укажите ее перед запросом, ниже приведен синтаксис [3]

```
EXPLAIN [ ( option [, ...] ) ] sql_statement;
```

В качестве ‘option’ может быть использовано семь вариантов но чаще всего используется ‘ANALYZE’ пример результата выполнения:

```
EXPLAIN analyze SELECT * FROM test WHERE i < 100000;
                                        QUERY PLAN
----------------------------------------------------------------
Bitmap Heap Scan on test  (cost=4.37..39.99 rows=10 width=8)
(actual time=0.023..0.043 rows=10 loops=1)
  Recheck Cond: (i < 100000)
  Heap Blocks: exact=9
  ->  Bitmap Index Scan on i1  (cost=0.00..4.37 rows=10 width=0)
(actual time=0.011..0.011 rows=10 loops=1)
        Index Cond: (i < 100000)
Planning Time: 0.182 ms
Execution Time: 0.070 ms
```

анализируя результат данной функции можно понять на какую часть операции тратится больше всего времени и изменить ее используя советы приведенные в части 1. Схожими инструментами обладает большинство популярных на сегодняшний день реляционных баз данных изучив документацию по которым вы смело можете применять ее в действии.

Заключение

В заключении хотелось бы отметить что нет универсального метода для решения всех проблем связанных с эффективностью sql запросов но есть комплекс мер и подходов основные из которых были рассмотрены в рамках данной статьи которые позволят писать запросы эффективней а также оптимизировать уже существующие запросы.

Библиография

1. Руководство по SQL: Как лучше писать запросы (Часть 1) - [online] [дата обращения 02.03.2023], Доступно: https://habr.com/ru/post/465547/
2. Mihail Curchi, “PostgreSQL – Explaining the EXPLAIN Command” - [online] [дата обращения 02.03.2023], Доступно: https://isd-soft.com/tech_blog/postgresql-explaining-explain-command/
3. Mihail Curchi, “PostgreSQL – Explaining the EXPLAIN command part 2” - [online] [дата обращения 02.03.2023], Доступно: https://isd-soft.com/tech-blog/enpostgresql-explaining-the-explain-command-part-2/

ANALIZAREA MODELELELOR DE DATE OBIECT-RELAȚIONALE DJANGO ȘI DOCTRINE

Mihail ULINICI

Departamentul Ingineria Software şi Automatică, grupa TI-191 F/R, Facultatea Calculatoare, Informatică şi Microelectronică, Universitatea Tehnică a Moldovei

Autorul corespondent: Mihail ULINICI, e-mail: mihail.ulinici@isa.utm.md

Conducător ştiinţific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *În articol dat este realizat o analiză coparativă a două modele de date: Django și Doctrine, acestea fiind modele de date obiect-relaționale. Este descris modul de serializare, mecanismele de protecție a integrității bazei de date incluse în aceste modele, și sisitemele de gestionare a bazelor de date pe care le suportă. Accentul fiind pus asupra paradigmelor folosite de aceste modele și de asemenea pe implementarea lor și anume: Active Record și Data Mapper, cu scopul de a reda cât mai clar diferența dintre aceste două modele, de asemenea punctând avantajele și dezavantajele la utilizarea unui model sau a altuia.*

Cuvinte cheie: *ORM, django, doctrine, data mapper, active record.*

Introducere

Mapare Obiect Relaţională (ORM) face posibilă citirea dar și manipularea obiectelor fară a ține cont de sursa de date de unde vin aceste obiectele. În principal a fost creat pentru a acoperi diferențele dintre modelul orientat pe obiecte și modelul relațional (folosit de majoritatea sistemelor de gestiune a bazelor de date). Tot o dată nu putem sa nu remarcam că astazi din ce in ce mai mult se atestă o tot mai mare tendință în direcția integrari cu SGBD (sisteme de gestiune a bazelor de date) non-relaţionale [1].

Redarea datele într-un graf interconectat este ceva specific limajelor de programare orientate pe obiect, iar bazele de date relaționale dupa cum știm folosesc o reprezntare tabelară. A conecta atributele clasi la câmpurile tabelelor bazei de date este indezirabilă, astfel scopul unui ORM este anume acela de a stabili o relație, transparentă și de durată între cele două modele.

Acțiunea de a stoca și salva obiecte într-o bază de date relațională folosind un framework ORM, implica transformarea obiectelor în tabelele corespunzătoare și mai apoi conexiunea dintre ele facânduse prin metadatele. Un framework ORM ne ofere funcționalități precum:[2]

- o interfață de programare (API) pentru a efectua operațiuni precum (create, read, update, delete), CRUD.
- un anume limbaj prin care ne-ar permite interogarea clasele și atributele acestora.
- un anume mecanizm care ne-ar permite definirea metadatelor pentru mapările dintre obiect și tabel.
- o tratare logica coerenta în gestionarea tranzacțiilor, a datelor care sunt stocate în cache și deasemenea asocierea între clasele respective.

1. Prezentarea ORM-urilor alese

1.1. Django ORM În domeniul tehnologilor web mai exact în libajul de programare Phyton putem spune ca Django este indispensabil. Acest limbaj are la baza concepte care previn evitarea și înlaturarea repetabilităţii, şi web-site-uri ghidate de baze de date. Astfel oferindune posibilitatea de a folosi interfeţe simple pentru operați de creare, citire, modificare şi ştergere a obiectelor. Django ORM include urmatoarele funcții importante:

- un sistem de a serializa şi valida datele din formulare HTML astfel încât acestea să poată fi salvate în baza de date,

- un sistem de serializare care poate genera şi citi reprezentări XML şi JSON, incât acestea sa fie în continuare citite și prelucrate,

- ofera o protecție ipotriva injecţiilor SQL și desemenea asupra atacurilor CSRF care au ca scop de prelua accesul asupra unui cont al unui utilizator deja conectat, interceptânt astfel date foarte importante uneori chiar cu caracter personal.

Django ORM poate fi utilizat cu o mulțime de sisteme de gestiune a bazelor de date precum: Oracle, PostgreSQL, SQLite, MySQL, Microsoft SQL Server. O variantă neoficială a acestuia este compatibilă deasemenea şi cu MongoDB [3].

Doctrine ORM are la baza mai multe biblioteci ale limbajului PHP care promit oferirea serviciilor de persistenţă şi a funcţionalităţilor asociate. Unul din lucrurile iportante este ca Doctrine folosește limbajului DQL, și asta ne permite foarte comod pentru a construi interogările necesare.

O interogare DQL poate fi:

$query = $em ->createQuery('SELECT u FROM MyProject\Model\User u WHERE u.age > 20').

Prin prezența un Query Bilder se face simplăma transforma din limbajul DQL în SQL. Însa nu permanent utilizam interogarile DQL, întrucât Doctrine are un set de funcţii ce ascund oarecum interogările parametrizate, făcând uneori neiportanta intelegerea pe deplin a limbajului SQL.

La bază, biblioteca Doctrine utilizeaza clasa PDO, facând ușor dar consistent accesarea bazelor de date. Putem observa asta în figura 1.

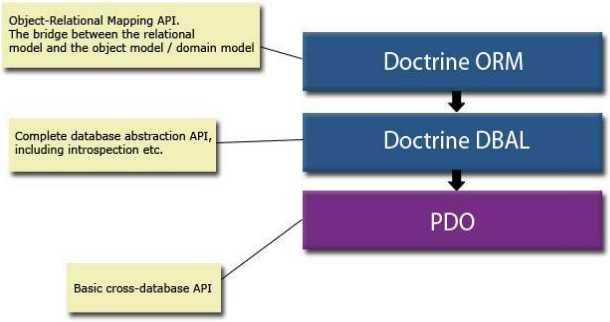


Figura 1. Structura bibliotecii Doctrine

Docrine ORM cuprinde o arie destul de mare fiind adaptabil cu multe SGBD relaţionale da nu numai, avem chiar și cu SGBD non-relaţionale cum ar fi : mongoDB, CouchDB, OrientDB, PHPCR.

În această analiză scurtă nu încerc sa compar ORM-urile alese din punc de vedere al limbajului în care au fost scrise. Ceea ce am încercat defapt este analiza din perspectiva paradigmelor implementate și mai exact, Active Record – de către Django ORM şi, Data Mapper de către Doctrine ORM.

2. Analiza comparativă a paradigmelor

2.1 Active Record. Acest model difera semnificativ de modelul Data Mapper în primul rând datorită perspectivei sale asupra obiectelor. Maparea obectelor se face exact pe un rând din tabel din baza de date, având aceleaș câmpuri și aceleaș coloane, iar în proiectele care folosesc acest ORM, nu poate fi vorba de o anumita indepentenţa faţa de SGBD, însă un avantaj este ca ne permite dezvoltarea priectelor rapid și ușor. De asemena o altă trasatura importantă pe care o putem menționa este integrarea anumitor funcţiilor de interacţiune cu baza de date în componenta obiectelor. Un exeplu de acest fel pot fi conținerea funcțiilor delete() şi save().

În utilizarea este destul de usor de folosit insa nu respectă conceptul specializării claselor. Așa ca obiectele acestei clase nu au ca scop nu numai pastrarea informației denspre anumite procese dar și interacţionează cu baza de date.

2.2 Data Mapper

Conceptul de obiect în Data Mapper e puțin diferit fața de Active Record. Aici, obiectul este o înfățișare a business-proceselor și nu este nevoie ca acesta sa se mapeze exact pe un rând dintr-un tabel. Pot avea zeci de schimari între un rând din tabel și obiectul propriuzis. De asemenea, legătura cu baza de date nu se efectuează în constructorul clasei. În loc să fie incluse direct, informațiile despre proprietățile câmpurilor sunt stocate ca metadate sau așa zisele "doc-block-uri". [4]

Pentru efectuarea interogarilor aici lucrurile stau diferit, pentru acesta sunt folosite clase separate. Un exeplu ar fi pentru a edita un obiect se foloste un Entity Manager (Manager de entităţi).

Aceste Entităţile sunt obiecte cu identitate fiind practic la fel ca și obiectele din ORM-urile Active Record. Extragerea de date și deasemeane salvarea modificarilor asupra lor este și scopul un Manager de entităţi. Acesta este un exeplu de manager de entităţi :

```
$em->persist($user);
$em->flush();
```

Avem metoda perisist() care nu are nici un efect asupra bazei de date, ea doar înregistrează obiectul şi flush() care trimite spre execuţie interogarea.

Pattern-ul Data Mapper nu este foarte raspândit este folosit destul de rar în coparație cu Active Record care este foarte popular. Fapt ce se datorează în principal bibliotecilor care îl implementează deoarece sunt mai greu de folosit. Exemple de astfel de ORM-uri sunt Doctrine, Hibernate, SQL Alchemy, Nhibernate şi Slazure, DataMapper.

Concluzii

În alegerea ORM-ului potrivit, trebuie să ținem cont de mai mulți factori care ar trebui luați în considerare la fazele icipinte ale proiectului. Cu siguranța nu putem afirma că unul este mai bun ca altul, dar putem spune ca un ORM de tip Data Maper și anume Doctrine ar fi de preferat in realizarea proiectelor mari, unde regulile de bisness logica sunt explicite, de asemenea este mult mai usor de lucrat cu baze de date mari. Prin comparați Ative Recorder ar fi de preferat în proiectele simple, dezvoltarea și implimetare rapide, cu o imagine clar definita și înțeleasa asupra bazei de date. În final alegerea îi aparține specialistului care ar fi indicat sa prevada și situațile de dozvolatrea ulterioara a prioectului, astfel find conștient de problemele și rezolvarea lor.

Referințe

1. Martin Fowler, *Patterns of Enterprise Application Architecture, 2003.*
2. *Data Mapper Pattern* [online] [accesat 02.02.2023], Disponibil: https://en.wikipedia.org/wiki/Data_mapper_pattern
3. *Active Record Pattern*. [online] [accesat 02.02.2023], Disponibil: https://en.wikipedia.org/wiki/Active_record_pattern
4. *Diferenţa dintre Active Record şi Data Mapper* [online] [accesat 02.02.2023], Disponibil: http://culttt.com/2014/06/18/whats-difference-active-record-data-mapper/

AMAZON OPENSEARCH SERVICE

Vadim VICOL

Departamentul Ingineria Software și Automatică, grupa TI-191 F/R, Facultatea CIM, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Vadim VICOL, e-mail: vadim.vicol@isa.utm.md

Conducător științific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *In acest articol stiintific este descrisa tehnologia OpenSearch, un protocol deschis pentru căutare care permite utilizatorilor să efectueze căutări într-un site sau aplicație prin intermediul unui motor de căutare web de bază. Implementarea OpenSearch permite utilizatorilor să adauge site-ul sau aplicația la lista de motoare de căutare din browserul web sau aplicația de căutare, astfel încât căutările să poată fi făcute direct din acea aplicație. Amazon OpenSearch Service este un serviciu cloud oferit de Amazon care permite utilizatorilor să implementeze și să gestioneze un motor de căutare bazat pe OpenSearch, oferind o interfață ușor de utilizat și instrumente pentru a indexa și căuta în conținutul lor. Utilizarea acestui protocol permite dezvoltatorilor să ofere utilizatorilor o experiență de căutare mai relevantă și personalizată în site-urile sau aplicațiile lor.*

Cuvinte cheie*: amazon, aws, cloud, service*

Ce este OpenSearch?

OpenSearch este o versiune bifurcată a motorului de căutare ElasticSearch. Amazon Web Services (AWS) a anunțat că va bifurca ElasticSearch după ce a obținut drepturi de autor asupra codului sursă al ElasticSearch prin intermediul achiziției ElasticSearch BV. OpenSearch oferă funcționalitățile de bază ale ElasticSearch și este compatibil cu majoritatea instrumentelor și bibliotecilor care au fost create pentru ElasticSearch. În plus, OpenSearch adaugă și unele caracteristici noi și îmbunătățite, precum și o licență de sursă deschisă care este diferită de cea folosită de ElasticSearch. Prin bifurcarea ElasticSearch și crearea OpenSearch, AWS dorește să ofere o alternativă solidă la ElasticSearch și să ofere o mai mare transparență și control utilizatorilor săi [1].

Motivația Amazon pentru bifurcarea ElasticSearch și crearea OpenSearch

Amazon Web Services (AWS) a achiziționat ElasticSearch BV, compania care dezvoltă și menține proiectul ElasticSearch, și a obținut drepturile de autor asupra codului sursă ElasticSearch. Amazon a decis să bifurce ElasticSearch și să creeze OpenSearch din mai multe motive:[2]

Controlul mai mare: Prin bifurcarea ElasticSearch și crearea OpenSearch, Amazon și-a asumat controlul complet asupra codului sursă și a putut adăuga funcționalități noi și îmbunătățite. Acest lucru a permis AWS să își personalizeze versiunea motorului de căutare în funcție de nevoile și cerințele clienților săi.

Transparența mai mare: OpenSearch utilizează o licență de sursă deschisă care este diferită de cea folosită de ElasticSearch. Această licență oferă mai multă transparență și protecție pentru comunitatea Open Source, deoarece limitează capacitatea companiilor de a profita de codul sursă open-source și de a încorpora modificări în produsele lor proprietare.

Conformitatea cu standardele Open Source: Amazon a creat OpenSearch pentru a fi un proiect open-source complet. De exemplu, Amazon a anunțat că va dona codul sursă OpenSearch la OpenSearch Foundation, o organizație non-profit care va gestiona și promova proiectul OpenSearch ca un proiect open-source complet.

Atrage noi clienți: AWS poate atrage noi clienți prin bifurcarea ElasticSearch și crearea OpenSearch. Mulți utilizatori ai ElasticSearch și-au exprimat îngrijorarea cu privire la achiziționarea ElasticSearch BV de către Amazon și au început să caute alternative la ElasticSearch. OpenSearch oferă o alternativă la ElasticSearch care este compatibilă cu instrumentele și bibliotecile existente și care adaugă noi funcționalități.

Opțiunile de utilizare a OpenSearch în AWS

OpenSearch este disponibil ca serviciu gestionat în cadrul platformei AWS (Amazon Web Services), cunoscut sub numele de Amazon OpenSearch Service. Există mai multe opțiuni de utilizare a OpenSearch în AWS, astfel:[3]

Utilizarea Amazon OpenSearch Service ca serviciu gestionat: Amazon OpenSearch Service este un serviciu gestionat care permite utilizatorilor să creeze, să configureze și să administreze clustere OpenSearch într-un mod simplu și eficient. Acesta oferă o serie de funcții precum securitate, scalabilitate și disponibilitate, fără a necesita o administrare și o întreținere complexă.

Utilizarea OpenSearch în mașini virtuale EC2: Utilizatorii pot instala și configura OpenSearch pe mașini virtuale EC2 (Elastic Compute Cloud) și să administreze OpenSearch independent. Acesta oferă o flexibilitate mai mare în ceea ce privește configurarea și personalizarea OpenSearch, dar necesită o administrare și o întreținere mai complexă.

Utilizarea Amazon OpenSearch Connector: Amazon OpenSearch Connector este un conector care permite utilizatorilor să integreze OpenSearch cu alte servicii AWS, precum Amazon S3 și Amazon Redshift. Acest lucru permite utilizatorilor să efectueze căutări avansate și analize de date în aceste servicii.

Beneficiile utilizării amazon opensearch service

Scalabilitate: Amazon OpenSearch Service poate fi utilizat pentru a căuta în cantități mari de date, permițând utilizatorilor să adauge noi documente sau să mărească volumul de căutare fără a fi nevoie să se preocupe de administrarea infrastructurii.

Fiabilitate: Amazon OpenSearch Service este construit pe platforma Amazon Web Services, care este proiectată pentru a oferi fiabilitate și disponibilitate înaltă.

Personalizarea rezultatelor: Amazon OpenSearch Service oferă instrumente și opțiuni pentru a personaliza și optimiza rezultatele de căutare, astfel încât să se potrivească nevoilor utilizatorilor.

Interfață ușor de utilizat: Amazon OpenSearch Service oferă o interfață de gestionare ușor de utilizat pentru a permite utilizatorilor să indexeze și să căutare în conținutul lor.

Integrare cu alte servicii amazon: Amazon OpenSearch Service poate fi integrat cu alte servicii Amazon, cum ar fi Amazon S3 sau Amazon CloudFront, pentru a oferi o experiență de căutare completă.

Cost eficient: Amazon OpenSearch Service este un serviciu platit dupa utilizare, astfel utilizatorii platesc doar pentru ceea ce folosesc.

Procesul de implimentare si configurare

Crearea unui cont aws: Înainte de a utiliza Amazon OpenSearch Service, trebuie să aveți un cont AWS valabil.

Crearea unui domeniu opensearch: După crearea contului, trebuie să creați un domeniu OpenSearch pentru a începe să indexați și să căutați în conținutul dvs.

Indexarea conținutului: După crearea domeniului, trebuie să indexați conținutul dvs. utilizând instrumente precum AWS SDK sau Amazon CloudSearch API.

Configurarea setărilor de căutare: După indexarea conținutului, trebuie să configurați setările de căutare pentru a se potrivi nevoilor dvs. de căutare.

Conectarea la aplicația dvs: După configurarea setărilor, trebuie să conectați Amazon OpenSearch Service la aplicația dvs. utilizând una dintre metodele de integrare disponibile.

Testarea si monitorizarea: După conectarea la aplicație, trebuie să testați și monitorizați performanța serviciului pentru a vă asigura că funcționează corect.

Optimizarea: Continuati sa optimizati serviciul in functie de nevoile utilizatorilor si de performanta serviciului.

Opțiunile de personalizare și optimizare a rezultatelor de căutare

Filtrare: Permite utilizatorilor să filtreze rezultatele de căutare după diferite criterii, cum ar fi data, autorul sau categoria.

Sortare: Permite utilizatorilor să sorteze rezultatele de căutare după diferite criterii, cum ar fi relevanța, data sau prețul.

Facetare: Permite utilizatorilor să navigheze prin rezultatele de căutare utilizând diferite categorii sau etichete.

Sugestii de căutare: Oferă sugestii de căutare bazate pe ceea ce utilizatorul a tastat deja pentru a-i ușura procesul de căutare.

Corectarea automată a erorilor de scriere: Corectează automat erorile de scriere făcute de utilizatori pentru a se asigura că se returnează rezultate relevante.

Personalizare rezultate: Permite afișarea personalizată a rezultatelor, cum ar fi imagini sau informații suplimentare legate de document.

Relevanță: Utilizeaza algoritmi de relevanța pentru a afișa cele mai relevante rezultate pentru utilizatori, in functie de cuvintele cheie si de contextul căutarii.

Utilizarea instrumentelor de gestionare și monitorizare a amazon opensearch service

Filtrare: Permite utilizatorilor să filtreze rezultatele de căutare după diferite criterii, cum ar fi data, autorul sau categoria.

Sortare: Permite utilizatorilor să sorteze rezultatele de căutare după diferite criterii, cum ar fi relevanța, data sau prețul.

Facetare: Permite utilizatorilor să navigheze prin rezultatele de căutare utilizând diferite categorii sau etichete.

Sugestii de căutare: Oferă sugestii de căutare bazate pe ceea ce utilizatorul a tastat deja pentru a-i ușura procesul de căutare.

Corectarea automată a erorilor de scriere: Corectează automat erorile de scriere făcute de utilizatori pentru a se asigura că se returnează rezultate relevante.

Personalizare rezultate: Permite afișarea personalizată a rezultatelor, cum ar fi imagini sau informații suplimentare legate de document.

Relevanță: Utilizeaza algoritmi de relevanța pentru a afișa cele mai relevante rezultate pentru utilizatori, in functie de cuvintele cheie si de contextul căutarii.

Utilizarea instrumentelor de gestionare și monitorizare a amazon opensearch service

Interfața de gestionare: Oferă o interfață web ușor de utilizat pentru a gestiona domeniul OpenSearch și setările de căutare.

Aws sdk: Oferă o interfață de programare a aplicațiilor pentru a gestiona și monitoriza serviciul prin intermediul codului.

Amazon cloudwatch: Permite monitorizarea performanței serviciului și generarea rapoartelor prin intermediul măsurătorilor în timp real.

Amazon cloudtrail: Permite monitorizarea activităților serviciului și generarea de jurnale de activitate pentru audit.

Amazon sns: Permite configurarea notificărilor pentru a fi informat atunci când se produc evenimente specifice sau atunci când performanța serviciului depășește anumite limite.

Amazon elasticsearch service: Permite gestionarea infrastructurii și monitorizarea performanței motorului de căutare Elasticsearch, care este utilizat de Amazon OpenSearch Service.

Amazon cloudformation: Permite automatizarea procesului de configurare, gestionare și monitorizare a serviciului prin intermediul unor template-uri de infrastructură.

Concluzii

In concluzie, Amazon OpenSearch Service este un serviciu puternic și flexibil oferit de Amazon, care poate fi utilizat pentru a implementa un motor de căutare în site-urile sau aplicațiile dvs. Acest serviciu oferă o soluție scalabilă și fiabilă pentru căutarea în cantități mari de date, precum și instrumente și opțiuni pentru a personaliza și optimiza rezultatele de căutare pentru a se potrivi nevoilor utilizatorilor dvs. Utilizarea instrumentelor de gestionare și monitorizare oferite de Amazon OpenSearch Service, precum și automatizarea procesului de configurare, gestionare și monitorizare prin intermediul unor template-uri de infrastructură, facilitează administrarea serviciului și monitorizarea performanței acestuia. In cazul in care doriti sa implementati o căutare performanta, personalizata si scalabila, Amazon OpenSearch Service poate fi o alegere excelenta. Este recomandat sa testați serviciul înainte de a-l lansa și să continuați să-l optimizați pe măsură ce se schimbă nevoile utilizatorilor.

Referinţe

1. Amazon OpenSearch Service Documentation, [online] [accesat 02.02.2023], Disponibil: https://docs.aws.amazon.com/opensearch-service/index.html
2. What is Amazon OpenSearch Service? [online] [accesat 02.02.2023], Disponibil: https://docs.aws.amazon.com/opensearchservice/latest/developerguide/what-is.html
3. OpenSearch Documentation, [online] [accesat 02.02.2023], Disponibil: https://opensearch.org/docs/latest/

АВТОМАТИЗАЦИЯ ПРОЦЕССОВ НА ПРЕДПРИЯТИИ С ПОМОЩЬЮ 1С

Илья ЯМБОГЛО

Департамент Программной Инженерии и Автоматики, группа TI-192 F/R, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Илья ЯМБОГЛО, e-mail: ilie.iamboglo@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, Департамент Программной Инженерии и Автоматики, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Резюме. *В статье описываются области использования 1С: Предприятия. Приводятся краткие обзоры популярных решений, описывается программный код 1С. Выделяются основные компоненты программного продукта 1С: Предприятие.*

Ключевые слова: *автоматизация, бухгалтерия, бизнес-процесс.*

Введение

Многие даже до сих пор думают, что программа 1С полезна только бухгалтеру. Наверное, именно потому, что «1С:Бухгалтерия» стала тогда первым и самым популярным продуктом.

Однако такое мнение ошибочно. Программа «1С» — это продукт для автоматизации предприятий. Перед пользователями открыт широкий ассортимент типовых решений и отраслевых конфигураций. С их помощью возможно автоматизировать бизнес-процессы в любой компании, независимо от масштабов и направлений.

Области применения продуктов 1С

Программы 1С:Предприятие созданы специально для решения всех известных задач по управлению компанией и учету. Конфигурация может содержать в себе не только стандартный набор общих функций, но и возможности под конкретную отрасль с учетом индивидуальных задач отдельной компании.

Любую программу «1С:Предприятие 8» можно адаптировать для конкретной отрасли.

Области, где для автоматизации применяются программы «1С:Предприятие»:

- Бухгалтерский и налоговый учет по стандартам России и стран СНГ;
- Торгово-складской учет и управление торговой компанией;
- Управленческий учет и бюджетирование, консолидация;
- Учет денежных средств и финансовое планирование;
- Управление производством и учет производственных затрат, расчет себестоимости;
- Управление логистикой, закупками;
- Розничная торговля;
- Управление документооборотом;
- Кадровый учет и управление персоналом, ведение кадровой политики и расчетов с сотрудниками;
- Управление взаимоотношениями с клиентами (CRM);
- Комплексное управление предприятиями (ERP) различных сфер деятельности, в т.ч. – производство, торговля, строительство, сельское хозяйство, ТЭК, ЖКХ, финансовый сектор, общественное питание, гостиничный бизнес, медицина и пр.

Для правильного определения какая программа 1С сможет решать ваши задачи, нужно учитывать много нюансов.

1С: Предприятие

Все программные решения фирмы «1С» созданы на базе единой технологической платформы «1С: Предприятие».

То есть, *программа 1С* – это сочетание технологической платформы и конфигурации. Прикладное решение устанавливается на платформу. Платформа выполняет роль основы, фундамента.

Надо понимать, что существует достаточно много конфигураций, но *платформа* для них только одна.

Уже потом на фундаменте-платформе программисты создают различные конфигурации программ для пользователей.

Конфигурация — это весь функционал, который обеспечивает работоспособность платформы.

Ценность в том, что программисты могут доработать прикладное решение под особенности любой компании. При этом сама платформа меняться не будет [1].

На сегодняшний день система 1С:Предприятие не уступает продвинутым и растиражированным западным решениям. У системы множество преимуществ, современные продукты предлагают широкие функциональные возможности по доступным ценам.

Самые распространенные продукты 1С

Если у вас небольшая фирма или вы управляете крупным холдингом, вы всегда сможете найти для себя самое оптимальное решение из линейки фирмы «1С».

1С: Бухгалтерия — это универсальное решение, которое подойдет для любых направлений бизнеса. Программа помогает вести учет компаний вне зависимости от того, какая у вас система налогообложения, а также обмениваться данными с другими конфигурациями 1С.

- Подготовка бухгалтерской, налоговой и статистической отчетности: пользователь заполняет и печатает необходимые отчеты в одной программе;
- Расчет и начисление заработной платы, а также ведение персонифицированного учета. Обеспечивается формирование унифицированных форм по трудовому законодательству;
- Работа со счетами учета материально-производственных запасов и расчетов с контрагентами;
- Возможность ведения учета сразу по нескольким организациям.

1С:Зарплата и управление персоналом («1С:ЗУП») — кадровый учет на предприятии и расчет с сотрудниками.

- Ведение учета движения кадров и персональных данных по сотрудникам. Анализ средней численности работников. Ведение штатного расписания и учета рабочего времени.
- Автоматический расчет заработной платы сотрудников с начислением и удержанием.
- В программе можно рассчитать им сформировать все социальные отчисления и необходимые налоги.
- Есть документы и все нужные инструменты для составления регламентированной отчетности, согласно действующему законодательству.
- Возможность анализировать качество состава кадров, рабочего времени сотрудников, а также фонд оплаты труда.

1С:Управление торговлей («1С:УТ») — используется как для оптовой, так и розничной торговли.

- Ведение учета товара и работа с номенклатурой. Пользователь может выбрать удобную классификацию справочника по разным сегментам и видам.
- Учет контрагентов по видам и выполнение операций по каждому из них. Программа объединяет в группу клиента, поставщика или конкурента по признакам и экономическим показателям. Вы можете оформить сделку и оценить ее перспективность, а также работу менеджера.
- Работа с закупками и продажами. Возможность оформления заказов, формирование счетов и корректировка данных по реализации. Использование подарочных сертификатов.
- Ведение учета всех складских операций с возможностью составления различных отчетов. Работа с перемещением и списанием товара. Проведение инвентаризации с отражением данных в документах.
- Управление финансами. Учет доходов и расходов.

1С:Управление нашей фирмой («1С:УНФ») — программа идеально подходит для небольших организаций. Поэтому ее часто выбирают представители малого бизнеса.

- Можно выполнять необходимые операции при поступлении, и продажи товара, а также устанавливать и изменять цены на номенклатуры;
- Ведение подробной базы ваших клиентов.
- Планирование и учет выполнения работ, услуг. Пользователь может отражать работы и услуги, которые запланированы или уже выполнены;
- Вести учет производственных операций, в том числе учет производства за каждую смену;
- Выполнение всех операций по прямым и косвенным затратам, а также их анализ;
- Возможность формирования уставного капитала с учетом всех требований законодательства [2].

Комплексная автоматизация

1С:ERP Управление предприятием» («1С:ERP») — это инновационное и эффективное решение от компании «1С» для создания комплексной информационной системы управления любым предприятием. Этот продукт позволяет автоматизировать основные бизнес-процессы, контролировать ключевые показатели деятельности предприятия, организовать взаимодействие служб и подразделений, координировать деятельность производственных подразделений, оценивать эффективность деятельности предприятия, отдельных подразделений и персонала [3].

Для чего используют

- для оптимизации процесса производства, составления достоверного графика деятельности с учетом загрузки оборудования и обеспечения ресурсами;
- для простого и удобного отслеживания ключевых показателей работы предприятия на всех уровнях управления;
- для согласованной работы служб предприятия при построении и исполнении планов продаж, производства и закупок;
- чтобы повысить эффективность работы коммерческих и логистических служб, улучшить качество обслуживания клиентов, повысить точность и оперативность получения информации;

Заключение

Несмотря на то, что система «1:С Предприятие» является далеко не единственной на рынке программных продуктов аналогичного назначения, она безусловно относится к наиболее популярным.

Сам факт умения работать в данной программе намного повышает значимость любого специалиста. Изучение принципов системы, освоение ее функциональных возможностей во многом способствует грамотной постановке учетных процессов на предприятии.

Широкие функциональные возможности, заложенные в программе, предусматривают ведение различных видов учета: бухгалтерского, управленческого, оперативного, финансового, консолидированного и др.

Решению задач консолидированного учета во многом способствует возможность интеграции системы «1С:Предприятие» с другими системами (в частности, поддержка нескольких Интернет-протоколов, прием и отправка электронной почты, поддержка DBF- и XML-форматов).

На сегодняшний день также существует мобильная версия 1С которая позволяет разрабатывать конфигурации для мобильных платформ [4].

Библиография

1. 1С: Управление нашей фирмой 8 [online] [дата обращения 02.03.2023], Доступно: https://solutions.1c.ru/catalog/small-firm/features
2. 1С-Паблишинг: 1С: Программирование для начинающих. Разработка в системе «1С: Предприятие 8.3», 781 стр.
3. 1С: ERP Управление предприятием [online] [дата обращения 02.03.2023], Доступно: https://solutions.1c.ru/catalog/1cerp
4. Мобильные технологии «1С» [online] [дата обращения 02.03.2023], Доступно: https://v8.1c.ru/tekhnologii/mobilnye-tekhnologii-1s/

TEHNOLOGIA DBLINK ÎN POSTGRESQL

Dumitru BEȘELEA

Departamentul Ingineria Software și Automatică, Grupa TI-191F/R, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul correspondent: Dumitru BEȘELEA, e-mail: beselea.dumitru@isa.utm.md

Conducător științific: Dorian SARANCIUC, DISA, FCIM, UTM

***Abstract**: DBLink este o tehnologie care permite utilizatorilor să acceseze și să interacționeze cu date din alte baze de date, care sunt găzduite pe același sau diferite servere, folosind SQL (Structured Query Language). Acesta permite, de asemenea, crearea de legături între tabele din diferite baze de date, precum și interogarea datelor din aceste tabele ca și cum ar fi una singură.*

***Cuvinte cheie:** foreign data wrapper, serverul foreign, utilizatorul, foreign table.*

Introducere

Dblink-ul este un modul din PostgreSQL care permite conectarea la alte baze de date PostgreSQL prin intermediul unei conexiuni de rețea. Acest modul permite executarea de interogări și comenzi între bazele de date conectate, făcând posibilă accesarea și manipularea datelor între diferite servere sau baze de date. Dblink-ul permite utilizatorilor să acceseze și să manipuleze datele din bazele de date conectate, fără a fi nevoie să se conecteze direct la acestea. Acest modul este util în scenarii de tip "data warehousing", unde se dorește să se acceseze date din mai multe surse și să se combine într-o singură bază de date pentru analiză sau raportare [1].

Transferul de date între baze de date

Este posibil să se transfere date între baze de date utilizând dblink în PostgreSQL, deși acest lucru poate fi mai complicat decât să accesați sau să executați comenzi SQL pe o bază de date remote. Un mod de a face acest lucru ar fi utilizarea comenzilor SQL cum ar fi "INSERT INTO" sau "COPY" pentru a transfera datele din baza de date sursă către baza de date destinație.

Exemplu, pentru a copia datele dintr-o tabelă din baza de date sursă într-o tabelă din baza de date destinație: [2]

INSERT INTO tabela_destinatie SELECT * FROM dblink('nume_dblink', 'SELECT * FROM tabela_sursa') AS data(col1 type, col2 type, ...);

Comenzile INSERT, UPDATE, SELECT, DELETE folosind Dbink

INSERT adaugă o linie nouă într-un tabel din baza de date remonte.

Iată un exemplu de instrucțiune INSERT în SQL:[3]

SELECT dblink_exec('nume_dblink', 'INSERT INTO tabela (col1, col2, ...) VALUES (val1, val2, ...)');

UPDATE modifică înregistrările dintr-un tabel din baza de date remonte.

Iată un exemplu de instrucțiune UPDATE în SQL:

SELECT dblink_exec('nume_dblink', 'UPDATE tabela SET col1 = val1, col2 = val2, ... WHERE conditie');

SELECT selectează datele dintr-un tabel din baza de date remonte.

Iată un exemplu de instrucțiune SELECT în SQL:

SELECT dblink_exec('nume_dblink', 'SELECT * FROM plaine');

DELETE șterge datele dintr-un tabel din baza de date remonte.

Iată un exemplu de instrucțiune UPDATE în SQL:

SELECT dblink_exec('nume_dblink', 'DELETE FROM plaine WHERE id = 232');

Accesarea vizualizărilor din baza de date remonte

În bazele de date, o vedere (sau "view" în termeni englezi) reprezintă o prezentare virtuală a datelor din tabelele existente. Vederea este o interogare SELECT stocată care poate fi utilizată pentru a accesa datele din tabele într-un mod similar cu utilizarea unei tabele. Viziunile sunt folosite pentru a oferi o interfață simplificată pentru utilizatori și aplicații pentru a accesa datele din bază de date, fără a fi nevoie să cunoască structura internă a tabelelor sau să scrie interogări complexe.

De asemenea, viziunile pot fi utilizate pentru a restricționa accesul la datele din tabelele subiacente sau pentru a prezenta datele într-un format diferit.

Iată un exemplu de utilizare a unei instrucțiuni pentru a accesa o vizualizare dintr-o bază de date remonte.

SELECT * FROM dblink('nume_dblink', 'SELECT * FROM vizualizare') AS vizualizare_remote(col1 type, col2 type, ...);

Avantajele utilizării Dblink

Utilizarea dblink în PostgreSQL poate oferi mai multe avantaje, cum ar fi:

- Acces la date de la mai multe surse: dblink permite accesarea și modificarea datelor din mai multe baze de date într-un singur server PostgreSQL, fără a fi necesară replicarea sau migrarea datelor.
- Interogări complexe**:** dblink permite crearea de interogări complexe care implică date din mai multe baze de date, prin utilizarea comenzilor SQL normale.
- Automatizarea proceselor: dblink poate fi utilizat în procedurile stocate sau funcțiile definite de utilizator pentru a automatiza anumite operațiuni sau pentru a crea interogări complexe.
- Utilizarea cu alte tehnologii: dblink poate fi utilizat împreună cu alte tehnologii, cum ar fi trigger-uri sau programe de planificare, pentru a realiza sarcini de automatizare sau pentru a realiza actualizări periodice ale datelor.

Utilizarea dblink poate permite multiplexarea interogarilor pe mai multe baze de date, reducând astfel încărcarea și creând posibilitatea de a crea interogări complexe.

DbLink în procedurile stocate și funcțiile definite de utilizator

Pentru a utiliza un dblink într-o procedură stocată sau o funcție definită de utilizator, trebuie să creați procedura sau funcția și apoi să utilizați comenzile SQL normale pentru a accesa sau să modicați datele din baza de date remonte.

Exemplu pentru a crea o procedură stocată care copiază datele dintr-o tabelă din baza de date remote într-o tabelă locală:

```
CREATE OR REPLACE FUNCTION copiaza_tabela()
  RETURNS void
AS
$$
BEGIN
INSERT INTO tabela_locala SELECT * FROM dblink('nume_dblink', 'SELECT * FROM
tabela_remote')   AS data(col1 type, col2 type, ...);
END;
$$ LANGUAGE plpgsql;
```

Componentele DBLink-ului în PostgreSQL sunt:

- Foreign Data Wrapper (FDW) - este un mecanism care permite accesul la datele din alte baze de date. FDW-urile sunt obiecte separate care pot fi create, modificate și șterse cu ajutorul comenzilor SQL.
- Serverul foreign - este serverul care găzduiește baza de date cu care se dorește să se stabilească o conexiune.

- Utilizatorul - este utilizatorul care va utiliza FDW-ul pentru a accesa datele din altă bază de date.
- Conexiunea de rețea - este conexiunea fizică dintre serverul foreign și serverul care găzduiește baza de date de origine. Conexiunea poate fi prin TNS sau direct prin adresa IP și numele instanței.
- SQL - este limbajul de interogare utilizat pentru a accesa datele din alte baze de date.
- Driverul de conectare - este un program care permite aplicațiilor să comunice cu serverul foreign și să transmită interogări SQL pentru a accesa datele.
- Firewall-ul - poate fi configurat să permită sau să blocheze conexiunea între serverele foreign și serverele care găzduiesc bazele de date de origine, este important să fie setată configurația potrivită pentru a permite accesul.
- Foreign Table - este un obiect din baza de date care reprezintă tabelele din baza de date străină. Acestea pot fi create prin definirea FDW-ului și a parametrilor de conectare.
- Sistemul de operare - este sistemul de operare pe care rulează serverul foreign și serverul care găzduiește baza de date de origine.

Tabelul 1

Comparea între DBlink și JDBC

Rows	Result (DBlink)	Rezult (JDBC)
1	0,0003 (sec)	0,0002 (sec)
1000	1,5 (sec)	0,4 (sec)
5000	7 (sec)	2 (sec)
10000	0,75 (sec)	0,36 (sec)
100000	80 (sec)	24 (sec)

Concluzii

În concluzie, DBlink poate oferi mai multe avantaje, cum ar fi acces la date de la mai multe surse permite accesarea și modificarea datelor din mai multe baze de date într-un singur server PostgreSQL, fără a fi necesară replicarea sau migrarea datelor. La fel DBlink-ul ne permite multiplexarea interogărilor pe mai multe baze de date, reducând astfel încărcarea și creând posibilitatea de a crea interogări complexe.

Dar pe lângă avantajele oferite, ca orice tehnologie dblink-ul are careva limiări, una dintre ele este siguranța, utilizarea DBlink-ului crește riscul de acces neautorizat sau utilizaren incorectă a datelor, prin acordarea accesului la datele din mai multe baze de date.

Referințe

1. PostgreSQL 9.1 Official Documentation. [online] [Accesat: 23.11.2022]. Disponibil: https://www.postgresql.org/docs/9.1/contrib-dblink-function.html
2. PostgreSQL: dblink - Using dblink to access a remote database. [online] [Acessat: 23.11.2022]. Disponibil: https://docs.timescale.com/mst/latest/dblink-extension/
3. Dblink extension in PostgreSQL. [online] [Accesat: 23.11.2022]. Disponibil: https://pankajconnect.medium.com/dblink-extension-in-postgresql-c2b96d91bd16

GOOGLE CLOUD SQL

Serghei BUGA

Departamentul Ingineria Software și Automatică, grupa TI-191 F/R, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Serghei BUGA, e-mail serghei.buga@isa.utm.md

Conducător științific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *Cloud SQL este un serviciu oferit de Google Cloud Platform care permite utilizatorilor să ruleze, administreze și întreține instanțe de baze de date relationale în cloud. Acesta oferă suport pentru mai multe tipuri de baze de date, cum ar fi MySQL și PostgreSQL. Cu Cloud SQL, utilizatorii pot beneficia de avantajele cloud-ului, cum ar fi scalabilitatea, disponibilitatea ridicată și flexibilitatea. Serviciul oferă, de asemenea, instrumente de administrare și monitorizare pentru a ajuta la gestionarea bazelor de date și la detectarea problemelor. Cloud SQL poate fi utilizat împreună cu alte servicii Google Cloud pentru a construi și a deplasa aplicații în cloud.*

Cuvinte cheie: *serviciu cloud, baze de date relationale, scalabilitate, flexibilitate.*

1. Introducere

O bază de date este un element inseparabil al oricărei aplicații sau sistem IT. Ea păstrează informațiile noi și schimbările efectuate de utilizatori, alte dispozitive sau procese (de exemplu, pe platformele "Internet of Things") [1].

Este (în mare parte) datorită bazelor de date faptul că, după ce deschidem Facebook sau alt site de social media, nu trebuie să completăm din nou profilul nostru de la început - deoarece informațiile introduse despre nume, prenume, grad universitar, precum și postările noastre, au fost salvate în baza de date și sunt disponibile în fiecare dată când ne conectăm la portal.

2. Ce este Cloud SQL?

Cloud SQL este un serviciu oferit de Google Cloud Platform. Este un serviciu de bază de date în cloud sau o bază de date ca serviciu (DBaaS). Datele din baza de date sunt stocate și procesate în cloud, pe infrastructura unui furnizor de servicii cloud, iar accesul este oferit de la nivelul consolă Google Cloud Platform sau linie de comandă. Asta înseamnă că proprietarul aplicației poate utiliza puterea de calcul a Google Cloud, nu trebuie să-și piardă timpul întreținând propria infrastructură și poate conecta ușor serviciul de bază de date cloud cu alte servicii GCP - de exemplu, mașini virtuale sau serviciul de gestionare a containerelor Kubernetes [2].

În prezent (noiembrie 2020), Cloud SQL suportă trei sisteme de baze de date populare:

- MySQL 5.6, 5.7 și 8.0,
- PostgreSQL 9.6, 10, 11, 12 și 13,
- SQL Server în versiunea 2017.

3. Baza de date eficientă în Google Cloud

Cloud SQL este parte a infrastructurii Google Cloud Platform. GCP este o soluție escalabilă și flexibilă cu servicii de înaltă eficiență și disponibilitate. Toate mașinile Google Cloud sunt administrate de furnizorul de servicii, ceea ce înseamnă că, folosind cloud-ul, nu trebuie să pierzi timp întreținând sau actualizând echipamentul și încă poți beneficia de servicii serverless disponibile în multe regiuni și locații. Google Cloud crează propriul său rețea de conexiuni fibra optica între centrele de date, ceea ce poate fi important pentru proprietarii de aplicații cu rază globală. Viteza transferului prin rețeaua Google ajunge până la 10 Tbs, ceea ce permite creșterea eficienței în timp ce se menține același preț pentru serviciu. În plus, Google Cloud garantează disponibilitatea serviciului la nivelul 99,95-99,99%, cel mai mare SLA printre furnizorii populari de servicii cloud.

Aceasta înseamnă că Cloud SQL, ca unul dintre serviciile GCP, este un serviciu serverless, escalabil, care asigură disponibilitate și performanță înaltă, precum și flexibilitatea costurilor, deoarece utilizarea puterii de calcul Google este numărată pe secundă [3].

4. Opțiuni extinse de configurare a instanței

În Cloud SQL, avem opțiunea unei configurări detaliate a instanței pentru fiecare sistem de baze de date. Configurarea serviciului pentru cerințele aplicației va traduce în performanță, dar și în preț [4].

La crearea unei instanțe, putem alege:

- regiunea și locația noii instanțe,
- tipul de mașină virtuală, memoria și CPU-ul său; putem alege mașini cu nuclee partajate, mașini standard sau mașini cu memorie mare;
- disc - SSD cu viteză mai mare și QPS (interogare pe secundă) mai mare, sau HDD cu preț mai mic;
- capacitatea (stocarea datelor) variază de la 10 la 30720 GB; puteți configura, de asemenea, o creștere automată (permanentă) a stocării în cazul în care este necesar.

5. Scalabilitatea serviciului și flexibilitatea costurilor

Deoarece Cloud SQL folosește puterea de calcul a Google Cloud, costul depinde de spațiul de disc utilizat și de utilizarea pe secundă. Dacă există puține interogări la baza de date și utilizarea nu este ridicată, costul serviciului va fi mai mic. Când aplicația este sub încărcare (de exemplu, va fi utilizată de câteva ori mai mulți utilizatori decât de obicei), serviciul se va mări, iar Cloud SQL va gestiona ușor toate interogările. Utilizarea puterii de calcul va fi mai mare, iar costul serviciului va fi proporțional cu consumul.

Prețul serviciului depinde, de asemenea, de sistemul nostru de baze de date, configurația instanței sau regiunea. În orice caz, costul este facturat pentru utilizarea efectivă, nu pentru resursele „închiriate în cazul” [3].

6. Reducerea costurilor de întreținere datorită automatizării

Așa cum am menționat, serviciile Google Cloud Platform sunt complet gestionate - la fel și Cloud SQL. Echipa Google se ocupă de infrastructura fizică, ceea ce permite utilizatorului să transfere costurile de întreținere în zona de dezvoltare (în cazul Arena.pl, trecerea la Google Cloud Platform a permis reducerea costurilor de întreținere a infrastructurii interne cu 20-30% - vezi studiul de caz). Automatizarea serviciului nu permite doar o reducere a costurilor interne, ci prevene și greșelile.

O mulțime de automatizare poate fi implementată în serviciul Cloud SQL, care îți ia presiunea de pe echipa tehnică. Puteți automatiza, de exemplu:

- administrarea bazei de date,
- gestionarea capacității bazei de date și optimizarea spațiului de disc ocupat,
- efectuarea backup-ului și recuperarea datelor.

Crearea automată a backup-ului

Chiar și la etapa de configurare a instanței, GCP vă oferă opțiunea de a indica un interval de timp în care se va efectua backup-ul automat. O altă opțiune recomandată este recuperarea datelor până la o fracțiune de secundă, datorită înregistrărilor jurnalului binar. Backup-ul poate fi stocat într-o regiune diferită față de instanță. Costurile de backup sunt calculate pe gigabyte de date pe lună; cantitatea depinde de sistemul de baze de date, regiunea și configurația instanței.

7. Securitatea datelor în Cloud SQL

Serviciul Cloud SQL este compatibil cu standardele de securitate SSAE 16, ISO 27001, PCI DSS și HIPAA. Datele sunt criptate atât la repaus, cât și în tranzit. Cloud SQL suportă conectarea cu Virtual Private Cloud și fiecare instanță are un firewall care permite controlul accesului la rețeaua publică. Baza de date poate conecta doar la rețeaua publică prin intermediul SSL sau Cloud SQL Proxy.

8. Integrare ușoară cu alte servicii GCP

Aplicațiile din afara mediului Google Cloud pot fi conectate la Cloud SQL. Cu toate acestea, dacă aveți întreaga aplicație sau o mare parte din ea în cloud-ul Google, puteți obține mai mult, de exemplu prin conexiuni între servicii [4].

Cloud SQL va funcționa fără probleme cu alte servicii din oferta Google Cloud Platform, inclusiv grupul Compute & Serverless: Compute Engine, App Engine, Kubernetes Engine, Cloud Run sau Cloud Functions. Puteți implica BigQuery - un serviciu de depozit de date - pentru interogări rapide la baza de date și efectuarea analizelor imediate.

Puteți citi mai multe despre conectarea Cloud SQL la serviciul BigQuery pe site-ul Google Cloud: Interogări federate Cloud SQL.

9. Pornire rapidă și migrare ușoară

O instanță a serviciului Cloud SQL poate fi creată și configurată cu ușurință din consola Google Cloud. Procesul întreg constă din câteva clicuri și completarea unor câmpuri. Puteți afla cât de ușor este să creați o instanță Cloud SQL din tutorialul de mai jos.

După configurarea și pornirea serviciului, puteți muta baza dvs. de date la Cloud SQL. Procesul este descris clar în consolă și serviciul are instrumente integrate pentru a efectua migrarea. Efectuarea migrației nu este deranjantă (în multe cazuri, nu trebuie să faceți modificări în codul aplicației), doar importul bazei de date poate dura ceva timp.

10. Cât costă Cloud SQL?

Costul serviciului Cloud SQL depinde de utilizare și spațiul ocupat. Prețul final este format din mai mulți factori, incluzând: sistemul de baze de date utilizat (taxele sunt diferite pentru MySQL, PostgreSQL și SQL Server), regiunea și locația unei instanțe, CPU și memorie virtuală a mașinii, tipul de disc și spațiul utilizat, opțiunea de disponibilitate înaltă, rețeaua și metoda de transmisie a datelor, licențele deținute. Puteți calcula costul final al serviciului în calculatorul Google Cloud sau puteți contacta FOTC pentru o estimare. În plus, prin stabilirea unei cooperări cu un partener oficial Google Cloud, veți primi un voucher de $500 pentru a fi utilizat pe Google Cloud Platform.

Concluzii

Serviciul Cloud SQL este esențial pentru afaceri și organizații care depind de baze de date pentru a gestiona și stoca informațiile lor critice. Acest serviciu oferă flexibilitate și scalabilitate în gestionarea bazelor de date, permițând utilizatorilor să acceseze și să gestioneze informațiile lor de oriunde și în orice moment. De asemenea, Cloud SQL oferă siguranță și fiabilitate în stocarea și protejarea datelor, precum și opțiuni de disponibilitate înaltă pentru a asigura accesul continuu la date. În concluzie, serviciul Cloud SQL este esențial pentru a menține funcționarea eficientă a oricărei afaceri sau organizații care depind de baze de date.

Bibliografie

1. Site-ul web oficial al Google Cloud SQL [online]. [accesat 10.12.2022]. Disponibil: https://cloud.google.com/sql/
2. Documentația Google Cloud SQL [online]. [accesat 10.12.2022]. Disponibil: https://cloud.google.com/sql/docs/
3. Blog-ul Google Cloud [online]. [accesat 10.12.2022]. Disponibil: https://cloud.google.com/blog/products/databases/introducing-new-pricing-options-for-google-cloud-sql-for-mysql-and-postgresql
4. Comunitatea Google Cloud [online]. [accesat 10.12.2022]. Disponibil: https://cloud.google.com/community/sql

СРАВНЕНИЕ БЛОКЧЕЙН И БАЗЫ ДАННЫХ

Григорий ЧАГЛЕЙ

Департамент Программной Инженерии и Автоматики, группа TI-192 F/R, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Autorul corespondent: Григорий ЧАГЛЕЙ, e-mail: ceaglei.grigorii@isa.utm.md

Научный роководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация*: Статья посвящена сравнению технологии блокчейна и баз данных. Рассказано об этих технологиях, какие плюсы и минусы у этих технологий и их сравнение. Рассмотрено, в каких случаях можно использовать блокчейн, а в каких базу данных.*

Ключевые слова: *PostgreSQL, MongoDB, СУБД, таблицы, документы, управление данными, SQL, отношения.*

Введение

Базы данных и блокчейн появились в разное время. Базы данных были введены еще в 1960-х годах, как способ хранения и обработки большого объема информации. С тех пор они стали одним из основных инструментов для управления данными в различных отраслях

Блокчейн же появился гораздо позже, как технология для создания распределенных баз данных, которая использовалась в первую очередь для создания электронных денег и других децентрализованных приложений. С тех пор блокчейн развивается и применяется в различных отраслях для решения различных задач, таких как контракты смарт, логистика и тд.

Что представляют собой блокчейн и базы данных [1]

Блокчейн — это децентрализованная и распределенная система хранения данных, которая использует технологию блокчейна. Это означает, что данные хранятся на множестве устройств в сети, а не на централизованном сервере. Каждый блок в цепочке содержит набор данных, которые могут быть использованы для верификации и аутентификации транзакций и информации. Блокчейн обеспечивает высокую степень безопасности и неизменности данных, так как каждый блок защищен хешем и подписью цифровой информации. Блокчейн также используется в различных отраслях, таких как финансы, сетевая безопасность, логистика и многое другое.

База данных — это система хранения и управления данными, которая обеспечивает быстрый и эффективный доступ к данным, а также их обновление и модификацию. Базы данных могут быть реляционными или нереляционными. Реляционные базы данных организованы в виде таблиц со строками и столбцами, где каждая строка представляет собой запись, а каждый столбец - поле. Нереляционные базы данных, такие как документоориентированные и графовые базы данных, организованы по-другому. Базы данных часто используются в бизнесе, образовании, науке и правительстве для хранения и анализа большого количества данных.

Организация данных [2]

В блокчейне данные организованы в виде блоков, каждый из которых содержит набор транзакций или другой информации. Каждый блок связан с предыдущим блоком в цепочке блоков, что дает название технологии - "блокчейн" (blockchain). Каждый блок содержит несколько полей, таких как номер блока, время создания, список транзакций и хеш предыдущего блока.

Хеш — это специальный набор битов, который генерируется на основе данных в блоке и используется для идентификации и аутентификации блока. Это означает, что данные в блоке не могут быть изменены без изменения хеша, что делает блокчейн надежным и безопасным способом хранения данных.

Таким образом, блокчейн обеспечивает децентрализованное и распрределенное хранение данных, где каждый блок является независимым и независимо проверяемым элементом цепочки. Это означает, что данные в блокчейне не могут быть изменены или удалены без согласия большинства участников сети. Блокчейн также обеспечивает высокую степень безопасности и конфиденциальности данных, так как данные шифруются и распределены между участниками сети.

В базах данных данные организованы в структурированном виде, что позволяет управлять и извлекать данные быстро и эффективно. Организация данных в базе данных может быть реляционной или нереляционной, в зависимости от типа базы данных.

Реляционные базы данных организованы в виде таблиц со строками и столбцами, где каждая строка представляет собой запись, а каждый столбец - поле. Таблицы могут быть связаны между собой с помощью связей, что позволяет извлекать данные из нескольких таблиц с помощью одного запроса.

Нереляционные базы данных, такие как документоориентированные и графовые базы данных, организованы по-другому. Например, в документоориентированной базе данных каждая запись представляет собой документ, который содержит поля и значения. Эти документы могут быть организованы в коллекции или базы данных и могут быть извлечены и изменены напрямую, без необходимости создания связей между таблицами.

В общем, организация данных в базах данных зависит от типа базы данных и используемой модели данных. Однако, в обоих случаях, цель – это обеспечить быстрый и эффективный доступ к данным, а также их управление и модификацию.

Управление данными [3]

Управление данными в блокчейне основано на децентрализованной модели, где каждый участник имеет доступ к данным и может вносить изменения в них. Это означает, что управление данными в блокчейне осуществляется с помощью системы консенсуса, которая подтверждает изменения данных с помощью согласия большинства участников сети.

Добавление новых данных в блокчейн осуществляется с помощью транзакций, которые записываются в блоки. Эти блоки связаны между собой в цепочку, и содержат хеши предыдущих блоков, что обеспечивает безопасность и неизменность данных.

Изменение данных в блокчейне также может осуществляться с помощью транзакций, но это может быть более сложным, так как требуется согласие большинства участников сети и обновление всех копий блокчейна. В зависимости от типа блокчейна, могут использоваться различные методы управления данными, например, в блокчейнах с приватным консенсусом, управление данными может быть централизованным, с ограниченным доступом к управлению данными только для определенной группы авторизованных участников.

Управление данными в базах данных осуществляется с помощью специальных языков запросов и инструментов. Эти языки и инструменты позволяют добавлять, изменять, извлекать и удалять данные из базы данных. Например, язык SQL (Structured Query Language) используется для управления данными в реляционных базах данных, а для нереляционных баз данных используются специфические языки запросов, такие как MongoDB или Couchbase.

В дополнение к этому, существуют различные инструменты для управления базами данных, такие как менеджеры баз данных и администраторы баз данных, которые обеспечивают функциональность для управления базой данных, такую как создание и удаление таблиц, индексов и констрейнтов, а также мониторинг производительности и оптимизация базы данных.

В общем, управление данными в базах данных включает в себя множество функций и инструментов, которые помогают эффективно управлять и извлекать данные из базы данных.

Отношения между сущностями [4]

Отношения между сущностями в блокчейне могут быть реализованы на различных уровнях и использоваться для решения различных задач. Например, можно создать связи между транзакциями, чтобы отслеживать историю изменений данных, или создать связи между аккаунтами, чтобы отслеживать движение средств между ними.

В некоторых случаях можно использовать связи между сущностями для реализации сложных бизнес-логик и смарт-контрактов, например, создание контракта, который автоматически переводит средства между аккаунтами при выполнении определенных условий. В общем реализация отношений между сущностями в блокчейне зависит от конкретной проблемы и используемой технологии.

Отношения между сущностями в базах данных — это способ связывания данных между таблицами, который обеспечивает целостность и консистентность данных. Существует три основных типа отношений между сущностями: один ко многим, многие ко многим и один к одному.

Эти отношения между сущностями могут быть реализованы с помощью различных инструментов и технологий, включая внешние ключи, таблицы соединений и связи между объектами. Это позволяет сохранять целостность и консистентность данных, а также упрощает запросы и извлечение данных из базы данных.

Когда подходят Блокчейны и Базы данных [5]

Базы данных и блокчейны решают различные задачи и подходят для разных типов приложений.

Базы данных идеально подходят для систем, которые требуют быстрого доступа к данным и обработки большого количества транзакций, таких как интернет-магазины, социальные сети и финансовые системы. Они позволяют хранить и обрабатывать большое количество данных, с возможностью создания индексов и оптимизированных запросов для быстрого доступа к данным. Они также могут поддерживать транзакционность, что обеспечивает целостность данных.

Блокчейны, с другой стороны, подходят для систем, которые требуют децентрализованности и независимости от одного центрального авторитета. Они могут использоваться для отслеживания транзакций в криптовалюте или для создания децентрализованных приложений (dApps). Блокчейн обеспечивает неизменность и прозрачность данных, что делает его подходящим для систем, которые требуют надежности и неизменности записей.

В общем базы данных и блокчейны могут использоваться вместе или независимо, в зависимости от требований к приложению. Например, база данных может использоваться для хранения и обработки данных в реальном времени, а блокчейн может использоваться для отслеживания истории изменений данных и для обеспечения неизменности записей.

Когда не подходят Блокчейны и Базы данных [6]

Базы данных и блокчейны не всегда являются оптимальным выбором для всех типов приложений и ситуаций.

Базы данных могут быть неэффективны в системах, которые требуют децентрализованности и независимости от одного центрального авторитета. Также они могут быть менее безопасными в ситуациях, когда данные требуют неизменности и недоступности для изменения или изменения может быть опасно.

Блокчейны могут быть менее эффективны в системах, которые требуют быстрого доступа к данным и обработки большого количества транзакций, поскольку они часто имеют медленные и ограниченные возможности для запросов и обработки данных. Также они могут быть дороже и сложнее для настройки и обслуживания по сравнению с базами данных.

Кроме того, блокчейны могут быть неподходящими для приложений, которые требуют конфиденциальности данных, так как информация в блокчейне обычно является публичной и доступной для всех участников сети.

В общем, обе базы данных и блокчейны имеют свои преимущества и недостатки, и выбор между ними зависит от конкретных требований и целей приложения.

Комбинирование технологий [7]

Комбинирование баз данных и блокчейнов может принести множество преимуществ для различных типов приложений. Например, база данных может использоваться для хранения и обработки данных в реальном времени, а блокчейн может использоваться для отслеживания истории изменений данных и для обеспечения неизменности записей.

В системы, где требуется конфиденциальность и контроль доступа, база данных может использоваться для хранения части данных, а блокчейн может использоваться для хранения хэшей и другой информации, которая должна быть открытой для всех участников.

Комбинирование баз данных и блокчейнов также может помочь сократить затраты на разработку и обслуживание приложения, так как каждая технология может использоваться для решения специфических задач.

В целом, комбинирование баз данных и блокчейнов может быть полезным инструментом для различных типов приложений и отраслей.

Заключение

В заключение, базы данных и блокчейны являются различными технологиями, которые могут решать различные задачи и подходят для разных типов приложений. Базы данных идеально подходят для систем, которые требуют быстрого доступа к данным и обработки большого количества транзакций, в то время как блокчейны подходят для систем, которые требуют децентрализованности и независимости от одного центрального авторитета. В зависимости от конкретных требований и целей приложения, может быть предпочтительно использовать один из этих инструментов или их комбинацию.

Литература

1. Симонов Ф. В. *Введение в безопасность блокчейна*. [online] [дата обращения 02.03.2023], Доступно: https://morioh.com/p/5ccd193eac0c
2. Тарасенко Е. *Блокчейн и традиционная база данных.* [online] [дата обращения 02.03.2023], Доступно: https://merehead.com/ru/blog/private-blockchain-vs-traditional-centralized-database/
3. Amazon Web Services (AWS). *Что такое технология блокчейн*. [online] [дата обращения 02.03.2023], Доступно: https://aws.amazon.com/ru/what-is/blockchain/
4. Oracle Cloud Infrastructure (OCI). *Что такое база данных?* [online] [дата обращения 02.03.2023], Доступно: https://www.oracle.com/cis/database/what-is-database/
5. Роман Андреев, Selectel. *Виды баз данных.* [online] [дата обращения 02.03.2023], Доступно: https://selectel.ru/blog/databases-types/
6. 2bitcoins, *В чём различие между блокчейном и базой данных*. [online] [дата обращения 02.03.2023], Доступно: https://2bitcoins.ru/razlichie-mezhdu-blokchejnom-i-bazoj-dannyh/
7. MadData. *Blockchain и Big Data: что необходимо знать?* [online] [дата обращения 02.03.2023], Доступно: https://maddata.agency/blog/blockchain-i-big-data-chto-neobkhodimo-znat

TEHNICI DE RECUPERARE A DATELOR ÎN SQL SERVER

Eugeniu ROMANESCU

Departamentul Ingineria Software și Automatică, grupa TI-181 F/R, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Eugeniu ROMANESCU, e-mail: eugeniu.romanescu@isa.utm.md

Conducător științific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat: *În acest articol sunt reprezentate modelele și tipurile de recuperare a datelor într-o bază de date. Vom afla cum afectează modelele de recuperare tipurile de recuperare. Backup-ul este important pentru baza de date, pentru a asigura siguranța datelor introduse. Recuperarea face referință la readucerea unei baze de date într-o stare funcțională, după ce a suferit o pană sau o pierdere de date.*

Cuvinte cheie: *copie de rezervă, modele de recuperare, tipurile de recuperare, proprietăți, tranzacții, diferențial.*

Introducere

În calitate de administratori de baze de date, datoria noastră principală este să protejăm datele din bazele de date pentru care suntem responsabili. Aceste date reprezintă inima și sufletul companiei care ne angajează și ar putea distruge o afacere dacă nu vor fi întreținute corespunzător. Din această cauză soluția de copie de rezervă și restaurare este absolut necesară pentru a oferi un nivel înalt de securitate a datelor.

Ce este o copie de rezervă completă?

O copie de rezervă completă este un tip de copie de rezervă a datelor care implică realizarea unei copii a tuturor fișierelor, folderelor și datelor de pe un sistem computerizat sau un anumit dispozitiv de stocare.

În timpul unei copii de rezervă complete, toate datele sunt copiate și stocate într-o destinație de rezervă. O copie de rezervă poate fi salvată pe discul local sau partajat pe rețeaua disponibilă sau chiar în stocarea blob Windows Azure (dacă folosim SQL Server 2012 sau o versiune ulterioară). Aceasta înseamnă că, dacă orice date sunt pierdute sau deteriorate pe sistemul sau dispozitivul original, acestea pot fi restaurate din backup complet.

Modele de recuperare

Un model de recuperare este modul în care i se spune lui SQL Server să recupereze datele. Fig. 1 arată unde să căutați zona de configurare a modelului de recupare. Aceasta este găsit făcând clic dreapta pe o bază de date existentă și alegând „Proprietăți”, apoi selectând „Opțiuni” din meniul din stânga [1].

În partea de sus a acestei ferestre, a doua opțiune este pentru modelul de recuperare. Avem trei opțiuni pentru modelele de recuperare. Aceste opțiuni sunt:

- Complet: această opțiune permite recuperarea bazei de date la aproape orice moment în timp și este alegerea clară a multor SGBD;
- Înregistrat în bloc: similar cu recuperarea completă, dar acest scenariu permite ca înregistrarea să fie minimizată pentru operațiuni în vrac (copiere, în special);
- Simplu: aceasta este alegerea pentru baze de date mai mici. Nu permite un punct în timp ca recuperarea completă sau înregistrarea în bloc. Pur și simplu permite recuperarea utilizând ultima copie de rezervă.

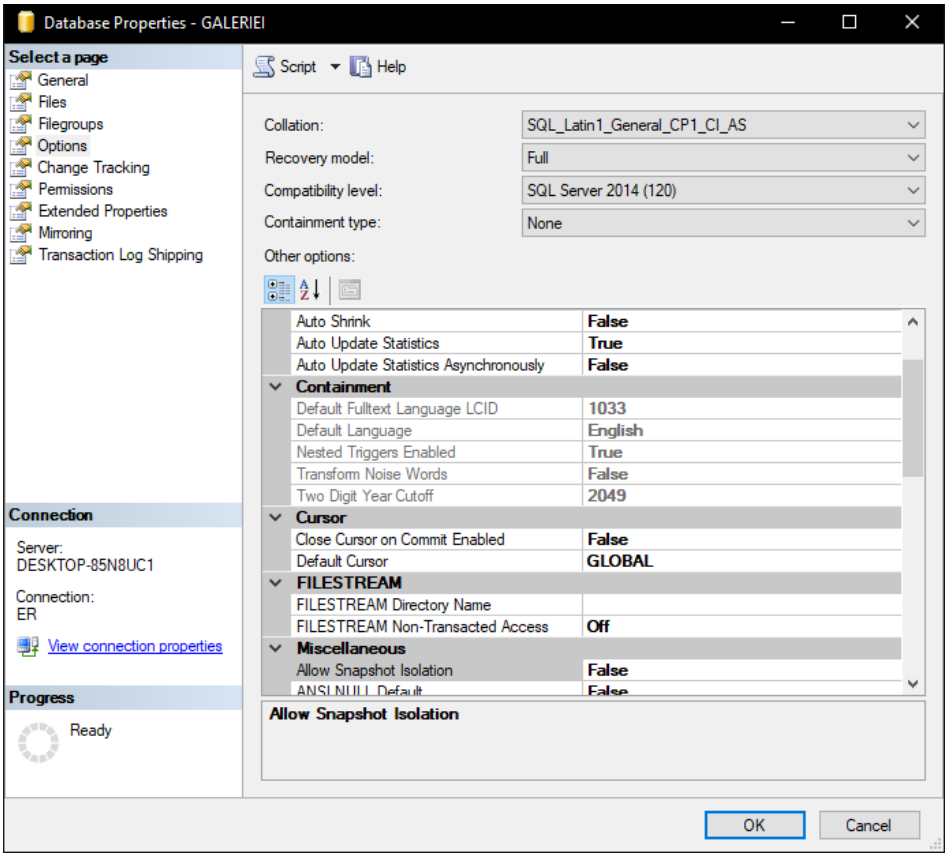


Figura 1. Fereastra unde găsim opțiunile de recuperare

Tipuri de copii de rezervă

SQL Server are trei tipuri de copii de rezervă unice. Fiecare acționează diferit și pot lucra împreună sau separat pentru a oferi o soluție de rezervă pentru datele bazei, acestea sunt:

- Recuperare completă: o copie de rezervă completă va face o copie de rezervă a întregii baze de date, care include jurnalul de tranzacții. Folosind această metodă, orice poate fi restaurat până la momentul în care s-a executat copia de rezervă;
- Copie de rezervă diferențiale: o copie de rezervă diferențială conține toate datele care nu au fost recuperate de la ultima copie de rezervă completă;
- Copie de rezervă ale jurnalului de tranzacții: o copie de rezervă a jurnalului de tranzacții va conține tranzacția care afectează starea actuală a bazei de date de la ultima copie de rezervă.

Avantajele copiilor de rezervă

- Recuperarea datelor: Backup-urile oferă o modalitate de a recupera datele în cazul pierderii datelor din cauza unei defecțiuni hardware, dezastre naturale, atacuri cibernetice sau erori umane;
- Minimează timpul de nefuncționare: Cu backup-uri regulate, vă puteți restaura rapid datele și sistemele la o stare anterioară, minimizând timpul de nefuncționare și asigurând continuitatea afacerii;
- Liniștea sufletească: a avea copii de rezervă vă oferă liniștea sufletească că datele dvs. importante sunt sigure și securizate și pot fi restaurate dacă este necesar;
- Conformitate: Backup-urile oferă redundanță, ceea ce înseamnă că datele sunt stocate în mai multe locații, reducând riscul pierderii datelor din cauza unui singur punct de defecțiune.

Dezavantajele copiilor de rezervă

- Cost: În funcție de dimensiunea și complexitatea datelor dvs., backup-urile pot fi costisitoare de implementat și întreținut, necesitând resurse suplimentare de hardware, software și personal;
- Consumatoare de timp: Backup-urile pot dura o perioadă o perioadă semnificativă de timp, mai ales dacă aveți o cantitate mare de date pentru a face backup;

- Probleme de compatibilitate: Este posibil ca backup-urile să nu fie compatibile cu versiuni mai noi de software sau hardware, ceea ce poate duce la probleme de compatibilitate atunci când se încearcă restaurarea datelor;

- Eroarea umană: Backup-urile sunt la fel de fiabile ca și persoana responsabilă cu crearea și întreținerea lor. Eroarea umană poate duce la pierderea datelor dacă backup-urile nu sunt gestionate corespunzător sau testate în mod regulat.

Concluzii

Tehnicile de recuperare a datelor sunt obligatorii de cunoscut pentru a putea lucra cu un oarecare SGBD. Este foarte importat ca datele înregistrate să fie în siguranță și în cazul unei pierderi să fie ușor de recuperat. În timp ce backup-urile complete pot consuma mult timp și necesită spațiu de stocare semnificativ, acest lucru va oferi cea mai completă protecție împotriva pierderii de date.

Bibliografie

1. BRADLEY BEARD. *Beginning Backupand Restore for SQL Server,* 2018.

OPTIMIZAREA OPERAȚIUNILOR SQL CU AJUTORUL LIMBAJULUI PYTHON

Daria NEDELCO

Departamentul Ingineria Software şi Automatică, grupa TI-191 F/R, Facultatea Calculatoare, Informatică şi Microelectronică, Universitatea Tehnică a Moldovei, Chişinău, Republica Moldova

Autorul corespondent: Daria NEDELCO, e-mail: nedelco.daria@isa.utm.md

Conducător ştiinţific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *În lucrare sunt prezentate metode de optimizare a operațiunilor SQL cu ajutorul limbajului Python. Combinarea posibilităților SQL cu limbajul Python adaugă flexibilitate și utilitate care crește performanța și reduce consumul de resurse necesare pentru execuția interogărilor.*

Cuvinte cheie*: Operațiuni SQL, limbajul Python, optimizare, flexibilitate.*

Introducere

SQL este un limbaj structurat de date care permite manipularea datelor in sistemele de manipulare a a bazelor de date relaționale. Cu toate că, SQL este un instrument puternic de manipulare, extragere și scriere a datelor intr-un sistem de bază de date, acestuia îi lipsește flexibilitatea și utilitatea limbajului de scriptare Python [1].

Ciclare prin multiple Tabele SQL

Cea mai evidentă metodă prin care limbajul Python ar putea optimiza codul SQL este incorporarea unui șir de interogare intr-o structură de ciclare Python pentru a itera mai multe interogări consecutiv.

Folosind variabilele definite in Python, putem crea o interogare de bază folosind atît text SQL cît și variabile Python.

De exemplu, pentru a extrage o listă a tuturor tabelelor și seturi de date dintr-un proiect, va fi necesar de scris următorul cod SQL:

```
SELECT * FROM
`my_project.dataset_1.INFORMATION_S
CHEMA`
UNION ALL
SELECT * FROM
`my_project.dataset_2.INFORMATION_S
CHEMA`
UNION ALL
SELECT * FROM
`my_project.dataset_3.INFORMATION_S
CHEMA`
```

Figura 1. Codul SQL care permite extragerea listei tabelurilor și seturilor de date din proiect

Pentru a evita introducerea manuală a interogărilor putem integra un script Python.

```
from google.cloud import bigquery

datasets = ['dataset_1',
'dataset_2', 'dataset_3']

bq_client = bigquery.Client()

for dataset in datasets:
    get_datasets =
bq_client.query("SELECT dataset_id,
table_id,         size_bytes,
ROUND(size_bytes / 10000000000), 2)
AS gb_size
FROM
`"+dataset.dataset_id+"`.__TABLES__
GROUP BY 1, 2, 3")

    tables = get_datasets.result()
    for table in tables:
        dataset_id = table.dataset
        table_id = table.table_id
        size = table.size_bytes
        gb_size = table.gb_size

        print(dataset_id, table_id,
size, gb_size)
```

Figura 2. Scriptul Python de formare a interogărilor

Chiar daca pare a fi mai complex, cu două cicluri, acest script rulează prin toată lista de șiruri de date, înlocuind UNION. Cicluri in SQL pot fi create, dar de obicei, sunt necesare pasuri adăugatoare ca definirea variabilelor si crearea UDF (User Defined Functions) [2].

Definirea Schemei Automate

The BigQuery Python Client permite definirea schemelor ca o listă, care poate fi transmisă către funcția de încărcare. Definirea unei scheme BigQuery in SQL manual poate duce la schimbarea tipului de date cînd este încărcată pe GCP (Google Cloud Platform):

```
schema = [

bigquery.SchemaField("first_name",
"STRING"),

bigquery.SchemaField("last_name",
"STRING"),
     bigquery.SchemaField("age",
"INTEGER")

]
```

Figura 3. Schema BigQuery

Introducerea manuală a datelor devine anevoiasă în caz că este nevoie de minim 100 de coloane. Soluția ar fi funcția Python care permite auto-completarea cîmpurilor.

```
def create_schema(field_list: list,
type_list: list):

    schema_list = []

    for fields, types in
zip(field_list, type_list):
        schema =
bigquery.SchemaField(fields, types)
        schema_list.append(schema)

    return schema_list
```

Figura 4. Funcția Python cu auto-completarea cîmpurilor

Convertirea în Data Frame intr-o linie de script Python

Există o metodă simplă de creare a Data Frame derivată din interogări SQL și este necesar doar de o linie de script Python, în special dacă stocăm interogările extern într-un fișier config.

```
query = """ SELECT * FROM
`my_project.dataset.table` """

query_job =
bq_client.query(cfg.query).to_dataf
rame()
```

Figura 5. Linia de script Python

În caz că dorim să salvăm rezultatul interogărilor, putem converti în fișier CSV în aceeași linie.

```
query_job =
bq_client.query(cfg.query).to_dataf
rame().to_csv('query_output.csv')
```

Figura 6. Convertirea în fișier CSV

Operațiuni Append/Truncate

Să presupunem că avem un tabel, care necesită editarea multiplă a datelor în timpul zilei și la sfîrșit de zi avem nevoie de toate datele introduse, în urma căruia pot apărea cîmpuri duplicate. În mod ideal, ar fi bine de eliminat toate duplicatele cînd încărcăm datele. Cu părere de rău, BigQuery nu suportă operațiile APPEND/TRUNCATE și în caz că dorim să rescriem un tabel SQL pentru o perioadă anumită de timp putem combina Python cu SQL. Odată cu pornirea script-ului, acesta arată toate schimbările introduse în timpul real [3].

```
crud_statement = """ DELETE FROM
table WHERE date = CURRENT_DATE()
"""

bq_client.query(crud_statement)

df = df[(df['date'] ==
date.today())]

bq_client.load_table_from_dataframe
(df, job_config)
```

Figura 7 - Scriptul Prezintă schimbările în timp real

Concluzii

Combinarea a unui limbaj script ca Python cu SQL oferă posibilitatea de a efectua operațiuni imposibile doar pentru limbajul SQL, cum ar fi:

1. Ciclare prin multiple tabele SQL;
2. Definirea schemei automate;
3. Convertirea în Data Frame intr-o linie de script Python;
4. Operațiuni Append/Truncate

Referințe

1. SQL BigQuery [online]. [accesat 10.12.2022]. Disponibil: https://cloud.google.com/bigquery/docs/reference/standard-sql/introduction
2. Combinarea SQL cu limbajul Python [online]. [accesat 10.12.2022]. Disponibil: https://towardsdatascience.com/improve-your-sql-skills-using-pure-python-9e50736b3d9c
3. SQL Queries in Python [online]. [accesat 10.12.2022]. Disponibil: https://towardsdatascience.com/a-simple-approach-to-templated-sql-queries-in-python-adc4f0dc511

СЕРИАЛИЗАЦИЯ ТРАНЗАКЦИЙ В БАЗАХ ДАННЫХ

Александр ЛУПУ

Департамент Программной Инженерии и Автоматики, Группа TI-192, Факультет Вычислительной Техники и Микроэлектроники, Технический Университет Молдовы, Кишинёв, Республика Молдова

Автор корреспондент: Александр ЛУПУ, e-mail: lupu.alexandru@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация. *Данная работа посвящена сериализации транзакций баз данных. В работе рассматриваются виды конфликтов, а также методы захвата и освобождения.*
Ключевые слова: *сериализация, транзакции, захват, освобождение.*

Введение

Одним из методов повышения производительности некоторых сериализованных проектов является сериализация транзакций. Предполагается, что основной задачей компонента SUBD, отвечающего за обработку транзакций, является предоставление такого механизма. Реальная автономия пользователя обеспечивается в системах, поддерживающих сериализацию транзакций [1].

Основным препятствием для реализации является выбор стратегии сериализации для ряда транзакций, что, несомненно, уменьшит их параллельность. Первое, что приходит на ум, как правило, считается выполнением сделок позже. Однако существует несколько сложностей, которые операторы различных транзакций могут выполнять в любом порядке при сохранении серийности. Примеры включают простое чтение операций и операций, которые не конфликтуют друг с другом на объектах базы данных.

Виды конфликтов

Между транзакциями существуют следующие виды конфликтов: [1]
W-W - транзакция 2 пытается изменить объект, измененной не закончившейся транзакцией 1;
R-W - транзакция 2 пытается изменить объект, прочитанной не закончившейся транзакцией 1;
W-R - транзакция 2 пытается читать объект, измененной не закончившейся транзакцией 1.

Практические методы сериализации транзакций основывающиеся на учете этих конфликтов

Методы «привлечения» и «освобождения» объектов, созданных по инициативе сделок, используются для обеспечения серийной транзакции: сделка «притягивает» объект, что, в свою очередь, приводит к его блокированию для других сделок и освобождает его только после завершения собственной сделки [2].

В этом случае результаты объекта по чтению одной транзакцией несовместимы с результатами того же объекта с чтением другой транзакции по записи, а последствия первого объекта различными транзакциями по записям не совместимы.

Считается, что более широко используемый подход в SUBD, основанный на архитектуре «клиент-сервер», является решением, которое достигает соответствия двухфазному протоколу объектов DB. В целом, протокол устанавливается тем фактом, что объект будет помещен в соответствующий объект, прежде чем какой-либо вид операции будет выполнен над объектом базы данных от имени транзакции. В согласовании с сведениями протоколом выполнение транзакции разбивается на 2 фазы:

1) транзакции – скопление захватов;
2) высвобождение захватов.

Самая большая проблема с последующим двухфазным процессом заключается в определении того, что квалифицируется как объект для achvata. Верховные вариации в реляционных базах данных, вероятно, включают: [3]

Файл - это физическая сущность, которая может хранить отношения и индексы; таблица - это сущност, используемая в экономическом анализе, которая основана на большом количестве записей для указанной связи; запись - это простое физическое вещество, найденное в базе данных; страница данных - физическая субстанция, содержащая записи одной или нескольких отношений, индекса или индексной информации.

Наименьшие потери будут включены в диапазон, так как чем больше объект, тем меньше будет каталога системы. Если вы выберете определенные элементы, чтобы назначить для файла или отношения, строка призраков будет устранена. Однако вероятность транзакционных аварий возрастает, и их допустимый уровень параллельной производительности падает больше всего, когда они используются для больших объектов. Чтобы увидеть события в ситуациях, когда их нет, синхронизированный сценарий намеренно ухудшается во время выполнения при усилении объекта.

Согласно этой аббревиатуре, транзакция на самом деле является завершенным блоком ссылок на основу данных и некоторые связанные с ними операции, для которых обеспечивается удовлетворение четырех требований, таких как так называемые квазиты: Атомность: Транзакционная деятельность разделяет неразрывный блок данных на начало и конец [2].

Либо вся эта единица производится, либо она вообще не генерируется. Возвращение к первоначальному состоянию происходит в том случае, если в процессе сделки возникает ошибка; последовательность: все предметы присутствуют и находятся в согласованном состоянии при заключении сделки;

Для предотвращения перекрестного влияния между транзакциями координируется то, чтобы все типы приложений имели одновременный доступ к разделенным объектам.

Основными режимами синхронизационных захватов являются:

Монопольный режим – X (эксклюзивный), представляющий собой монополистическое прикосновение объекта и необходимое для выполнения манипуляций касания, расширения и трансформации. Кооперативный режим — S (Shared), представляющий собой отделяемое прикосновение объекта и необходимый для выполнения операции чтения объекта. Захват объекта одной транзакцией чтения несовместим с захватом другой трансакцией того же объекта записи, а захваты первого объекта различными транзакционными записями являются не совместимыми. Однако захват объекта несколькими транзакциями чтения совместим, то есть несколько транзакций могут декларировать объект [2]. Критерии сопоставимости захватов 1-го объекта различными транзакциями изображены в таблице 1:

Таблица 1

Правила совместимости захватов одного объекта разными транзакциями

	X	S
-	да	да
X	нет	нет
S	нет	да

Вероятные состояния элемента перечислены в первой колонке как синхронизационные диапазоны. Когда указано, «-» обозначает состояние объекта, для которого в основном нет формулировки. Транзакция, которая запрашивает волна синхронизации объекта BD, останавливается до тех пор, пока волна от этого объекта не будет выпущена, потому что она в настоящее время используется другой транзакцией в неконкурентном режиме. Обратите внимание, что текст «нет» в нашей таблице фактически соответствует упомянутым более вероятным событиям транзакций по доступу к объектам базы данных. WW, RW и WR Сравнительность S-снимков аналогична тому, чем событие RR не является в реальности.

Методы сериализации транзакций

Сериализация транзакций имеет два фундаментальных различия: отличия, основанные на синхронизации объектов базы данных, и отличия на основе использования временных ярлыков. Экспозиция инстанций транзакций и их устранение порождают сущность обеих различий. Ниже мы рассмотрим эти различия относительно молодым взглядом.

На самом деле, существует два типа оптимизма - жизненно важный оптимизм и пессимизм - для каждого из этих различий [1].

При использовании пессимистических исторических подходов, где эпизоды бывают частыми, инциденты замечаются и обращаются с ними, как только они возникают. (actual or understood). Основой жизненно-утверждающих подходов является представление о том, что все операции трансформации базы данных производят результаты, которые хранятся в рабочей памяти транзакций. Только на этапе фиксации транзакции база данных подвергается действительным изменениям. Он проверяется, чтобы увидеть, были ли какие-либо инциденты с другими транзакциями в течение этого времени [1].

Дальше мы ограничимся рассмотрением больше популярных пессимистических видов способов сериализации транзакций. Пессимистические способы относительно элементарно модифицируются в собственные жизнеутверждающие варианты.

Типы временных целей. Для использования временных целей был разработан иной метод сериализации транзакций, который выполняется безупречно в соответствии с требованиями редких транзакционных событий и не требует создания понимания текущей транзакции. Вышеприведенное иллюстрирует основное понятие метода: в случае, если транзакция T1 началась до транзакции T2, система обеспечивает сопоставимый режим исполнения, как если бы T1 был завершен до начала T2.

Для каждой транзакции T, которая соответствует времени, начинающемуся с T, требуется краткосрочный знак t. Транзакция T записывает временную отметку и вид операции (читать или изменять) для каждой операции, выполненной на объекте r.

Транзакция T1 имеет следующие эффекты перед выполнением операции на объекте r:

Найдя объект, вы можете убедиться, что транзакция T все еще активна. T1 обозначает объект r и завершает его действие, если T завершен.

Транзакция T1 проверяет конфликты операций, если транзакция T не завершена. Если между операциями нет конфликта, транзакция T1 завершает свою операцию, а объект r остается временным тегом с малой значимостью.

При столкновении двух операций T1 и T, если t(T)> t (T1) (т.е. транзакция T более «молодая», чем трансакция T), T фиксируется, и T1 продолжает работать; если t/T t(T1) (то есть транзакции T старше, чем T1), T1 получает новую временную отметку, и она начинается заново.

Потенциально более частые транзакции, чем при использовании транзакций синхронизации, являются одним из недостатков метода временных тегов.

Конфликт сделок ориентирован более прямо как результат, поэтому. Несмотря на это, производство массивных маркеров времени столкновения с соотношением достигнутых успехов в распределенных системах недостаточно элементарно. (this is a different enormous science). Тем не менее, эти недостатки компенсируются в распределенных системах тем, что нет необходимости решать заторможенность; как мы уже отмечали, построение понимания ожиданий в распределённых системах довольно дорогое.

Заключение

В заключении можно отметить, что механизм сериализации транзакций является основной функцией компонента СУБД, ответственного за управление транзакциями.

Библиография

1. Методы захвата и освобождения – [online] [дата обращения 02.03.2023], Доступно: e-educ.ru/bd28.html
2. Сериализация. транзакций – [online] [дата обращения 02.03.2023], Доступно: https://studfile.net/preview/5133502/page:46/
3. Сериализация транзакций с использованием синхронизационных захватов – [online] [дата обращения 02.03.2023], Доступно: https://studme.org/411523/informatika/serializatsiya_tranzaktsiy_ispolzovaniem_sinhronizatsionnyh_zahvatov_blokirovok

PARTIȚIONAREA BAZELOR DE DATE

Victor ȘAPTEFRAȚI

Departamentul Ingineria Software și Automatică, grupa TI-191 F/R, Facultatea CIM, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Victor Șaptefrați, e-mail: saptefrativictor@gmail.com

Conducător științific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *Conceptul de partiționare este o tehnică de descompunere a bazelor de date care o împarte în unități mai mici, independente, numite partiții (sau „shards”). Scopul este de a distribui datele și sarcina pentru a îmbunătăți performanța și scalabilitatea sistemului de baze de date, precum și disponibilitatea datelor în caz de accidente. Punerea în aplicare necesită o planificare atentă, inclusiv alegerea cheii de partiționare, arhitectura partiționării, localizarea fragmentelor și gestionarea comunicării între fragmente.*

Cuvinte cheie*: Partiționare, scalabilitate, distribuție, cheie de partiție.*

Introducere

Partiționarea bazelor de date este o tehnică care permite scalarea bazelor de date relaționale prin distribuirea seturilor de date mari și a sarcinilor de lucru pe mai multe servere. O soluție promițătoare pentru a face față activităților de producere și de procesare a cantităților mari de date, partiționarea bazelor de date este o tehnică care optimizează stocarea datelor și performanța interogărilor prin împărțirea seturilor de date în segmente mai mici, mai ușor de gestionat [1]. Partiționarea bazelor de date este o abordare eficientă care permite organizațiilor să îmbunătățească performanța, scalabilitatea și mentenanța sistemelor lor de gestionare a datelor. Prin împărțirea unui set mare de date în partiții mai mici, această tehnică permite o alocare mai eficientă a spațiului de stocare, o execuție mai rapidă a interogărilor și îmbunătățirea capacităților de procesare paralelă. Partiționarea poate fi implementată la diferite niveluri, cum ar fi partiționarea orizontală (*sharding*) și verticală, fiecare cu setul său unic de avantaje și cazuri de utilizare.

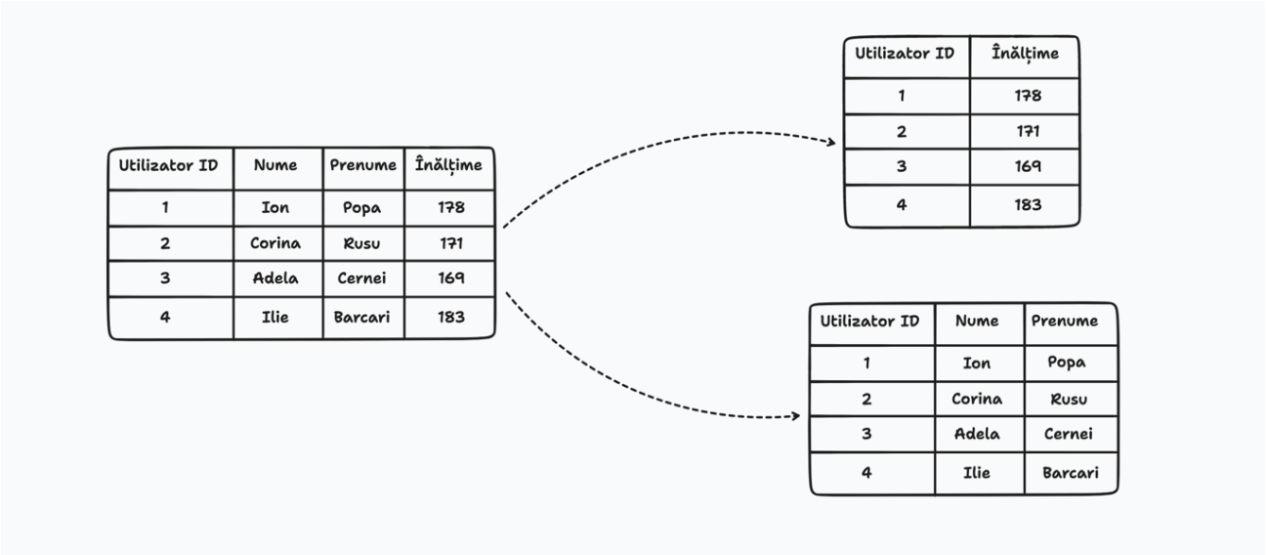


Figura 1. Exemplu de partiționare verticală

Partiționarea verticală

Cunoscută și sub denumirea de partiționarea pe rânduri, partiționarea verticală este o tehnică care facilitează împărțirea unui tabel mare de bază de date în subansambluri mai mici, mai ușor de gestionat, pe baza valorilor unei coloane cheie de împărțire. Această abordare poate îmbunătăți performanța interogărilor prin reducerea volumului de date care trebuie căutate în timpul operațiunilor de recuperare. Cu toate acestea, ea necesită, de asemenea, punerea în aplicare a unor procese de gestionare suplimentare pentru tabelele mai mici și poate prezenta provocări în ceea ce privește asigurarea coerenței datelor în întregul sistem [2].

În *figura 1* se observă conceptul de partiționare verticală, care separă datele coloanelor, *Nume și Prenume* și *Înălțime* în două tabele distincte.

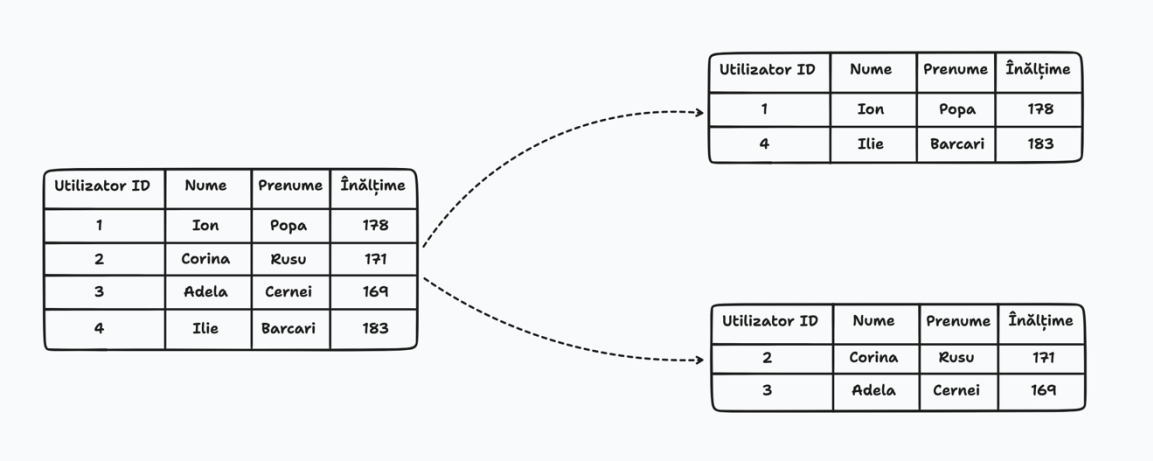


Figura 2. Exemplu de partiționare orizontală

Partiționarea orizontală

O altă o metodă de partiționare este cea orizontală, ilustrată în *figura 2*, constă în descompunerea tabelelor de baze de date în tabele mai mici, în funcție de un anumit interval al cheii de partiție. Acest lucru se realizează prin împărțirea datelor în intervale bazate pe o coloană, cum ar fi data sau ID-ul, și prin crearea unor tabele separate pentru fiecare interval. Acest lucru îmbunătățește performanța și scalabilitatea prin distribuirea datelor și a volumului de lucru pe mai multe servere sau locații și îmbunătățește disponibilitatea datelor și recuperarea în caz de dezastru, permițând realizarea de copii de rezervă și restaurare independente pentru fiecare partiție.

Beneficiile partiționării

Principalele beneficii ale partiționării unei baze de date sunt:

- *Flexibilitate crescută* – permite scalarea orizontală, prin adăugarea mai multor mașini la o stivă existentă, permițând un trafic mai mare și o procesare mai rapidă.
- *Timp de răspuns mai rapid* – împărțirea unei tabele în mai multe reduce numărul de rânduri pe care o interogare trebuie să le caute, ceea ce duce la timpi de răspuns mult mai rapizi.
- *Fiabilitate crescută* - partiționarea unei baze de date atenuează impactul întreruperilor, minimizând impactul general asupra aplicației sau site-ului web.
- *Gestiunea bazelor de date mari* – partiționarea este deosebit de util pentru aplicațiile cu o bază de date mare, monolitică, care altfel ar fi limitată de capacitatea de stocare și de calcul a unei singure mașini.
- *Securitate îmbunătățită* – partiționarea unei baze de date permite ca datele să fie partiționate și stocate în locații separate, ceea ce face mai dificil pentru hackeri să acceseze informații sensibile.
- *Performanță îmbunătățită* – partiționarea permite ca baza de date să fie distribuită pe mai multe mașini, rezultând o performanță îmbunătățită și o mai bună utilizare a resurselor.

Dezavantajele partiționării

Împărțirea unei baze de date poate îmbunătăți performanța și poate facilita extinderea, dar are și limitări. Una dintre principalele provocări ale partiționării este complexitatea implementării corecte, care poate duce la pierderea de date sau la tabele corupte. O altă problemă este dezechilibrarea partițiilor, care poate cauza încetinirea și blocarea aplicației pentru o parte semnificativă a utilizatorilor. Un alt dezavantaj major al partiționării este faptul că poate fi dificil să readucă baza de date la arhitectura sa originală. Ar trebui să se acorde o atenție deosebită înainte de a decide să se utilizeze o bază de date în mod fragmentat, deoarece este posibil ca aceasta să nu fie cea mai bună soluție pentru orice situație.

Arhitectura partiționării

Atunci când este descompusă o bază de date pe orizontală, este important să se determine modul în care datele vor fi distribuite între fragmente. Există mai multe metode de partiționare, fiecare având avantaje și dezavantaje specifice. Alegerea metodei potrivite depinde de cerințele specifice ale aplicației și de capacitatea de a gestiona și optimiza funcțiile de partiționare, intervalele de valori sau structurile de directoare. Deosebim trei tipuri principale de arhitectură de partiționare:

- ***Partiționare bazată pe chei*** utilizează o valoare din datele primite, cum ar fi ID-ul unui client, pentru a determina în ce partiție ar trebui să ajungă datele. Aceasta facilitează operațiunile paralele de citire și scriere, deoarece datele pot fi accesate și procesate pe mai multe servere simultan. O funcție partiție este utilizată pentru a converti valoarea într-un ID de partiție. Cu toate acestea, această metodă poate îngreuna adăugarea sau eliminarea dinamică a serverelor în baza de date, de aceea optimizarea funcției de partiționare și implementarea unor strategii eficiente de gestionare a modificărilor pot oferi o soluție robustă și de înaltă performanță pentru aplicațiile moderne.
- ***Partiționare bazată pe intervale*** presupune împărțirea datelor pe baza unor intervale de valori date, cum ar fi intervalele de prețuri. Această metodă este simplu de implementat, însă aceasta vine cu un dezavantaj prin faptul că nu asigură distribuirea uniformă a datelor, selectarea necorespunzătoare a intervalelor de valori poate duce la o încărcare neuniformă a serverului și poate avea un impact asupra performanței. O soluție pentru aceste provocări reprezintă o selecție atentă a intervalelor de valori, ajustări dinamice, monitorizare constantă a performanței și a zonelor cu accesări frecvente ca strategii de optimizare și echilibrare.
- ***Partiționare pe bază de directoare*** utilizează un tabel de căutare pentru a ține evidența ce date deține o anumită partiție. Această metodă necesită întreținerea tabelului de căutare, dar permite o distribuție mai flexibilă și mai eficientă a datelor și realocarea dinamică a datelor. Oferind cea mai mare flexibilitate, deoarece nu necesită o funcție de partiționare fixă sau intervale predefinite de valori, optimizarea structurii și gestionării tabelului de căutare este importantă și devine o sarcină complexă suplimentară în gestionarea bazei de date. Introducerea tehnicilor precum replicarea tabelului și algoritmi de căutare eficienți sunt esențiali pentru a asigura performanța.

Concluzii

În concluzie, partiționarea bazelor de date reprezintă o soluție eficientă pentru gestionarea și scalarea bazelor de date relaționale, ce vine ca un răspuns provocărilor asociate cu manipularea unor volume mari de date și sarcini de lucru complexe. Prin implementarea tehnicilor de partiționare verticală și orizontală, poate fi îmbunătățită performanța, scalabilitatea și disponibilitatea sistemelor lor de gestionare a datelor. Partiționarea verticală permite o gestionare mai ușoară a tabelelor mari prin divizarea acestora în subansamble bazate pe coloane-cheie, în timp ce partiționarea orizontală facilitează distribuirea datelor și a sarcinilor de lucru pe mai multe servere, îmbunătățind astfel capacitatea de procesare paralelă și redundanța datelor.

Implementarea partiționării bazelor de date necesită o planificare atentă și o gestionare corespunzătoare a cheilor de partiție, arhitecturii și comunicării între fragmente. Provocări precum mentenanța tabelelor mai mici și asigurarea coerenței datelor pot apărea în timpul implementării, necesitând strategii adecvate pentru a le soluționa.

Referinţe

1. ERIC RIES, *Sharding for startups* [online]. 2009. [accesat 20.11.2022]. Disponibil: http://www.startuplessonslearned.com/2009/01/sharding-for-startups.html
2. MAHDI YUSUF, *Database Sharding Explained* [online]. 2022 [accesat 22.11.2022]. Disponibil: https://architecturenotes.co/database-sharding-explained

LIMBAJE PENTRU BAZE DE DATE

Ana CASTRAVEȚ

Departamentul Ingineria Software și Automatică, grupa TI-181 F/R, Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Ana CASTRAVEȚ, e-mail: ana.bolocan@isa.utm.md

Conducător științific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *Acest rezumat se refera la limbajele utilizate pentru a interoga bazele de date. Limbajul SQL (Structured Query Language) este cel mai utilizat limbaj in bazele de date relaționale și este utilizat pentru a crea, a modifica și a interoga bazele de date. Limbajul PL/SQL (Procedural Language/ Structured Query Language) este un limbaj de programare utilizat în bazele de date Oracle. Bazele de date NoSQL utilizează, de asemenea, limbaje de interogare specifice, cum ar fi MongoDB Query Language și Cassandra Query Language. Toate aceste limbaje sunt esențiale in gestionarea datelor și permit utilizatorilor să efectueze operatii de tipul select, insert, update si delete.*

Cuvinte cheie: *SQL (Structured Query Language), interogare, programare, funcții, proceduri, operațiuni.*

Introducere

Limbajele pentru bazele de date sunt utilizate pentru a comunica cu sistemul de gestiune a bazei de date (DBMS) și pentru a manipula datele stocate în aceasta.

SQL este unul dintre cele mai populare și utilizate limbaje pentru baze de date. Acesta este utilizat pentru a crea, modifica și interoga bazele de date. SQL este un limbaj declarativ, ceea ce înseamnă că utilizatorii trebuie doar să specifice ceea ce doresc să realizeze, iar sistemul de gestiune a bazei de date va determina cum să îndeplinească aceste cerințe [1].

PL/SQL și T-SQL sunt limbaje procedurale care permit utilizatorilor să creeze proceduri stocate și funcții pentru a manipula datele. Acestea oferă utilizatorilor un grad mai mare de control și flexibilitate în manipularea datelor. În plus, exista și alte limbaje specializate pentru bazele de date, cum ar fi XML, care este utilizat pentru a gestiona datele în format XML, și RDF, care este utilizat pentru a gestiona datele în format PDF.

În general, limbajele pentru baze de date sunt esențiale pentru a lucra cu datele stocate în bazele de date și pentru a realiza operațiuni precum crearea, actualizarea, ștergerea și interogarea acestora. Întelegerea și utilizarea acestor limbaje este importantă pentru oricine dorește să lucreze cu baze de date.

Ce este SQL și cum funționează?

SQL (Structured Query Language) este un limbaj de programare utilizat pentru gestionarea și manipularea datelor în bazele de date relaționale. Acesta permite utilizatorilor să creeze, să modifice și să întrebe bazele de date pentru a obține informațiile dorite [2].

Ce este NoSQL și care sunt principalele sale caracteristici?

NoSQL (Not only SQL) este un termen generic care se referă la bazele de ate non-relaționale. Acestea sunt proiectate pentru a gestiona cantități mari de date, care pot fi structurate sau nestructurate. Principalele carateristici ale bazelor de date NoSQL includ scalabilitatea orizontală, toleranță la eșecuri, flexibilitatea structurii datelor și performanța.

Ce este MongoDB și cum funcționează ?

MongoDB este o bază de date NoSQL care utilizează un model de stocare documental. Acesta stochează datele în documente JSON (JavaScript Object Notation) și utilizează o abordare distribuită pentru a asigura scalabilitatea orizontală [3].

Ce este MySQL și cum funcționează?

MySQL este o bază de date relațională, care utilizează un limbaj de interogare SQL pentru a gestiona și manipula datele.

Ce sunt limbajele de interogare și care este rolul lor în gestionarea bazei de date?

Limbajele de interogare sunt limbaje de programare utilizate pentru a interoga bazele de date și a obține informații specifice. Acestea permit utilizatorilor să creeze interogări complexe pentru a căuta, sorta și filtra datele în baza de date. Prin utilizarea limbajelor de interogare, utilizatorii pot accesa și manipula datele in mod eficient și precis.

Limbajele de bază de date oferă suport pentru proceduri stocate, inclusiv SQL și PL/SQL pentru Oracle, Transact-SQL pentru Microsoft SQL Server, PL/pgSQL pentru PostgreSQL și PL/SQL pentru MySQL [4].

Procedura stocate pot fi utilizate pentru a realiza o varietate de sarcini, cum ar fi:

1. Validarea datelor introduse în baza de date;
2. Crearea și gestionarea tabelelor și a altor obiecte de bază de date;
3. Efectuarea de interogări complexe și raportări de date;
4. Actualizarea datelor existente în baza de date;
5. Executarea de operații de sincronizare și de replicare a datelor;
6. Gestionarea tranzacțiilor și a erorilor de bază de date;

Un avantaj important al procedurilor stocate este că acestea pot fi optimizate, ceea ce duce la o performanță mai buna și la o reducere a traficului de rețea. De asemenea, procedurile stocate pot fi securizate și controlate pentru a limita accesul la date și pentru a evita erorile de utilizare a bazei de date [5].

În general, utilizarea procedurilor stocate poate duce la o îmbunătățire semnificativă a performanței aplicaţtiilor de bază de date și la o reducere a complexității codului.

Concluzii

În concluzie, limbajele de bază de date sunt esențiale pentru gestionarea și organizarea datelor într-o manieră eficientă și ușor de accesat. Acestea permit programatorilor să creeze, să acceseze și să manipuleze datele dintr-o bază de date, oferind instrumentele necesare pentru a efectua operațiuni complexe și a asigura integritatea datelor.

Limbajele de bază de date diferă de la o platformă la alta, dar cele mai utilizate sunt SQL și NoSQL. SQL este un limbaj relațional utilizat în bazele de date relaționale, în timp ce NoSQL este utilizat în bazele de date non-relaționale. În plus, există și alte limbaje de bază de date, cum ar fi PL/SQL, T-SQL și PL/pgSQL, care sunt utilizate pentru a crea proceduri stocate și funcții.

Procedurile stocate sunt de asemenea foarte importante în gestionarea datelor într-o bază de date și permit programatorilor să creeze funcții personalizate care pot fi apelate pentru a efectua o varietate de sarcini.

În general, utilizarea limbajelor de bază de date și a procedurilor stocate poate duce la o gestionare mai eficientă și securizată a datelor și poate îmbunătăți performanța aplicațiilor care accesează bazele de date.

Referințe

1. Documentația oficială SQL [online]. [accesat 20.11.2022]. Disponibil: -SQL Tutorial (w3schools.com);
2. Documentația oficială NoSQL [online]. [accesat 20.11.2022]. Disponibil: -What Is NoSQL? NoSQL Databases Explained | MongoDB;
3. Documentația oficială Oracle PL/SQL [online]. [accesat 20.11.2022]. Disponibil: -Oracle Database Database PL/SQL Language Reference, 19c;
4. Documentația oficială Microsoft T-SQL [online]. [accesat 20.11.2022]. Disponibil: - Transact-SQL reference (Database Engine) - SQL Server | Microsoft Learn;
5. Documentația oficială PostgreSQL PL/pgSQL [online]. [accesat 20.11.2022]. Disponibil: - PostgreSQL: Documentation: 15: Chapter 43. PL/pgSQL – SQL Procedural Language.

АДМИНИСТРИРОВАНИЕ БАЗ ДАННЫХ

Александр ОСИНСКИЙ

Департамент Программной Инженерии и Автоматики, группа TI-192 F/R,
Факультет Вычислительной Техники, Информатики и Микроэлектроники,
Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Александр ОСИНСКИЙ, e-mail: osinschii.alexandr@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Администрирование данных — управление информационными ресурсами, включая планирование базы данных, разработку и внедрение стандартов, определение ограничений и процедур, а также концептуальное и логическое проектирование баз данных. Администратор данных отвечает за корпоративные информационные ресурсы, включая и не компьютеризированные данные.*

Cuvinte cheie*: Администрирование баз данных, разработка моделей, поддержка словаря данных организации, автоматизация процессов*

1. Что такое администрирование баз(ы) данных?

Администрирование баз данных - это функция управления и обслуживания программного обеспечения систем управления базами данных (СУБД). Стандартное программное обеспечение СУБД, такое как Oracle, IBM DB2 и Microsoft SQL Server, требует постоянного управления. Таким образом, корпорации, использующие программное обеспечение СУБД, часто нанимают специалистов информационных технологий, называемых администраторами баз данных или администраторами баз данных [1].

2. Основные обязанности

Администратор данных отвечает за корпоративные информационные ресурсы и данные. Это часто связано с управлением данными, которые являются совместно используемым ресурсом для различных пользователей и прикладных программ данной организации. В разных организациях количество сотрудников может отличаться и обычно определяется размерами самой организации. Одной из основных обязанностей можно отметить обмен консультациями и советами со старшими менеджерами, а также в слежении за тем, чтобы применение технологий баз данных продолжало соответствовать корпоративным целям. Должность администратора обычно принадлежит отделу информационных систем организации. В одних случаях администрирование данных может представлять собой отдельную функциональную задачу, а в других — совмещаться с администрированием базы данных. Итак, из обязанностей администратора можно отметить следующее:[2]

- Установка, настройка и обновление ПО сервера БД
- Установка и поддержка надежной политики и процедуры резервного копирования и восстановления данных
- Внедрение лучших практик по обеспечению безопасности данных
- Мониторинг производительности
- Масштабируемость
- Работа в команде и круглосуточный мониторинг

3. Типы администраторов баз данных

Существует 3 типа администраторов баз данных:

1. Администраторы баз данных-разработчиков. Они сосредоточены на развитии проекта и сотрудничают ближе с разработчиками. Данный аспект администрирования сосредоточен на логике, проектировании и обслуживании моделей данных. Также, генерация DDL (язык определения данных), написания и настройка SQL, добавление/редактирование хранимых процедур, которые наиболее подходят для функциональности СУБД.
2. Системные администраторы баз данных. Они более сосредоточены на установке СУБД, конфигурации, исправлении, резервном копировании данных, восстановлении, оптимизации, аварийном восстановлении и производительности БД.
3. Администраторы баз данных приложений. Администраторы баз данных приложений преодолевают барьер между СУБД и прикладным программным обеспечением и несут ответственность за обеспечение полной оптимизации приложения для базы данных и наоборот. Обычно они управляют всеми компонентами приложения, которые взаимодействуют с базой данных и выполняют такие действия, как установка и исправление приложений, обновление приложений, клонирование базы данных, создание и выполнение подпрограмм очистки данных и загрузка данных управление процессами.

4. Автоматизация администрирования баз данных

Также можно использовать комбинацию этих подходов для создания более комплексной стратегии автоматизации. Автоматизация администрирования баз данных может помочь повысить эффективность и надежность ваших систем баз данных и высвободить время для администраторов баз данных, чтобы они могли сосредоточиться на более сложных задачах.

1. Инструменты планирования: это инструменты, которые позволяют автоматизировать рутинные задачи обслуживания, планируя их запуск в определенное время или через определенные промежутки времени. Например, вы можете использовать инструмент планирования для автоматического запуска резервного копирования каждую ночь в полночь или еженедельной оптимизации индексов. Некоторые примеры инструментов планирования включают планировщик заданий Windows и daemon cron Unix.

2. Инструменты управления конфигурацией: это инструменты, которые позволяют автоматизировать развертывание серверов баз данных и управление ими, включая такие задачи, как подготовка новых серверов, применение исправлений и обновлений, а также настройка параметров сервера. Некоторые популярные инструменты управления конфигурацией включают Puppet, Chef и Ansible.

3. Инструменты мониторинга: это инструменты, которые постоянно контролируют производительность и состояние ваших систем баз данных и предупреждают вас, когда возникают проблемы, которые необходимо решить. Некоторые средства мониторинга также включают в себя функции, позволяющие автоматически решать проблемы, такие как перезапуск отказавших служб или перераспределение ресурсов для устранения узких мест в производительности. Примеры инструментов мониторинга включают Nagios, Zabbix и монитор производительности базы данных SolarWinds.

4. Функции самовосстановления и самонастройки. Многие современные системы управления базами данных включают встроенные функции, которые позволяют им автоматически решать проблемы и оптимизировать производительность, не требуя ручного вмешательства. Например, база данных может автоматически добавить серверу больше памяти или ресурсов ЦП, если обнаружит, что сервер сильно загружен, или может автоматически переписать плохо выполняющийся запрос, чтобы использовать более эффективный план выполнения.

5. Инструменты администрирования баз данных

Существует множество инструментов, помогающих администраторам баз данных (DBA) управлять и поддерживать свои системы баз данных. Некоторые распространенные категории инструментов DBA включают:

1. Инструменты резервного копирования и восстановления. Эти инструменты помогают администраторам баз данных создавать резервные копии своих баз данных и управлять ими, а также восстанавливать базы данных из резервной копии в случае потери или повреждения данных. Примеры включают Oracle Recovery Manager (RMAN) и MySQL Enterprise Backup.
2. Инструменты настройки производительности. Эти инструменты помогают администраторам баз данных выявлять и устранять проблемы с производительностью в своих базах данных. Они могут предоставлять такие функции, как оптимизация запросов, анализ индексов и мониторинг производительности в реальном времени. Примеры включают автоматический репозиторий рабочих нагрузок Oracle (AWR) и схему производительности MySQL.
3. Инструменты мониторинга. Эти инструменты обеспечивают представление в режиме реального времени производительности и состояния систем баз данных и могут включать такие функции, как предупреждения, отчеты и анализ тенденций. Примеры включают Oracle Enterprise Manager и MySQL Enterprise Monitor.
4. Инструменты моделирования и проектирования данных. Эти инструменты помогают администраторам баз данных проектировать и моделировать структуры данных и отношения в своих базах данных. Они могут включать такие функции, как визуальное моделирование данных, проектирование и рефакторинг базы данных, а также сравнение и синхронизация баз данных. Примеры включают Oracle SQL Developer Data Modeler и MySQL Workbench.
5. Инструменты контроля версий и управления исходным кодом. Эти инструменты помогают администраторам баз данных отслеживать и управлять изменениями в исходном коде и конфигурации их систем баз данных. Примеры включают Git и Subversion.

6. Заключение

Администратор базы данных (DBA) — это профессионал, который отвечает за проектирование, внедрение, обслуживание и ремонт систем баз данных организации. Администраторы баз данных работают над тем, чтобы базы данных, которыми они управляют, были надежными, эффективными и безопасными, а также отвечали потребностям пользователей организации.

Некоторые из конкретных задач, за которые может нести ответственность администратор баз данных, включают:

- Установка и настройка систем управления базами данных (СУБД)
- Проектирование и создание баз данных
- Импорт и экспорт данных
- Настройка учетных записей и разрешений пользователей
- Выполнение резервного копирования и восстановления
- Мониторинг и оптимизация производительности базы данных
- Применение исправлений и обновлений
- Обеспечение мер безопасности
- Устранение неполадок и устранение проблем с базами данных

Администраторы баз данных обычно работают с различными технологиями баз данных и могут работать с реляционными базами данных, базами данных NoSQL или их комбинацией. Они также могут отвечать за управление серверами баз данных, сетевым оборудованием и другой соответствующей инфраструктурой.

Библиография

1. Общее понятие темы “Администрирование баз данных”. - [online] [дата обращения 02.03.2023], Доступно: https://ru.wikipedia.org/wiki/%D0%90%D0%B4%D0%BC%D0%B8%D0%BD%D0%B8%D1%81%D1%82%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85
2. Обязанности, автоматизация и инструменты администраторов - [online] [дата обращения 02.03.2023], Доступно: ttps://ru.wikibrief.org/wiki/Database_administration

ПОДХОДЫ CODE FIRST, DATABASE FIRST И MODEL FIRST В СОЗДАНИИ БАЗЫ ДАННЫХ ИСПОЛЬЗУЯ ENTITY FRAMEWORK

Роман КИОРОГЛО

Департамент Программной Инженерии и Автоматики, группа TI-192 F/R, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Роман КИОРОГЛО, e-mail: chioroglo.roman@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Резюме. *При разработке на платформе .NET, во время этапа создания базы данных, происходит выбор между несколькими подходами к её созданию, а именно, Code First, Database First и Model First. Каждый из подходов имеет свои приемущества и недостатки перед другими. В данной работе описываются подходы, их отличия и случаи, в которых можно применить каждый из них.*

Ключевые слова*: Базы данных, .NET, Проектирование базы данных, Модель состояния, классы .NET, SQL, миграции*

Введение

При создании базы данных в платформе .NET у разработчика возникает вопрос, какой использовать подход? В Entity Framework существуют три основных подхода для создания базы данных:

Подход Code First предполагает создание базы данных на основе модели данных, которая описывается в коде приложения. На основе этой модели автоматически генерируется схема базы данных. Этот подход подходит для проектов, в которых необходимо быстро начать работу с проектом, не тратя много времени на описание модели данных [1].

Подход Database First предполагает создание базы данных на основе существующей схемы базы данных. Для этого необходимо создать файл модели EDMX на основе существующей базы данных. Этот подход подходит для проектов, в которых есть существующая база данных, с которой нужно работать.

Подход Model First предполагает создание базы данных на основе визуальной модели, которая описывается в Entity Framework Designer. На основе этой модели автоматически генерируется схема базы данных. Этот подход подходит для проектов, в которых необходимо быстро создать модель данных без необходимости создания базы данных с нуля [1].

Каждый подход имеет свои преимущества и недостатки, и выбор зависит от конкретных требований проекта и предпочтений команды разработчиков.

Code First

Code First - это один из подходов к разработке базы данных в Entity Framework, в котором разработка базы данных начинается с создания классов в языке программирования и затем преобразования этих классов в таблицы базы данных. Этот подход позволяет разработчикам сосредоточиться на разработке объектной модели, а не на проектировании базы данных.

Основные шаги в подходе Code-First:

1. Определение классов: Разработчики определяют классы для каждой таблицы в базе данных. Каждый класс представляет сущность в базе данных и имеет свойства, соответствующие столбцам в таблице.

2. Определение отношений: Разработчики определяют отношения между классами, используя аннотации или Fluent API. Отношения могут быть один-к-одному, один-ко-многим или многие-ко-многим.
3. Контекст данных: Разработчики определяют контекст данных, который настроен для работы с базой данных. Контекст данных включает в себя определения классов, отношений и настроек соединения с базой данных.
4. Создание базы данных: После определения классов и контекста данных, Entity Framework создает базу данных на основе этих определений. Разработчики могут выбрать, где хранить базу данных (локально или на сервере).
5. Изменение базы данных: После создания базы данных, разработчики могут использовать механизм миграции для внесения изменений в базу данных, когда требуется изменение модели.

Code-First предоставляет гибкость в разработке базы данных и позволяет разработчикам изменять модель базы данных без необходимости изменения схемы базы данных. Этот подход также обеспечивает лучшую поддержку для разработки с использованием практик Agile, таких как быстрые итерации и TDD (разработка через тестирование) [2].

Рассмотрим пример использования подхода Code First для создания модели данных для простого приложения, которое хранит информацию о книгах в библиотеке.

Определение классов модели данных:

csharp

Copy code

```
public class Book
{
   public int Id { get; set; }
   public string Title { get; set; }
   public string Author { get; set; }
   public int Year { get; set; }
}

public class LibraryContext : DbContext
{
   public DbSet<Book> Books { get; set; }
}
```

Класс Book определяет модель для книги, которая содержит свойства Id, Title, Author и Year. Класс LibraryContext наследуется от класса DbContext и содержит свойство Books, которое представляет коллекцию объектов Book в базе данных [2].

Конфигурация модели данных:

csharp

Copy code

```
public class LibraryContext : DbContext
{
   public LibraryContext() : base("name=LibraryContext")
   {
   }
   protected override void OnModelCreating(DbModelBuilder modelBuilder)
   {
      modelBuilder.Entity<Book>()
         .Property(b => b.Title)
         .IsRequired()
         .HasMaxLength(100);
```

```
        modelBuilder.Entity<Book>()
            .Property(b => b.Author)
            .IsRequired()
            .HasMaxLength(50);
    }
    public DbSet<Book> Books { get; set; }
}
```

В методе OnModelCreating мы можем использовать Fluent API для определения свойств и настроек для нашей модели данных. В данном случае мы устанавливаем обязательное наличие свойств Title и Author и ограничения на максимальную длину.

Создание базы данных:

csharp

Copy code

```
using (var context = new LibraryContext())
{
    context.Database.CreateIfNotExists();
}
```

Создание базы данных происходит автоматически при первом обращении к контексту данных.

Таким образом, подход Code First позволяет определять модель данных с помощью классов и свойств, а затем создавать базу данных на основе этой модели. Благодаря этому подходу разработчик может иметь большую гибкость при работе с моделью данных и управлении ее изменениями.

Database First

Database First - это один из подходов к разработке базы данных в Entity Framework, в котором база данных уже существует, и классы создаются автоматически на основе существующей схемы базы данных. Этот подход позволяет быстро создавать объектную модель на основе существующей базы данных, что может быть полезно, если база данных уже существует или если проектирование базы данных является первоочередным приоритетом.

Основные шаги в подходе Database First:

1. Создание модели из базы данных: Разработчики используют визуальный дизайнер модели Entity Framework или командную строку, чтобы создать модель базы данных на основе существующей базы данных.
2. Генерация классов: После создания модели базы данных Entity Framework автоматически генерирует классы, которые соответствуют таблицам в базе данных. Эти классы могут быть использованы для доступа к данным в базе данных.
3. Контекст данных: Разработчики создают контекст данных, который настроен для работы с базой данных. Контекст данных включает в себя определения классов, отношений и настроек соединения с базой данных.
4. Использование объектной модели: Разработчики могут использовать объектную модель для доступа к данным в базе данных и для выполнения запросов, обновления и вставки данных.

Database First позволяет быстро создавать объектную модель на основе существующей базы данных и предоставляет возможность использовать уже существующую базу данных без необходимости создания модели с нуля. Однако этот подход может быть менее гибким, чем Code-First, когда требуется внести изменения в схему базы данных.

Рассмотрим пример использования подхода Database First для создания модели данных для того же приложения, которое хранит информацию о книгах в библиотеке.

1. Создание базы данных
2. Сначала мы создадим базу данных, содержащую таблицу Books, которая будет использоваться в нашем приложении. Это можно сделать с помощью SQL скрипта или визуального редактора баз данных, такого как Microsoft SQL Server Management Studio.
3. Генерация модели данных
4. Затем мы можем использовать встроенный инструмент в Entity Framework для генерации модели данных на основе существующей базы данных. Для этого нужно выполнить следующие шаги:
   - Добавить новый элемент модели в проект, используя контекстное меню проекта в Visual Studio.
   - В диалоговом окне добавления элемента выбрать "ADO.NET Entity Data Model" и дать ему имя.
   - Выбрать "EF Designer from database" в качестве типа модели и нажать кнопку "Next".
   - Указать строку подключения к базе данных и выбрать таблицы, которые мы хотим использовать в нашей модели данных. В нашем случае это таблица Books.
   - Нажать кнопку "Finish", чтобы сгенерировать модель данных.
5. Использование модели данных

После генерации модели данных мы можем использовать ее в нашем приложении. Например, мы можем получить список всех книг из базы данных следующим образом:

csharp
Copy code
```
using (var context = new LibraryEntities())
{
   var books = context.Books.ToList();
   foreach (var book in books)
   {
      Console.WriteLine($"{book.Id} - {book.Title} by {book.Author}, published in {book.Year}");
   }
}
```

Здесь LibraryEntities - это имя контекста данных, который был сгенерирован вместе с моделью данных. Мы можем использовать его для доступа к данным в базе данных.

Model First

Model First - это один из подходов к разработке базы данных в Entity Framework, в котором разработчики создают модель данных с помощью визуального дизайнера и затем используют эту модель для генерации базы данных и классов Entity Framework. Этот подход удобен, когда требуется создать базу данных, но отсутствует готовая схема или требуется создать модель данных на основе предметной области.

Основные шаги в подходе Model First:

1. Создание модели: Разработчики используют визуальный дизайнер модели Entity Framework для создания модели данных, которая представляет сущности, отношения и атрибуты в предметной области.
2. Генерация базы данных и классов: После создания модели Entity Framework автоматически генерирует базу данных и классы, которые соответствуют модели. Эти классы могут быть использованы для доступа к данным в базе данных.
3. Контекст данных: Разработчики создают контекст данных, который настроен для работы с базой данных. Контекст данных включает в себя определения классов, отношений и настроек соединения с базой данных.

4. Использование объектной модели: Разработчики могут использовать объектную модель для доступа к данным в базе данных и для выполнения запросов, обновления и вставки данных.

Model First позволяет разработчикам быстро создавать модель данных на основе предметной области и генерировать базу данных и классы на основе этой модели. Этот подход удобен для разработки приложений с небольшой или средней сложностью, где требуется быстрый старт без особого уделяя внимания разработке схемы базы данных. Однако он может быть менее гибким, чем Code-First, когда требуется внести изменения в схему базы данных.

Выводы

Выбор подхода к работе с Entity Framework (Code First, Database First или Model First) зависит от конкретной задачи и требований к проекту. Рассмотрим некоторые рекомендации по выбору подхода.

Code First

Подход Code First подходит для проектов, в которых:

- Необходимо создать базу данных с нуля.
- Нужна гибкость в определении модели данных.
- Есть предпочтение использованию кода для описания модели данных вместо визуального редактора.
- Проект разрабатывается командой разработчиков, которые знакомы с принципами объектно-ориентированного программирования.

Database First

Подход Database First подходит для проектов, в которых:

- Есть существующая база данных, с которой нужно работать.
- Необходимо быстро начать работу с проектом, не тратя много времени на описание модели данных.
- Необходимо работать с базой данных, которая разработана вне проекта.
- Необходимо работать с базой данных, которая поддерживается другими системами.

Model First

- Подход Model First подходит для проектов, в которых:
- Нужно быстро создать модель данных без необходимости создания базы данных с нуля.
- Нужно создать визуальное представление модели данных, которое будет использоваться в проекте.
- Есть возможность создать модель данных визуально, используя Entity Framework Designer.
- Есть необходимость быстро изменять структуру базы данных, не затрагивая код.

В целом, выбор подхода зависит от конкретных требований проекта и предпочтений команды разработчиков. Каждый подход имеет свои преимущества и недостатки, и выбор зависит от конкретной ситуации.

Библиография

1. Tutorial: Get Started with Entity Framework 6 Code First using MVC 5. [online] [дата обращения 02.03.2023], Доступно: https://learn.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application
2. Database Initialization in Entity Framework 6. [online] [дата обращения 02.03.2023], Доступно: https://www.entityframeworktutorial.net/code-first/database-initialization-in-code-first.aspx

UTILIZAREA DIVERSELOR TEHNOLOGII BD ÎN DEPENDENȚĂ DE SCOP

Filip FLORIN

Departamentul Ingineria Software și Automatică, grupa TI-181 F/R , Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Filip FLORIN, e-mail: florin.filip@isa.utm.md

Conducător științific: Dorian SARANCIUC, DISA, FCIM, UTM

Abstract: *Această lucrare descrie care sunt tehnologiile BD contemporane si cu care din ele trebuie sa lucram atunci cand avem anumite scopuri pe care vrem sa le atingem. In ziua de azi avem o varietate larga de tehnologii BD care ofera diferite facilități in planul flexibilitatii, comodității, functionalitatii, performantei, vitezei si a cantitatii de resurse utilizate si alegerea tehnologiei potrivite va asigura o functionare corecta si eficienta a sistemului informatic in care este utilizata.*

Cuvinte cheie*: Tehnologiile BD, Alegerea tehnologiei, Cerințe BD, Scopuri BD, Utilizarea BD.*

Întroducere

In ziua de astazi avem o larga varietate de tehnologii de baze de date care se deosebesc prin faptul ca au fost proiectate cu diverse seturi de priorități: viteza de operare, cantitatea de resurse utilizate, viteza de scriere, viteza de citire, strictetea respectarii de reguli, flexibilitatea, universabilitatea, performanta, etc [1].

Totodata, asupra produselor software contemporane sunt inaintate cerinte de operare cu volume mari de date la viteze exceptionale, aceea ce ne impune sa alegem cu atentie tehnologia care se va potrivi cel mai bine cu scopurile si cerintele care sunt fata de produsele software pentru a atinge o eficienta maxima.

1. Criterii de selectare ale tehnologiei

În bazele de date, la alegerea tehnologiei se folosesc criterii diferite:[2]

- Personalul de achiziții pentru baze de date acordă mai multă atenție costurilor de achiziție, inclusiv cerințelor de stocare și rețea .
- Administratorilor de baze de date (DBA) le pasă de:
  - Costuri de operare si intretinere :
    - Un sistem fiabil de monitorizare și alertă
    - Suport pentru backup și restaurare
    - Costuri rezonabile de actualizare și migrare
    - O comunitate activă de sprijin
    - Ușurință de reglare a performanței
    - Ușurință de depanare
  - Stabilitatea serviciului:
    - Suport pentru mai multe replici(duplicate) de date
    - Servicii foarte disponibile
  - Performanţă:
    - Latența (Ascuns, care nu se observă, nu se manifestă, dar poate izbucni oricând.)
    - Interogări pe secundă (QPS)
    - Dacă acceptă funcții de stocare ierarhice mai avansate
  - Scalabilitate: dacă este ușor de scalat orizontal și vertical
  - Securitate: Indiferent dacă îndeplinește cerințele de audit și previne injecțiile SQL și scurgerea de informații

- Dezvoltatorilor de aplicații le pasă de:
  - Servicii stabile
  - Performanţă
  - Scalabilitate
  - Ușurința dezvoltării interfeței bazei de date
  - Ușurința de modificare a schemei bazei de date
  - Usurinta interactionarii cu baza de date

2. Tipuri de sisteme

Sistemele bazelor de date au 4 mari categorii:
SQL - baze de date relaționale, care au structura și strictețe
NoSQL - baze de date nerelationale, care au flexibilitate.
OLTP - care sunt orientate spre capacitatea de a face repede multe tranzacții
OLAP - care sunt orientate spre puterea mare analitică

3. Utilizarea tehnologiei în dependentă de scop

Tehnologiile populare si masurile in care ele apartin fiecarui tip de tehnologie sunt prezentate in urmatoarea figura:

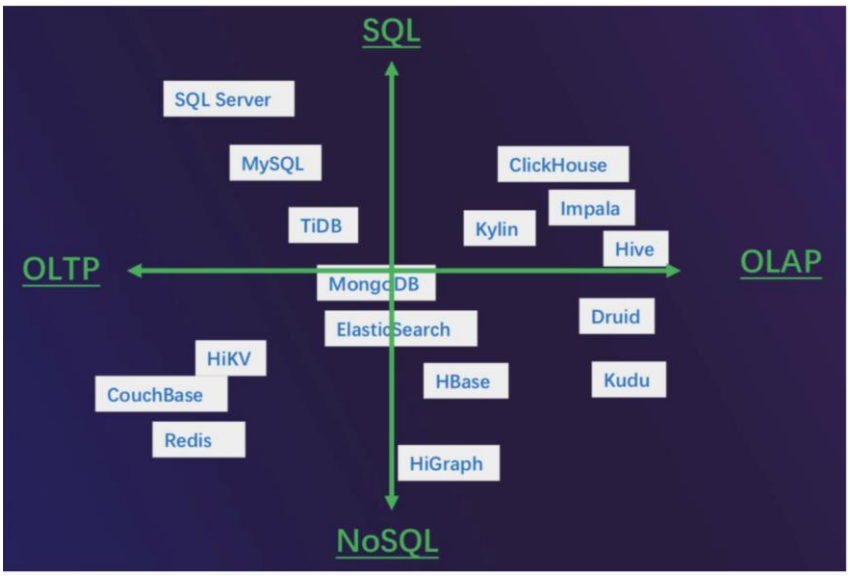


Figura 1 – Apartenenta limbajelor la tipuri de tehnologii

- Axa X reprezintă scenarii de aplicație: OLTP vs. OLAP.
- Axa Y reprezintă interfețele bazei de date: SQL vs. NoSQL.

  Această figura are următoarele caracteristici:
- Pe stanga
  - În colțul din stânga sus

Bazele de date acceptă încărcături de lucru OLTP și limbajul SQL. De exemplu, MySQL acceptă diferite niveluri de izolare a tranzacțiilor, QPS ridicat și latență scăzută. Îl folosim în principal pentru a stoca informații despre tranzacții și date critice, cum ar fi comenzi și informații VIP.

  - În colțul din stânga jos

Folosim baze de date NoSQL pentru a optimiza scenarii speciale. În general, aceste baze de date au scheme simple sau sunt fără schemă, cu debit mare și latență scăzută. Le folosim în principal ca cache-uri sau baze de date cheie-valoare (KV).

- Pe dreapta

Toate sunt sisteme analitice de date mari OLAP, cum ar fi ClickHouse și Impala. În general, acceptă limbajul SQL și nu acceptă tranzacții. Au o scalabilitate bună și o latență lungă de răspuns. Putem adăuga mașini pentru a mări capacitatea de stocare a datelor, iar întârzierea de răspuns este mai mare.

- În jurul punctului de întâlnire al celor două axe

Aceste baze de date sunt neutre și le numim baze de date HTAP, cum ar fi TiDB. Când cantitatea de date este mică, au performanțe bune. Când dimensiunea datelor este mare sau interogările sunt complexe, performanța lor nu este rea. În general, pentru a răspunde nevoilor diferitelor aplicații, folosim diferite motoare de stocare și motoare de interogare.

Arbori de decizie practice pentru alegerea eficientă a unei baze de date
pentru o baza de date relationala:

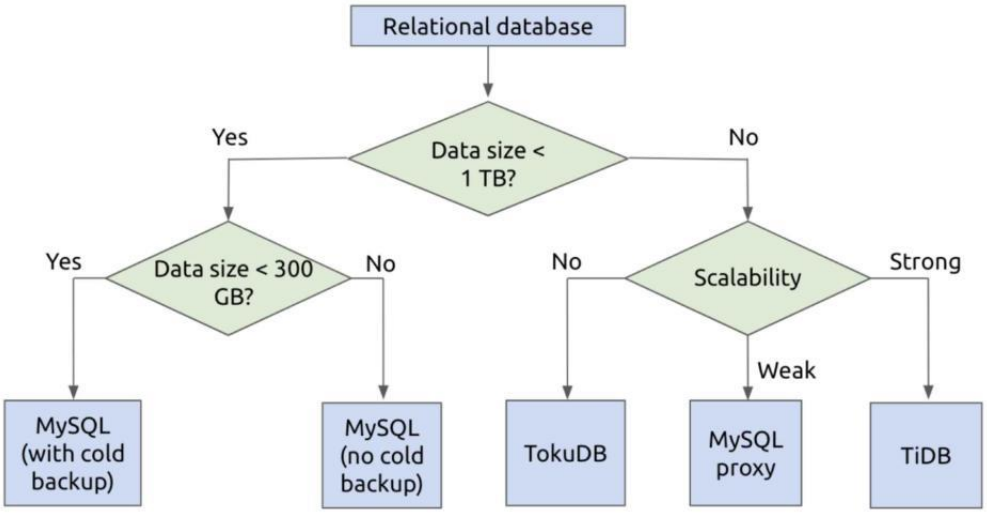


Figura 2 – Arbore decizie pentru alegerea unei baze de date relationale

pentru o baza de date NOSQL

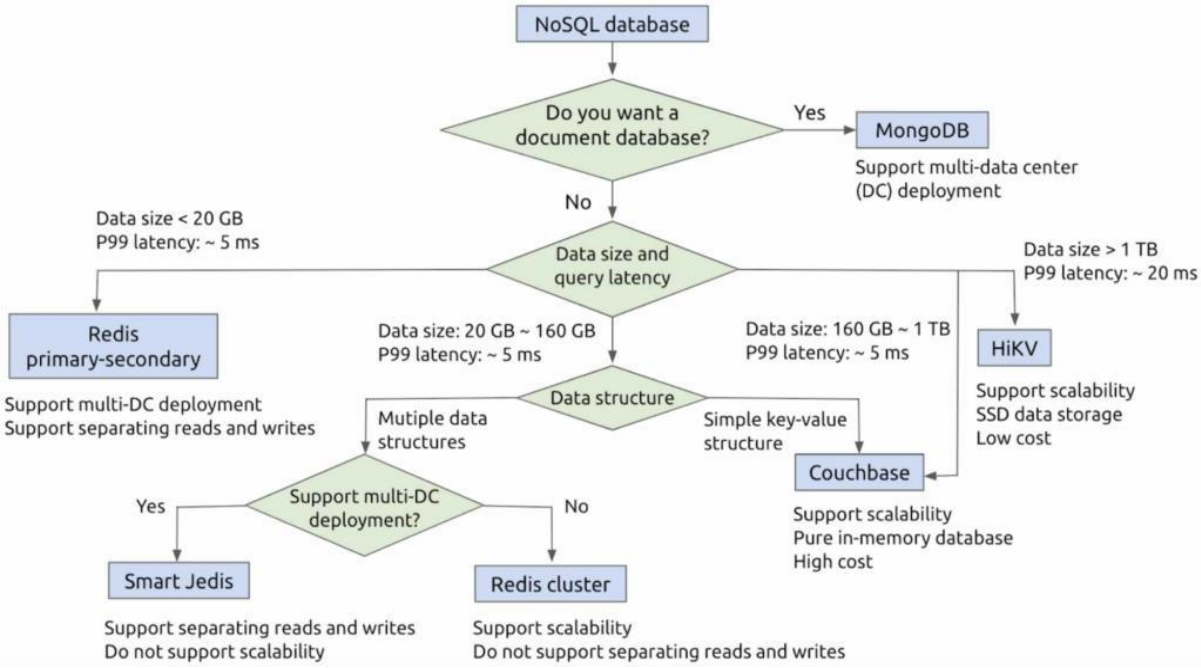


Figura 3 – Arbore de decizie pentru alegerea unei baze de date NOSQL

Concluzii

Având la dispoziţie toate aceste informaţii despre tehnologiile moderne populare utilizate pentru baze de date putem ajunge la concluzia ca varietatea proprietatilor acestora, impreuna cu cunostintele despre particularitatile lor ne vor permite sa alegem tehnologia care se va potrivi cel mai bine cu scopurile noastre. In ziua de astazi asupra produselor software sunt inaintate cerinte ridicate de performanta si alegerea unei tehnologii de baze de date potrivita va contribui pozitiv asupra eficientei sistemului informatic, ceea ce ne va asigura cu proprietatile optime ale acestuia si ne va permite sa avem rezultate placute in procesul dezvoltarii, deservirii si a exploatarii sistemului nostru informatic.

Bibliografie

1. @anuupadhyay, How to Choose The Right Database for Your Application? [online] [Accesat: 23.11.2022]. Disponibil: https://www.geeksforgeeks.org/how-to-choose-the-right-database-for-your-application/
2. Leitao Guo, How to Efficiently Choose the Right Database for Your Applications [online] [Accesat: 23.11.2022]. Disponibil: https://en.pingcap.com/blog/how-to-efficiently-choose-the-right-database-for-your-applications/

INIȚIERE ÎN TEORIA BAZELOR DE DATE

Ariadna ONISIM

Departamentul Ingineria Software și Automatică, grupa TI-191 F/R , Facultatea Calculatoare, Informatică și Microelectronică, Universitatea Tehnică a Moldovei, Chișinău, Republica Moldova

Autorul corespondent: Ariadna ONISIM, e-mail: ariadna.onisim@isa.utm.md

Conducător științific: Dorian SARANCIUC, DISA, FCIM, UTM

Rezumat. *Acest articol propune o descriere colectivă a bazelor de date, și anume: noțiuni generale, avantajele și dezavantajele utilizării, tipurile și de ce se recomandă să aplicați una și ce fac diferitele secțiuni ale unei baze de date. Nomenclatura este axată spre bazele de date generale, și folosește un întreg ansamblu de produse pentru baze de date. O bază de date conține toate comunicările necesare despre obiectele ce intervin într-o totalitate de aplicații, și ajustează relațiile logice dintr-o firmă și tehnicile de adaptare pentru aceste informații. În bazele de date se face o înglobare a datelor, în ideea că mai multe fisiere sunt citite în totalitate, excluzându-se pe cât posibil informatiile inutile. In acelasi moment se aprobă accesul concomitent la aceleasi date a mai multor persoane.*

Cuvinte cheie*: SGBD, BD, DD, LMD, LDD*

1. Introducere

Majoritatea informațiilor sunt colectate și organizate folosind o bază de date. Acestea pot stoca informații despre persoane, lucruri, afaceri, școli sau orice altceva. Mai multe baze de date pot fi utilizate ca sumă într-o aplicație de procesare a textului sau ca flux de calcul. Pe măsură ce lista crește, datele redundante și inconsistente încep să apară. Acestea devin dificil de înțeles atunci când sunt prezentate ca o listă și există doar câteva metode pentru căutarea sau extragerea intervalelor de date pentru verificare. Atunci când aceste dificultăți apar pentru prima dată, este înțelept să modificați datele dintr-o bază de date creată de un sistem de gestionare a bazelor de date (DBMS), cum ar fi Access [1].

O bază de date computerizată este o colecție de informații care poate include mai multe tabele. De exemplu, un sistem de control al inventarului care utilizează trei tabele nu reprezintă trei baze de date, ci mai degrabă o singură bază de date care are trei table. O bază de date pentru Access stochează tabele împreună cu alte obiecte, cum ar fi formule, rapoarte, macrocomponente și module într-un singur fișier dacă nu au fost plătite taxe suplimentare pentru a utiliza datele sau codul dintr-o altă sursă. Baze de date create în versiunile anterioare de Access au o extensie de fișier finală de.accdb, în timp ce bazele de date Access mai vechi au o extensiune de fișiere de.mdb. Puteți utiliza Access 2016, Access 2013, Access 2010 sau Access 2007 pentru a crea fișiere într-un format de arhivare mai mare.

Baza de date reprezintă o culegere de documente sau de date structurate, stocate electronic într-un computer. Aceasta este verificată, de regulă, de un sistem de management al bazelor de date. Împreunate, datele, sistemele de management și aplicațiile asociate reprezintă un sistem de baze de date, numit prescurtat bază de date. QL este un limbaj de programare folosit în mare parte de către toate bazele de date relaționale, pentru interogarea, gestionarea și definirea datelor, precum și pentru controlul oferirii accesului. Cu ajutorul companiei Oracle, în anii 1970 a fost avansat întâi SQL de IBM, acesta a fost un pas esențial ce a dus la implimentarea standardului SQL ANSI. Deși acest limbaj este în continuare folosit la scară largă, apar în continuare noi limbaje de programare.

În cele mai elementare baze de date, informațiile sunt împărțite pe linii și coloane, în diferite tabele, pentru a eficientiza procesarea și interogarea datelor. Acestea se pot accesa, gestiona, schimba, actualiza, controla și organiza cu ușurință. Totalitatea bazelor de date folosesc un limbaj de programare structurat de interogare pentru scrierea și interogarea datelor.

Limbajul de programare SQL permite interogări dinamice asupra unor seturi de informații complexe, astfel este inutil utilizarea unei baze de date SQL doar pentru stocarea persistentă a datelor orientate obiect.

Pentru a oferi programatorului posibilitatea ca aplicațiile să depoziteze date care nu au o schemă predefinită, în 1998 vin în ajutor bazele de date noSQL. De atunci până în prezent aceste baze de date au crescut și dezvoltat imens. Acestea pot fi folosite pentru a marca o treaptă importantă în dezvoltarea de la baze de date relaţionale către baze de date cu performanţe ridicate [2].

Bazele de date NoSQL exprimă şase trăsături de bază, și anume:

- Posibilitatea de a scala orizontal pe multiple servere;
- Abilitatea de a împărți datele pe diverse servere;
- CLI (call level interface) - oferă simplitate;
- Modelul concurenţial este mai slabit decât cel relaţional ;
- Aplicare utilă a RAM-ului pentru o stocare cu mult mai eficientă;
- Se pot adăuga noi atribute la înregistrările ce deja există.

2. Avantajele utilizării bazelor de date

Bazele de date au numeroase avantaje, și anume îmbunătățește eficiența, usurează organizarea și elimină informațiile inutile, în timp ce dezavantajele sunt probleme de potrivire cu computerele și costuri mai ridicate ale software-ului [3].

Sistemele de agregare a datelor sunt folosite pentru a colecta, organiza și sistematiza seturi mari de date care sunt valoroase pentru cercetare în domeniile medicinei, științei, finanțelor și multor alte discipline. Atunci când ating un volum mare, datele pot deveni redundante și dificil de gestionat. De asemenea, ele pot deveni repetitive. Prin urmare, o metodă de colectare, stocare și control al datelor este destul de valoroasă în multe profesii. Capacitatea sistemelor de baze de date de a se sincroniza cu numeroase sisteme de calcul le face cruciale. Acestea sunt produse ale numeroaselor întreprinderi independente și sunt compatibile cu numeroase aplicații existente. După finalizarea lucrărilor, bazele de date pot oferi un fel de securitate, deoarece stochează în siguranță informațiile într-o singură locație electronică. În plus, bazele de date îmbunătățesc coerența generală și reduc numărul și severitatea erorilor de actualizare.

În bazele de date relaționale simplificările se fac prin definițiile index. Modelul de date pune importanță pe securitatea, concretizarea și integritatea datelor. Un avantaj major îl are și baza de date NoSQL, care a apărut și dezvoltat în urma nevoilor din ziua de azi, și oferă cereri prin arhitecturi construite pe cloud și sisteme distribuite. Bazele de date non-relaționale urmărește datele care nu îndeplinește garanțiile ACID. Printre primele aplicații mari, ce au semnalat problema scalabilității, se regăsește și Facebook. Această rețea de socializare are un număr imens de utilizatori, și este folosit în orice colț al lumii. Prin introducerea bazelor de date NoSQL creșterea masivă a proceselor tehnice a adus un beneficiu imens.

Tipurile de interogre în NoSQL se bazează pe căutarea unor indici primari sau a unui câmp ID și pe lipsa unei interogări pe alte domenii. Ca un exemplu tipic ar fi baza de date MongoDB și CouchDB, acestea permit interogări mai extinse, cum ar fi cele static cu creștere limitată pe nodurile bazei de date.

Bazele de date NoSQL simplifică dezvoltarea aplicațiilor utilizând modele de documente, coloane și valori importante, oferind un raspuns rapid, concret, de doar câteva milisecunde la fel se utilizează pentru a prelua informațiile primare de sortare în seturi de date. Oferind o flexibilitate înaltă, rigidă și prestabilită, astfel încât utilizatorii să se poată concentra pe crearea de sisteme pentru deservirea clienților.

3. Dezavantajele utilizarii bazelor de date

Blocurile de date sunt destul de utile odată ce sunt inițiate și activate, dar ajungerea la acest punct poate fi dificilă atât pentru persoane fizice, cât și pentru întreprinderi care intenționează să le folosească. Costul acestor sisteme este una dintre principalele slăbiciuni ale sistemelor de baze de date. Este posibil ca utilizatorii să fie nevoiți să-și actualizeze hardware-ul existent sau să achiziționeze echipamente noi care să susțină adăugarea unei baze de date de date, deoarece acestea pot ocupa o cantitate rezonabilă de spațiu și memorie. Această sarcină poate fi destul de costisitoare atunci când este combinată cu instrumentele necesare pentru operarea și implementarea sistemelor de baze de date [4].

Un alt aspect care ar trebui luat în considerare atunci când cumpărați o bază de date este că se sincronizează cu majoritatea, dacă nu cu toate, aplicațiile pe care le folosește o companie. Acest scenariu este avantajos deoarece simplifică toate aplicațiile și creează un mediu de lucru mai productiv și mai eficient. Toată lumea are acces la informații și date și poate chiar să folosească baze de date pentru a colabora mai eficient. Cu toate acestea, o problemă cu sistemul bazei de date ar putea afecta alte aplicații, ceea ce ar putea reduce productivitatea muncii. Un alt dezavantaj al sistemelor de baze de date este că procesul de mutare a fișierelor în bazele de date poate fi consumator de timp și bani.

În plus, pot apărea probleme de compatibilitate dacă sistemele de bază de date învechite nu se traduc corect în noul sistem de baze de date. În paragrafele care urmează, un alt dezavantaj al utilizării unei baze de date este că poate dura ceva timp pentru a instrui oamenii în aplicația sa, ceea ce ar putea reduce productivitatea muncii.

4. Tipuri de baze de date

Dacă tot am înaintat cu câteva noțiuni generale despre baza de date, n-ar strica să ofer și câteva exemple. Desigur că există mult mai multe tipuri de date, fiecare fiind mai mult sau mai puțin importantă pentru una sau altă afacere. Cea mai bună bază de date pentru o anumită întrprindere depinde de modul în care aceasta intenționează să o utilizeze [5].

- Baze de date relaționale – apărute la început și au avut o mare importanță în 1980. Elementele din bazele de date relaționale sunt aranjate ca un set de tabele, ce are coloane și rânduri. Tehnologia acestora presupune cel mai bun și flexibil mod de a accesa informațiile structurate.
- Baze de date orientate pe obiecte - informațiile dintr-o bază de date orientată pe obiecte sunt aranjate sub formă de obiecte, așa cum este și în programarea orientată pe obiecte.
- Baze de date distribuite - este creată din două sau mai multe fișiere aflate în site-uri diferite. Baza de date poate fi stocată pe diverse dispozitive aflate în aceeași locație fizică.
- Depozite de date - este creat pentru interogări și analize accelerate.
- Baze de date NoSQL – numită și nerelațională, permite stocarea și gestionarea datelor nestructurate sau semi-structurate, acestea au fost populare ca aplicații web.
- Baze de date grafice – sunt folosite pentrua depozita datele sub forma unor entități și a relațiilor dintre acestea.

5. Software pentru baza de date

Software-ul are un rol important pentru baze de date și este folosit pentru a stoca, modifica și întreține fișiere și înscrieri ale bazei de date, ca urmare acestea duc la crearea mai simplă a fișierelor și înregistrărilor precum și înglobarea, editarea, actualizarea și raportarea datelor. Software-ul gestionează, de asemenea, depozitarea, backupul și raportarea datelor, precum și controlul multi-acces și securitatea. În prezent sunt destul de multe sustrageri și fraude de date, care duc la urmări seriose și problematice, de aceea siguranța bazelor de date este foarte importantă și pusă pe prim plan [6].

Software-ul pentru baze de date simplifică gestionarea datelor și permite utilizatorilor să stocheze datele într-o formă structurată și apoi să le acceseze. Prin intermediul interfeței grafice acesta crează și gestionează datele, oferind posibilitatea der a crea propriile baze de date.

6. Concluzii

În concluzie, bazele de date au un rol extrem de important în managementul informațiilor utilizate în diverse firme, fiind preferate pentru multiple avantaje pe care le oferă și pentru faptul că poate fi rulate pe aproape toate dispozitivele software din toată lumea.

Principalul avantaj al datelor moderne este că au transformat toate sferele vieții și muncii, oferind companiilor acces la mai multe informații decât oricând. Folosind baze de date și alte instrumente de afaceri computerizate sau inteligente, organizațiile pot acum să utilizeze informații detaliate pentru a opera mai eficient, pentru a lua decizii mai bune și pentru a deveni mai utilizabile și mai scalabile. Îmbunătățirea accesării informațiilor și a datelor despre și într-o companie este foarte importantă în prezent, deoarece volumul de date urmărit este din ce în ce mai mare. Este favorabil, așadar, o platformă care să poată oferi performanța, scalarea și flexibilitatea de care au necesitate companiile, pe măsură ce se tehnica se dezvoltă și apar multiple furncționalități.

Baze de date moderne și contemporane sunt gata să activeze în mod semnificativ aceste funcționalități prin automatizarea sarcinilor manuale consumatoare de timp și de muncă intensă. Acest lucru eliberează orele de lucru ale utilizatorilor și le permite să se angajeze mai mult cu datele lor. Utilizatorii obțin verificare și independență, menținând în același timp standarde cruciale de securitate, având control direct asupra potențialului de a crea și aplica baze de date.

Referinţe

1. SEEGER, M. Computer Science and Media Ultra-Large-Sites SS09 Stuttgart, Germany, 21 Septembrie 2009.
2. The ultimate reference for NOSQL Databases. [online] [Accesat: 23.11.2022]. Disponibil: http://nosql-database.org/
3. WU, Suzanne: How Much Information IS There in the World. University of Southern California, 10 Februarie 2011.
4. CATTELL, R.: Scalable SQL and NoSQL Data Stores, SIGMON Record, Decembrie 2010, Vol 39, Nr 4.
5. Aslett, M.:" MySQL vs. NoSQL and NewSQL - survey results", 22 May 2012, 451 Research Group.
6. SHALOM, N.: The Common Principles Behind The NoSQL Alternatives, December 2009,- Blog post of 2009-12-15. [online] [Accesat: 23.11.2022]. Disponibil: http://natishalom.typepad.com/nati_shaloms_blog/2009/12/the-common principlesbehind-the-nosql-alternatives.html.

СУБД MS ACCESS

Адриана БУШУЕВ

Департамент Программной Инженерии и Автоматики, группа TI-192 F/R, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Адриана БУШУЕВ, e-mail busuev.adriana@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Статья посвящается СУБД MS Access, в каких областях применяется и что из себя представляет. Дана краткая характеристика MS Access, преимущества, недостатки и функциональные возможности.*

Cuvinte cheie*: СУБД, MS Access, VBA.*

1. Введение

Для принятия обоснованных и эффективных решений в производственной деятельности, в управлении экономикой и в политике современный специалист должен уметь с помощью компьютеров и средств связи получать, накапливать, хранить и обрабатывать данные, представляя результат в виде наглядных документов. В современном обществе информационные технологии развиваются очень стремительно, они проникают во все сферы человеческой деятельности.

В разных областях экономики зачастую приходится работать с данными из разных источников, каждый из которых связан с определенным видом деятельности. Для координации всех этих данных необходимы определенные знания и организационные навыки [1].

И так, продукт корпорации Microsoft Access объединяет сведения из разных источников в одной реляционной базе данных. Создаваемые в нем формы, запросы и отчеты позволяют быстро и эффективно обновлять данные, получать ответы на вопросы, осуществлять поиск нужных данных, анализировать данные, печатать отчеты, диаграммы и почтовые наклейки.

2. Что из себя представляет MS Access?

Система MS Access — это набор инструментов конечного пользователя для управления базами данных. В ее состав входят конструкторы таблиц, форм, запросов и отчетов. Эту систему можно рассматривать и как среду разработки приложений. Используя макросы или модули для автоматизации решения задач, можно создавать ориентированные на пользователя приложения такими же мощными, как и приложения, написанные непосредственно на языках программирования. При этом они будут включать кнопки, меню и диалоговые окна. Программируя на языке VBA(Visual Basic for Applications), можно создавать такие мощные программы, как сама система Access [2].

В Access в полной мере реализовано управление реляционными базами данных. Система поддерживает первичные и внешние ключи и обеспечивает целостность данных на уровне ядра (что предотвращает несовместимые операции обновления или удаления данных). Кроме того, таблицы в Access снабжены средствами проверки допустимости данных, предотвращающими некорректный ввод вне зависимости от того, как он осуществляется, а каждое поле таблицы имеет свой формат и стандартные описания, что существенно облегчает ввод данных. Access поддерживает все необходимые типы полей, в том числе текстовый, числовой, счетчик, денежный, дата/время, MEMO, логический, гиперссылка и поля объектов OLE. Если в процессе специальной обработки в полях не оказывается никаких значений, система обеспечивает полную поддержку пустых значений.

3. Краткая характеристика MS Access

Microsoft Access является настольной СУБД (система управления базами данных) реляционного типа. Достоинством Access является то, что она имеет очень простой графический интерфейс, который позволяет не только создавать собственную базу данных, но и разрабатывать приложения, используя встроенные средства. В отличие от других настольных СУБД, Access хранит все данные в одном файле, хотя и распределяет их по разным таблицам, как и положено реляционной СУБД. К этим данным относится не только информация в таблицах, но и другие объекты базы данных, которые будут описаны ниже. Для выполнения почти всех основных операций Access предлагает большое количество Мастеров(Wizards), которые делают основную работу за пользователя при работе с данными и разработке приложений, помогают избежать рутинных действий и облегчают работу неискушенному в программировании пользователю. Области применения Microsoft Access можно выделить следующие структуры:[3]

- применение в малом и среднем бизнесе (бухгалтерский учет, ввод заказов, ведение информации о клиентах, ведение информации о деловых контактах, кадрах и т.п.);
- при разработке программ и хранилищ данных на заказ (разработка внутриотраслевых приложений, разработка межотраслевых приложений, автоматизация некоторых функций предприятий);
- в крупных корпорациях (приложения для рабочих групп, системы обработки информации, документооборот);
- в качестве персональной СУБД (справочник по адресам, ведение инвестиционного портфеля, поваренная книга, каталоги книг, пластинок, видеофильмов и т. п.);
- в качестве средства хранения данных, которое используется в других приложениях.

4. Преимущества и недостатки

Создание многопользовательской БД Access и получение одновременного доступа нескольких пользователей к общей базе данных возможно в локальной одно ранговой сети или в сети с файловым сервером. Так как Access не является клиент серверной СУБД, возможности его по обеспечению многопользовательской работы несколько ограничены. В плане поддержки целостности данных Access отвечает только моделям БД небольшой и средней сложности. В нем отсутствуют такие средства как триггеры и хранимые процедуры, что заставляет разработчиков возлагать поддержание бизнес логики БД на клиентскую программу [4].

В отношении защиты информации и разграничения доступа Access не имеет надежных стандартных средств. В стандартные способы защиты входит защита с использованием пароля БД и защита с использованием пароля пользователя. Снятие такой защиты не представляет сложности для специалиста.

Однако, при известных недостатках MS Access обладает большим количеством преимуществ по сравнению с системами подобного класса. В первую очередь можно отметить распространенность, которая обусловлена тем, что Access является продуктом компании Microsoft, программное обеспечение и операционные системы которой использует большая часть пользователей персональных компьютеров. MS Access полностью совместим с операционной системой Windows, постоянно обновляется производителем, поддерживает множество языков. Также необходимо отметить ориентированность на пользователя с разной профессиональной подготовкой, что выражается в наличии большого количества вспомогательных, развитую систему справки и понятный интерфейс. Эти средства облегчают проектирование, создание БД и выборку данных из нее [4].

MS Access предоставляет в распоряжение непрограммирующему пользователю разнообразные диалоговые средства, которые позволяют ему создавать приложения не прибегая к разработке запросов на языке SQL или к программированию макросов или модулей на языке VBA. Access обладает широкими возможностями по импорту/экспорту данных в различные форматы, от таблиц Excel и текстовых файлов, до практически любой серверной СУБД через механизм ODBC(Open Database Connectivity).

5. Функциональные возможности MS Access

Ввод данных. Ввод данных может осуществляться следующими способами:

- вручную прямо в таблицу;
- вручную в поля формы;
- прямой импорт данных из других источников (базы Access, текстовые файлы, формат DBF, электронные таблицы, источники данных ODBC);
- программным методом, который может сочетать в себе любые средства, которые возможно реализовать на VBA.

Изменение данных. Редактирование возможно следующими способами:

- вручную прямо в таблице;
- в полях форм;
- в окне браузера, в котором загружена web-страница из БД;
- программным методом.

Вывод данных. Здесь Access предоставляет такие возможности:

- вывод на экран монитора в табличном виде, полях форм или отчетов;
- экспорт в другие форматы данных (те же, что при импорте);
- вывод на печать, в основном в виде отчетов;
- вывод данных в интернет-браузер с помощью объекта Страница в пределах определенной сети;
- программный экспорт и вывод информации.

Взаимодействие с другими источниками и потребителями информации. В этом плане Access может выступать как сервер или клиент автоматизации.

Использование базы данных MS Access другими приложениями. Такой вариант использует файл MDB как хранилище данных.

6. Заключение

Так как интеграция информационных систем во все сферы жизни увеличивается с каждым днем, то актуально становится разработка подобных баз данных. При этом разработчик должен учитывать то, что наиболее простые БД могут быть подвержены избыточности, но при этом нельзя и увлекаться делением БД на много составных таблиц. Также современные средства дружественного интерфейса позволяют разработать интуитивно понятные приложения, что является одним из основных требований заказчика. При создании БД необходимо принять во внимание область, для которой разрабатывается база данных. Например, при формировании БД для магазина, разработчик должен ориентировать её в первую очередь на продажу.

Библиография

1. Нейбауэр, А.; MS Access 2013 для занятых – Санкт Петербург: Питер, 2017. – 368 с.: ил.
2. Кронан, Дж.., Андерсен, В., Андерсон, Б. Б.; Microsoft Office Access 2013. Быстрые шаги – Москва : НТ Пресс, 2015. – 226 с.
3. Шевченко, Н. А.; Access 2013. Искусство создания базы данных – Москва : НТ Пресс, 2015 – 160 с.
4. Дженнингс, Р.; Использование Microsoft Office Access 2014 – Москва : Издательский дом «Вильямс», 2014. - 1312 с.

APACHE CASSANDRA

Яна ТОПАЛ

Департамент Программной Инженерии и Автоматики, группа TI-192 F/R, Факультет Вычислительной Техники, Информатики и Микроэлектроники, Технический Университет Молдовы, Кишинев, Республика Молдова

Автор корреспондент: Яна ТОПАЛ, e-mail: iana.topal@isa.utm.md

Научный руководитель: Дориан САРАНЧУК, DISA, FCIM, UTM

Аннотация: *Данная работа посвящена базе данных под названием Apache Cassandra. Рассматривается ее преймущества и недостатки, краткая характеристика данной базы данных,ее архитектура.*

Cuvinte cheie*: СУБД, Apache Cassandra, NoSQL.*

Введение

Apache Cassandra - это распределенная система управления базами данных (DBMS), которая разработана для обработки больших объемов данных с высокой доступностью и масштабируемостью без единой точки отказа.

Cassandra была создана в Facebook в 2008 году и в 2009 году она была открыта как проект с открытым исходным кодом в Apache Software Foundation. Она написана на языке Java и использует модель данных "ключ-значение" с широкими колонками (wide column).

Cassandra спроектирована для обработки больших объемов данных с высокой производительностью и масштабируемостью, позволяя добавлять новые узлы в кластер для обработки дополнительных запросов и хранения большего объема данных без перезапуска системы. Cassandra также обеспечивает высокую доступность, реплицируя данные на несколько узлов, что обеспечивает отказоустойчивость и предотвращает потерю данных в случае сбоев в системе [1].

Cassandra используется в широком спектре приложений, включая системы управления контентом, аналитику, мониторинг, веб-аналитику, социальные сети и многое другое.

Одним из ключевых преимуществ Cassandra является ее способность обеспечивать быстрый доступ к данным при записи и чтении. Она достигает этого, используя распределенную архитектуру и оптимизированный алгоритм распределения данных по узлам кластера.

Cassandra Query Language

Cassandra также обладает гибкими возможностями конфигурирования, позволяющими настраивать систему для определенных требований производительности и доступности данных. Кроме того, Cassandra имеет поддержку языков запросов, в том числе CQL (Cassandra Query Language), который предоставляет SQL-подобный интерфейс для запросов к данным [2].

Среди других преимуществ Cassandra следует отметить:

- Горизонтальное масштабирование без прерывания работы системы
- Высокая доступность благодаря распределенной архитектуре и репликации данных
- Поддержка транзакций и консистентности данных в распределенной среде
- Поддержка множества языков программирования для разработки приложений, включая Java, Python, Ruby, C++, и другие.

Apache Cassandra является распределенной системой управления базами данных, которая работает на множестве узлов, обеспечивая высокую доступность и отказоустойчивость. Работа Cassandra основана на распределенной архитектуре, где данные хранятся на множестве узлов в кластере. Вот как работает Apache Cassandra: Кластер Cassandra состоит из нескольких узлов, которые работают в совершенно равноправном режиме. Каждый узел может быть добавлен в кластер, чтобы распределить данные и обеспечить отказоустойчивость. Cassandra использует алгоритм консенсуса, называемый gossip protocol, для обмена информацией между узлами кластера. Это позволяет узлам кластера обмениваться информацией о топологии кластера, состоянии узлов и другой полезной информации. Cassandra использует механизмы репликации данных, чтобы обеспечить высокую доступность данных в случае сбоев в работе узлов. Данные реплицируются на нескольких узлах, и если один узел выходит из строя, данные могут быть восстановлены из других узлов. Cassandra использует алгоритм маршрутизации запросов, чтобы обеспечить быстрый доступ к данным в кластере. Когда запрос приходит к кластеру, он маршрутизируется к соответствующему узлу, который может обработать запрос. Cassandra поддерживает множество механизмов для обработки запросов, включая механизмы чтения и записи, агрегации данных и поиска по индексам. Cassandra также поддерживает транзакционную целостность данных, чтобы обеспечить согласованность данных в кластере. Cassandra имеет механизмы управления данными, включая механизмы для добавления, удаления и обновления данных в кластере. Cassandra также предоставляет механизмы управления метаданными, которые описывают структуру данных в кластере.

Cassandra является одной из наиболее популярных распределенных систем управления базами данных в мире и используется в таких компаниях, как Netflix, Twitter, Apple и другие, для обработки и хранения больших объемов данных в реальном времени [3].

Apache Cassandra обладает рядом преимуществ, которые делают ее популярной среди разработчиков и инженеров, работающих с большими объемами данных. Вот некоторые из главных преимуществ Apache Cassandra:

1. Горизонтальное масштабирование без прерывания работы системы. Cassandra позволяет добавлять новые узлы в кластер без прерывания работы системы, что позволяет легко масштабировать систему с увеличением объема данных и количества запросов.
2. Высокая доступность благодаря распределенной архитектуре и репликации данных. Cassandra использует механизм репликации данных, который позволяет хранить несколько копий данных на разных узлах кластера, обеспечивая высокую доступность и отказоустойчивость.
3. Поддержка транзакций и консистентности данных в распределенной среде. Cassandra обеспечивает транзакционную целостность данных в распределенной среде, используя механизмы консенсуса, такие как Paxos или Raft.
4. Поддержка множества языков программирования. Cassandra имеет API для множества языков программирования, таких как Java, Python, Ruby, C++, и другие. Это делает Cassandra очень гибкой для разработки приложений на разных языках.
5. Высокая производительность при записи и чтении данных. Cassandra предлагает оптимизированный алгоритм распределения данных по узлам кластера, который обеспечивает быстрый доступ к данным при записи и чтении.
6. Гибкость в настройке системы. Cassandra имеет гибкие возможности конфигурирования, которые позволяют настраивать систему для определенных требований производительности и доступности данных.

7. Поддержка языка запросов CQL (Cassandra Query Language). CQL предоставляет SQL-подобный интерфейс для запросов к данным в Cassandra, что делает разработку и выполнение запросов более удобным для разработчиков.

В целом, Apache Cassandra является мощной распределенной системой управления базами данных, которая обладает высокой доступностью, масштабируемостью и гибкостью настройки, что делает ее популярным выбором для различных приложений и инфраструктур.

Несмотря на то, что Apache Cassandra имеет множество преимуществ, у нее также есть некоторые недостатки, которые следует учитывать при выборе этой системы для решения конкретных задач. Вот некоторые из недостатков Apache Cassandra:

1. Сложность в управлении и настройке системы. Cassandra является сложной системой, которая требует определенных знаний и опыта в управлении и настройке. Управление кластером Cassandra может быть сложным и трудоемким процессом.
2. Ограниченная поддержка для сложных запросов. Cassandra предоставляет механизмы для выполнения базовых запросов, но поддержка для сложных запросов может быть ограниченной.
3. Ограниченная поддержка для транзакций. Cassandra поддерживает транзакционную целостность данных, но в некоторых случаях могут возникать ограничения при выполнении транзакций.
4. Высокий порог входа для начинающих разработчиков. Cassandra может быть сложной для понимания и использования для начинающих разработчиков, которые не имеют опыта работы с распределенными системами управления базами данных.
5. Неудобство в использовании для небольших проектов. Cassandra может быть избыточной и сложной для использования в небольших проектах, где объем данных не очень велик.
6. Высокая стоимость масштабирования на уровне узлов. Добавление новых узлов в кластер может быть дорогостоящим процессом из-за необходимости обеспечения их отказоустойчивости и высокой доступности.

В целом, Apache Cassandra является мощной системой управления распределенными базами данных с некоторыми ограничениями и сложностями в использовании. Однако, при правильной настройке и управлении, Cassandra может быть эффективным инструментом для работы с большими объемами данных в распределенной среде.

Вывод о Apache Cassandra:

Apache Cassandra - это мощная распределенная система управления базами данных, которая обеспечивает высокую доступность, отказоустойчивость и масштабируемость. Cassandra использует сложные механизмы для обработки запросов и управления данными, что позволяет эффективно работать с большими объемами данных в кластере. Cassandra также имеет множество преимуществ, включая гибкую модель данных, высокую производительность и открытый исходный код [4].

Однако, Apache Cassandra также имеет некоторые недостатки, включая сложность настройки и управления кластером, а также ограничения в использовании транзакционных операций. Также может быть сложно понимать и использовать модель данных Cassandra для разработчиков, не знакомых с NoSQL технологиями.

В целом, Apache Cassandra является одним из лучших выборов для работы с масштабируемыми и отказоустойчивыми базами данных в больших и сложных приложениях. Однако, использование Cassandra требует глубокого понимания ее особенностей и методов работы с данными в распределенной среде.

Библиография

1. Wikipedia [online] [дата обращения 02.03.2023], Доступно: https://ru.wikipedia.org/wiki/Apache_Cassandra
2. Автор статьи под ник-неймом Pleha. 17.10.1012г. Название статьи: «Как устроена apache Cassandra» [online] [дата обращения 02.03.2023], Доступно: https://habr.com/ru/post/155115/
3. Автор статьи Анна Вичугова 15.12 Название статьи «ТОП-10 достоинств и 5 главных недостатков Apache Cassandra» [online] [дата обращения 02.03.2023], Доступно: https://www.bigdataschool.ru/blog/cassandra-key-features.html
4. Общее понятие об информационных технологиях в юридической деятельности [online] [дата обращения 02.03.2023], Доступно: http://eor.dgu.ru/lectures_f/ИТ в ЮД курс лекций/Лекция 1.htm