МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ БОГДАНА ХМЕЛЬНИЦЬКОГО
ЧЕРКАСЬКА ОБЛАСНА ДЕРЖАВНА АДМІНІСТРАЦІЯ
НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕЧЕНКА
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ
ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІЛЛІ ІЛЛІЧА МЕЧНІКОВА
WROCŁAW UNIVERSITY OF SCIENCE AND TECHNOLOGY
OPOLE UNIVERSITY

ІНФОРМАЦІЙНІ МОДЕЛЮЮЧІ ТЕХНОЛОГІЇ, СИСТЕМИ ТА КОМПЛЕКСИ (ІМТСК-2021)

ТРЕТЯ МІЖНАРОДНА
НАУКОВО-ПРАКТИЧНА КОНФЕРЕНЦІЯ

ЗБІРКА ТЕЗ

27-28 травня, 2021
Черкаси, Україна

MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
THE BOHDAN KHMELNYTSKY NATIONAL UNIVERSITY OF CHERKASY
CHERKASY REGIONAL STATE ADMINISTRATION
THE NATIONAL ACADEMY OF SCIENCES OF UKRAINE
NATIONAL AVIATION UNIVERSITY
NATIONAL TECHNICAL UNIVERSITY OF UKRAINE
"IGOR SIKORSKY KYIV POLYTECHNIC INSTITUTE"
TARAS SHEVCHENKO NATIONAL UNIVERSITY OF KYIV
ODESSA STATE ENVIRONMENTAL UNIVERSITY
ODESSA ILYA ILYICH MECHNIKOV NATIONAL UNIVERSITY
WROCŁAW UNIVERSITY OF SCIENCE AND TECHNOLOGY
OPOLE UNIVERSITY

INFORMATION MODELING TECHNOLOGIES, SYSTEMS AND APPLICATIONS (IMTSA-2021)

THIRD INTERNATIONAL
SCIENTIFICALLY-PRACTICAL CONFERENCE

COLLECTED ARTICLES

May 27-28, 2021
Cherkasy, Ukraine

УДК 004

Інформаційні моделюючі технології, системи та комплекси (ІМТСК-2021). Третя міжнародна науково-практична конференція, 27-28 травня 2020 р., Черкаси, Україна. – Черкаси: Черкаський національний університет імені Богдана Хмельницького, 2021. – 162 с. (збірка тез)

В матеріалах конференції відображені результати теоретичних та експериментальних досліджень, які пов'язані з перспективними напрямками розвитку інформаційних технологій, технологій моделювання інформаційних та інтелектуальних систем і комплексів.

Матеріали конференції подаються в авторській редакції мовою оригіналу.

Програмний комітет:

Осауленко І.А.	д.т.н., доцент (Україна, Черкаси)
Тесля Ю.М.	д.т.н., професор (Україна, Черкаси)
Квасніков В.П.	д.т.н., професор (Україна, Київ)
Кулаков Ю.О.	д.т.н., професор (Україна, Київ)
Теленик С.Ф.	д.т.н., професор (Україна, Київ)
Чмир І.О.	д.т.н., професор (Україна, Одеса)
Гунченко Ю.О.	д.т.н., професор (Україна, Одеса)
Казимир В.В.	д.т.н., професор (Україна, Чернігів)
Дорош М.С.	д.т.н., професор (Україна, Чернігів)
Власенко В.О.	д.т.н., професор (Польща, Ополе)
Хлєбус Е.	д.т.н., професор (Польща, Вроцлав)
Кулаковська А.	д.т.н., доцент (Польща, Ченстохова)
Цибровський С.В.	віце-президент eKreative (Україна, Черкаси)

Організаційний комітет:

Осауленко І.А.	д.т.н., доцент
Онищенко Б.О.	к.ф.-м.н., доцент
Жирякова І.А.	к.т.н., доцент
Супруненко О.О.	к.т.н., доцент
Веретельник В.В.	к.т.н.
Любченко К.М.	

UDC 004

Information modeling technologies, systems and applications (IMTSA-2021). Third international scientifically-practical conference, May 27-28, 2021, Cherkasy, Ukraine. – Cherkasy: The Bohdan Khmelnytsky national university of Cherkasy, 2021. – 162 p. (collected articles)

The proceedings of the conference containing results of scientific and practical research in relation to promising directions of information technologies, odeling technologies and applications.

The articles are the original work of the author, and submitted by original author's language and editorial.

The programme committee:

Igor Osaulenko	Dr. Tech. Sc., Assoc. Prof. (Ukraine, Cherkasy)
Yurii Teslia	Dr. Tech. Sc., Prof. (Ukraine, Cherkasy)
Volodymyr Kvasnikov	Dr. Tech. Sc., Prof. (Ukraine, Kyiv)
Yurii Kulakov	Dr. Tech. Sc., Prof. (Ukraine, Kyiv)
Serhii Telenyk	Dr. Tech. Sc., Prof. (Ukraine, Kyiv)
Igor Chmyr	Dr. Tech. Sc., Prof. (Ukraine, Odessa)
Yurii Hunchenko	Dr. Tech. Sc., Prof. (Ukraine, Odessa)
Volodymyr Kazymyr	Dr. Tech. Sc., Prof. (Ukraine, Chernihiv)
Mariia Dorosh	Dr. Tech. Sc., Prof. (Ukraine, Chernihiv)
Viktor Vlasenko	Dr. Tech. Sc., Prof. (Poland, Opole)
Edward Chlebus	Dr. Tech. Sc., Prof. (Poland, Wrocław)
Anna Kułakowska	Dr. Tech. Sc., Assoc. Prof. (Poland, Częstochowa)
Serhii Tsybrovskyi	vice president eKreative (Ukraine, Cherkasy)

The organising committee:

Igor Osaulenko	Dr. Tech. Sc., Assoc. Prof.
Borys Onyshchenko	Ph.D. Phys.-Math. Sc., Assoc. Prof.
Iryna Zhyriakova	Ph.D. Tech. Sc., Assoc. Prof.
Oksana Suprunenko	Ph.D. Tech. Sc., Assoc. Prof.
Vitalii Veretelnyk	Ph.D. Tech. Sc.
Kostiantyn Liubchenko	

ЗМІСТ

Секція 1
Моделюючі системи і технології

УДК 621.365.23

ДОСЛІДЖЕННЯ РОЗПОДІЛУ ЕНЕРГІЇ В ЕЛЕКТРОДІ ТА ОБ'ЄМІ РОБОЧОГО ПРОСТОРУ ДІЮЧОЇ КРУГЛОЇ ФЕРОСИЛІКОМАРГАНЦЕВОЇ ЕЛЕКТРОПЕЧІ РКЗ-16,5 ЗА ДОПОМОГОЮ УЗАГАЛЬНЕНОЇ МАТЕМАТИЧНОЇ МОДЕЛІ ПО МЕТОДУ ІНТЕГРАЛЬНИХ РІВНЯНЬ

Нежурін В.І., Куваєв В.Ю., Ярошенко Я.Г.

Національна металургійна Академія України

Рішення завдання підтримки оптимального електродного, шихтового і електричного режимів конкретного технологічного процесу отримання сплаву в рудовідновлювальних електропечах (РВП) забезпечується вибором оптимальних геометричних параметрів ванни печі; пічного контуру і підтриманням раціонального електричного режиму плавки, що створює необхідний, з точки зору термодинаміки, розподіл введеної енергії в робочому просторі ванни печі. Формування самообпалювальних електродів і металургійні процеси в печі прямо залежать від картини розподілу струму і потужності в них.

Авторами вирішувалося завдання розробки математичної моделі розподілу густини струму в перерізі самообпалювального електрода, питомої активної потужності в об'ємі робочого простору ванни РВП по методу вторинних джерел в формі інтегральних рівнянь Фредгольма II роду з залученням експериментальних даних діючих печей [1, с.38]. Структура реакційної зони робочого простору круглої трьохелектродної РВП (наприклад, печі РКЗ-16,5) симетрична осі кожного електрода, тому, виходячи з умов осьової симетрії, розглядався меридіальний перетин електрода і реакційної зони з підведенням струму через надпровідний контакт електрода, що знаходиться в неоднорідному середовищі, і відведенням через надпровідну ванну сплаву (рис. 1) [3, с.34]. Можливість створення раціональної картини електромагнітного поля рудовідновлювальної електропечі на етапі проектування є актуальним завданням, яка вирішує дві проблеми: оптимізацію металургійного процесу і економію електроенергії. Поверхневий ефект і ефект близькості при промисловій частоті проявляється не дуже сильно, поле можна розглядати як квазістаціонарне; нехтуючи поверхневим ефектом та ефектом близькості поле печі розглядається як поле постійного струму. Речовини, які проводять струм, неоднорідні в об'ємі ванни по провідності. Однак орієнтуючись по температурному полю, можна виділити в об'ємі ванни неоднорідні зони, в яких провідність вважаємо постійною, тобто з достатнім ступенем точності розглядати електромагнітне поле рудовідновлювальної електропечі як електричне поле постійного струму в шматочно-однорідному провідному середовищі.

Алгоритм розрахунку поля густини струму, питомої активної потужності наступний:

1. Розраховують розподіл вторинних джерел на поверхні розділу середовищ з різною провідністю $\gamma 1 \div \gamma 6$.

2. За розподілом вторинних джерел розраховується напруженість поля в обраних точках перетину електрода і робочого простору печі.

3. Розраховується густина струму (ji) і питома активна потужність (pi) в обраних точках робочого простору ванни печі $ji = \gamma i \cdot Ei$ та $pi = \gamma i \cdot Ei2$.

При розрахунку вважали, що магнітна проникність вмісту ванни відповідає магнітній проникності виділених зон, а геометрія ванни і електродів відповідає в масштабі параметрам ванни діючої печі.

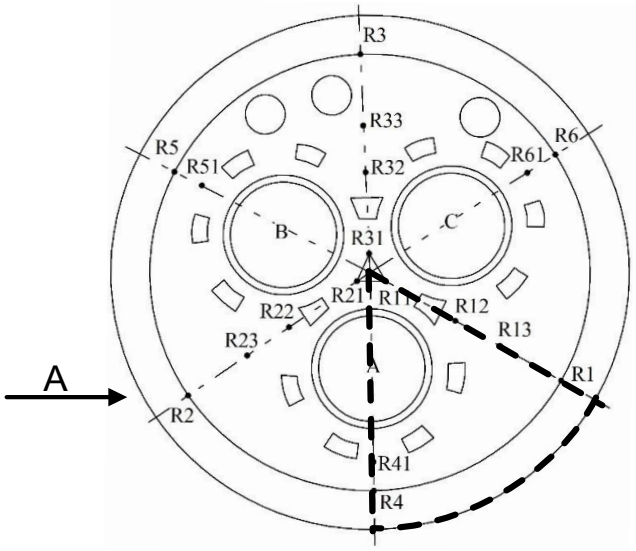


Рис. 1. Розрахункова частина реакційного простору діючої електропечі.

Модель розроблена для напівсферичної форми торця електрода і глибини його занурення в ванну печі 1 м. У роботі представлені епюри розподілу щільності струму по перетину електрода і лінії рівної питомої активної потужності в об’ємі робочого простору в припущенні, що потужність зосереджена в центрі обраних елементарних об’ємів фрагмента ванни. На рис. 2 представлено розподіл вищевказаних величин в симетричному осі електрода об’ємі ванни електропечі і в перетині електрода. На точність отриманих результатів істотно впливає кількість обраних розрахункових точок в виділених на основі експериментальних досліджень діючої печі зон неоднорідності, характерних для конкретного типу сплаву. Аналіз показує, що значення питомих активних потужностей в виділених розрахункових точках і експериментально отриманих в доступних для вимірювань точках робочого простору діючої печі, наприклад, при виплавці феросилікомарганцю, збігаються з інженерної точністю.

Практичне використання запропонованої математичної моделі при розрахунку параметрів рудовідновлювальної печі може виглядати наступним чином:

1. Визначення основних електричних і геометричних параметрів печі за допомогою відомих інженерних методів розрахунку

2. Розробка і розрахунок математичної моделі проектованої печі для виявлення впливу геометрії ванни на розподіл потужності в об'ємі ванни печі і знаходження оптимального значення діаметра розпаду електродів і глибини ванни, вважаючи критерієм оптимальності максимальне значення виділеної потужності в міжелектродному і піделектродному просторі ванни печі.

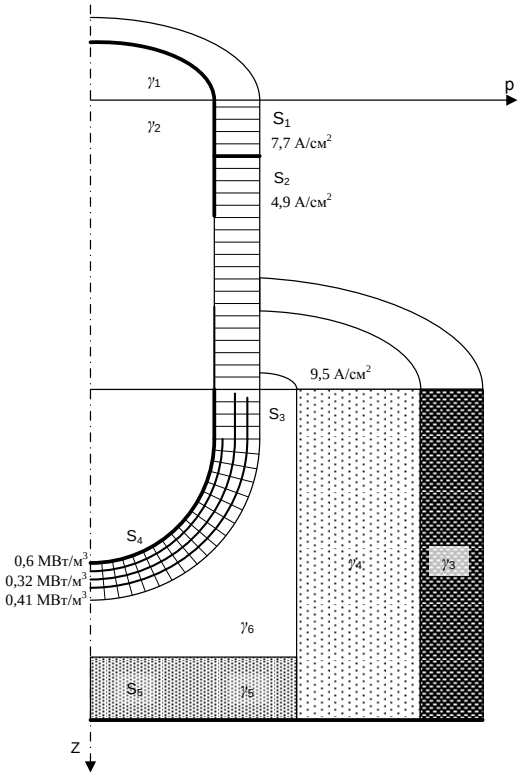


Рис. 2. Результати математичного моделювання поля в об'ємі робочого простору ванни електропечі РКЗ-16,5.

3. Уточнення попередньо розрахованих геометричних параметрів рудовідновлювальної печі.

4. Запропонована узагальнена математична модель може бути використана при реконструкції діючих і проектуванні нових рудовідновальних електропечей.

Література:

1. Карманов Э.С., Нежурин В.И. Исследование скорости и характера схода шихтовых материалов при выплавке марганцевых сплавов в закрытых руднотермических электропечах [Текст] / Сталь, 1991. – № 7. – С. 37-40.

2. Тозони О.В. Расчет трехмерных электромагнитных полей. – Киев: Техника, 1974. – 322 с.

3. Ольдзиевский С.А., Кравченко В.А., Нежурин В.И., Борисенко И.А. Математическое моделирование электрических полей печей рудной электротермии [Текст]. – М.: Металлургия. – 1990. – 112 с.

УДК 004.9

ЗАБЕЗПЕЧЕННЯ ГРУПОВОГО УПРАВЛІННЯ ПРОЕКТАМИ ЗА ДОПОМОГОЮ СИСТЕМИ ФРАКТАЛЬНОГО АНАЛІЗУ

Жицький А.М., Свинчук О.В.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Управління проектом є невід'ємною складовою успішності бізнес процесів. Застосування різноманітних прийомів та інструментів дозволяє досягти поставлених проектних цілей і задовільнити потреби клієнтів. Важко уявити сучасну організацію, котра не використовує проектне планування [1] та менеджмент задач на проекті. Наразі існує багато програмних рішень для управління проектами на базі операційних систем Windows та Linux (Microsoft Project, TeamBridge), але користувач не може отримати доступ до більшості таких систем зі свого мобільного чи планшету. Більшість проектних менеджерів зосередженні на максимальній підтримці конкретного проекту і надають його учасникам широкий набір функцій для організації проектної роботи. У свою чергу менеджери проектів отримують певний перелік необхідних інструментів для організації розподілення завдань та делегування ролей серед учасників команди. Проте все частіше у зв'язку з розширенням ринку і значним зростанням кількості проектів які виконують лідируючі світові компанії, все частіше виникає потреба у груповому управлінні, що дозволить працювати з декількома проектами одночасно.

Розроблений програмний продукт враховує багаторівневі завдання проекту і надає одночасний доступ для багатьох учасників і керівників проекту на кожному рівні завдань. Кожен користувач системи може бути задіяний у декількох проектах одночасно. Менеджери проектів мають змогу керувати завданнями багатьох проектів, а також можуть бути учасниками інших проектів без загальних функцій управління.

Для проектного планування використовується теорія фракталів [2], що дозволяє більш широко прогнозувати успіх проекту на кожному етапі. Поняття фрактала було запропоноване французько-американським математиком Бенуа Мандельбротом. У 1977 році він опублікував роботу «Фрактальна геометрія природи», в якій стверджував, що випадкові, на перший погляд, форми є насправді складними геометричними фігурами, що складаються з менших фігур, які при збільшенні масштабу точно повторюють велику фігуру. Фрактали володіють цікавими властивостями:

- вони мають хаотичну поведінку;
- вони мають дробову нескінченну розмірність;
- вони складаються із частинок, подібних усій фігурі;
- вони можуть бути представлені простим алгоритмом.

Фрактал в найбільш загальному сенсі означає нерегулярну, самоподібну структуру, безліч, підмножини і елементи якої подібні самій безлічі, але в

іншому масштабі. Фрактали можуть бути детермінованими або стохастичними. Крім того фрактали можна класифікувати відповідно до їхньої самоподібності. Розрізняють три типи самоподібності у фракталах: точна самоподібність (виглядає однаково при різних збільшеннях); майже самоподібність (фрактал виглядає приблизно (але не точно) самоподібним при різних збільшеннях); статистична самоподібність (фрактал має чисельні або статистичні міри, що зберігаються при збільшенні). Прикладами фракталів є множина Кантора, фрактал Ляпунова, трикутник Серпінського, килим Серпінського, губка Менгера, сітка Аполлонія, крива дракона, крива заповнення простору, межі множин груп Кліні, крива Коха. Також в останній час увага приділяється функціям, які мають фрактальні властивості і складну «варіаційну поведінку». Це фрактальні функції: сингулярні функції, ніде не диференційовні функції, ніде не монотонні функції, функції канторівського типу. Такі функції дозволяють моделювати часові ряди та визначати момент, коли система стає нестабільною та готова перейти в новий стан.

Властивість фрактальної системи проявляється в тому, що цикл розвитку цілісної системи при зміні масштабу його розгляду залишається подібним до початкового. Сформована система враховує вхідні показники і за рахунок оцінки ефективності виконання задач надає найбільш вірогідні результати на межі того чи іншого етапу проекту як окремого фракталу. Таким чином досліджується кожен з таких фракталів та їх дробова розмірність [3], величина якої в даному випадку визначається: кількістю людських та часових ресурсів; особливостями та конкретними вимогами кожного етапу; кількістю, а також складністю завдань та цілей, що стоять перед учасниками проекту, та історією проходження проекту в цілому.

Такий підхід дозволяє не тільки підвищити продуктивність керівників проекту, а також створити комфортних умов задля забезпечення максимального управління часом [4]; більш ефективно проводити проектне планування, а також заощадити значну кількість різноманітних ресурсів, як людських, так і матеріальних. Добре побудована фрактальна модель дозволяє отримати достатньо точні та адекватні прогнози.

При моделюванні систем взагалі і, зокрема, для цілей структурного аналізу використовуються різні моделі, що відображають:

- функції, які система повинна виконувати;
- процеси, що забезпечують виконання зазначених функцій;
- дані, необхідні для виконання функцій, і відносини між цими даними;
- організаційні структури, що забезпечують виконання функцій;
- матеріальні та інформаційні потоки, що виникають в ході виконання функцій.

Модель являє собою сукупність об'єктів і відносин між ними, яка адекватно описує лише деякі властивості модельованої системи. Модель є лише одним з багатьох можливих тлумачень системи. Це тлумачення має влаштовувати користувача в даній ситуації, в даний момент часу.

Для моделі в загальному випадку характерні чотири властивості:

- зменшений масштаб (розмір) моделі, точніше, її складність, ступінь якої завжди менше, ніж у оригіналу;
- збереження ключових співвідношень між різними частинами;
- працездатність моделі, тобто можливість в принципі працювати;
- адекватність моделі дійсним властивостям оригіналу (ступінь достовірності).

Для розробки системи було використано мову програмування Java. Серверна частина виконана на мові програмування PHP. База даних створена за допомогою системи управління MySQL, а для полегшення роботи з нею було застосовано сервіс адміністрування phpmyAdmin. Програмний продукт було розроблено у вигляді мобільного додатку за допомогою універсального набору класів Activity і Fragment. Мобільний додаток з'єднується з сервером за допомогою широкого набору запитів, що дозволяє йому як отримувати інформацію з бази даних, так і вносити нові дані. Додаток працює на пристроях з операційною системою Android версією не нижче ніж Jelly Bean 4.3.

Основні функції системи:

- розподілення прав користувачів і рівнів доступу до проектів і завдань;
- доступ користувачів до особистої інформації в системі;
- надання даних про поточні проекти кожному користувачу;
- надання даних про поточні завдання кожного проекту відповідно;
- забезпечення вищого рівня доступу для менеджерів проектів;
- реалізація управління завданнями на кожному проекті;
- редагування статусів завдань,
- аналіз кожного з етапів та прогнозування наступного,
- визначення коефіцієнту успішності проекту в режимі реального часу.

У системі присутні процеси автентифікації та авторизації, паролі користувачів зберігаються у захищеному вигляді.

Таким чином, теорія фракталів демонструє якісно новий підхід у моделюванні управління проектами. Система переважно розрахована на IT компанії, які зацікавленні в управлінні проектними завданнями, а також мають необхідність розподілення великої кількості людських ресурсів на декількох проектах одночасно. Проте програмний продукт може бути також ефективним і для невеликих компаній, задля підвищення продуктивності праці і рівня зосередженості і концентрації працівників.

Література:

1. J. Davidson Frame. Managing Projects in Organization: how to make the best use of time, techniques, and people. 3rd ed, 2003. 251 c.

2. Морозов А.Д. Введення в теорію фракталів. Москва-Іжевськ: Інститут комп'ютерних досліджень, 2002. 160 с.
3. Мандельброт Б. Фрактальна геометрія природи. М.: Інститут комп'ютерних досліджень, 2002. 656 с.
4. Гонтарева И. В., Нижегородцев Р. М., Новиков Д. А. Управление проектами. М: Либроком, 2009. 384 с.

УДК 004.9

ФРАКТАЛЬНИЙ АНАЛІЗ КОСМІЧНИХ ЗНІМКІВ ДЛЯ МОНІТОРИНГУ ТА КЛАСИФІКАЦІЇ ЛІСОВИХ НАСАДЖЕНЬ

Богач А.Г., Свинчук О.В., Бандурка О.І.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Моніторинг навколишнього середовища є основним методом контролю екологічного стану країни. Розвиток ІТ-технологій дозволяє розвивати наші можливості для покращення екологічної ситуації не тільки в межах нашої країни, але у світі в цілому. Одним з таких інструментів є дистанційне зондування Землі (ДЗЗ) – це спостереження нашої планети та визначення властивостей об’єктів на земній поверхні, отриманих за допомогою знімальних пристроїв, які встановлені на повітряних літальних апаратах та штучних супутниках Землі. Отримані дані – це аерокосмічні знімки спостережуваної частини земної поверхні в цифровій формі у вигляді растрових зображень. Ці дані підлягають подальшому детальному аналізу, визначенню кількісних характеристик досліджуваного об’єкту, які необхідні для прогнозування розвитку явища чи процесу. Обробка та інтерпретація даних дистанційного зондування тісно пов’язані з цифровою обробкою зображень.

Моніторинг лісів на великих територіях вимагає багато часу та великих людських витрат. Для усунення цих проблем та здійснення ефективного моніторингу стану лісів доцільно використовувати ДЗЗ, що дозволяє отримувати з космосу високоякісні зображення лісових масивів, які допомагають вирішувати завдання аналізу стану лісових покривів та прогнозувати ситуації певних лісових зон. Космічна фотографія має багато переваг, які роблять її все більш популярною. Це регулярність, простота порядку, існування архівів протягом багатьох років, найбільше висвітлення єдиного зображення, цифрова форма подання.

В Україні існує Лісовий кодекс України, що регулює правові відносини на території країни з метою забезпечення підвищення продуктивності, посилення корисних властивостей, охорони та відтворення лісів [1], а також Закон України про охорону навколишнього природного середовища, який регулює відносини у галузі охорони, використання і відтворення природних ресурсів, забезпечення екологічної безпеки, запобігання і ліквідації

негативного впливу господарської та іншої діяльності на навколишнє природне середовище [2].

За останнє десятиліття здоров'я лісів значно погіршилось через негативні наслідки глобальних змін клімату (підвищення температури та зменшення кількості опадів), які призвели до збільшення площі сушильних насаджень та масового ураження шкідниками. Останніми роками також зростає інтенсивність лісових пожеж, яка обумовлена кліматичними змінами та антропогенним чинником. Відповідно збільшується і частка згарищ у лісокультурному фонді, на яких потрібно виростити новий ліс. Статистика свідчить, що за останні півстоліття на нашій планеті було знищено не менше лісів, ніж за всю історію людської цивілізації. Саме тому терміновий облік лісів необхідний, адже за останній час ми втратили велику кількість лісової площі через стихійні лиха. Оцінити конкретні масштаби трагедії дуже важко, тому що потрібно провести колосальну роботу по обліку тих лісових насаджень, які вціліли на території України.

Перспективним напрямком підвищення інформативності космічних знімків є застосування методів фрактального аналізу зображень [3]. Для будь-якої множини існує поняття топологічної розмірності – ціла розмірність множини, тобто кількість координат, необхідних для координації елементу в множині. Деякі ж об'єкти характеризуються тільки дробовою фрактальною розмірністю. Вона дозволяє здійснювати дешифрування космічних зображень. Для лісового покриву також можна порахувати фрактальну розмірність та апроксимувати його детермінованим фракталом, розмірність якого співпадає з розмірністю нашої досліджуваної області лісу. На основі даних дистанційного зондування та використання фрактальної розмірності є можливість визначати типи лісових насаджень.

Для отримання фрактальної розмірності досліджувану область розбивають частини. Чим більша кількість цих частин, тим більша точність підрахунку фрактальної розмірності. Підраховуємо розмірність кожного з компонентів. Встановлюємо найбільшу розмірність компонента, яка і буде фрактальною розмірністю всієї області. З розрахунку випливає, що лісовий покрив потребує апроксимації детермінованим фракталом з такою самою розмірністю.

Програмна алгоритмізація передбачає як побудову детермінованого фракталу за допомогою його математичного опису – геометричний ітераційний закон, алгебраїчний або алгебраїчний з випадковим елементом, так і вже побудованого зображення фракталу. У програмі закладено побудова детермінованих фракталів, з вказанням їх фрактальної розмірності в окремому модулі програмного забезпечення.

Користувач вказує або тип структури, що досліджується, або кількість фаз середовища, а також довжину комірки розбиття. За допомогою цього відбувається градація та підрахунок фрактальної розмірності багатофазової структури. Ймовірність належності компоненту даній комірці здійснюється за допомогою кольорової гами, що відповідає за вибраний компонент в середовищі. Рівняння регресії, за допомогою якого обраховується фрактальні

розмірність компонентів, передбачає побудову як залежності від x, так і залежність від y. Вихідні дані, на вибір користувача, можна зберігати в різних форматах.

Програмний продукт повинен бути розрахований на роботу з великим обсягом даних, адже саме швидкодія системи є необхідним чинником для роботи у цій галузі.

При написанні програми завжди потрібно обрати гнучку мову програмування, яка дозволить в подальшому вдосконалювати програмний продукт та додавати новий функціонал. Саме тому було обрано мову Python, яка є незалежною, міжплатформною, відкритою мовою програмування, швидкою, потужною і легкою в освоєнні. Вона широко використовується і підтримується. В якості бази даних було обрано MySQL, що доступною, швидкою та має легкий спосіб доступу й підключення. Це реляційна система управління базами даних, яка є у вільному доступі. Відмінністю цієї системи від інших є можливість користування великою кількістю різних типів таблиць. Користувач може користуватися звичними для нас таблицями, а також працювати з транзакціями на рівні окремих записів.

Інформаційна система створена у вигляді програмного вікна, яке в подальшому можна буде встановлювати локально на ПК. Система не має авторизації, так як у програмі відсутні рівні доступу. Кожен користувач має однаковий функціонал незалежно від прав доступу. Для того, щоб розпочати роботу достатньо буде запустити програму та натиснути кнопки, які нам потрібні та цікаві. Під час розробки програми було прийнято рішення розподілити функціонал на окремі кнопки, щоб зробити роботу простішою та швидшою. Тому кожен блок відведений під різну кнопку. Потрібно самостійно створити базу даних лісництв, кварталів, виділів, завантажити файл з описами порід дерев на обраній ділянці. Вхідні дані, які будуть зберігатися у базі даних, повинні відповідати певному формату, а саме: номер класу, назва класу, кількість дерев в обраній місцевості. З використанням фракталів виділяємо класи для кожної породи та позначаємо її власним Id кодом. Після того, як ми обробили карту та виділили кожен клас, проводимо автоматичну класифікацію зображення. В результаті отримуємо карту виділів кожної породи. Для кожного класу відповідає індивідуальний колір, що спрощує процес аналізу класифікації.

Важливим аспектом використання програмного забезпечення є моніторинг лісів та автоматизація оцінки становища лісових угідь з плином часу. Таким чином, програмний продукт є зручним засобом для вирішення проблеми, яка постає перед користувачем: обробка, класифікація та аналіз лісових насаджень.

Література:

1. http://search.ligazakon.ua/l_doc2.nsf/link1/T385200.html.
2. https://zakon.rada.gov.ua/laws/show/1264-12#Text.

3. Залевська О.В. Фрактальна розмірність структури поверхневого шару деталей з композитних матеріалів. Праці Таврійського державного агротехнологічного університету, вип.4: Прикладна геометрія та інженерна графіка, т. 49, 2011. С. 139-142.

УДК 004.942 : 519.179.2

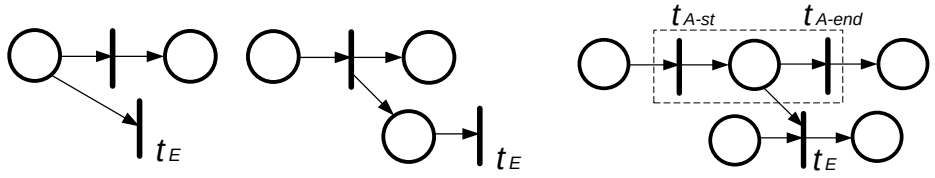
МОДЕЛЮВАННЯ ПРОЦЕСІВ СКАСУВАННЯ ТА БЛОКУВАННЯ ЗАДАЧ ПРИ ПРОЕКТУВАННІ ПРОГРАМНИХ ЗАСОБІВ

Супруненко О.О.

Черкаський національний університет імені Богдана Хмельницького

При проектуванні програмного забезпечення часто виникає потреба у реалізації припинення чи скасування певних задач, що почали виконуватись, але їх завершення та результати не потрібні для подальшого функціонування програмної системи. Це потрібно для зменшення часу обробки всього завдання в цілому. Для коректної реалізації блокування чи скасування задач, що виконуються у паралельних гілках, потрібно промоделювати варіанти такої реалізації та вибрати найбільш оптимальний для проекту. Для моделювання у роботі застосовуються мережі Петрі, що використовують безпечну та оціночну інтерпретації [1].

Блокування та скасування процесів розглянемо на основі патернів для моделювання керування потоками робіт [2]. Патерн «Скасувати завдання» (рис. 1) дозволяє припинити виконання завдання, якому передана мітка (а) (до початку його виконання), або запущене на виконання завдання (б-в) (під час його виконання) і видалити мітку з поточної гілки. Патерн 19 (в) відображає особливості припинення запущеного на виконання завдання у UML, яке полягає у розміщенні такого завдання у перервній області, яка може бути активована чи деактивована у будь-який момент часу іншим завданням або сигналом управління через вершину t_E. У багатьох інших спеціалізованих мовах та програмних середовищах [23] припинення запущеного завдання здійснюється через механізми обробки помилок та несправностей.



а) б) в)

Рис. 1. Патерн «Скасувати завдання».

Патерн «Скасувати процес» (рис. 2) забезпечує припинення виконання певного процесу і всіх завдань, що з ним пов’язані. У варіанті (а) цього

патерну реалізовано механізм вилучення міток з усіх проміжних вершин місць процесу, що скасовується.

Але для коректного скасування всіх задач, потрібно, щоб серед них не було таких задач, які пов'язані із задачами інших процесів, оскільки після скасування з інших процесів у скасоване завдання можуть надходити мітки. Тому варіант (б), в якому представлені нормальна робота процесу та альтернатива скасування процесу, дозволяє контролювати активності у процесі та у разі його скасування блокувати будь-які події з обробки завдань.

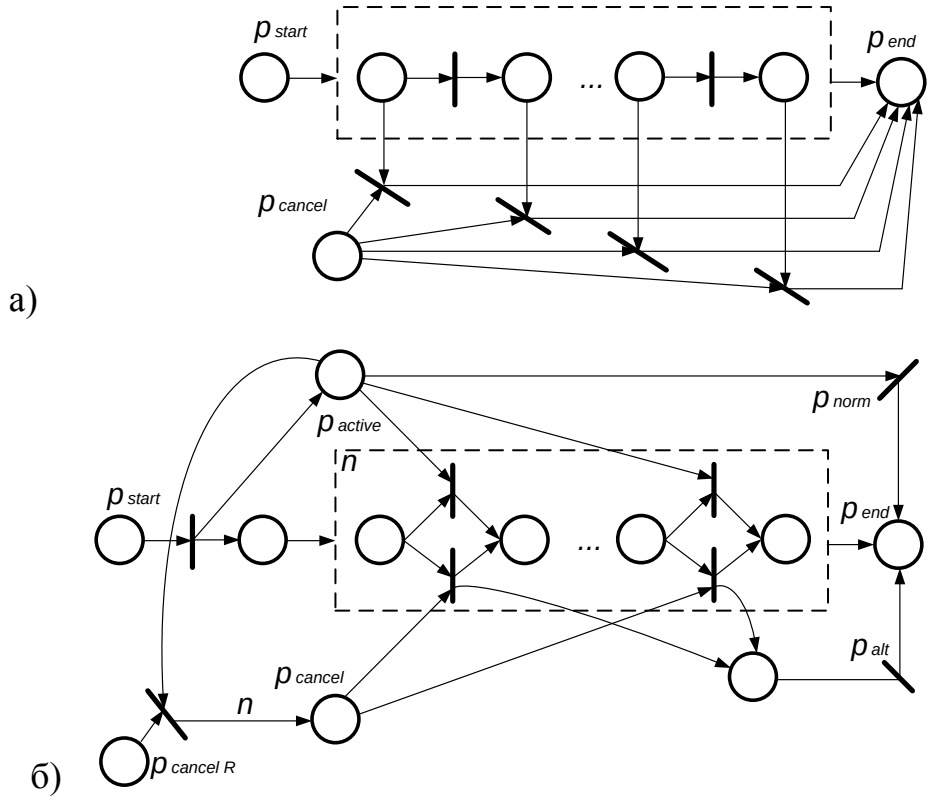


Рис. 2. Патерн «Скасувати процес»:
а) скасування виконання всіх активних завдань,
б) нормальна робота та альтернативний варіант(скасування процесу).

При моделюванні процесів у програмних системах важливо, щоб після скасування окремого процесу з усіх вершин місць цього процесу не тільки вилучалися мітки, але і у «контролюючих» вершинах місць (якщо такі застосовуються у моделі) встановлювалась кількість міток, яка відповідає початковій розмітці цього процесу (для забезпечення коректного запуску цього процесу у наступному циклі моделювання).

Патерн «Скасувати багаторазову активність» (рис. 3, а) дозволяє скасувати подальшу обробку завдань при паралельному їх виконанні, які ще незавершені (не оброблені у вершині переходу t_{Ve}), завершені екземпляри завдань не скасовуються. При реалізації даного патерну з усіх вершин місць, які відображають виконані завдання, але не оброблені у синхронізуючій вершині переходу t_{Ve}, вилучаються всі мітки у вершину t_{cancel}, також і з вершин переходів мітки переходять у вершину місця p_{can}, а далі до вершини

переходу t_{cancel}, результати скасування – у вершині місця p_{cancel}. Патерн «Завершити багаторазову активність» (рис. 3, б) має відмінність від попереднього патерну у тому, що виконання завдань, які не завершились, завершується, але інші екземпляри не створюються – мітки з вхідних місць до вершин переходів $t_1 \ldots t_n$ виконання завдань вилучаються у вершину переходу t_{cancel}, і як тільки завдання завершились і мітки у вершину t_{cancel} вилучені, на наступному кроці роботи моделі потік управління може бути переданий наступному процесу через кінцеву вершину місця p_{Ve}.

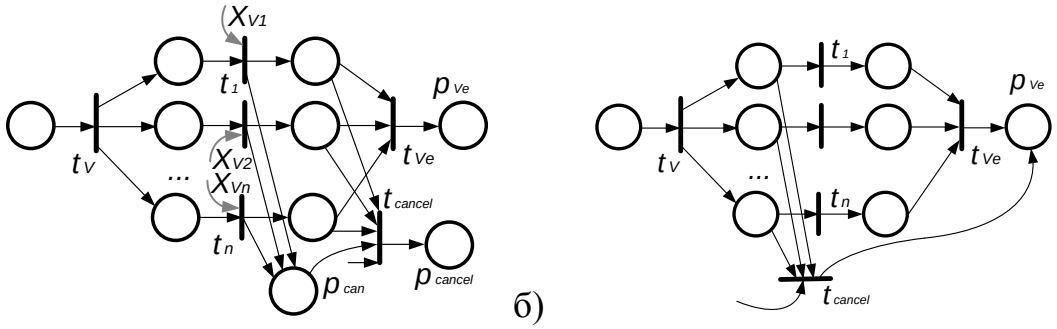


Рис. 3. Патерни:
а – «Скасувати багаторазову активність»
б – «Завершити багаторазову активність».

Патерн «Блокування часткового з'єднання» (рис. 4) відображає можливість паралельного виконання у *n* гілках лише по одному екземпляру певного завдання, результати виконаних *m* перших завдань будуть впливати на наступні завдання (через вершину p_{v2}), а результати *(n-m)* завдань будуть блокуватися (у вершині місця t_e). Наступні *m* завдань (по одному екземпляру кожного завдання) можуть виконатися тільки після передачі наступному процесу результатів поточного виконання патерну та відновленню розмітки.

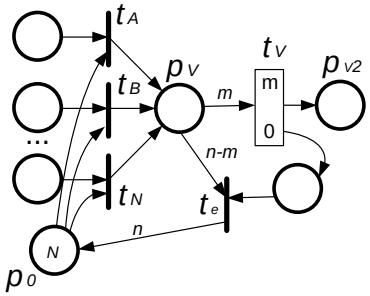


Рис. 4. Патерн «Блокування часткового з'єднання».

Таким чином, запропоновані конструкції патернів дозволяють змоделювати основні варіанти скасування та блокування запущених на виконання задач та передбачити у проекті варіанти реалізації процесів припинення виконанні задач, результати яких не впливають на подальші процеси у програмному засобі.

Література:

1. Кузьмук В.В., Супруненко О.О. Модифицированные сети Петри и устройства моделирования параллельных процессов: Монография. – К.: Маклаут, 2010. – 252 с.
2. N. Russell, A.H.M. ter Hofstede, W.M.P. van der Aals, N. Mulyar. Workflow Control-Flow Patterns: A Revised View. BPM Center Report BPM-06-22 , BPMcenter.org. Available at: http://www.workflowpatterns.com/patterns/control/ (accessed 03.03.21).
3. Control-flow patterns. [E-resource] Available at: http://www.workflowpatterns.com/patterns/control/index.php (Accessed 09.05.21).

УДК 004.8

ВИКОРИСТАННЯ БІОНІЧНИХ ПРИНЦИПІВ ПРИ МОДЕЛЮВАННІ РУХУ УГРУПУВАННЯ АВТОНОМНИХ АГЕНТІВ

Ярмілко А.В., Нікітюк В.С.

Черкаський національний університет імені Богдана Хмельницького

Дослідження мультиагентних систем за сценаріями прикладного використання, притаманними swarm robotic, одним з ключових напрямів мають вибір оптимальної структури зв’язків у групі агентів та адекватного типу їх поведінки у багатофакторному функціональному просторі. Одним з продуктивних підходів є використання моделей біонічної природи.

Біонічні моделі спираються на спостереження комунікативних структур та поведінкових алгоритмів, притаманних живим організмам. У випадку систем swarm robotic прототипом є поведінка зграйних істот. Загальні вимоги до деяких типових сценаріїв поведінки агентів у групі роботів представлені, зокрема, у публікаціях [1, 2]. Вони передбачають формування та переформатування конвоїв, безпечний рух у групах роботів за умови відсутності апріорних знань про оточення, рух за сценаріями втечі від зграї або захоплення зграєю вказаного рухомого об’єкту, синхронне виконання рухів за спостереженням визначеного лідера, т. і..

При проектуванні та реалізації алгоритмів керування груповою поведінкою таких мобільних агентів потрібно враховувати ряд притаманних swarm robotic особливостей. Вони обумовлені, зокрема, тим, що кожному з роботів (автономних агентів) доступна лише локальна інформація про його оточення. Така ситуація є об’єктивним наслідком обмеженості каналів отримання даних, їх діапазонів та радіусу дії, особливостей комунікаційної структури у межах групи та інших факторів, обумовлених властивостями функціонального простору. Тому ці особливості прикладної області слід враховувати при визначенні топології групи та базових сценаріїв поведінки агентів на елементарних кроках керування.

У розглянутому нами варіанті постановки задачі групової поведінки агентів – консолідація групи навколо рухомого лідера – використана графова модель комунікаційних зв’язків всередині групи [3, 4]. Поведінка кожного з

агентів на кожному з кроків вирішення групового завдання формується на основі спостереження за агентами-сусідами. При цьому у застосованій моделі кожен агент здійснює спостереження за одним довільним членом групи. Така топологія зв'язків ґрунтується на дослідженнях з іхтіології [5, 6], які спростовують чимало припущень щодо механізмів самоорганізації угрупувань зграйних риб. Зокрема, ці дослідження показують, що швидка і успішна консолідація особин у зграї відбувається на підставі слідування окремої рибини маневрам довільного сусіда, але з можливістю на певному кроці виконувати довільні (випадкові) рухи.

У реалізованій програмній системі моделювання поведінки зграї агентів комунікаційна структура задається графом, який може формуватися перед початком сеансу моделювання дослідником або генерується автоматично з урахуванням статусу агентів (лідер або переслідувач). Це дозволяє забезпечити високу варіабельність модельних ситуацій навіть без зміни початкових координат агентів. Подальше підвищення цього показника вбачається при впровадженні у модель консолідаційної задачі динамічного графа, який забезпечуватиме перебудову комунікаційних зв'язків у групі на кожному кроці моделювання. Такий розвиток структури зв'язків між агентами забезпечить більш повну відповідність модельних ситуацій природним аналогам та є практичною відповіддю на обмеженість реальних каналів і апаратури моніторингу, здатності апаратних версій мобільних агентів здійснювати маневри слідування конкретним агентам-сусідам з множини можливих. Для врахування потенційно можливих похибок у роботі рушійних механізмів, датчиків, програмних алгоритмів у системі моделювання була передбачена деяка міра невизначеності у формі стохастичного коефіцієнту.

Досліди з імітації групової поведінки у розробленому середовищі моделювання продемонстрували адекватність отриманих динамічних моделей поведінці реальних прототипів таких систем у функціональних середовищах зі складними властивостями. Зокрема, стохастичний коефіцієнт дозволив врахувати неідеальність параметрів функціонального середовища та технічної реалізації агентів (рис. 1).

Рис. 1. Залишкові траєкторії руху агентів у процесі модельного експерименту з консолідації навколо рухомого лідера (0 – позиція лідера; 1, 2, 3, 4, 5, 6 – позиції агентів-переслідувачів).

Таким чином, використання біонічних підходів є виправданим кроком при вирішенні задач swarm robotic. Вони дозволяють отримати ефективні

рішення для розв’язання як теоретичних, так і практичних завдань при створенні багатоагентних інтелектуальних систем.

Література:

1. Gonzalez R. How a flock of drones developed collective intelligence / Robbie Gonzalez / Available at: https://www.wired.com/story/how-a-flock-of-drones-developed-collective-intelligence/
2. Зенкевич С.Л., Чжу Х. Управление движением группы роботов в строю типа «конвой». Мехатроника, автоматизация, управление. 2017;18(1):30-34.
3. Ярмілко А.В. Web-сервіс для дослідження динаміки автономних самокерованих модулів у процесі моделювання консолідованого руху / А.В. Ярмілко, В.С. Нікітюк // Інформаційні моделюючі технології, системи та комплекси (ІМТСК-2020) / Матеріали другої міжнародної науково-практичної конференції (Черкаси, 27-29 травня 2020 р.). – Черкаси: ЧНУ, 2020. – С. 19-21.
4. Ярмілко А. Імітаційне моделювання консолідованого руху автономних агентів / А.В. Ярмілко, В.С. Нікітюк // Проблеми зняття з експлуатації об’єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 21) : збірник матеріалів VI Міжнародної конференції (27–29 квітня 2021, м. Славутич). – Чернігів : НУ «Чернігівська політехніка», 2021. – С. 304-306.
5. Jiang L, Giuggioli L, Perna A, Escobedo R, Lecheval V, Sire C, et al. (2017) Identifying influential neighbors in animal flocking. PLoS Comput Biol13(11): e1005822. https://doi.org/10.1371/journal.pcbi.1005822 pmid:29161269
6. Jhawar J., Morris R.G., Amith-Kumar U.R. et al. Noise-induced schooling of fish // Nat. Phys. 16, 488–493 (2020). https://doi.org/10.1038/s41567-020-0787-y

УДК 004.8

МОДЕЛЮВАННЯ ДИНАМІЧНИХ ПАРАМЕТРІВ БАГАТОАГЕНТНОЇ СЦЕНИ ЗА РЕЗУЛЬТАТАМИ ВІЗУАЛЬНОГО СПОСТЕРЕЖЕННЯ РУХУ АГЕНТІВ

Мормуль О.І., Ярмілко А.В.

Черкаський національний університет імені Богдана Хмельницького

Багато сучасних прикладних інформаційних систем використовують алгоритми, які базуються на відстежуванні об’єктів, тобто визначенні місцеположення рухомого об'єкту (декількох об'єктів) в часі за допомогою камери. Такі алгоритми аналізують кадри відео та видають координати положення рухомих цільових об'єктів щодо кадру.

Основну проблему в відслідковуванні складає зіставлення положень цільового об'єкта на послідовності кадрів, особливо якщо об'єкт рухається швидко щодо частоти кадрів. Таким чином, системи відслідковування

зазвичай використовують модель руху, яка описує можливі зміни зображення цільового об'єкта при різноманітних його рухах [1]. Прикладами таких простих моделей руху є:

відслідковування плоских об'єктів, модель руху – 2D перетворення;

якщо цільовим є жорсткий 3D об'єкт, модель руху визначає вид залежно від його положення в просторі і орієнтації;

зображення деформованого об'єкта може бути покрито сіткою (mesh), рух об'єкта задається положенням вершин цієї сітки.

Основне завдання алгоритму відслідковування – це послідовний аналіз кадрів відео для оцінки параметрів руху. Ці параметри характеризують стан цільового об'єкта [1]. Для детектування об'єкта на зображенні застосовуються алгоритми розпізнавання. Алгоритм розпізнавання зображень приймає кадр в якості вхідних даних і виводить список об’єктів, що міститься на даному зображенні.

Класифікація зображень проводиться поетапно. На першому кроці вхідне зображення найчастіше попередньо обробляється для нормалізації контрасту і яскравості, а також на цьому кроці вхідне зображення обрізається і масштабується до фіксованого розміру.

На другому кроці необхідно спростити зображення шляхом вилучення важливої для ідентифікації інформації, оскільки вхідне зображення містить занадто багато додаткової інформації, яка не потрібна для класифікації. Цей крок називається витяганням ознак. Існує досить велика кількість ознак, що використовуються в комп'ютерному зорі, – це ознаки Хаара, HOG (Histogram of Oriented Gradients), SIFT (Scale-Invariant Feature Transform), SURF (Speeded Up Robust Feature) та інші.

На третьому кроці алгоритм класифікації приймає вектор ознак в якості вхідних даних і визначає, до якого класу належить зображення.

Відстеження об'єкта в деяких випадках може виконуватися за допомогою алгоритмів детектування. При детектуванні основна ідея полягає в тому, щоб спочатку визначити регіони інтересу (ключові точки), які будуть незалежні до перетворень. Потім для кожного регіону інтересу будується його векторне подання – дескриптор. Далі на кожному кадрі буде виконуватися пошук об'єкта і виділення області його розташування прямокутником.

При застосуванні технології тренінгу метою є знаходження об'єкта в поточному кадрі, якщо він успішно спостерігався на всіх попередніх кадрах. Оскільки об'єкт був відстеженим до поточного кадру, відомими є параметри моделі руху: швидкість і напрямок руху об'єкта в попередніх кадрах. Тому можна передбачити нове місце розташування об'єкта, спираючись на його модель руху, і воно буде дуже близьким до реального нового положення об'єкта [2].

У ході дослідження було виконано аналіз переваг та недоліків перелічених методів і засобів їхньої програмної реалізації. За наслідками цього аналізу пошук об’єктів на зображенні виконується за допомогою інструментів OpenCV. Застосовано підхід для відстежування об’єкта на

основі виявлення їх контурів. Вибір метода обумовлений його придатністю для роботи з монохромними зображеннями, що дозволяє ефективно знижувати шуми шляхом перетворення кольорової гами та розмиття зображення за заданим коефіцієнтом. Для кожного контуру обчислюється максимальна обмежуюча (габаритна) рамка. В процесі відстежування виконується зіставлення вмісту кожної обмежуючої рамки з однією з відомих (зразкових) фігур. У проекті кожна обмежена габаритною рамкою область масштабується до розміру шаблону і при їхньому порівнянні обчислюються відмінності в градієнті.

Однак застосований підхід не є універсальним: він буде неефективним, коли фігури накладені одна на одну (мають спільну межу). Виявлення контуру вловлює дві сусідні фігури як єдиний контур (один обмежуючий прямокутник). У такому випадку цей крок аналізу, очевидно, зазнає невдачі. Також обмеження методу пов’язані з наявною бібліотекою зразкових образів, які утворюють базу ідентифікації.

Для спостереження за поведінкою об’єкта реалізовано визначення швидкості руху, прискорення, часу, пройденої об’єктом відстані. Всі результати дослідження заносяться до бази даних. За наслідками формується звіт, в якому графічно відображується схема руху, тренди зростання або спадання швидкості руху за час дослідження, виводяться аналітичні данні.

В результаті проведених робіт було отримано програму відстеження рухомих агентів на мультиагентній сцені, яка дозволила провести експерименти з ідентифікації об’єктів за заданими параметрами форми та визначити їх положення і параметри руху, формувати графічні звіти щодо динаміки руху та аналітичних даних. Запропонований інструментальний засіб буде максимально ефективним при відслідковуванні об’єктів наперед визначених типів, візуальні образи яких зберігаються у бібліотеці програми. Похибки детектування, спричинені накладанням об’єктів на окремих кадрах, можуть бути компенсовані застосуванням інтелектуальних алгоритмів, які використовують моделі відновлення даних щодо руху об’єктів на відрізках траєкторії зі зближенням з сусідами.

Література:

1. Трекинг (компютерная графика) [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/Трекинг_(компьютерная_графика)
2. Аналіз відео [Електронний ресурс]. – Режим доступу: https://neerc.ifmo.ru/wiki/index.php?title=Анализ_видео

Секція 2

Системне та прикладне програмне забезпечення

УДК 004.052.42

ОСОБЛИВОСТІ ТЕСТУВАННЯ НА СТАДІЇ КОДУВАННЯ

Іванов М.О., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

Упродовж останніх років, люди користуються великим різноманітним програмним забезпеченням (ПЗ), до якого вони звикли настільки, що все це їм здається чимось простим та буденним. Наприклад, пошукові запити на будь-яку тематику, використовуючи веб-браузери на своїх телефонах або комп'ютерах, спілкування з друзями, родичами або якимось незнайомцями в соціальних мережах, месенджерах, коментарях під відео або за допомогою звичайних SMS мобільних операторів та навіть очищення підлоги дому із використанням робота порохотяга тощо.

Люди звикли до того, що все повинно працювати якісно, без помилок та затримок під час завантаження або обробки якихось запитів. Навіть дуже маленьке розходження між очікуванням та результатом, може викликати у користувачів обурення, що у свою чергу складе погане враження про саме ПЗ та команду розробників. Зазвичай користувачам однаково, хто саме займався розробкою – багатомільярдна корпорація з тисячами співробітників чи невеликий стартап з декількох осіб, які роблять усе власноруч, починаючи від дизайну та закінчуючи фінальним тестуванням.

У таких невеликих командах дуже часто всі її члени є як розробниками так і тестувальниками. Тому, умовно тестування можна розділити на:

- ручне, яке полягає в тому, що після кожного оновлення ПЗ, необхідно власноруч перевіряти новий функціонал на коректність роботи та чи немає він ніякого впливу на інші частини додатку;
- автоматизоване, яке включає написання певного коду, який буде сам перевіряти деякі частини ПЗ, що у свою чергу допомагає скоротити час на тестування і спростити його процес.

Зазвичай жоден із стартапів не може обійтися без автоматизованого тестування, тому що додавання нового функціоналу, оновлення та виправлення вже існуючого та навіть якісь глобальні зміни в архітектурі ПЗ відбуваються досить часто у таких командах. Найбільш популярним є підхід «TDD» або «Розробка через тестування».

Розробка через тестування – це техніка розробки ПЗ, яка базується на перетворені вимог до певних частин проекту у вигляді окремих тест-кейсів [1]. Спочатку пишеться сам тест, який перевіряє нову частину коду або внесені зміни у вже існуючому. Далі реалізовується необхідний функціонал, використовуючи раніше написаний тест. Після успішного проходження тесту, виконується рефакторинг для збереження єдиного стилю коду у проекті. Детальна діаграма цього процесу наведена на рис. 1.

Рис. 1. Техніка «TDD».

Як видно з рис. 1, описаний на діаграмі підхід, виконується циклічно, де на кожній ітерації розробляється або редагується якась частина додатку. Такий підхід розробки можна застосувати до будь-якого типу ПЗ, починаючи з невеликих додатків, які, наприклад, просто зберігають персональні замітки користувачів, закінчуючи складними фінансовими системами.

Для прикладу розглянемо тестування власної функції додавання двох чисел. Спочатку створюємо тест (рис. 2), де вказуються аргументи, які необхідно додати, та очікуваний результат обчислень.

```
func TestSumSuccess(t *testing.T) {
    a, b := 5, 10
    expectedResult := 15

    result := sum(a, b)

    if result != expectedResult {
        t.Errorf( format: "Expected to get %v but got %v", expectedResult, result)
    }
}
```

Рис. 2. Тест для перевірки правильності обчислення функції додавання двох чисел.

Далі виконуємо виклик пустої функції «sum», яку необхідно буде реалізувати пізніше. Після чого виконуємо перевірку чи дорівнює результат виклику функції очікуваному результату. Якщо відповіді не співпадають, виводимо повідомлення про помилку обчислень.

Виконаємо перший запуску тесту, який відобразить помилку, тому що функція нічого не повертає, а очікувана відповідь дорівнює числовому значенню. Після того як переконалися, що тест працює і виводить помилку, можна реалізувати саму функцію додавання за допомогою вбудованого оператора «+» у мові програмування Golang [2]. Знову запустимо тести, щоб переконатися у правильності обчислень (рис. 3). Тепер змінимо вхідні дані на неправильні і запустимо тест ще один раз, щоб переконатися у тому, що логіка перевірки працює вірно (рис. 4).

```
=== RUN   TestSumSuccess
--- PASS: TestSumSuccess (0.00s)
PASS
```

Рис. 3. Успішне проходження тесту.

```
=== RUN   TestSumError
    empty_test.go:40: Expected to get 5 but got 4
--- FAIL: TestSumError (0.00s)

FAIL
```

Рис. 4. Неуспішне проходження тесту.

При подальшій розробці даного додатку, можна буде з легкістю переконуватися у тому, що новий функціонал ніяк не вплинув на логіку роботи вже існуючого. Також такий підхід дуже сильно економить час як

розробникам, так і тестувальникам за рахунок того, що після усіх змін у коді, вже відомо, що все працює правильно.

Література:

1. Introduction to Test Driven Development (TDD) : веб-сайт. URL: http://agiledata.org/essays/tdd.html (дата звернення: 14.03.2021).
2. Golang: основи для початківців : веб-сайт. URL: https://echo.lviv.ua/dev/6247 (дата звернення: 14.03.2021).

УДК 004.054

ТЕСТУВАННЯ ПРИЗНАЧЕНОГО ДЛЯ КОРИСТУВАЧА ІНТЕРФЕЙСУ

Міщенко А.І., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

Є такий вираз «Зустрічають по одягу…», який можна застосувати не тільки до вигляду людини при першій зустрічі, а й до сучасних програм та сайтів. Коли користувач вперше знайомиться з додатком, найперше він звертає увагу на вигляд користувацького інтерфейсу, а через деякий час він вже розуміє наскільки цей додаток зручний та приємний в користуванні.

Користувацький інтерфейс – це набір графічних елементів (кнопки, поля текст, картинки, анімації тощо, з якими взаємодіє користувач). Від нього часто залежить чи буде людина задоволена користуванням і чи вирішить у майбутньому стати клієнтом для компанії, яка розробляла цей додаток. Тому дуже важливо проектувати користувацький інтерфейс із зрозумілим візуальним виглядом, щоб користувачу зрозуміло було відразу як працювати з таким продуктом. Також важливо, щоб відбувалася взаємодія інтерфейсу із користувачем, тобто при натисканні кнопок або при наведенні на активні елементи відповідь була у вигляді анімацій. Для досягнення такого результату потрібно проводити тестування користувацького інтерфейсу. Деякі компанії нехтують цим, а після цього втрачають клієнтів, і внаслідок цього відбувається у спаді продаж [1; 2].

Однак, як і будь-який програмний продукт, тестування користувацького інтерфейсу має ряд недоліків та переваг [3]. До переваг можна віднести:

- перевірка взаємодії компонентів з сервісами;
- перевірка дій зі сторони користувача;
- перевірка взаємодії інтерфейсу з користувачем;
- перевірка надійності додатку.

Недоліками такого тестування вважають:

- великий затрачений час на проведення тестування;
- важкий процес автоматизації тестування;
- не підходить для тестування невеликих додатків.

Також до уваги необхідно брати який метод тестування береться за основу. До основних методів тестування відносять [3]:

1. Тестування паперових прототипів (див. рис. 1). Розробляється та візуалізується прототип на папері, і проводиться тестування кількома людьми – цільовою аудиторією. Це допомагає виявити проблеми навігації, дизайну та функціональності ще перед безпосереднім початком розробки додатку. У подальшому це заощадить час та кошти.

2. Тестування інтерактивного прототипу. Такі тести допомагають протестувати базову логіку та виявити очікування користувачів, але 8 із 10 тестувальників часто забувають, що вони тестують всього лише прототип.

3. Метод оцінки сприйняття дизайну – дозволяє зрозуміти, чи викликає дизайн цільові емоції, для цього використовується сформований список необхідних емоцій.

4. Метод тестування сподівань – тест на очікувану відповідь від додатку при якійсь взаємодії з інтерфейсом.

5. Метод «думки вголос» – при тестуванні учасники завжди висловлюють свої думки вголос.

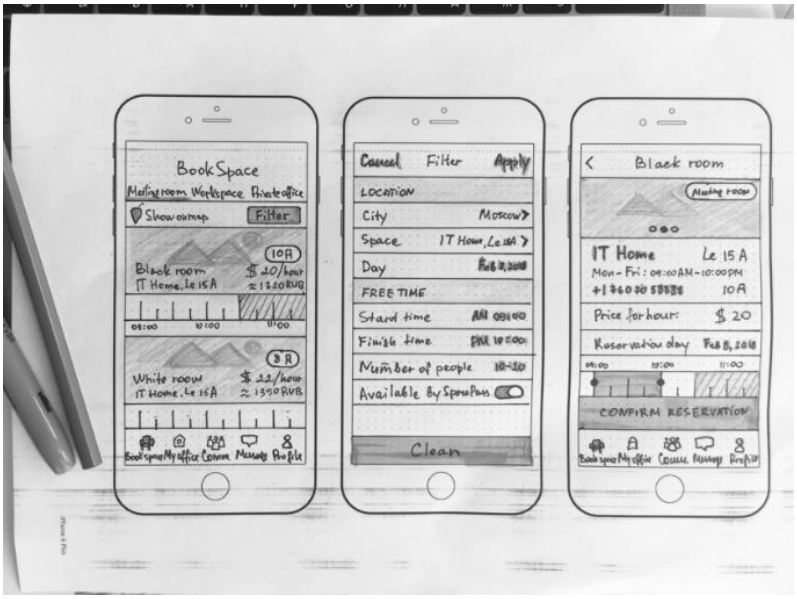


Рис. 1. Тестування паперового прототипу.

При тестуванні користувацького інтерфейсу варто звернути увагу на правильність написання тексту та локалізації. Якщо продукт відповідає цим вимогам та має локалізацію на різні мови світу, то користуватись таким додатком зможуть в більшості куточків планети, та й цінність продукту в очах користувачів значно підвищиться.

Отже, можні зробити висновок, що у необхідність тестування призначеного для користувача інтерфейсу має для людей велике значення, плюси та мінуси такого тестування, методи та їх опис. Варто зазначити що для невеликих додатків немає великої необхідності використовувати важкі методи для тестування інтерфейсу, так як це буде не ефективно. Також необхідно звернути увагу на тестування локалізації в продуктах, так як

правильна локалізація полегшує користувачам роботу з додатком, що в свою чергу залучає більше користувачів.

Література:

1. UI тесты – всегда ли нужны? : веб-сайт. URL: https://habr.com/ru/post/335776/ (дата звернення: 14.03.2021).
2. Тестирование UI (пользовательского интерфейса) : веб-сайт. URL: https://woxapp.com/ru/our-blog/testing-the-ui-user-interface/ (дата звернення: 14.03.2021).
3. Лекційні матеріали навчального курсу «Software testing for universities». Провідна українська компанія з тестування програмного забезпечення QATestLab Ukrainian HI-Tech Initiative, Київ, 2019. 353 с.

УДК 004.054

ПЛАНУВАННЯ ТЕСТУВАННЯ

Мормуль О.І., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

Розробка якісного продукту неможлива без налагодженого процесу тестування. Сучасні програмні продукти використовують велику кількість можливостей взаємодії з користувачами, тестування яких часто важко піддається автоматизації. Тому ручне тестування залишається важливим елементом в процесі підготовки якісного продукту. З іншого боку, автоматичні тести вимагають великих трудовитрат і не завжди дають повне покриття, тому ручне тестування, як і раніше залишається важливою частиною процесу розробки. Багато команд і компаній стикаються з проблемою відсутності зв’язків між інструментарієм розробників та інструментарієм тестувальників. Помилки та доопрацювання заносяться в баг-трекери, тестові сценарії, самі результати тестування містяться в текстових файлах або в інших базах даних. Як наслідок, розробники працюють з баг-трекером, керівники з інструментом планування, тестувальники з текстовими файлами, а тімліда і з тим і з іншим, витрачаючи на це величезну кількість часу [1].

Відсутність зв’язку між розробкою і тестуванням негативно позначається в підсумку на якість продукту в проекті, що породжує хаос і напругу, яка в підсумку виливається в незадоволення результатами роботи та загальним зниженням мотивації команди. Планування будь-якого процесу визначається як діяльність, пов’язана з оптимальним розподілом ресурсів, постановкою завдань для успішного проведення такого процесу.

Тестування планування повинно складатися з [2]: створення тест-плану, продумування стратегії тестування, оцінки трудовитрат, прогнозування

термінів і складання графіка проведення тестування, діяльності з оцінки ризиків, визначення використовуваних інструментів.

Найчастіше результатом планування є створений окремий документ – план тестування (тест-план). Перед початком планування тестування необхідно мати чітке розуміння бачення проекту, вимог до проекту і плану випуску проекту. Ці відомості необхідно зібрати в результаті спільної роботи з іншими зацікавленими особами, сам процес планування тестування буде включати наступні етапи [1-2].

Етап 1. Створення та інтеграція планів тестування – створення плану тестування, який можна пов’язати з будь-якими заданими вимогами та інтегрувати з планом розробки випуску. Якщо для випуску призначено декілька ітерацій, то можна створити основний план тестування випуску і допоміжні плани тестування для кожної ітерації в такій послідовності: створення основного плану тестування; зв’язність з набором вимог; зв’язність з планом розробки випуску.

Етап 2. Визначення вимог до якості, які задаються у плані тестування, і які можна використовувати для визначення загальних вимог до якості випуску, а також необхідних вхідних і вихідних критеріїв. Вхідними критеріями визначаються попередні вимоги, які необхідно виконати перед початком тестування. Вихідними критеріями визначаються умови, які потрібно виконати перед підбиттям підсумків тестування. Ці цілі можна використовувати для визначення рівня якості, якого необхідно досягти для того, щоб вважати продукт готовим: задати загальні вимоги до якості; задати вхідні критерії; задати вихідні критерії.

Етап 3. Створення тестових наборів, які необхідні для задоволення кожної вимоги, і кожен з них пов’язується із завданням плану розробки. Він включає наступні етапи: узгодження плану тестування з вимогами: зв’язати плани тестування з набором вимог та створення тестових наборів для кожної вимоги або створення безлічі тестових наборів на підставі набору вимог; узгодження плану тестування з планом розробки випуску: зв’язати кожен тестовий набір із завданням плану випуску та переконатись, що для кожного завдання існує принаймні один тестовий набір; створення допоміжних тестових наборів для підгруп; призначення тестувальникам завдання над кожним допоміжним планом тестування.

Етап 4. Створення середовищ тестування визначає пріоритети платформ і середовищ тестування, які потрібно протестувати. Для створення відповідних комбінацій можна використовувати інструменти автоматизації. Наприклад, може знадобитися обмежити тестування деякої бази даних, визначеного браузера або сервера додатків однією операційною системою або апаратною платформою, а тестування іншої бази даних, іншого браузера або сервера – іншою. Цей етап повинен включати: планування огляду потрібних платформ; створення середовища тестування; додавання існуючих середовищ тестування в план тестування.

Етап 5. Перевірка плану тестування з зацікавленими особами включає підтвердження охоплення всіх вимог і пунктів плану (сюжетів), а потім перевірки цього плану тестування із замовниками.

Література:

1. Планирование тестирования веб-сайт. URL: https://tmguru.ru/baza-znanij/protsess-testirovaniya/planirovanie/ (дата звернення: 14.03.2021).
2. Планирование тестирования веб-сайт. URL: https://www.ibm.com/support/knowledgecenter/ru/SSYMRC_6.0.1/com.ibm.team.concert.doc/topics/s_calm_plantest.html (дата звернення: 14.03.2021).

УДК 004.054

ІНТЕГРАЦІЙНЕ ТЕСТУВАННЯ ДЛЯ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ

Оношко М.І., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

Багато етапів та змін пройшло у сфері комп'ютерних технологій. Покращувались як самі електронно-обчислювальні машини, так і засоби роботи з ними. З'являлися нові мови програмування, змінювались підходи до програмування і все для того, щоб створити максимально зручну платформу та інструментарій майже для кожної можливої задачі, яка потребує програмного вирішення. Самим популярним підходом до розробки програмного забезпечення є об'єктно-орієнтоване програмування (ООП), яке підтримує більшість сучасних мов програмування.

Суть ООП полягає в тому, що всі компоненти, якими оперує предметна область виносяться в класи та об'єкти. Клас описує функціонал та/або дані (поля) з якими будуть створюватися об'єкти. Вважається правильним підходом розділяти класи на ті, які будуть використовуватися для передачі даних (Data Transfer Object – DTO) та класи, які описують або реалізують функції в об'єктах цього класу(функціональні класи – інтерфейси та їх реалізації) [1-2].

Добре написаний код з ООП повинен відповідати трьом парадигмам:

- наслідування – винесення спільного для декількох класів функціоналу та правил в спільний батьківський клас, де у класах нащадках повинні бути вже описані більш конкретні риси цього нащадка;
- інкапсуляція – в коді не має бути прямого доступу до даних, які зберігає клас. Дані, які зберігаються в екземплярах класів повинні модифікуватися та отримуватися через спеціальні методи;

– поліморфізм – полягає в тому, що функціонал, який необхідний від компонента виносять в інтерфейс і використовують екземпляр інтерфейсу замість прямого виклику метода класу.

Тестування коду, який написаний з використанням ООП, повинен мати перевірки як на дотримання вище описаних парадигм, так і на коректу взаємодію між компонентами системи. Для того, щоб перевірити як взаємодіють компоненти системи між собою використовується інтеграційне тестування, що дозволяє зробити перевірку на відповідність вимогам, при якому перевіряються цілі модулі або окремі частини системи. Інтеграційним тестуванням також перевіряють працездатність всієї системи, а не тільки її окремих компонентів [2-3].

При інтеграційному тестуванні існує декілька підходів [3]:

1. Знизу вверх. Спочатку збираються компоненти та/або модулі самих низьких рівнів, поступово піднімаючись по ієрархії доходячи до цілої системи. Для використання цього підходу необхідно, щоб всі компоненти були реалізовані і готові до тестування.

2. Зверху вниз. Цей підхід починає тестування з самого високого рівня системи і йде вниз до самих мінімальних компонентів. У процесі тестування цим методом в кожен компонент вищого рівня замість підстановки справжніх компонентів нижчих рівнів підставляються їх заглушки, що дозволяє тестувати логіку тільки поточного компонента, не спираючись на його залежності.

3. Великий вибух. Всі компоненти збираються в єдину повну систему і тестуються на функціях системи.

Для прикладу протестуємо простий приклад системи, яка використовує дизайн шаблон MVC. У цій системі є чотири функціональні класи, які розміщені на трьох рівнях (рис. 1). Для тестування використаємо перший підхід – зверху вниз.

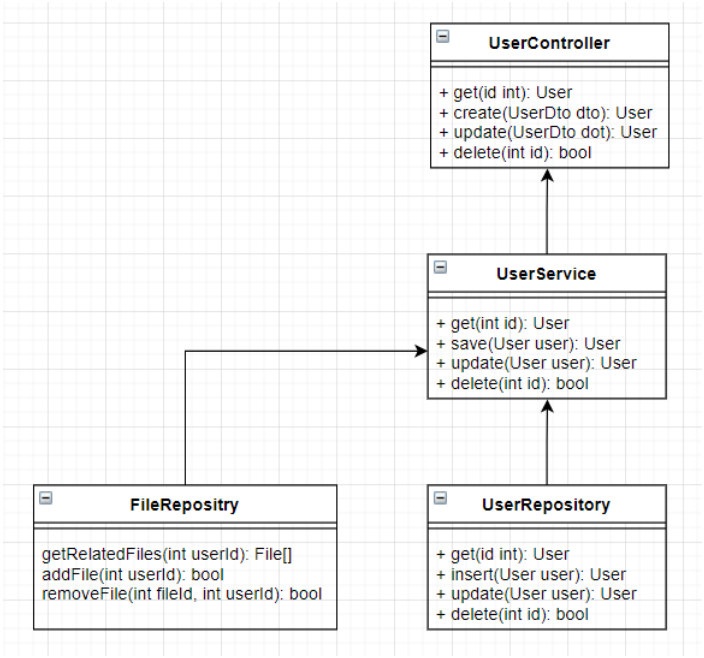


Рис. 1. Діаграма класів модулю.

Оскільки тестувальнику текст програми відомий з детальністю до виклику всіх модулів, то використання структурних критеріїв на даному етапі можливе і виправдане [1]. Тому, тестуючи кожен рівень, потрібно підставити заглушки компонентів нижчих рівнів – це дозволить тестувати тільки код, який стосується безпосередньо тестованого компоненту та забезпечить дотримання інкапсуляції усіма компонентами, оскільки в тесті кожен компонент матиме доступ тільки до своїх даних і якщо в коді системи виконується спроба дістати дані з залежного компоненту, що порушує інкапсуляцію – це буде видно в тест-кейсах (рис. 2).

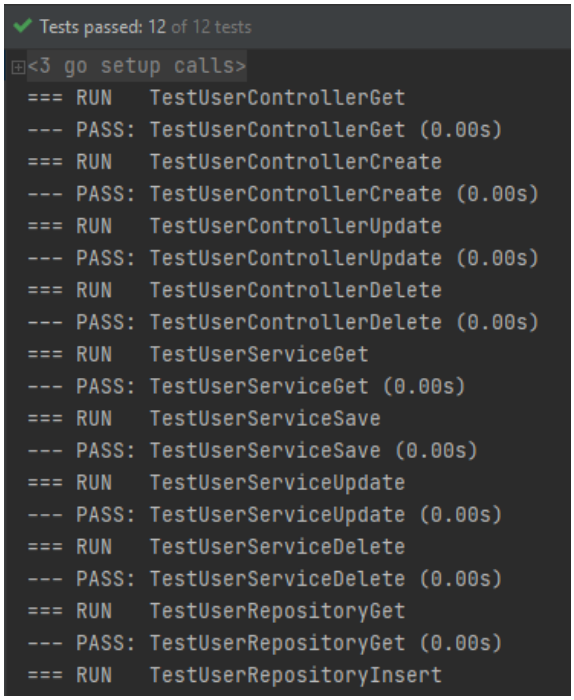


Рис. 2. Запуск всіх тест-кейсів.

Отже, було отримано три набори тестів. Кожен набір для кожного рівня ієрархії класів. Можна зробити висновок, що інтеграційне тестування для об’єктно-орієнтованого програмування дає можливість виявити помилки та усунути їх на цьому етапі без повного переписування коду.

Література:

1. Авраменко В.С., Авраменко А.С., Косенюк Г.В. Тестування програмного забезпечення. Навчальний посібник. Черкаси : ЧНУ імені Богдана Хмельницького, 2017. 283 с.
2. Гамма Є., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы обьектно-ориентированного программирования. Паттерны проектирования. СПб. : Питер, 2001. 368 с. URL: http://lib.mdpu.org.ua/e-book/vstup/L/gamma.pdf (дата звернення 13.03.2021).
3. Куликов С. С. Тестирование программного обеспечения. Базовый курс. Минск : Четыре четверти, 2017. 312 с.

УДК 004.054

АВТОМАТИЗАЦІЯ ТЕСТУВАННЯ ВЕБ-ДОДАТКІВ

Нікітюк В.С., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

При написанні різноманітного типу програм завжди постає проблема тестування створеного програмного продукту. Це важливий етап, на який розробники витрачають велику кількість часу. Тому вони намагаються звести кількість ручного тестування до мінімуму та використовувати автоматизацію тестування.

Розглянемо приклад. Існує деякий веб-додаток, у якому для внесення нового функціоналу був змінений певний модуль всієї ієрархії. У такому разі доведеться перевіряти не лише нові функції додатку, а й сторінки (або окремо панелі, кнопки тощо), які залежали безпосередньо від зміненого модуля. Для цього, потрібно в коді знайти всі залежності та перевірити їх по черзі одну за одною. Що робити якщо присутня множинна зв’язність або потрібно провести оновлення головного пакету додатку, який повністю впливає на працездатність всієї програми? В такому разі потрібно провести повне тестування, перевірити кожну частину програми, а у великих системах це може займати не один день і навіть неділю. Саме в таких випадках на допомогу приходить автоматизація.

Автоматизоване тестування передбачає використання спеціального програмного забезпечення для контролю виконання тестів і порівняння очікуваного фактичного результату роботи програми. Цей тип тестування допомагає автоматизувати часто повторювані, але необхідні для максимізації тестового покриття завдання [1; 2].

Основу увагу буде приділено автоматизації так званого End-to-End (E2E) тестування, яке впливає на систему через її самі зовнішні інтерфейси та перевіряє очікувану реакцію системи через ці ж інтерфейси [3]. Існує декілька можливих фреймворків для автоматизації E2E тестування:

1. Selenium - це цілий набір інструментів, який дозволяє здійснювати браузерную автоматизацію. Відмінними рисами Selenium є можливість написання сценаріїв на JavaScript, C#, Java, Ruby, Python і підтримки більшості сучасних браузерів (Chrome, Firefox, Safari, Edge) [4].

2. Puppeteer - це open-source фреймворк, який надає високорівневий API для запуску, контролю та управління браузера - Chromium. На відміну від Selenium, комунікація відбувається безпосередньо з браузером.

3. Cypress - це open-source фреймфорк для E2E тестування, який є відносно молодим інструментом, проте він вносить нові концепції і рішення в здійснення автоматизації та тестування. Ключовою особливістю Cypress є те, що він виконується всередині самого браузера [5].

4. TestCafe - це кросбраузерності фреймворк для E2E тестування, який дозволяє працювати з Chrome, Safari, Mozila, IE та здійснює тестування за допомогою встановлення комунікації з сайтом за допомогою Proxy сервера.

Кожен із перерахованих фреймворків при використанні у автоматизованому тестуванні має ряд переваг та недоліків, але зупинимося на Cypress і розглянемо його більш детально. Його встановлення та конфігурування займає лічені хвилини. Є можливість створення тестів навіть без доступу до коду веб порталу, який потрібно тестувати (тобто будь-хто може створювати автоматизоване тестування для будь-якого проекту, до якого навіть немає доступу). Зручна та зрозуміла документація. Велика активна спільнота, яка готова допомогти із будь-якою проблемою протягом мінімально затраченого часу.

Для того, щоб почати працювати з Cypress достатньо ввести одну команду в кореневій папці проекту та зачекати кілька хвилин (перед цим потрібно перевірити наявність nodejs та npm на своєму комп'ютері). Її можна знайти в документації Cypress [5]. Встановлення не залежить від операційної системи, проекту, характеристик комп'ютера тощо. Після успішного встановлення можна починати написання тестів.

Для прикладу було створено сценарій, який перевіряє усім відомий пошуковий сайт Google на можливість збору та відображення інформації за певним пошуковим запитом (рис. 1). Всі команди в Cypress починаються із доступу до глобальної змінної 'cy'. Разом з кодом наведені коментарі для пояснення команд.

```
describe('Test Google', () => { // Опис сценарію
   it('Type search request', () => { // Опис тесту
      cy.visit('https://www.google.com/'); // перехід на головну сторінку
      cy.get('.gLFyf').type('Cypress e2e'); // Введення пошукового запиту в
                                            // текстове поле
      cy.contains('Пошук Google').click(); // Натиснути кнопку пошуку
      cy.contains('Приблизна кількість результатів').should('be.visible');
                                    // Перевірка відображення результату
   });
   it('Check searching algorithm', () => { // Опис тесту
      cy.contains('cypress.io').click(); // Перевірка присутності очікуваного
                                       // результату пошуку та перехід на сайт
      cy.contains('Install Cypress for Mac, Linux,
                   or Windows').should('be.visible');
                                    // Контент відповідного сайту відображений
   });
});
```

Рис. 1. Написання сценарію тестів для перевірки Google.com

Проходження створеного сценарію представимо на рис. 2. Зліва у вікні браузера знаходиться меню для відображення інформації про поточну дію, яку виконує фреймворк. Після проходження тестів за допомогою цієї панелі можна переміщатись між проходженням тестів, переглянути вигляд екрану до виконання команди та після, або подивитись більш розгорнуту інформацію про кожну із виконаних команд в консолі браузера.

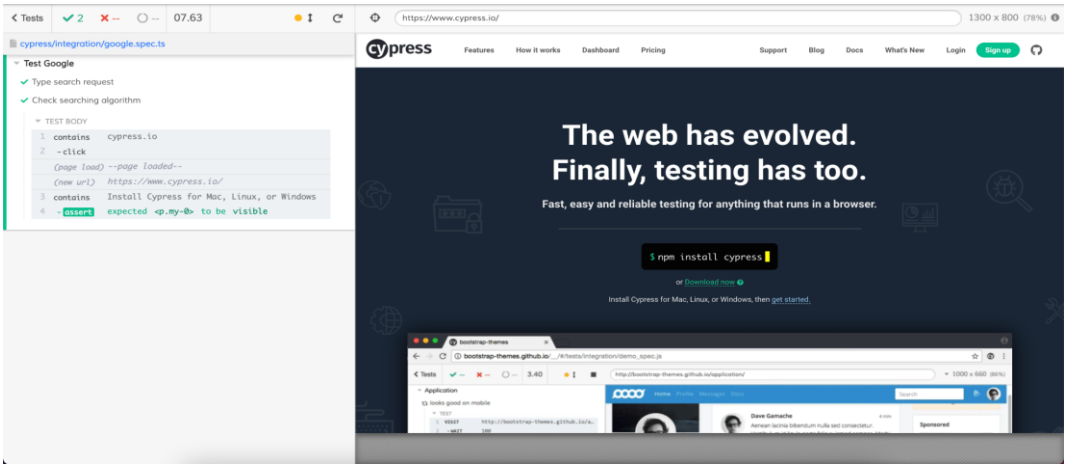


Рис. 2. Вигляд вікна після проходження тестів.

Звісно не слід забувати і про ручне тестування, адже жодна машина наразі не зможе замінити людину. Однак із використанням автоматизації часто повторюваних завдань із передбачуваною поведінкою можна зберегти багато часу та завдяки цьому зробити розробку додатків швидшою, стабільною та приємною.

Література:

1. Ручное и автоматизированное: веб-сайт. URL: https://qalight.ua/ru/baza-znaniy/ruchnoe-i-avtomatizirovannoe/ (дата звернення: 14.03.2021).
2. Що таке автоматизоване тестування програмного забезпечення? : веб-сайт. URL: https://devrepublik.com/uk/shho-take-avtomatizovane -testuvannya-programnogo-zabezpechennya/ (дата звернення: 14.03.2021).
3. E2E автоматизация веб-приложений 2k19: выбери свой инструмент! веб-сайт. URL: https://medium.com/@themillenialscrolls/e2e- автоматизация-веб-приложений-2k19-выбери-свой-инструмент-9624bbafc7e9 (дата звернення: 14.03.2021)
4. 12 лучших фреймворков автоматизированного тестирования PHP : веб-сайт. URL: https://www.internet-technologies.ru/articles/ 12-luchshih-freymvorkov-avtomatizirovannogo-testirovaniya-php.html (дата звернення: 13.03.2021).
5. Introduction to Cypress. веб-сайт. URL: https://docs.cypress.io/guides/ core-concepts/introduction-to-cypress.html (дата звернення: 14.03.2021)

УДК 004.054

ТЕСТ ДИЗАЙН ТА ТЕСТ-КЕЙСИ

Черненко Д.М., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

На сьогоднішній день зросла потреба у створенні комп’ютерних програм, додатків, веб-сайтів, це зумовлено тим, що все прагне до

комп’ютеризації, удосконалюються не тільки самі програми, а і пристрої (комп’ютери, телефони і ін.), тому виникає потреба не тільки в розробці програм під нові і старі пристрої, а і у їх тестуванні, без тестування ризиковано віддавати програмний продукт на використання, якщо у програмі виявиться безліч багів то програмним продуктом перестануть користуватися, що є невигідно для розробника.

Людина завжди намагається оточити себе якісними речами сюди відноситься і налагоджене програмне забезпечення. Якісний програмний продукт, це продукт який виконує поставлені перед ним завдання і задовольняє очікування користувачів. Для досягнення цього результату будь-яка програма спочатку проходить тестування і тільки потім потрапляє в руки кінцевого споживача. Так як терміни тестування мають тенденцію прагнути до нескінченності, нам необхідно грамотне вибудовування процесу і тут уже не обійтися без веб-дизайну. Для найбільш ефективного тестування і в подальшому виправлення багів використовують тест дизайн та тест-кейси.

Тест-дизайн – це процес тестування ПЗ (програмного забезпечення), відповідно до визначених критеріїв цілей тестування, проектуються і створюються тестові випадки, по іншому тест-кейси. Відповідно, тест-дизайнер - це співробітник, в чиї обов'язки входить створення набору тестових випадків, що забезпечують оптимальне тестове покриття додатки.[1]

Тест-кейси – один із засобів тестування програмного продукту. Цей засіб дозволяє провести перевірку продукту без ознайомлення з усією документацією. Це точний опис дій, які необхідно виконати, для того щоб перевірити роботу програми (поля для введення, кнопки і т.д.) [2]. Даний опис містить:

- дії, які треба виконати до початку перевірки - передумови;
- дії, які треба виконати для перевірки - кроки;
- опис того, що має статися, після виконання дій для перевірки - очікуваний результат. [3]

Проблема в тому, що за допомогою кейсів не можна виявити усі помилки, допущені при розробці, але можна скоротити їх кількість. Багато тестувальників на початку не розуміють навіщо взагалі потрібні тест-кейси, як правильно написати кейс і які елементи кейса є обов'язковими, чи треба використовувати певну методологію, який стандарт брати за основу, як швидко і наочно переконати замовника, що дійсно всі основні сценарії перевірені.

Тестування, засноване на кейсах, є більш об'єктивним, ніж дослідне тестування. Другою гідністю є чітке розуміння часу, яке треба на проведення тестування. [4]

Існує велика кількість методів і технік, які використовуються при тест-дизайні. Розглянемо найбільш поширені:

1. Тестування Класами Еквівалентності (Equivalence Class Testing). Тестові дані розбиваються на певні класи допустимих значень. В рамках кожного класу виконання тесту з будь-яким значенням тестових даних

призводить до еквівалентного результату. Після визначення класів необхідно виконати мінімум один тест в кожному класі.

2. Тестування Граничних Значний (Boundary Value Testing). Ця техніка заснована на тому факті, що одним з найслабших місць ПП (програмного продукту) є область граничних значень. Для початку вибираються діапазони значень - як правило, це класи еквівалентності. Далі визначаються межі діапазонів, на кожну з меж створюються три тест-кейса: для перевірки значення кордону, для перевірки значення нижче межі і для перевірки значення вище кордону.

3. Таблиця Прийняття Рішень (Decision Table Testing). Таблиці рішень - це зручний інструмент для фіксування вимог і опису функціональності додатку. Дуже зручно описувати бізнес-логіку програми таблицями і вони можуть служити відмінною основою для створення тест-кейсів.

4. Таблиці рішень (табл.1) описують логіку додатка, грунтуючись на умовах системи, що характеризують її стан. Кожна таблиця повинна описувати один стан системи (існує безліч сценаріїв, наприклад, перевірка логіну та паролю).

Таблиця 1.

Приклад «Таблиці Прийняття Рішень»

	Тест 1	Тест 2	…	Тест p
Умова				
Умова – 1				
Умова – 2				
…				
Умова – m				
Дія				
Дія – 1				
…				
Дія – n				

5. Метод Парного Тестування (Pairwise testing) - заснований на наступній ідеї: переважна більшість багів виявляється тестом, перевіряючим або один параметр, або поєднання двох. Помилки, причиною яких стали комбінації трьох і більше параметрів, як правило, значно менш критичні.

6. Сценарій використання (Use Case Testing). Use Case описує сценарій взаємодії двох і більше учасників (як правило - користувача і системи). Користувачем може виступати як людина, так і інша система. Для тестувальників Use Case є чудовою базою для формування тестових сценаріїв (тест-кейсів), так як вони описують, в якому контексті повинно проводитися кожну дію користувача. Use Case, за замовчуванням, є тестованими вимогами, так як в них завжди зазначена мета, якої потрібно досягти, і кроки, які треба для цього відтворити.

7. Вичерпне Тестування (Exhaustive Testing) – це перевіряються усі можливі комбінації вхідних значень, які повинні знайти всі проблеми. На практиці застосування цього методу не представляється можливим через величезну кількості вхідних значень. [5]

Далі розглянемо структуру написання самих тест-кейсів:

1. Заголовок: повинен бути чітким, стислим, зрозумілим і однозначно характеризувати суть тест-кейса, він не може містити виконувані кроки і очікуваний результат.

2. Передумова: може містити повну інформацію про стан системи або об'єкта, необхідному для початку виконання кроків тест-кейса, може містити посилання на інформаційні джерела, які необхідно вивчити перед проходженням тест-кейса (інструкції, опис систем ...), не може містити посилання на тестований ресурс.

3. Кроки перевірки: повинні бути чіткими, зрозумілими і послідовними, слід уникати зайвої деталізації кроків.

4. Очікуваний результат повинен бути у кожного кроку перевірки, має бути коротко і зрозуміло описано стан системи або об'єкта, що наступає після виконання відповідного кроку, не повинно бути надмірного опису.

5. Загальні вимоги до тест-кейсів: мова опису тест-кейсів повинна бути зрозуміла більшості користувачів, а не вузької групи осіб, тест-кейс повинен бути максимально незалежний від інших тест-кейсів і не посилатися на інші тест-кейси, тест-кейси групуються в функціональні блоки за призначенням; в тест-кейсах перевіряючих роботу функціонала скріншотів бути не повинно, інакше ви будете присвячувати сотні годин на зміну всіх скріншотів в тисячах тест-кейсах при зміні інтерфейсу програми, що тестується. Скріншоти можуть бути додані тільки в тест-кейси перевіряючі відображення сторінок і форм.[3]

Якщо дотримуватися цих правил, то тест-кейси будуть легко підтримуваними, легко читаютимуться, не викликатимуть відторгнення і можуть бути використані всіма учасниками команди в процесі розробки програмного забезпечення.

Розглянутий список далеко не повний і дає тільки загальне уявлення про принципи тестування і техніках тест-дизайну. Тестування, яке покриває всі можливі сценарії і виявляє всі помилки, існує тільки в теорії. Це пов'язано з тим, що перевірка всіх параметрів і станів займе дуже багато часу. Однак, чим більш досвідчений QA-фахівець, тим кращих результатів він може досягти, і для цього важливо вміти правильно підбирати і комбінувати техніки. Також важливо виконувати усі правила у роботі з тест-кейсами це допоможе швидше зрозуміти проблему і якісніше усунути різні баги.

Література:

1. Лекційні матеріали навчального курсу «software testing for universities». Провідна українська компанія з тестування програмного забезпечення QATestLab Ukrainian HI-Tech Initiative.

2. Авраменко В.С., Авраменко А.С., Косенюк Г.В. Тестування програмного забезпечення. Навчальний посібник. – Черкаси: ЧНУ імені Богдана Хмельницького, 2017. – 283

3. Джек Фолк, Сем Канер, Енг. Кек Нгуєн. Тестування програмного забезпечення. Видавництво ДиаСофт, 2001.
4. Рекс Блек. Ключові процеси тестування - М.: Видавництво Лорі, 2014. - 544 с.
5. Роман Савін. Тестування dot com. Видавництво «Дело» Москва 2007.

УДК 004.054

РОБОТА З БАЗОЮ ДАНИХ В ПРОЦЕСІ ТЕСТУВАННЯ

Мисюра Ю.О., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

В умовах цифрового прогресу неможливо зберігати інформацію у паперовому вигляді, на допомогу прийшли бази даних (БД), в яких масиви інформації зберігаються, оброблюються та структуруються. Оскільки БД є важливою складовою багатьох сучасних програмних продуктів, тому інформація, що зберігається в них, є дуже важливою і відмова або порушення цілісності даних можуть призвести до втрати великих об’ємів інформації, тому тестування баз даних є досить важливим. Для кожної системи, яка розроблюється, необхідно провести ряд тестів, щоб мінімізувати проблеми у використанні в майбутньому.

Тестування БД у залежності від функції та структури БД можна розділити на три типи: структурне тестування БД, функціональне тестування БД, нефункціональне тестування БД. Основними атрибутами тестування БД є: транзакції, схема БД, тригер, процедури [1].

При проведенні тестування БД умовно можна виділити такі види:

1. Тестування CRUD-методів (CRUD розшифровується як Create (створення), Read (вибірка), Update (оновлення), Delete (видалення)) полягає у перевірці збереження цілісності даних. Це означає, що не повинна бути можливість, наприклад, видалити запис у базі, на який за допомогою зовнішнього ключа посилаються інші записи. Таке тестування найкраще робити за допомогою SQL-запитів.

2. Тестування тригерів та вбудованих процедур. Сучасні системи керування базами даних дозволяють програмістам додавати код, що виконує різні додаткові функції [2]. Це відкриває багато можливостей, але й підвищує потенційну кількість багів.

3. Тестування швидкодії та навантаження є необхідним при розробленні веб-сайтів та інших програмних продуктів з високим навантаженням на сервер. Його метою є перевірка швидкодії та адекватної віддачі при різних рівнях навантаження на сервер БД [3].

4. Інтеграційне тестування полягає у перевірці взаємодії серверної частини додатку з БД, а саме перевірки конфігурації сервера (дані для авторизації, адреса і порт доступу до БД, кодування даних, наявність

шифрування, часовий пояс тощо), а також правильності роботи HTTP-запитів – GET, POST, PUT і DELETE [4].

Розглянемо більш детально використання тестування CRUD-методів на прикладі бази даних поданої на рис. 1.

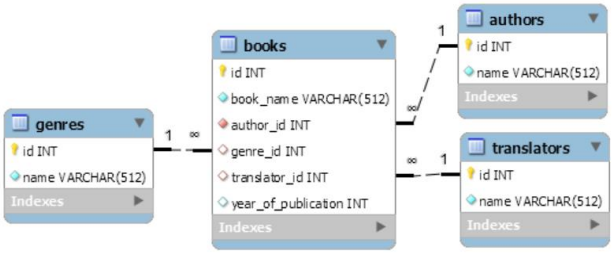


Рис. 1. Діаграма бази даних.

Далі необхідно протестувати CRUD-методи для чотирьох таблиць як це показано на рис. 2-6.

```
select * from books
```

id	book_name	author_id	genre_id	translator_id	year_of_publication
26	Нейромант	1	1	NULL	1984
45	Володар перснів	4	8	6	1950
46	Сильмариліон	4	8	6	1977
NULL	NULL	NULL	NULL	NULL	NULL

Рис. 2. Результати вибірки для таблиці books.

```
insert into books (book_name, author_id, genre_id)
values ('book1', 1, 1);
select * from books
```

id	book_name	author_id	genre_id	translator_id	year_of_publication
26	Нейромант	1	1	NULL	1984
45	Володар перснів	4	8	6	1950
46	Сильмариліон	4	8	6	1977
49	book1	1	1	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL

Рис. 3. Результат вставки у таблицю books.

```
update books
set book_name = 'book2'
where id = 49;
select * from books
```

id	book_name	author_id	genre_id	translator_id	year_of_publication
26	Нейромант	1	1	NULL	1984
45	Володар перснів	4	8	6	1950
46	Сильмариліон	4	8	6	1977
49	book2	1	1	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL

Рис. 4. Результат оновлення запису у таблиці books.

```
delete from books
where id = 49;
select * from books
```

id	book_name	author_id	genre_id	translator_id	year_of_publication
26	Нейромант	1	1	NULL	1984
45	Володар перснів	4	8	6	1950
46	Сильмариліон	4	8	6	1977
NULL	NULL	NULL	NULL	NULL	NULL

Рис. 5. Результат видалення запису з таблиці books.

CRUD-методи для таблиці books працюють належним чином. Таблиці authors, translators і genres перевіряються аналогічним чином. Застосуємо операцію видалення (рис. 6).

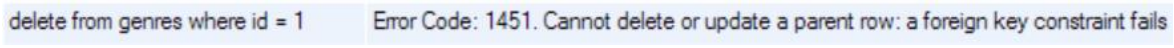


Рис. 6 Результат видалення запису з таблиці genres.

Як видно з рис. 6, не вдалося видалити цей запис з таблиці genres, оскільки на нього посилаються за допомогою зовнішнього ключа інші записи.

Отже, було розглянуто основні види тестування БД та на прикладі наведено застосування CRUD-методу. Тестування БД дозволяє мінімізувати ризики, які пов'язані з цілісністю даних, здатністю системи реагувати на навантаження та стрес, до введення продукту в використання.

Література:

1. Тестирование баз данных: на что обратить внимание? : веб-сайт. URL: https://training.qatestlab.com/blog/technical-articles/database-testing-what-to-look-for/ (дата звернення: 21.04.2021).
2. Глава 6. Триггеры : веб-сайт. URL: http://www.rldp.ru/mysql/mysqlpro/triggers.htm (дата звернення: 21.03.2021).

УДК 004.415.25

СТАНДАРТИ POSIX В СИСТЕМНОМУ ПРОГРАМУВАННІ

Стеценко А.П., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

Наявність стандартів допомагає комп'ютерам спілкуватися між собою та інтерпретувати дані в однаковому форматі незалежно від платформи та місцезнаходження. Раніше, коли не існувало загальних стандартів для розробки комп'ютерної операційної системи, для досягнення сумісності розробникам доводилося індивідуально розробляти застосунок для кожної моделі пристрою з операційною системою. Що в рази збільшувало час розробки та вартість програм. Через появу помилок в кожній новій моделі комп'ютера сам процес відлагодження також проходив складно. Тому міжнародна організація IEEE ввела стандарти для підтримки сумісності між операційними системами (ОС), які дозволили уникнути написання коду для різних моделей системи, але допомогли пришвидшити розробку мультиплатформних додатків. Таким стандартом став POSIX (англ. Portable Operating System Interface for uniX) – мобільний інтерфейс ОС [1].

У сфері програмування інтерфейс означає місце, в якому програмні коди можуть взаємодіяти між собою використовуючи чітко описані формати

даних. Інтерфейс очікує, що при взаємодії з кодом, можуть надійти параметри, де у відповідь на виклик, інтерфейс зазвичай повертає результат певного типу [1-2].

У контексті стандартів POSIX, терміни «інтерфейс» та «портативний» нерозривно пов'язані. Поняття портативності у POSIX стосується саме програмного вихідного коду, а не скомпільованого двійкового коду. Код написаний з дотриманням POSIX вважається платформонезалежним, тому що його можна компілювати для будь-якої ОС сумісної з POSIX.

Теоретично, вихідний код з підтримкою стандартів POSIX, можна безперешкодно переносити на велику кількість ОС. Однак при перенесенні додатків є ризик виникнення специфічних системних проблем.

Стандарт POSIX описує поведінку комп'ютерної системи через певний набір типів даних, функціональних інтерфейсів та системних утиліт. Стандарт описує інтерфейс, а не реалізацію. Головною метою стандарту було перенесення прикладних програм в системних середовищах UNIX за допомогою їх документації та чіткого плану специфікації інтерфейсу операційної системи [3].

Технічні характеристики стандарту POSIX не стосуються того, як слід організувати процеси розробки ОС або програми, основна увага приділяється взаємозв'язку між ними.

Спочатку стандартні правила були розбиті на чотири основні категорії [4]: POSIX.1, POSIX.1b, POSIX.1c та POSIX.2.

Стандарти POSIX.1 описують основні служби усіх моделей ОС, до їх основних функцій можна віднести:

- створення та контроль процесу;
- тригери процесу;
- операції з файлами і каталогами;
- помилки сегментації;
- помилки пам'яті;
- помилки при обчисленнях з плаваючою комою;
- міжпроцесорні конвеєри;
- сигнали;
- реалізація стандартної бібліотеки С;
- стандартний інтерфейс введення / виведення й управління.

Крім основних функцій описаних стандартом POSIX.1, існують додаткові функції, спеціально для розробки додатків у реальному часі. Список деяких механізмів описаних в POSIX.1b:

- алгоритми планування процесора;
- передача повідомлення;
- спільна пам'ять;
- семафор;
- інтерфейси блокування пам'яті;
- інтерфейси синхронної та асинхронної передачі даних.

Стандартна категорія POSIX.1c описує функції пов’язані з багатопотоковістю:

- створення потоку;
- контроль потоку;
- видалення потоку;
- синхронізація потоків;
- планування потоків.

Стандарти POSIX.2 стосуються основних функціональних можливостей ОС. Перелік включає всі загальні утиліти та загальноприйняті інструменти, які використовуються користувачами: uname, tty, cd, ls, mkdir, echo, cp, rm та mv. Згодом, для опису ще чотирьох функціональних областей, було додано стандарт POSIX.1-2008:

- Base Definition Volume (загальні терміни, концепції та інтерфейси);
- Systems Interfaces Volume (визначення функцій й підпрограм системних служб. Також включає портативність, обробку помилок і відновлення роботи додатків після помилок);
- Shell and Utilities Volume (визначення інтерфейсів усіх програм до командних оболонок і загальних службових програм);
- Rationale Volume (містить інформацію та історію про виконані функції, а також обґрунтування рішень).

Стандарт POSIX.1-2008 не описує графічні інтерфейси, інтерфейси баз даних, портативність об’єктного або двійкового коду, конфігурації системи, операції введення-виведення або механізми доступу до ресурсів [5].

Специфікації POSIX задають стандартні механізми взаємодії додатків та ОС. Тому відповідність стандарту POSIX для ОС та апаратної платформи повинно бути сертифіковано за допомогою прогону на них тестових наборів [POSIXTestSuit], які існують лише для POSIX 1003.1. Якщо ОС не є Unix-подібною, то поставщики ОС можуть реалізувати лише частину стандартного інтерфейсу. В ОС родини Windows стандарти POSIX використовуються в підсистемах зі стандартними функціями. На відміну від macOS, яка використовує стандарти POSIX, більшість дистрибутивів Linux використовують окрему базу стандартів Linux Standard Base (LSB), яка собою представляє надмножину стандартів POSIX, тобто незалежний перелік стандартів з розширеним набором функцій.

Для виконання основних системних дій за допомогою певної мови програмування потрібно перевірити, чи має вона стандартну бібліотеку, яка відповідає за взаємодію з ОС. Більшість стандартних бібліотек, які забезпечують підтримку основних функціональних можливостей системи, написані на мові С. Оскільки в POSIX стандартною мовою є С, то розробники можуть вільно обирати будь-яку мову та впроваджувити для неї стандарт. Технічні характеристики в стандарті POSIX є короткими, але вони мають достатній рівень абстракції для розробки різноманітних систем та програмного забезпечення.

Таким чином, програми, написані із дотриманням цих стандартів, будуть однаково виконуватися на всіх POSIX-сумісних системах і не будуть обмежувати вибір постачальників ОС по вибору архітектури системи та способів її реалізації, а також будуть характеризуватися високою економічною ефективністю.

Література:

1. POSIX - Что такое интерфейс переносимой операционной системы? : веб-сайт. URL: https://fossbytes.com/posix-what-is-the-portable-operating-system-interface (дата звернення: 25.04.2021).
2. Стандарты, унифицирующие интерфейсы прикладных систем (POSIX) : веб-сайт. URL: http://emag.iis.ru/arc/infosoc/emag.nsf/BPA/7aaebe3fb4d9b9c7c32575c9004211b
3.
3. CMPSC 311, The Posix Standard : веб-сайт. URL: www.cse.psu.edu/~deh25/cmpsc311/Lectures/Standards/PosixStandard.html (дата звернення: 25.04.2021).
4. Practical System Programming with C: Pragmatic Example Applications in Linux and Unix-Based Operating Systems [Текст] / Sri Manikanta Palakollu. 2021. С. 65-68 (дата звернення: 25.04.2021).
5. Posix Standard – Linux Hint : веб-сайт. URL: https://linuxhint.com/posix-standard (дата звернення: 25.04.2021).

УДК 004.652

ELASTICSEARCH В ЯКОСТІ NOSQL БАЗИ ДАНИХ

Стеценко А.П., Розломій І.О.

Черкаський національний університет ім. Богдана Хмельницького

Elasticsearch – повнотекстова розподілена NoSQL база даних [1]. В цій базі даних, підхід NoSQL означає те, що замість використання реляційних схем та таблиць, всі вхідні дані зберігаються як документи формату JSON.

Elasticsearch спроектована так, щоб можна було створювати розподілені системи, тобто одночасно використовувати декілька вузлів всередині кластера. Кожного разу, при запуску екземпляра Elasticsearch, фактично запускається один вузол [2]. Кластером називають набір підключених між собою вузлів. Таким чином збережені документи розподіляються по кластеру і до них можна негайно отримати доступ з будь-якого вузла [3].

Розподіленість, дозволяє масштабувати систему, забезпечити більшу надійність та швидкість роботи. Перевагою Elasticsearch над більшістю реляційних БД є закладена в архітектуру системи, підтримка механізмів розподілення вузлів. Оскільки логіка розподілення присутня, інтеграція

Elasticsearch з прикладними додатками відбувається за оптимальну кількість змін всередині додатка.

Серед всіх вузлів присутніх у кластері обирається один головний, який відповідає за управління загальнокластерними змінами, такими як створення та видалення індексу, або додавання й видалення вузла з кластера. Головний вузол не повинен брати участь у змінах або пошуку на рівні документів. Отже, зростання трафіку не створює вузьких місць навіть при наявності лише одного головного вузла в кластері. Будь-який вузол може стати головним. Користувач може взаємодіяти з будь-яким вузлом кластера, включаючи головний вузол. Кожен вузол знає, де розташований кожен документ, і може переслати запит безпосередньо вузлу, що містить дані, які цікавлять. Elasticsearch прозоро управляє всіма процесами пошук інформації. Тобто, при взаємодії з будь-яким вузлом кластера, він проводить збір даних, отримання та повернення остаточних результатів клієнту.

При невдалому проектуванні, можуть виникати конфлікти між незалежними вузлами та кластерами, які вважають себе головними й паралельно вносять змін до БД, що призводить до безповоротної втрати даних. Хоча рівень складності розподілених систем вважається їх недоліком, Elasticsearch намагається приховати цю складність. До операцій, які є прихованими від користувача і відбуваються автоматично всередині БД відносяться наступні операції:

- розбиття документів на різні контейнери або фрагменти, які можна зберігати в одному або декількох вузлах;
- копіювання кожного фрагмента БД для забезпечення надмірними копіями ваших даних, щоб запобігти втраті даних у разі несправності обладнання;
- маршрутизація запитів від будь-якого вузла кластера до вузлів, що містять дані, які вас цікавлять;
- плавна інтеграція нових вузлів в наявний кластер або перерозподіл фрагментів для відновлення після втрати вузла.

Масштабування БД може відбуватися як вертикально, так і горизонтально. При вертикальному масштабуванні чинні сервери оснащують більш потужним обладнанням. При горизонтальному, до існуючих серверів додають нові. Через те, що вертикальне масштабування має ряд фізичних обмежень, для досягнення вагомого пришвидшення роботи рекомендовано використовувати горизонтальне.

Варто зазначити, що на відміну від SQL систем, котрі підтримують стандартні механізми транзакцій, Elasticsearch використовує функцію «write-ahead-log». Ця функція забезпечує надійність проведення операцій без використання більш повільної та важкої «commit», однак провести операцію «rollback», тобто повернути документ до попереднього стану – неможливо. Відсутність звичних транзакцій, обумовлена тим, що вони ускладнюють та уповільнюють виконання запитів до бази даних. Такий підхід дозволяє досягати максимальної ефективності запитів та зберігати велику частину трохи застарілих даних в кеш. Час відгуку в одну секунду вважається

достатнім для комфортної роботи, сучасні користувачі очікують, що додаток відповідатиме «миттєво». Таку швидкість можна приблизно оцінити в 0,1 секунди, враховуючи затримки в мережі, котрі також можуть сильно варіюватися [4]. Цього часу не завжди достатньо навіть для бази даних з вбудованим пошуковим двигуном. Щоб розв'язати відповідну проблему, потрібно якомога більше інформації зберігати в пам'яті.

Elasticsearch витрачає багато ресурсів на створення та постійне обслуговування різноманітних даних в кеш-пам'яті: частини структур індексу, що часто використовуються, такі як словники термінів, списки публікацій, знаходяться переважно в кеші сторінок операційної системи; значення полів, які використовуються для сортування або всередині скриптів, завантажуються в кеш поля; фільтри можна кешувати як растрові зображення «bitmap» в кеші фільтрів, це робить подальше їх використання неймовірно швидким.

Elasticsearch орієнтований на документи, це означає, що вся множина об'єктів, між котрими буде проводитись пошук, повинна бути проіндексована [1]. Перед індексацією документи мають бути денормалізовані. Денормалізація збільшує швидкість зчитування та зменшує швидкість запису даних через збереження надлишкових копій й групування вже наявної інформації. Розміщення збиткової інформації вимагає більшого дискового простору, але консистентність даних та їх актуальність забезпечуються. Отже, це ідеальний варіант, коли дані необхідно зберегти один раз, а потім багаторазово їх зчитувати.

Elasticsearch не підтримує функції аутентифікації та авторизації користувача. Більш того, відсутня можливість обмежити доступу до таблиць, операцій та індексів бази даних [4]. Отже, розробникам потрібно налаштовувати аутентифікацію та авторизацію на рівні додатку. Також, слід обмежити доступ до бази даних напряму з інтернету й прибрати можливість відправлення довільних SQL запитів невідомими користувачами. Можна зробити висновок, що Elasticsearch має велику кількість вразливостей, якими можуть скористатися зловмисники для заволодіння інформацією. Більш того, без обмежень доступу, будь-які користувачі мають право на запити будь-якого рівня складності. Вони можуть призводити до перевантаження системи і пошкодження даних. Elasticsearch часто застосовується для повнотекстового пошуку, так як спеціалізується в цій сфері й пропонує різноманітні функції та результати. Задача повнотекстового пошуку передбачає більше, ніж просте знаходження ключових слів у фрагменті тексту. Для реалізації точних моделей релевантності, огляду всього масиву результатів та виконання таких дій, як перевірка орфографії та автозаповнення в Elasticsearch застосовуються знання, характерні для доменної області [1].

Кожне поле в документі проіндексоване і може бути присутнє в запиті. Під час одного запиту Elasticsearch може використовувати всі наявні індекси для повернення результатів з максимальною швидкістю. Пошук можна провести за будь-якими відсортованими атрибутами документу. При

повнотекстовому запиті, всі документи, що відповідають ключовим словам пошуку, повертаються у відсортованому за релевантністю вигляді.

Зазвичай, Elasticsearch використовується як додаткова БД для швидкого пошуку інформації, поряд з основною базою, яка є більш надійною, захищеною, забезпечує оновлення через механізм транзакцій та містить більш жорсткі обмеження. Якщо спочатку зберігати інформацію в основну базу даних, а вже потім асинхронно записувати її в Elasticsearch, то можна мінімізувати випадки безповоротної втрати. Хоч інформація для пошуку буде застаріла на деякий час, її можна буде синхронізувати з актуальною, без втрати швидкості пошуку та непомітно для кінцевих користувачів.

Література:

1. Elasticearch as NoSQL Database [Електронний ресурс]. – Режим доступу до ресурсу: https://www.elastic.co/blog/found-elasticsearch-as-nosql
2. Григорьев Ю.А., Плутенко А.Д., Плужникова О.Ю. (2018) Реляционные базы данных и системы NoSQL. Амурский гос. ун-т, МГТУ им. Н. Э. Баумана. Благовещенск: Изд-во Амурского гос. ун-та, 420.
3. Пьянзин С. А., Григорьев Ю.А., Щеглов Д. С. (2020) Корпоративное решение с использованием поисковой системы Elasticsearch. Научно-образовательный журнал для студентов и преподавателей «StudNet», 6, 663 – 672.
4. Barakhnin V.B., Kozhemyakina O. Yu., Mukhamediev R. I., Borzilova Yu. S., Yakunin K. O. (2019) The design of the structure of the software system for processing text document corpus. Business Informatics, 4(13), 60 – 72.

УДК 004.93'14

ІНФОРМАЦІЙНА СИСТЕМА АНАЛІЗУ ГЕОДАНИХ ДЛЯ ВІДСЛІДКОВУВАННЯ ЗМІН РОСЛИННОСТІ

Бандурка О.І., Барабаш О.В.

Національний Технічний Університет України
«Київський Політехнічний Інститут ім. Ігоря Сікорського»

Аварія на Чорнобильській АЕС за своїми масштабами і наслідками не має аналогів у світі. Багаточисельні радіоактивні викиди в довкілля спричинили до незворотніх наслідків, в тому числі і до порушень в екосистемах [1].

Метою дослідження є моніторинг стану рослинності пошкодженої радіонуклідами зони на основі використання супутникових знімків із застосуванням нормалізованого вегетаційного індексу (NDVI) та методу кластеризації к-середніх.

Дані ДЗЗ із космосу дозволяють вирішувати різноманітні задачі, які мають специфічні вимоги до властивостей зображень та систем запису

(частота та регулярність зйомки, висока просторова і радіометрична роздільна здатність).

Детальний аналіз характеристик зображень, що надійшли від датчиків, правильний підбір програмного забезпечення та відповідність вихідних даних поставленій задачі є основою обробки просторових зображень та отримання необхідних кінцевих даних.

За основу досліджень взяті знімки Landsat, які мають велику базу архівних даних , програмний модуль створено на основі мови Python.

Найпоширенішим вегетаційним індексом є нормалізований індекс відмінностей рослинного покриву.

NDVI – нормалізований диференційний показник вегетації – кількісний показник фотосинтетичних властивостей активної біомаси. NDVI часто називають показником вегетації.

NDVI – це один із найпоширеніших та найбільш часто використовуваних показників для вирішення задачі кількісної оцінки рослинності.

Обчислюється NDVI за формулою:

$$NDVI = \frac{NIR - RED}{NIR + RED}.$$

де NIR – амплітуда відбиття в ближній інфрачервоній області спектра, RED – амплітуда відбиття в червоній області спектра.

Відповідно до формули, щільність рослинності (NDVI) у певний момент зображення дорівнює різниці між інтенсивністю відбитого світла в інфрачервоній та червоній зонах, поділеною на суму їх інтенсивності. Розрахунок NDVI базується на двох найбільш стабільних областях (незалежно від інших факторів) спектральної кривої відбиття судинних рослин. Максимальне поглинання сонячної радіації хлорофілом вищих судинних рослин знаходиться в червоній зоні спектру (0,6 - 0,7 мкм), а область максимального відбиття структур клітин знаходиться в інфрачервоній зоні (0,7 - 1,0 мкм).

В даний час існує два підходи до реалізації автоматичної класифікації: контрольована класифікація та некерована (кластеризація). Керована класифікація аналізує пікселі кожного еталонного полігона і створює спектральні сигнатури для кожного типу покриття. Зображення класифікується шляхом порівняння спектральних значень пікселів із створеними легендами.

Класифікація за методом мінімальної відстані полягає у обчисленні евклідової відстані значень відбиття пікселів від середнього спектрального значення кожної сигнатури. Піксель присвоюється класу з найменшою відстанню. Класифікація за методом максимальної вірогідності вважається однією з найкращих, оскільки вона заснована на імовірнісних принципах.

Завдяки кластерному аналізу можна виділяти контури з неконтрастною по спектральній яскравості структурою, наприклад, рослинність, відкритий ґрунт, вода, хмари та інші об’єкти. Використовуючи алгоритми кластеризації,

зображення автоматично поділяється на групи пікселів із подібними спектральними властивостями (кластери). Ці алгоритми вимагають мінімум початкової інформації (кількість класів та ітерацій) [2].

Кластеризація зображень за алгоритмом ISODATA ґрунтується на різниці середніх значень кластера (мінімальна спектральна відстань між класовими центрами) [3].

При підготовці до класифікації матеріалів космічних знімків, які використовуються в даному дослідженні, був проведений аналіз рослинності для виявлення змін в її структурі, а також для відокремлення рослинності від інших об’єктів та встановлення пікселів відкритого ґрунту. За результатами проведеної роботи створено тематичні зображення вегетаційного індексу.

Індекс вегетації використовує властивості червоного (який поглинає рослинність) та поблизу інфрачервоних смуг (що сильно відображає рослинність). Як випливає з назви, ми використовуємо її для моніторингу здоров'я та енергії рослинності. Класифікація знімків за розрахованими індексами вегетації відображена на рисунку 1.

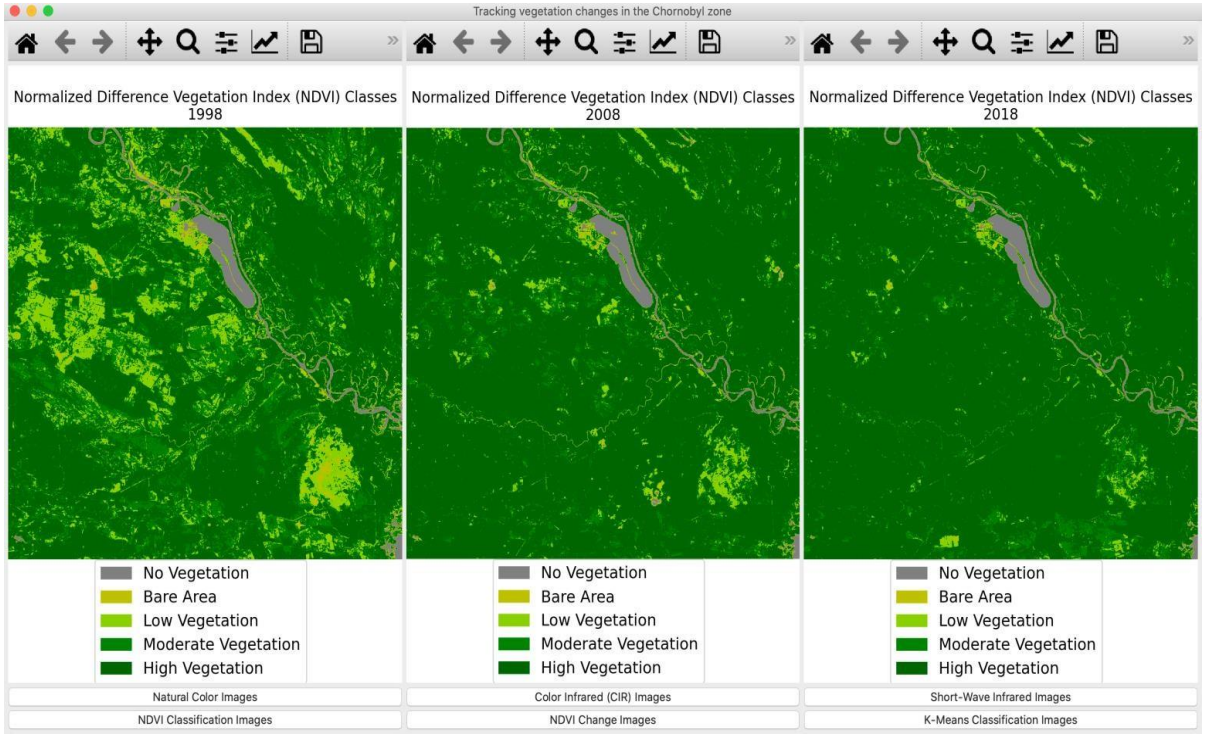


Рис. 1. Класифікація знімків за розрахованими індексами вегетації.

На основі проведених досліджень було розроблено методику комплексної автоматизованої оцінки стану рослинного покриття території зони відчуження з використанням космічних знімків. Розглянуті на прикладі Чорнобильської зони зазначені підходи довели свою ефективність для аналізу динаміки стану рослинних угруповань в часі та дозволили виявити загальні тенденції позитивної зміни рослинного покриву на досліджуваній території.

Література:

1. 20 років Чорнобильської катастрофи. Погляд у майбутнє: Національна доповідь України [Електронний ресурс] – К.: Атіка, 2006. – 224 с.
2. И.А. Зубков, В.О. Скрипачев. Применение алгоритмов неконтролируемой классификации при обработке данных ДЗЗ. ФГУП «Научный центр космических информационныхсистем и технологий наблюдения». Москва 2007.
3. Руководство пользователя. Программа обработки данных дистанционного зондирования Земли ScanEx Image Processor v. 3.0. Москва, 2008.

УДК 519.686.4

СТАТИЧНИЙ АНАЛІЗ КОДУ C# З ВИКОРИСТАННЯМ ROSLYN API

Білоус І.В., Нагорний П.В.

Національний університет «Чернігівська політехніка»

Важливим етапом розробки програмного забезпечення є його тестування та налагодження. Існує два основних підходи до пошуку помилок у написаному коді: статичний та динамічний аналіз. Кожен з підходів має свої переваги та недоліки. В контексті даного дослідження розглянемо саме статичний аналіз, який передбачає пошук помилок коду без його виконання.

Для статичного аналізу коду C# існує значна кількість рішень, таких як PVS-Studio. Але таким програмним продуктам властива досить значна кількість помилок 2-х типів: хибні спрацьовування та ігнорування справжніх помилок в коді. Крім того, в окремих ситуаціях весь функціонал статичних аналізаторів не є необхідним – можна обмежитися лише окремими невеликими утилітами [1].

За таких умов корисним є дослідження спеціалізованого Roslyn API як засобу для розробки статичних аналізаторів різних рівнів складності. Roslyn API передбачає можливість розробки утиліт статичного аналізу коду C# чи VB. На базі цього API створена вже зазначена PVS-Studio.

Основними поняттями Roslyn API є синтаксичне дерево та семантична модель [2]. Синтаксичне дерево – спеціальна структура (дерево) для інтерпретації певного коду, в якій вузлами позначаються оператори, а листками – операнди. Аналіз синтаксичного дерева дозволяє знаходити помилкові конструкції в коді. Але для більш якісного аналізу слід мати можливість отримувати інформацію про об’єкти досліджуваної програми, представлені в моделі синтаксичного дерева. З цією метою запропоновано використовувати семантичну модель, яка дозволяє отримати інформацію не тільки про самі об’єкти, а й про їх типи.

Для отримання синтаксичного дерева певного файлу з кодом слід, в першу чергу, під’єднати проект. Це можна виконати за наступною послідовністю дій [2]:

- створити робочий простір викликом MSBuildWorkspace.Create();
- відкрити поточне рішення викликом workspace.OpenSolutionAsync (solution Path).Result;
- отримати список проектів викликом currentSolution.Projects;
- серед переліку IEnumerable<Project> обрати необхідний.

Перелік файлів проекту наведено у project.Documents. Після вибору необхідного файлу його синтаксичне дерево можна отримати з використанням виклику file.GetSyntaxTreeAsync().Result. Для отримання семантичної моделі слід також виконати компіляцію проекту викликом project. GetCompilationAsync().Result, а після цього, з використанням вже отриманого синтаксичного дерева, отримати семантичну модель за допомогою виклику compilation.GetSemanticModel(tree) [3].

В синтаксичному дереві Roslyn API можна виокремити три основні елементи: Syntax nodes, Syntax tokens, Syntax trivia. Syntax nodes представляють собою синтаксичні вузли, які можуть бути надалі аналізовані більш детально (об’явлення, оператори, вирази тощо). Ці елементи синтаксичного дерева зберігають інформацію батьківський вузол та дочірні вузли, елементи Syntax trivia тощо. Syntax tokens, на противагу Syntax nodes, зберігають інформацію про лексеми коду, тобто елементи коду, які не підлягають більш детальному розбору (ідентифікатори, ключові слова тощо). Syntax tokens зберігають інформацію про текстове представлення, числове значення (для літералів), оточуючі елементи Syntax trivia. Syntax trivia описують додаткову синтаксичну інформацію, яка не буде скомпільована (пробіли, коментарі тощо). Робота з Syntax trivia найчастіше зводить до визначення їх типу та, інколи, перегляду тексту [3].

Наводимо приклад фрагменту синтаксичного дерева для виразу x *= (y+10).

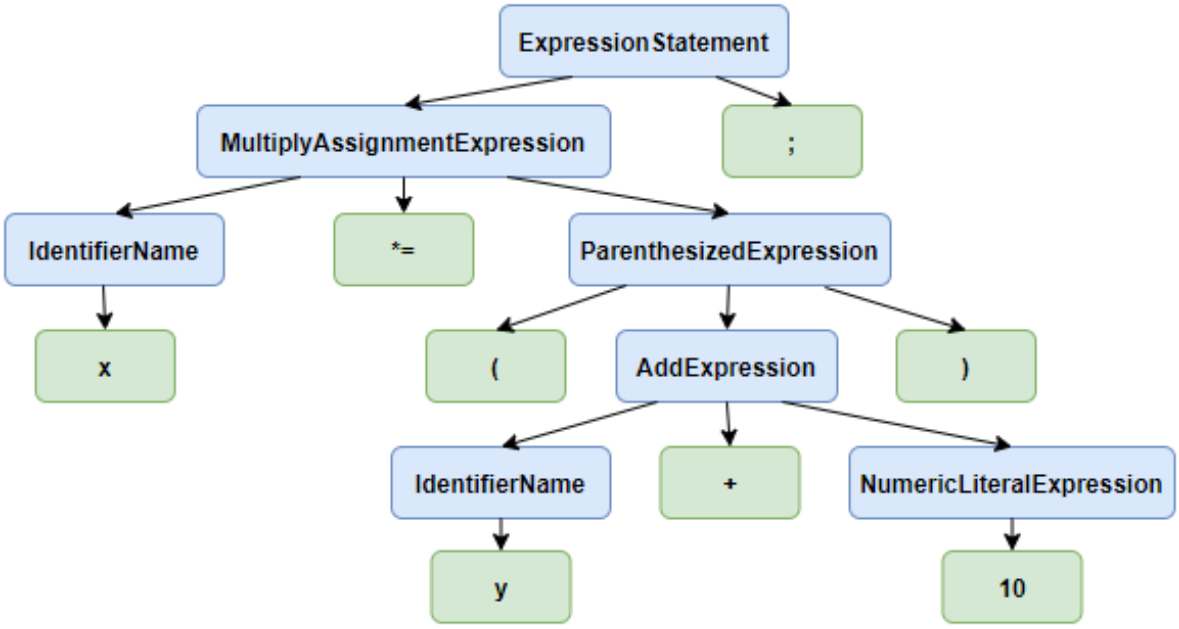


Рис. 1. Приклад фрагменту синтаксичного дерева.

Розглянемо також семантичну модель. Вона дозволяє отримувати інформацію про об'єкти, їх типи, та їх константні значення. Інформацію про об'єкти представляють символи (symbols). Необхідні методи для роботи з символами задекларовані в базовому інтерфейсі ISymbol та похідних IFieldSymbol, IMethodSymbol тощо. Інформацію про тип об'єкта представляють спеціалізовані символи, які реалізують інтерфейс ITypeSymbol. Константні значення можна отримати за допомогою методу GetConstantValue() [3].

Існує можливість візуалізувати синтаксичне дерево. Для цього існує розширення для середовища Visual Studio, яке входить в комплект Roslyn SDK – Syntax visualizer. Для кожного вузла синтаксичного дерева Syntax visualizer дозволяє отримати інтерфейси ISymbol та ITypeSymbol. Застосування Syntax visualizer є доречним для простішого розуміння структури коду [4].

Наведемо приклад аналізу семантичної моделі – метод перевірки чи є поле константним.

```
Boolean IsFieldConst(SemanticModel mod, IdentifierNameSyntax ident)
{
  ISymbol symbol = mod.GetSymbolInfo(ident).Symbol;
  if (symbol == null)
    return null;
  return symbol.Kind == SymbolKind.Field && (symbol as IFieldSymbol).IsConst;
}
```

Roslyn має ряд переваг та недоліків. До переваг можна віднести: велику кількість типів вузлів дерева, що дозволяє виконувати більш якісний та глибокий аналіз; зручну навігацію по дереву; наявність концепції семантичної моделі для отримання інформації про об'єкти та їх типи; відкритий код Roslyn API, що спрощує його модифікацію. До недоліків Roslyn API можна віднести: можливість проблем з відкриттям певних проектів, файлів; відсутність окремих реалізацій корисних опцій (наприклад, метод Equals для вузлів); високі вимоги до пам'яті технічних засобів, де відбувається запуск застосунків, написаних на базі Roslyn API.

Таким чином, Roslyn API надає необхідний функціонал для розробки застосунків статичного аналізу коду і може бути ефективно застосований під час аналізу програм, написаних на мові програмування C#.

Література:

1. Лучкова А. В. Аналіз методів верифікації програмного забезпечення. URL: https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/10955/328.pdf?sequence=3&isAllowed=y (дата звернення: 24.04.2021)
2. Vasiliev, S. Introduction to Roslyn and its use in program development. URL: https://pvs-studio.com/en/blog/posts/csharp/0399/ (дата звернення: 25.04.2021)

3. Панченко В. Roslyn: мастерство статического анализа. URL: https://assets.ctfassets.net/9n3x4rtjlya6/5NUCLWWohm5Lj2kODAzYyS/c10dcd2e75df10acf3042c6645473b7a/100745_1750911309_Vladimir_Panchenko_Roslyn_Masterstvo_staticheskogo_analiza.pdf (дата звернення: 26.04.2021)
4. Gordon, S. Getting started with the Roslyn APIs: writing code with code. URL: https://www.stevejgordon.co.uk/getting-started-with-the-roslyn-apis-writing-code-with-code (дата звернення: 27.04.2021)

УДК 004.65

УДОСКОНАЛЕНИЙ МЕТОД ОБРОБКИ ЗАПИТІВ В РОЗПОДІЛЕНИХ БАЗАХ ДАНИХ НА ОСНОВІ МЕХАНІЗМУ КЕШУВАННЯ ІНДЕКСІВ

Барабаш А.О., Корнага Я.І., Свинчук О.В.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Загальні і прикладні проблеми створення або інтеграції систем управління базами даних (СУБД) полягають у тому, що процедури розробки та інтеграції гетерогенних баз даних (БД) характеризуються виникненням проблем синхронізації даних і проблем контролю обробки даних. При інтеграції різнорідних баз даних у виробничий процес різних підприємств і установ, а також органів державної влади, виділяється ряд недоліків щодо організації сучасних систем управління та моніторингу подій в базах даних [1,2]. Особливістю таких систем є відсутність єдиного механізму і політики визначення загроз для всієї розподіленої комп'ютерної системи.

При реалізації моніторингу подій в розподілених базах даних зростає час пошуку інформації, що викликано додатковими витратами ресурсів при обробці запитів в БД. Зокрема, при використанні дерев з недостатнім наповненням вузлів, час пошуку збільшується відповідно до «висоти» дерева [3]. Існуючі підходи до формування і використання індексів в сучасних розподілених базах даних мають певні недоліки, а саме: недостатнє наповнення вузла дерева пошуку; використання однакових індексів для різних таблиць; надмірне завантаження в оперативну пам'ять (ОП) СУБД індексів, що мало використовуються [4].

Існуючі механізми модифікації даних в таблицях мають певні особливості: виконується зміна ключів у відповідних вузлах дерева з подальшою розбудовою індексу, що значно впливає на швидкість запису в БД, та особливо важливо при істотному збільшенні числа запитів записи (модифікацій) до бази даних.

При інтенсивному пошуку даних і використанні одних і тих же індексів доцільно зберігати найбільш часто використовувані індекси в оперативній пам'яті, що знижує час доступу до фізичного диску зберігання даних і, відповідно, зменшує час, що витрачається на пошук інформації. Практично для всіх таблиць БД характерною особливістю є наявність ідентифікаторів,

оскільки порівняння таблиць для вибору даних з них проводиться майже завжди саме за допомогою ідентифікаторів основних полів.

Отже, потрібно створити нові методи для підвищення швидкості пошуку інформації в базах даних, які дозволять збільшити наповнення вузлів дерева і завантажувати в оперативну пам'ять лише індекси з найбільшим числом використання при виконанні запитів за час збору статистики, а по іншим індексам – проводити пошук лише після їх завантаження з диска.

У першому випадку B +-дерево модифікується в такий спосіб:

- збільшується мінімальне наповнення вузла;
- змінюються правила зчеплення і розщеплення вузлів дерева;
- зменшується «висота» дерева в зв'язку зі зміною наповнення вузлів;
- розширюються можливості листових вузлів, наприклад, шляхом додавання нового параметра.

Ці дії призводять до отримання модифікованого дерева пошуку (МДП), яке відрізняється від B +-дерева, де n – це параметр дерева, який приймає значення не менше 2, а, як правило, від 50 до 2000 (табл. 1).

Таблиця 1.

Властивості дерев пошуку

Властивості дерева	B+-дерево	Модифіковане дерево пошуку
1.Число елементів, що містяться в корені дерева	від 1 до 2n елементів	від 1 до 2n елементів
2.Число елементів, що містять вузли дерева (крім кореня)	від n до 2n елементів	від 3/2n до 2n елементів
3.Число нащадків вузлів (крім листових), що містять m елементів	m+1 нащадків	m+1 нащадків
4.Листові сторінки дерева знаходяться	на одному рівні	на одному рівні
5.Розщеплення вузлів виконується шляхом	розбиття вузла навпіл на два нових вузла	розбиття вузла з двома його сусідами на чотири нових вузла
6.Зчеплення вузлів виконується шляхом	з'єднання двох вузлів в один новий вузол	з'єднання чотирьох вузлів в три нових вузла
7.Мінімальная висота дерева (t)	$t\geq 2$	$t\geq 2$
8.Посилання на дані	на одну таблицю	на декілька таблиць

Табл. 1 показує, що різниця між B +-деревом і модифікованим деревом пошуку полягає в розбіжностях властивостей 1 і 2. У модифікованому дереві пошуку, згідно властивості 1, є більше наповнення елементів в корені вузла, що призводить до зменшення «висоти» дерева пошуку. За властивістю 2

різниця по мінімальному ступені наповнення вузлів в модифікованому дереві пошуку, на відміну від B +-дерева, становить 3/4 вузла, що дозволяє заощадити місце на жорсткому диску і збільшити швидкість доступу до інформації за рахунок того, що зчитування вузлів модифікованого дерева пошуку, як і B +-дерева, виконується по блокам, і відповідно, від ступеня наповнення блоку залежить кількість одержуваної інформації.

Властивість 8 показує, що в листку модифікованого дерева пошуку необхідно зберігати запис посилання на однакові поля в різних таблицях, тоді як в звичайному B + -дерева зберігається запис посилання на однакові поля в одній таблиці. Даний факт призводить до збільшення часу перебування посилань на дані в дереві, що, відповідно, має на увазі загальне прискорення пошуку.

Для другого випадку потрібно удосконалити процедуру обробки індексів в оперативній пам’яті СУБД наступним чином:

- зменшити обсяг зберігання в ОП СУБД індексів, що мало використовуються;
- обробляти індекси, що мало використовуються, шляхом їх зчитування з диску.

Класичний пошук за допомогою індексів в СУБД виконується в декількох напрямках: спочатку проводиться аналіз даних, що знаходяться в оперативній пам’яті, а потім інформація зчитується з дискового простору.

Наведемо класичний алгоритм завантаження індексів, використовуваних в запитах, в оперативну пам’ять СУБД:

- після отримання запиту СУБД визначає список даних, необхідних для пошуку відповідних даних;
- знайдені дані завантажуються в оперативну пам’ять;
- після обробки наступного запиту СУБД перевіряє, чи знаходяться дані знаходиться в ОП. Якщо індекс присутній в ОП, то виконується пошук по нього, далі він переміщається на перше місце в списку індексів ОП, якщо ж індекс відсутній – завантажує його з індексного файлу диска і розміщує на перше місце в списку, при цьому інші індекси зсуваються на рівень нижче;
- якщо для завантаження наступного індексу пам’ять відсутня, СУБД видаляє індекс з пам’яті, і він займає останнє місце в списку індексів.

Даний алгоритм має свої переваги і недоліки. Однією з переваг є те, що він досить простий в реалізації і дозволяє зберігати індекси в оперативній пам’яті. Недоліком його є те, що в пам'ять потрапляють вкрай рідко використовувані індекси, які і займають там місце до тих пір, поки їх не витісняють інші.

Література:

1. Кренке Д. Теория и практика построения баз данных. Спб.: Питер, 2012. 800 с.

2. Хомоненко А.Д., Цыганков В.М., Мальцев М.Г. Базы данных. Спб.: Корона, 2014. 736 с.
3. Мухін В.Є., Корнага Я.І., Снєгірєв Л. Підвищення ефективності механізмів пошуку в базах даних на основі К-дерев. Вісник Національного університету «Львівська політехніка» «Комп'ютерні науки та інформаційні технології», Львів, 2012, № 744. С. 53-57.
4. Корнага Я.І. Особливості застосування методу кешування індексів в розподілених базах даних. V Всеукраїнська науково-практична конференція «Інформатика та системні науки», Полтава, 2014. С. 155-157.

УДК 004.413

ОСОБЛИВОСТІ ЗАСТОСУВАННЯ АРХІТЕКТУРНОГО ШАБЛОНУ MVC ДЛЯ РОЗРОБКИ МОБІЛЬНИХ ДОДАТКІВ ПІД ОПЕРАЦІЙНУ СИСТЕМУ ANDROID

Аль-Савах М.М.

Черкаський національний університет ім. Богдана Хмельницького

У реаліях сьогодення наше життя неможливе без обчислювальної техніки. Вона стала невід'ємною частиною нашого побуту. Її найголовніша задача полегшувати людям життя, сприяючи виконанню за них різних задач. Для виконання різного типу задач потрібне різне програмне забезпечення. На розробку програмного забезпечення потрібно чимало ресурсів, в яких одним з головних виступає час витрачений на його розробку.

Чим складніша програма та функціонал, який вона підтримує, тим більше часу та зусиль потрібно на її розробку. Для збільшення ефективності процесу розробки програмних продуктів за історію існування такого напрямку діяльності, як програмування, було розроблено чимало так званих архітектурних шаблонів. Вони являють собою ефективні способи вирішення проблем, які часто зустрічаються в ході розробки програмного продукту. Одним з архітектурних шаблонів є MVC.

MVC (Model-View-Controller) – найстаріший з архітектурних шаблонів. Перекладається як модель-представлення-контролер. Вперше був представлений разом із мовою програмування Smalltalk-76 її творцем Трюґве Реенскаугом у 1970-х роках, а пізніше впроваджений в Smalltalk-80 у 1980-х роках, але офіційно вперше пояснений у статті в The Journal of Object Technology у 1988 році [1, 2].

Основною метою MVC було ввести чітке розділення між рівнем даних та рівнем представлення, тобто розбити програмний продукт на компоненти, забезпечуючи розділення відповідальності між ними. У даному архітектурному шаблоні виділяють три компоненти: модель, представлення та контролер (рис. 1).

Рис. 1. Компоненти архітектурного шаблону MVC.

Розглянемо детальніше, що представляє собою кожен з них:

- модель – це рівень даних, відповідальний за управління бізнес-логікою та джерелами даних, такими як бази даних або мережеві API;
- представлення – це те, що бачить користувач на екрані. Тобто віджет на поверхні екрану, який здатний лише намалювати себе і відображає дані моделі;
- контролер – компонент, що відповідає за обробку вхідних даних користувача. Він фактично є проміжною ланкою між двома попередніми компонентами, зв'язуючи їх між собою [3, 3].

З часом MVC еволюціонував і підлаштовувався під потреби різних систем. Однією з цих систем стала Android – найпоширеніша операційна система для мобільних пристроїв. Класична форма шаблону, яка була націлена більше на настільні та веб-додатки, для неї не підходила. Специфічний для системи Android компонент «активність» (Activity), який повинен грати роль контролера, вміщував в собі також логіку користувацького інтерфейсу. Таким чином він був як контролером, так і представленням, що протирічить ідеї розподілу відповідальності. Для вирішення цієї проблеми була створена модифікація архітектурного шаблону MVC (рис. 2).

В модифікованій версії активність виноситься як окремий компонент шаблону. Вона повинна бути зв'язуючим ланцюгом між представленням (файлом верстки) та контролером. Тепер в неї входить лише логіка, пов'язана з користувацьким інтерфейсом.

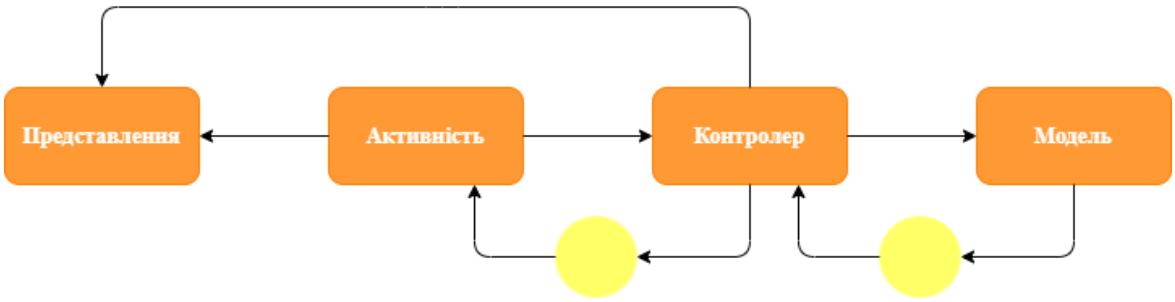


Рис. 2. Компоненти модифікованого архітектурного шаблону MVC.

Таким чином, в даній модифікації роль представлення грають активність та файл верстки з розширенням xml. Контролер же виноситься в окремий клас, який відповідає за обробку дій користувача та отримання, оновлення й обробку даних моделі. В свою чергу модель являє собою звичайний клас,

який відтворює дані. Для того, щоб зв’язати всі компоненти між собою використовуються слухачі, які являють собою інтерфейси. Вони зображені на рисунку 2 у вигляді жовтих кругів.

В результаті даної модифікації було зменшено обсяг коду, що міститься в активності, дотримано принципу розподілу відповідальності між компонентами та загалом покращено здатність програмного продукту, в якому при розробці буде використовуватися цей архітектурний шаблон, до модифікування та підтримки [1, 2, 3].

Література:

1. Architecture Patterns: Model-View-Controller. URL: https://android.jlelse.eu/architecture-patterns-model-view-controller-de312417b4bd (дата звернення 17.12.2020)
2. Karina Sokolova, Marc Lemercier, Ludovic Garcia. Android Passive MVC: a Novel Architecture Model for Android Application Development. PATTERNS 2013: The Fifth International Conferences on Pervasive Patterns and Applications
3. Yun Cheng, Aldo Olivares Domínguez. Advanced Android App Architecture (First Edition): Real-world app architecture in Kotlin 1.3. Razeware LLC, 2019. 250 с.
4. Android Architecture Patterns Part 1: Model-View-Controller. URL: https://medium.com/upday-devs/android-architecture-patterns-part-1-model-view-controller-3baecef5f2b6 (дата звернення 17.12.2020)

УДК 004.624

ТЕХНОЛОГІЇ СТВОРЕННЯ ДОДАТКУ З ПІДТРИМКИ ОБЛІКУ ОСОБИСТИХ ФІНАНСІВ

Луцик Е.С., Розломій І.О.

Черкаський національний університет ім. Богдана Хмельницького

Комп'ютеризація є центральною та обов'язковою умовою розвитку інформаційних взаємодій, що визначають становлення життєдіяльності людини. Сучасне програмне забезпечення вирішує ряд задач в кожній галузі, однією з яких є економічна або ж фінансова. Дана галузь є важливою, тому що забезпечує людей матеріальними умовами існування. Люди ведуть фінансовий облік з різними цілями: виплатити кредити, організувати накопичення, скоротити витрати, спланувати дати платежів і надходжень, тому стикаються з надходженням чи витратами майже кожного дня. Важливо вміти відслідковувати витрати та надходження аби правильно користуватися своїм грошовим ресурсом. При наявності великої кількості товарів та реклами, яка залучає людей витрачати гроші, потрібно мати змогу об’єктивно оцінити стан речей та бути фінансово грамотними. Грамотний

облік особистих фінансів – це оптимізація витрат та надходжень, що включає ряд задач, які можна автоматизувати:

- відслідковування надходжень з різних джерел: готівкові гроші, банківські картки різного типу;
- облік купівлі та продажу товарів в різних валютах і автоматичний перерахунок вартості;
- встановлення фінансових цілей та відслідковування накопичень.

В зв’язку з цим актуальною є розробка додатку з підтримки обліку особистих фінансів.

Для проектування UI/UX було використано Figma – векторний онлайн-сервіс розробки інтерфейсів та прототипування з можливістю організації спільної роботи, що розробляється однойменною компанією. Працює у двох форматах: у браузері та як клієнтський додаток на робочому столі користувача[1–2]. Розробка UI/UX дизайну допомагає значно полегшити проектування додатку. Figma має необхідний набір інструментів, для того щоб створити як простий прототип так і деталізований графічний дизайн. При виконанні кваліфікаційної роботи було створено прототип користувацького потоку виключно з простих форм та об’єктів. Це допомагає зрозуміти всі можливі стани додатку, екрани та потік користувача. Наступним кроком було уточнення прототипу до графічного дизайну з використанням кольору та графічних елементів. В результаті було отримано повноцінний графічний дизайн, який значно полегшив роботу додатку.

Для написання серверної частини використано Node.js. – кросплатформенне середовище виконання з відкритим вихідним кодом, яке дозволяє розробникам створювати різні серверні інструменти і додатки використовуючи мову JavaScript. Середовище виконання призначене для використання поза контекстом браузера (тобто виконується безпосередньо на комп'ютері або на серверній ОС). Таким чином, середовище виключає API-інтерфейси JavaScript для браузера і додає підтримку більш традиційних OS API-інтерфейсів, включаючи бібліотеки HTTP і файлових систем [3].

Express – найпопулярніший веб-фреймворк для Node. Він є базовою бібліотекою для ряду інших популярних веб-фреймворків Node [3–4]. Він надає наступні механізми:

- написання обробників для запитів з різними HTTP-методами в різних URL-адреси (маршрутах);
- інтеграцію з механізмами відтворення представлення «Rendering view», для генерації відповідей, вставляючи дані в шаблони;
- установка загальних параметрів веб-додатку, такі як порт для підключення, і розташування шаблонів, які використовуються для відображення відповіді;
- «Проміжне ПЗ» для додаткової обробки запиту в будь-який момент в конвеєрі обробки запитів.

У той час як сам Express досить мінімалістичний, розробники створили сумісні пакети проміжного програмного забезпечення для вирішення

практично будь-якої проблеми з веб-розробкою. Існують бібліотеки для роботи з куки-файлами, сеансами, входами користувачів, параметрами URL, даними POST, заголовками безпеки і багатьма іншими. Знайти список пакетів проміжного програмного забезпечення, підтримуваних командою Express в Express Middleware можна поряд зі списком деяких популярних пакетів сторонніх виробників [4].

В якості бази даних було використано MongoDb. MongoDB – документо-орієнтована система управління базами даних, яка не потребує опису схеми таблиць. Вважається одним з класичних прикладів NoSQL-систем, використовує JSON-подібні документи і схему бази даних. Застосовується в веб-розробці, зокрема, в рамках JavaScript-орієнтованого стека MEAN.

Вся система MongoDB може представляти не тільки одну базу даних, що знаходиться на одному фізичному сервері. Функціональність MongoDB дозволяє розташувати кілька баз даних на декількох фізичних серверах, і ці бази даних зможуть легко обмінюватися даними і зберігати цілісність [5].

Для клієнтської частини було використано React. React (іноді React.js або ReactJS) – JavaScript-бібліотека з відкритим вихідним кодом для розробки призначених для користувача інтерфейсів. React розробляється і підтримується Facebook, Instagram і співтовариством окремих розробників і корпорацій. React може використовуватися для розробки односторінкових і мобільних додатків. Його мета – надати високу швидкість, простоту і масштабованість. Як бібліотеки для розробки призначених для користувача інтерфейсів React часто використовується з іншими бібліотеками, такими як MobX, Redux і GraphQL [6–7].

Разом з React було використано бібліотеку Redux, яку також використано для створення клієнтської частини. Redux – це така база даних в програмі. Вона зберігає в собі стан (тобто дані) додатку. Redux відповідає тільки за стан і ніяк не пов'язаний з браузером, DOM і фронтенду в цілому. Його можна використовувати, самого по собі, навіть на бекенді в Node.js. Redux, з точки зору коду – це об'єкт, всередині якого лежать дані. Він використовується іншими частинами програми для їх зберігання, зміни і вилучення. У термінології Redux він називається контейнером, так як дані зберігаються всередині [7].

Написання коду здійснене в середовищі WebStorm. JetBrains WebStorm – інтегроване середовище розробки на JavaScript, CSS&HTML від компанії JetBrains, розроблене на основі платформи IntelliJ IDEA. WebStorm забезпечує автодоповнення, миттєвий аналіз коду, навігацію по коду, рефакторинг, налагодження, та інтеграцію з системами управління версіями. Важливою перевагою інтегрованого середовища розробки WebStorm є робота з проектами (в тому числі, рефакторинг коду JavaScript, що знаходиться в різних файлах і папках проекту, а також вкладеного в HTML). Підтримується множинна вкладеність, коли в документ на HTML вкладений скрипт на Javascript, в який вкладено інший код HTML, всередині якого вкладено Javascript, в таких конструкціях підтримується коректний рефакторинг [8].

Таким чином, в статті обгрунтовано необхідність створення додатку, який дозволяє керувати особистими фінансами, також розглянуто основні технології, які були використані при проектуванні додатку.

Література:

1. Остапенко Е.В. (2020) Обзор и сравнение ПО для разработки пользовательских интерфейсов (UI, UX). Научно-образовательный журнал для студентов и преподавателей «StudNet», 9, 328–334.
2. Соловьева А. А., Шуклин Д. А. (2020) Сравнение программного обеспечения для разработки пользовательских интерфейсов и их прототипирования. Наука без границ. Технические науки, 4(44), 55–60.
3. Чепегин И.Д. (2020) Серверный JAVASCRIPT – преимущества и недостатки NODE.JS. Вестник науки и образования, 12(90), 18–20.
4. Итан Браун (2017) Веб-разработка с применением Node и Express. Полноценное использование стека JavaScript. Санкт-Петербург: Питер, 336.
5. Eelco Plugge, Peter Membrey and Tim Hawkins (2010) The Definitive Guide to MongoDB. The NoSQL Database for Cloud and Desktop Computing. SpringerVerlag, New York. Apress, Inc, 328.
6. Стефанов С. (2017) React.js. Быстрый старт. Санкт-Петербург: Питер, 304.
7. Бэнкс А., Порселло Е. (2018) React и Redux: функциональная веб-разработка. Санкт-Петербург: Питер, 2018. 336.
8. Арисова Д. А., Чернова С. В. (2018) К вопросу о веб-разработках. Вестник науки и образования, 15 (51). 24–26.

УДК 004.451.9

РОЗРОБКА ANDROID-ДОДАТКУ ДЛЯ ПЛАНУВАННЯ ПОДОРОЖЕЙ НА ПЛАТФОРМІ APACHE CORDOVA

Пільгун І.А., Розломій І.О.

Черкаський національний університет ім. Богдана Хмельницького

Життя сучасної людини неможливе без подорожей, тому якісне планування є основою комфортного відпочинку. Мобільні додатки для планування подорожей допомагають вирішити ряд питань при організації подорожі, серед яких: бронювання квитків, номерів в готелях, організація екскурсій, пошук однодумців, планування витрат та інші. Такі додатки вивели самостійні подорожі на новий рівень – всі можливості невеликої турагенції тепер з легкістю можна помістити в «кишеню». GPS і мапи, конвертер валют і путівник по місту, словник і зручний пошук готелю – грамотний підбір додатків дозволяє планувати подорож і суттєво економити час та кошти. Планувати подорож стало ще простіше завдяки таким додаткам.

Актуальність розробки додатку для планування подорожі не викликає сумнівів. Огляд аналогів показав, функціонал додатків для планування подорожей різноманітний – від простих нагадувань і списків, до складних систем підбору квитків, трансферу до готелю, бронювання кімнат в готелях і столиків в ресторанах. Запропонований додаток пропонує користувачу перелік визначних місць, виходячи з місця локації користувача. Користувач вказує місто, в якому знаходиться, далі додаток пропонує місця, які можна відвідати в цьому місті. Місця, які вподобав користувач додаються у список для відвідування і надається детальніша інформація про них, ті місця, які були відхилені – відкидаються і більше не пропонуються. Додаток розроблено на платформі Apache Cordova.

Apache Cordova – платформа для розробки мобільних додатків з відкритим кодом, що продовжує розвивати платформу PhoneGap, після передачі проекту проекту компанії Adobe в руки фонду Apache. Вона дозволяє використовувати стандартні веб-технології, такі як HTML, CSS та JavaScript для розробки крос-платформних додатків. Додатки виконуються всередині обгортки націленої на кожну платформу і покладаються на стандартні API для доступу до датчиків пристрою, даних та стану мережі [1].

В результаті розробки додатки є гібридними, це означає, що вони не є по-справжньому ні мобільними застосунками(тому що вся генерація макета здійснюються за допомогою web-view замість основної структури користувацького інтерфейсу платформи), ні web-додатками [2].

Рекомендується використання Apache Cordova в наступних випадках:

- якщо необхідно вже існуючий мобільний додаток «розширити» на більше ніж одну платформу, без необхідності повторної реалізації з використанням мови та відповідного набору інструментів кожної платформи
- якщо при розробці додатку необхідно використовувати native компоненти додатку разом із WebView, який може отримати доступ до API на рівні пристрою [1].

Додатки Cordova покладаються на загальний файл config.xml, який містить інформацію про додаток та визначає параметри, які впливають на те, як працює додаток [1].

Сам додаток реалізовано як веб-сторінку, за замовчуванням локальний файл під назвою index.html, який посилається на будь-які CSS, JavaScript, зображення, файли мультимедіа або інші ресурси необхідні для його запуску. Додаток виконується як WebView в межах оболонки додатка. Інтерфейс розробленого додатку для планування подорожей представлений на рис.1.

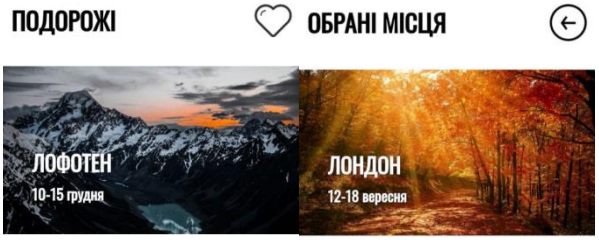


Рис. 1. Інтерфейс додатку для планування подорожей.

WebView з підтримкою Cordova може представляти додаток і повністю його користувацький інтерфейс. На деяких платформах вона також може бути компонентом у великих, гібридних програмах, які об'єднують WebView з іншими компонентами програми.

Плагіни є невід'ємною частиною екосистеми Cordova. Вони забезпечують інтерфейс для взаємодії Cordova та власних компонентів і прив'язку до стандартних API пристроїв. Це дозволяє викликати власний код з JavaScript. Плагін представляє собою пакет із вбудованого коду, що дозволяє WebView Cordova, у якому відображається додаток, взаємодіяти з платформою, на якій працює плагін. Плагіни надають доступ до функцій пристроїв та платформ, які зазвичай недоступні для веб-додатків. Всі основні функції Cordova API реалізовані у вигляді плагінів та існує багато інших, що включають функції, такі як сканери штрих-кодів, NFC-комунікації або адаптований календар інтерфейсів. Для розробників існує реєстр доступних плагінів. Проект Apache Cordova підтримує набір плагінів, які називаються Core Plugins. Це основні плагіни, що надають програмі доступ до можливостей пристрою, таких як споживання акумулятора, камера, контакти тощо. На додаток до основних плагінів, існує кілька сторонніх плагінів, які надають додаткові прив'язки до функцій, не обов'язково доступних на всіх платформах. Їх можна знайти за допомогою пошуку плагінів або npm.

Починаючи з версії 3.0 можна використовувати два основні шляхи розробки мобільного додатку. Хоча і можна використовувати будь-який з запропонованих шляхів для виконання певних задач, кожен з них має певні переваги.

Кросплатформна розробка. Рекомендується використовувати цей варіант розробки, якщо необхідно, щоб додаток запускався та працював на якнайбільшій можливій кількості мобільних платформ, з мінімальними потребами для платформо-специфічної розробки. Цей робочий процес формується навколо утиліти Cordova CLI, яка була доступна починаючи з Cordova 3.0. CLI (command-line interface) – це високорівневий інструмент, який дозволяє будувати проекти якнайбільшої кількості платформ одночасно, абстрагуючи якнайбільше функціонал низько-рівневих скриптів. CLI копіює загальний набір веб-ресурсів до підкаталогів для кожної мобільної платформи, вносить будь-які необхідні для кожного зміни конфігурації, запускає сценарії побудови для генерації двійкових файлів додатків. CLI також надає загальний інтерфейс для застосування плагінів до програми [2].

Платформо-орієнтовна розробка. Рекомендується використовувати цей варіант розробки, якщо необхідно зосередитись на створенні програми для однієї платформи і треба мати можливість модифікувати її на нижчому рівні. Цей робочий процес спирається на набір сценаріїв оболонки нижчого рівня, розроблених для кожної підтримуваної платформи та окрему утиліту Plugman, яка дозволяє застосовувати плагіни. Починаючи з версії 9, Apache Cordova в даний час підтримує розробку для операційних систем Apple iOS, Google Android, Windows 8.1, Windows Phone 8.1, Windows 10 та Electron – програмна база, яка працює на Windows, Linux та macOS.

В результаті було розроблено мобільний додаток для планування подорожей на платформі Apache Cordova, який має зручний користувацький інтерфейс та дозволяє максимально спростити процес організації відпочинку.

Література:

1. Шамсутдинова Е.М. Симакина Н. И. (2016) Применение платформы Apache Cordova при разработке мобильного приложения. Вестник ПГГПУ, 1, 7 –75.
2. Захаров В.Б, Мальковский М.Г., Мостяев А.И. (2017) Проблемы выбора языков программирования при разработке кроссплатформенных приложений. International Journal of Open Information Technologies, 7(5), 29–37.

УДК 004.415.25

РОЗРОБКА ТЕКСТОВОГО РЕДАКТОРА ДЛЯ WINDOWS

Рева А.К., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

Комп’ютерна грамотність не може складатися без знань та умінь використання текстових редакторів. На сьогоднішній час існує безліч програм-редакторів, за допомогою яких вирішуються питання форматування тексту, будь-то літературна доповідь чи програмний код. І не завжди всі функції, які є в цих редакторах можуть знадобитися, однак їх відсутність стане тією причиною, чому користувач не захоче ним користуватися або навпаки буде надавати перевагу.

Текстовий редактор – це програма, яка призначена для роботи з текстом. Головною відмінністю текстових процесорів від текстових редакторів є те, що перші дозволяють редагувати текст (здійснювати форматування) та вставляти нетекстові об’єкти (зображення, діаграми, таблиці, аудіо та відео) [1].

Розглянемо більш детально особливості створення та функціонування текстових редакторів.

На початку 70-х років 20 ст., текстовими процесорами називали спеціальні електронні пристрої, які були призначені для роботи з текстом. Пізніше, так стали називати програмні додатки. Текстовий процесор є різновидом текстового редактору та характеризується тим, що присутня можливість вставляти у документ не тільки текстові дані, а й таблиці, картинки, відеофайли. Тому, файли, які були створені у текстових процесорах неможливо відкрити у текстових редакторах [2].

Першим текстовим редактором у операційній системі Windows є Notepad (стандартний блокнот Windows). Працює з розширенням файлів .txt, але може працювати й з файлами інших типів. Його особливістю є те, що,

якщо потрібно прибрати усе форматування з виділеного блоку речення, достатньо помістити його у блокнот та скопіювати цей блок назад. У буфері обміну буде знаходитись звичайний текст, без непотрібного форматування.

Далі, за рівнем популярності слідує Notepad++, який є вільно розповсюджуваним текстовим редактором з відкритим вихідним кодом, спочатку для операційної системи Windows, а пізніше і для дистрибутивів Debian, а саме Ubuntu та macOS. Notepad++ став покращеною версією звичайного блокноту, завдяки розширеному функціоналу. Найбільшою відмінністю від блокноту частини Windows є підсвітка синтаксису, що є досить зручним при програмуванні на різних мовах. у Notepad++ використовуються налаштування для підсвітки до: HTML, C++, Java, Pascal, Python, PHP та ін.

Тому за основу обрано текстовий редактор Notepad, який є простим, швидким, має у собі такі основні функції: збереження тексту у файл, створити новий файл у поточному вікні текстового редактору, робота з буфером обміну операційної системи, можливість вибору шрифтів, які доступні у операційній системі, вибір локалізації додатку.

Перед реалізацією текстового редактору стояв вибір, як саме буде працювати додаток: у браузері як Web-розробка, при якому для бекенду можна вибрати Java та Spring Framework, для фронтенду – JavaScript та бібліотеку jQuery або безпосередньо в операційній системі Windows із використання мови програмування Java та ключової її бібліотеки Swing, яка і дає можливість створювати програми для вікон Windows [3, 4].

До обрання потрібної технології варто зрозуміти переваги та недоліки кожної з них.

До переваг Web-додатку текстового редактору можна віднести: кросплатформенність, багатофункціональність, не потрібно знати фреймворки для написання саме десктопних додатків.

Основними недоліками Web-додатку текстового редактору є досить повільна розробка, складність розробки для реалізації основного функціоналу текстового редактору, необхідність знань Web-платформи та мови програмування для можливості розробки.

До переваг десктопної версії текстового редактору відносять:

- швидка розробка у порівнянні з Web-додатком;
- кросплатформенність (завдяки мові програмування Java);
- бібліотека Swing вже має готовий основний функціонал для основи текстового редактору, а саме редагування тексту, збереження у файл, друк текстового файлу, зміна шрифту, зміна візуального вигляду додатку;
- пряма взаємодія з файловою системою операційної системи;
- можливість використання буферу обміну уже реалізована у Java, задля задіяння таких функцій, як “Вирізати”, “Копіювати”, “Вставка”
- висока продуктивність додатку.

До основних недоліків десктопних додатків відносять:

- функціонал та його особливості потрібно реалізувати самостійно, у випадку його відсутності у стандартній бібліотеці мови програмування;
- знання особливостей та роботи з бібліотекою для створення десктоп додатків;
- неможливість детального удосконалення загального вигляду додатку.

У результаті реалізації вийшов повноцінний текстовий редактор та завдяки кросплатформенності може бути використаний у будь-якій операційній системі.

Отже, розроблений текстовий редактор може існувати та використовуватися як повноцінний додаток. Його основною відмінністю від тих, що вже існують є зміна мови з англійської на українську. Для цього було створено словник усіх назв у додатку. У подальшому він може бути доповнений іншими функціями згідно вимог користувача.

Література:

1. Текстовые редакторы и текстовые процессоры : веб-сайт. URL: https://inf1.info/wordprocessor (дата звернення: 21.04.2021).
2. Текстовые редакторы : веб-сайт. URL: https://www.researchgate.net/profile/A-Ospanova-3/publication/350451641_Tekstovyj_redaktory/links/60608a54458515e8347749d0/Tekstovyj-redaktory.pdf (дата звернення: 21.04.2021).
3. Trail: Creating a GUI With JFC/Swing : веб-сайт. URL: https://docs.oracle.com/javase/tutorial/uiswing (дата звернення 24.04.2021).
4. Java Swing Tutorial : веб-сайт. URL: https://www.javatpoint.com/java-swing (дата звернення 24.04.2021).

УДК 621.43.056:632.15

РОЗРОБКА ВУЗЛА ДАНИХ ЗА ДОПОМОГОЮ GRAPHQL

Гученко М.С., Отрох С.І.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Проблема негнучкості при побудові сучасного API дуже давно. З поміж багатьох рішень більш за всіх виділяється GraphQL.

GraphQL – це мова запитів для API. А також середа для виконання запитів з даними. Сервіс створений на основі цієї технології є транспортно незалежним, але зазвичай обслуговується через HTTP [1].

За розробкою GraphQL стоїть Facebook. Їм потрібен був гнучкий та потужний API для отримання даних, який може впоратись із усіма

завданнями Facebook, але простий і простий у вивченні та використанні розробниками продуктів. GraphQL був розроблений для розробки власних мобільних додатків Facebook.

За допомогою використання GraphQL немає надмірного або недостатнього завантаження, і GraphQL гарантує, що програма завантажує лише відповідні та абсолютно необхідні дані з сервера. GraphQL, сервер оголошує, які ресурси доступні, а клієнт запитує, які саме йому потрібно. На рис. 1 зображено процес роботу Rest API та GraphQL, з якого видно, що для клієнта процес запиту даних стає на порядок простішим.

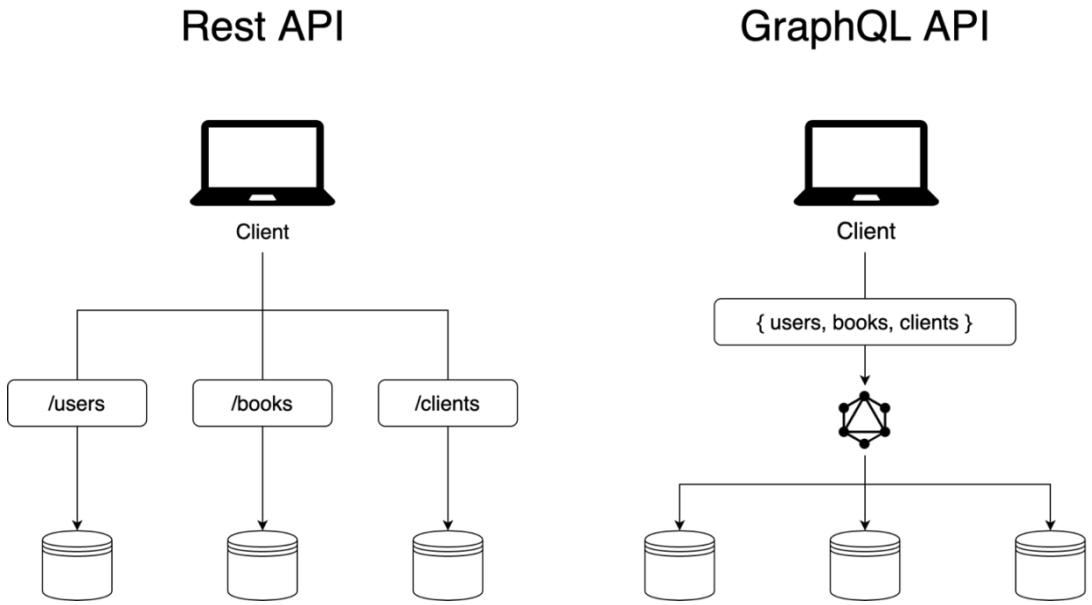


Рис. 1. Порівняння роботи Rest API з GraphQL.

За допомогою GraphQL можна подолати багато проблем, пов’язаних із традиційними Rest API. Але іноді GraphQL робить деякі завдання більш складними.

Основною проблемою GraphQL є кешування HTTP. За допомогою кешування HTTP ми можемо використовувати об’єкт, який він уже має в пам’яті, замість того, щоб робити новий виклик до бази даних. Цей процес зменшує кількість трафіку між серверами до клієнта. У Rest API, клієнти можуть легко використовувати кеш HTTP, щоб уникнути повторного отримання ресурсів. У GraphQL існує лише одна вузол, а не декілька як у Rest, куди надсилаються всі запити. Оскільки кожен запит може бути різним, використовувати цей тип кешування значно складніше.

Переваги, що виділяють GraphQL з поміж інших рішень. Добре підходить для розробки складних систем та мікросервісів. Коли постає питання переходу з монолітної на мікросервісу архітектуру, API GraphQL може допомогти забезпечити зв’язок між кількома мікросервісами, об’єднуючи їх в одну схему GraphQL. Має ієрархічну структуру. GraphQL слідує ієрархічній структурі, де взаємозв'язки між об’єктами визначаються в графічній структурі. Надає можливість визначати форму даних. При запиті

на сервер GraphQL, сервер з допомогою GraphQL повертає відповідь у простій, безпечній та передбачуваній формі. Забезпечує строгу типізацію. GraphQL це стротипізована мова запитів, де кожен рівень запиту GraphQL відповідає певному типу, і кожен тип описує набір доступних полів.

Тим не менш GraphQL має значні недоліки. Обмеження швидкості. В GraphQL складно обмежити кількість запитів за проміжок часу, на відміну від REST, де це робиться відносно просто. Кешування GraphQL. Реалізація кешу у GraphQL реалізувати значно складніше, ніж впроваджувати його в REST. Ми отримуємо доступ до ресурсів у REST API за допомогою URL-адрес, тому ми можемо кешувати на рівні ресурсу, оскільки ми маємо URL-адресу ресурсу як ідентифікатор. У той самий час, в GraphQL він дуже складний, оскільки кожен запит може бути різним, навіть якщо він працює на одній і тій же сутності. Складність запитів GraphQL. Коли робиться запит, сервер здійснює доступ до бази даних. Постає проблема, коли клієнт одночасно запитує занадто багато вкладених полів даних. Для цього має бути такий механізм, як максимальна глибина запитів, зважування складності запитів, уникання рекурсії або постійні запити, щоб зупинити неефективні запити з боку клієнта.

Звертаючи увагу на те, які існують недоліки розроблених вузлів на основі GraphQL, було розроблено вузол який, вирішує наведені вище проблеми з обмеженням швидкості, кешуванням та складністю запитів. У той самий час зберігаються всі переваги, насамперед гнучкість.

Література:

1. Banks A. Learning GraphQL: Declarative Data Fetching for Modern Web Apps / A. Banks, E. Porcello., 2018. – 198 с.

УДК 004.415.25

РОЗРОБКА СИСТЕМНОГО ДОДАТКУ ДЛЯ ЗРУЧНОГО НАПИСАННЯ ТЕСТ-КЕЙСІВ

Кравченко Е.Л., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

Одним із основних етапів розробки програмного забезпечення (ПЗ) є його тестування. За тестування ПЗ відповідають кваліфіковані спеціалісти, які мають знання в цій області. Їхня робота забезпечити якість ПЗ та бути гарантом того, що продукт працює, так як цього очікували. У результаті своєї роботи спеціаліст з тестування ПЗ може мати багато артефактів, одним із таких є список тестових випадків (Test Cases) [1] .

Тестовий випадок (Test Case) − набір вхідних даних, умов виконання та очікуваних результатів, розроблений з метою перевірки тієї чи іншої

властивості або поведінки програмного засобу. Тест-кейс – відповідний документ (артефакт), який представляє формальний його запис [1].

Для полегшення та збереження тестових випадків можна створити спеціальний системний додаток, який буде зберігати потрібну інформацію в реляційній базі даних (БД) MS SQL.

Системним додатком часто називають додатки, що вже встановленні на персональних комп`ютерах користувачів, які неможливо видалити, але можна дозволити або заборонити доступ до них, то цікавим є питання – які ж технології використовуються для їх розробки. Взагалі технологій для реалізації потрібного функціоналу досить багато. Перевагу було надано мові програмування C#, БД MS SQL та середовищу Microsoft Visual Studio 2017.Перевагу було надано мові програмування C#, БД MS SQL та середовищу MSVS 2017.

C# − об'єктно-орієнтована мова програмування із безпечною системою набору тексту для платформи .NET. Синтаксис мови іноді схожий до Java та C++. Мова має строгий статичний тип введення, адаптується до поліморфізму, коментарям формату XML, перевантажених операторів, вказівники на функції членів класу, атрибутів, подій та вінятків. [2].

Microsoft SQL Server (MS SQL) − система управління базами даних, розробкою якої займається компанія Майкрософт. Для виконання та створення запитів використовує мову Transact-SQL. T-SQL – це реалізація ANSI/ISO стандарту із розширеною структурованою мовою запитів SQL. Дану СУБД, зазвичай, викристовують в малих й середніх БД та великих БД на рівні підприємства. Протягом багатьох років успішно конкурує з іншими системами управління базами даних [3].

Microsoft Visual Studio − середовище розробки консольних та графічних програм, спеціально розроблено компанією Microsoft для покращення роботи з кодом. Таке середовище розробки дозволяє підтримувати розробку Windows Forms програм, веб-програм, веб-сайтів, та веб-служб з власним кодом [4]. Підтримка MS SQL, Windows Forms і стали визначальними для вибору цього середовища розробки.

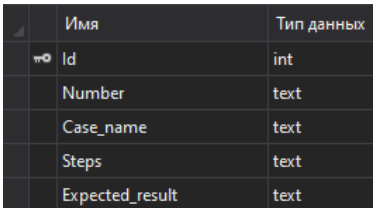
У створенні системного додатку для реалізації тест-кейсів доцільно було використовувати Windows Forms, адже це дозволить користувачу легко та зручно працювати з програмою. В свою чергу розробник може використовувати вже готові візуальні рішення для розробки користувацького інтерфейсу. Для реалізації основного функціоналу розроблюваної програми було визначено основні функції, які вона буде виконувати, а саме: запис, перегляд та видалення даних.

Функцію запису тест-кейсів до БД можна реалізувати за допомогою запиту "INSERT INTO" (рис. 1).

```
{
 SqlCommand command = new SqlCommand("INSERT INTO Test_cases (Number,
         Case_name, Steps, Expected_result) VALUES (@Number, @Case_name,
         @Steps, @Expected_result)", sqlConnection);
         command.Parameters.AddWithValue("Number", textBox4.Text);
         command.Parameters.AddWithValue("Case_name", textBox1.Text);
         command.Parameters.AddWithValue("Steps", textBox2.Text);
         command.Parameters.AddWithValue("Expected_result",
         textBox3.Text);
await command.ExecuteNonQueryAsync();
}
```

Рис. 2. Алгоритм запису до таблиці.

Такий запит через користувацький інтерфейс запише необхідні дані до таблиці, яка містить інформацію про тест-кейси (рис. 2).



	Имя	Тип данных
🔑	Id	int
	Number	text
	Case_name	text
	Steps	text
	Expected_result	text

Рис. 2. Таблиця тест-кейсів.

Як видно з рис. 2, таблиця містить в собі 5 полів: Id − це унікальний номер кейсу для легкого доступу в БД; Number − це номер тест-кейсу, який вкаже користувач; Case_name − це назва тест-кейсу; Steps − це кроки для відтворення тест-кейсу; Expected_result − це очікувані результати до кроків.

Наступним запитом є перегляд інформації з таблиці тест-кейсів. Для реалізації цієї функції було використано простий запит:

```
SqlCommand command = new SqlCommand ("SELECT * FROM test_cases", sqlConnection);
```

Даний запит виводить всю існуючу інформацію із таблиці користувачу. За необхідності розробник може з легкістю змінити стиль виведення тест-кейсів за допомогою простих інструментів Windows Forms та використання примітивного коду.

Ще однією функцією є видалення. Функція видалення реалізована за допомогою наступного коду (рис. 3).

```
{
SqlCommand command = new SqlCommand("DELETE FROM Test_cases WHERE Id=@Id",
            sqlConnection);
      command.Parameters.AddWithValue("Id", textBox5.Text);
 await command.ExecuteNonQueryAsync();
}
```

Рис. 3. Реалізація функції видалення.

Видалення відбувається за допомогою введення користувачем унікального номера тест-кейсу в таблиці. Унікальний номер тест-кейсу користувач може дізнатись тоді, коли виконує запит, який був описаний вище.

Також важливо враховувати той випадок, коли користувач введе невірні дані в поля користувацького інтерфейсу (рис. 4). Для цього реалізовано наступний код.

```
if (label4.Visible)    label4.Visible = false;
if (!string.IsNullOrEmpty(textBox4.Text) &&
!string.IsNullOrWhiteSpace(textBox4.Text) &&
!string.IsNullOrEmpty(textBox1.Text) &&
!string.IsNullOrWhiteSpace(textBox1.Text) &&
!string.IsNullOrEmpty(textBox2.Text) &&
!string.IsNullOrWhiteSpace(textBox2.Text) &&
!string.IsNullOrEmpty(textBox3.Text) &&
!string.IsNullOrWhiteSpace(textBox3.Text))
else
{  label4.Visible = true;
   label4.Text = "Enter full information about test-case";
}
```

Рис. 4. Реалізація функції перевірки невірних даних.

Наприклад, якщо залишити пустими поля, які стосуються інформації щодо тест-кейсів − система повідомляє користувачеві про це повідомленням “Enter full information about test-case”.

Отже, було створено такий системний додаток, який можуть використовувати спеціалісти для полегшення та організації своєї роботи. Перевагою такої реалізації є те, що цей системний додаток буде використовувати менше ресурсів машини та працюватиме стабільніше, що в свою чергу відобразиться на роботі машини в цілому. Він є безкоштовним та не потребує ніяких грошових вкладень для активації.

Література:

1. Святослав Куликов Тестирование программного обеспечения. Базовый курс. 3-е издание. EPAM Systems, 2015-2021. 300 с. [Текст]. URL: https://svyatoslav.biz/software_testing_book/
2. Wikipedia − C# : веб-сайт. URL: https://wikipedia.org/wiki/C_Sharp (дата звернення: 25.04.2021).
3. Wikipedia − Microsoft SQL Server : веб-сайт. URL: https://wikipedia.org/wiki/Microsoft_SQL_Server (дата звернення: 25.04.2021).
4. Wikipedia − Microsoft Visual Studio : веб-сайт. URL: https://wikipedia.org/wiki/Microsoft_Visual_Studio (дата звернення: 25.04.2021).

УДК 004.415.25

РОЗРОБКА ОНЛАЙН-ГРИ ДЛЯ ANDROID ІЗ ЗАСТОСУВАННЯМ КРОС-ПЛАТФОРМНОГО ФРЕЙМВОРКА LIBGDX

Аль-Савах М.М., Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

У реаліях сучасного світу обходитися без мобільних пристроїв – неможливо, адже звернення до будь-якої установи, банку тощо, потребує підтвердження та авторизацію через свій особистий мобільний пристрій. Тому вільний час, що з’являється у людини у знаходженні в чергах, проїздах в транспорті, у подорожах і т.д., можна провести використовуючи свій гаджет та додатки, які є у ньому для відпочинку. За статистикою операційна

система Android є найпопулярнішою серед мобільних пристроїв. Тому і була зроблена спроба розробити гру під Android із застосуванням фреймворка libGDX, який є крос-платформним і переважно використовується для розробки ігор. Він має підтримку для таких операційних систем як Windows, macOS, Linux, Android, iOS, підтримує браузери з WebGL та мови програмування Java, C++ та Kotlin. Для мови програмування Kotlin, яка зараз активно використовується для розробки мобільних додатків під операційну систему Android, є окрема його версія (бібліотека на його базі) libKTX [1].

Розроблена гра полягає в наступному: гравець має збирати бонуси для нарахування балів та оминати перешкоди. При зіткненні з перешкодою гра закінчується. Задача гравця зібрати якомога більше балів, рекордна кількість буде зберігатися і відображуватися на основному екрані. Гравець може рухатися лише по осі x. Інші ж сутності (бонуси та перешкоди) будуть рухатися лише по осі y. Особливістю розробленої гри є те, що через деякі проміжки часу швидкість руху бонусів та перешкод буде поступово збільшуватися.

Для розбиття гри на логічні частини, полегшення підтримки та подальшого розвитку використовується допоміжний фреймворк Ashley, який забезпечує застосування архітектурного шаблону Entity-component-system (ECS, “сутність – компонент – система”) та базується на принципі Composition over inheritance (композиція над успадкуванням). Цей принцип полягає у віддачі переваги поліморфізму над успадкуванням. Тобто згідно з ним, класи повинні прагнути повторного використання коду за допомогою поліморфізму, а не успадковувати поведінку від базового чи батьківського класу. Таким чином одній сутності дозволяється охоплювати кілька компонентів, водночас не пов’язуючи їх між собою [2].

Отже, є чотири основні складові, які забезпечують застосування архітектурного шаблону ECS в Ashley. Розглянемо їх більш детально:

1. Сутність (Entity) – це складова, яка по суті являє собою контейнер для потрібних компонентів. Вона є, зазвичай, моделлю персонажів та предметів в грі. У нашому випадку сутностями є гравець, бонуси та перешкоди.

2. Компонент (Component) – це клас, який являє собою контейнер даних. Він моделює якусь характеристику й забезпечує доступ до її даних. У розробленій грі потрібні компоненти для графіки, руху, гравця, інших сутностей, анімації, трансформації та видалення сутностей.

3. Система (System) – це складова, в якій обробляються сутності та прописується сама логіка гри. Вона обробляє події та оновлює стан сутностей на кожен кадр гри. Одна система відповідає за якусь певну область логіки. В цій грі – це системи для гравця, інших сутностей, руху, анімації, обробки зображень та видалення сутностей.

4. Двигун (Engine) – це складова, яка запускає та керує системами, що були передані їй. Вона також відповідає за створення та видалення сутностей, компонентів та систем, які містяться в ній.

Представимо ці складові на рис. 1. Як видно з рис. 1, всі ці складові тісно взаємодіють між собою і дозволяють прописати логіку роботи гри, яка

полягає в наступному: рух гравця забезпечується обробкою вхідних даних: аналізуються координати, які надсилає сенсорний екран, та поточне положення гравця, після отриманих даних вирішується в яку сторону рухатися гравцеві. Тобто, якщо координати сенсора знаходяться зліва від гравця – його положення зміниться, тоді його поточна координата по осі x зменшиться і навпаки.

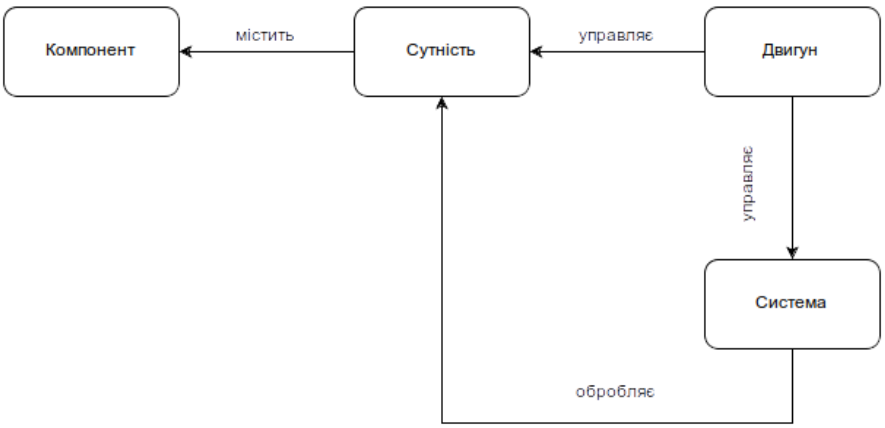


Рис. 1. Складові архітектурного шаблону ECS в Ashley.

Для інших сутностей вказується швидкість, з якою вони будуть змінювати своє положення по осі y. Ця швидкість є змінною, оскільки вона збільшується на певний відсоток з плином певної кількості часу. З оновленням кадру гри перевіряється чи слід змінювати швидкість, тобто чи пройшов заданий відрізок часу. При оновленні значення швидкості лічильник пройденого часу від останнього оновлення обнуляється. Породження сутностей відбувається випадковим чином. Є мінімальна та максимальна кількість пройденого часу від останнього породження, які використовуються для наступних породжень сутностей.

Тип сутності визначається згідно з випадковим шаблоном, обраним з множини наперед заданих шаблонів типів породжених сутностей. Тобто кожен такий шаблон містить наперед задану кількість позицій (порядок породження). У кожному шаблоні прописується тип сутності, яка буде створюватися системою. Якщо сутність не зіткнулася з гравцем, то при досягненні нею нижньої частини екрану (координати 0), вона видаляється. Зіткнення обраховується за допомогою вбудованого в libGDX методу Rectangle.overlaps(Rectangle r), що визначає зіткнення двох прямокутників.

Для зручності розробки та оптимізації використання ресурсів було використано пакувальник текстур (GDX Texture Packer), що входить до інструментів libGDX [3]. Він дозволяє об’єднати в одну структуру всі рисунки, які використовуються при розробці гри, яка може складатися з одного або декількох файлів зображення з розширенням .png та одного файлу з розширенням .atlas, який містить дані про розташування кожного окремого рисунка. Завдяки атласу текстур всі зображення використовують лише одну велику текстуру для їх промальовування, що призводить до економії

ресурсів, адже прив'язка текстур доволі затратна операція. Екранні форми розробленої гри представлено на рис. 2.

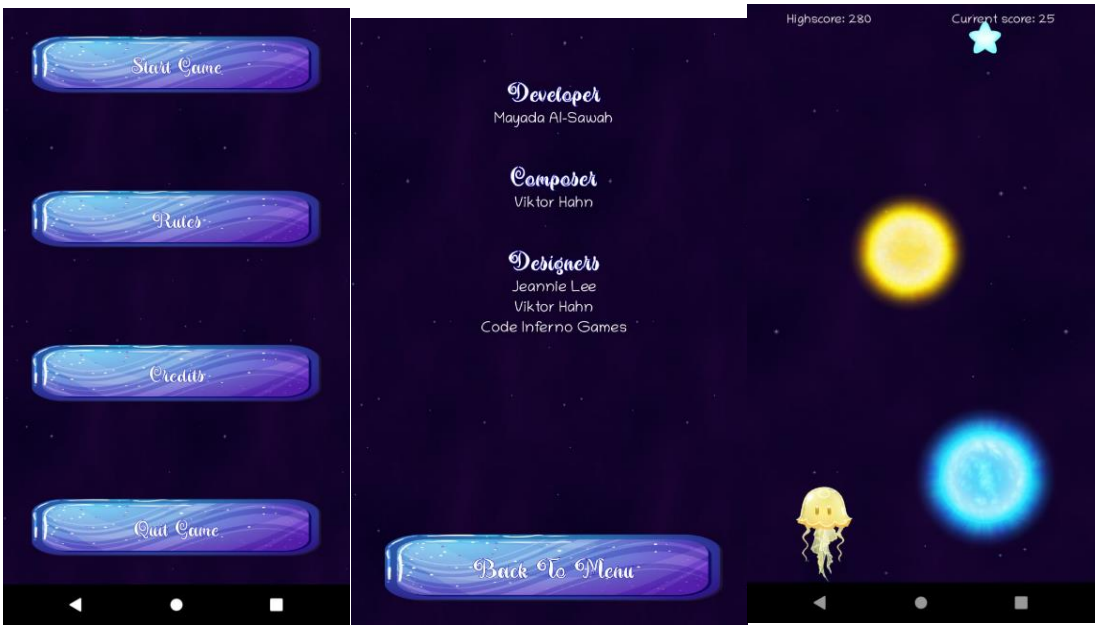


Рис. 2. Екранні форми розробленої гри.

Отже, libGDX являє собою досить зручний і легкий для розуміння й застосування крос-платформний фреймворк для розробки ігор. Його можна застосовувати для розробки як 2D, так і 3D ігор. Крім того, в нього наявні утиліти та інструменти, які покращують процес розробки.

Література:

1. libGDX : веб-сайт. URL: https://en.wikipedia.org/wiki/LibGDX (дата звернення: 29.04.2021).
2. Entity component system : веб-сайт. URL: https://en.wikipedia.org/wiki/Entity_component_system (дата звернення: 29.04.2021).
3. Texture packer : веб-сайт. URL: https://github.com/libgdx/libgdx/wiki/Texture-packer (дата звернення: 29.04.2021).

УДК 004.75

ОСОБЛИВОСТІ ВИКОРИСТАННЯ GRID-ТЕХНОЛОГІЙ ТА ХМАРНИХ ОБЧИСЛЕНЬ

Бєсєдіна С.В.

Черкаський національний університет ім. Богдана Хмельницького

Сьогодні для обробки великих об'ємів даних все більшого поширення набуває впровадження Grid-технологій та хмарних обчислень із гнучким,

скоординованим й безпечним загальним доступом до ресурсів. Застосування цих технологій є ефективним при розбитті великого завдання на більш маленькі та незалежні частини. Grid-технології дозволяють об'єднувати в єдину систему обчислювальні ресурси та ресурси зберігання даних і тісно пов'язані із хмарними технологіями, які включають системи розробки (development framework), розподілені масштабовані обчислення, Grid-обчислення, програмне забезпечення з відкритим кодом, віртуалізацію, ASP, Web 2.0 і т.д. Тому актуальним залишається питання яку ж технологію кращу обрати із застосуванням в тій або іншій сфері діяльності.

Зазначені технології суттєво відрізняються від паралельних обчислень. Для їх реалізації потрібні високопродуктивні розподілені обчислювальні системи, до складу архітектури яких входить:

1. Кластер, що дозволяє організовувати обчислення в кластерній системі і повинен містити методи та засоби сучасних бібліотек й мов програмування; бібліотеки для організації паралельних обчислень; локальні системи управління ресурсами.

2. Клієнт-серверну архітектуру, що організовує розподіл обчислень з використанням серверів додатків, наприклад: для Microsoft – це .NET Framework, Java Platform, Enterprise Edition та інші.

3. Розподілену систему, що забезпечує організацію розподілених обчислень, а саме: Grid-технології: проміжне програмне забезпечення ARC, gLite та їх загальні системні сервіси; хмарні технології: Amazon Web Services, Microsoft Windows Azure, загальні системні сервіси хмарної інфраструктури та системи віртуалізації [1-4].

Хмарні обчислення (cloud computing) або Хмара – це програмно-апаратне забезпечення, що представляються у вигляді сервісу, який має зручний інтерфейс для віддаленого доступу і працює із виділеними ресурсами через Інтернет або локальну мережу [2].

До основних моделей розгортання хмарних обчислень як IT-інфраструктур можна віднести:

1. Публічні (Public cloud) – використовуються одночасно великою кількістю компаній і сервісів, де власник відповідає за їх управління та обслуговування, користувачі такої можливості не мають.

2. Приватні (Private cloud) – безпечні та контрольовані в інтересах лише однієї організації, яка може керувати приватною хмарою самостійно або долучаючи із зовні третю сторону.

3. Гібридні (Hybrid cloud) – для вирішення поставлених завдань об'єднують кращі особливості публічної й приватної хмари.

4. Спільноти (Community cloud) – призначена для виключного використання хмарних обчислень певним співтовариством для вирішення спільних проблем [1, 2].

Сьогодні багато компаній розробляє хмарні сервіси для різних організацій, незалежно від їх чисельності та виду діяльності, де основним фактором їх вибору є використання надійних і високоякісних даних. Тому й

існує досить велика кількість сервісів хмарних обчислень, основні з яких представлено на рис. 1.

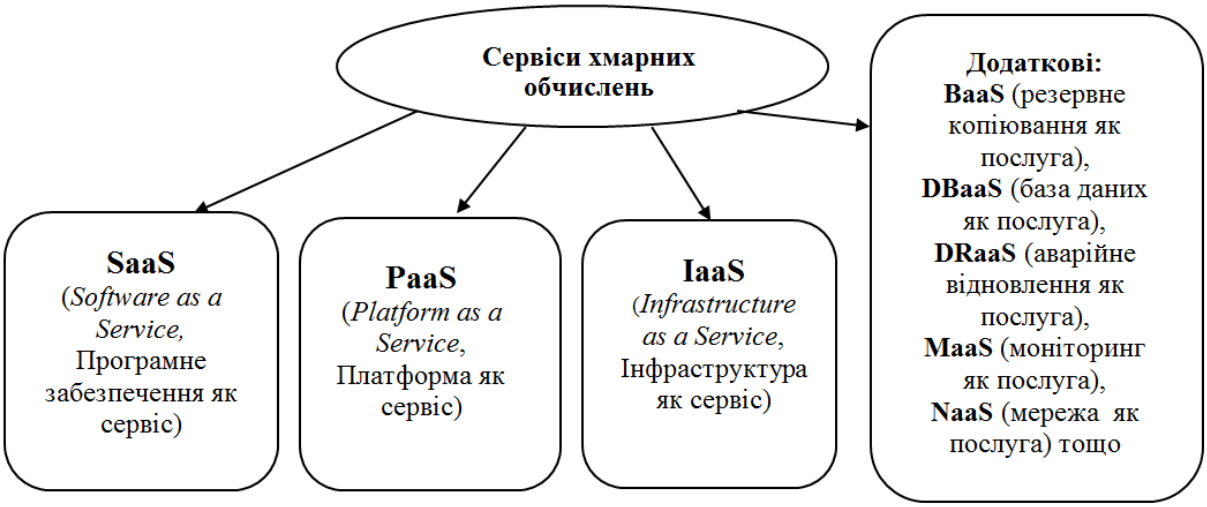


Рис. 1. Сервіси хмарних обчислень.

Якщо раніше користувачеві просто пропонувався простір на хмарних платформах (Amazon, Google) або можливість розміщення своїх хмарних інструментів в існуючих локальних мережах (Microsoft, IBM), то сьогодні необхідність у використанні різних платформ із використанням загальнодоступної Хмари та можливості перенесення до неї різних систем без необхідності їх адаптації для роботи із встановленим програмним забезпеченням чи наданими послугами суттєво зростає.

Основні характеристики порівняння технології Grid та Хмари подано у таблиці 1 [1, 3, 4].

Таблиця 1.

Порівняльна характеристика технології Grid та Хмари

Характеристика	Grid	Хмари
Призначення	для обробки великих обсягів даних з обмеженим часом виконання	для підтримки довгострокових сервісів і задач із великим часом виконання
Розташування обчислювальних ресурсів	в обчислювальних центрах, що розподілені по сайтах, країнах, континентах	у приватних централізованих центрах провайдера даних
Переваги	власність, прозорість, співпраця, стійкість	гнучкість, надійність, простота використання, низька вартість
Недоліки	надійність, складність	безпека, жорсткість, непрозорість, постійне з’єднання з мережею

Отже, можна зробити висновок, що об’єднання Grid-технологій та хмарних обчислень може принести зменшення витрат на підтримку функціонування таких систем, зниження ціни та тривалості обчислень. Тому їх використання інтенсивно йде на заміну звичному Інтернету з його web-

послугами і все це позитивно надалі буде впливати на розвиток різних галузей науки, промисловості та бізнесу.

Література:

1. Хмарні та Грід-технології: конспект лекцій [Електронний ресурс] : навч. посіб. для студ. спеціальності 121 «Інженерія програмного забезпечення». Київ, КПІ ім. Сікорського, 2019. 264 с. URL: https://ela.kpi.ua/bitstream/123456789/29960/1/Khmarni_ta_grid-tekhnolohii_Konspekt_lektsii1.pdf (дата звернення: 28.04.2021).
2. Літошенко А. В. Хмарні обчислення як своєрідний вид аутсорсингу комп'ютерних сервісів та його перевага. URL: http://www.economy.in.ua/pdf/6_2017/18.pdf (дата звернення: 29.04.2021).
3. Хмарні технології визначення. Що таке хмарні технології? Чим хмарні технології відрізняються від звичайних : веб-сайт. URL: https://crashbox.ru/installation-and-configuration/oblachnye-tehnologii-opredelenie-chto-takoe-oblachnye/ (дата звернення: 28.04.2021).
4. Все про хмарних технологіях. Приклади «хмарних технологій» : веб-сайт. URL: https://iotvetnik.ru/uk/vse-ob-oblachnyh-tehnologiyah-primery-oblachnyh-tehnologii-plyusy-i.html (дата звернення: 26.04.2021).

Секція 3

Захист інформації та телекомунікаційні системи

УДК 004.42:621.311

RESEARCH OF EFFICIENCY OF ELECTROMAGNETIC SCREENS AND MEANS OF INCREASE OF THEIR PROTECTIVE PROPERTIES

Kozachuk A.D.

National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"

An electromagnetic field is a special form of matter, through which an action is carried out between electrically charged particles. The physical reasons for the existence of an electromagnetic field are associated with the fact that a time-varying electric field generates a magnetic field, and a changing magnetic field - a vortex electric field: both fields, continuously changing, excite each other. The generated electromagnetic waves exist independently of the source (for example, radio waves do not disappear in the absence of current in the antenna that emitted them).

The effect of electromagnetic radiation (EMR) on a person depends on the following parameters: intensity of EMF (value); radiation frequency; duration of exposure; signal modulation; combination of EMF frequencies, frequency of action [1].

The combination of the above parameters can give significantly different consequences for the response of an irradiated biological object.

The constant impact of EMR on a person affects resonance processes at the molecular and cellular level in various organs and systems of the body. EMR leads to headaches, fatigue, disorders of the cardiovascular and nervous systems, the human immune system suffers. Blood and eyes are most susceptible to EMR, the incidence of oncological diseases and the development of cataracts increases, and the number of people suffering from skin diseases increases.

So, for example, a computer, in addition to the well-known negative impact on a person, during operation generates an electrostatic field around itself, which deionizes the environment, and when heated, the monitor case releases harmful substances into the air. Such air can lead to allergic diseases, respiratory diseases.

Depending on the wave frequency and intensity, the radiation energy can be converted into heat in tissues.

Along with the negative effect of EMR, there is a direction in medicine - magnetotherapy, with the help of which the functions of various organs are restored, and inflammatory diseases are treated.

The following sources of EMR can be distinguished: radio communication, television communications, radio location, radio navigation, radio astronomy, laser systems, electrical engineering, electric power engineering, high-frequency industrial technologies, scientific installations, power transmission lines, physiotherapy equipment, transport (electric trains, including metropolitan areas). ten, trams, trolleybuses, aviation), household appliances, devices and other technical means intended for the transmission and use of electricity and other

processes associated with the generation and use of electromagnetic energy. Another source of EMP is the geopathogenic zones of the earth.

On a current day, the most effective method for collecting the most efficient types of electromagnet injections is the registration of electromagnet screens. By expanding electromagnetical screens with a regular structure, screened surfaces, displayed at the eyes of the sows, albeit at a glance, the efficiency of such surfaces is sometimes not unambiguous. The peculiarities of such constructions are very likely to be found in the form of electro-magnetic vibrations of ultra-high and high frequencies є those, which are practically indicative of the fact that it is practically impossible not to accumulate in the form of metal, or to an alloy of the smelly stinks. Make a name for them when they are broken up and ready to spiral into the economy of the world.

To determine the degree of protection, it is necessary to know the frequency of radiation below which the holes can be considered completely impermeable to radiation [2]. Although in any case, in the presence of many holes in the structure, they are extra-waveguides.

Therefore, to use this effect in the field of electromagnetic safety, it is advisable to use the ratio:

$$f_3 = \frac{1.8*10^4}{d}, \tag{1}$$

where f_3 is the cutoff frequency, MHz; d is the diameter of the hole, mm.

Calculations using this ratio have shown that the apertures become impermeable to radiation at their diameters or wavelengths (for a rectangular aperture) equal to approximately 0.1 of the incident electromagnetic wavelength. This is in good agreement with experimental research and is of practical importance.

Thus, the controllability of the protective properties of such a screen can be ensured by making holes in the form of waveguides of the required length (Fig. 1).

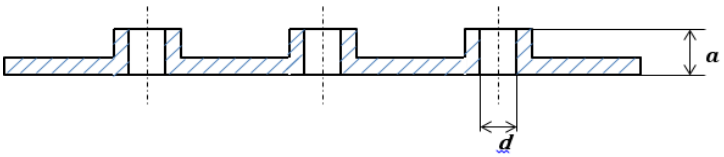


Fig. 1. Section of the electromagnetic screen with holes in the form of waveguides.

The combination of the two above criteria allows to determine with acceptable accuracy for real conditions the required shielding efficiency and screen parameters in terms of reasonable sufficiency.

The latter is due to the localization of the source in space and the spatial distributions of the directions of radiation propagation. Therefore, before designing an electromagnetic screen, it is necessary to measure the frequency-amplitude characteristics of electromagnetic radiation that requires shielding, and determine the spatial patterns of its propagation. Using computational methods, for example in [3], the required parameters of wave electromagnetic fields are determined on

the basis of spherical harmonic analysis with acceptable assumptions and simplifications.

Tests of electromagnetic screens made on the basis of the proposed approaches have proven their high efficiency in both industrial and domestic conditions.

References:

1. Kovalenko V.V. Research of efficiency of modern electromagnetic screens / V. Kovalenko, O. Tikhenko // Safety of life and activity of the person - education, science, practice: collection of sciences. works XV between the people. scientific-practical conf., Kyiv, May 19-20, 2016 - K: 2016. - 126-129 p.
2. Demsky D.V. Method of calculation of shielding efficiency for inhomogeneous electromagnetic screens: dis. … Cand. tech. Sciences: 05.12.14 / Demsky Dmitry Viktorovich. - M., 2014. - 114 p.
3. Kirpanev A.V. Electromagnetic field: theory of identification and its application / A.V. Kirpanev, V.Ya. Lavrov. - M. University book, 2012. - 278p.

УДК 004.715

СЛУЖБОВІ МЕРЕЖЕВІ ПРОТОКОЛИ ДЛЯ СТЕГАНОГРАФІЧНОЇ ІНСКАПСУЛЯЦІЇ

Алексеєнко А.А., Розломій І.О.

Уманський національний університет садівництва

Проблемам інформаційної безпеки на сьогоднішній день приділяється велика увага. Це пов'язано з якісними змінами інформаційних процесів, в результаті яких дані, що зберігаються на комп'ютерах і передаються по мережі, набувають все більшої цінності. Разом з цим зростає кількість загроз і вразливостей, а паралельно з ними розвиваються методи захисту даних. Серед таких технологій окремого розгляду заслуговує стеганографія – наука, що вивчає методи приховування одних даних усередині інших. На сьогоднішній день стеганографія має багато застосувань. На її основі працюють, наприклад, технології так званих цифрових відбитків, коли в кожен екземпляр документа, зображення або повідомлення впроваджується деяка унікальна мітка, що не може бути виявлена стандартними засобами, однак, що дозволяє, в разі необхідності, ідентифікувати даний екземпляр і однозначно пов'язати його з користувачем, якому він був переданий [1].

Мережевою стеганографією називається прихована передача інформації через комп'ютерну мережу, включаючи проміжне обладнання, заснована на модифікації даних в заголовках або полях корисного навантаження мережевих пакетів, а також зміну правил передачі пакетів в тому чи іншому мережевому протоколі [2].

В даний час набув великої популярності підхід, заснований на принципах мережевої стеганографії – сукупності методів прихованої передачі даних з використанням особливостей роботи мережевих протоколів.

Всі мережеві протоколи умовно можна розділити на дві групи: протоколи, призначені для передачі корисного навантаження (як правило призначених для користувача даних) і службові протоколи, використовувані тільки для обміну інформацією про стан мережі, її топології, зіставлення адрес, а також виконують велику кількість інших функцій, безпосередньо не пов'язаних з передачею даних користувача, але необхідних для коректної роботи мережі. До першої групи належать всі протоколи прикладного рівня, а також протоколи TCP, UDP, IP і Ethernet, до другої – ICMP, ARP, DHCP, DNS, всі протоколи маршрутизації і також багато інших менш поширені протоколи.

Як правило, під час налаштування систем аналізу трафіку, для службових протоколів використовують менш строгі правила ніж для транспортних, а іноді і зовсім ігнорують їх, що пов'язано з економією обчислювальних ресурсів, а також уявленням про те, що в службових протоколах не буде призначених для користувача даних, що представляють інтерес для аналізу. Найбільш простим для інкапсуляції тунельованих даних є протокол ICMP, в пакетах якого передбачено місце для розміщення будь-яких даних, безпосередньо не використовуваних самим протоколом.

Іншим поширеним службовим протоколом є ARP (Address Resolution Protocol), призначений для зіставлення IP і MAC адрес. Його пакети часто передаються по мережі, не мають чітких ознак, що дозволяють відрізнити реальний пакет від підробленого, і не блокуються засобами захисту мереж. Незважаючи на те, що в даному протоколі не передбачено окреме поле для запису даних, можна використовувати для цих цілей поля його заголовка, призначені для одного з адрес відправника.

У разі, коли в мережі застосовується динамічна маршрутизація, пакети використовуваного протоколу маршрутизації можна застосовувати для транспортування повідомлень користувача. Такі пакети містять велику кількість даних, наприклад, про топології мережі, для протоколу ICMP, або про вже наявних маршрутах, для протоколу RIP. Довжина цих даних не обмежена, а їх вміст не може бути перевірено проміжним вузлом, що здійснює аналіз трафіку, в результаті чого можливе додавання в пакет значних обсягів даних, які не є маршрутною інформацією.

Для сучасного мережевого обладнання та операційних систем, використовують стек TCP/IP, підтримка протоколу ICMP обов'язкова, що є його безперечною перевагою при використанні для передачі прихованих повідомлень. З цього факту випливають ще кілька переваг стеганографії на основі ICMP-інкапсуляції. Однією з них є періодична розсилка ICMP-повідомлень вузлами мережі, яка відбувається при виявленні помилок в IP-адресі пакетів, відсутності маршрутів до потрібної мережі і багатьох інших випадках, що призводить до великої кількості ICMP-пакетів, що містять

службову мережеву інформацію, серед яких можна приховати пакети з будь-якої іншою інформацією.

Відключення даного протоколу або його блокування за допомогою мережевих екранів, списків контролю доступу та інших засобів безпеки не є хорошим способом захисту від пересилання стеганографічних повідомлень, так як може порушити роботу мережі внаслідок неможливості обміну службовою інформацією. Інша особливість протоколу ICMP, що відрізняє його від інших службових протоколів, і визначається довгою історією розвитку і великою кількістю виконуваних функцій – складність внутрішньої структури пакетів, яка призводить до наявності великого числа можливих поєднань полів заголовка і корисного навантаження.

Незалежно від обраного несучого протоколу, використання принципів стеганографії для впровадження даних в службові мережеві пакети зводиться до визначення потенційно надлишкової інформації в структурі пакета, її компресії і запису в місце, що звільнилося необхідної інформації. Даний принцип показаний на рис. 1.

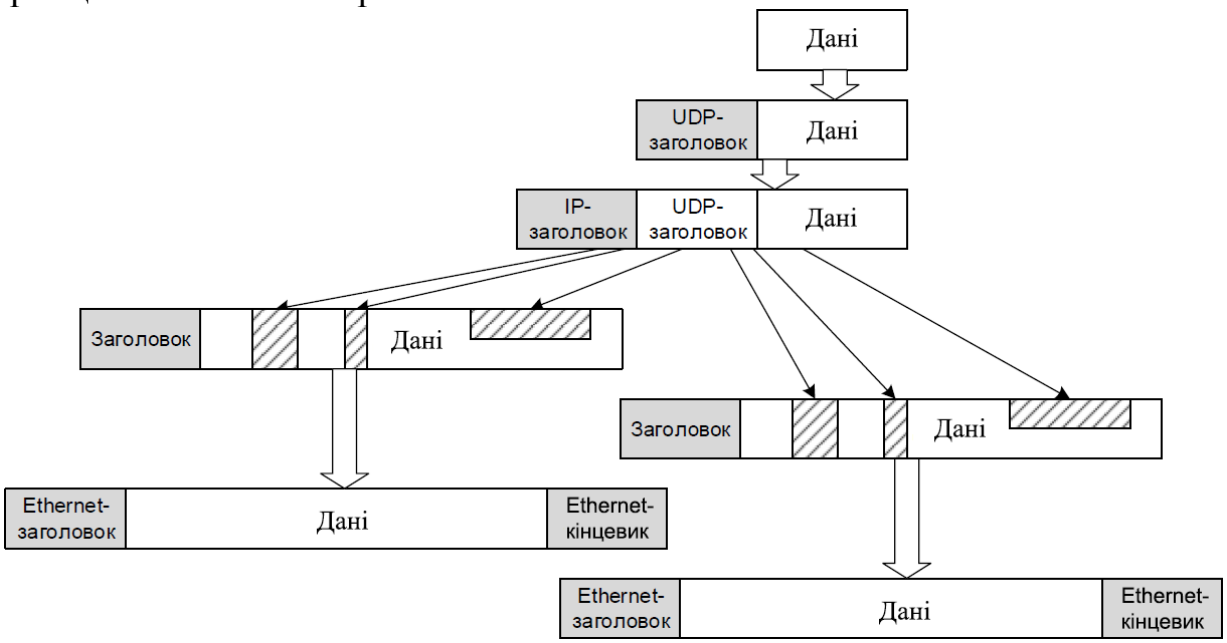


Рис. 1. Стеганографічна інкапсуляція пакетів.

Запропонований підхід до побудови віртуальних тунелів з використанням службових мережевих протоколів дозволяє обходити обмеження, що накладаються фізичною топологією мережі, а використовувані при цьому методи стеганографії забезпечують певний рівень конфіденційності переданих повідомлень. Широкий набір службових протоколів дозволяє використовувати описані методи в будь-яких мережах, а також розділяти передану інформацію між кількома протоколами, підвищуючи загальну швидкість передачі і ускладнюючи аналіз даного трафіку.

Практичне застосування даного методу стеганографічної інскапсуляції мережевих пакетів, що використовуються для обходу різного роду обмежень,

що накладаються інтернет-провайдерами, міжмережевими екранами чи системами фільтрації трафіку.

Література:

1. Пескова О.Ю., Халабурда Г.Ю. (2012) Применение сетевой стеганографии для скрытия данных, передаваемых по каналам связи. Известия южного федерального университета. Технические науки, 12(137), 167 – 176.
2. Голубев Е.А., Емельянов Г.В. (2009) Стеганография как одно из направлений обеспечения информационной безопасности. T-Comm – Телекоммуникации и транспорт, 3, 185 – 186.
3. Галушка В.В. (2016) Сети и системы передачи информации: учебное пособие. Издательский центр ДГТУ, 105.
4. Wojciech Frączek, Krzysztof Szczypiorski (2016) Perfect undetectability of network steganography. Security Comm. Networks, 2998 – 3010.
5. Павлин Д.В., Макосий А.И., Жданов О.Н. (2014) О сетевой стеганографии. Реализация алгоритма RSTEG. Решетневские чтения, 18(2), 322 – 324.

УДК 004.715

РЕАЛІЗАЦІЯ ВІРТУАЛЬНИХ ТУНЕЛІВ НА ОСНОВІ СЛУЖБОВИХ ПРОТОКОЛІВ

Парубочий В.М., Розломій І.О.

Уманський національний університет садівництва

Сучасні мережеві технології являють собою складний набір взаємозв'язаних протоколів, інтерфейсів і алгоритмів взаємодії різних пристроїв, правильне спільне застосування яких дозволяє вирішувати широке коло завдань щодо забезпечення інформаційного обміну. Топології мереж, сформовані в результаті тривалого історичного розвитку шляхом накопичення різних технологій, відрізняються складністю і різноманітністю. У зв'язку з цим актуальним завданням є розвиток методів тунелювання, які дозволяють створювати віртуальні з'єднання, і таким чином формувати логічну топологію мережі, незалежну від фізичного з'єднання її пристроїв.

Взаємодію вузлів комп'ютерної мережі прийнято описувати за допомогою багаторівневих моделей, таких як OSI, що представляє скоріше теоретичний інтерес, і TCP/IP, повсюдно використовується на практиці. Дані моделі описують узгоджений набір протоколів, що називається стеком протоколів, спільне застосування яких забезпечує передачу як призначених для користувача даних, так і службових мережевих повідомлень, необхідних для належного функціонування мережі. Найважливішим протоколом стека TCP/IP є протокол IP (Internet Protocol), на якому побудована вся взаємодія як

в інтернеті, так і в локальних мережах, проте на певному етапі розвитку зіткнувся з наступними проблемами: зростання складності маршрутизації великого числа мереж; залежність адреси від провайдера, складність масової зміни адрес; вичерпання IP-адрес.

Для вирішення останньої проблеми найбільш ефективним і, як наслідок, найбільш поширеним засобом є трансляція мережевих адрес (NAT – Network Address Translation). Дана технологія дозволяє замінювати адреси великої кількості комп'ютерів в локальній мережі на одну адресу шлюзу у зовнішній мережі, якою, як правило, є інтернет. Крім економії IP-адрес, використання NAT також призводить до підвищення безпеки за рахунок приховування інфраструктури внутрішньої мережі, однак дана технологія має важливий недолік – внутрішня мережа, яка перебуває за пристроєм NAT, виявляється ізольованою від зовнішніх з'єднань і повністю «невидима» з інтернету. Така ситуація є неприйнятною для територіально розподілених організацій, чиї інформаційні ресурси розосереджені по декількох внутрішніх мережах, тому для забезпечення зв'язку між ними створюють віртуальні приватні мережі, в основі яких лежать дві технології: шифрування і тунелювання.

Завдання управління тунельованими віртуальними середовищами [1], в тому числі управління інформаційними потоками в віртуальних мережах, в даний час викликає у фахівців в області інформаційних технологій і систем особливий інтерес [2, 3].

Тунелювання, яке найчастіше називається переадресацією портів, являє собою процес передачі даних, призначених тільки для приватного використання. Зазвичай, це включає конфіденційну інформацію в корпоративній мережі через загальнодоступну мережу таким чином, що вузли, маршрутизовані в загальнодоступній мережі, стають необізнаними про те, що процес передачі даних є частиною приватної мережі. Простіше кажучи, тунелювання – це протокол зв'язку, що дозволяє переміщати дані з однієї мережі в іншу. Тунелювання – процес, в ході якого створюється логічне з'єднання між двома кінцевими точками за допомогою інкапсуляції різних протоколів. Він включає в себе спеціальні кроки, які дозволяють передавати приватні мережеві комунікації через загальнодоступну мережу, цей процес називається інкапсуляцією. Інкапсуляція – це процес передачі даних з верхнього рівня додатків вниз по стеку протоколів до фізичного рівня. При просуванні пакету даних за рівнями зверху вниз кожен новий рівень додає до пакету свою службову інформацію у вигляді відповідних заголовків. У цьому процесі інкапсуляції пакети даних виглядають так, як ніби вони є загальнодоступними для публічної мережі, коли насправді вони розглядаються як приватні пакети даних. Це дозволяє їм пройти непоміченими [4].

В процесі тунелювання дані будуть розбиватися на більш дрібні фрагменти, відомі як пакети, які будуть переміщатися по «тунелю» для транспортування до кінцевого пункту призначення. Коли ці пакети проходять через тунель, вони шифруються і інкапсулюються. Приватні мережеві дані і супутній їм інформаційний протокол також інкапсулюються в передавальні

пристрої мережі загального користування для відправки. На приймаючій стороні буде відбуватися процес декапсуляції і дешифрування. Крім того, тунель розглядається в якості логічного шляху або з'єднання, яке буде інкапсулювати пакети, що проходять через транзитну внутрішню мережу. Цей протокол тунелювання буде шифрувати вихідний кадр, щоб вміст не було інтерпретовано за межами маршруту. Для того щоб процес дійсно працював, дані будуть відправлятися, як тільки тунель буде вже встановлено, і клієнти або сервер будуть використовувати один і той же тунель для передачі і отримання даних через внутрішню мережу. Передача даних буде залежати від протоколів тунелювання, які використовуються для передачі [5].

В процесі інкапсуляції, що виконується при тунелюванні, виділяють наступні типи протоколів:

- протокол, що транспортується тобто той, чиї дані потрібно передати через тунель;
- протокол інкапсуляції, який виконує службові функції передачі параметрів тунелю і позначення самого факту тунелювання;
- несучий протокол, призначений для передачі всієї перерахованої вище інформації з питань зовнішньої мережі.

Від звичайних багаторівневих мережевих моделей (таких як OSI або TCP/IP) тунелювання відрізняється тим, що протокол, який транспортується відноситься до того ж або більш низького рівня, ніж використовуваний в якості тунелю несучий протокол (рис. 1).

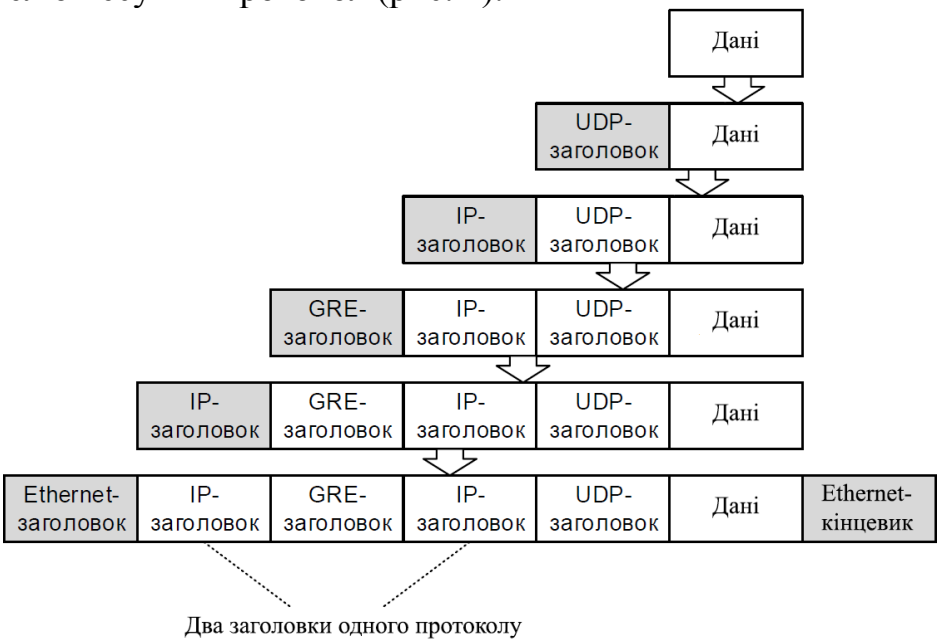


Рис. 1. Інкапсуляція пакетів при тунелюванні.

На рис. 1 протоколами, що транспортуються є UDP і IP, протоколом інкапсуляції – GRE, а несучим протоколом, також як і транспортованим – IP. Описаний підхід є стандартним, внаслідок чого легко розпізнається системами аналізу трафіку, а в разі відсутності додаткового захисту піддається подальшому аналізу для вилучення корисної інформації.

Література:

1. Таненбаум Э., Уэзеролл Д. (2012) Компьютерные сети. 5-е изд. СПб.: Питер, 960.
2. Mazurczyk, W., Wendzel S., Zander S., and Szczypiorski K. (2016) Information hiding in communication networks: fundamentals, mechanisms, applications, and countermeasures. John Wiley & Sons, 100–104.
3. Шейда В.В. (2010) Использование протоколов TCP и UDP для защищенной передачи информации по SSL-VPN-туннелям. Доклады ТУСУРа, 1(21), 225–229.
4. Усков А.В., Иванников В.Л., Усков А.Д. (2008) Технологии обеспечения информационной безопасности корпоративных образовательных сетей. Educational Technology & Society, 11(1), 472–479.
5. Хесус Назарет Бенитес Гонсалес (2016) Технологии создания виртуальных частных сетей. Молодой исследователь Дона, 2(2), 1–3.

УДК 004.738.5:343.346.8

СИСТЕМА ВИЯВЛЕННЯ ХАКЕРСЬКИХ АТАК НА ОСНОВІ АНАЛІЗУ ПОВЕДІНКИ ВІДВІДУВАЧІВ ВЕБ-САЙТУ

Гаркавий В.В., Розломій І.О.

Уманський національний університет садівництва

В умовах інформаційного протиборства коли ціна інформації вельми висока стає очевидним, що для забезпечення успішної протидії атакам необхідно, щоб порушник діяв в умовах апріорної невизначеності. Для цього пропонується створення облудної системи, метою якої є залучення правопорушника в свого роду «гру», збільшуючи тим самим час, необхідний на обхід системи захисту інформації. Робота обманних систем полягає в тому, що вони емулюють ті або інші відомі вразливості, які в реальності не існують.

Облудна система класифікує відвідувачів веб-сайту на основі запитаних ними веб-сторінок. Основна ціль – відокремити потенційних зловмисників від звичайних відвідувачів, тобто скласти білі і чорні списки IP-адрес. Актуальні чорні і білі списки IP-адрес важливі для сучасних засобів захисту інформації. Пісочниці і інші інструменти динамічного аналізу виявляють шкідливу активність за зверненнями до IP-адрес з поганою репутацією. Антивірус і засоби статистичного аналізу можуть діставати IP-адреси із сканованих файлів і видавати висновок на основі відомостей про ці адреси. Міжмережеві екрани, системи по виявленню і запобіганню вторгнень та інші інструменти для відстеження мережевої взаємодії можуть в своїй роботі спиратися на списки відомих IP-адрес джерелом атак. Чорні і білі списки IP-адрес в цілому необхідні в індустрії інформаційної безпеки [1].

В чорний список повинні потрапляти IP-адреси тих, хто атакує облудну систему. Мережеві адреси звичайних відвідувачів, які безпечно зчитують контент, повинні зберігатися в білому списку.

Для класифікації відвідувача облудна система визначає три поведінкові ознаки: розраховується число звернень відвідувача веб-сайту до неіснуючих сторінок; визначається число веб-сайтів, на яких був здійснений вхід з одного і того ж IP; фіксується число таких переходів по посиланням, які показують, що відвідувач розуміє сенс тексту на сайті. Звичайні користувачі, як правило, зчитують контент веб-сайтів з однієї і тієї ж IP-адреси і не можуть порівнюватися в розумінні семантики тексту з людиною. Хакери в свою чергу запускають автоматизовані сканери вразливості, які перевіряють наявність визначених сторінок на безлічі сайтів. В запропонованій облудній системі таких вразливих сторінок немає.

На кожному веб-сайті в розробленій облудній системі заздалегідь відомо список викладених файлів. В ході аналізу файлів журналу веб-сервера звернення до неіснуючих файлів легко знаходяться. Веб-сайти в облудній системі створені за одним шаблоном і відрізняються лише доменним іменем і мовою контексту. Доменні імена зареєстровані в один день тому користувачі, зайшовши на один із сайтів, як правило, заходять і на інші. Тести, містяться на веб-сайтах, написані на чистому HTML за допомогою гіперпосилань, щоб спростити аналіз журналів веб-сервера.

В результаті експерименту була зафіксована активність обхідників відомих пошукових компаній і поведінка автоматизованих хакерських утиліт. Були виявлені шляхи до файлів, в яких зловмисники очікували найти вразливості. Поведінка реальних користувачів відрізнялась від активності автоматизованих утиліт, як за кількістю помилок в тестах, так і за мовними вподобаннями. Всі зібрані IP-адреси були розмічені по передбачуваним країнам-джерелам за допомогою сервісу геолокації.

Існуючі застосуванні на практиці облудні системи, як правило, виявляють атаки на один інформаційний ресурс, спираючись на події, зв’язані тільки з цим захищеним ресурсом. Окрім цього, існують обманні системи які не використовують контент при аналізі поведінки потенційно атакуючих. Наприклад, веб-орієнтована облудна система Glastopf збирає інформацію про атаки на веб-програму. В пастці зберігаються шкідливі файли, які зловмисники пробують розповсюдити, експлуатуючи різні вразливості, які емулює Glastopf. Зокрема, в описі системи вказана можливість емулювати PHP-ін'єкції і SQL-ін'єкції.

Завантажені шкідливими файли зберігає також облудна система Геральда Вагенера. Ця високо інтерактивна облудна система замаскована під SSH-сервер. Поведінка пастки визначається теоретико-ігровою моделлю і машинним навчанням. Діями облудної системи є виконання команд зловмисника або заміна програми, яка повинна виконати команду. Перемога облудної системи визначається кількістю нових виявлених небезпечних об’єктів, зокрема шкідливих файлів, які завантажив атакуючий [2].

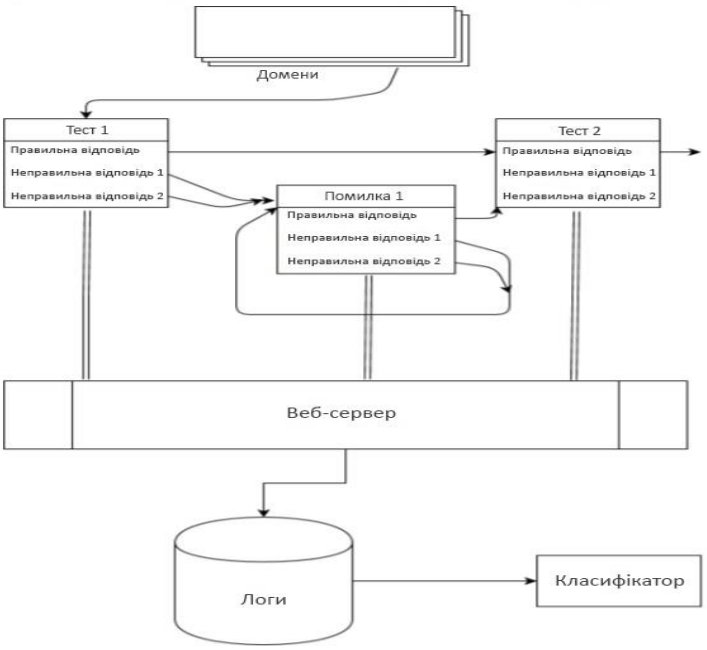


Рис. 1. Структурна схема облудної системи.

Запропонована система представляє собою набір із двадцяти доменних імен в зоні RU, прив'язаних до сайтів на виділеному сервері. На сервері встановлений Apache і викладені HTML-файли з тестом і зображенням. Текст сторінок на кожному із двадцяти веб-сайтів відповідає назві домена. На кожному сайті є головна сторінка і сторінки з тестами на знанням іноземних мов. В тестах необхідно вибирати із трьох слів те, яке відповідає зображенню. Виклик по невірній відповіді веде на сторінку з помилкою. Перехід на сторінку з помилкою так само, як і вибір правильної відповіді фіксується в файлі журналу веб-сервера Apache. Програма на мові Python розбирає отримані журнали і класифікує IP-адреси.

Оскільки всі внутрішні посилання на веб-сайтах в облудній системі вели на існування сторінки, спроби доступу до неіснуючих сторінок вважалися атаками. У звичайного користувача немає можливості натиснути по посиланню так, щоб була зафіксована атака. Хибне спрацьовування можливе тільки, якщо користувач ненавмисно помилиться при ручному введенні назви веб-сторінки. Розпізнання обхідників засноване на кількості веб-сайтів, на які зайшли з однієї й тієї ж IP-адреси. Доменні імена веб-сайтів в облудній системі обрані так, як ніби це сайти для вивчення іноземних мов по карткам з картинками.

Спосіб збору підозрілих IP-адрес здійснюється таким чином: аналізатор розбиває файл журналу веб-сервера на групи запитів від кожної окремої IP-адреси, потім обчислює число помилок в тестах, число відвіданих доменів серед двадцяти підконтрольних облудній системі і визначає наявність звернень до веб-сторінок, відсутнім на сайті. При класифікації поведінки за допомогою тесту допускається, що людина може помилятися. Передбачається, що звичайний користувач може допускати в тесті не більше

трьох помилок. Якщо допущено чотири і більше помилок, вважається, що це схоже на поведінку веб-обхідника, який переходить по всіх посиланнях.

У даній роботі поведінку відвідувачів веб-сайтів було проаналізовано за допомогою веб-орієнтованої облудної системи. Використання облудної системи показало, що число відвіданих сайтів-пасток є суттєвою ознакою, що характеризує хакерські атаки. Запропонований поведінковий підхід дозволяє визначити джерела хакерських атак.

Література:

1. Zhong J., Yan C., Yu W., Zhao P., Wang M. A (2014) Kind of Identity Authentication Method Based on Browsing Behaviors. 2014 Seventh International Symposium on Computational Intelligence and Design. Hangzhou. 279 – 284.
2. Davide Canali, Leyla Bilge, and Davide Balzarotti. On the effectiveness of risk prediction based on users browsing behavior. In Proceedings of the 9th ACM symposium on Information, computer and communications security (ASIA CCS ‹14). ACM, New York. NY. USA. 2014. 171–182.

УДК 004.658.2

КРИПТОГРАФІЧНИЙ ЗАХИСТ ХМАРНИХ БАЗ ДАНИХ ЗА ПРИНЦИПОМ ГОМОМОРФІЧНОГО ШИФРУВАННЯ

Білоус І.В., Нагорний П.В.

Національний університет «Чернігівська політехніка»

Одним з найбільш перспективних напрямків розвитку computer science є розробка та інтеграція хмарних технологій в корпоративні програмні системи. При цьому очевидною сферою застосування хмарних технологій є адміністрування баз даних. Слід зазначити, що організація несанкціонованого доступу до хмарних серверів в багатьох випадках простіша, аніж до приватних серверів компанії. Тому проблема захисту конфіденційних даних компанії, особливо в умовах досить частих кібератак, актуальна. З цією метою вченими запропоновано застосовувати криптографічне шифрування даних.

Сучасні бази даних відрізняються від звичайного сховища даних можливістю виконувати певні операції над даними – запити. Тому виникає питання принципів поводження із зашифрованими даними, потенційною можливістю виконувати над ними необхідні операції. Найчастіше застосовний підхід полягає у збереженні на сервері не тільки зашифрованих даних, а й відповідного ключа. Але такий підхід відрізняється своєю недосконалістю та високим ризиком несанкціонованого доступу до даних. Інший підхід полягає у тимчасовому копіюванні зашифрованих даних бази, розшифруванні та виконанні запитів на стороні клієнта. Проте такий підхід

характеризується високими вимогами до апаратного забезпечення клієнта та повністю нівелює переваги хмарних технологій [1].

Найскладнішим, але й найбільш перспективним є принцип гомоморфічного шифрування. Основна ідея шифрування за таким принципом полягає у здійсненні всіх операцій над зашифрованими даними за правилами математичного гомоморфізму [1]. Нехай маємо два повідомлення – x_1 та x_2, які належать певному полю P_x із встановленими операціями додавання + та множення ×. Операцію шифрування позначимо E(x), а операцію дешифрування – D(x). Тоді для алгоритмів повного гомоморфічного шифрування має виконуватися $D\big(E(x_1) + E(x_2)\big) = x_1 + x_2$ та $D\big(E(x_1) \times E(x_2)\big) = x_1 \times x_2$. Якщо виконується лише одне з рівнянь – алгоритм називають частково гомоморфічними. Такі алгоритми (для звичайної операції множення на полі цілих чисел) існують: RSA, схема Ель-Гамаля. Проте частково гомоморфічні шифрувальні алгоритми все ж таки потребують часткового розшифрування даних на сервері, що різко знижує надійність таких алгоритмів. В той самий час, повністю надійні та ефективні схеми повного гомоморфічного шифрування ще не розроблені [2].

Розглянемо більш детально модель бази даних, яка реалізує алгоритм гомоморфічного шифрування. Для спрощення вважатимемо, що база даних має лише одну таблицю table, яка має n атрибутів $a_1, a_2, \dots, a_n$, та представляє собою набір кортежів $R_i = \langle v_1, v_2, \dots, v_n \rangle_i$, де $\{v_j\}_i$ – значення атрибута a_j для i-го запису в базі даних. Тепер нехай клієнт може здійснювати запити SELECT * FROM table WHERE $(a_1 = x_1)$ AND $(a_2 = x_2)$ AND … AND $(a_n = x_n)$ та SELECT * FROM table WHERE $(a_1 = x_1)$ OR $(a_2 = x_2)$ OR … OR $(a_n = x_n)$. Клієнт володіє секретним ключем шифрування, який є невідомим для серверу. В базі даних на сервері зберігаються виключно зашифровані дані. Клієнт за допомогою секретного ключа на своїй стороні шифрує параметри запиту та починає його виконання. На початку виконання запиту сервер отримує від клієнта пари $\big(a_j, E(x_j)\big)_i$ та починає їх опрацьовувати [2].

Під час виконання запиту сервер порівнює $\{E(x_j)\}_i$ та $\{E(v_j)\}_i$ за допомогою спеціальної функції f(), яка повертає логічне значення у гомоморфному вигляді: якщо $\{E(x_j)\}_i$ та $\{E(v_j)\}_i$ рівні, функція повертає $E(1)$, де 1 – одиниця у полі P_x; в іншому випадку функція повертає $E(0)$, де 0 – нуль у полі P_x. Для записів R_i сервер вираховує $r_i = f(E(x_1), E(v_1))\hat{}f(E(x_2), E(v_2))\hat{} \dots \hat{}f(E(x_n), E(v_n))$ для випадку запиту з умовою WHERE AND … AND, та вираховує значення $r_i = f\big(E(x_1), E(v_1)\big)|f\big(E(x_2), E(v_2)\big)| \dots |f(E(x_n), E(v_n))$ для випадку запиту з умовою WHERE OR … OR. Сервер повертає клієнту кортеж $\langle r_1, r_2, \dots, r_m \rangle$, де m – кількість записів у таблиці table.

Клієнт отримує значення необхідних індексів, використовуючи наступну умову: індекс і є шуканим тоді і тільки тоді, коли $D(r_i) = 1$. Необхідні записи

клієнт отримує за стандартною процедурою отримання зашифрованих даних, описаною в [3].

Описана методологія гомоморфічного шифрування має ряд недоліків: зокрема, клієнт має опрацьовувати обсяг даних, рівнопотужний стовбцю таблиці table, яка може бути дуже великою. Тим не менш, більш ефективних підходів гомоморфічного шифрування на сьогоднішній день не розроблено.

Гомоморфічне шифрування має свої переваги та недоліки. До переваг можна віднести: збереження ключа виключно на стороні клієнта, відсутність необхідності навіть тимчасового розшифрування даних на серверах, просту концептуальну модель взаємодії клієнта та сервера в аспекті шифрування інформації. До недоліків можна віднести: складну та недосконалу математичну модель шифрування, неефективність сучасних гомоморфічних алгоритмів шифрування [4].

Тим не менш, гомоморфічне шифрування є перспективним напрямком розвитку криптографічних систем захисту інформації в базах даних, та в майбутньому може бути ефективно застосовне під час розробки корпоративної системи захисту конфіденційної інформації.

Література:

1. Kim, M., Lee, H. T., Ling, S., Ren, S. Q., Tan, B. H. M., Wang, H. Better Security for Queries on Encrypted Databases. // IACR Cryptology ePrint Archive. 2016. № 470. URL: http://eprint.iacr.org/2016/470.pdf (дата звернення: 01.05.2021)
2. Бабенко Л. К., Буртыка Ф. Б., Макаревич О. Б., Трепачева А. В. Полностью гомоморфное шифрование. Вопросы защиты информации. 2016. № 3. С. 3-25.
3. Boneh D., Gentry C., Halevi S., Wang D., Wu D. J. Private database queries using somewhat homomorphic encryption. Applied cryptography and network security Springer, 2013, p. 102–118.
4. Ступень П. В., Соколов С. О., Золкіна О. Ю. Застосування гомоморфного шифрування для захисту числових даних у хмарних сховищах. Наукові праці Чорноморського державного університету імені Петра Могили. Серія: Комп'ютерні технології. 2015. № 254. С. 71-75.

Секція 4

Інтелектуальні системи, технології та робототехнічні комплекси

УДК 004.93'1, 004.032.26

ДОСЛІДЖЕННЯ МЕТОДІВ ВИЯВЛЕННЯ ЕМОЦІЙ ЗА ФОТО

Мостовий І.Д., Бушин І.М.

Черкаський національний університет ім. Богдана Хмельницького

Емоції є реакцією на зовнішні та внутрішні подразники, набутою у процесі еволюції, нині відіграють ключову роль у взаємодії між людьми.

Як відомо, під час розмови увагу зосереджено саме на обличчі співбесідника, тому невербальна поведінка, така як рухи тіла та міміка, є одним із визначаючих факторів взаємин між людьми. Зміна виразу обличчя завжди має емоційне навантаження, здебільшого відображаючи емоційний стан людини. У той же час, прояв тих чи інших емоцій можуть бути ознакою фізичних чи психічних порушень. Вчасне виявлення таких ознак дуже важливе, а під час пандемії COVID-19 [1] та, як наслідок, вимушеної ізоляції, стало ще більш актуальним.

Добробут суспільства протягом його історичний розвитку нерозривно пов’язаний з технічним розвитком. Інформаційні системи та інтелектуальні технології як результат науково-технічного прогресу нині відіграють важливу роль в житті людини. Їх використання автоматизує та спрощує життя людей у всіх сферах діяльності. У свою чергу, діджиталізація та науково-технічний розвиток створили відносно нову галузь штучного інтелекту, яка нині використовується для моніторингу та захисту даних, зокрема із застосуванням технологій розпізнавання голосу та обличчя. Інновацією в цій галузі стало розпізнавання емоцій за зображеннями.

Процесом розвитку інформаційних технологій, їх складових займалися світові та вітчизняні науковці: Н. Вінер, М. Шлезінгер, О. Івахненко та ін. Природу фізичного виявлення емоцій досліджували О. Селфрідж [2], П. Екман [3], Р. Плутчик [4], У. Фрізен, Д. Хагер. Проте розробка ефективного механізму відслідковування емоцій за зображеннями розвинута недостатньо та потребує додаткових досліджень.

Мета - представлення механізму визначення емоцій за зображенням.

Пандемія COVID-19 актуалізувала процес інноватизації та потребу науково-технічних розробок, зокрема у сфері інформаційних технологій. Наслідки її протікання спостерігаються у всіх сферах діяльності, а також на загальному стані здоров’я населення. Тому для запобігання розвитку депресії, хронічного пригнічення та інших психічних захворювань важливим стала розробка технологій визначення емоцій.

Нині виділяють три основні моделі категоризації емоцій, які широко застосовуються на практиці, а саме: дискретна, багатовимірна та гібридна.

Аналізуючи специфіку дискретної моделі, визначено, що цей підхід пов’язує кожну емоцію із семантичним параметром або масивом значень. Прикладами реалізації такої моделі є теорії базових універсальних емоцій. Варто зазначити, що питання поділу складних емоцій на базові залишається

спірним, тож різні дослідники і експерти описують різну кількість базових емоцій та їх типів. Наприклад, П. Екман виділяє такі базові емоції: радість, сум, страх, відраза, злість та здивування. У свою чергу, Р. Плутчик у своїй психоеволюційній теорії визначив 8 основних емоцій: радість, страх, злість, здивування, незадоволення, довіра, сум, очікування та зв'язані з ними, більш складні, емоції. Дж. Грей вважає, що варто говорити лише про 3 основні емоції, а на думку Мауера – про дві: біль і задоволення.

Багатовимірна модель дозволяє розмістити емоції в координатному багатовимірному просторі, а враховуючи неперервність простору, встановлено, що досліджувані емоції відрізнятимуться за одним або декількома параметрами. При цьому не виключається можливість, що вони мають однакове підґрунтя. Прикладом застосування даної моделі можна вважати напрацювання Дж. Рассела [5] та його модель емоцій. В ній продемонстровано двовимірний базис, в якому кожна емоція характеризується знаком валентності (valence) та інтенсивністю (arousal) (рис. 1).

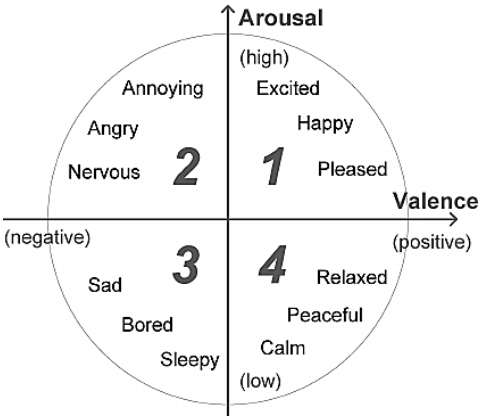


Рис. 1. Двовимірний простір моделі Дж. Рассела.

Гібридні моделі об'єднують підходи дискретних та багатовимірних моделей в одну. Так, модель «Пісочний годинник емоцій», яку представили та описали Е. Камбрія, А. Лівінгстон, А. Хусейн у своїй книзі «The Hourglass of Emotions», демонструє використання цієї моделі, де кожен вимір характеризується 6 рівнями сили, з якою виражена емоція. У свою чергу, ці рівні представлені набором із 24 емоцій. Тому за даною теорією кожну емоцію можна розглядати і як стан, і як частину простору, пов'язану з іншими емоціями.

Найбільш поширеним і ефективним нині методом класифікації емоцій є визначення емоції за допомогою геометричного аналізу ключових точок на фотографії обличчя людини.

Згідно з цим методом, визначається від 5 до 70 ключових точок на обличчі (залежно від заданої точності та вимог до швидкодії), які визначають положення губ, очей, брів, зморшок, носа та щелепи, що дозволяє отримати геометричне представлення змін міміки людини. Далі отримані координати точок обробляються класифікатором, в ролі якого найчастіше використовують метод опорних векторів (рис. 2).

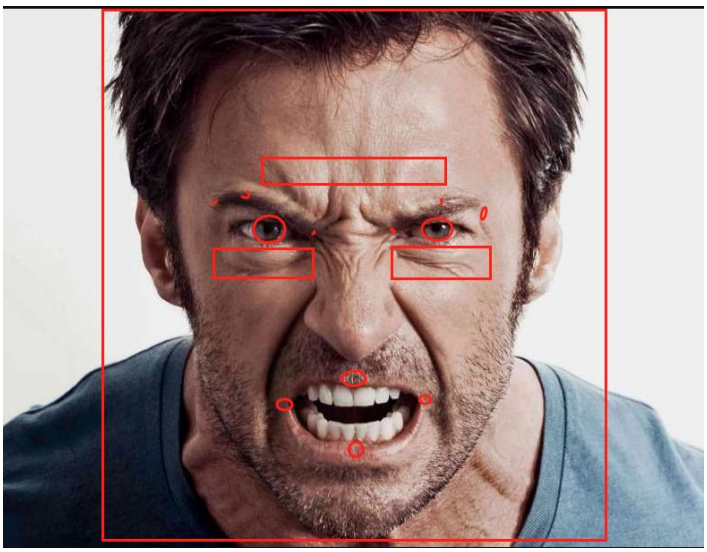


Рис. 2. Приклад розмітки ключових областей і точок.

З підвищенням рівня розвитку нейронних мереж популяризувалося їх використання у технологіях розпізнавання обличчя, голосу та емоцій. Більшість експертів визначають найкращим методом аналізу візуальних даних використання згорткових нейронних мереж. Для забезпечення роботи такого способу потрібна послідовність кадрів, а також умови вирішення додаткової задачі аналізу потоку кадрів. Дану задачу можна вирішити, додатково використавши рекурсивні мережі або архітектуру 3D-CNN, що накладає додаткові вимоги на апаратне забезпечення та збільшує час, необхідний для класифікації емоції.

Зважаючи на недостатній для сучасного рівня діджиталізації суспільства розвиток технологій визначення емоцій та потреба їх контролю, яка виникла під час пандемії, наукові дослідження та розробки у галузі інформаційних технологій потребують подальшого удосконалення.

Таким чином, можемо зробити висновок про те, що створення нової інтелектуальної системи, яка враховувала б більшість наявних напрацювань та теорій відомих науковців (таких як П. Екман, Р. Плутчик, У. В. Фрізен та Д. К. Хагер), а також відповідала б стандартам, заданих сучасним рівнем діджиталізації суспільства, нині є особливо актуальним.

Оглянувши роботи вітчизняних та світових фахівців, а також варіанти текстурних та геометричних підходів аналізу фотографій, вважаємо, що інтелектуальна система розпізнання емоцій людини має працювати за наступним алгоритмом:

1. Отримати на вхід фото або кадр відео;
2. Локалізувати область обличчя людини;
3. Провести текстурний аналіз;
4. Виділити ключові геометричні точки обличчя;
5. Провести геометричний аналіз ключових точок;
6. Зробити висновок про емоцію людини на фото, базуючись на результатах текстурного та геометричного аналізів.

Література:

1. COVID-19 definition. National Cancer Insitute [e.d.], URL: https://www.cancer.gov/publications/dictionaries/cancer-terms/def/covid-19
2. Oliver Selfridge`s Pioneering Program in Pattern Recognition. Jeremy Norman`s Historyofinformation. [e. d.] URL: https://www.historyofinformation.com/detail.php?id=3906
3. Dr. Paul Ekman. PaulEkmanGroup [e. d.]. URL: https://www.paulekman.com
4. Колесо Эмоций Роберта Плутчика. Лия Марк. Art Terra. [e. d.] URL: https://artterra.club/2017/11/13/колесо-эмоций-роберта-плутчика/
5. APA PsycNet(American psychological association) [e.d.]. URL: https://psycnet.apa.org/record/1981-25062-001

УДК 004.93'1, 004.032.26

ВІДНОВЛЕННЯ ЗАШУМЛЕНИХ ЦИФРОВИХ ЗОБРАЖЕНЬ

Гриценко В.В., Бушин І.М.

Черкаський національний університет ім. Богдана Хмельницького

Відновлення різкості зображення зі значними деталями з вхідного зашумленого зображення, вже давно є активною областю досліджень у сфері комп'ютерного зору. Зі збільшенням використання портативних пристроїв, таких як мобільні телефони та бортові камери, зашумленість стає усюдисущою проблемою, з якою потрібно боротися.

На практиці зображення часто спотворюються шумом, який з'являється на етапах їхнього створення та/або передачі. У наукових публікаціях значну увагу приділено визначенню різних видів шумів. На підставі аналізу літературних джерел встановлено, що шум є накладеною на зображення маскою пікселів випадкового кольору та яскравості. Причинами виникнення шуму на зображенні можуть бути несправність використовуваної апаратури, збої в роботі каналу зв'язку, несприятливі зовнішні умови та ін. Шуми, що виникають при цьому, підлягають класифікації з метою їх вивчення, формалізації для подальшого усунення чи мінімізації їх шкідливого впливу. Існує досить багато типів шумів, які негативно впливають на оброблення та аналіз зображень. Серед них – гаусівський шум, шум Пуассона, імпульсний шум, шум зерен фотоплівки, спекл-шум, шум Перліна та ін.

Фундаментальним завданням в області обробки зображень є ефективне видалення шуму, тобто побудова деякого наближення вихідного зображення по заданому (зашумленому) зображенню. Очевидно, що наше наближення має бути якомога ближче до вихідного зображення. Складність вирішення даного завдання істотно залежить від розглянутої моделі шуму. В принципі, чим більше ми знаємо про оператор спотворення і про функції шуму, тим ближче буде оброблене зображення до вихідного зображення [1].

Найчастіше використовуваними техніками шумозаглушення є класичні методи базовані на фільтрах, такі як адаптивна фільтрація, медіанна

фільтрація, математична морфологія, розмивання Гауса, вейвлет-перетворення а також фільтри Вінера та ін.

Опріч методів, заснованих на фільтрах, існують підходи до моделювання шуму, що не залежать від навчання, такі як EPLL, KSVD, BM3D, Марковські випадкові поля та Total Variation Denoising (загальна зміна шумоподавлення). Такі методи засновані на попередньому моделюванні шуму, і вони мають деякі недоліки, такі як обчислювальна складність та необхідність точного налаштування параметрів. Їх ефективність сильно залежить від попередніх знань про тип шуму (наприклад, гаусовий, «сіль та перець», тощо) та його статистичні властивості (наприклад, середнє значення та дисперсія).

На сьогодні обчислювальна техніка дуже добре розвинена, також добре опрацьовані алгоритми навчання глибоких нейронних мереж, які можливо застосовувати не тільки в задачах розпізнавання та класифікації. Останнім часом з розвитком технологій глибокого навчання з'явилася можливість реалізовувати також різні методи видалення шуму з зображень, що показують високі результати [2].

Методи глибокого навчання стали найефективнішими методами, що використовуються в багатьох реальних проблемах цифрової обробки зображень.

Як можна дізнатися з відповідної наукової літератури, алгоритми на основі нейронних мереж нічим не поступаються «ручним» просторовим алгоритмам, а в багатьох випадках перевершують їх за якістю. Вони використовуються як варіант заміни незалежних від навчання фільтрів, та підходів, що потребують попередніх знань про наявні шуми.

Перевагою таких підходів є те, що нема потреби підбирати особливе математичне перетворення для вирішення завдання шумозаглушення, нейронні мережі роблять це самі. Також, на відміну від колишніх технологій, нейромережі дають вищу якість вихідних зображень, зокрема, зменшують «замиленість» зображення і як правило, менше піддаються впливу нелінійних характеристик механізмів генерації шуму.

Серед таких підходів багатошарові перцептрони (MLP) протягом тривалого часу були одним з найбільш досліджуваних методів машинного навчання для відновлення зашумлених зображень [3-5]. З недавніми зростанням потужності обробки комп’ютерної графіки MLP замінили згорткові нейронні мережі (CNN), особливо стосовно завдань обробки зображень.

Згорткова нейромережа (ЗНМ) має спеціальну архітектуру, яка дозволяє їй максимально ефективно аналізувати образи. Сама ідея ЗНМ базується на чередуванні згорткових та субдискретизуючих шарів (pooling), а її структура є однонаправленою. ЗНМ отримала свою назву від операції згортки, яка передбачає, що кожен фрагмент зображення буде помножено на ядро згортки поелементно, при цьому отриманий результат повинен бути підсумований та записаний у схожу позицію вихідного зображення. Така архітектура забезпечує інваріантність розпізнавання відносно зсуву об'єкта, поступово

збільшуючи «вікно», на яке «дивиться» згортка, виявляючи все більші й більші структури і шаблони у зображенні.

Розглянемо моделі процесів спотворення та відновлення зображень.

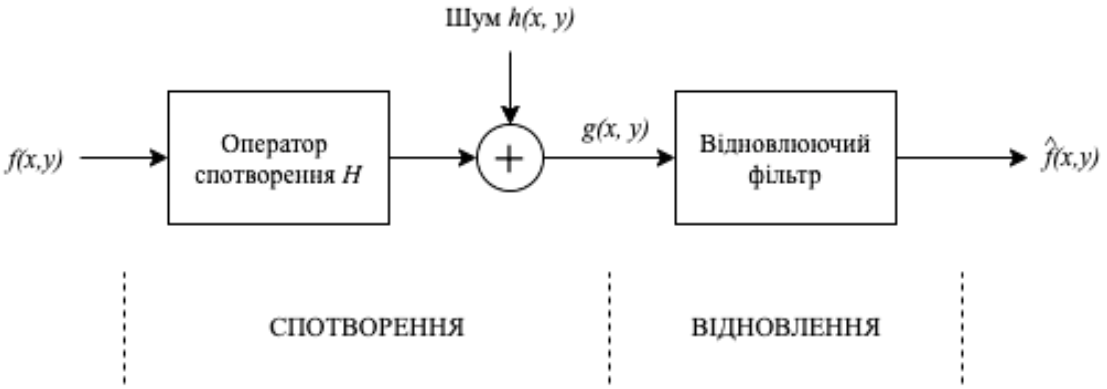


Рис. 1. Модель процесу спотворення/відновлення зображень.

Як показано на рис. 1, процес спотворення зображення полягає в тому, о деякий оператор спотворення H діє на вихідне зображення f(x, y), так, що в результаті додавання адаптивного шуму отримуємо спотворене зображення g(x, y).

Процес відновлення полягає в побудові деякого наближення вихідного зображення по спотвореному зображенню g(x, y), за деякою інформацією щодо оператора спотворення H і адитивного шуму η(x, y). При цьому наше наближення має бути якомога ближче до вихідного зображення. Спотворене зображення може бути представлене в просторовій області у вигляді:

$$g(x,y) = h(x,y) * f(x,y) + \eta(x,y), \quad (1)$$

де h(x, y) – функція, що представляє спотворює оператор в просторовій області, а символом «*» позначається згортка.

Згортка в просторовій області еквівалентна множенню в частотній області, тому вище наведена рівність (1) може бути представлена в частотної області у вигляді:

$$G(u,v) = H(u,v)F(u,v) + N(u,v), \quad (2)$$

де зазначені функції – Фур'є-образи відповідних їм функцій в (1).

Так як спотворення зображення це результат згортки, то для його відновлення необхідно знайти такий фільтр, застосування якого призводило б до зворотного процесу. Тому для позначення лінійного процесу відновлення часто використовується термін реконструкція (деконволюція) зображень. Аналогічно, фільтри для відновлення часто називаються реконструюючими фільтрами.

Нами передбачається розробка алгоритму видалення шумів на цифрових зображеннях за допомогою ЗНМ та здійснення його програмної реалізації засобами Python з використанням фреймворка для глибокого навчання Keras. Даний фреймворк є надбудовою над TensorFlow – відкритою платформою для машинного навчання, що має гнучку екосистему інструментів, бібліотек

та ресурсів, та дозволяє дослідникам просувати найсучасніші технології машинного навчання, а розробники можуть легко розгортати додатки.

Література:

1. J. Najeer Ahamed, V. Rajamani Design of hybrid filter for denoising images using fuzzy network and edge detecting // American Journal of Scientific Research, Issue 3, 2009, pp 5-14.
2. Методы удаления шумов на изображениях на основе применения искусственных нейронных сетей [Електронний ресурс] URL: https://www.lib.tpu.ru/fulltext/c/2010/C01/V2/170.pdf
3. H. C. Burger, C. J. Schuler, and S. Harmeling, “Image denoising: Can plain neural networks compete with BM3D?” in Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012, pp. 2392–2399.
4. C. J. Schuler, H. C. Burger, S. Harmeling, and B. Scho ̈lkopf, “A machine learning approach for non-blind image deconvolution,” in Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on. IEEE, 2013, pp. 1067–1074.
5. R. G. Pires, D. F. S. Santos, L. A. M. Pereira, G. B. De Souza, A. L. M. Levada, and J. P. Papa, “A robust restricted boltzmann machine for binary image denoising,” in 2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI). Ieee, 2017, pp. 390–396.

УДК 004.852, 004.896

ПЕРЕТВОРЕННЯ КОМАНДИ НАТУРАЛЬНОЮ МОВОЮ НА ВИКЛИК ПІДПРОГРАМИ

Бочок В.О.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Насамперед визначимо проблему. У ході виконання дипломної роботи виникла необхідність обробляти текстовий запит користувача, сформований англійською мовою, та визначати, яку команду з наперед відомої множини найімовірніше слід обрати. Тобто, необхідно зіставити текстову команду з відповідним наявним класом. У даному випадку англійська мова обрана тільки тому, що вже наявні у вільному доступі великі текстові корпуси (набори даних), часто з додатковими метаданими (залежно від набору це можуть бути частини мови, ключові слова тощо). Вибір мови не є принциповим, але, варто враховувати, що дана задача стосується природних мов, оскільки саме такі мови людина використовує у повсякденному спілкуванні.

Прикладами систем, що використовують описану схему роботи, є розумні помічники, такі як: Google Assistant, Siri, Amazon Alexa і т.д.

Перераховані застосунки отримують команду користувача (голосом або текстом), обробляють її та виконують певну дію.

Умова використання природної мови призводить до певних обмежень: неоднозначність (залежність від контексту), вільна структура й варіативність (однаковий або дуже схожий сенс може бути переданий різними способами). У той час, коли штучні мови через їхню однозначність, чітку структуру й лаконічність можуть бути оброблені системами типу транслятора або ж простими умовними конструкціями, природні мови вимагають використання методів Natural Language Processing (NLP) і не гарантують 100% точності. Методи NLP у деяких випадках наближаються до людської точності, а іноді й випереджають спеціаліста.

Залежно від умов та вимог розв'язання завдання можливо звести до класифікації, регресії, ранжування тощо. Відповідно, розв'язувати проблему можливо методами навчання з вчителем (supervised learning), навчання без вчителя (unsupervised learning) та іншими.

Якщо задано певний фіксований набір команд, а також наявна база запитів, кожному з яких зіставляється певна команда, — завдання класифікації. Такий підхід за умови наявності достатньо великого набору даних може давати найбільшу точність, адже відрізняється прогнозованністю і дає можливість оптимізувати систему до певних сценаріїв.

У випадку, коли набір команд не є фіксованим, необхідно розробити унікальний метод порівняння будь-яких граматично і семантично вірних текстів. У такому випадку набір команд буде описуватися природною мовою, а запити користувача порівнюватися з цим описом. Надалі розглядаються лише короткі тексти (1-2 речення), проте, деякі методи можуть бути задіяні і для великих текстів з різним результатом. Успіх для коротких текстів не гарантує його для великих.

У ході порівняння гарним варіантом є отримання для кожної пари текстів певного числа, що відображатиме міру "однаковості". Це надасть можливість вирішити, які пари є більш релевантними, і відповідно розглядати тільки їх. Надалі таке число буде називатися метрикою схожості.

Варто зазначити, що запит користувача може бути зовсім не актуальним до набору наявних команд, так що кращим варіантом реакції системи буде повідомлення, що такої команди не знайдено, або ж виконання певної підпрограми на цей випадок, як, наприклад, Інтернет-пошук. Для реалізації такої логіки необхідно не тільки мати міру схожості, а й розуміти її максимальні-мінімальні значення. Наприклад, можна визначити таку метрику в діапазоні від 0 до 1, або ж від -1 до 1. Тут -1 означає, що тексти або ж слова протилежні за значенням, 0 — взагалі не стосуються один одного, 1 — ідентичні. Ідентичність залежно від метрики може означати як повний побуквений збіг, або ж "однакове" семантичне значення.

У роботі не розглядаються методи, засновані на порівнянні наявності певних слів, n-gram (груп слів) чи морфологічних конструкцій у двох текстах. Робота фокусується більше на семантичній схожості текстів.

Більшість розглянутих методів базуються на перетворенні текстів у багатовимірний вектор чисел. Такий підхід дає можливість застосовувати багато математичних операцій до оброблюваного тексту. Ідея такого вектора, що слово або речення в такому векторі кодує в багатовимірному просторі семантичне значення. А, відповідно, тексти зі схожими значеннями у вигляді таких векторів будуть розміщуватися у просторі відносно поряд. У дипломній роботі розглянуто кілька методів перетворення тексту у вектор, переваги і недоліки кожного. Варто зазначити лише, що вектори мають бути досить великими, аби з найменшими втратами закодувати семантичний зміст всіх можливих (або найбільш ймовірних) текстових конструкцій [1].

Для визначення метрики подібності використовується косинусна відстань, яка набуває значень від -1 до 1. Значення косинусної відстані означає косинус кута між двома векторами. Довжина векторів ігнорується, що частково розв’язує проблему порівняння текстів різної довжини.

Література:

1. Efficient Estimation of Word Representations in Vector Space [Електронний ресурс] / T. Mikolov, K. Chen, G. Corrado, J. Dean. – 2013. – Режим доступу до ресурсу: https://arxiv.org/abs/1301.3781.

UDC 004.6

AN OVERVIEW OF BUSINESS INTELLIGENCE CORE CAPABILITIES AND SOFTWARE PLATFORMS

Orlovskyi D.L., Kopp A.M.

National Technical University “KhPI”

When someone mentions terms “data analysis” or “data visualization”, also if referring “buzzwords” shown in Fig. 1, the term Business Intelligence (BI) almost always appears. Unfortunately this term still does not have unified formal definition, while generally referred as collection of methods and software tools that support translation of raw data taken from transactional systems (e.g. from ERP – Enterprise Resource Planning, CRM – Customer Relationship Management, and HR – Human Resource systems) or semi-structured files into a human-readable view suitable for analysis and decision making (e.g. using OLAP – Online Analytical Processing tools for multi-dimensional data querying or dashboards for data visualization). Usually transactional data is preliminary aggregated and stored in so-called data warehouses, denormalized for more efficient data processing. Whereas Gartner IT Glossary [1] refers BI as “the umbrella term that includes applications, infrastructure, tools, and best practices that enable access to and analysis of information in order to improve and optimize decisions and performance”. Taking into account authority and value of such source, we may

stop on this particular reference, at least in this paper. Basic BI components and how they are interrelated are represented in Fig. 1 below.

When covering BI major components and capabilities, we also are going follow a common approach existing in the industry and consider the topics of data analysis and visualization through the prism of Business Intelligence. Main work of gathering together all of the related methodologies, governance practices, and software tools was already made by Forrester Research, Inc. [2] and presented in its architectural landscape of the BI stack. However, it will be cumbersome and sophisticated to go along all of the described BI architectural layers and components. Hence, we could briefly overview topics related to reporting and dashboards, ad-hoc querying, OLAP (Online Analytical Processing), data mining, and scorecards.

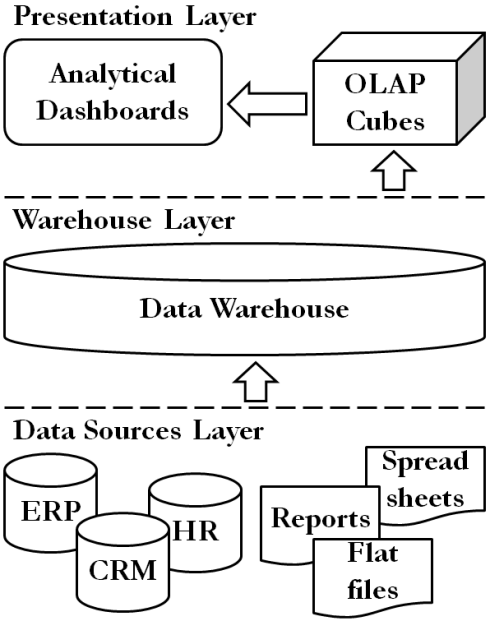


Fig. 1. Simplified and generalized Business Intelligence architecture.

Let us start this brief review with the core BI capabilities. There are three major capabilities though:

- data extraction;
- data storage;
- data analysis.

By data analysis we may understand more specific data-handling approaches and techniques, which include data visualization, data exploration, data discovery or combinations of these approaches, or even all of them together.

However, there are concrete techniques exist that support these capabilities by providing the following services:

1. Reporting – taking raw transactional data and turning it into information used to make intelligent business decisions.

2. Dashboards – pages where reports, graphs, and charts can be placed in order to create a central location for critical business information.

3. Ad-hoc querying – real-time reporting in order to explore business questions while looking through the data.

4. OLAP – a technique, which aims at efficient multidimensional processing of large data volumes to provide fast and interactive answers to large aggregate queries.

5. Data mining and machine learning – approaches, which serve to dig through huge amounts of data and come up with predictions.

6. Scorecards – specialized reports designed to measure progress toward goals by various groups of measures.

By using Gartner's Magic Quadrants of Analytics and BI Platforms [3], we can define leading software vendors and their Business Intelligence products, which are Power BI by Microsoft, QlikView by Qlik, and Tableau by the same company.

With Power BI various data sources can be connected to the dashboards: there are third-party applications, cloud services, streaming data, and Excel spread sheets. Through the API (Application Programming Interface), users can connect their own applications to the Power BI dashboards. Interactive dashboards are available on any device (desktop workstation or smartphone, or tablet) and can display real-time data. Users can share information in several ways, since Power BI works on all platforms: cloud, desktop and mobile [4].

QlikView software provides the uniform analytics workflow, which works in all of its versions. Qlik data visualization products provide fast and interactive data visualization. Qlik can be used in collaborative mode by several members of a team, which can share to each other any applications (dashboards or reports). Created data visualizations can be used for samples to further reuse and studies [5].

Tableau users can create software components (dashboards) for reporting and analytics. This system works with all devices where there are data streams exist, so users and developers do not need to worry about hardware or software requirements. Information panels may be provided with access to data warehouses, so applications for creating dashboards can be created by business users themselves. Several users can work on a Tableau report in collaborative mode [6].

In general, reviewed BI platforms support heterogeneous data sources and, like Power BI are available as desktop, mobile, and cloud solutions. Moreover, all these software platforms provide interactive data visualization and reporting, and support collaborative development mode by several users (team members), for example like QlikView. Finally, like the Tableau, all these BI platforms were created to be used by non-technical business users after short training sessions, which bridges the gap between IT (Information Technology) and business organizational domains.

References:

1. Gartner Glossary, Analytics and Business Intelligence. URL: https://www.gartner.com/en/information-technology/glossary/business-intelligence-bi (date of access: 28.04.2021).

2. Evelson B. The Forrester Wave Enterprise Business Intelligence Platforms. URL: https://www.slideshare.net/Cezar_Cursaru/The-Forrester-Wave-Enterprise-Business-Intelligence-Platforms-Q3-2008 (date of access: 29.04.2021).
3. Levy-Yurista G. Gartner Magic Quadrant for Analytics and BI Platforms. URL: https://dzone.com/articles/the-next-generation-of-bi-is-here-and-so-is-the-20 (date of access: 30.04.2021).
4. Microsoft, What is Power BI? URL: https://powerbi.microsoft.com/en-us/what-is-power-bi/ (date of access: 01.05.2021).
5. Qlik, Data Analytics & Data Integration Solutions. URL: https://www.qlik.com/us/ (date of access: 01.05.2021).
6. Tableau, Business Intelligence and Analytics Software. URL: https://www.tableau.com/ (date of access: 02.05.2021).

УДК 004.85

ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ НАВЧАННЯ РОБОТА В ІГРОВІЙ ФОРМІ

Любченко К.М., Ільченко Д.Ю.

Черкаський національний університет ім. Б. Хмельницького

Системи з елементами штучного інтелекту набувають все більшої популярності. Одним з видів таких систем, де розробники намагаються впровадити штучний інтелект, є комп’ютерні ігри, які на зорі свого становлення були аркадними розвагами. Згодом стали з’являтися ігри, відносно яких можна говорити про занурення, опрацювання світу, про атмосферу. Крім правдоподібної фізики, користувач зазвичай вимагає правдоподібної поведінки ігрових персонажів, керованих комп'ютером, що забезпечується симуляцією наявності штучного інтелекту у відповідному програмному продукті.

Проектування інтерактивної системи навчання робота в ігровій формі описано в [1].

При створенні відеогри або іншого ігрового продукту одним з найперших питань є вибір рушія. Залежно від того, чи буде вибраний рушій готовим, чи буде написаний власноруч, буде залежати такі важливі аспекти, як мова програмування, кількість і складність роботи, необхідність модифікувати вже існуючі аспекти рушія. На нашу думку, найвдалішим вибором для розроблюваної системи є рушій Unity. Він є безкоштовним, має більшість функцій, які будуть необхідні при створенні гри, також має велику кількість аддонів з відкритим кодом для більш спеціалізованих задач.

На початку розробки було створено шутер з видом зверху для відладки всіх аспектів гри: геометрії об’єктів на рівні, коректної їх фізичної взаємодії, поведінки супротивників на рівнях, правильного функціонування дій гравця. Рушій Unity має фізичну модель поведінки, яку можна застосувати для об’єктів. Для супротивників прописано алгоритмічний штучний інтелект, що

базується на алгоритмі A*, імплементацію якого було взято з відкритого джерела [2]. Алгоритм A* – алгоритм для пошуку найкоротшого шляху між двома вершинами графу – був використаний для навігації противників по ігровому полю (рис. 1). До цього алгоритму додано власний, який надає супротивникам поле зору та можливість шукати гравця при втраті зорового контакту з ним.

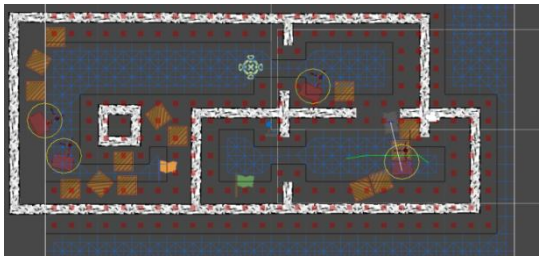


Рис. 1. Дебаг екран. Зеленою лінією позначений найкоротший шлях до останнього місця, де противник бачив гравця.

Для створення тренувальних рівнів гравцем був використаний вбудований функціонал користувацьких інтерфейсів рушія Unity. У цьому редакторі у гравця є можливість додати об’єкти у вільному порядку для тестування пристосованості і подальшого тренування нейронної мережі (рис. 2).

Рис. 2. Редактор рівнів.

Для вимірювання пристосованості кожної мережі була додана система контрольних точок, за проходження яких додається по тисячі очок, які поступово віднімаються. Таким чином надається можливість задати перші напрямки для мережі і «мотивувати» її проходити контрольні точки швидше. Також немає можливості пробігати супротивників, адже вони при «зоровому» контакті починають атакувати гравця, а втрата всіх життів теж має негативні наслідки до забігу (рис. 3).

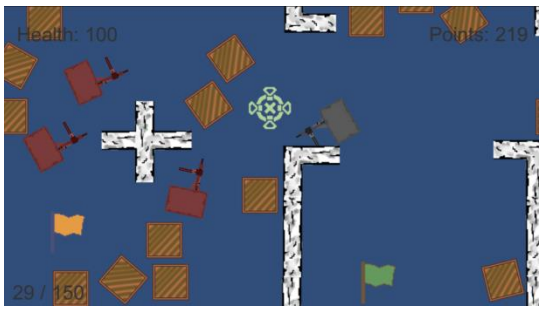


Рис. 3. Ігровий процес.

Реалізована для навчання робота нейронна мережа була створена у вигляді окремого класу, який можна ініціалізувати в будь-якому об’єкті гри для відстежування успіхів мереж. Попередня імплементація механік шутера з видом зверху також використовується в нейронній мережі. В результаті вона обчислює, які дії (натиски на кнопки миші та клавіатури, положення миші на екрані) буде симулювати програма. На вхід нейронна мережа буде отримувати інформацію про об’єкти, супротивників на рівнях тощо.

Дана нейронна мережа еволюціонує мутаціями вагів нейронних зв’язків. Для прискорення швидкості навчання спочатку створюється масив об’єктів нейронної мережі, які будуть еволюціонувати певну кількість поколінь. Серед них 20% найбільш успішних вибираються для подальшої еволюції, а 80% менш успішних замінюються копіями більш успішних мереж.

Отже, у результаті проведеної роботи створена інтерактивна система навчання робота, що наочно дозволяє спостерігати роботу нейронної мережі, на базі якої здійснюється симуляція штучного інтелекту в ігровій формі.

Література:

1. Любченко К. М. Проектування інтерактивної системи навчання робота в ігровій формі / К. М. Любченко, Д. Ю. Ільченко // Автоматизація та комп'ютерно-інтегровані технології у виробництві та освіті: стан, досягнення, перспективи розвитку: матеріали Всеукраїнської науково-практичної Internet-конференції, 11-21 березня 2021 року. – Черкаси, 2021. – С. 292-294.
2. A* Pathfinding Project [Електронний ресурс]. – Режим доступу: https://arongranberg.com/astar/

УДК 004.75

ОСНОВНІ КОНЦЕПЦІЇ ЗАХИСТУ ВІД ПЕРЕНАВАНТАЖЕННЯ В ІНФОРМАЦІЙНИХ РОЗПОДІЛЕНИХ СИСТЕМАХ

Білоус І.В., Нагорний П.В.

Національний університет «Чернігівська політехніка»

В сучасному світі інформаційні розподілені системи набувають все більшого поширення. Зважаючи на виключне значення таких систем, важливо досліджувати їх атрибути якості. До таких атрибутів прийнято відносити цілісність, доступність та стійкість до мережевих проблем. У відповідності до відомої CAP-теореми, тільки двоє з трьох атрибутів можуть бути досягнуті в реальній практиці. При цьому більшість спеціалістів погоджується надавати першочергове значення саме стійкості до мережевих проблем, адже такі проблеми мають недетермінований ймовірнісний характер та не можуть бути повністю усунені за будь-якого розвитку технологій. Саме тому вибір зазвичай здійснюється з PA та PC типів розподілених систем [1].

Важливість стійкості до мережевих проблем обґрунтована неодноразово, тому розроблено ряд підходів до її забезпечення, серед яких повторні запити, які переадресуються на інший робочий сервер. Але такі повторні запити створюють додаткове навантаження на систему. Слід врахувати також високий рівень навантаження на розподілені системи за звичайних умов роботи. Саме тому існує значний ризик перенавантаження в інформаційних розподілених системах [1]. Розглянемо основні концепції захисту від такого перенавантаження.

Більшість концепцій захисту від перенавантаження в інформаційних розподілених системах базується на принципі обмеження частоти запитів. Обмеження може відбуватися в двох площинах: серед основної маси запитів та серед повторних запитів (площину повторних запитів слід розглядати окремо через наявність позитивного зворотного зв'язку [2]). Доречним є комбінування обмеження частоти запитів в обох площинах.

Розглянемо спочатку основні концепції обмеження частоти запитів, сформованих за стандартного режиму роботи. До таких концепцій можна віднести: алгоритм «маркерної корзини», алгоритм фіксованого вікна, алгоритм ковзного вікна [2].

Алгоритм «маркерної корзини» полягає у розробці додаткової структури, дещо схожої за принципом роботи на протікаюче відро, для обмеження частоти запитів. В «корзині» знаходиться деяка кількість маркерів. Обсяг маркерів постійно збільшується з певною встановленою регульованою швидкістю R. Кожен наступний запит, що надходить в систему, запитує наявність маркера в корзині. У випадку його наявності запит зменшує загальну кількість маркерів на один та надходить на подальшу обробку [3]. У випадку відсутності маркерів у корзині запит відхиляється системою. До переваг цього способу можна віднести легку зміну пропускної здатності каналу, постійність середньої частоти запитів в часі. До недоліків можна віднести додаткове навантаження на постійну роботу «корзини з маркерами».

Алгоритм фіксованого вікна полягає у розбитті часової шкали на однакові інтервали, протягом кожного з яких працює лічильник кількості запитів. Для кожного запиту, що надійшов до системи, визначається його часове «вікно» та порівнюється значення лічильника з максимальним значенням. Відповідно, приймається рішення щодо подальшої обробки запиту [3]. До переваг цього способу можна віднести легкість організації ті інтеграції. До недоліків можна віднести стабільність середньої частоти запитів лише всередині вікон, а не на межі.

Алгоритм ковзного вікна схожий на алгоритм фіксованого вікна, але проблему неоднорідності частоти запитів на межі вікон вирішує шляхом додавання до лічильника поточного вікна зваженого значення лічильника попереднього вікна. Таке рішення не є ідеальним, адже спирається на припущення рівномірності розподілу частоти запитів у попередньому вікні, але на практиці дає прийнятну точність [3]. Зважаючи на простоту реалізації алгоритму ковзного вікна, цей підхід є найбільш широко застосовним серед

концепцій обмеження частоти запитів, сформованих за стандартного режиму роботи.

Розглянемо також основні концепції обмеження частоти повторних запитів. До таких концепцій можна віднести: бюджет повторних запитів, зупинка повторних запитів, затримка повторних запитів [4].

Дієвим та легким у реалізації підходом є обмеження кількості повторних запитів на стороні клієнта. Для цього необхідно вести лічильники звичайних запитів та повторних запитів, та порівнювати їх значення [4]. Недоліком цього способу є наявність додаткових навантажень на стороні клієнта.

Підхід зупинки повторних запитів є дієвим якщо перенавантаження спостерігається на всіх серверах розподіленої системи. Необхідно вести для кожного запиту лічильник спроб. Якщо маємо велику кількість запитів зі значною кількістю спроб, скоріш за все, система перенавантажена [4]. Недоліком цього підходу є додаткове навантаження на сторону клієнта у вигляді необхідності аналізу кількості спроб попередніх запитів.

Ймовірність виникнення повторної помилки системи відразу після попередньої досить значна. Тому ефективним та простим у реалізації є підхід здійснення затримки між повторними запитами. Затримку слід визначати за експоненціальною функцією, адже таким чином вдається досягти поступового «затухання» процесу надсилання повторних запитів. Щоб запобігти колізій від одночасного надсилання різних повторних запитів, до отриманого часу затримки слід додавати певну випадкову величину [4].

Таким чином, існують різні підходи захисту від перенавантаження в інформаційних розподілених системах. В залежності від конкретної задачі той чи інший підхід може виявитися найбільш ефективним. В той же час, задача розробки алгоритмів захисту інформаційних розподілених систем від перенавантаження не є вирішеною до кінця та потребує подальших досліджень.

Література:

1. Бейер Б., Джоунс К. Site Reliability Engineering. Надежность и безотказность как в Google. Спб.: Питер. 2019. 596 с.
2. Рубцов Д. В. Методы защиты от перегрузок в распределенных системах обработки информации. Информатика и вычислительная техника и управление. Серия: Естественные и технические науки. 2020. № 4-2. С. 53-60.
3. Абросимов Л. И. Методы проектирования и анализа сетей ЭВМ. Изд. Palmapium academic publishing. 2013. 210 стр.
4. Царев Р. Ю. К проблеме оценки надежности сложных программных систем. Журнал Сибирского федерального университета. Серия: Техника и технологии. 2015. № 1. С. 33-47.

УДК 004.042

ЗНАХОДЖЕННЯ СТОРОННІХ ВКЛЮЧЕНЬ У ПОСЛІДОВНОМУ ПОТОЦІ ІНФОРМАЦІЇ

Мисюра Ю.О., Розломій І.О., Ярмілко А.В.

Черкаський національний університет ім. Богдана Хмельницького

Існують задачі, розв'язання яких потребують аналізу деякого послідовного потоку даних. Такий потік можна отримати, наприклад, з камери відеоспостереження або мікрофона. До таких задач відноситься знаходження сторонніх включень – наприклад, нових об'єктів на зображенні – у даному нам потоці інформації.

Одним із способів розв'язання подібної задачі є використання алгоритмів Deep Learning та засобів сторонніх бібліотек [1]. Але у нього є один важливий недолік – він підходить лише для одного типу даних (відео, звук і т.д.).

Тому у даній роботі ми спробуємо досягти незалежності алгоритму розв'язання задачі від типу вхідного потоку.

Відомо, що дані будь-якого типу можна представити у двійковому вигляді. Звідси слідує, що потік даних представляється у вигляді двійкового потоку. Але комп'ютерна система не може обробляти весь неперервний потік відразу. Це означає, що дані необхідно подавати порціями деякого заздалегідь визначеного розміру – назвемо ці порції фреймами.

Наш метод буде ґрунтуватися на порівнянні трьох сусідніх фреймів. Розглянемо його докладніше.

Задамо нашій програмній системі буфер розміром у три фрейми. Це потрібно для того, щоб програма приймала тільки три фрейми за раз. Далі кожен із фреймів ділимо на менші фрагменти даних, розмір кожного з яких не перевищує k байт, де k дорівнює:

$$k = \log_2 Size_f$$

де $Size_f$ – розмір фрейма.

Далі ми обчислюємо хеші відповідних фрагментів поточної трійки фреймів (тобто ми на кожному кроці обчислюємо хеші *i*-того фрагмента першого фрейма, *i*-того фрагмента другого фрейма та *i*-того фрагмента третього фрейма). Потім ми для кожного з цих трьох хешів обчислюємо відносне відхилення від середнього:

$$d_{ij} = \frac{\left| h_{ij} - \overline{h_{ij}} \right|}{\overline{h_{ij}}}$$

де h_{ij} – хеш і-того фрагмента j-того фрейма з трійки, $\overline{h_{ij}}$ – середнє арифметичне трійки обчислених хешів.

Якщо $d_{ij} > 0{,}368$, то ми вважаємо цей фрагмент відповідного фрейма стороннім включенням. Константа 0,368 була обрана, тому що $0{,}368 = \frac{1}{e}$, де $e \approx 2{,}718$.

Для практичної реалізації та демонстрації роботи нашого алгоритму ми використали мову C#, у якій є клас FileStream [2], який має усі необхідні нам методи. Але для реального впровадження подібного алгоритму слід використати мову C або C++, оскільки така програма на цих мовах у загальному випадку матиме більшу швидкодію.

Універсальність нашого алгоритму є одночасно перевагою і недоліком. Недолік полягає у тому, що він не враховує специфічні особливості формату вхідних даних, тому його точність може бути менша, ніж у алгоритмів, пристосованих до конкретного формату даних.

Отже, ми дослідили один із способів знаходження сторонніх включень у деякий послідовний потік даних. Отримані результати досліджень можуть бути корисні у розробці систем безпеки та аналізу даних різних типів.

Література:

1. Python: распознавание объектов в реальном времени : веб-сайт. URL: https://proglib.io/p/real-time-object-detection (дата звернення: 05.05.2021).
2. FileStream. Чтение и запись файла : веб-сайт. URL: https://metanit.com/sharp/tutorial/5.4.php (дата звернення: 05.05.2021).

УДК 004.8

USE OF ARTIFICIAL INTELLIGENCE IN THE STOCK MARKET

Budiakova O.

Kyiv National University of Technology and Design

Budiakov V.

"SMART TEAM" LIMITED LIABILITY COMPANY, Kyiv

Artificial intelligence has not yet captured the world, but already subjugates the world of digital marketing. And this should not be alarming, because artificial intelligence is beginning to perform simple tasks for people and opens up many new opportunities. What are its benefits for digital marketing? It can analyze consumer behavior and search patterns, data from social networks and blogs, help companies determine how customers find products or services, and more.

In the near future, artificial intelligence will be a driving force for many areas of marketing and will give recommendations on products, communicate with consumers, create content. Businesses that have implemented artificial intelligence will be able to reduce staff costs and accelerate growth by bypassing their competitors. And those who are late with the use of this tool are likely to be

uncompetitive. Stock markets are no exception. But in the Ukrainian stock market [1] the use of electronic technologies is not one of the characteristic features that provide high dynamics of operations, significantly speed up settlements, expand the range of participants and reduce risks on the stock exchange.

The stock exchange is an organized, regularly functioning securities market. Stock exchanges are designed to: mobilize temporarily free funds of enterprises and individuals through the sale of securities, to facilitate the movement of money capital between different economic entities [2].

Artificial intelligence is a branch of computer science that includes the development of methods for modeling and reproducing with the help of computers certain functions of human creativity, solving the problem of presenting knowledge in computers and building knowledge bases, creating expert systems, developing so-called intelligent robots.

The history of using artificial intelligence in working with investment projects began in the 1970s in America. Nowadays, with the development of computer capacity, the scope of their use is developing: now machines often make certain investment decisions on their own in many companies that manage the stock exchange. The neural network, on the technology of which the work of the employee of the company managing the stock exchange is based, is developed on the basis of previously obtained real data on the work of the stock market and already implemented investment projects. Each portfolio investment situation is mathematically a common set of variables, and this is how a robot perceives certain events and numbers. Going through similar events in his memory, he finds the model of behavior that best fits the relevance and implements it. Thus, investment decisions made by the machine consist of an analysis of real data and the experience of predecessors. The robot is able to process absolutely all market indicators, information about which has entered the system, and it can find the relationship between events that a person, even with a very rich imagination can not connect. The disadvantages of artificial intelligence on the stock exchange are almost non-existent: it forms the optimal portfolio at the moment, which brings maximum benefits. The human factor in investment management is excluded, incomes increase. The only disadvantage of the car – it can not guess, intuitively making a illogical but successful decision. This is the prerogative of man. At the same time a person takes risks, and the machine seeks to minimize risks. Many experts are absolutely convinced that soon the trade of traders will come to naught, because even now the speed of transactions is no longer available to the human mind, let alone manual transactions. This may well be the case, as evidenced by the actions of some exchanges, which not only certify robots and impose fees for exceeding the transaction limit, but also create special tools, such as increasing the commission for active robots.

But among professional participants in the securities market is mostly skeptical about automatic trading. As a rule, traders are confused by three things:

1. A person is not able to constantly monitor the accuracy of data issued by a mechanical trading system or an exchange robot.

2. In the event of a system failure, the program can generate incorrect signals for the conclusion of transactions for a long time, which will lead to losses.

3. Many players believe that online trading should combine a systematic approach and personal intuition, ie subjective assessment [3; 4].

The global trend allows us to conclude that trading robots on stock exchanges will become more and more. It is possible that long-term trading will give way to short-term, in which positions will open not even for seconds, but for milliseconds. Traders are already using pre-market trading to avoid competition with artificial intelligence. In terminology, a premarket is a pre-market period from the English pre-market, ie "to the market". But, do not be afraid that stock exchanges will turn into large halls with computers without people – the work is effective only in trading in highly liquid securities.

References:

1. Закон України Про цінні папери та фондовий ринок. (Відомості Верховної Ради України (ВВР), 2006, № 31, ст. 268) URL: https://zakon.rada.gov.ua/laws/show/3480-15
2. Словник економічних термінів. Букви Т-Я, 2013. URL: http://epi.cc.ua/fondovaya-birja-34138.html
3. Будякова О.Ю., Будяков В.Є. Використання штучного інтелекту на фондовому ринку. Marketing of innovations. Innovations in marketing (2020). Materials of the International Scientific Internet Conference (December, 2020). Bielsko-Biala: WSEH. [E-edition]. С. 93-95. http://dspace.puet.edu.ua/bitstream/123456789/10012/1/Marketing_innovations_2020_WSEH__%20%281%29.pdf
4. Будякова О.Ю., Будяков В.Є. Сучасні тенденції підготовки фахівців фондового ринка. Збірник тез IV Всеукраїнської науково-практичної конференції "Нові інформаційні технології управління бізнесом". – Київ: Спілка автоматизаторів бізнесу, 11.02.2021. – 532 с. С. 45-49. http://dkrkm.org.ua/cache/konfer/2021/zbirka_tez_2021.pdf

УДК 004.42

ПОРІВНЯННЯ АЛГОРИТМІВ PPO ТА SAC ДЛЯ ЗАДАЧІ ПОШУКУ АГЕНТОМ ЗАДАНОГО ОБ'ЄКТУ

Дядюн С.В., Супруненко О.О.

Черкаський національний університет ім. Богдана Хмельницького

Задача пошуку агентом заданого об'єкту вирішується у численних програмних системах з кількома користувачів-агентів, які взаємодіють між собою та з системою у певному графічному середовищі. У даній роботі представлені результати у середовищі Unity, в якому реалізована бібліотека ML-Agents, що дозволяє перетворити будь-яку сцену Unity на навчальне

середовище та сформувати поведінку агентів за допомогою різноманітних алгоритмів машинного навчання [2]. Крім того, це дозволяє передавати навченого агента у сцени Unity, щоб у подальшому використовувати його у проектах.

Дана бібліотека надає такі основні функціональні можливості. Визначення агентів, як активних сутностей, поведінка яких буде вивчатися; в даному контексті агенти – це сутності, які генерують певну спостережувану поведінку (за допомогою датчиків чи характеристик), можуть навчатися, здійснюють дії та отримують зворотну реакцію від навколишнього середовища (автономні інтелектуальні агенти). Визначення поведінки (пов'язаної з агентом чи гіпотетичної): певні процеси, які визначають, як повинен діяти агент при різних зовнішніх впливах; кілька агентів можуть спільно використовувати одну і ту ж поведінку, а також у сцені може використовуватись декілька процесів різної поведінки агентів. Демонстрація навчання агента певній поведінці та можливість запису цього процесу для його аналізу (додаткові засоби). Вбудовування агента з певною поведінкою чи поведінки, яку можна застосувати для іншого агента, у сцену за допомогою Unity Inference Engine; вбудована поведінка може застосовуватися до агента у отриманому вигляді чи розвиватися з допомогою подальшого навчання агента [1].

Proximal Policy Optimization (PPO) – алгоритм, основною мотивацією якого є питання: як отримати найбільше можливе поліпшення результатів за один крок на основі тих даних, що вже є і при цьому не погіршити поточні показники ефективності. У роботі застосовується варіант методу PPO, який використовує підхід на підтвердження того, що нова поведінка не надто сильно відрізняється від старої [2]. Наскільки випадковими будуть результати наступного кроку обчислень залежить від початкових умов і процедури навчання. Такий підхід може створити так званий «локальний оптимум».

Soft Actor Critic (SAC) – алгоритм, в якому оптимізується стохастична політика дій, не спираючись на існуючу політику. Основною особливістю SAC є регуляризація ентропії [3]. Політика дій створюється таким чином, щоб оптимізувати компроміс між очікуваною нагородою і ентропією, яка є показником випадковості в політиці дій. Контроль над ентропією має допомогти уникнути ситуації з «локальним оптимумом».

Порівняння алгоритмів PPO та SAC для задачі пошуку агентом заданого об'єкту проводилось у двох серіях дослідів, у другій серії дослідів ставилась задача складніша за попередню, що дозволило порівняти ефективність алгоритмів в різних умовах складності. Для кожного алгоритму проводились кілька дослідів і з них вибиралися найвдаліші, за найменшою кількістю ітерацій, за яких було досягнуто ефективність навчання 90 та 99%.

Для порівняння алгоритмів досліди проводитись у двох графічних середовищах (рис. 1, а та б). У першій серії дослідів (рис. 1, а) в центрі поля знаходився агент для навчання і відносно нього – чотири позиції, в яких може з'явитись об'єкт для пошуку: зліва, справа, спереду та позаду від

агента. Якщо агент знаходить цільовий об’єкт, то результат вважається позитивним, при зіткненні зі стінками або якщо ітерація навчання не завершується за 30 секунд, результат вважається негативним.

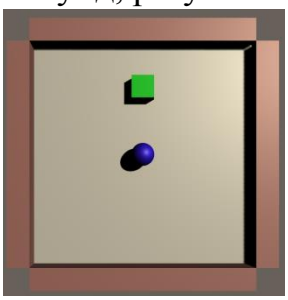
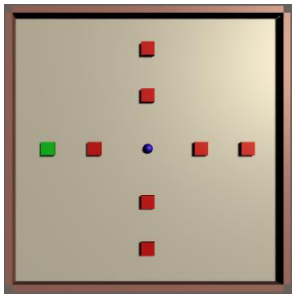
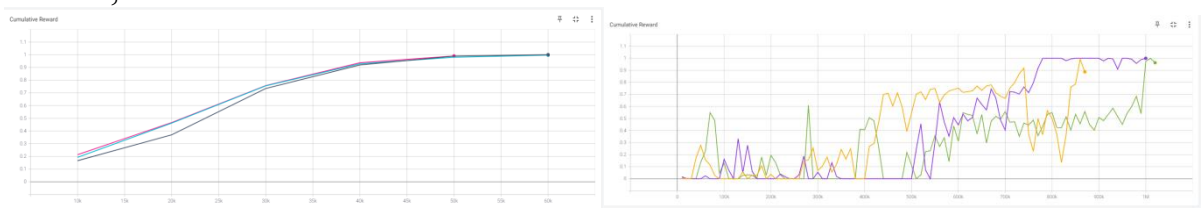
а)  б) 

Рис. 1. Середовища для проведення дослідів:
а) – для першого, б) – для другого досліду.

На рис. 2 та у табл. 1 відображено результати навчання нейронної мережі алгоритмами PPO та SAC відповідно у першій серії дослідів, де к - тисяча, м – мільйон.



а) б)

Рис. 2. Графіки навчання за алгоритмом
PPO (а) та SAC (б) у першій серії дослідів.

Таблиця 1.

Навчання алгоритмом PPO та SAC у першій серії дослідів

Ефективність (0 … 1)	Дослід 1, ітерацій	Дослід 2, ітерацій	Дослід 3, ітерацій	Кращий результат
Алгоритм PPO				
0.9	40к	40к	40к	40к
0.99	50к	60к	50к	50к
Алгоритм SAC				
0.9	740к	770к	1m	740к
0.99	860к	780к	1.1m	780к

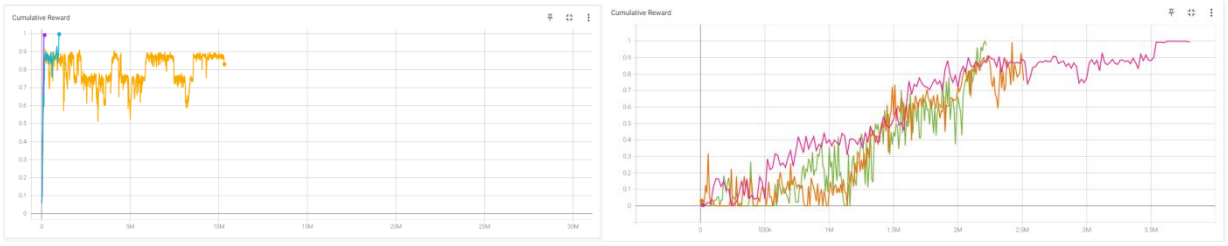
У другій серії дослідів використовувалось поле для навчання агента, більше та складніше ніж попереднє. В ньому точок появи шуканого об’єкту вісім (рис. 1, б), вони розташовані схожим чином, як в першому досліді, але перші чотири позиції знаходяться ближче до агента, а інші чотири далі.

У табл. 2 та на рис. 3 відображено результат навчання нейронної мережі алгоритмом PPO та SAC відповідно у другій серії дослідів.

Таблиця 2.

Навчання алгоритмом PPO та SAC у другій серії дослідів

Ефективність (0 … 1)	Дослід 1, ітерацій	Дослід 2, ітерацій	Дослід 3, ітерацій	Кращий результат
Алгоритм PPO				
0.9	640к	120к	200k	120к
0.99	990к	140к	-	140к
Алгоритм SAC				
0.9	2.08m	2.2m	2.15m	2.08m
0.99	3.54m	2.42m	2.21m	2.21m



а) б)

Рис. 3. Графіки навчання за алгоритмом PPO (а) та SAC (б) у другій серії дослідів.

У першій серії дослідів обидва алгоритми показали стабільні результати, хоча алгоритм PPO значно швидше навчався. У більш складних умовах другої серії дослідів результати значно відрізняються у кожному досліді. У третьому досліді за алгоритмом PPO за 10.32m ітерацій не було досягнуто результату в 99 відсотків ефективності і дослід був зупинений. Натомість при навчанні за алгоритмом SAC було досягнуто більш стабільних результатів, хоча час навчання був теж значно більшим. Таким чином, у подальших дослідженнях потрібно дослідити, які саме особливості алгоритмів PPO і SAC впливають на стабільність досягнення результату та ефективність навчання.

Література:

1. About ML-Agents package [Електронний ресурс] – Режим доступу: https://docs.unity3d.com/Packages/com.unity.ml-agents@2.0/manual/index.html. Перевірено 05.05.2021.
2. Proximal Policy Optimization [Електронний ресурс] – Режим доступу до ресурсу: https://spinningup.openai.com/en/latest/algorithms/ppo.html. Перевірено 05.05.2021.
3. Soft Actor-Critic [Електронний ресурс] – Режим доступу до ресурсу: https://spinningup.openai.com/en/latest/algorithms/sac.html. Перевірено 05.05.2021.

УДК 004.8

ФОРМУВАННЯ АУДІОВІЗУАЛЬНОГО ПОРТРЕТУ ЕМОЦІЙНИХ СТАНІВ

Кравченко Е.Л., Ярмілко А.В.

Черкаський національний університет ім. Богдана Хмельницького

У 2019 році ринок розпізнавання лиця штучним інтелектом оцінювався у 3,2 мільярда доларів [1]. Станом на сьогодні, люди можуть використовувати сканер свого обличчя для покупок в магазині, доступу до об`єктів, тощо. Але багато де така технологія є забороненою в цілях конфіденційності. Розпізнавання обличчя може бути корисним і шкідливим явищем водночас.

Розпізнавання обличчя, за визначенням, означає технологію, яка здатна розпізнати людину за зображенням її обличчя. Розпізнавання обличчя ґрунтується на складних математичних алгоритмах штучного інтелекту та машинного навчання, які фіксують, зберігають та аналізують риси обличчя, щоб поєднати їх із зображеннями людей у вже існуючій базі даних і інформацією про них у цій базі даних. У цей процес також можна включити багато інших додаткових підпроцесів, наприклад – розпізнавання емоції людини. Розпізнавання обличчя є частиною загального технічного терміну біометрії, який також включає розпізнавання відбитків пальців, відбитків долонь, сканування очей, розпізнавання голосу та підписів.

Разом з цим, постійно розвивається музична індустрія. Вченими доведено, що обрана музика та мелодія конкретного музичного напряму впливає на емоції, почуття та психіку людини. Крім того, слід враховувати окремі звуки, які також впливають на людину. Наприклад, шум моря допомагає людині налаштуватись на продуктивну роботу, звуки лісу допоможуть людині зняти стрес або відпочити [2]. Біометричні методи аналізу надають засоби для зв’язування певного музичного контенту з конкретними персоналіями за встановленими правилами, зокрема – за емоціями. Така задача є актуальною як для пересічних меломанів, так і для спеціалістів з психології, реабілітації, управління трудовими кондиціями виробничого персоналу.

Існує багато готових алгоритмів розпізнавання обличчя, написаних на мовах програмування Python, C++, Lisp, Prolog або Java, реалізація яких відрізняється трудоміскістю, бюджетними вимогами, орієнтацією на конкретні дослідницькі чи бізнес-цілі [3]. Аналіз використаних у них методів дозволяє вважати перспективними для сервісу з розпізнавання емоції лиця людини та формування набору їх музичного супроводу такі три моделі: згорткова нейронна мережа (Convolution Neural Network – CNN), Vision Transformer, залишкова нейронна мережа (Residual Neural Network – ResNet).

Згорткова нейронна мережа (CNN) – це клас штучних нейронних мереж, який став домінуючим у різних завданнях що стосуються комп'ютерного зору. CNN призначений для автоматичного та адаптивного вивчення

просторових ієрархій функцій шляхом зворотного розповсюдження за допомогою декількох будівельних блоків, таких як згорткові шари, агрегувальні шари та повноз'єднані шари [4].

Роботу Vision Transformer можна описати наступним чином: першим кроком моделі Vision Transformer є розподіл вхідного зображення на послідовність патчів зображення. Відповідно до методу, зображення ділиться на сегменти 16 x 16; Потім ці ділянки зображення проходять через тренувальний лінійний проекційний шар. Цей шар виконує роль шару вбудовування і виводить вектори фіксованого розміру; Потім вбудовані позиції додаються до послідовності патчів зображення [5]. Ідея того, як працює вбудовування позиції, продемонстрована на зображенні нижче (рис. 1):

Рис. 1. Робота Vision Transformer на прикладі фото міста.

Залишкова нейронна мережа (ResNet) – це така штучна нейронна мережа (ANN), яка базується на конструкціях, відомих з досліджень пірамідальних клітин кори головного мозку. Залишкові нейронні мережі використовують пропускання з'єднань, або ярлики, для переходу через деякі шари. Типові моделі ResNet реалізовані з дво- або тришаровими пропусками. Додаткова матриця ваги може бути використана для вивчення пропуску ваг – ці моделі відомі як HighwayNets. Моделі з декількома паралельними пропусками називаються DenseNets [6].

Кожна з оглянутих моделей має свої переваги та недоліки, що, в свою чергу, несе загрозу неточності розпізнавання емоцій та, відповідно, хибного аудіовізуального портрету особистості. Тому є потреба у проведенні ряду дослідів, в яких використовувалися б наведені вище моделі, з метою вибору найбільш ефективної з них щодо ідентифікації емоційних станів. Загальна ж структура пропонованої системи формування аудіовізуального портрету за емоційними станами зображена на рис. 2.

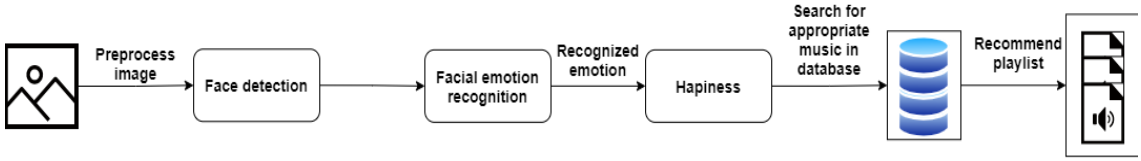


Рис. 2. Структура системи формування
аудіовізуального портрету за емоційними станами

Вхідне зображення обличчя формату JPG або PNG обробляється нейронною мережею, яка розпізнає лице людини. Далі система ідентифікує

емоцію лиця, яке вона розпізнала на попередньому кроці. Після визначення всіх необхідних даних, система формує плейлист за набором музичних творів у відповідній базі даних, доповнює його відповідними візуальними атрибутами та пропонує даний контент користувачеві.

Таким чином, проведений етап досліджень дозволив окреслити коло перспективних методів обробки зображень обличчя людини для аналізу її емоцій. Оцінка ефективності цих методів при вирішенні поставленої задачі потребує залучення експериментальних методів. Вирішення задачі оптимального вибору моделі визначення емоційного стану людини має забезпечити належний рівень адекватності аудіовізуального портрету, що створює перспективи використання такої технології в наукових та комерційних цілях.

Література:

1. Facial recognition: top 7 trends (tech, vendors, markets, use cases & latest news). Електронний ресурс – Режим доступу: https://www.thalesgroup.com/en/markets/digital-identity-and-security/government/biometrics/facial-recognition
2. Ynpress - Роль музыки в жизни людей. Електронний ресурс – Режим доступу: https://ynpress.com/archives/56659#:~:text=В%20различных%20случаях%20музыка%20влияет,поднимать%20настроение%2С%20когда%20становится%20тоскливо.
3. AI Programming: 5 Most Popular AI Programming Languages. Електронний ресурс – Режим доступу: https://ncube.com/blog/ai-programming-languages
4. Convolutional neural networks: an overview and application in radiology. Електронний ресурс – Режим доступу: https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9
5. Vision Transformer (ViT) - Using Transformers for Image Recognition. Електронний ресурс – Режим доступу: https://www.section.io/engineering-education/vision-transformer-using-transformers-for-image-recognition/
6. Wikipedia - Residual_neural_network. Електронний ресурс – Режим доступу: https://en.wikipedia.org/wiki/Residual_neural_network

УДК 004.67

ІНТЕЛЕКТУАЛЬНИЙ МОДУЛЬ ОБРОБКИ СОЦІОЛОГІЧНИХ ДОСЛІДЖЕНЬ

Цвіркун Р.С.

Черкаський національний університет ім. Богдана Хмельницького

У сучасному світі, ефективне управління соціумом, передбачає конкретні, цілеспрямовані, диференційовані аспекти інформації певної групи

людей про процеси, що відбуваються з ними і навколо них. Дану інформацію сформовано в різних джерелах таких як: документи, наради, матеріали зборів, у пам'яті окремих особистостей, але вона не завжди є об'єктивною і не в повному обсязі відбиває повної сукупності соціальних фактів. Тому для більш детального вивчення окремого об'єкта потрібно проводити соціологічне дослідження [1].

Не менш важливим аспектом є обробка результатів отриманих під час проведення досліджень. Від вибраного способу обробки залежать вартість, строки і сам успіх дослідження. Зазвичай кількість інформації є дуже обширною і обробка її вручну не є ефективною тому виникає необхідність в розробці програмного забезпечення, яке б виконувало цю роботу.

Розглянемо найбільш практичний і ефективний метод збору даних на сьогодні. Ним являється опитування населення. Як правило, опитування має формат звичайного тесту де можна обирати правильну на думку респондента відповідь, або ж тест може містити питання на яку потрібно дати розгорнуту відповідь.

Від поставлених респонденту питань напряму залежить інформація, яку ми отримуємо, тому тематика тестів має конкретно відповідати тим об'єктам, які ми досліджуємо [3].

Отже програмний продукт, який буде задовольняти потреби державних установ: сільських і міських рад, різних організацій, тощо, повинен збирати інформацію від населення шляхом проходження опитування, мати потрібний функціонал для формування питань, щоб змінювати сфери дослідження, а також швидко і оперативно обробляти всі зібрані дані на основі заданих параметрів та формувати звіт, в якому буде відображена загальна статистика, яка формується від набраної кількості балів кожним респондентом.

Реалізувати даний програмний модуль було вирішено у вигляді додатка, який пропонує респонденту тест, в залежності від його віку та сфери діяльності, записує всі дані в таблицю формату Excel, та формує звіт, як по всіх даних так і по даних конкретного респондента. Додаток було написано на мові C# з використанням Windows Forms [2].

У даній роботі головною темою дослідження було те, наскільки люди розуміють, що таке системний аналіз.

Розглянемо реалізацію користувацького інтерфейсу. Користувачеві потрібно буде лише ввести свій вік і вибрати сферу діяльності із запропонованих, а саме: абітурієнт, студент, роботодавець.

Після чого натиснути кнопку «Розпочати тест» і додаток сам підбере питання, які відповідають заданій користувачем категорії (рис 1).

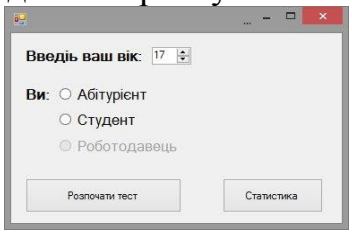


Рис. 1. Користувацький інтерфейс програмного продукту.

Після проходження тесту (рис. 2), додаток запропонує ввести своє прізвище та ініціали, або ж можна пройти тест анонімно для цього є відповідна кнопка. Всі отримані дані записуються в таблицю (рис. 3).

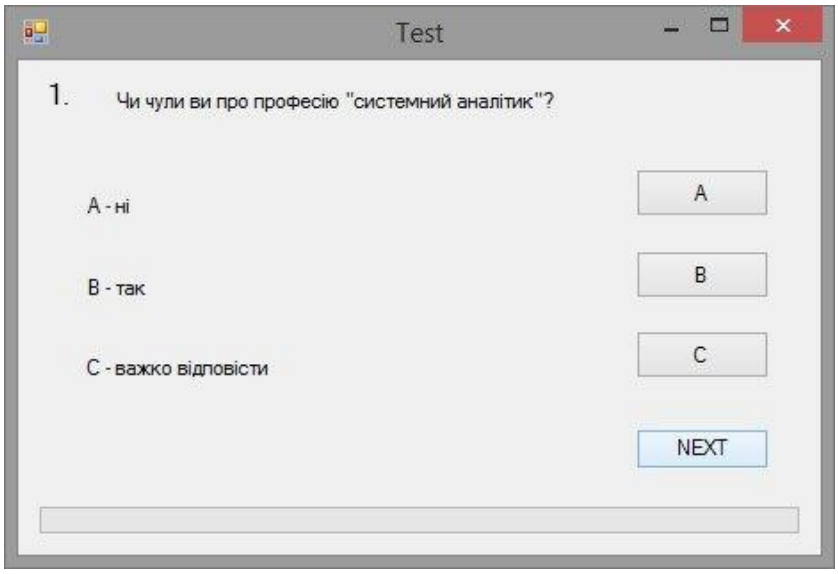


Рис. 2. Вікно «Тест».

	A	B	C	D	E	F
1	Порядковий №	Ім'я	Відповідь 1	Відповідь 2	Відповідь 3	Відповідь 4
2	1	Гладкий О.С.	Б	Б	А	В
3	2	Семенченко П.Й.	А	В	В	Б
4	3	Титаренко І.А.	Б	Б	Б	Б

Рис. 3. Таблиця результатів.

Також реалізована можливість обробки всіх отриманих даних, для цього на початковому вікні є відповідна кнопка статистика де надано такі можливості:

1. Зміна кількості балів за варіанти відповідей.
2. Перегляд загальної статистики: за сферою діяльності, за віком.
3. Перегляд статистики конкретного користувача (якщо він проходив тест не анонімно).
4. Відображення результату конкретного користувача залежно від його сфери діяльності:

– абітурієнт: рівень доуніверситетської підготовки, шанс вступу на спеціальність «Системний аналіз»;
– студент: рівень засвоєння матеріалу, шанс подальшого працевлаштування;
– роботодавець: на скільки людина, яка навчається за спеціальністю «системний аналіз» буде корисною для нього, як співробітник та загальний рівень розуміння системного аналізу.

Література:

1. Курчаба Т. Соціологія : навч.-метод. посіб. / Тетяна Курчаба. – Львів: ПП Сорока Т. Б., 2015. – 183 с.
2. Крістіан Нейгел C# 5.0 и платформа .NET 4.5 для профессионалов = Professional C# 5.0 and .NET 4.5. — М.: «Диалектика», 2013. — 1440 с.
3. Широков Р. Ю., Єпік М. О. Інтелектуальна система аналізу і прогнозування результатів соціологічних досліджень. Вісник студентського наукового товариства ДонНУ імені Василя Стуса. 2020 т.2 №12 с.359-363. Режим доступу: https://jvestniksss.donnu.edu.ua/article/view/9302/9246

Секція 5

Моніторингові технології, системи та комплекси в сучасному інформаційному суспільстві

УДК 681.518:004.422.63

МОДЕЛЬ ДАНИХ АНАЛІЗУ КОНСОЛІДОВАНОЇ ІНФОРМАЦІЇ З РІЗНИХ ДЖЕРЕЛ У СИСТЕМІ РЕАЛЬНОГО СВІТУ

Тараненко Р.А.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Двадцяте століття стало етапом переходу науки до умовного іншого виміру [1], внаслідок якого вимоги до аналітичних та формалізованих моделей суттєво ускладнилися. Це призвело до необхідності створення методів здатних якісно працювати з великими і дуже великими обсягами інформації в обсягах представлень реального світу та режимі реального часу – N= «ВСЕ» [2]. І особливого значення це набуло під час активного залучення всіх сфер життєдіяльності суспільства у нові трансформаційні процеси закономірного розвитку, виражені суттєвою невизначеністю та необхідністю всеохоплюючої «цифровізації». Проблема, що сам термін «інформація» досі трактується інтуїтивно, неоднозначно та її значення змінюється з часом як змістовно так і за цінністю. Інформаційні технології та системи у своєму розвитку вступили до фази активного формування де головну роль відіграють інформаційні представлення рис. 1.

Рис. 1. Ієрархічна піраміда сучасних інформаційних технологій.

Великі дані надають нових можливостей, суттєво змінюючи вимоги до представлення та обробки інформації, вимагаючи зіставляти різні категорії та види інформації у великих обсягах. Фактично невизначеність призводить до суттєвого падіння ефективності отримання аналітичних рішень у роботі з великими даними, перетворюючі більшу частину їх обробки у «ручну роботу». Ручна обробка може сягати рівня 80% і більше. Склалася ситуація, коли нові можливості досить складно реалізувати. Потрібні нові підходи, що реалізують можливості розкриття потенціалу великих даних повною мірою та підвищать ефективність у роботі з ними. Один з шляхів є розробка відповідної моделі одиниці інформації та подання її у моделі даних, яку для збереження можливості співставлення різних видів та категорій інформації доцільно розглядати у системі реального світу (СРС).

Глибокий взаємозв'язок, взаємодія і взаємопроникнення всіх речей говорять нам про те, що всі взаємини, в системі реального світу, мають дві

складові – явну і неявну або іншими словами формалізовану і не формалізовану.

Явна складова, являє собою фрагментарну систему знань про реальний світ, виділену суспільством в процесі його життєдіяльності у вигляді сукупності різних моделей.

Неявна складова, внаслідок своєї складнішої природи, виступає керуючою і утворюючою по відношенню до явної.

В процесі своєї життєдіяльності суспільство, за допомогою сукупності різних, в першу чергу формалізованих інструментів, намагається опанувати неявну складову реального світу. Фактично це зводиться до усунення несправедливості спотворення існуючих моделей по відношенню до системи реального світу.

Взаємозв'язок, взаємодія і взаємопроникнення всіх речей вказують нам на наступні важливі зауваження до розуміння законів навколишнього світу: - світ такий, що може бути пізнаним; - світ пронизаний закономірними явищами (він суворо впорядкований); - немає нічого випадкового.

Дане розуміння є наслідком методологічних передумов в результаті розгляду всіх явищ в контексті їх цілісного системного уявлення. При цьому основним критерієм розгляду явищ стає розширення понять його якостей до кордонів, що описують його показники, як невід'ємну складову його станів [3]. Відповідно, за висловом Гегеля - "Якість є взагалі тотожна з буттям безпосередня визначеність ...", "Щось є завдяки своїй якості те, що воно є, і втрачаючи свою якість, воно перестає бути тим, що воно є ..."

Внаслідок спрощення кожна модель має обмеження, обумовлені ступенем адекватності описуваного явища. За межами цих обмежень настає конфліктно-некерована ситуація. Розглядаючи знання про систему реального світу як такі що, представлені, формалізовано, у вигляді моделей на мові інформації, ми спостерігаємо їх ускладнення, зростання і розширення області їх застосування.

Закономірним результатом розширення області застосування моделей є, наявність апріорної інформації, яка не піддається поясненню механізмами вже існуючих моделей. Це вимагає адекватної зміни фундаментальної парадигми. Ця інформація отримана при співставленні уявлень існуючих моделей, з новими межами або областями реального світу внаслідок охоплення неявною складовою явну, і безпосередньо до вимог застосування великих даних. Або іншими словами відбувається процес формування нових моделей шляхом формалізації неявної складової системи реального світу.

Головні вимоги до моделі даних представлення інформації.

5. Модель даних повинна забезпечити гнучкість та адекватність аналітичних моделей у роботі з різними видами та категоріями інформації великих даних.
6. Модель даних повинна сприяти подоланню проблем застосування спрощених, обмежених підходів у великих даних.
7. Модель даних повинна сприяти – покращити вилученню знань з інформації.

Модель даних розглядається, як складова 4-и вимірного простору великих даних, в рамках якого проводиться пошук рішень:

- 1-ий вимір – Data (дані);
- 2-ий вимір – Volume (обсяг);
- 3-ий вимір – Velocity (швидкість);
- 4-ий вимір – Knowledge (знання) – впорядковані представлення, отримані на основі цілісного підходу.

Кожен фактор явища чи процесу, як і його складові вписують у простір обмежений «вузловими крапками», або певний період що можливо також представити, як «життєвий цикл».

Згідно сучасних наукових уявлень модель даних розглядається як складна кібернетична система з відповідними складовими табл. 1.

Таблиця 1.

Складові фактори моделі даних

Системні показники (критерії)	Управляючі показники (критерії)	**Вихідні дані**
Елементарні показники (критерії)	**Цілісність** - явище, чи процес, що можна охарактеризувати як певна цілісність	Управляючі фактори
Вхідні дані	Елементи	Система

Фактори моделі даних подані у табл. 1 дозволяють отримати універсальну модель даних, що при відповідному розвитку дозволяє ефективно зіставляти різні види та категорії інформації та формувати простори отримання ефективних рішень у складних явищах та процесах і безпосередньо в задачах «Великих Даних» (Big Data).

Література:

1. Клир Дж. Системология. Автоматизация решения системных задач: Пер. с англ. - М.: Радио и связь, 1990. - 544 с.
2. Майер-Шенбергер В. Большие данные. Революция, которая изменит то, как мы живем, работаем и мыслим / Виктор Майер-Шенбергер, Кеннет Кукьер; пер. с англ. Инны Гайлюк. - М.: Манн, Иванов и Фербер, 2014. - 240 с.
3. Тараненко Р.А. Качество информации: обзор представлений в контексте целостной парадигмы // УСиМ. – 2005. - № 5. - С. 3-12.

UDC 004.582

INTELLIGENT SYSTEM FOR CHOOSING CLIMATIC EQUIPMENT

Viznuk A.V.

Taras Shevchenko National University of Kyiv

Intelligent systems are often used as a computer replacement for a human expert. It saves a lot of users' time in decision-making especially if they lack knowledge in some specific area [4].

The purpose of the system described below is to help people choose the climatic equipment. For now, the focus is mostly on recommending heaters for the user's room.

Assume a system is needed to recommend products by selected features. In addition, the system should always show the best products available. What if almost every feature is selected by the user and no product fully satisfies his or her criteria? If products to be filtered by matching features, the user wouldn't see any wishful results. One of the possible ways to deal with the problem is to sort products by a number of matching features in descending order. But what if some features are more valuable to a customer? The more flexible solution offered is to assign weighting coefficients to each selected feature. A more valuable feature has a greater weight. Use-case diagram of such system is shown in Fig. 1.

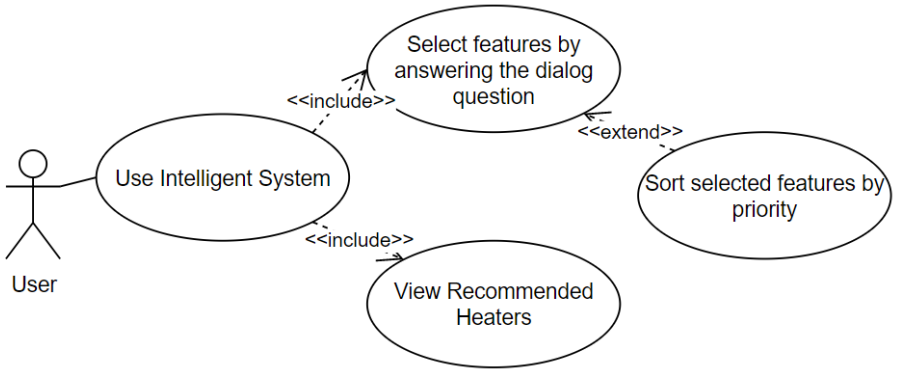


Fig. 1. Use cases for the intelligent system for choosing climatic equipment.

Feature categories taken into account are [1]:

- heating capacity (depends on the user's room square);
- control type (mechanical or electronic);
- mobility (portability);
- heating speed;
- mounting type;
- protection;
- other additional features (like remote control or electronic display).

Whenever the order of the selected features is changed by a user, the corresponding coefficients for each feature is calculated by the system utilizing the geometric progression rule [3]. The sum of all coefficients is set to be one permanently. The coefficient for the first feature in the list can be calculated as:

$$coeff_1 = \frac{ratio - 1}{ratio^n - 1} \quad (1)$$

where n – the number of selected features by the user. The coefficients for the remaining features are calculated via the following expression:

$$coeff_k = coeff_1 ratio^{k-1} \quad (2)$$

where k is a feature index in the array of selected features.

The next step is to evaluate the total score for each selected feature. The resulting score is represented as the sum of heater features coefficients which are in the list of selected features by the user. The general rule of total score determination looks as follows:

$$H : SF \to C$$
$$score_k = \sum_{f \in SF \cap HF_k} H(f) \quad (3)$$

where k – heater index, SF – selected features, ordered by priority, HFk – features of the heater. H – bijection from set SF to set C, where C – set of coefficients for each selected feature. H(f) – the corresponding coefficient for feature f.

When all scores are calculated for each heater, all products are sorted in descending order by that score.

The next critical question to take into account is the appropriate choice of the heater power. It is widely known that the power choice for the heater depends on the room area. If the room is of standard height (2.5 – 2.7 meters) then the power of the heater should be no less than [2]:

$$power = \frac{square}{10} kW \quad (4)$$

where square is the room area in square meters, 10 – Kilowatts per square meter.

When recommended power is calculated, heaters are selected with power no less than the calculated one. Afterwards the list of recommended heaters is displayed for the user.

The solution given provides more flexible functionality than a simple filter. An appropriate implementation of priority functionality guarantees that the top-recommended products from the stock have the most valuable features for the user.

In near future, users will be able to leave their score on the products purchased and the total calculated score will be taken into account while using the intelligent system developed.

References:

1. How to choose a heater: help in defining criteria (In Russian). URL: https://www.ixbt.com/home/heaters-buyers-guide-2018.html (date of request: 08.04.2021).

2. Calculation of heating capacity by the room area – a detailed analysis of the methods (In Russian). URL: https://otopleniedomov.com/otoplenie/raschet-otopleniya-po-ploshhadi-pomeshheniya-podrobnyjj-razbor-metodov.html (date of request: 08.04.2021).
3. Geometric progression. URL: https://en.wikipedia.org/wiki/Geometric_series (date of request: 09.04.2021).
4. Machine Learning Paradigms / G. A. Tsihrintzis, M. Virvou, E. Sakkopoulos, and L. C. Jain. Cham: Springer International Publishing, 2019. 568 pp.

УДК 330.341.1:65.012

БАГАТОРІВНЕВА МОДЕЛЬ ЦИФРОВОЇ ТРАНСФОРМАЦІЇ ІННОВАЦІЙНОЇ СИСТЕМИ

Осауленко І.А., Жирякова І.А.

Черкаський національний університет ім. Богдана Хмельницького

Процеси цифрової трансформації призводять до докорінних змін у переважній більшості сфер людської життєдіяльності. Тому не дивно, що дослідження питань цифрового розвитку останнім часом привертають все більшу увагу урядових структур і бізнесу, наукової спільноти і фахівців-практиків. Результатом цієї роботи стає як розроблення амбітних стратегій [1], так і створення відповідного науково-методологічного базису.

Зокрема, розглядаються поняття “оцифровування” (переведення документів, зображень, сигналів у цифрову форму з метою їхнього подальшого оброблення й використання за допомогою цифрових пристроїв) та “цифровізація” (широке впровадження цифрових пристроїв та комунікаційних мереж з подальшою автоматизацією бізнес-процесів). Натомість сутність цифрової трансформації визначається як зміна форми діяльності, оновлення бізнес-моделей, розширення кола та запровадження нових форм обслуговування клієнтів, у тому числі у вигляді цифрових платформ [2].

Цифрова трансформація охоплює різноманітні функції управління, бізнес-процеси, бізнес-моделі, мережі партнерів та стейкхолдерів, бізнес-активи, організаційну культуру. З огляду на це, для здійснення успішних перетворень в інноваційній системі необхідно чітко структурувати цілі і завдання, а також визначити ключові показники ефективності. При цьому ІТ-стратегія повинна розглядатись як невід’ємна складова загальної стратегії розвитку системи.

Одним із найефективніших сучасних підходів до стратегічного управління є побудова стратегічних карт і збалансованої системи показників. Останнім часом цей інструментарій почав застосовуватись для аналізу процесів, пов’язаних з цифровою трансформацією [3].

Класична модель збалансованої системи показників передбачає виділення в бізнес-системі процесів чотирьох рівнів: фінансових,

клієнтських, внутрішніх, самовдосконалення. Процеси кожного рівня характеризуються власними параметрами, у загальному випадку як кількісними, так і якісними, хоча пріоритет все ж надається кількісним індикаторам. У розрізі характеристик цифрової трансформації показником для процесів самовдосконалення може бути, наприклад, кількість працівників, які пройшли курси підвищення кваліфікації з використання цифрових технологій. Досконалість внутрішніх виробничих процесів пов'язується, зокрема, з кількістю інтелектуальних пристроїв та приладів. Для клієнтських процесів серед критеріїв варто відзначити час оформлення і виконання замовлення, кількість доступних нових сервісів. Нарешті, фінансова складова може оцінюватись на основі обсягів використання цифрових активів. Розвиток системи передбачає досягнення відповідними показниками встановлених цільових значень (рис. 1).

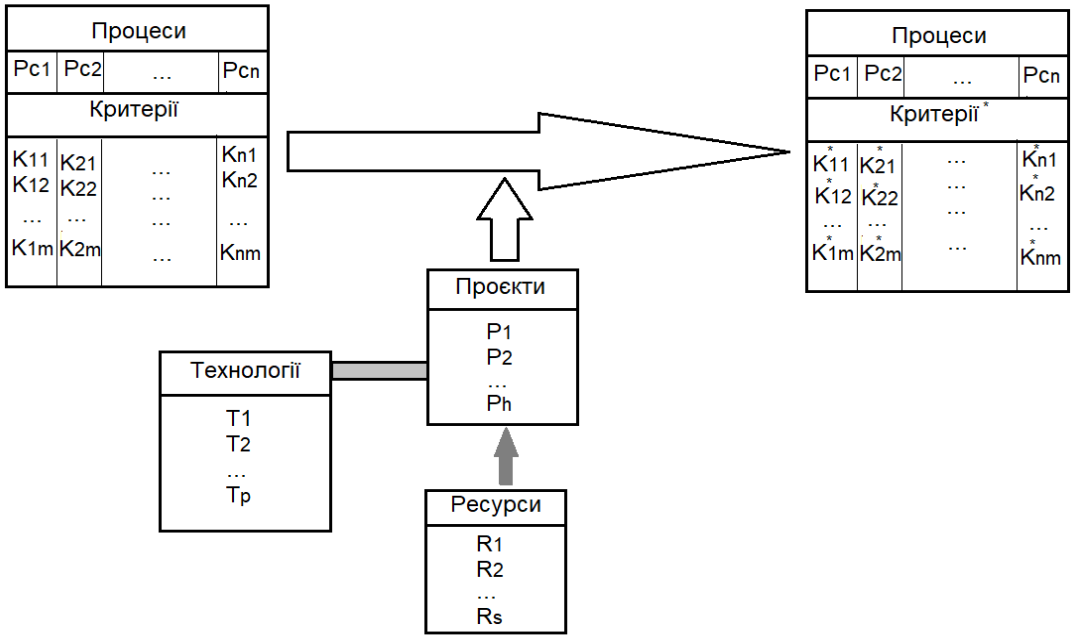


Рис. 1. Спрощена модель трансформації бізнес-системи.

Звичайно, перелік процесів і відповідний набір критеріїв будуть суттєво залежними від характеру діяльності інноваційної системи, оскільки і самі інновації можуть принципово різнитись за змістом і сферою використання (продуктові, технологічні, управлінські, організаційні, маркетингові, екологічні, освітні, соціальні тощо). Разом з тим, найбільший інтерес для дослідження становлять такі системи, в яких органічно поєднуються інновації різних типів на основі використання сучасних цифрових платформ.

У цьому контексті привертає особливу увагу концепція інноваційної екосистеми, яка за своєю сутністю є мережевою формою взаємовигідного поєднання зусиль і ресурсів зацікавлених сторін для досягнення спільних інноваційних результатів. До особливостей інноваційних екосистем, поряд із високою самоорганізацією та децентралізацією прийняття рішень, відносять взаємний розвиток під час взаємодії партнерів (коеволюцію). Цілком очікувано, що зазначена властивість стосуватиметься також і питань цифрового розвитку.

Крім того, при побудові моделі потрібно враховувати, що в мережевій структурі бізнес-процеси можуть бути наскрізними, тобто виходити за межі окремих суб'єктів взаємодії. Водночас деякі процеси при структуризації будуть по-різному класифікуватись для окремих стейкхолдерів. Наприклад, процеси надання послуг з навчання персоналу для освітнього закладу або технічного обслуговування обладнання для сервісної організації розглядатимуться як клієнтські. При цьому для виробничої компанії процес першого типу буде віднесено до процесів самовдосконалення, процес другого типу пов'язуватиметься із забезпеченням внутрішньої діяльності, у нашому випадку підтриманням належного технічного стану основних виробничих засобів. Аналогічна ситуація з використанням у виробничому процесі сировини, матеріалів, напівфабрикатів, отриманих від сторонніх організацій, наданням логістичних та інших послуг.

Таким чином, в моделі цифрової трансформації інноваційної системи може бути виокремлено три рівні: рівень інноваційної системи (спільноти), рівень окремого підприємства чи установи та рівень певної категорії процесів підприємства. На рівні інноваційної системи доцільно першочергово здійснювати трансформацію наскрізних процесів, а на рівні окремих суб'єктів основна увага має приділятись клієнтським процесам, оскільки саме це створює передумови для впровадження нових бізнес-моделей на основі цифрових платформ. Цифровий розвиток неможливий без реалізації в системі проєктів впровадження нових інформаційних технологій та модернізації ІТ-інфраструктури. Разом з цим, для управління такими проєктами варто застосовувати гнучкі методології [4], оскільки в інноваційній системі завжди досить висока ймовірність коригування попередніх планів, а безпосередня взаємодія між людьми зазвичай важливіша, ніж використовувані інструменти.

Література:

1. Europe's Digital Decade: Commission sets the course towards a digitally empowered Europe by 2030 [Electronic source]. European Commission official site. Access mode: https://cutt.ly/rb6xOv4 (Last accessed: 15.05.21).
2. Дергачова Є. М., Колешня Я. О. Цифрова трансформація бізнесу: сутність, ознаки, вимоги та технології [Електронний ресурс]. Економічний вісник НТУУ "КПІ". 2020. №17. С. 280–290. Режим доступу: https://cutt.ly/Hb6xGgk (дата звернення: 15.05.21).
3. Yamamoto S. A Strategic Map for Digital Transformation [Electronic source]. Procedia Computer Science. 2020. Vol. 176. Pp. 1374–1381. Access mode: https://cutt.ly/Tb6xJBF (Last accessed: 15.05.21).
4. Gurusamy K., Srinivasaraghavan N., Adikari S. An Integrated Framework for Design Thinking and Agile Methods for Digital Transformation [Electronic source]. 5-th International Conference DUXU 2016: Design, User Experience, and Usability: Design Thinking and Methods. Proceedings. Toronto, Canada. 2016. Part I. P. 34–42. Access mode: https://cutt.ly/vb6xCux (Last accessed: 15.05.21).

УДК: 519.876.5

ALGORITHM FOR INCREASING THE LEVEL OF CONSISTENCY FOR THE ALTERNATIVES PAIRWISE COMPARISONS MATRIX IN THE ANALYSIS OF COMPLEX INHOMOGENEOUS SYSTEMS

Skiter I.S., Saveliev M.V.

National Academy of Science of Ukraine,
Institute for Safety Problems of Nuclear Power Plants

The Chernobyl exclusion zone is a complex system where many qualitatively heterogeneous subsystems can be distinguished. Each subsystem is characterized by its own unique set of factors and indicators. The work [1] proposes to use integral quality factors for a numerical assessment and comparison of subsystems. The Saaty method is used to compare qualitatively heterogeneous factors. This work proposes an algorithm for assessing the degree of inconsistency of expert opinions and their correction for a more consistent pairwise comparison of factors and minimization of subjectivity (expert assessments).

The necessity and correction of expert assessments in the formation of the matrix of pairwise comparisons is carried out on the proposed method for assessing the degree of collinearity of the comparison vectors of "alternatives" a_{ij}. The collinearity of the comparison vectors leads to a sufficient level of the concordance index (IC) and the correctness of determining the weights of the factors ω_i in the system / subsystem for which the estimation is carried out. Accordingly, the level of inconsistency of comparisons can be estimated by the value of the cosine of the angle between vectors.

The procedure for correcting pairwise comparisons in order to achieve an acceptable level of consistency can be carried out by three methods: automatically, iteratively, by the method of repeated expert evaluation [2].

It is advisable to use automatic correction in conditions when the value of the concordance index of the matrix of paired comparisons $\mathrm{IC} > 50\%$, which indicates an unsatisfactory level of consistency of expert assessments

It is advisable to apply iterative correction procedures to increase the level of consistency, provided that $20\% \leq \mathrm{IC} < 50\%$ - the level of consistency of pairwise comparisons is good, the weight coefficients of factors in the system can be trusted.

We propose to perform the task of assessing the consistency of paired comparisons, as well as assessing the level of consistency based on the analysis of the statistical characteristics of the matrix of paired comparisons. In this case, the diagonal elements are excluded from the pairwise comparison matrix.

Then, based on the $n(n-1)$ estimates of paired comparisons, the mean and standard deviation σ are calculated. The value of the standard deviation in this case shows the degree of dispersion of the agreement of expert comparisons of factors.

In [3], based on the analysis of the SLE from the consistency index for matrices of different dimensions, it was found that the probability of rational

adjustment of the matrix of pairwise comparisons is the greater, the greater the scatter of the consistency of judgments, since in these cases one can single out the most related expert assessments.

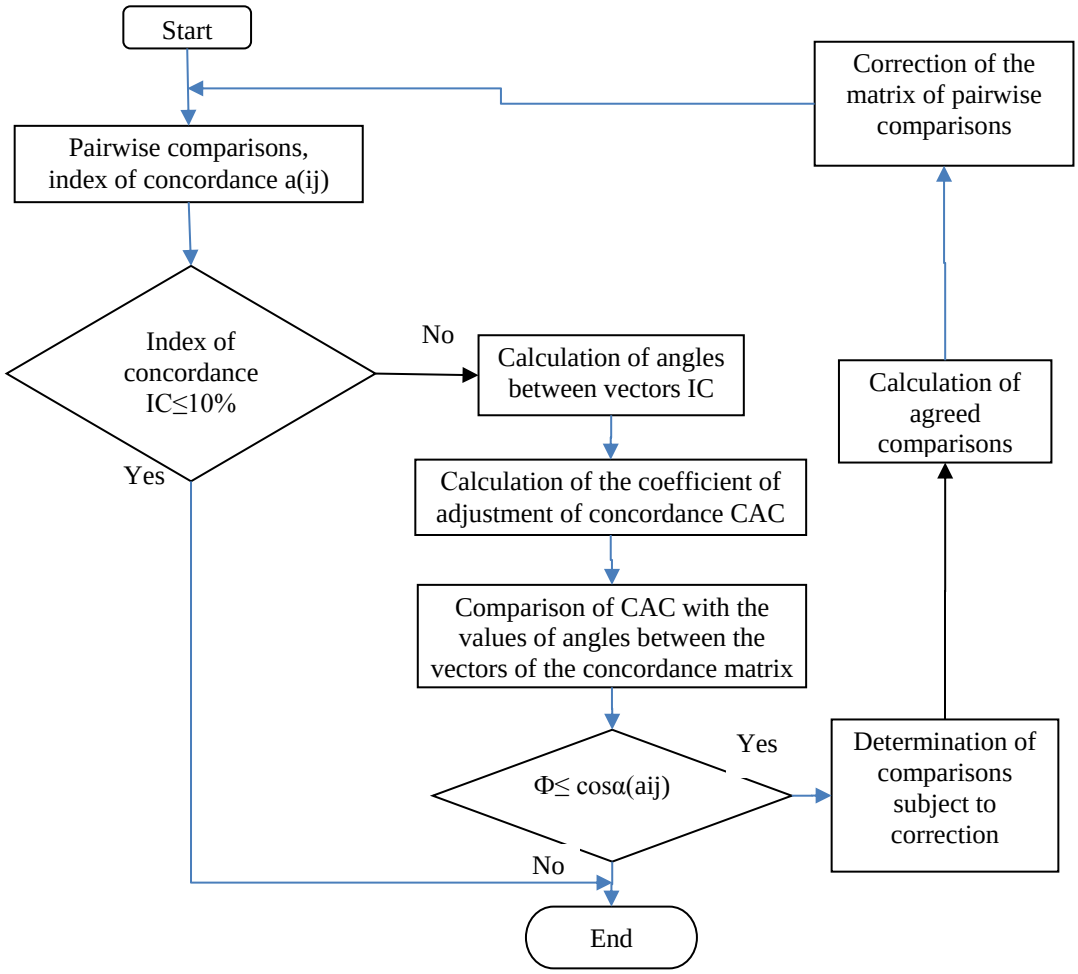


Fig. 1. Algorithm for constructing a consistent matrix of pairwise comparisons in a modified Analytic Hierarchy Process.

In the matrix of pairwise comparisons, the standard deviation value lies in the range $0 < \sigma \leq 0{,}5$. Then, according to [4], it is proposed to determine the lower bound of the standard deviation as $\sigma^* \leq 20\%\sigma_{max}$. Or numerically $\sigma^* = 20\% \times 0{,}5 = 0{,}1$.

Thus, matrices of pairwise comparisons, for which the SPC of the assessments of the consistency of individual judgments exceed 0.1, can be corrected by the presented methods. The algorithm for choosing the method of matrix correction is shown in Fig. 2.

The proposed approach makes it possible to improve the correctness of determining the weighting coefficients of factors that characterize the system / subsystem. The use of the algorithm will make it possible to carry out numerical assessments of the effectiveness of various subsystems of the Chernobyl exclusion zone and form effective management decisions on their basis.

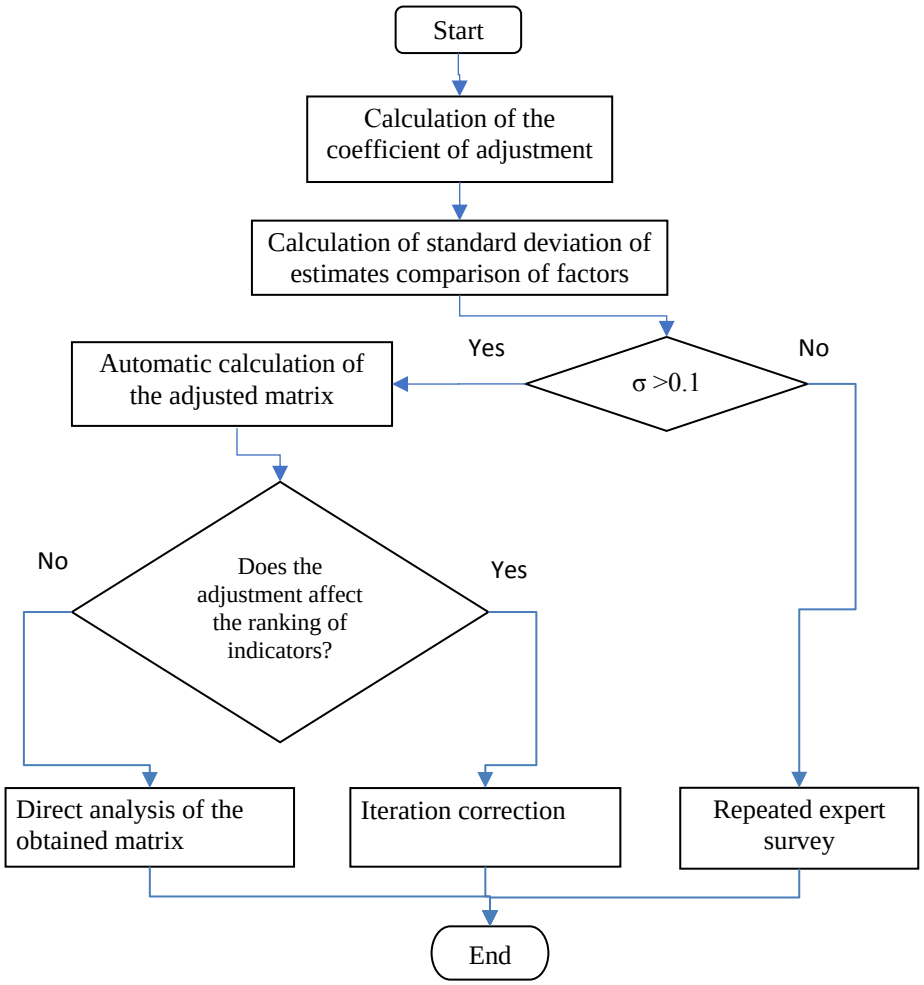


Fig. 2. Algorithm for choosing a correction method in the process of increasing the degree of consistency of paired comparisons.

References:

1. Паскевич С., Скітер І., Дереговський В. Моделювання інтегрального коефіцієнту інвестиційного потенціалу регіону. Проблеми зняття з експлуатації об'єктів ядерної енергетики та відновлення навколишнього середовища (INUDECO 21) : збірник матеріалів VI Міжнародної конференції (27–29 квітня 2021, м. Славутич). – Чернігів : НУ «Чернігівська політехніка», 2021. – с. 219-226.

2. Margasov D., Sakhno E., Skiter I. Development of a model and modification of the hierarchy analysis method for energy efficiency level estimation/Eastern-European Journal of Enterprise Technologies 5 (2), 26-32.

3. J. Ramík. Ranking Alternatives by Pairwise Comparisons Matrix and Priority Vector. Scientific Annals of Economics and Business 64 (SI), 2017, p. 85-95. https://doi.org/10.1515/saeb-2017-0040

4. G. R. Vasconcelos, C. M. de Miranda Mota, "Exploring Multicriteria Elicitation Model Based on Pairwise Comparisons: Building an Interactive Preference Adjustment Algorithm", Mathematical Problems in Engineering,vol. 2019, Article ID 2125740, 14 pages, 2019. https://doi.org/10.1155/2019/2125740

УДК 004.5

«THE.ГРОМАДЯНИН»: КОНКУРСНИЙ ПРОЄКТ НА ХАКАТОН «МІСТО НОВИХ ІДЕЙ»

Губенко О.В., Літвінов Р.Р., Олексюк Б.Ю., Поліщук Д.С., Сухіна А.С., Ярмілко А.В.

Черкаський національний університет ім. Богдана Хмельницького

Команда Черкаського національного університету у складі студентів 2 курсу факультету обчислювальної техніки, інтелектуальних та управляючих систем взяла участь у хакатоні «Місто нових ідей», який проводився на полях VI Міжнародної конференції INUDECO 2021 (27–29 квітня 2021, м. Славутич). Метою хакатону було виконання протягом однієї доби проектної пропозиції за заданою актуальною для міста-організатора тематикою. Командою було розроблено та представлено проєкт «The.ГРОМАДЯНИН», орієнтований на вдосконалення контролю за міською інфраструктурою, а саме – якістю дорожнього покриття.

При формуванні проектної пропозиції авторський колектив виходив з того, що час потенційного користувача (водія або пішохода) дуже цінний. Водій не повинен відволікатися від дороги і виходити з машини. Пішохід не може відкласти важливу зустріч. Обидва типи користувачів рівноцінні для системи. У зв’язку з цим сценарій користувача (user flow) повинен бути побудованим найкоротшим шляхом та враховувати такі позиції:

1. Для досягнення мети користувачу потрібно зайти в бот, прочитати перше повідомлення-інструкцію та зробити наступні дії: зробити фото, надіслати геомітку, описати проблему.

2. Пріоритетними для користувача визначені критерій оцінки швидкості дій і можливість надіслати дані пізніше.

3. Опис проблеми є найскладнішим і найдовшим пунктом сценарію, але його можна зробити пізніше, тому цей крок був поставлений на останнє місце.

4. Фотофіксація займає не так багато часу і є найпріоритетнішим пунктом, адже фото потрібно зробити «тут і зараз». Цей пункт поставлено на перше місце.

5. Відправка геомітки необхідна, але її, як і опис проблеми, можна надіслати пізніше – тож її зазначення у сценарії поставлено на друге місце.

6. Обов’язковою є потреба у нагадуванні користувачеві про те, що йому слід очікувати на відповідь від влади, і що зазначена ним проблема перебуває в стадії обробки. Це є зворотнім зв’язком у системі.

7. Також мають бути забезпечені повідомлення користувачеві у випадку хибних дій (наприклад, при вказуванні на об’єкт, який не є дорогою).

Аналіз показав, що водій може зробити фото через скло, не виходячи з машини, адже, за дослідженням, кожен третій водій за кермом використовує свої комунікаційні пристрої [1]. Не було сенсу розробляти проект у

розрахунку на пасажирів громадського транспорту або велосипедів: за результатом опитування місцевих жителів, респонденти не користуються ні тим, ні іншим. Типові способи пересування – таксі, власний автомобіль або пішки, оскільки місто невелике, компактне, а прокат велосипедів наразі відсутній.

Структура запропонованого програмного продукту представлена на рис. 1.

Рис. 1. Загальний алгоритм роботи у системі «The.ГРОМАДЯНИН».

При формуванні проекту враховувалося, що цільова база його впровадження, місто Славутич – дуже молоде місто. Тож важливим було гейміфікувати процес, аби цей проект був цікавим і для молодшої аудиторії, а не лише водіям. Досягнення особистого характеру дуже сильні: вони спираються на внутрішню мотивацію користувача, від чого він прагне продовжувати працювати і бути краще. Психологи стверджують, що мотивацію може формувати одне з двох джерел:

- внутрішнє – нами рухає почуття цікавості, гордості, любові і радості або бажання чогось досягти: «ви робите це, тому що хочете»;
- зовнішнє – імпульс, отриманий від когось або чогось; він може мати різну форму, наприклад, похвалу, диплом або гроші: «ви робите це, тому що отримуєте вигоду».

Внутрішня мотивація триває набагато довше, ніж зовнішня, і після перемоги може викликати почуття ейфорії.

Персоналізація цілей і надання користувачам можливості періодично їх переглядати розглядалася як засіб підштовхнути їх до більшої активності. Тому передбачені винагороди – те, що система можемо запропонувати в обмін на час і зусилля, витрачені при взаємодії з нею. Серед нагород були запропоновані ачівки, тобто звання за кожні 10 покращень. Проєкт

передбачав створення рейтингової таблиці для підтримання духу конкуренції серед громадян-користувачів. Відображення показників, які відстежуються, є ключовим моментом у створенні цього компоненту сценарію взаємодії, і це повинно бути пов'язано з цілями користувачів при налаштуванні «гри» – кількості покращень. Таблиці лідерів дають людям можливість змагатися один з одним. Стати лідером, а потім утримати лідерство – це неймовірно потужний мотиватор для збереження залученості. Отже, таблиці лідерів – це доступна метрика, яка допоможе «гравцям» зрозуміти, хто з них працює найкраще і / або має найбільший вплив на ситуацію в місті та прогрес. Однак слід проявляти обережність, щоб таблиці лідерів не викликали розбіжностей. Постановка особистих цілей в рамках ландшафту, а також забезпечення того, щоб увага приділялася всім «гравцям», мають допомогти пом'якшити будь-які негативні ефекти.

Час – фактор, який теж потрібно враховувати. Тому не варто відображати таблицю лідерів довше, ніж необхідно. Постійне перебування в нижній частині списку може демотивувати багатьох людей. Мета має виглядати досяжною, щоб мотивувати користувача повернутися. Якщо цілі здаються недосяжними, це може демотивувати і знизити залученість. Найуспішніші техніки гейміфікації забезпечують перетворення негативних емоцій досвіду в позитивні. Тому при проектуванні даного програмного продукту враховувалися такі аспекти:

1. Контроль: люди хочуть контролювати ситуацію і ненавидять, коли їх змушують робити те, чого вони не хочуть.

2. Релевантність: чим більше користувачі впевнені, що продукт був розроблений спеціально для них, тим він релевантніший.

3. Професіоналізм: ніхто не любить відчувати себе некомпетентним. Такі почуття можуть виникнути при перевантаженні користувача складною інформацією. Тому пропонувалося використання технологій прогресивного розкриття [2].

Урахування зазначених критеріїв та факторів дозволив сформувати конкурентну проєктну пропозицію, яку журі хакатону відзначило призовим місцем. Проєкт придатний для розширення функціоналу, перенесення та масштабування. За умови реалізації та впровадження даного проєкту громадський сектор суспільства зможе отримати додаткові сучасні і доступні інструменти для участі кожного мешканця у житті своєї громади.

Література:

1. Исследование: каждый третий водитель за рулем читает сообщения : веб-сайт. URL: https://rus.delfi.ee/statja/88992935/issledovanie-kazhdyy-tretiy-voditel-za-rulem-chitaet-soobshcheniya?fbclid=I

2. Nielsen Norman Group: Progressive Disclosure : веб-сайт. URL: https://www.nngroup.com/articles/progressive-disclosure/

УДК 004.414.22

ОГЛЯД СУЧАСНИХ НАПРЯМКІВ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЇ В ПРОЄКТАХ «SMART SITY»

Кузнецов А.В.

Черкаський національний університет ім. Богдана Хмельницького

Розумне місто - це все більш поширена стратегія розвитку міст, яка використовує нові технологічні досягнення для вирішення своїх проблем, часто збираючи величезні обсяги даних, які збираються за допомогою щоденних дій мешканців цього міста, для того щоб знайти найефективніший спосіб керувати певними системами в майбутньому.

Хоча питання концепції розумного міста не є новим, воно все ще перебуває на стадії концептуалізації. Це пов'язано з недостатньою послідовністю результатів досліджень, проведени багатьма різними експертними групами в різних регіонах світу, де міста забезпечували вирішення питань залежно від місцевих економічних, соціальних, технологічних та екологічних факторів. Тому експерти зосереджуються на різних визначеннях, для різних факторів як складових розумного міста.

Згідно з науково-технічною інформацією, «розумне місто - це місто, яке контролює та інтегрує елементи інфраструктури – дороги, мости, тунелі, метро, аеропорти, морські та річкові порти, водні та каналізаційні води, комунікації, що, в свою чергу, дозволяє оптимізувати міських ресурсів, і це означає максимальне надання послуг громадянам, водночас зменшуючи витрати та покращуючи рівень життя» [1].

Треба брати до уваги те, що розумне місто - це поєднання інтегрованих інфраструктур: фізичної, соціальної, ділової та IT [2].

Як можна бачити, наведені визначення незважаючи на деякі відмінності, пропонують необхідність інтеграції існуючих інформаційних систем (або більш глобальних ресурсів управління знаннями) як єдино можливий спосіб досягнення повного, ефективного, стійкого та справедливого управління бізнесом процеси, що відбуваються всередині та за межами кожного міста, прагнучи досягти розумного статусу.

Використання органів, що приймають рішення, сучасних управлінських концепцій, підходів, процедур та технологій може забезпечити новий і вкрай бажаний порядок та баланс між вимогами екологічного, соціального, економічного та технологічного аспектів [3]. Відновлення порядку та послідовності у функціонуванні міст стало однією з головних проблем влади та кожного агломераційного комплексу. Тому дуже важливо, щоб усі внесені в містах зміни включали впровадження інформаційно-комунікаційних технологій, що в результаті повинно визначати стійкий розвиток міста на основі знань яки ми отримали за попередні роки.

Як можна бачити на рис. 1, ілюструється схема основних складових розумного міста. Головна мета існування та функціонування інфраструктури

інформаційно-комунікаційних технологій необхідна для інтеграції ключової інформації, що генерується її користувачами, у кожну сферу, яка полягає у наданні повного переліку вимог, вказівок щодо технічного обслуговування та вдосконалення в аспектах: екологічне життя міст своїх громадян, громадська безпека, державні послуги та ведення будь-якої комерційної та промислової діяльності.

Рис. 1. Складові розумного міста.

Якщо більш конкретно, Smart mobility або інтелектуальна мобільність може буди представлена в різних формах виконання, одним з них може буди використаний супутниковий зв'язкок GPS. Одним з прикладів компанія «U-online» надають системи з технологіями GPS трекінгу транспорту. Цю систему треба впровадити в сітку міського транспорту для забезпечення швидкого моніторингу міських маршрутів. Smart living в перекладі розумний спосіб життя передбачає використання різних інформаційних систем для покращення та полегшення життя для пересічних громадян. Це може бути система онлайн-запису на візит до лікаря «Мій медкабінет», що набагато допомагає та економить час. Smart governance, що означає інтелектуальне управління, в цій сфері одним з рішень було запроваджено систему «Електронний кабінет» в якому здається вся фінансова звітність по організації або ФОП.

Загалом модель «розумного міста» складається з шести основних складових, які представлені на рис. 2, технологічні, економічні, екологічні та соціальні цілі слід розглядати компонентами рівнів чи складових концепції «розумного міста».

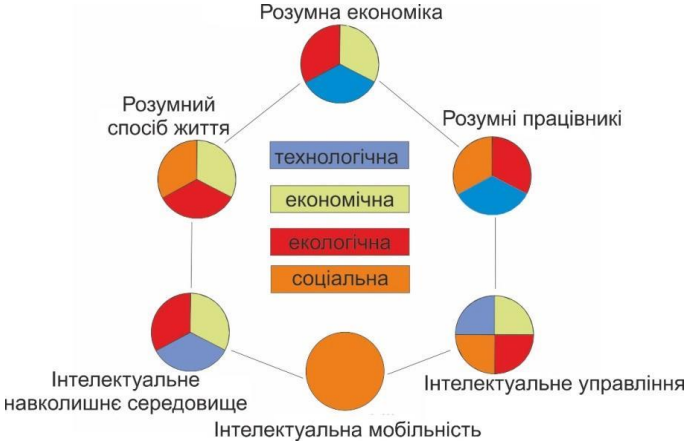


Рис. 2. Модель розумного міста.

Таким чином, модель «розумне місто» може бути втілена в життя за умови формування: «розумної економіки», «розумних працівників», «інтелектуального управління», «інтелектуальної мобільності», «інтелектуального навколишнього середовища», «розумного способу життя».

Першочерговим завданням, яке здебільшого вирішувало підвищення рівня інформатизації діяльності всіх підсистем міста, якщо розглядати концепцію «розумного міста» з точки зору інформаційних технологій, то вона вимагає організаційного забезпечення обміну даними між об'єктами міської інфраструктури, жителями, представниками міської адміністрації, співробітниками організацій, що працюють в сфері міського господарства, контрольно-наглядовими службами [4]. Таким чином, формується єдине інформаційне середовище «розумного міста».

Література:

1. Hall R. E, Vision of the smart city: веб-сайт. URL: https://www.osti.gov/biblio/773961 (дата звернення: 14.04.2021)
2. Чукут С. А., Дмитренко В. І. Смарт-сіті чи електронне місто: сучасні підходи до розуміння впровадження е-урядування на місцевому рівні. Інвестиції: практика та досвід. 2016. № 13. С. 89–93.
3. Smart Governance a zarządzanie rozwojem w mieście przyszłości. Architektura, Wydawnictwo Politechniki Krakowskiej, Kraków 2012, zeszyt 1, University of Technology Publishing Office in Krakow, Vol.1 2012
4. Касич А.О., Федоряк Р.М., Собяніна А.П., Інноваційна технологія "smart city" як механізм покращення рівня життя в сучасному місті. Науковий вісник Міжнародного гуманітарного університету. 2017. С. 50–51

УДК 004.041

СИСТЕМА ЕКСПЕРТНОГО ОЦІНЮВАННЯ АЛЬТЕРНАТИВ

Чемерис М.М.

Черкаський національний університет ім. Богдана Хмельницького

Основна ідея експертних оцінок полягає у використанні інтелекту людей, тобто їх здатності шукати та знаходити розв'язки слабо формалізованих задач. Крім того використання колективного знання відкриває ширші можливості для пошуку найбільш прийнятних та ефективних рішень [1].

Експертні методи використовуються у сфері прогнозування розвитку науки і технологій, політичних подій, бізнесу, соціальної сфери, зокрема вибору та реалізації регіональних проектів. Очевидно, що останні роки державні органи управління все частіше визнають значення колективного мислення цивільних, громадських організацій, академічних кіл тощо.

Разом з тим, одним з недоліків експертного методу оцінювання виділяють відстань між експертами. Вона може зробити процес отримання даних дуже тривалим. Внаслідок цього, актуальним є застосування різних веб-сервісів, зокрема, веб-орієнтованого середовища обліку, оцінки та планування регіональних заходів, що дозволить зацікавленим особам як запропонувати власні проєкти для обговорення та оцінювання, так і самим виступити в якості експертів для інших проєктів [2].

Іншим недоліком також виділяють суб'єктивізм у позиціях до розгляду проблеми. Суб'єктивна оцінка залежить від індивідуальних особливостей експертів, їх досвіду, кваліфікації, ерудиції, особистих смаків [1] . Це, в свою чергу, шкодить загальній об'єктивній оцінці предмету дослідження. Рішенням цієї проблеми може стати застосування технологій оцінювання думок самих експертів. В такому випадку індивідуальні відгуки розглядаються як випадкові величини та обробляються статистичними методами. Таким чином, сучасна експертиза - це система організаційних, логічних та математико-статистичних процедур, направлена на передачу від спеціалістів інформації та аналіз її з метою вироблення оптимальних рішень.

Найпростішим і найбільш застосовуваним методом проведення експертизи вважається метод переваг, або ранжування. Згідно цього методу, експерти розставляють оцінювані об'єкти за рангами у порядку зниження якості. Місце, яке зайняте кожним об'єктом, визначає число балів: чим більше (тим менше) сума балів, тим вище зайняте місце [3].

Одержані експертні оцінки перевіряються на узгодженість та надійність. Для цього для кожної і-ої альтернативи розраховуються варіація показників Var, дисперсія D, середнє квадратичне відхилення σ та коефіцієнт варіації V (1-5):

$$Var_i = x_{i\max} - x_{i\min} \tag{1}$$

$$D_i = \frac{\sum_{j=1}^{J}(x_{ij} - \overline{X}_i)^2}{J} \tag{2}$$

$$\sigma_i = \sqrt{D_i} \tag{3}$$

$$V_i = \frac{D_i}{\overline{X}_i} \cdot 100\% \tag{4}$$

$$\overline{X}_i = \frac{\sum_{j=1}^{J} x_{ij}}{J} \tag{5}$$

де x_{imax}, x_{imin} – максимальне та мінімальне значення кількісної оцінки і-ої альтернативи, проставленої експертами;

$\overline{X}_i$ – середня оцінка і-го об'єкту (проєкту, рішення);

x_{ij} – оцінка і-го об'єкту j-тим експертом;

J – загальна кількість осіб, задіяних в експертизі.

Чим вище середня оцінка альтернативи $\overline{X}_i$, тим більша імовірність того, що саме даний проєкт чи рішення прийметься до реалізації. Високе значення коефіцієнту варіації V вказує на те, що оцінки різних експертів x_{ij} значно відхиляються від середньої оцінки $\overline{X}_i$, низьке значення – варіанти оцінок зосереджені біля середнього значення.

Статистична значимість отриманих результатів визначається розрахунком довірчого інтервалу, в який з певною похибкою попадає величина, що оцінюється.

Так спільна для групи експертів імовірнісна оцінка за умови 5% похибки знаходиться в інтервал (6):

$$\overline{X}_i - 2{,}23 \cdot \frac{D_i}{\sqrt{J}} \le x_{ij} \le \overline{X}_i + 2{,}23 \cdot \frac{D_i}{\sqrt{J}} \qquad (6)$$

Програмно реалізований метод зображено на рис. 1.

Оцінки експертів, що не потрапили в довірчий інтервал, програма зафарбовує в червоний колір. По можливості слід визначити, який з експертів поставив найбільшу кількість таких оцінок, що не ввійшли в довірчі інтервали, і визначити причину цього. З точки зору статистики, дані оцінки вважаються випадковими тому у деяких методиках експертних опитувань відкидаються і не враховуються при визначенні кінцевого результату [3].

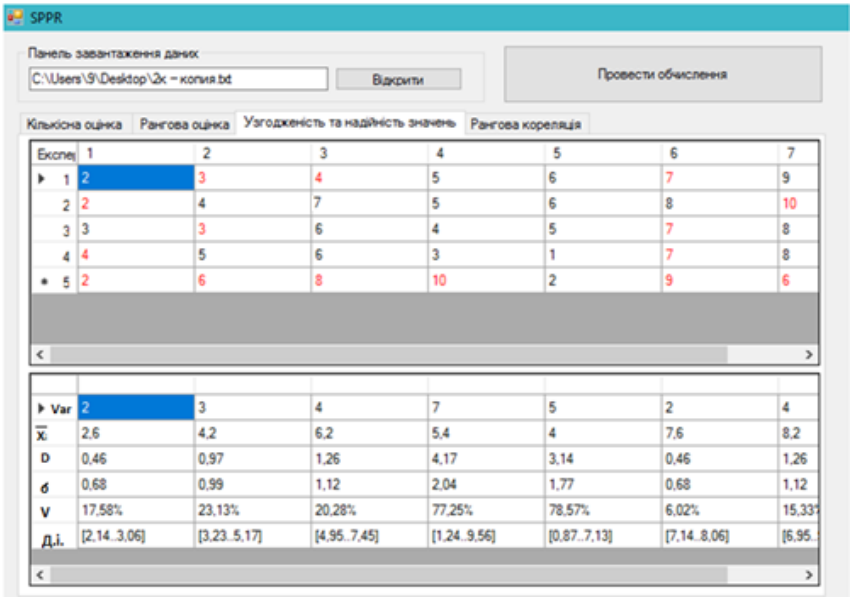


Рис. 1. Інтерфейс системи експертного оцінювання альтернатив.

В інших методиках оцінювання може повторюватись до тих пір, поки експерти не прийдуть до спільної думки. Критерієм цього може бути, наприклад, довжина довірчого інтервалу - як тільки він стане менше, голосування припиняється.

Література:

1. Кількісні методи експертного оцінювання : наук.-метод. розробка / уклад. : В. П. Новосад, Р. Г. Селіверстов, І. І. Артим. - К. : НАДУ, 2009. - 36 с.

2. Чемерис М.М. Передумови створення Web-сервісу обліку та планування заходів міста / М.М. Чемерис, М. В. Житков // Матеріали міжнародної наукової інтернет-конференції «Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 39)». – Тернопіль. – 2019. – 140 с. Режим доступу - http://www.konferenciaonline.org.ua/arhiv-konferenciy/arhiv-konferenciy11-06-2019.
3. Федорец А.Г. Вероятностно-статистические методы оценки профессиональных рисков [Текст] / Федорец А.Г.// Безопасность в техносфере. - 2007. - № 3. - С. 41-56.

УДК 004.9

АНАЛІЗ ДИНАМІКИ ОБ'ЄКТІВ ЗА ТЕХНОЛОГІЄЮ RTLS

Іванов М.О., Ярмілко А.В.

Черкаський національний університет ім. Богдана Хмельницького

Система позиціювання об'єктів у реальному часі RTLS (Real-Time Locating System) – це система, яка відповідає за визначення місцезнаходження певних підконтрольних об'єктів, зазвичай в межах визначеної території. Зазвичай дана система обробляє та зберігає інформацію про рух людей, будь-яких неживих об'єктів, зокрема – смартфонів та всіх видів транспортних засобів.

RTLS відкриває величезний спектр нових можливостей для різних сфер діяльності. Вона є актуальною починаючи від автоматизації певних небезпечних для людей процесів, таких як робота у шахтах, будівництво нових або реконструкція старих будівель, де завжди є небезпека від можливих обвалів та руйнації конструкцій, закінчуючи моніторингом переміщення автомобілів швидкої допомоги, поліцейських тощо при пошуку найближчого патруля для виклику.

Технологій для реалізації таких систем існує багато, але основними та найпопулярнішими є:

- радіочастотні технології (Wi-Fi, Bluetooth);
- супутникові технології (GPS);
- ультразвукові технології;
- радіочастотні мітки.

Більш грубо їх можна поділити на технології локального та глобального позиціювання. Локальне позиціювання має дуже високу точність визначення положення об'єкта, але працює лише на обмеженій території внаслідок потреби встановлення великої кількості обладнання як у самій робочій області, так і на кожен об'єкт, за яким планується спостереження. Щодо глобального позиціювання, то воно програє у точності визначення координат, але вимагає лише наявності будь-якого GPS-приймача, який приймає та обробляє сигнали з супутників та, в свою чергу, є доступним на кожному сучасному смартфоні.

У даному дослідженні розглядався саме варіант з використанням глобального позиціювання, тому що він дозволяє працювати з будь-яким об’єктом, обладнаним GPS-приймачем, в будь-якій точці світу, будь-то літак чи корабель. Спрощений процес роботи системи представлено на рис. 1.

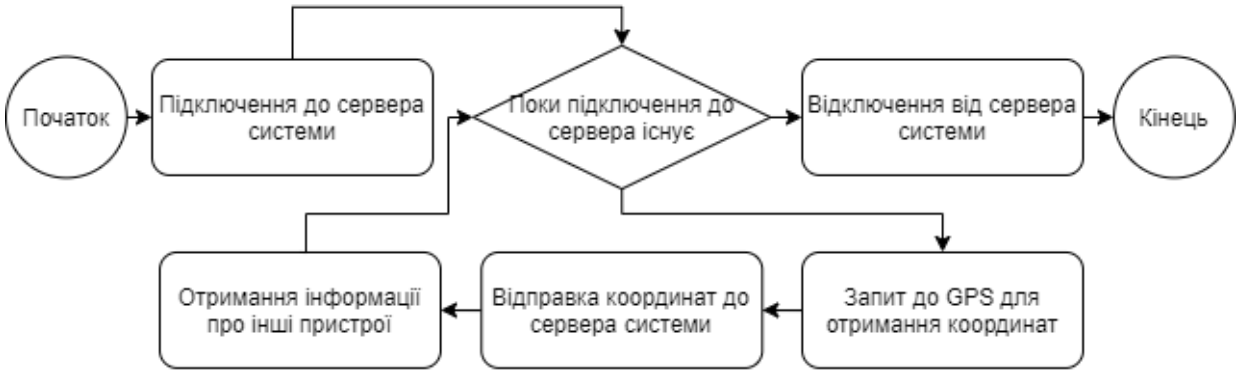


Рис. 1. Діаграма роботи пристрою системи з використанням GPS.

Як видно, на кожній ітерації циклічного процесу виконується запит до супутників системи GPS для уточнення свого місцезнаходження, після чого пристрій відсилає здобуті дані до сервера та отримує інформацію про інші під’єднанні до сервера пристрої, якщо такі є. Також паралельно до цього процесу можуть виконуватися якісь інші операції збору інформації для подальшої аналітики, наприклад, розрахунок швидкості руху, прискорення, відстань до інших об’єктів, довжину пройденого шляху тощо.

Прототип системи було розроблено на основі доступних модулів – смартфонів на платформі Android з вбудованими GPS-приймачами (рис. 2).

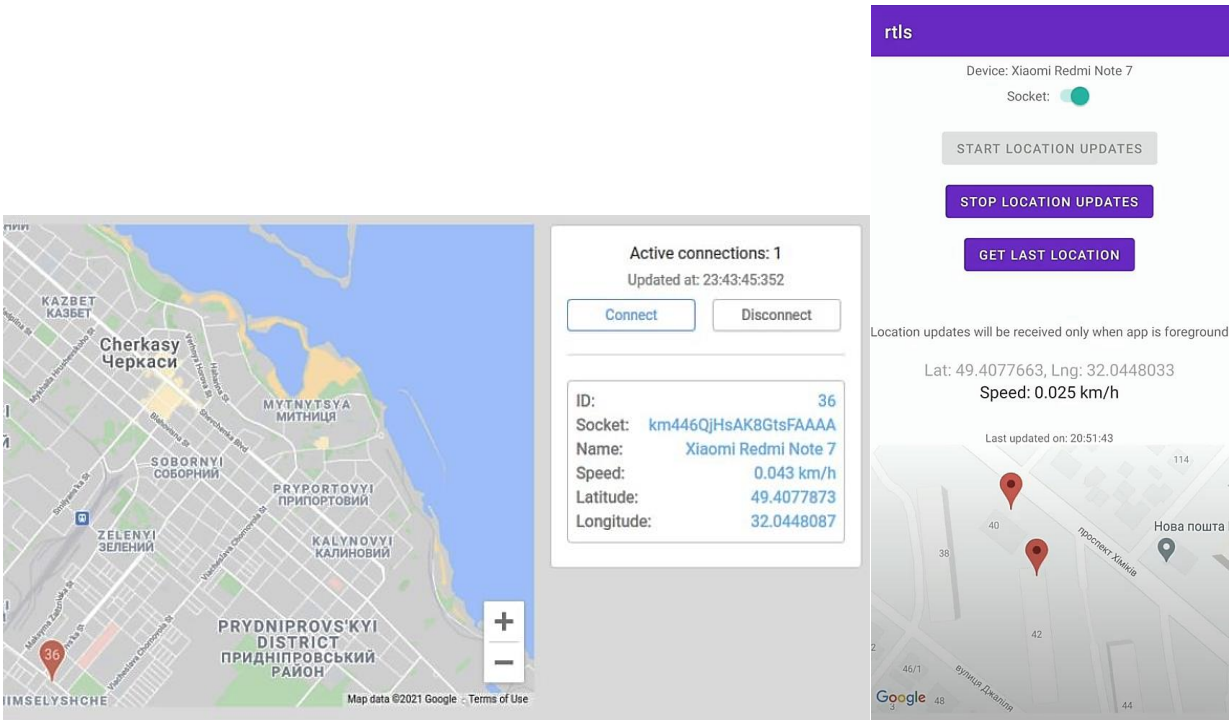


Рис. 2. Інтерфейси додатків системи.

Також сама операційна система Android має певний функціонал, який спрощує роботу з GPS, та додаткові потрібні можливості. Для передачі інформації між сервером та клієнтом було використано сокети, які були створені для передачі даних в реальному часі та оптимізовані для великої кількості одночасних підключень. В якості компонента, який відповідає за моніторинг всіх актуальних пристроїв, було розроблено веб-додаток для доступності на будь-якій платформі.

Інтерфейс Android-додатку за допомогою вбудованих можливостей смартфона зчитує власне положення та швидкість руху. Також доступною є карта, на якій зображені інші під'єднані до системи моніторингу пристрої, рух яких відображається з мінімальною затримкою. У веб-додатку все виглядає трохи інакше. В ньому також є карта, але вже з нумерацією маркерів приєднаних пристроїв. На панелі праворуч є список всіх під'єднаних пристроїв з інформацією про них, вона оновлюється у реальному часі.

Аналізуючи результати експериментального дослідження функціонування розробленого прототипу, можна зробити висновки, що дана система є відповідною для ситуацій, в яких агенти формують свою поведінку відповідно до взаємного положення відносно рухомих сусідів. Знаючи координати, швидкість руху та його напрямок, об'єкти зможуть комунікувати між собою та взаємодіяти при виконанні групових завдань без втручання людини. Однак при експлуатації системи слід враховувати принципові недоліки роботи глобального позиціювання (неідеальна точність, особливо у закритих приміщеннях, обмеження на роздільну здатність). При подальшому розширенні функціоналу даного додатку його можна буде легко адаптувати під будь-яку сферу діяльності з просторово розподіленою мережею взаємопов'язаних рухомих агентів.

Література:

4. RTLS – Вікіпедія : веб-сайт. URL: https://uk.wikipedia.org/wiki/RTLS (дата звернення: 14.05.2021).
5. GPS – Вікіпедія : веб-сайт. URL: https://uk.wikipedia.org/wiki/GPS (дата звернення: 14.05.2021).

Секція 6

Інформаційні технології в освіті

UDC 004.94

A WEB SERVICE TO DETECT MODELING MISTAKES IN BPMN 2.0 BUSINESS PROCESS MODELS

Kopp A.M., Orlovskyi D.L.

National Technical University "KhPI"

Business Process Management (BPM) helps to analyze organizational activities in order to provide quality products and (or) services and to find the ways to improve these activities. Business processes could be considered as scenarios of activities (or tasks) driven by events and decisions. Business process models are used to describe business processes and help to design and analyze information systems (IS), and also to communicate with stakeholders. In the BPM lifecycle there are two "weak" spots, where the business process models of poor quality may affect a whole success of the IS re-design or business process re-engineering project (see Fig. 1) [1]:

- business process discovery, after which the "as-is" model is created;
- business process re-design, after which the "to-be" model is created.

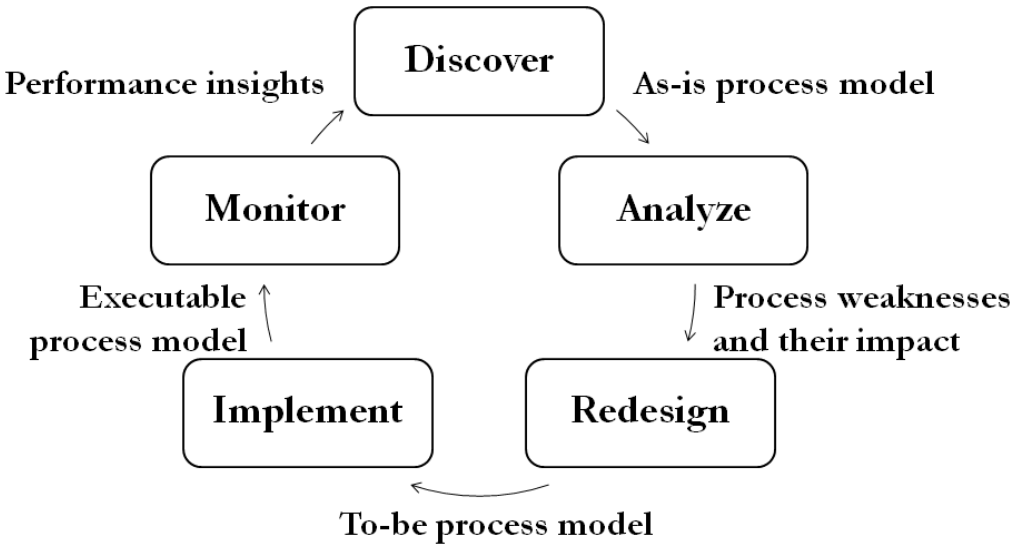


Fig. 1. Generic phases of a BPM lifecycle.

Nowadays in BPM and close industries, BPMN (Business Process Model and Notation) models are used as the de-facto standard for business process description, analysis, and improvement. There are following benefits of using BPMN:

- BPMN is the Object Management Group (OMG) standard, which means it is developed by the group of practitioners and researchers, continuously revised, supported, and updated;
- BPMN models serve as the main tool for collaboration between stakeholders, as well as for alignment of business and IT (Information Technology);

– BPMN 2.0 standard (appeared later after original BPMN was introduced) has certain extensions for automation of depicted business processes and supports XML-based (eXtensible Markup Language) inter-exchange document format for executable business process models.

Visual BPMN 2.0 models (also referred as diagrams) are based on the following graphical elements (see Fig. 2) [2].

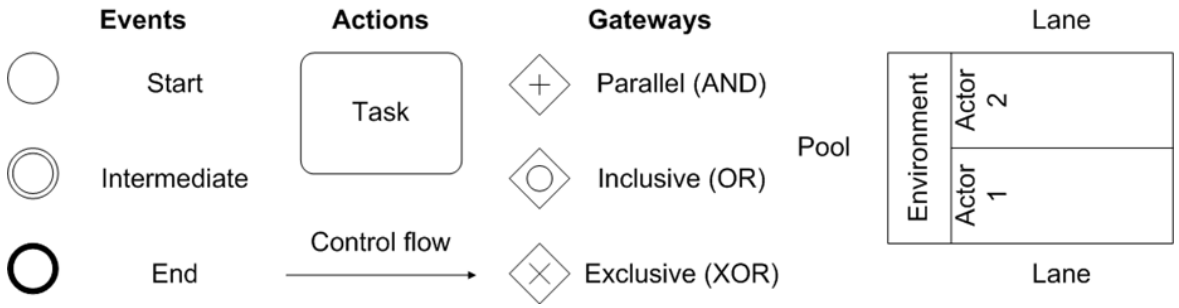


Fig. 2. Generic phases of a BPM lifecycle.

All of these elements connected together using control flows or sequence flows form a BPMN process model, which graphically represents a real business process that is already executed somewhere in organization or planned to be deployed.

However, when creating such business process models, it is quite easy even for experienced business analysts to make several mistakes, especially when models are large enough or there are too many business processes to be modeled. For beginners in BPMN modeling, the diagramming activity becomes even more complicated, so they make more mistakes in simpler business process models.

Commonly, there are two main types of mistakes presented in business process models:

– large model size (a number of activities exceeds the limit of human perception);
– incorrect usage of elements (diagram nodes, such as tasks, events, and gateways are incorrectly connected to each other).

Considering these mistakes, there were formulated following business process modeling rules:

1. There should be up to 8 activities (since human perception allows to consider between 7 and 9 objects at once) depicted in a business process model. Larger models should be decomposed using sub-processes.

2. Tasks should have one incoming and one outgoing sequence flow. There should not be tasks that implicitly start or end the business process (start or end events should be used for this purpose respectively). Also there should not be activities that implicitly split or join a business process workflows (gateways of a certain type – OR, XOR, or AND – should be used for this purpose).

3. Start events should have one outgoing sequence flow. Otherwise the start event is useless, since it does not trigger execution of any business process.

4. End events should have one incoming sequence flow. Otherwise the end event is useless, since it does not finish execution of any business process.

5. Intermediate events should have only one incoming and one outgoing sequence flow. The motivation is the same as for tasks.

6. Gateways should be either splits (one incoming and multiple outgoing sequence flows) or joins (multiple incoming and one outgoing sequence flows).

Using these rules, we formalized them into a web service for quality analysis of BPMN 2.0 models, which can detect business process modeling mistakes based on formulated rules 1–6. It is user interface is demonstrated in Fig. 3 below.

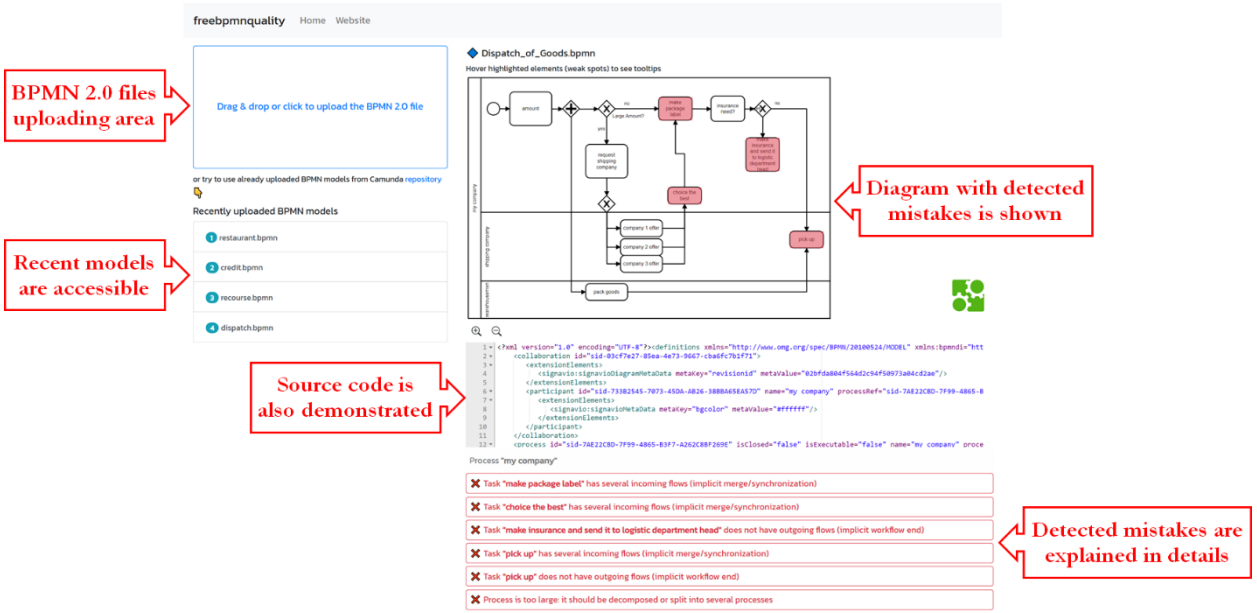


Fig. 3. User interface of the web service for BPMN 2.0 models analysis.

Proposed solution contains a web page, where users are able to upload BPMN files, view displayed diagrams with highlighted mistakes, and get detailed comments on detected mistakes (see Fig. 3). For experienced users besides the low-code model there is also demonstrated the source code of the executable BPMN 2.0 file.

Presented service is available at [3] and for now it was used by specialists from France, United States, Germany, Canada, Greece, Philippines, and Hong Kong.

References:

1. Fundamentals of Business Process Management / M. Dumas et al. Heidelberg: Springer, 2016. 527 p.

2. Ceballos, H. G., Flores-Solorio, V., Garcia, J. P. A Probabilistic BPMN Normal Form to Model and Advise Human Activities. International Workshop on Engineering Multi-Agent Systems. 2015. P. 51–69.

3. freebpmnquality – Free Online BPMN Quality Analysis Tool. URL: https://freebpmnquality.github.io/demo.html (date of access: 01.05.2021).

УДК 004.94

ДОСВІД ВИВЧЕННЯ МЕХАТРОННИХ СИСТЕМ ЗА ДОПОМОГОЮ MODELICA ТА ARDUINO

Копей В.Б., Бурак О.В.

Івано-Франківський національний технічний університет нафти і газу

Вивчення мехатроніки потребує ефективних засобів імітаційного моделювання мехатронних систем та недорогих засобів для реалізації різноманітних лабораторних стендів і експериментування з ними. Одними з таких засобів є мова моделювання Modelica та програмно-апаратна платформа Arduino [1].

Метою роботи є опис принципів вивчення мехатроніки за допомогою середовища OpenModelica, бібліотек PlanarMechanics [2] та Arduino [3], деталей для плати Arduino, а також мови програмування Python.

На рис. 1 показано модель простого лабораторного стенда для вивчення мехатронних систем [4].

Механічна частина стенда являє собою маятник. Для побудови моделі маятника використано компоненти бібліотеки PlanarMechanics. Альтернативою можуть бути 3D компоненти MultiBody стандартної бібліотеки мови Modelica, але використання 2D компонентів набагато простіше початківцям. Зауважимо, що OpenModelica 1.16.2 має помилку, яка полягає в несумісності установленої версії Modelica 3.2.3 та PlanarMechanics 1.5. Потрібно завантажити стару версію PlanarMechanics 1.4.1 з офіційного репозиторію і розпакувати в папку OpenModelica1.16.2-64bit\lib\omlibrary\PlanarMechanics 1.4.1.

Відкрити цю бібліотеку в OpenModelica Connection Editor можна шляхом перетягування файлу package.mo на вікно браузера бібліотек. Перед використанням бібліотеки рекомендується ознайомитись з вбудованими прикладами – у першу чергу з прикладом Pendulum (вільні коливання плоского маятника) та PendulumExcited (вимушені коливання плоского маятника).

Маятник (рис. 1) будується з компонентів fixed (зафіксований фланець), revolute (шарнір), fixedTranslation (жорстка ланка заданої довжини без інерції), body (тіло з масою та моментом інерції), planarWorld (система координат та поле гравітації). Компонент cutForce не є обов'язковим і може бути використаний для вимірювання сили між фланцями.

Перед студентами можуть бути поставлені різні завдання програмування мікроконтролера – гальмування вимушених коливань, підтримка заданої амплітуди тощо.

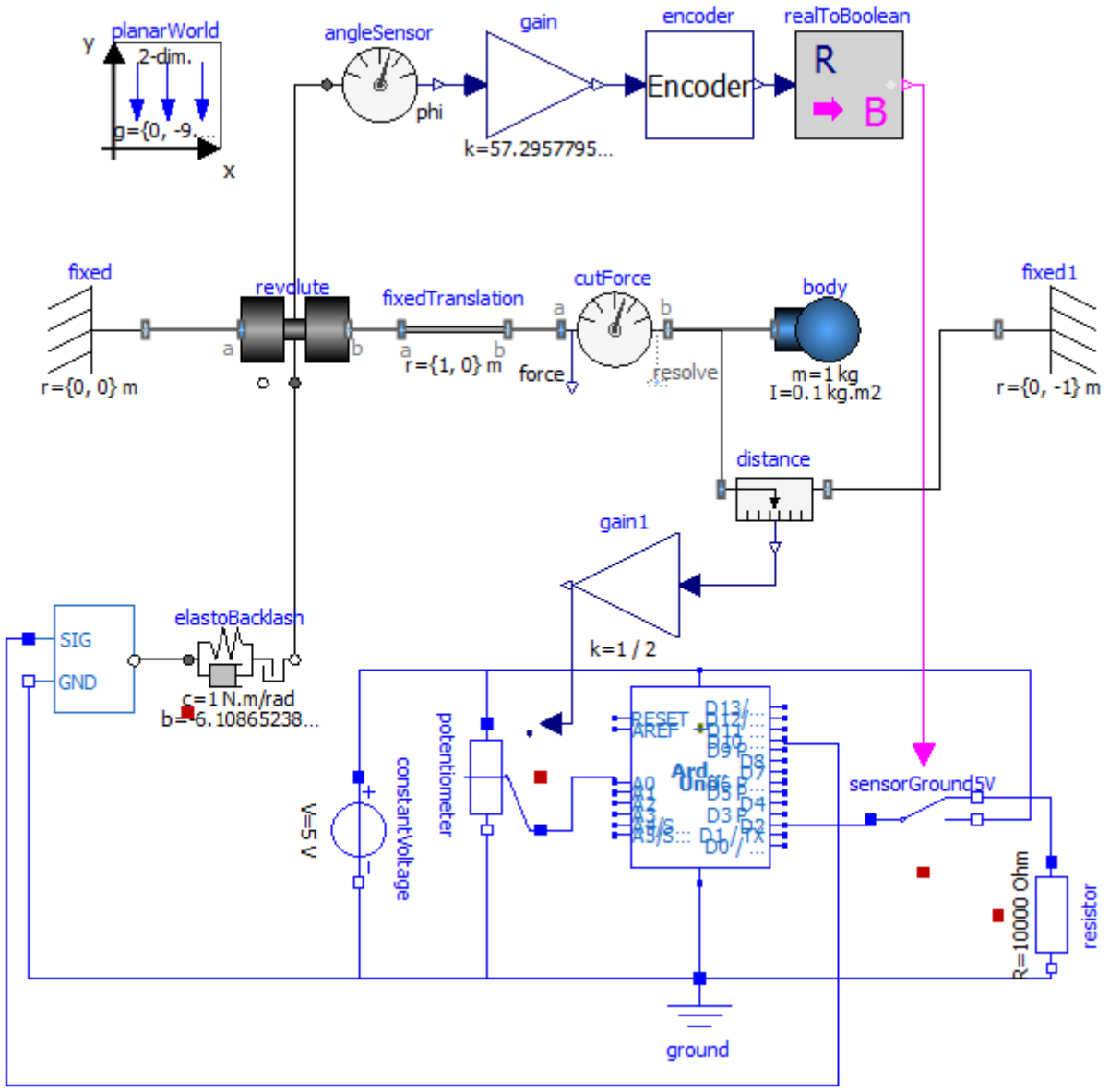


Рис. 1. Модель мехатронної системи

В даному випадку ціллю системи автоматичного керування є підтримування коливань маятника. Система досягає цієї цілі шляхом визначення положення маятника за допомогою сенсорів та підштовхування маятника за допомогою актуатора, якщо рух відсутній. Використовуються два сенсори – інкрементний обертовий енкодер та сенсор відстані на основі світлодіода та фоторезистора. Модель енкодера складається з компонентів angleSensor (сенсор кута повороту), gain (множення на 57.3 градуса), encoder (перетворює дійсний сигнал кута повороту в булевий), sensorGround5V (перемикач, в залежності від стану якого цифровий порт Arduino D2 отримує 0 або 1). Модель сенсора відстані складається з компонентів distance, який вимірює відстань між body і нерухомим фланцем fixed1 і передає сигнал на potentiometer (керований цим сигналом змінний резистор, середній контакт якого з'єднаний з аналоговим входом A0). Для моделювання актуатора використовується модель сервопривода з бібліотеки Arduino та компонент elastoBacklash (1D обертова пружина-демпфер з зазором), який під'єднаний

до 1D обертового фланця шарніра. Застосовується бібліотека Arduino 0.1.0 [3], яка створена для використання в Dymola, тому є певні особливості її використання в OpenModelica 0.16. На комп'ютері повинна бути установлена Microsoft Visual Studio 14. Командний файл \lib\omlibrary\Arduino 0.1.0\Resources\Source\Arduino\build_sketch.bat дозволяє відкомпілювати довільний скетч Arduino. Але у будь-якому випадку шлях до нього повинен бути незмінним: \lib\omlibrary\Arduino 0.1.0\Resources\Sketches\Blink.ino. В результаті компіляції отримується бібліотека ModelicaArduino.dll, яку потрібно скопіювати в папку з моделлю перед стимуляцією. Компонент encoder відсутній в стандартній бібліотеці Modelica, але його можна легко створити шляхом успадкування класу блокового компонента SISO. Для прикладу код моделі encoder може бути таким [4]:

```
block Encoder
  extends Modelica.Blocks.Interfaces.SISO;
  parameter Real angle(start=90);
equation
  if noEvent(mod(u,angle)>=0 and mod(u,angle)<=angle/2) then
     y = 1;
  else
     y = 0;
  end if;
end Encoder;
```

Тут для ідентифікації отвору на диску оптичного енкодера використовуються операція остачі від ділення mod. Змінна u – це кут повороту диску, y – цифровий вихід енкодера, параметр angle – кутовий крок отворів.

Ця модель реалізована в простому лабораторному стенді на основі недорогих компонентів: Arduino Uno, сервопривід SG90, енкодер на LM393, світлодіод, фоторезистор, деталі металевого конструктора [4]. В Arduino завантажено скетч, що дозволяє обмінюватися даними з персональним комп'ютером за допомогою протоколу firmata [5]. На ПК виконується Python-програма, яка керує системою [4]. Керувати можна також виключно за допомогою Arduino-скетча [4], але Python і ПК мають більше можливостей, зокрема, дозволяють візуалізувати сигнал датчиків в реальному часі та застосовувати більш складні алгоритми керування.

Такий підхід до вивчення мехатронних систем має свої переваги. Можна створювати будь-які мультидоменні моделі (у тому числі з використанням Arduino) у вільному середовищі OpenModelica, а також можливо відразу реалізувати систему з недорогих компонентів та верифікувати результати симуляції шляхом експерименту. З кодом моделей і програм, які розроблені авторами для розглянутої системи, можна ознайомитись на GitHub [4].

Література:

1. Arduino Uno Rev3. URL: https://store.arduino.cc/arduino-uno-rev3.

2. A free Modelica library for planar mechanical multi-body systems. URL: https://github.com/dzimmer/PlanarMechanics.
3. Simulate circuits and sketches on a virtual Arduino Uno in Modelica. URL: https://github.com/CATIA-Systems/Modelica-Arduino.
4. Arduino Pendulum - Modelica-model and implementation. URL: https://github.com/vkopey/mechatronics2.
5. firmata/protocol. URL: https://github.com/firmata/protocol.

УДК 531

РОЗВИТОК ПРОГРАМНИХ СИСТЕМ ЗАБЕЗПЕЧЕННЯ ТЕХНОЛОГІЇ ДОПОВНЕНОЇ РЕАЛЬНОСТІ В ОСВІТІ ПІД ЧАС ПАНДЕМІЇ

Жуков Б.С.

Черкаський національний університет ім. Богдана Хмельницького

Сьогодні можна сміливо говорити про стрімке старіння традиційних віконних графічних інтерфейсів, керованих клавіатурою і мишкою, початок яким було покладено ще в 80-ті роки минулого століття. Стрімкий розвиток інтерактивних мультимедійних технологій вимагає появи новітніх інтерфейсів людино-машинної взаємодії, які мають елементи тривимірного світу, який звичний для сприйняття користувачем.

З розвитком технологій та інфраструктури та світом, який назавжди змінився пандемією, доповнена реальність рухається швидкими темпами без ознак її уповільнення.

Пандемія COVID-19 змусила світ зупинити своє звичне функціонування та змусила шукати нові шляхи для існування. Технології стали рятівним кругом як для бізнесу, так і для звичайних користувачів, надаючи можливість продовжувати виконувати роботу та залишатися на зв'язку з друзями та родиною. Під час перебування вдома, наше використання та покладання на такі технології, як відеоконференції та доповнена реальність, триватиме і цього року і, швидше за все, залишиться після пандемії.

Технології розширеної реальності (англ. Extended reality), включаючи віртуальну реальність (VR), доповнену реальність (AR) та змішану реальність (MR), все ще перебувають на початковій фазі впровадження, але вони швидко розвиваються. Хоча більшість випадків використання VR та AR все ще пов’язані з іграми, розвагами та соціальними медіа, різноманітність програм розширюється, оскільки все більше споживачів та підприємств випробовують цей захоплюючий досвід.

Очікується, що 58,9 мільйона людей в США будуть використовувати VR і 93,3 мільйона будуть використовувати AR щонайменше раз на місяць у 2021 році. Ці цифри становлять 17,7% та 28,1% населення США відповідно.

Оскільки пандемія змусила багатьох людей працювати, спілкуватися, вчитися та робити покупки вдома, вони використовують досвід XR для заміни особистого досвіду. Індуковані пандемією умови викликали інтерес

до обох технологій, і розробники активізують роботу, щоб задовольнити підвищений попит. Впровадження 5G, штучного інтелекту та хмарної обробки даних незабаром полегшить надання безперебійного, приємного та економічного досвіду XR на різноманітних підключених пристроях.

Обидві технології (AR та VR) мають очевидні переваги для навчання та тренувань, що спостерігається у школах та на курсах підвищення кваліфікації. Наприклад, люди, які навчаються будівництву або роботі на будь-яких небезпечних або складних локаціях, можуть тренуватися без будь-яких ризиків, які зазвичай пов'язані з ними.

Використовуючи AR, початкові фахівці в цих галузях зможуть навчатись у змодельованому середовищі, яке забезпечує значний ступінь точності, уникаючи ризиків, пов’язаних з помилкою. AR також може бути корисним, оскільки працівники зможуть практикувати завдання в режимі реального часу, що може покращити їхні навички та загальну продуктивність.

Ще до пандемії передбачалося, що 25-30% працівників працюватимуть з дому кілька днів на тиждень до кінця 2021 року. Спільні події, такі як конференційні дзвінки, часто підриваються відсутністю безпосередньої особистої присутності. Однак доповнена реальність може створювати параметри змішаної реальності, де всі учасники конференц-зв'язку можуть бачити одне одного в більш соціально сприятливих умовах.

Microsoft рухається вперед, використовуючи бета-версію системи відеодзвінків, яка використовує доповнену реальність для створення голографічних зображень учасників. Cisco Systems також працює над проектом під назвою Musion, який об'єднує свої мережеві продукти з AR-технологіями.

Сеанси віддаленої допомоги на основі AR - це варіант використання, який може сприяти інноваціям. Поєднання WebRTC та AR дозволяє проводити роботи з технічного обслуговування та усунення несправностей у режимі реального часу. Використовуючи паралельну потокову передачу даних, постачальники допомоги можуть безпосередньо долучитися до процесів обслуговування, конфігурації та ремонту.

Навчальні заклади найбільше постраждали від політики соціального дистанціювання в умовах пандемії COVID-19 2020 року. Однак доповнена реальність має ряд додатків, які можуть допомогти покращити досвід навчання для студентів, які навчаються в режимі електронного навчання. Wikitude створив додаток під назвою Ai.R-Cord. Ця програма спрямована на допомогу учням початкових класів у вивченні понять та термінів за допомогою досвіду доповненої реальності.

Потенціал доповненої реальності збільшити увагу студентів на зміст курсу, перебуваючи вдома, не можна недооцінювати. Ця технологія також робить режими навчання вдома більш різноманітним, розширюючи візуальний вміст для учнів, зорієнтованих на візуальне сприйняття інформації. Це повинно допомогти урізноманітнити монотонні

відеоконференції та записані конспекти лекцій, щоб покращити залучення студентів.

Однією з головних переваг доповненої реальності в освітньому просторі є можливість студента самостійно перевіряти модель із різних кутів. Пересуваючись навколо віртуального об'єкта або обертаючи його в просторі, вони можуть краще вивчити і зрозуміти певні поняття. Найголовніше, що це дозволяє студентам отримати перевагу домашнього експериментального навчання. Цей вид навчання швидше запам'ятається і буде більш зрозумілим студентам порівняно з іншими методами.

Безпека та довіра є першорядними для всіх технологічних рішень, але особливо важливі для AR та AI, де комп'ютери сканують і редагують навколишній світ. По мірі впровадження та прийняття доповненої реальності, зростає потреба у структурах, нормативних актах та соціальних контрактах, які тримають права людини в центрі, особливо в міру того, як ми рухаємось до носних пристроїв таких як окуляри доповненої реальності. Технологічні компанії, уряди, органи стандартизації та кінцеві споживачі повинні оцінювати технології, дивлячись не лише на їх достоїнства, а й враховуючи їх вплив на людину, дивлячись на них з точки зору конфіденційності, доступності, різноманітності та стійкості. Ранні рішення, прийняті цими інститутами сьогодні, сформують доповнену реальність майбутнього.

Література:

1. Augmented Reality Trends 2021: What to expect from AR this year. [Електронний документ]. Режим доступу: https://www.reydar.com/augmented-reality-trends-2021/
2. Augmented/Virtual Reality Strategies for 2021 and Beyond. [Електронний документ]. Режим доступу: https://nodo.cc/drops/en/augmented-virtual-reality-strategies-for-2021-and-beyond/
3. 21 Augmented Reality Trends to Keep an Eye on for 2021. [Електронний документ]. Режим доступу: https://www.linkedin.com/pulse/21-augmented-reality-trends-keep-eye-2021-tom-emrich/
4. US Virtual and Augmented Reality Users 2021. [Електронний документ]. Режим доступу: https://www.emarketer.com/content/us-virtual-augmented-reality-users-2021
5. Trends Transforming The Augmented and VR Industry in 2021. [Електронний документ]. Режим доступу: https://linchpinseo.com/trends-in-augmented-virtual-reality/
6. 10 augmented reality trends in 2021: the future is here. [Електронний документ]. Режим доступу: https://mobidev.biz/blog/augmented-reality-future-trends-2018-2020
7. Future of augmented reality. [Електронний документ]. Режим доступу: http://www.vrs.org.uk/augmented-reality/future.html

УДК 004.91

ПРОГРАМНИЙ КОМПЛЕКС ДЛЯ УПРАВЛІННЯ ОСВІТНІМИ ПРОГРАМАМИ У ЗАКЛАДАХ ВИЩОЇ ОСВІТИ

Онищенко Б.О.

Черкаський національний університет ім. Богдана Хмельницького

Розробка, впровадження в освітній процес, та оновлення освітніх програм є невід'ємною складовою освітньої діяльності сучасних закладів вищої освіти [1]. Від організації якісного процесу управління освітніми програмами залежить як якість надання освітніх послуг за відповідними освітніми програмами так і успішна процедура акредитації самих освітніх програм експертами національного агентства із забезпечення якості вищої освіти.

Більшість закладів вищої освіти надають освітні послуги за багатьма освітніми ступенями та спеціальностями і на деяких спеціальностях відкрито більше ніж одну освітню програму. З іншого боку, міністерством освіти і науки України майже завершило затвердження освітніх стандартів по кожному освітньому ступеню та спеціальності, які обов'язково потрібно враховувати у процесі розробки нових та підтримки існуючих освітніх програм [2]. Отже розробка системи для управління освітніми програмами у закладах вищої освіти є нагальною та актуальної задачею.

Під час формування вимог до системи та розробки її структурної схеми потрібно врахувати те, що вона повинна надавати можливість працювати як із власне освітніми програмами, так і із стандартами вищої освіти відповідних спеціальностей. Але з огляду на те, що освітня програма та освітній стандарт спеціальності дещо відрізняються кількістю та складом розділів, то і інформаційна модель системи та структура її бази даних також мають врахувати усі відмінності та спільні дані таким чином, щоб мінімізувати збитковість та залишитись нормалізованими.

З боку закладів вищої освіти найчастіше працювати з зазначеною системою мають гаранти освітніх програм, тому і базовий набір її функцій має бути орієнтованим саме на них. Кожен гарант освітньої програми повинен мати можливість для створення освітньої програми на основі на основі існуючого стандарту або без нього, а також на основі уже існуючої освітньої програми у тому випадку, коли виникає необхідність у її модифікації. У випадку, коли освітня програма створена на основі вже існуючої – гаранту має бути доступна можливість порівняння змісту декількох пов'язаних між собою освітніх програм.

Для кожної освітньої програми обов'язково мають зазначатися дати її запровадження та завершення дії, рівень освіти, спеціальність, склад групи забезпечення. Однак основним модулем для роботи з освітніми програмами має виступати підсистема, що пов'язує між собою компетентності, програмні результати навчання та освітні компоненти, якими будуть забезпечуватись як

програмні результати навчання, так і зазначені у освітній програмі компетентності. Результатом роботи такої підсистеми будуть матриці відповідності освітніх компонентів і програмних результатів навчання та освітніх компонентів і компетентностей. Також модуль має допомогти гарантові освітньої програми визначати покриття чи не покриття освітніми компонентами зазначених у освітній програмі компетентностей та програмних результатів навчання, що є одним із основним показників під час оцінки якості тієї чи іншої освітньої програми.

Важливою складовою системи для управління освітніми програмами має бути чітке розмежування рівнів доступу між користувачами системи. Зокрема гаранти освітніх програм повинні мати можливість, крім повного доступу до створених ними освітніх програм, також переглядати і, можливо, імпортувати зміст і інших освітніх програм як мінімум у межах однієї галузі, що дасть можливість більш ефективно організовувати роботу в межах одного навчально-наукового інституту чи факультету, на яких проводиться навчання за спорідненими спеціальностями.

Також у системі слід передбачити можливість обговорення проектів та вже існуючих освітніх програм усіма зацікавленими стейкхолдерами, адже процес обговорення тієї чи іншої освітньої програми також є невід'ємною складовою забезпечення якості вищої освіти.

В подальшій перспективі система управління освітніми програмами у закладі вищої освіти може бути інтегрованою до загальної системи управління закладом вищої освіти, що дозволить використовувати інформацію з неї під час складання навчальних планів відповідних спеціальностей.

Література:

1. Розроблення освітніх програм. Методичні рекомендації. URL: http://ibhb.chnu.edu.ua/uploads/files/metodrada/Rozroblennya_osv_program.pdf (перевірено: 04.05.2021).
2. Затверджені стандарти вищої освіти. URL: https://mon.gov.ua/ua/osvita/visha-osvita/naukovo-metodichna-rada-ministerstva-osviti-i-nauki-ukrayini/zatverdzheni-standarti-vishoyi-osviti (перевірено: 04.05.2021).

Наукове видання

ІНФОРМАЦІЙНІ МОДЕЛЮЮЧІ ТЕХНОЛОГІЇ, СИСТЕМИ ТА КОМПЛЕКСИ (ІМТСК-2021)

ТРЕТЯ МІЖНАРОДНА
НАУКОВО-ПРАКТИЧНА КОНФЕРЕНЦІЯ

27-28 травня, 2021
Черкаси, Україна

ЗБІРКА ТЕЗ

Тези надруковані в авторській редакції на одній з робочих мов конференції

Оригінал-макет
підготовлено на кафедрі
інтелектуальних систем прийняття рішень
Черкаського національного університету імені Богдана Хмельницького