Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут»
Теплоенергетичний факультет

Київська державна академія водного транспорту
імені П.Конашевича-Сагайдачного

Інститут кібернетики ім.В.М.Глушкова НАН

V науково-практична дистанційна конференція
молодих вчених і фахівців
з розробки програмного забезпечення

«СУЧАСНІ АСПЕКТИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»

15 травня 2018

м. Київ

УДК 004.42:519.7
ББК 973.20-018.2я7

Рекомендовано Вченою радою Теплоенергетичного факультету Національного технічного університету України «Київський політехнічний інститут» (протокол № від року)

Сучасні аспекти розробки програмного забезпечення: Збірник наукових праць V науково-практичної дистанційної конференції молодих вчених і фахівців з розробки програмного забезпечення, 15 травня 2018 р. – Черкаси: видавець Чабаненко Ю.А., 2018. – 216 с.

У збірнику наукових праць V науково-практичної дистанційної конференції молодих вчених і фахівців з розробки програмного забезпечення «Сучасні аспекти розробки програмного забезпечення» наведено результати наукових досліджень та практичних розробок за наступними напрямами: інженерія програмного забезпечення, комп'ютерний еколого-економічний моніторинг, системи автоматизованого проектування.

Матеріали друкуються в авторській редакції.

ISBN

© Авторські тексти, 2018

ЗМІСТ

Бадаєв Ю.І., Каценко Т.С. Використання цифрових водяних знаків для захисту графічної інформації в системах передачі даних 8

Бандурка О.І., Курсенко Л.О., Соловйов С.О. Система вибору оптимальних схем вакцинації населення з використанням епідеміологічних даних 12

Верлань А. А., Лалак Б. О. Система конвертації баз даних різних типів 19

Гайдаржи В.І., Сук С. В. Хмарна реалізація підсистеми ведення бібліотеки гідроакустичних моделей системи моделювання гідроакустичних процесів 24

Гайдаржи В.І., Чистякова Д.Ю. Загальна структура системи моделювання гідроакустичних процесів та керуючій модуль системи 34

Дацюк О.А., Амброс С.М. Розробка мобільного додатку моніторингу споживання енергії 45

Дацюк О.А., Березюк В.Д. Розробка додаткового інструментального засобу редагування онтології 51

Дровозюк В.О., Корнійкук М.А. Програмні засоби тестування й зв'язку з наносупутником 58

Дрозд Д. С. Використання алгоритмів кластеризації для побудови рекомендаційних систем сайту новин 65

Карпенко Є.Ю., Гуменний А.А. Архітектура мобільних додатків на основі ОС Android 68

Карпенко Є.Ю., Касьяненко І.І. Розподілена система керування організаційними процесами кафедри 74

Карпенко Є.Ю., Фішер О.Є. Візуалізація стану ґрунтів України на основі геоданих 76

Карпенко С.Г., Защик С.М. Сортування масивів об'єктів з використанням багатопоточності на C++ 82

Кублій Л.І., Івашин В.В. Вибір інструментів для фіксації і відстеження прогресу виправлення помилок 91

Кублій Л.І., Костенко І.П. Огляд ефективності паралельного генетичного алгоритму на базі теоретичної моделі "зірка" 97

Кублій Л.І., Костенко О.П. Проблеми і завдання взаємодії MATLAB і C# 104

Кублій Л.І., Семенчук І.О. Оцінка перспективності районів кроводач на основі аналізу статистики захворюваності населення 110

Кублій Л.І., Тобілко А.О. Захист інформації в комплексі моделювання гідроакустичних процесів 117

Левченко Л.О. Орел Д.С. Інтеграція PDM та CAD систем 122

Мажара О.О., Витвицький Д. А. Формування навчальної вибірки в задачі класифікації музичних інструментів 125

Медведєва В. М., Панченко О.О. Система фільтрації та децимації GPS-даних 131

Медведєва В.М., Симоненко Б.О. Організація оптимального обчислювального процесу в корпоративній мережі на основі платформи .NET 139

Сегеда І.В., Зіліцький В.Є. Застосування QR-кодування в Україні 145

Сич М. В. Контроль енергоефективності систем вентилювання: аналіз, проблеми та альтернативні рішення 151

Смаковський Д.С., Соломкін М.В. Підходи до побудови відмовостійких розподілених систем 162

Тарнавський Ю.А., Горб І. Роль аналітичних систем в організації оперативного енергетичного менеджменту 168

Тарнавський Ю.А., Крижанівська Ю.В. Моделювання режимів роботи інтегрованих систем енергозабезпечення 171

Тарнавський Ю.А., Смоліженко Д.П, Михайлюк А.В. Захист веб-систем на основі комп'ютерних тестів 173

Титенко С. В., Астахов А.Г. Онтологічно-орієнтована навчальна система для вивчення дисциплін історичного спрямування 180

Титенко С.В., Білоус А.О. Інтерактивний доступ до професійно-навчальної інформації 195

Шалденко О.В., Байда Д.В. Нейронні мережі як засіб для семантичної сегментації зображення 201

Шаповалова С.І., Бараніченко О.М. Встановлення зв'язків між об'єктами логічної задачі 204

Шаповалова С.І., Колот С.С. Системи моделювання Convolutional Neural Networks 209

Шаповалова С.І., Терпіль Д.О. Вдосконалення графічної симуляції навколишнього середовища 214

Щепак А. С. Моделювання динаміки розповсюлження забруднень 220

Iryna Husyeva Information and analytical system for innovation analysis..... 223

Бадаєв Ю.І., Каценко Т.С.

Використання цифрових водяних знаків для захисту графічної інформації в системах передачі даних

В даний час широко розповсюджена передача цифрових зображень від авторів споживачам або посередникам засобами Internet, а також на різних цифрових носіях. Легкість копіювання таких зображень призводить до того, що постійно зростає число порушень авторських прав на графічні роботи, в зв'язку з незаконним, несанкціонованим використанням та їх зміною третіми особами.

Актуальність досліджень у цьому напрямку витікає з необхідності розв’язування задач захисту прав власності на інформацію, яка представлена у цифровому вигляді. Одним з найбільш ефективних способів вирішення цієї проблеми є використання цифрових водяних знаків (ЦВЗ).

Видимі ЦВЗ є помітними для людського ока, тому їх досить просто видалити або замінити, використовуючи графічні або текстові редактори. Невидимі ЦВЗ – це вбудовані в графічні файли вставки, які не сприймаються людськими органами зору. Тому ЦВЗ повинні задовольняти наступні вимоги:

- непомітність для користувачів;
- можливість виявити несанкціоноване використання файлу;
- неможливість видалення неуповноваженими особами;
- стійкість до різного виду атак (до зміни зображення-контейнера, його формату і розмірів, до масштабування, стиску, повороту, фільтрації, введення спецефектів, монтажу).

Це зумовлює необхідність розробки нових програмних засобів та удосконалення алгоритмів вбудовування невидимих ЦВЗ, що забезпечують незмінність вбудованої інформації при спотворенні чи компресії зображення-контейнера та мінімальний вплив методу вбудовування ЦВЗ на якість самого зображення.

За надійністю ЦВЗ поділяють на крихкі, напівкрихкі та надійні (робастні).

Крихкі ЦВЗ руйнуються за незначної модифікації заповненого контейнера. Їх використовують для підтвердження цілісності зображення. Відмінність від засобів електронного цифрового підпису полягає в тому, що крихкі ЦВЗ все ж допускають деяку модифікацію контенту.

Напівкрихкі ЦВЗ стійкі відносно до одних дій і нестійкі відносно до інших. Майже всі ЦВЗ можуть бути віднесені до цього типу.

Серед методів вбудовування ЦВЗ виділяють дві великі групи: приховування ЦВЗ у просторовій та частотній областях зображення.

Одним з найбільш поширених методом приховування інформації в часовій (просторовій) області є метод заміни найменшого значущого біта (LSB), який однак не забезпечує високої надійності і має слабкий захист до пасивних та активних атак та є не стійким до більшості видів спотворень.

Основною вимогою до методу вбудовування ЦВЗ є забезпечення незмінності вбудованої інформації при спотворенні чи компресії зображення-контейнера та мінімальний вплив методу вбудовування ЦВЗ на якість самого зображення.

Тому найбільшого поширення набули методи дискретного вейвлет перетворення (ДВП) та дискретного косинус перетворення (ДКП), які базуються на частотних перетвореннях та забезпечують невеликий вплив ЦВЗ при його вбудовуванні на якість зображення, а також стійкість до багатьох спотворень, особливо до компресії з втратами.

Існує декілька модифікацій методів вбудовування ЦВЗ з використанням ДВП, ДКП, а також їх комбінацій, проте всі вони не є повністю робастними, тобто стійкими до атак зміни зображення-контейнера. Щоб підвищити стійкість алгоритму було запропоновано провести модифікацію алгоритму, об'єднавши дискретне вейвлет перетворення та дискретне косинус перетворення, схема якого зображена на рисунку 1.

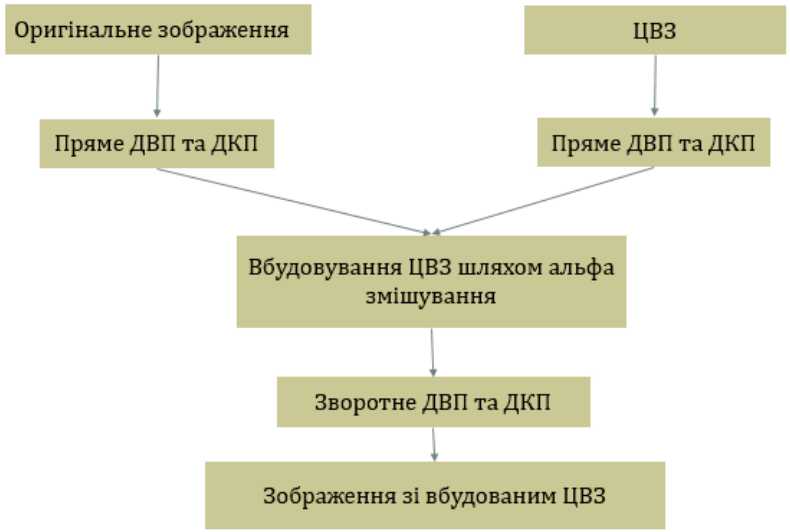


Рисунок 1 – Алгоритм вбудовування цифрового водяного знаку в зображення

Суть алгоритму полягає у вбудовуванні в почерговому застосуванні дискретного вейвлет перетворення та дискретного косинус перетворення до оригінального зображення та до зображення, що виконує роль ЦВЗ. Вбудовування відбувається методом альфа змішування, формула якого представлена нижче:

$$WMI = k*LL + q*WM,$$

де WMI – зображення зі вбудованим ЦВЗ, LL –апроксимація оригіналу зображення, WM –апроксимація ЦВЗ, k, q – масштабні коефіцієнти.

Результати аналізу показників якості та структура стеганографічного методу показали, що даний метод забезпечує досить надійний захист ЦВЗ при спотворенні чи зміні зображення-контейнера, відсутній візуальний вплив ЦВЗ на якість зображення, високу стійкість ЦВЗ до ущільнення зображення та малу стійкість до зміни кольору.

Перелік використаних джерел

1. Ingemar J., Cox, Matthew L., Miller, Jeffrey A., Bloom, Jessica, Fridrich and Ton, Kalker (2008), Digital Watermarking and Steganography, Elsevier Inc., 502 p.

2. Конахович Г. Ф., Пузыренко А. Ю. Компьютерная стеганография. Теория и практика. – К.: «МК-Пресс», 2006. – 288 с.

3. Бедратюк С. Л. Стійкий до активних атак алгоритм постановки цифрових водяних знаків / С. Л. Бедратюк // Штучний інтелект. — 2010. — № 3. — С. 291—298. — Бібліогр.: с. 298.

Бандурка О.І., Курсенко Л.О., Соловйов С.О.

Система вибору оптимальних схем вакцинації населення з використанням епідеміологічних даних

Проблема розповсюдження різних інфекційних захворювань та епідемій є важливою для всього людства. Ротавірусна інфекція (РВІ) є основною причиною гострого вірусного гастроентериту у дітей молодшого віку та високого рівня дитячої захворюваності та смертності в усьому світі. Щоб зменшити показник розповсюдження інфекції застосовують вакцинопрофілактику. В умовах обмеженості ресурсів проводити вакцинацію всього населення не є можливим, тому задача визначення оптимальних суспільних та індивідуальних витрат на вакцинацію і лікування є та буде залишатись актуальною.

Необхідно враховувати, що на даний момент існує багато підходів для вирішення задач, пов’язаних з вакцинацією, але різні захворювання мають певні особливості, що необхідно враховувати при розробці епідеміологічних моделей. Саме тому у сучасній практиці не розроблено універсального програмного продукту для моніторингу ефективності вакцинопрофілактики. Через значну варіативність та складність параметрів жодний з існуючих програмних продуктів не є універсальним засобом.

Розроблена система використовує епідеміологічну *SISV* — модель. Основний принцип побудови епідеміологічної моделі для РВІ: все населення ділиться на декілька категорій: сприйнятливі особи (S), які чутливі до інфікування; інфіковані особи (I), які заражені інфекційною хворобою, яка представляє інтерес, здатні заражати інших; вакциновані особи (V), отримали імунітет, недоступні до інфікування. Модель є динамічною системою, яка змінюється в часі,

тому при проходження певного часового періоду частина населення переміщується з однієї категорії в іншу.

Перехід осіб з однієї категорії населення в іншу характеризується за допомогою параметрів, наприклад: сила інфекції, швидкість одужання, народжуваність; смертність, міграція тощо. Сила інфекції λ — ймовірність того, що певна особа буде інфікована в певний момент часу. Швидкість одужання γ — ймовірність того, що певна особа буде вилікувана у певний момент часу.

Використання *SIS* — моделі передбачає, що інфіковані особи (I) одужають по проходженні певного часу і знову стають сприйнятливими до захворювання (рисунок 1).

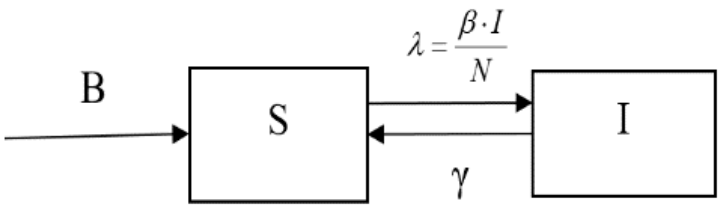


Рисунок 1 — *SIS* — модель

де, *B* — *народжуваність*, *N* — все населення, *S* — сприйнятливі особи, *I* — інфіковані особи, λ — сила інфекції, β — параметр передачі збудника, γ — швидкість одужання (втрата інфекційності).

Параметри моделі, як і вся система вцілому, можуть бути динамічними. Математично модель може бути описано за допомогою системи різницевих рівнянь (1):

$$\begin{cases} S_{t+1} = B_t + S_t - \dfrac{\beta_t \cdot I_t}{N_t} \cdot S_t + \gamma_t \cdot I_t \\ I_{t+1} = I_t + \dfrac{\beta_t \cdot I_t}{N_t} \cdot S_t - \gamma_t \cdot I_t \end{cases} \quad (1)$$

де, *t* — період часу.

Якщо в SIS — модель додати вакцинацію, то отримаємо *SISV* — модель (рисунок 2).

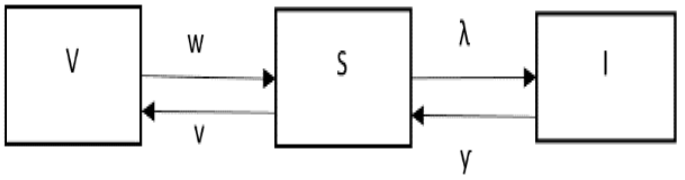


Рисунок 2 — *SISV* — модель

де, *V* – вакциновані особи, *v* – охоплення населення вакцинацією, *w* – спадання імунітету.

Математично модель може бути описано за допомогою системи рівнянь (2):

$$\begin{cases} S_{t+1} = S_t - v_t \cdot S_t + w_t \cdot V_t - \lambda_t \cdot S_t + \gamma_t \cdot I_t \\ I_{t+1} = I_t + \lambda_t \cdot S_t - \gamma_t \cdot I_t \\ V_{t+1} = V_t + v_t \cdot S_t - w_t \cdot V \end{cases} \quad (2)$$

Таким чином за допомогою *SISV* — моделі можна визначити оптимальну схему вакцинації населення для запобігання розповсюдження інфекційних захворювань.

Програмне забезпечення використовує епідеміологічні моделі, які дозволяють досліджувати процес розповсюдження інфекційних захворювань, складати прогноз на майбутнє, визначати ефективність вакцинопрофілактики, вибирати оптимальні схеми вакцинації з використанням епідеміологічних даних [1]. Завдяки використанню епідеміологічних моделей стає можливим своєчасне попередження епідемій інфекційних захворювань.

Система складається з трьох програмних модулів: модуль доступу до даних, модуль бізнес логіки та модуль представлення. Модуль доступу до даних відповідає за зчитування, обробку та

збереження вхідних епідеміологічних даних. Обробка вхідних епідеміологічних даних необхідна для усунення надлишкових даних та автоматизації процесу збору та розрахунку статистики.

Модуль бізнес логіки виконує побудову епідеміологічних моделей, складає прогноз на майбутнє. Модуль визначає оптимальні схеми вакцинації в залежності від вхідних епідеміологічних даних.

Модуль представлення використовує графічний інтерфейс користувача для виводу отриманих результатів роботи системи (рисунок 3).

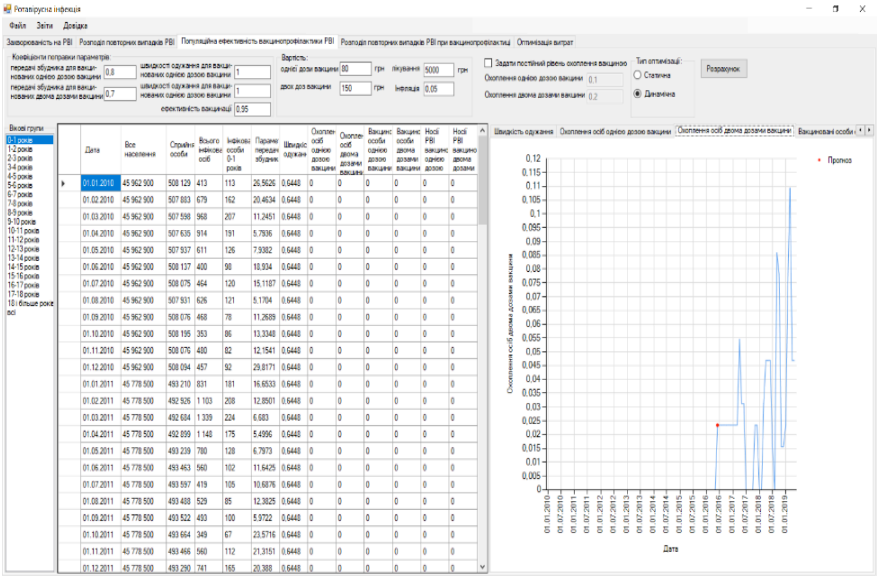


Рисунок 3 — *Головне вікно* розробленої системи

Розроблена система надає можливість визначити оптимальні індивідуальні та суспільні витрати на вакцинопрофілактику(рисунок 4).

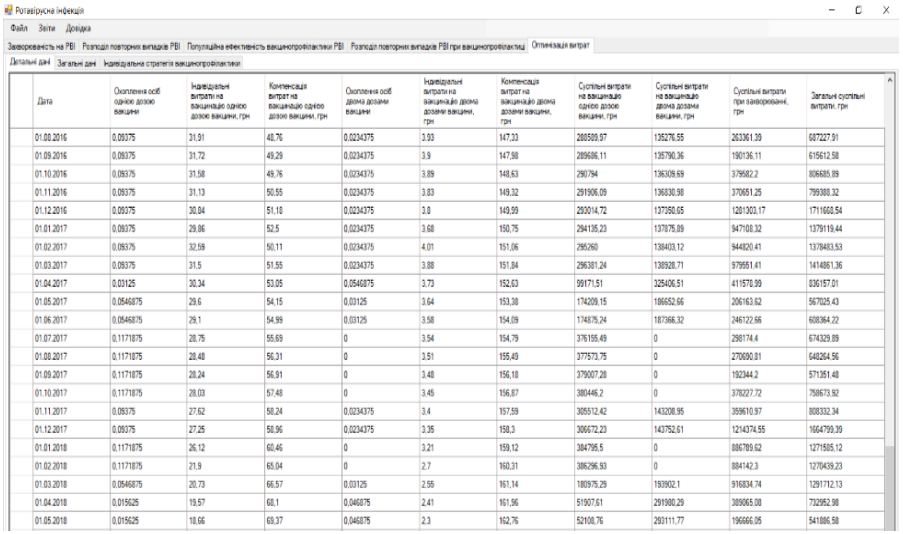


Рисунок 4 — *Оптимізація витрат*

UML діаграма прецедентів демонструє можливі варіанти дій доступних для користувачів системи (рисунок 5).

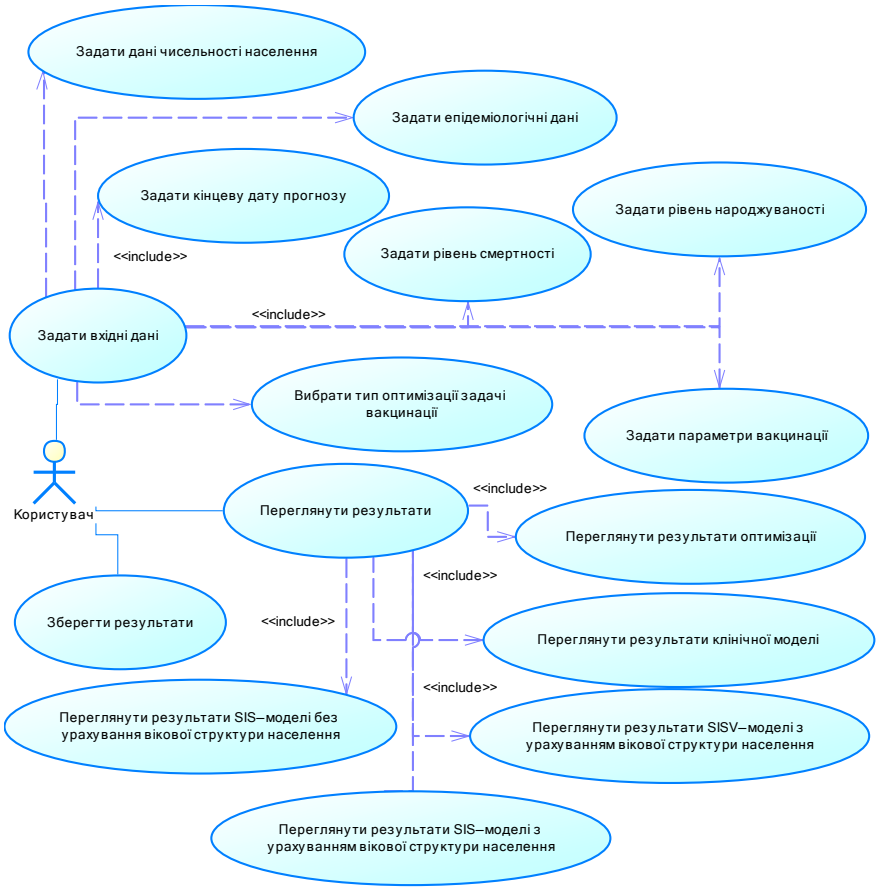


Рисунок 5 — *Діаграма* прецедентів розробленої системи

Розроблену систему можливо застосувати для дослідження PBI, складання прогнозу захворюваності, вибору оптимальних схем вакцинації, а також визначення витрат на лікування та вакцинопрофілактику.

Перелік використаних джерел:

1. Karaieva N.V. The assessment of environmental external cost as a tool of management of the Ukrainian energy system: review of approaches / N.V. Karaieva, O.I. Bandurka / Managing economic growth: marketing, management and innovations. 1st edition, Illiashenko, S.M., Strielkowski, W. (eds.). — Prague Institute for Qualification Enhancement: Prague, 2016. — P. 396-407.

Верлань А. А., Лалак Б. О.

Система конвертації баз даних різних типів

Бази даних та системи управління ними використовуються у дуже великій кількості сфер та різноманітних програмних забезпеченнях, самих же типів баз даних існує декілька видів: реляційні, об'єктно-орієнтовані, NoSQL, пост-реляційні бази даних, а систем управління ними існує досить немало на кожен з типів БД.

Очевидно, що маючи таку різноманіть, іноді з'являється необхідність перейти з однієї СУБД на іншу і у цей момент постає проблема з вирішення задачі конвертації даних так щоб зберегти всю необхідну інформацію та структуру.

Конвертація даних - перетворення даних з одного формату в інший. Зазвичай із збереженням основного логічно-структурного змісту інформації. Перетворення даних БД пов'язане з різницею логічних структур даних, а також з багатомодельністю представлення даних, різницею в логічних структурах та ін. [2]

Як правило, задача по конвертації баз даних з'являється при організації робіт по міграції з одного програмного забезпечення на інше, в результаті чого є необхідним перенести дані з однієї СУБД на іншу, але, частіше за все, затрати на перенесення бази даних на нову СУБД (з усіма таблицями, даними, процедурами, тощо) є дуже витратним у сенсі часу та фінансів [3].

Саме через це є сенс зробити систему конвертування баз даних між різними СУБД.

Конвертування БД, проходить у декілька етапів:

Етап 1: Резервне копіювання.

Етап 2: В залежності від того, чи необхідно відключати користувачів від БД на час перенесення його можна поділити на дві ланки:

1. Конвертування БД та перенесення інформації (якщо користувачі відключені).

2. Конвертування БД та перенесення інформації з back-up бази, а потім доконвертування тієї інформації що з'явилась при роботі користувачів під час перенесення.

Метою статті є виокремлення характеристик системи конвертації баз даних різних типів.

Системи конвертації баз даних

Наразі реалізована значна кількість систем конвертації баз даних. Далі наведені лише деякі з них.

Конвертер WhiteTown

Конвертер WhiteTown являє собою інструментарій для конвертації різних файлів баз даних між собою. Також наявні декілька пакетів інструментів, як конвертери між двома типами файлів (EXCEL to CSV), так і для сімейства операційних систем (Windows та Mac OS).

Розробники заявляють, що конвертери прості у використанні, але володіють широкими можливостями.

Набір конвертерів включає в себе:

з DBF у:

- CSV (дані, розділені комами)
- Excel (підтримка XLS та XLSX форматів)
- Access (підтримка MDB та ACCDB форматів)
- XML
- SQL скрипт, який можна застосувати на SQL-сервері
- SQLite (поки що, лише для Mac)
- У інші формати DBF (dBase III, FoxPro, Visual FoxPro, dBase IV)

з CSV у

- DBF
- Excel

- SQLite
- из Excel в
- DBF
- CSV
- SQlite

з Access (лише для Windows) в

- DBF
- Excel
- CSV

Конвертер Ispirer MnMTK

Ispirer MnMTK дозволяє конвертувати Oracle у Microsoft SQL Sever, включаючи схему бази даних (DDL), дані, бізнес-логіку на стороні серверу..

Ispirer MnMTK дозволяє конвертувати процедури і функції в Java і .NET.[22]

Характеристики програми Ispirer MnMTK для міграції з Microsoft SQL Server:

- швидка і зручна конвертація SQL-скриптів - ви можете легко почати конвертацію існуючих SQL-скриптів: операторів DDL, дамп бази даних SQL, SQL запитів тощо;
- конвертація з підключенням до бази даних - Ispirer MnMTK може під'єднатися до Oracle, щоб витягти і конвертувати як базу даних повністю, так і лише певні об'єкти або схеми;
- Ispirer MnMTK генерує SQL-скрипти в синтаксисі Microsoft SQL Server;
- підтримка командного рядка може допомогти автоматизувати весь процес міграції;
- тестування міграції - дана програма порівнює кількість таблиць, кількість рядків, виконує перевірку даних для всіх або

зазначених стовпців (заснованих на первинних ключах або на певній команді).

Характеристики конвертеру баз даних різних типів

Для створення системи конвертації баз даних різних типів, необхідно щоб система вирішувала наступні задачі з конвертації таких об’єктів БД:

- Таблиці та дані:
  - Конвертувати визначення стовпців – типи даних, значення за замовчуванням, обмеження NOT NULL, ідентичності,
  - Конвертувати обмеження цілісності – первинні та зовнішні ключі, обмеження unique та check,
  - Конвертувати коментарі,
  - Ліквідувати конфлікти зарезервованих слів та ідентифікаторів,
  - Переносити дані(включаючи специфічні типи даних, наприклад, CLOB та BLOB);
- Індекси, Представлення та Синоніми,
- Послідовності,
- Зв’язки з базою даних,
- Вбудовані пакети,
- Системні таблиці та представлення,
- Збережені процедури, Пакети, функції та тригери;[1]

На основі проведених дослідженнь було виявлено, що основні складнощі виникають при перенесенні збереженого коду, такого як тригери чи збережені процедури БД і якщо між такими СУБД як SQL Server та Oracle DB ці труднощі мінімальні, оскільки у відповідні комірці опису БД зберігається команда створення певної процедури і для перенесення достатньо лише змінити деякі ключові слова, то, наприклад, перенести збережену процедуру з SQL Server на MongoDB вже буде досить непроста задача, з точки зору автоматизації цього процесу.

Найскладнішим, у цьому випадку, блоком буде саме дія яку повинен виконати тригер при спрацюванні, оскільки це скрипт мовою SQL яких необхідно інтерпретувати в універсальний вигляд, який можна буде використовувати для створення ідентичних тригерів на інших СУБД. Наступним етапом буде створення екземпляру класу із записом у властивості відповідних блоків команди.

Таким чином, маючи структурований вигляд команди, маємо змогу описати такі алгоритми, що за допомогою даного елементу можна було створити ідентичні команди на різних СУБД.

Література:

1. Itzik Ben-Gan - Microsoft SQL Server 2012 T-SQL Fundamentals (Developer Reference)
2. Гарсиа-Молина Г., Ульман Дж., Уидом Дж. Системы баз данных. Полный курс = Database Systems: The Complete Book. — Вильямс, 2003. — 1088 с. — ISBN 5-84590384-X.
3. Миграция с Oracle на Microsoft SQL Server [Электронный ресурс]. -— Режим доступу: http://www.ispirer.ru/products/oracle-to-sql-server-migration

Гайдаржи В.І., Сук С. В.

Хмарна реалізація підсистеми ведення бібліотеки гідроакустичних моделей системи моделювання гідроакустичних процесів

Анотація. Вирішення задач спостереження за поведінкою рухомих морських об’єктів з метою їх класифікації та ідентифікації потребують створення систем моделювання призначених для вимагають створення адаптивних імітаційних моделей складних динамічних об'єктів і побудови планів проведення експериментів на їх основі з наданням веб- інтерфейсу для користувача

Сучасною технологією є хмарні сховища даних, які замінюють фізичні носії даних і мають значні переваги у порівнянні з ними. Якщо потрібно дати доступ до колективу розробників, у випадку з хмарним сховищем це можна зробити за лічені секунди, замість того щоб переписувати ці дані на інші носії чи відправляти по Інтернету. Саме тому дана програмна система для вирішення поставлених задач використовує саме ці концепції — вона є веб-додатком для роботи з бібліотеками і підсистемами комплексу моделювання гідроакустичних експериментів, що завантажуються з хмарного сховища даних Google Drive. Для написання програмного комплексу були використані сучасні технології ASP.NET та WPF.

Ключові слова: гідроакустика, система моделювання, експеримент, хмарна технологія, бібліотеко моделей, інформаційна модель

Постановка проблеми

Розглядаються проблема створення веб-інтерфейсу підсистем ведення бібліотек моделей і довідників системи управління

гідроакустичними експериментами. Веб інтерфейс має забезпечити наступні можливості:

— активізація модулів реалізації гідроакустичних моделей;
— ведення довідників системи;
— візуалізація документів опису моделей.

Оскільки програми, що запускаються, мають десктопну реалізацію, а веб-інтерфейс працює в браузері, це викликає певні складнощі в реалізації задачі. По-перше, стандартно браузер не має доступу до файлової системи комп'ютера і працює ізольовано- це зроблено з питань безпеки, щоб неможливо було отримати віддалений доступ до чужого комп'ютера через Інтернет. По-друге, веб-інтерфейс повинен працювати з будь-якого комп'ютера- інакше втрачається сенс у його веб-реалізації. Для цього потрібно мати пакет програм, що запускаються з нього, на жорсткому диску комп'ютера. Вирішенню цієї задач присвячена дана статуя.

Огляд підходів до побудови моделюючого комплексу.

Прототипом до розроблюваного програмного продукту є моделюючий комплекс для відпрацювання алгоритмів і програмного забезпечення виявлення, класифікації та визначення параметрів руху морських об'єктів.

Моделюючий комплекс є програмним застосунком з десктопною реалізацією, розробленим на мові C# для платформи .NET з використанням технологій Windows Forms та WCF. До складу моделюючого комплексу входять:

— підсистема моделювання морського середовища;
— підсистема моделювання гідроакустичної системи;
— підсистема аналізу гідроакустичних сигналів;
— підсистема розпізнавання морських об'єктів;
— підсистема ведення бази знань предметної області;
— підсистема «Машина виводу»;
— інформаційне сховище МК

— Засоби адміністрування та керування;

Кожна з вищезгаданих складових модулюючого комплексу має свій набір функцій, які перераховані нижче.

Основна частина

Технологічної Платформа розробки системи .NET Framework — програмна платформа від компанії Microsoft. Платформа .NET Framework складається з загальномовного середовища виконання (середовища CLR) і бібліотеки класів .NET Framework. [1]. Платформа .NET Framework не тільки надає кілька базових середовищ виконання, але також підтримує розробку базових середовищ виконання незалежними виробниками [2]. Бібліотека класів Середа CLR управляє пам'яттю, виконанням потоків, виконанням коду, перевіркою безпеки коду, компіляцією і іншими системними службами. Ці засоби є внутрішніми для керованого коду, який виконується в середовищі CLR. З міркувань безпеки керованим компонентам присвоюються різні ступені довіри, що залежать від ряду факторів, у число яких входить їх походження (наприклад, Інтернет, мережа підприємства або локальний комп'ютер). Це означає, що керований компонент може або не може виконувати операції доступу до файлів, операції доступу до реєстру або інші важливі функції, навіть якщо він використовується в одному активному застосунку [3].

Програмний комплекс розроблено мовою C# під платформу .NET Framework., та складаються з двох підсистем

— веб-застосунок реалізації основного функціоналу;
— підсистема налаштування комп'ютера користувача.

Веб-інтерфейс виконує функції обліку моделей в базі даних та запуску моделей на виконання. Для цього він зв'язується з базою даних та взаємодіє з нею за допомогою SQL-запитів, повертаючи користувачеві результат запиту у вигляді веб-сторінки.

Для активізації моделі користувач посилає запит веб-внтерфейсу, який активізує модель, запускаючи відповідну програму на комп'ютері користувача через зовнішній протокол.

Перед використанням веб-інтерфейсу користувач має налаштувати комп'ютер за допомогою підсистеми для налаштування комп'ютера користувача. Ця підсистема виконує налаштування комп'ютера користувача перед використанням веб-інтерфейсу, а саме — завантажує гідроакустичні моделі з Google-диску та вносить зміни в реєстр для реєстрації зовнішніх протоколів, за допомого яких завантажені програми та файли запускаються з веб-інтерфейсу.

Веб-застосунок являє собою веб-інтерфейс підсистем ведення моделей і довідників системи управління гідроакустичними експериментами, розроблений за допомогою технології ASP.NET MVC і реалізує дві функції: робота з базою даних та з пакетом програм та документів, які зберігаються на комп'ютері клієнта. Для роботи з базою даних використовується бібліотека WorkWithDb.dll. Для запуску програм та документів, що зберігаються на комп'ютері клієнта було використано зовнішній протокол пошукового завдання у вигляді діалогових вікон з полями для вводу характеристик.

Веб-застосунок складається з контролерів для роботи з гідроакустичними та інформаційними моделями, відповідних представлень та моделі, і використовує бібліотеку WorkWithDb.dll для роботи з базою даних (рисунок 1).

Опис контролерів. Головна сторінка запускається методом Index контролера HomeController. Для роботи з гідроакустичними моделями було створено контролер ModelManagementController, а для роботи з інформаційними моделями- ReferencesController.

В контролері ModelManagementController є два перевантаження методу Index для GET- і POST-запитів відповідно. Це регулюється за допомого атрибутів [HttpGet] і [HttpPost].

Метод для GET-запиту зчитує дані з відповідної таблиці за допомогою методів getHeaders() і getData() класу Select з бібліотеки WorkWithDb, що передаються представленню через модель Table.

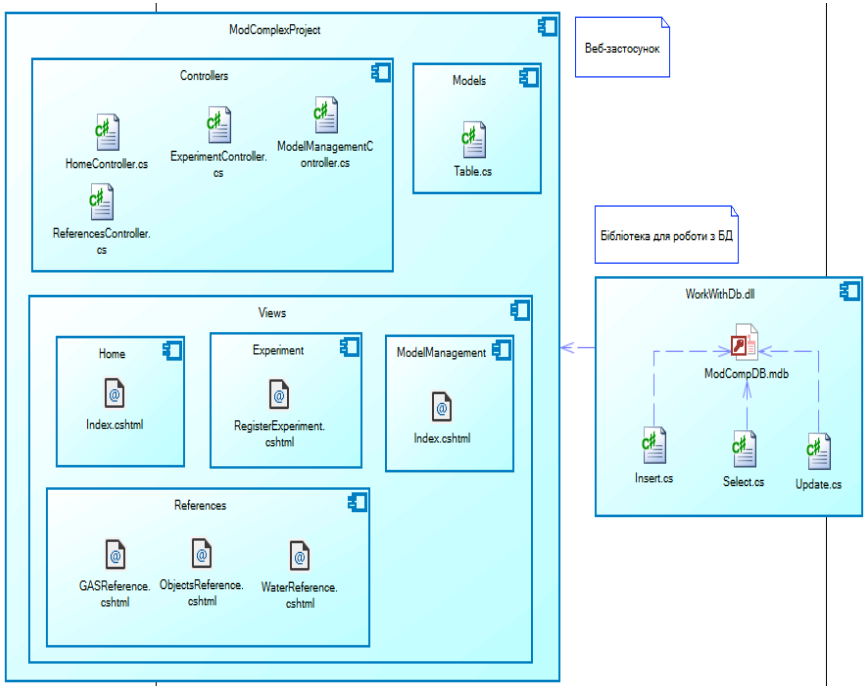


Рисунок 1 — Структура веб-застосунку

Метод для POST-запиту приймає один параметр — дані таблиці у JSON-форматі. Він десеріалізується в масив об'єктів-рядків. Елементи масиву перебираються і над кожним з них виконується перевірка на коректність параметрів.

Для роботи з інформаційними моделями служить контролер ReferencesController. Всього є три інформаційні моделі — акваторій, гідроакустичних систем і морських об'єктів. Для їх обробки існують відповідно методи WaterReference, GASReference, ObjectsReference. В кожному методі два перевантаження для обробки GET- і POST-запитів, і

працюють вони аналогічно методу Index попередньо розглянутого контролера.

Опис представлень. В концепції ASP.NET MVC представлення містяться в папці Views проекту. В шаблонному представленні @Layout.cshtml даного проекту міститься HTML-розмітка, де підключаються необхідні скрипти, а також часткове представлення Menu.cshtml. Воно містить спливаюче меню у шапці сайту. В меню наявні посилання на відповідні методи контролерів або посилання для запуску зовнішніх протоколів моделей. Для формування посилань використано HTML-хелпер @Html.ActionLink, який приймає три параметра — назва посилання, що відображається на сторінці, назва методу та назва контролера.

Також в папці Shared міститься типізоване часткове представлення Table.cshtml, що в якості моделі містить модель Table. Воно являє собою HTML-таблицю, в якій заголовки та вміст задані за допомогою колекцій і зберігаються у відповідних полях класу Table.cs з папки Models.

Представлення контролерів типізовані класом моделі таблиці бази даних, з яким працює його метод. Вони містять таблицю даних (часткове представлення Table.cshtml з папки Shared) та форму для відправки даних на сервер. Для полів таблиці використовуються строго типізовані HTML-хелпери. Ці хелпери приймають в параметрах лямбда-вираз, в якому вказана та властивість моделі, до якої має бути прив'язаний даний хелпер. Для id та інших полів, що не змінюються або змінюються не вручну, а в залежності від інших полів, але потребують відправки на сервер, представлені хелпером @Html.HiddenFor, що створює приховане поле в формі. Текстові поля створюються хелпером @Html.TextBoxFor, мітки- @Html.LabelFor. Кожен хелпер транслюється в HTML-код, а саме в input-елемент форми, в якому атрибут name- це назва властивості моделі.

Функції користувача :

Створити карту обліку моделі - додає новий рядок в кінець таблиці.

Фіксувати зміни в таблицю- фіксує зміни, що внесенні в полях форми, у відповідний ряжок таблиці.

Зберегти зміни в БД - змінені рядки записуються у масив та кодуються у форматі json, що передається на сервер POST-запитом за допомогою Ajax. Для роботи з Ajax використовується функціональність бібліотеки jQuery.

Перегляд опису моделі - відкривається документ MS Word, шлях до якого записаний у відповідному полі.

Активізація моделі - використовуються дані двох полів: «Шлях до файлу виконання» та «Шлях до файлу проекту». Якщо в якості шляху до файлу виконання записаний шлях до програми Matlab, то в полі «Шлях до файлу проекту» має бути записаний шлях до файлу з розширенням «.m». Цей шлях передається в якості параметра при запуску Matlab.

Якщо в полі «Шлях до файлу виконання» міститься шлях до іншого exe-файлу, а поле «Шлях до файлу проекту» порожнє, то запускається програма за зовнішнім протоколом, що відповідає цьому exe-файлу.

Шляхи до цих файлів містяться в базі даних.

Опис моделі. В даному застосунку в якості моделі створений клас Table.cs, в якому містяться наступні властивості:

— public IList<string> headers { get; set; } — відповідає заголовкам

таблиці;

— public IEnumerable<IList<string>> data { get; set; } — що відповідає рядкам даних таблиці;

— public int[] skipIndexes { get; set; } — містить номери полів HTML-таблиці, які непотрібно відображати на сторінці, наприклад Id.

В класі є конструктор, де за замовчуванням ініціалізується поле skipIndexes єдиним елементом «0», оскільки в усіх таблицях поле Id має нульовий індекс. Але якщо потрібно, skipIndexes можна

ініціалізувати масивом з більшою кількістю індексів або взагалі передати туди пустий масив, якщо потрібно відображати всі рядки.

Опис бібліотеки WorkWithDb.

Бібліотека WorkWithDb розроблена для роботи з базою даних. Вона використовує технологію ADO.NET і взаємодіє з базою даних через модель бази даних, створену засобами Visual Studio. Бібліотека містить модель бази даних, що являє собою сукупність класів, що відповідають таблицям бази даних, з полями, що за типами даних і назвами відповідають полям таблиць бази даних, класи Select, Insert та Update, які реалізують відповідні методи для доступу до бази даних.

Для підключення до бази даних використовується клас ConnectionInfo, який має рядок internal static string connectionString { get; }, що зберігає рядок подключення до бази даних. Клас має статичний конструктор, де при першому запуску програми поле connectionString заповнюється потрібними значеннями і залишається незмінним впродовж роботи програми.

Для забезпечення роботи веб-додатку необхідно, щоб комп'ютер користувача був спеціальним чином налаштований для цього, а саме:

— на жорсткому диску повинні зберігатися програми і документи, що запускаються з браузера;

— для можливості запуску з браузера повинні бути внесені відповідні зміни в реєстр — створені протоколи обробки подій запуску та записані відповідні шляхи до потрібних файлів. З питань безпеки веб-браузер не має традиційних засобів доступу до файлової системи клієнта. З цим пов'язані деякі складності в реалізації задач даної програмної системи. Щоб дати користувачу веб-додатку доступ до програм і документів, що зберігаються на його комп'ютері, було налаштовано обробник зовнішнього протоколу. Для цього були внесені певні зміни в реєстр Windows/ Всі дані в реєстрі зберігаються у форматі "ключ-значення". Щоб зареєструвати обробник зовнішнього протоколу для певного URL в реєстрі, потрібно додати ключ в HKEY_CLASSES_ROOT, який буде відповідати певній URL [4]:

Завантаження файлів відбувається з хмарного сховища Google Drive. Для цього використовується бібліотека Google Drive API, яка дозволяє взаємодіяти з Google Drive засобами .NET.

Доступ до сервісів Google зі сторонніх програм здійснюється через проекти Google API. Щоб програма завантаження могла отримати доступ до Google Drive, спочатку потрібно створити Google API Project. Для цього потрібно увійти до облікового запису Google і перейти в Google API Console за адресою https://console.developers.google.com і вибрати в меню «Створення проекту» Google Developers Console — це спеціальний інструмент для розробників, що дозволяє писати програми з доступом до сервісів Google і їх використанням.

Висновки.

В даній статті наведено опис програмне забезпечення, що реалізує Web-інтерфейс підсистем ведення бібліотек моделей і довідників системи управління гідроакустичними експериментами.

Програмне забезпечення складається з двох підсистем:

— програмна реалізація веб-інтерфейсу для обліку та активації гідроакустичних моделей;

— підсистема для налаштування комп'ютера користувача перед використанням веб-інтерфейсу, а саме завантаження гідроакустичних моделей та файлів опису моделей з Google-диску та внесення змін в реєстр для створення зовнішніх протоколів.

Література

1. Общие сведения о платформе .NET Framework [Електронний ресурс]. — Режим доступу: https://msdn.microsoft.com/ru-ru/library/aa767914(v=vs.85).aspx
2. Рихтер Джеффри CLR via C#. Программирование на платформе Microsoft.NET Framework 4.5 на языке C#; Питер - М., 2016. - 36 с.
3. Jeffrey Richter CLR via C#; - Москва, 2012. - 80 с.

4. Registering an Application to a URI Scheme [Електронний ресурс]. — Режим доступу: https://msdn.microsoft.com/enus/library/windows/desktop/ms724871(v=vs.85).aspx

Гайдаржи В.І., Чистякова Д.Ю.

Загальна структура системи моделювання гідроакустичних процесів та керуючій модуль системи

Анотація. Вирішення задач спостереження за поведінкою рухомих морських об’єктів з метою їх класифікації та ідентифікації потребують створення систем моделювання призначених для вимагають створення адаптивних імітаційних моделей складних динамічних об'єктів і побудови планів проведення експериментів на їх основі. Основним атрибутом таких систем є здатність накопичення і використання знань (формалізованих посилок, висновків, закономірностей, що відносяться до досліджуваного об'єкта) в процесі моделювання поведінки складних динамічних об'єктів.

Розробка та використання складних багатофункціональних моделюючих систем породжує проблему керування функціональними підсистемами. З іншого боку, користувач такої системи зацікавлений в наявності простого, інтуїтивно зрозумілого інтерфейсу взаємодії з системою. Отже постає проблема розробки керуючого модуля системи моделювання гідроакустичних процесів. Опису загальної структури моделюючого комплексу та її керуючого модуля присвячена дана стаття

Ключові слова: гідроакустика, система моделювання, експеримент, керуючий модуль, інформаційна модель

Постановка проблеми

Комп'ютерне моделювання як інструмент дослідження має цілий ряд переваг в порівнянні з реальним експериментом, зокрема, комп'ютерний експеримент може бути виконаний в таких умовах,

коли проведення натурного експерименту ускладнено або навіть неможливо [1].

Система моделювання гідроакустичних процесів надає можливість фіксації та аналізу експериментальних досліджень гідроакустичних процесів пов'язаних з вивченням особливостей шумовипромінювання морських об'єктів кінцевою метою яких є класифікація та ідентифікація цих об'єктів на основі використання ряду відповідних алгоритмів. В зв’язку з цим постала проблема вивчення та формування оцінок різних моделей гідроакустичних процесів з позиції адекватності відображення реальних фізичних процесів поширення акустичних сигналів, а також алгоритмів їх розпізнавання і на цій основі проведення класифікації та ідентифікації випромінюючих об'єктів [2].

Для автоматизації процесу моделювання гідроакустичних процесів створюються інформаційні системи, які, як правило, є багатофункціональними та складаються з кількох підсистем, таким чином ставиться задача розробки програмного комплексу покликаного вирішувати основні задачі моделювання, в тому числі:

— моделювання процесів поширення акустичних сигналів в гідросередовищі;

— моделювання процесів виділення сигналів, що надійшли від різних рухомих шумовипромінюючих об'єктів;

— моделювання процесів класифікації та ідентифікації об'єктів — джерел випромінювання.

Крім того, комплекс повинен вирішувати завдання організації дослідного процесу, що полягає в проведенні ряду експериментів, включаючи такі завдання як облік експериментів, організацію сцен, формування сценаріїв, оцінку апробованих в системі моделей.

Огляд підходів до побудови моделюючого комплекса

Actran — програмний комплекс для аналізу акустики, якій надає можливості з моделювання виникнення, поширення і поглинання звуку в акустичних. Повністю пов'язана віброакустична

постановка дозволяє врахувати взаємний вплив акустичної середовища і конструкції. У випадках, коли впливом середовища на конструкцію можна знехтувати, можливий облік однобічного зв'язку; при цьому спрощується моделювання і підвищується швидкість розрахунку[3]. В основі системи Actran лежать методи кінцевих і нескінченних елементів, що дозволяють моделювати акустичні ефекти як в замкнутих обсягах (порожнинах), так і у відкритому просторі з урахуванням умов невідображення звуку на кордонах розрахункової області. Багата бібліотека елементів і матеріалів надає широкі можливості для моделювання.

Глобальна система моніторингу морської обстановки

Лабораторією ООГІС СПІІРАН (Науково-дослідна лабораторія об'єктно-орієнтованих геоінформаційних систем Санкт-Петербурзького інституту інформатики і автоматизації Російської академії), створена і функціонує глобальна система моніторингу морської обстановки, яка отримує і обробляє інформацію від різних джерел інформації (РЛС, приймачі АІС, безпілотні літальні апарати, автоматичні гідрометеорологічні станції, міжнародні системи моніторингу середовища і світового океану), використовуваних відповідними регіональними системами моніторингу обстановки[4].

Специфічними особливостями глобальної системи моніторингу морської обстановки є:

— об'єднання в своєму складі великого числа інформаційних ланок і підсистем (інформаційні центри різного рівня управління, різноманітні джерела інформації, розвинені інформаційно-телекомунікаційні системи);

— обробка великих обсягів різнотипної інформації з безлічі різних джерел;

— використання різних одиниць вимірювання для параметрів середовища і параметрів об'єктів із заданою дискретністю зміни їх значень;

— реакція системи і її окремих інформаційних ланок

(інформаційні центри) на надходять події в реальному режимі часу.

Наведені системи не в повною мірою дозволяють вирішувати задачі керування системою гідроакустичних експериментів, метою верифікації алгоритмів та методів моделювання тому розроблено нову систему моделювання гідроакустичних процесів на основі реєстрації та оцінки гідроакустичних експериментів для реалізації саме таких задач

Основна частина

Загальна структура програмного комплексу

У таблиці(Таблиця 1) наведено опис програмних модулів що реалізують задачі основних функціональних підсистем моделюючого комплексу.

Таблиця 1 — Опис програмних модулів функціональних підсистем с

Назва модуля	Опис підсистеми
Реєстрація експерименту	Створення та редагування нового гідроакустичного експерименту
Реєстрація сцени	Створення та редагування нової сцени проведення експерименту
Створення сцени	Створення сцени для проведення експерименту з використанням геоінформаційної системи
Створення сценарію	Створення та редагування нового сценарію виконання експерименту на певній сцені
Перегляд результатів	Зручний перегляд результатів проведення експерименту
Генерація сигналу	Генерація акустичного сигналу
Визначення сигналу	Визначення отриманого акустичного сигналу
Класифікація об’єкта	Класифікація об’єкта за акустичним

	сигналом
Ідентифікація об’єкта	Ідентифікація об’єкта за акустичним сигналом
Інші моделі	Ведення бібліотеки моделей з можливостями реєстрації, модифікації, видалення та активізації моделей
Керування моделями	Створення на редагування нової моделі
Довідник акваторії	Ведення інформаційної моделі акваторій
Довідник ГАС	Ведення інформаційної моделі ГАС з можливостями реєстрації, модифікації, видалення та активізації ГАС
Довідник об’єкти	Ведення інформаційної моделі морських об'єктів, умов спостереження для фіксації гідроакустичних портретів, акустичних портретів об'єкта

На рисунку 1 наведена загальна структура комплексу моделювання гідроакустичних процесів.

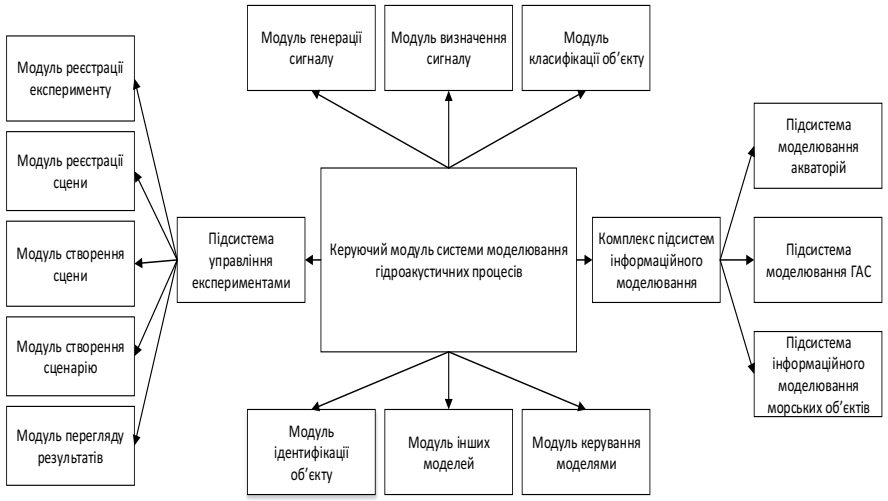


Рисунок 1 — Структура системи

База даних с системи моделювання складається з таблиць:

Experiment — експеримент, містить інформацію про експерименти

— GAS_Scene — сцена ГАС, пов'язує таблиці Сцена та Довідників ГАС

— Step_model — крок сценарію, містить крок для сценарію проведення експеримента

— Model — модель, містить інформацію про моделі

— Object_Scene — об'єкт сцени експерименту

— Sprav_GAS — довідник ГАС, містить інформацію про ГАС

— Sprav_Acva — довідник акваторій, містить інформацію про акваторії

— MarineObjects — "Морські об'єкти", містить технічні

характеристики суден

— MarineObjectsPurposes — призначення морських об'єктів

— Categories — категорії морських об'єктів (вантажне, пасажирське)

— Conditions — "Умови" містить параметри середовища моделювання

— Acoustic portraits — "Акустичні портрети" містить характеристики акустичного сигналу

Інформаційна модель морського об'єкту

Формування гідроакустичних портретів — одна з основних задач системи моделювання гідроакустичних процесів. Зафіксований гідроакустичний портрет є основою вирішення задачі ідентифікації, яка вирішується шляхом порівняння виділеного сигналу морського об'єкту з множиною гідроакустичних портретів.

Кінцева мета підсистеми інформаційного моделювання морських об'єктів — отримати базу гідроакустичних портретів відомих морських об'єктів в заданих умовах. підсистема інформаційного моделювання морських об'єктів надає функції з формування та актуалізації

— параметрів морських об'єктів

— умов спостереження для фіксації гідроакустичних портретів

На рисунку 2 наведено класифікацію морських об'єктів. За допомогою даної класифікації спроектовано базу даних [6][7].

Інформація про параметри морських об’єктів — надводного та підводного, яка використовувалася при моделюванні бази даних та користувацької форми фіксується відповідній карті реєстрації

Карта реєстрації надводного об’єкт містить наступні параметри:

Призначення, Тип, Проект, Назва, Основні ТТХ(Габарити (довжина, ширина, осадка), Водотоннажність(стандартна, повна), Мах швидкість хода)

Карта реєстрації підводного об’єкт містить наступні параметри:

Призначення, Тип, Серія (назва),Основні ТТХ(Габарити(довжина, ширина, осадка), Водотоннажність(надводна, підводна), Мах швидкість хода(надводна, підводна), Глибина занурення(оптимальна, максимальна)

. Дані характеристики перенесено до програми та до інтерфейсу користувача, також додано характеристики, яких не вистачало. Наприклад, потужність двигуна та кількість гвинтів.

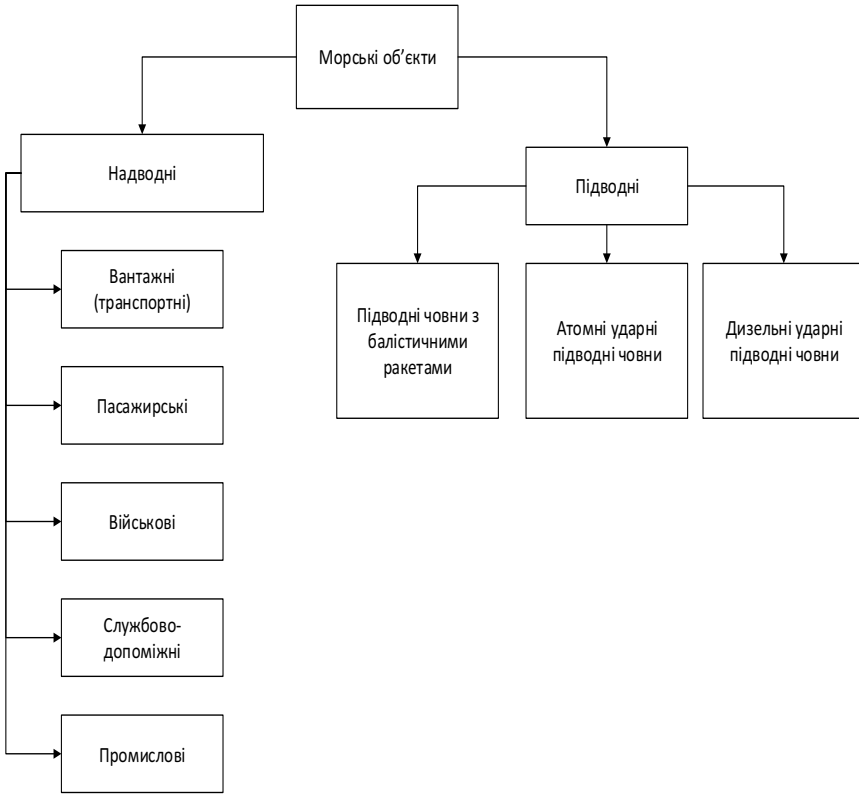


Рисунок 2— Схема класифікації морських об’єктів

Висновки.

В даній статті:

1.Проведено аналіз існуючих систем моделювання акустичних процесів та виявлено їх недоліки.

2. Розроблено алгоритм роботи керуючого модуля та

спроектовано базу даних системи.

3. Розроблено керуючий модуль системи моделювання гідроакустичних процесів, який відповідає вимогам:

— управління роботою моделюючого комплексу;
— формування та ведення бази даних і довідників;
— підтримка функції локалізації;
— надання користувачу інтуїтивно зрозумілого інтерфейсу.

4. Розроблено інформаційну модель морського об'єкту та базу даних для підсистеми інформаційного моделювання морських об’єктів [29].

5. Розроблено підсистему інформаційного моделювання морських об’єктів. Підсистема надає функції з формування та актуалізації:

— параметрів морських об'єктів;
— умов спостереження для фіксації гідроакустичних портретів;
— акустичних портретів об'єкта.

Література

1. Тупицына А. И. Методы компьютерного моделирования физических процессов и сложных систем. Учебное пособие— СПб: Университет ИТМО, 2014. — 48 с.

2. Евтютов А. П. Справочник по гидроакустике / А. П. Евтютов, А. Е. Колесников. — Судостроение, 1982. — 340 с.

3. Actran [Електронний ресурс]. — Режим доступу: http://www.mscsoftware.ru/products/actran

4. Интеллектуальные геоинформационные системы в системах мониторинга обстановки [Електронний ресурс]. — Режим доступу: http://docplayer.ru/36099617-Intellektualnye-geoinformacionnye-sistemy-v-sistemah-monitoringa-obstanovki.html

5. Шилдт Г. C# 4.0: полное руководство.: Пер. с англ. — М.: ООО "И.Д. Вильямс", 2011. — 1056 с.

6. Слижевский Н. Б. Энциклопедия судов / Под общ. ред. проф. Н.Б. Слижевского. — Николаев: НУК, 2005. — 172 с.

7. Современные подводные лодки. Самые смертоносные системы морских вооружений мира: Иллюстрированная энциклопедия / Пер.с англ./Шант К. - М.: «Омега», 2009. - 192 с.

Дацюк О.А., Амброс С.М.

Розробка мобільного додатку моніторингу споживання енергії

При сучасному розвитку побутової електроніки люди стараються забезпечити собі комфортне існування у житлових приміщеннях. Та сьогодні, в час активних підвищень тарифів на комунальні послуги актуальним є вирішення задачі раціонального використання електроенергії та збереження бюджетів родини. Намагаючись зекономити, люди встановлюють дво- та тризонні лічильники, відповідно до яких ціна на електроенергію за нічними тарифи вдвічі менша ніж за денним. Оскільки в активному житті сучасних громадян не завжди є час відслідковувати та підраховувати вартість витраченої електроенергії, аналізувати як енерговикористання можна було б оптимізувати, актуальною є задача створення системи аналізування та моніторингу споживання електричної енергії. Користувачем системи може бути власник системи «Розумний дім».

Основа програми моніторингу споживання енергії в будинку полягає в можливості комунікації домашніх з сервером, де проводиться збір та обробка інформації по споживанню енергії різними пристроями та аналіз споживання енергії вцілому, підраховуватиме загальну вартість використаної електроенергії

відповідно до тарифів, а також даватиме поради щодо можливості більш раціонального використання енергії.

На теперішній момент існують системи для мобільної платформи Android схожі до даної. Наприклад, програма Electrical Cost [1] підраховує енергоспоживання приладів і виводить ціну на електроенергію за різний термін(день, місяць, рік). Програма EvoEnergy - Electricity Cost Calculator Free запитує в користувача, яку потужність має прилад, скільки він пропрацював і відповідно да даних враховує вартість спожитої електроенергії.

Споживання електроенергії будинком піддається коливанням, які залежать від того, скільки людей у ньому мешкає, як часто вони перебувають вдома і якими електричними пристроями користуються.

Аналогічно сучасним системам виду «розумний дім» прилади можуть бути під’єднані до комп’ютерної мережі, що дозволяє керувати ними за допомогою ПК (рисунок 1). Вхідними даними в програмі є дані щодо енергії приладів, а саме скільки часу вони працювали, в який саме час та скільки кВат енергії було спожито. Вихідними даними є графіки енергоспоживання, підрахована сумарна енергія за день, вартість спожитої електроенергії, дані за попередній день споживання енергії, а також повідомлення про прилади та коли їх можна було б увімкнути для більш раціонального використання електроенергії з точки зору ціни.

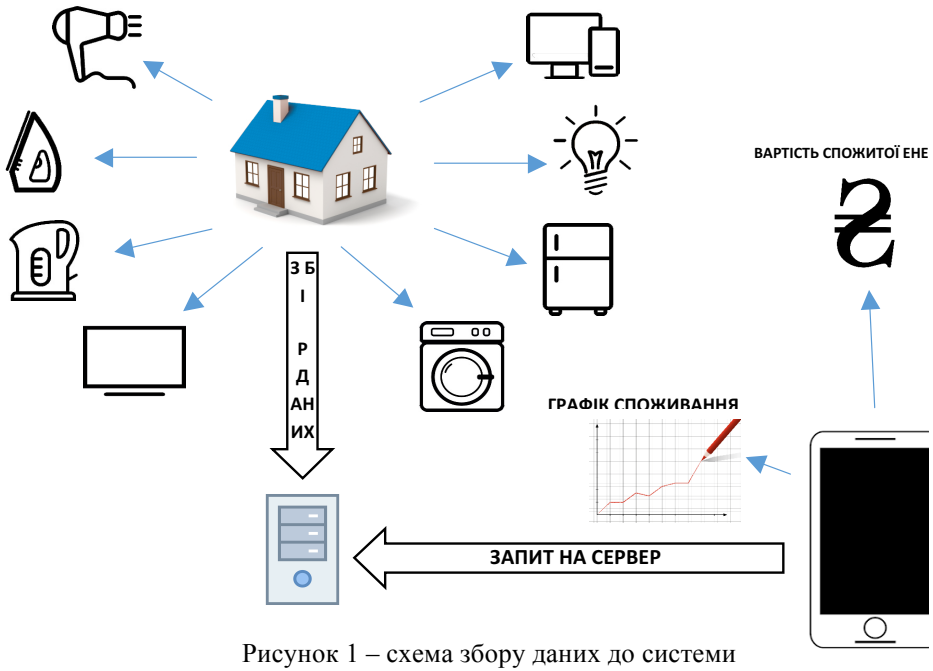


Рисунок 1 – схема збору даних до системи

В розробленій програмі, в порівнянні з іншими системами існує можливість вибору конкретного приладу (рисунок 2) та побудови графіку енергоспоживання відповідно до даного приладу, можливість побудови графіків за декількома чи всіма приладами, що є в будинку, а також можливість аналізування та підрахунків вартості спожитої електроенергії відповідно до двозонного лічильника, який стає більш розповсюдженим в нашій країні, порівнювати дані за декілька днів (рисунок 2), та ін.

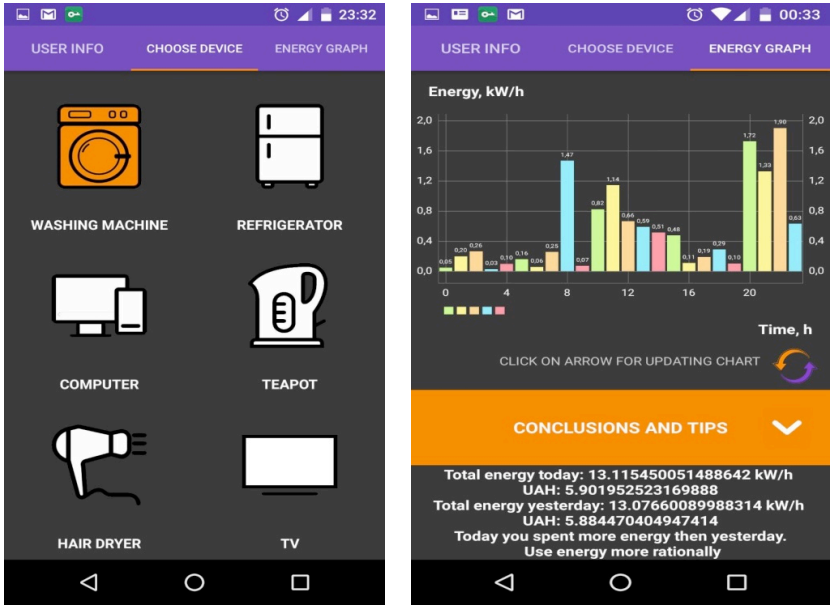


Рисунок 2 – Приклади роботи мобільного додатку

В ході система роботи було вирішено наступні задачі:

– підключення до приладів та отримання інформації щодо їх використання, а саме час роботи та енергія, яку вони споживають;

– визначення енергоспоживання кожного приладу за годину;

– підрахунок енергоспоживання за день відповідно до різноманітних приладів в будинку та час їх роботи;

– вирахування ціни за спожиту електроенергію в відповідності до двозонного лічильника [3];

– побудова графіків споживання електроенергії за конкретним приладом та за усіма приладами в будинку;

– аналіз споживання електроенергії, підрахунок загального часу та годинної зони в яку використовувався прилад;

– підрахунок вартості спожитої електроенергії та порівняння її з попередніми даними.

Таким чином, розроблена система дозволить користувачу відслідкувати та оптимізувати енергоспоживання його будинку.

Ведення статистики споживання всіх ресурсів будинку дозволяє проаналізувати та спланувати більш ефективне використання електроенергії.

Можна також подивитися споживання електроенергії окремими приладами або кожним електроприладом окремо. Система виводить не просто «сухі» цифри, вона показує статистику споживання у таблицях та діаграмах. Це дає можливість ефективно використовувати всі ресурси будинку і економити кошти його мешканців

. Перелік використаних джерел

1. Changing Utility Cost Pathways amid Rising Deployment of Distributed Energy Resources [Електронний ресурс] // EPRI Customer Assistance Center (CAC). – 2016. – Режим доступу до ресурсу: https://www.epri.com/#/pages/product/3002008211/

2. Evo Energy - Electricity Consumption Cost Calculator Product description [Електронний ресурс] // – Режим доступу до ресурсу: http://www.mycafecup.com/evowallet/

Нові тарифи на «світло». Скільки доведеться платити за електроенергію з 1 березня Главком. – 2018. – Режим доступу до ресурсу: https://glavcom.ua/publications/novi-tarifi-na-svitlo-skilki-dovedetsya-platiti-za-elektroenergiyu-z-1-veresnya-370083.html.

Дацюк О.А., Березюк В.Д.

Розробка додаткового інструментального засобу редагування онтології

Онтологія – опис понять, об’єктів та відношень між ними в рамках певної предметної області. На практиці часто застосовують предметно-орієнтовані онтології, які використовують знання з певної предметної області. В теоретичних задачах використовують загальні онтології, які передбачають подання абстрактних понять, спільних для набору предметних областей. Такі онтології містять базовий набір термінів, глосарій або тезаурус, що використовується для опису термінів предметних областей.

Все частіше з’являються задачі, які вимагають об’єднання вже існуючих розроблених онтологій та їх даних. Для інженера з онтологій це серйозне завдання. Подібні онтології часто несумісні одна з одною, хоча можуть представляти близькі галузі знань. Різниця виникає через використання синонімів в назвах об’єктів однакових предметних областей або просто внаслідок використання іншої мови опису.

На даний момент об'єднання онтологій, а також їх редагування доводиться виконувати вручну, це трудомісткий, повільний і дорогий процес. Існують наукові праці з різними підходами до технологій об'єднання, але вони в більшій мірі теоретичні і не мають широкого ринкового застосування. До таких робіт можна віднести [1, 2].

Громіздкою та незручною є також робота по наповненню онтології даними. Така задача як мінімум вимагає знань з роботи з редактором онтологій. А для створення якісної та цілісної бази знань

користувач повинен розуміти конкретну предметну область та інженерію онтологій.

Редагування великої за обсягом онтології вимагає від оператора серйозної підготовки та виконання значної кількості кроків під час створення нових елементів онтології. Ці етапи вимагають від користувача постійної концентрації та уваги. Це може призвести до появи логічних, структурних, лексичних та/або синтаксичних помилок в онтології. Крім того програмні редактори створення онтологій не мають української локалізації, що вимагає від оператора навичок володіння іноземною (англійською) мовою.

Тому розробка додаткового інструментального засобу редагування онтології дозволить спростити заповнення онтології даними та може полегшити процес об’єднання онтологій.

Необхідними можливостями, які має забезпечувати модуль, є:

- завантаження файлу онтології з комп’ютера користувача;
- створення контексту онтології;
- запис контексту онтології у файл онтології;
- можливість перегляду існуючої онтології;
- можливість роботи зі створеною онтологією стандартними засобами;
- можливість додавати до онтології основні елементи (класи, об’єкти, властивості, зв’язки);
- можливість задати установки елементам онтології (родинний зв’язок, схожість, тип);
- можливість перекладу інтерфейсу українською мовою.

Для створення онтологій використовується мови опису онтологій. Мова опису онтологій – формальна мова, що використовується для кодування онтології. Для роботи з мовами опису онтологій існують редактори онтологій, які дозволяють

створювати та змінювати структуру онтології, наповнювати її даними та можуть мати графічний редактор для відображення онтологій.

Protege – відкритий редактор онтологій. Онтології, побудовані в Protege, можуть бути експортовані у інші формати, включаючи RDF, OWL і XML.

В основі мови OWL лежить модель даних «об'єкт – властивість». Кожному елементу опису в цій мові (в тому числі властивостями, що зв'язує об'єкти) ставиться у відповідність URI [3].

Онтології зазвичай містять такі елементи: класи (поняття), екземпляри цих класів, їхні атрибути (властивості) та значення цих властивостей, а також відношення між класами та екземплярами класів. Крім того, онтологія може містити певні обмеження на використання класів та їх відношень [4]. Зазвичай відношенням є атрибут, значенням якого є інший об'єкт.

Розроблений додаток надає користувачу можливість завантажити файл онтології з комп'ютера користувача, записати онтологію у файл, переглядати існуючу онтологію, відкрити онтологію за допомогою Protégé, додати до онтології основні елементи (класи, об'єкти, властивості, зв'язки), задати установки елементам онтології (родинний зв'язок, схожість, тип), перекласти інтерфейс українською мовою.

Програмний додаток розроблявся з використанням мови програмування C# .NET Framework 4.0 та бібліотеки dotNetRDF, для створення контексту онтології. Програмний засіб складається з наступних компонент (рисунок 1):

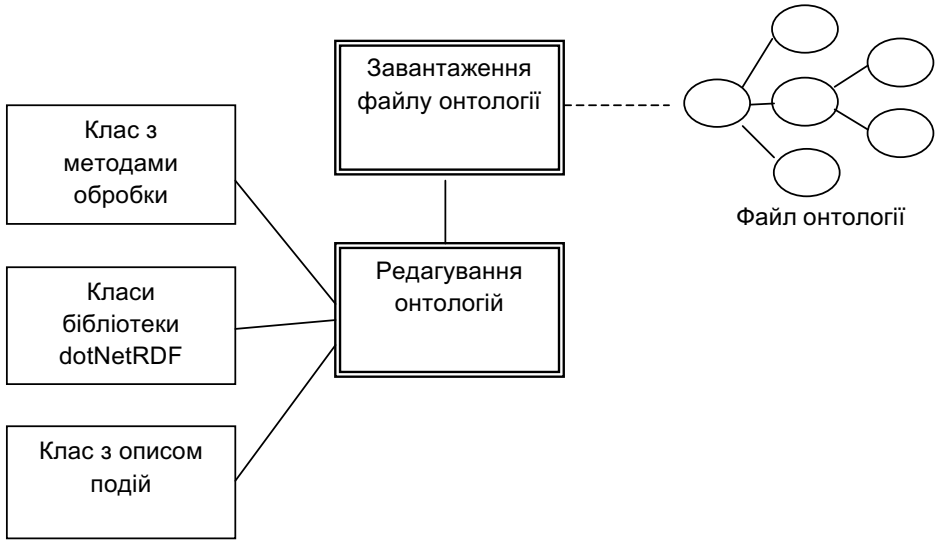


Рисунок 1 – Структура програмного засобу

У файлі головної форми описані методи завантаження та збереження файлу онтології. Потім створюється об'єкт класу OntologyGraph – це і є контекст, граф, з яким і потрібно надалі працювати ресурсами бібліотеки dotNetRDF. Далі проводиться зчитування файлу за вказаним шляхом та запис у контекст.

Так як метою створення програми було спрощення роботи з даними онтології, розроблений програмний продукт має основні функції Protégé, але вимагає від оператора меншої кількості етапів під час введення даних до онтології (рисунок 2).

а)

б)

Рисунок 2 – Порівняння етапів створення нового зв'язку: а) в системі Protégé, б) в розробленому додатку

Отже, розроблена програма надає можливість швидкого доступу до даних та структури онтології. Більш зручна робота можлива завдяки наявності української локалізації, іншому розташуванню полів, для введення даних, використанню інших UI рішень, для візуалізації елементів онтології.

В рамках роботи програми запропонований варіант сценарної мови для редагування онтологій – команда додавання нового класу до онтології.

Розробка сценарної мови дозволила додатково продумати можливість об'єднання онтологій в «ручному режимі» за рахунок редагування структури онтології.

Використання мови програмування C# забезпечило можливість створити, за допомогою бібліотеки dotNetRDF, об'єктно-орієнтовний контекст файлу онтології та пов'язати UI додаток WinForm з файлом онтології. Тобто на базі створеного додатку можливо подальше використання технологій програмування платформи .NET Framefork 4.0

Перелік використаних джерел

3. О.С. Вороной. Засоби інтеграції онтологій предметних областей для створення баз знань інтелектуальних навчальних систем / О.С. Вороной, Г.А. Єгошина / «Искусственный интеллект» 2'2010. – С.124-130,

4. Вороной А.С. Використання онтологій для підвищення якості пошуку інформації для поповнення баз знань інтелектуальних систем / А.С. Вороной // Матеріали Міжнародної науково-технічної конференції «Комп'ютерні науки і інженерія 2009»). – Львів, 2009. – С.364-366.

5. OWL 2 Web Ontology Language RDF-Based Semantics (Second Edition) [Електронний ресурс] – Режим доступу: https://www.w3.org/TR/owl2-rdf-based-semantics/

6. What is an Ontology? [Електронний ресурс] – Режим доступу: http://www-ksl.stanford.edu/kst/what-is-an-ontology.html.

Дровозюк В.О., Корнійкук М.А.

Програмні засоби тестування й зв'язку з наносупутником

Вступ

Запуск наносупутників університетами став звичною річчю в останнє десятиліття, університети з усього світу намагаються приєднатися до вивчення безмежного космосу. Наносупутники найчастіше використовуються в навчальних цілях та для тестування новітніх технологій, проте з кожним роком їм знаходять дедалі більше застосувань. Проте програмного забезпечення для роботи з наносупутниками не так багато, особливо програмних засобів для тестування й зв'язку з ними.

Постановка задачі

В задачу цієї роботи входить розгляд існуючих програмних засобів передачі даних по протоколу Modbus із їх подальшим об’єктним відображенням та виявлення аномальних значень в даних діагностики систем наносупутників.

Алгоритм отримання даних із супутника та їх декодування й подальше відображення

Провівши аналіз існуючих програмних засобів для роботи з протоколом Modbus, був розширений функціонал бібліотеки Jamod, а

саме запропоновано спосіб декодування й подальшого відображення отриманих даних та удосконалено сам процес передачі даних. Алгоритм об'єднує в одну транзакцію запити даних з регістрів, які знаходяться близько один від одного, і навпаки, посилає окремими транзакціями запити даних до регістрів які знаходяться далеко один від одного [1]. Це значно мінімізує витрати часу на виконання всіх операцій, в підтвердження чого приведені розрахунки нижче.

Перший не удосконалений варіант алгоритму (надалі «варіант 1»), який може бути використаний, кожний раз вичитує з пристрою значення абсолютно з усіх регістрів в проміжку від одного необхідного регістру до наступного необхідного (включаючи дані з «зайвих» регістрів). Другий не удосконалений варіант алгоритму (надалі «варіант 2») в кожну транзакцію включає інформацію про лише один числовий параметр, який може займати декілька регістрів [1]. Розроблений алгоритм об'єднує в одну транзакцію запити даних з регістрів, які знаходяться близько один від одного, і навпаки, посилає окремими транзакціями запити даних до регістрів які знаходяться далеко один від одного [2]. Це значно мінімізує витрати часу на виконання всіх операцій.

Враховуючи що по протоколу Modbus в одній транзакції можна передати дані максимум зі 125 регістрів, удосконалений алгоритм працює з такими проміжками регістрів. Алгоритм знаходить кількість «непотрібних» регістрів (назвемо її N), розташованих один за одним, час передачі даних з яких антеною буде еквівалентний

повному часу посилання запиту з Землі й отримання відповіді без врахування часу передачі даних антеною.

Формула, з якої можна знайти значення N виглядає наступним чином:

$$2 * t_{\text{тр.д}} = k*(\frac{N*16}{V_{\text{пер.д}}} + t_{\text{пер.д.д}} + t_{\text{тиш}}),$$

де ttp.д. – час за який дані проходять відстань від Землі до супутника; k – статистичний показник, який вказує скільки транзакцій в середньому треба відправити щоб хоча б одна із них дійшла до супутника; tпер.д. – час за який дані з регістрів будуть передані антеною; Vпер.д. – швидкість з якою антена передає дані; tпер.д.д. – час за який додаткові дані будуть передані антеною; ttиш. – час тиші в кінці транзакції.

Обрахувавши це число регістрів (N), ми тепер знаємо, що якщо з проміжку в 125 регістрів ми викинемо неперервний проміжок в N «непотрібних» нам регістрів, то це буде еквівалентно, якщо ми використаємо додаткову транзакцію.

Нижче приведемо тестову ситуацію. Нехай для маємо умовний відрізок в 125 регістрів, який зображено на рисунку 1. Регістри, які необхідно зчитати з нього, позначені синім кольором.

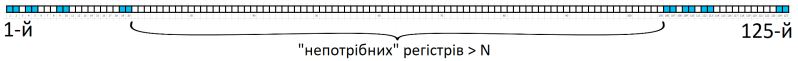


Рисунок 1 – проміжок в 125 регістрів для 1-ї тестової ситуації

Нехай в цьому тестовому прикладі, нам необхідна інформація із 60-ти таких умовних відрізків.

Обрахуємо і порівняємо результати роботи усіх трьох алгоритмів. Зробивши необхідні розрахунки за формулами описаними вище, отримаємо наступні результати для 3-х алгоритмів. Отримуючи із супутника всі дані («варіант 1»), це займе 1,29 сек., якщо кожний проміжок регістрів вичитувати окремо («варіант 2»), то це займе 5,39 сек., а якщо використати розроблений удосконалений алгоритм, то це займе 0,73 сек., що на 44% швидше ніж вичитувати всі дані відразу і на 87% швидше ніж вичитувати кожний необхідний регістр окремою транзакцією.

Методика випробувань для наносупутника

Виявляти аномалії у даних можна за допомогою декількох методів, наприклад метод відновлення щільності або метод класифікації, проте ці методи потребують значних ресурсів для розрахунків, що є значним недоліком при діагностуванні наносупутників, тому був використаний метод стандартизованої оцінки. Стандартизована оцінка(z-оцінка) – метод машинного навчання, який відносять до статистичних методів, оцінка розраховується за формулою[3]:

$$z = \frac{x\text{-}\bar{x}}{S_x},$$

де x – нове значення параметра; $\bar{x}$ – середнє значення на основі попередніх значень параметра; S_x – стандартне відхилення підраховане для множини даних x_i.

Розглянемо використання методу стандартизованої оцінки на прикладі вибірки температур процесора наносупутника. Дані вибірки зображено на рисунку 2.

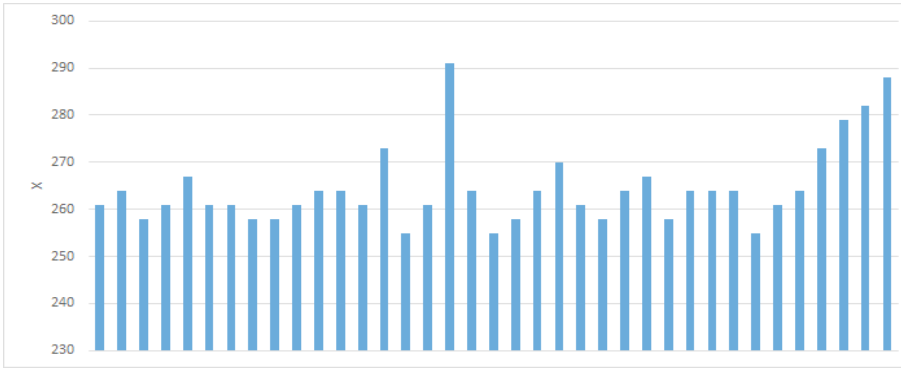


Рисунок 2 – друга вибірки

$\bar{x}$= 9792 / 37 ≈ 264.649;S_x^2 =($(261 - 264.649)^2$+ … + $(288 - 264.649)^2$) / 37 ≈ 67.19;S_x ≈ 8.197;

Тепер підрахуємо значення z-оцінки і побудуємо графік який зображений на рисунку 3.

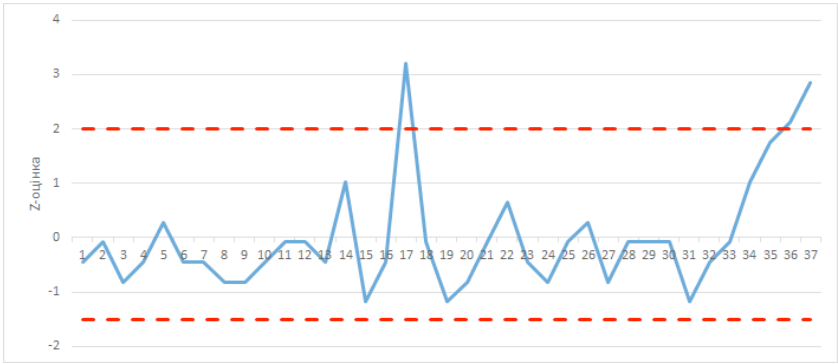


Рисунок 3 – z-оцінка для параметрів другої вибірки

Виявити аномалії за z-оцінкою можна встановивши допустимі межі значень які може приймати оцінка. Встановимо їх в межах [-1.5;2]. Тоді на нашому графіку можна побачити аномалію, яка виникає при значенні температури 291.

Висновки

Було досліджено протокол передачі даних Modbus й проаналізовано існуючі бібліотеки для роботи з ним. Була створена алгоритмічна та програмна база для декодування отриманих даних та їх відображення рефлексивними засобами. Був удосконалений алгоритм передачі даних по протоколу Modbus, а саме - удосконалено процес використання транзакцій, що в результаті скорочує витрати часу на отримання даних. Було проаналізовано існуючі методи та програмні засоби для діагностування систем наносупутника. Було проведено порівняльний аналіз методів виявлення аномалій під час діагностування компонентів наносутутника і за його результатами був обраний один із статистичних методів, а саме метод стандртизованої оцінки. Метод діагностики наносупутників був покращений за рахунок використання машинного навчання для виявлення аномалій.

Використана література:

1. Simply Modbus Master [Electronic resource]. — Access mode: http://www.simplymodbus.ca/RTUmaster.htm
2. Chipkin P. Modbus For Field Technicians / P. Chipkin , M: CreateSpace Independent Publishing Platform, 2011 — 78 с.

3. Kreyszig, E. Advanced Engineering Mathematics. John Wiley & Sons Inc, 4th edition, 880 p. (1979)

Дрозд Д. С.

Використання алгоритмів кластеризації для побудови рекомендаційних систем сайту новин

В сучасному світі ЗМІ генерують все більше і більше інформації кожен день. Користувачі не мають достатньо часу, щоб спожити всю інформацію, тому обирають найголовніше. Оскільки більшість ЗМІ працюють по рекламній моделі, тобто отримують оплату за перегляди реклами під час читання новин користувачами, то вони в свою чергу зацікавлені в тому, щоб читач прочитав якомога більше матеріалів.

Для того, щоб користувач прочитав, якомога більше новин, йому треба надавати новини, які з високою ймовірністю будуть йому цікаві. Майже кожний сайт новин містить блок «Вам також радять прочитати», який розміщується після тексту новини, в рамках цього блоку використовуються рекомендаційні системи.

Зараз існує велика кількість алгоритмів та методів побудови рекомендаційних систем, однак універсального рішення і досі немає. Рекомендаційна система тісно залежить від предметної області, наявності великої кількості історичних даних, швидкістю надання рекомендації та багатьох інших факторів.

Існує декілька основних підходів до розробки рекомендаційних систем. Колоборативна фільтрація, яка працює краще коли маємо багато даних користувача, але оскільки користувачі переглядаючи новини, не залишаючи багато інформації про себе, то якість рекомендацій на основі такого підходу часто залишає бажати кращого.

Крім колоборативної фільтрації є підхід побудови рекомендацій на основі фільтрації змісту, який будує зв’язки між статтями використовуючи алгоритми аналізу тексту новини.

За допомогою алгоритму аналізу тексту TEF— IDF можна побудувати матрицю відповідності частоти ключових слів та документів, в яких вони часто зустрічаються. Цю матрицю можна представити у виглядів векторів та за допомогою алгоритму k-середніх знайти кількість та сигнатури кластерів по яким індентифікувати ці кластери.

Наприклад, в результаті аналізу масиву даних змісту та заголовків статей, який складав 10 тис. статей сайту BBC Nigeria були визначені наступні тематичні кластери сайту (Рис. 1)

```
Top terms per cluster:
Cluster 0: governor government lagos said nigerians people new federal photos former
Cluster 1: police arrested command lagos officers officer arrest allegedly inspector force
Cluster 2: pdp apc party democratic peoples national chairman congress election progressives
Cluster 3: ya na buhari ba najeriya ta sa ce ne tsohon
Cluster 4: haram boko army borno terrorists insurgents attack troops sect military
Cluster 5: follow comment twitter facebook share click us read photos see
Cluster 6: news latest naij com read headlines page articles away world
Cluster 7: actress wedding nollywood woman photos year old lady baby actor
Cluster 8: manchester chelsea united arsenal league city madrid star eagles super
Cluster 9: man old year wife photos fetur young woman girlfriend caught
Cluster 10: ta na najeriya sun ya su kungiyar ne ba ga
Cluster 11: buhari president muhammadu jonathan former nigerians obasanjo said goodluck apc
```

Рисунок 1. Результат кластиризації сайту BBC Nigeria

Виходячи з цього можна ідентифікувати основні тематичні кластери наступним чином:

1. Уряду Нігерії
2. Кримінальні новини
3. Новини парламенту
4. Релігії Панагізм
5. Терористичної організації Боко-Харам
6. Посилання на соц.мережі
7. Короткі новини

8. Про весілля
9. Гендерні новини
10. Новини на діалектах Хауса
11. Новини президента

Маючи ці кластери можна легко проаналізувати нову статтю по алгоритму TEF-IDF та знайти відповідний їй кластер.

Маючи інформацію про кластер у блоці рекомендацій, який показується користувачеві новин можна виводити найбільш близькі за змістом статті з того ж кластеру, що і сама стаття.

Підхід кластеризації є досить вдалим рішенням при контентній фільтрації, оскільки дає можливість зекономити обчислювальні ресурси на обрахунку зв'язків між статтями, тобто не потрібно обраховувати кожну статтю окремо. Також алгоритм допомагає підбирати до поточної новини, інші, які з високою ймовірністю доповнюють новину або розкривають її з іншого боку.

Список використаних джерел

1. Riki F. Recommender system handbook / F. Riki, L. Rokach. – London. – 845 с.
2. ГРАНІК М. О. Метод визначення схожості новинних текстів на основі статистичної міри "Term frequency-inverce document frequency" : дис. канд. / ГРАНІК М. О., 2015. – 182 с.
3. Spanger A. Building the Next New York Times Recommendation Engine [Електронний ресурс] / Alexander Spanger – Режим доступу до ресурсу: https://open.blogs.nytimes.com/2015/08/11/building-the-next-new-york-times-recommendation-engine/.

Карпенко Є.Ю., Гуменний А.А.

Архітектура мобільних додатків на основі ОС Android

На даний момент Android найпопулярніша операційна система для мобільних пристроїв. Вона встановлена на 80% всіх смартфонів у світі.

За десять років існування android, розробники платформи приділяли увагу покращенню роботи самої системи: намагалися продовжити час роботи від акумулятора, підвищити ефективність використання обчислювальних ресурсів та комфортність інтерфейсу (швидкість відгуку, плавність, обробка помилок), розширити функціонал, якого дійсно бракувало в системі. І поки фундамент покращувався, фасад був спущенний на самоплив – уніфікованої архітектури представлено не було, а першоджерело – документація від Google – було наповнене прикладами, що направлені на демонстрацію функціоналу на низькому, відокремленному від реального коду, рівні. Дані приклади переносяться в реальні проекти з мінімальними змінаи, і як результат, Google Play наповнений погано написаними, нестабіно працюючими додатками, і платформа втрачає користувачів. Подібна неоднозначна ситуація обумовлює програмістів створювати певні стандарти та гайди з розробки та проектування додатків із ціллю підтримки високої якості роботи програм та ефективної організації робочого процесу. Але усі ці практики не є обов'язковими, і будь хто може написати програму у власному стилі.

Тому останнім часом із ціллю уніфікації архітектури під android Google став приділяти увагу розробці нових практик, які було б комфортно використовувати навіть новому чи невідповідальному

програмісту. Як результат, у 2017 році Google представив кілька проривних бібліотеки, такі як LiveData, Lifecycle і т.д, а також створив кілька статей, у яких представляє своє бачення на архітектуру андроїд додатків.

Типовою проблемою для android є реалізації логіки екранів, які представлені класами Fragment та Activity у фреймворку android. У даних класів складний життєвий цикл і тому є багато місць, де можна допустити баг. Тому у бібліотеці Lifecycle дані екрани були перевизначені класами LifecycleActivity, LifecycleFragment, після чого за життєвим циклом стало легше слідкувати. Новий клас Lifecycle дозволяє неекранним компонентам підписуватися на події життєвого циклу екранів без засмічення їх коду.

Реалізації класу Activity відповідають за різноматнітні функції, такі як інтерфейс користувача, завантаження та обробка даних – у чистому вигляді антіпатерн God Object.

І активіті, і фрагмент повинні буди відповідальними тільки за одну функцію – інтерфейс користувача – рендерінг даних та обробка взаємодії користувача із екраном, наприклад кліків. Така проблема існує не тільки в андроїд програмуванні, тому комьюніті програмістів розробило такі патерни проектування як MVP (Model-View-Presenter) чи MVVM (Model-View-ViewModel), які допомогають відділити логіку інтерфейсу від бізнес логіки у програмі. І зараз Google представив реалізацію MVVM патерну під назвою View Model – сховище стану для інтерфейсних компонентів, яке здатне пережити знищення UI компонентів, наприклад, під час зміни конфігурації екрану. Дана бібліотека також дозволяє облегшити взаємодію між окремими екранами, без необхідності створення прямих посилань на об’єкти.

За 10 років існування android було представлено багато реалізацій ORM та обгорток для бази даних. Всі ці рішення не є офіційними, опираються на різні концепти, і тому вибір потрібної реалізації потребує багато часу. Тепер Google запропонував офіційне рішення – Room Persistence Library, яка дозволяє описувати базу

даних використовуючи ієрархію класів, що дозволяє уникнути багатьох помилок ще на етапі компіляції. Бібліотека використовує механізм Java анотацій для представлення операцій sql, що також дуже зручно та легко для розуміння.

Всі ці бібліотеки від Google, а також певні напрацювання від інших компаній, таких як Square, зливаються в єдину архітектуру, що є наразі рекомендованою для додатків android.

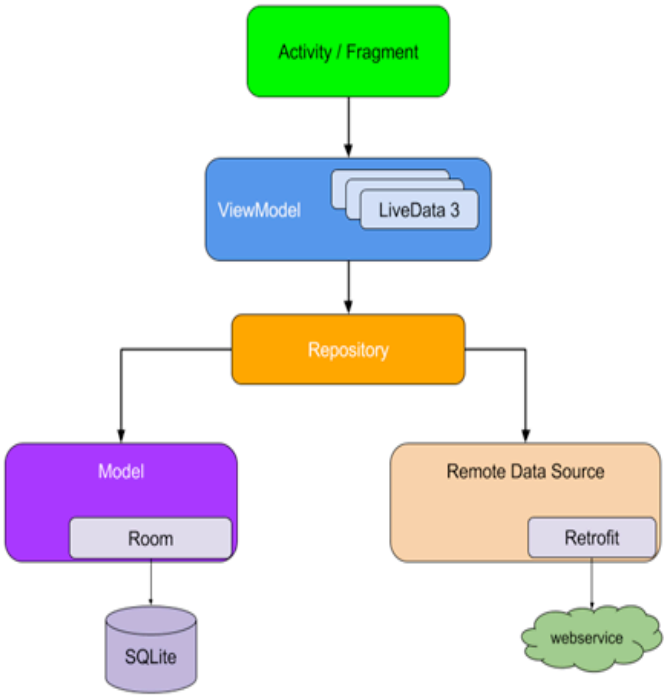


Рисунок 1 – Архітектура від Google

Якщо придивитись уважно і згадати тенденції в розробці android додатків за останні три роки, то у даній діаграмі можно впізнати архітектуру, відому під назвою Clean Architecture, що

з’явилася ще задовго до платформи android, і є актуальною для будь якої сучасної програми.

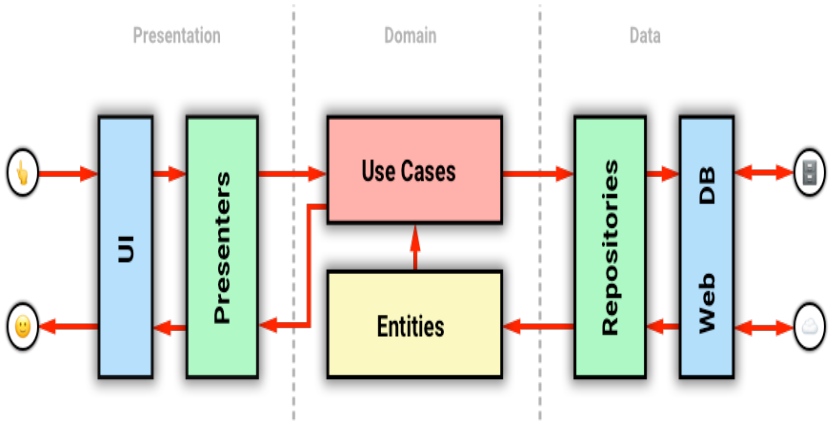


Рисунок 2 – Clean Architecture

Порівнявши діаграми, можна зрозуміти що архітектура це те ж саме, тільки повернуте на 90 градусів.

Як бачимо, на обох діаграмах програма логічно розбита на три модулі – інтерфейс, бізнес логіка- та сховище даних. Але ця детальн е є основною для побудови правильної архітектури.

У канонічному зображенні чистої архітектури роль грають не шари, а напрямок зв’язків між ними.

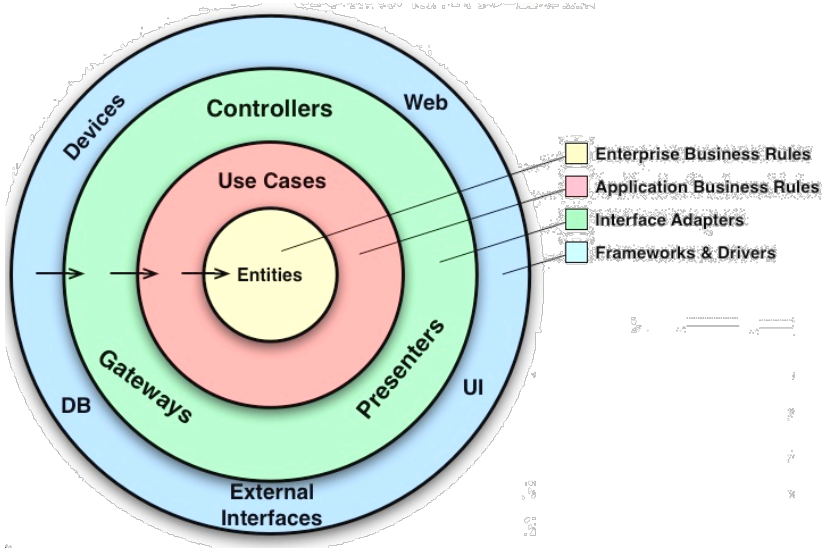


Рисунок 3 – Канонічне зображення Clean Architecture

Зверніть увагу на напрям стрілок у канонічній діаграмі Clean Achitecture.

В середині знаходяться сутності бізнес-логіки. Далі йдуть правила бізнес-логіки, або методи для роботи із сутностями. Ні сутності, ні методи не опираються на конкретні технології чи платформи – це звичайні класи, що можна переносити з проекта у проект, без підтягування нових залежностей. Звичайно, правила залежать від сутностей, але сутності не залежать від правил. Таким чином, залежність направлена в одну сторону – з зовнішнього шару до центру. На звонішньому шарі знаходяться фреймворки (орм для баз даних, http-клієнти, сам фреймворк android). Використання тих чи інших бібліотек та фреймворків залежить від бізнес логіки, але не навпаки. Тобто однонаправлений зв’язок прослідковується і тут.

Як висновок, слід зазначити, що основною задачею при побудові архітектури є визначення шарів логіки таким чином, щоб зв’язки зберігались тільки в одному напрямку. Створення подібних логічних обмежень облегшує у подальшому процес розширення

проекту та вибір правильних методик при виникненні неоднозначних на перший погляд методик.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Guide to app architecture [Електронний ресурс] — Режим доступу: https://developer. android.com/jetpack/docs/guide.
Freeman E. Design Patterns [Електронний ресурс] / Eric Freeman — 2004 — Режим доступу: http://shop.oreilly.com/product/9780596007126.do.

Карпенко Є.Ю., Касьяненко І.І.

Розподілена система керування організаційними процесами кафедри

Розвиток комп'ютерів і мережі інтернет створює основу для стрімкого поширення розподілених мобільних та веб додатків. Багато тривіальних розподілених систем застарівають та заміняються більш сучасними та ефективними електронними аналогами.

Особливу увагу слід звернути на системи, які забезпечують навчальний процес.

В розвинених європейських країнах питання модернізації освіти ставиться пріоритетною ціллю.

В Україні проблеми освіти розглядаються недостатньо [1]. Нормою є застарілі програми, методології навчання, майже відсутнє впровадження електронних технологій в освіту, це все призводить до загальної регресії освітньої системи.

Однією з насущних проблем вищих навчальних закладів є повільні та малоефективні схеми взаємодії студентів з викладачами, відсутність уніфікованого інструменту впливу на процес навчання через інтернет, застарілість організації процесу вибору тем дипломних робіт студентів та їх узгодження.

Не завжди узгоджений графік прийому та перевірки лабораторних робіт, нечіткі терміни та правила спричиняють створення черг студентів, розпливчате усвідомлення процесу написання та здачі завдань, це все спричиняє неефективний розподіл часу та зусиль студентів та викладачів.

Роздрібненість інформації про кафедри, їх направлення, досягнення створює розпливчасте усвідомлення загального стану факультету, тим самим, відштовхуючи абітурієнтів.

Було запропоновано створити розподілену систему [2], яка дозволить зручно та ефективно організувати процес взаємодії викладачів зі студентами, надасть консолідовану інформацію про напрями та роботу кафедр, дозволить вирішувати організаційні моменти в онлайн режимі.

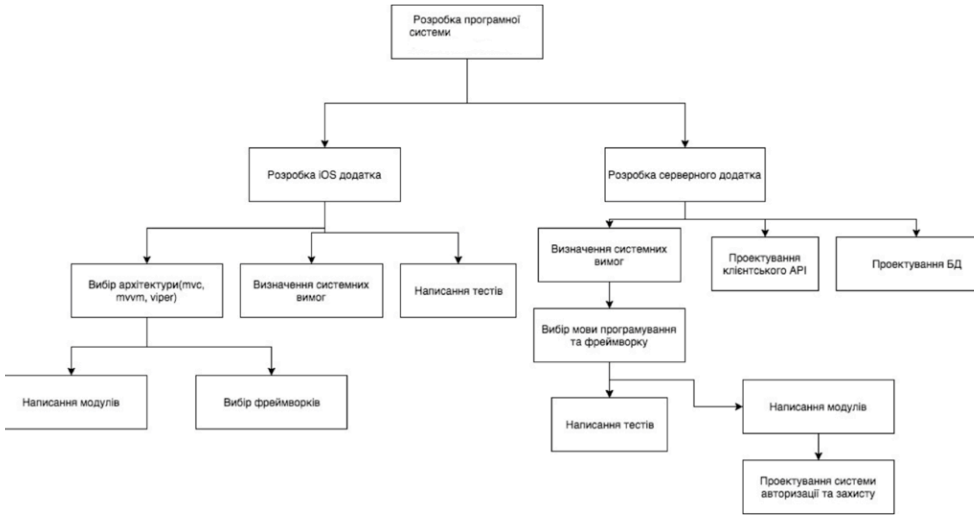


Рисунок 1 - Дерево цілей

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. https://mon.gov.ua/ua/tag/yakist-osviti
2. http://www.sm-cloud.com/ios-client-server-integration-approach/

Карпенко Є.Ю., Фішер О.Є.

Візуалізація стану ґрунтів України на основі геоданих

Вступ. Для багатьох держав ринок землі вже давно став нормою. Країни активно розробляли нормативно-правові бази для реалізації виходу власних внутрішніх ринків землі на світові торгівельні площадки [1]. Наразі Україна – є однією із 6 країн у світі, що блокують національний вихід на ринок землі. Інші такі країни це: Південна Корея, Куба, Венесуела, Таджикистан і Конго. Аграрна економіка таких країн дещо послаблена через відсутність вільної можливості продавати земельні угіддя іноземним інвесторам, а через відсутність конкуренції – фермери, які здають землі в оренду компаніям, що знаходяться на території країни, недоотримують значні кошти.

Законодавча влада України працює у напряму скасування мораторію на продаж землі (2). Формуються нові нормативно-правові бази для ухвалення цього рішення через значну складність процесів та необхідність активної регуляції ринку на перших етапах. Як і з будь-яким ринком, новостворений ринок землі буде невідомий ні для жителів України, ні для іноземних інвесторів. Найкращим варіантом для таких ситуацій – є програмне забезпечення, яке може надати оцінку земельним ділянками та зорієнтувати зацікавлених людей, які бажають вкласти гроші у Українські землі.

Існує ряд програмних систем, розроблених за кордоном, що здійснюють оцінку земельних ділянок, дозволяють робити порівняльний аналіз вартості протягом різних років та дізнаватись типи ґрунтів, та їх родючість. Такі системи визначаються своєю публічною доступністю та національно фінансовою підтримкою.

AcreValue – це американська програмна система з оцінки стану ґрунтів та вартості земельних ділянок. Програма AcreValue була розроблена приватною компанією Granular Inc. і опублікована для вільного доступу на публічному сайті. Програма виконана у вигляді карти із відповідними шарами полів та вбудованою аналітичною системою, що дозволяє будь-кому зацікавленому у реалізації земельних ділянок безкоштовно дізнатись ціни та стан землі, вільні для продажу ділянки, найвигідніші пропозиції. В основі AcreValue знаходиться Географічна база даних опису ґрунтів - SSURGO, що була розроблена Національним кооперативом огляду ґрунтів – NCSS. Недоліком цієї програмної системи є те, що вона розроблена виключно для території США і не може бути застосована на території України. З технічної точки зору – програмна система зроблена на вищому рівні і актуальність даних про ґрунти підтримується на національному рівні силами NCSS.

Віддаленим українським аналогом системи з аналізу ґрунтів є декілька публічних сайтів, які дозволяють отримати інформацію про стан ґрунтів на Україні. Одним із публічних засобів отримання цієї інформації – є сайт http://geomap.land.kiev.ua. На сайті пропонується певна інформація щодо типізації ґрунтів, їх родючості та багато різної загальної геоінформації. Сайт представляє собою не суцільний інструмент для оцінки землі, а набір тематичних карт загального призначення. Сайт, створений саме для ознайомлення широкого кола користувачів з особливостями навколишнього середовища загалом та його окремих природних компонентів. Таким чином на сайті представлено електронні карти природних компонентів для території України (3).

Іншим публічним сайтом з отримання часткової геоінформації про стан ґрунтів України – є http://map.land.gov.ua/kadastrova-karta. Мапа представляє собою великий інтерактивний каталог по кадастровим територіям із наявними шарами для відображення різних даних і розроблена Держгеокадастром. Карта має багато корисних інструментів та деякі можливості для аналізу стану ґрунтів, однак

нажаль, ця карта спрямована на забезпечення громадян кадастровою інформацією. Система не має прикладної функціональності з візуалізації стану ґрунтів, екологічної ситуації чи розвитку транспортних системи. Наявні лише фільтри по адміністративним одиницям і пошук за кадастровим номером.

Відсутність систем з оцінки вартості землі для Українського ринку не тільки гальмує вихід України на світовий ринок землі, але і ускладнює роботу Українських фермерів. Без ретельного аналізу земельної ділянки, що продається важко сказати наскільки буде продуктивна ця ділянка, яка її реальна ціна, яка екологічна ситуація та які тип ґрунтів. Такий ретельний аналіз зазвичай доступний не для кожного фермеру.

Постановка задачі. Метою цього дослідження є оцінка вартості земельної ділянки за формулами, розробленими у постанові про затвердження методики нормативної грошової оцінки земель сільськогосподарського призначення (від 16 листопада 2016 р. № 831)[1], покращення цієї оцінки за допомогою використання значущих шарів, що впливають на фінальну оцінку земельної ділянки, та створення програмного забезпечення для автоматизації цього процесу.

Програмна система призначена для використання фермерами, банкірами, інвесторами, та будь-якими іншими зацікавленими особами. Система має бути реалізована у вигляді карти та інструментарію, що дозволяє проводити фільтрацію та вибір земельних ділянок. Результатом роботи має бути оцінка земельної ділянки та підрахунок її можливої вартості.

Обрахунок оцінки земельної ділянки. У постанові про затвердження методики нормативної грошової оцінки земель сільськогосподарського призначення визначається формула, яка дозволяє розрахувати вартість земельної ділянки:

$$\Gamma_{зд} = \Sigma \left(П_{агр} \times \Gamma_{агр}\right) + П_{нсг} \times \Gamma_{нсг}$$

де $Г_{агр}$ – нормативна грошова оцінка агровиробничої групи ґрунтів відповідного сільськогосподарського угіддя природно-сільськогосподарського району, гривень за гектар;

$П_{агр}$ – площа агровиробничої групи, гектарів;

$П_{нсг}$ – площа несільськогосподарських угідь, гектарів;

$Г_{нсг}$ – норматив рентного доходу несільськогосподарських угідь на землях сільськогосподарського призначення, гривень за гектар.

Формула представлена у постанові використовує табличні дані для оцінки вартості гектару землі, зокрема $Г_{зд}$ та $Г_{нсг}$ – табличні дані вказані в додатку до постанови, відрізняються для різних областей та районів.

Нажаль, ця формула не бере до уваги фізичні властивості місцевості земельної ділянки: такі як особливості ґрунту, екологічну ситуацію, наявність доріг, населених пунктів, водних ресурсів. На практиці, наприклад, земельні ділянки, що ближче знаходяться до населених пунктів будуть коштувати більше, ніж земельні ділянки, що знаходяться далі, навіть у тому самому регіоні (4).

У програмній системі запропоновано покращити цю формулу використовуючи шари земельних ділянок з пронормованими значеннями для певних територій. Так наприклад, використовуючи на карті шар ґрунту з пронормованими значеннями за показником родючості від 1 до 100, можна скорегувати оцінку земельної ділянки. Для корегування будемо використовувати обернену залежність, тобто:

$$СГО = НГО \times (1 - \frac{1}{Родючість})$$

де СГО – скорегована грошова оцінка;

НГО – нормативна грошова оцінка;

Родючість – пронормований показник родючості ґрунту від 1 до 100.

З формули витікає що при умовній родючості земельної ділянки в 1, аграрна вартість такої земельної ділянки прямує до нуля, що має сенс. Таким чином запропоновано використати додаткові пронормовані шари для збільшення точності оцінювання земельної ділянки, а саме:

- віддаленість від населених пунктів;
- наявність транспортних магістралей, залізниць;
- виснаженість ґрунту;
- екологічна ситуація на території.

Для створення програмної системи запропоновано використати відкриті програмні бібліотеки зі створення геоінформаційних систем. Для серверної частини – спеціалізований сервер з відкритим кодом GeoServer. Для клієнтської веб-сторінки пропонується використати бібліотеку OpenLayers.js, яка дозволяє відображати шари від різних геосерверів у популярних форматах даних кодування шарів мап.

Висновки. Проведено аналіз існуючого програмного забезпечення з візуалізації стану ґрунтів та оцінки вартості земельних ділянок, розробленого в Україні та в США. Обґрунтовано необхідність створення програмного забезпечення з візуалізації стану ґрунтів для України на основі геоданих. Проаналізована нормативно-правова база та запропоновано метод покращення оцінки стану землі. Запропоновано програмний інструментарій для розробки програмної системи.

Перелік використаних джерел

1. Fraser E. https://www.oaklandinstitute.org/blog/who-owns-agricultural-land-ukraine [Електронний ресурс] / Elizabeth Fraser. – 2015. – Режим доступу до ресурсу: https://www.oaklandinstitute.org/blog/who-owns-agricultural-land-ukraine.
2. Мораторій на продаж землі буде скасований до кінця року - меморандум МВФ [Електронний ресурс] // Економічна

Правда. – 2017. – Режим доступу до ресурсу: https://www.epravda.com.ua/news/2017/04/5/623477/

3. Чугуєнко М. В. Моя Україна. Ілюстрована енциклопедія для дітей. — Харків: Веста: Видавництво «Ранок», 2006. — 128 с. іл.
4. Капралов Е. Г. Основы геоинформатики / Е. Г. Капралов, А. В. Кошкарев, В. С. Тикунов., 2004.

Карпенко С.Г., Защик С.М.

Сортування масивів об’єктів з використанням багатопоточності на C++

У мові C++, підтримка багатопоточності з’явилась із введенням міжнародного стандарту “C++11”. Його було прийнято у 2011 році, на зміну застарілим стандартам “C++98” і “C++03”. Багатопотояність цей стандарт реалізує в класі thread, що визначений в заголовковому файлі thread.h. Щоб запустити новий потік виконання необхідно створити об’єкт класу thread і ініціалізувати його, передавши в конструктор ім’я функції, що повинна виконуватися в потоці і, за потреби, її параметри. У загальному випадку, розпаралелювання операцій можливе, коли кінцеве завдання складається з незалежних підзавдань, які можна виконати одночасно на різних фізичних ядрах-процесорах. Такий розподіл операцій і називається паралельним (або багатопоточним) [1].

Форма апаратної реалізації паралелізму залежить від рівня, на якому комп'ютер його підтримує. Якщо система багатоядерна (існує кілька потоків опрацювання даних) процеси справді можуть виконуватись одночасно, як у випадку роботи на різних процесорах. Якщо ж існує лише одне фізичне ядро, то планувальник завдань системи з досить малим інтервалом часом переключається між виконанням умовно паралельних потоків для створення ефекту одночасності [2, с.24-26].

Щоб практично перевірити ефективність використання багатопоточності для сортування масиву об’єктів, було створено програму на мові C++, що містить ієрархічну модель класів, контейнер з масивом кінцевих сутностей, метод для сортування масиву об’єктів у кілька потоків та аналогічний йому однопоточний

метод.

Для того, щоб масив можна було відсортувати кількома потоками виконання, кожен з них повинен працювати лише з частиною масиву. Оскільки потоки виконуються асинхронно, не можна, щоб вони одночасно працювали з цілим масивом. В такому випадку можлива ситуація, коли, наприклад, два паралельні потоки приблизно одночасно перевірили, що елемент №3 більший ніж елемент №4. Тоді, логічно, запустяться функції "свапу" – обміну місцями об'єктів у масиві. Подвійний "свап" в такому разі поверне все до попереднього стану і умовно "відсортована" пара елементів насправді такою не буде. Відповідно алгоритм працюватиме некоректно. Отже сортувати потрібно частини масиву, а потім об'єднати вже відсортовані підмасиви в один.

Для того, щоб час об'єднання підмасивів мінімально впливав на результат, було обрано метод сортування "Бульбашка". Він здійснює фіксовану кількість проходжень, не залежно від значень елементів масиву та ступеня його початкової відсортованості. Його трудомісткість T:

$$T(N) = N^2, \qquad (1)$$

де N – кількість елементів масиву. Нехай для кожного з N проходжень потрібно здійснити n_0 операцій. Тоді загальна їх кількість k_0, відповідно до формули (1):

$$k_0 = n_0 * N^2 \qquad (2)$$

Щоб показати залежність часу сортування від кількості потоків виконання, сортування в один потік теж було розбите на частини. Якщо просто повністю відсортувати масив об'єктів, кількість операцій визначається формулою (2). Однак якщо розбити цей масив на дві

половини, відсортувати їх, а потім об'єднати за одне проходження — кількість операцій k_2 буде дорівнювати

$$k_2 = n_0 * \left(\frac{N}{2}\right)^2 + n_0 * \left(\frac{N}{2}\right)^2 + n_0 * 1 = n_0 * \frac{N^2}{2} + n_0. \qquad (3)$$

Де N – кількість елементів масиву, n_0 – кількість операцій при порівнянні елементів, третій доданок – кількість дій для об'єднання половин масиву, що здійснюється за одне його проходження. При великих довжинах масивів, доданок n_0 достатньо незначний і ним можна знехтувати. Формула знаходження k_2 в такому разі матиме вигляд:

$$k_2 = n_0 * \frac{N^2}{2}. \qquad (4)$$

Звідси – розбиваючи на частини отримуємо виграш обернено пропорційний кількості частин розбиття (справедливо, поки кількість частин розбиття значно менша кількості елементів в частинах).

Дані формул (2), (4) підтверджуються експериментально запуском однопоточного сортування без розбиття на частини і з розбиттям на 2, 3, 4 частини відповідно (Рис. 1).

Час сортування подано в секундах:

length	Mono parts 1	Mono parts 2	Mono parts 3	Mono parts 4
1000	0.107	0.054	0.036	0.027
5000	2.795	1.358	0.907	0.684
10000	12.851	5.526	3.641	2.735

Рис. 1. Результат робот програми. Виграш у часі від розбиття на частини обернено пропорційний кількості частин розбиття.

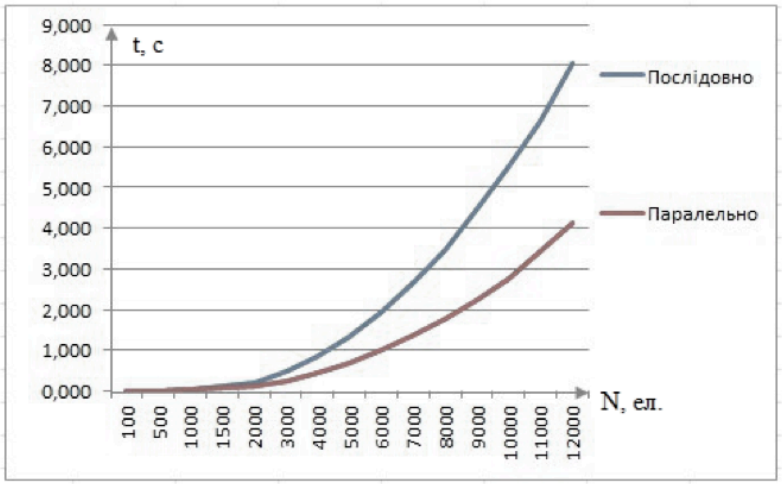
Для максимально об'єктивної оцінки виграшу в часі сортування тільки за рахунок багатопоточності, порівняння

здійснюється для однієї і тієї ж кількості частин. У випадку багатопоточності сортування ці частини сортуються паралельно, у випадку однопоточності - послідовно.

На основі результатів роботи програми виявлено залежність — час виконання сортування обернено пропорційний кількості паралельних потоків виконання. Для значень довжини масиву у 100, 500, 1000, 1500, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000, 11000, 12000 елементів, середнє значення часу сортування:

Таблиця 1. Час сортування при розбитті масиву на частини

	Час сортування [сек]			
Кількість частин розбиття масиву	2	3	4	8
Послідовне сортування	2,391	1,586	1,192	0.594
Паралельне сортування	1,216	0,549	0,310	0,164
Номер рисунку	2	3	4	5



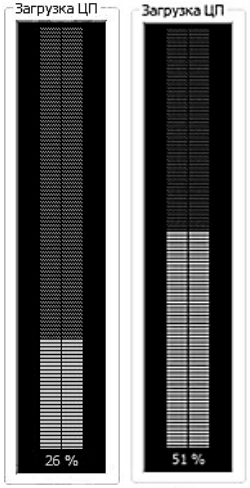
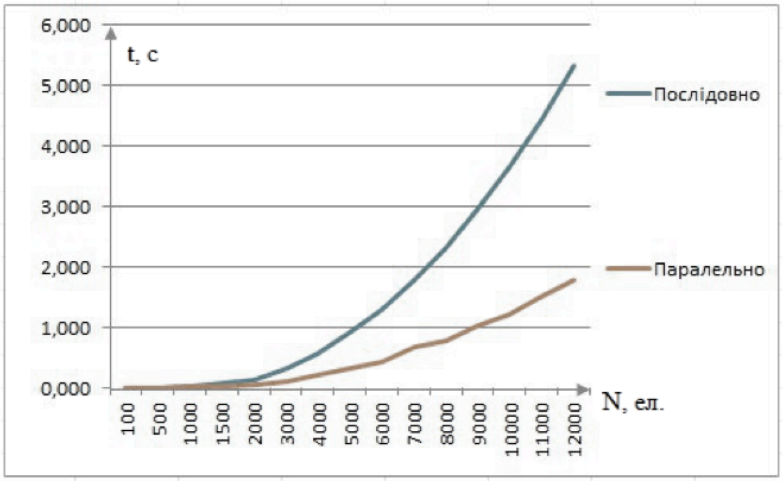


Рис. 2. Залежність часу сортування від довжини масиву при розбитті на 2 частини. Загрузка ЦП 26% послідовно, 51% паралельно.



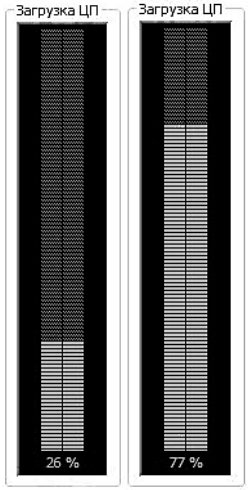
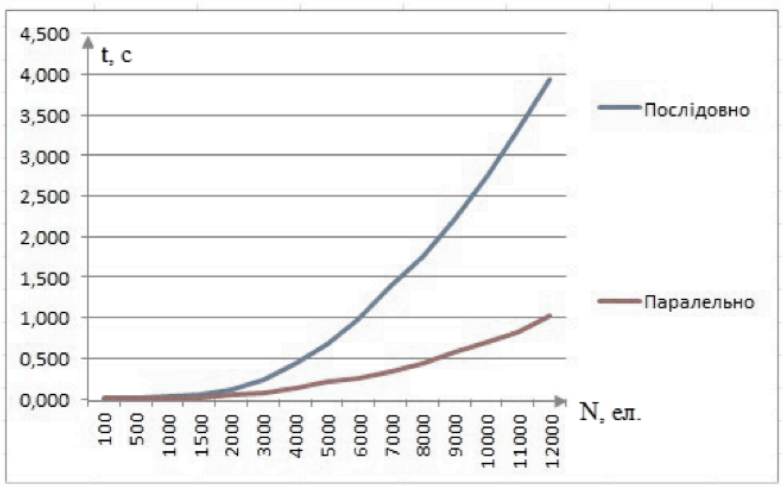


Рис. 3. Залежність часу сортування від довжини масиву при розбитті на 3 частини. Загрузка ЦП 26% послідовно, 77% паралельно.

На рис. 4-5 масштаб графіків відрізняється за рахунок розбиття на більшу кількість частин, хоча пропорції (чотирьохразова перевага у швидкості паралельного сортування над послідовним) та завантаження процесора однакові. Перевага у швидкості за рахунок багатопоточності обмежена кількістю фізичних ядер процесора. Масштаб змінився за рахунок розбиття на більшу кількість частин (формули 2, 4). Тести проводились на пристрої з чотирма фізичними ядрами процесора, тому максимальний виграш у швидкодії за рахунок розпаралелювання у даному випадку ─ чотирьохразовий.

Таким чином, паралельне сортування оптимальне, коли кількість паралельних потоків рівна кількості фізичних ядер процесора. Збільшувати кількість потоків виконання далі ─ не раціонально, оскільки витрачається час на ініціалізацію потоків та їх обслуговування системою при відсутньому виграші від розпаралелювання. Замість подальшого збільшення кількості потоків раціонально розбивати масив на більшу кількість частин, сортуючи їх на оптимальній (рівній кількості ядер) кількості потоків.



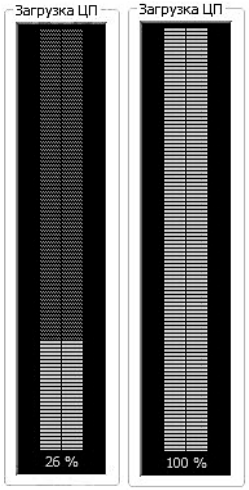
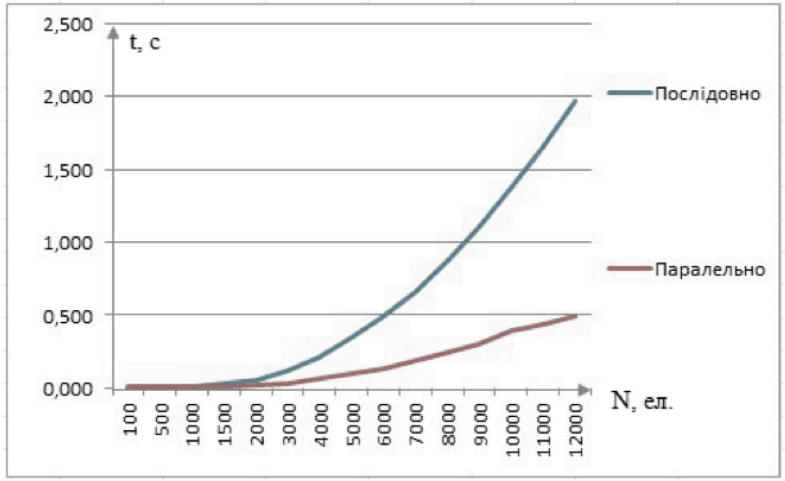


Рис. 4. Залежність часу сортування від довжини масиву при розбитті на 4 частини. Загрузка ЦП 26% послідовно, 100% паралельно.



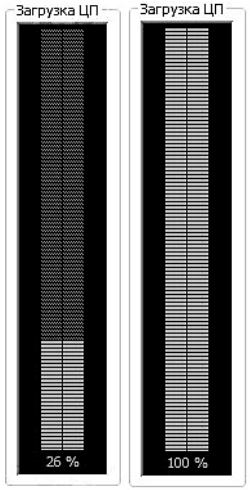


Рис. 5. Залежність часу сортування від довжини масиву при розбитті на 8 частин. Загрузка ЦП 26% послідовно, 100% паралельно.

Список літератури

1. Многопоточность, новые возможности стандарта C++11 [Електронний ресурс] / Режим доступу: http://www.quizful.net/post/multithreading-cpp11.

Энтони Уильямс. Параллельное программирование на C++ в действии. Практика разработки многопоточных программ. — М.: ДМК Пресс, 2012.– 672с.

Кублій Л.І., Івашин В.В.

Вибір інструментів для фіксації і відстеження прогресу виправлення помилок

У наш час розробляють великі й складні програмні продукти. Технічне обслуговування таких складних систем досить затратне, а програмне забезпечення не завжди досконале. Це означає, що для програмного забезпечення може знадобитися додатковий модуль або вдосконалення існуючого модуля, після чого він може містити деякі непомітні або неперевірені помилки, які час від часу залишаються в програмному забезпеченні. Помилка може з'являтися на будь-якій стадії розробки програмного забезпечення: аналізу вимог (RA), проектування (SD), кодування (SC), тестування (ST), реалізації (SI), обслуговування системи (SM).

У проектах з відкритим вихідним кодом зі швидким збільшенням кількості програмістів, які постійно роблять свій внесок у розробку і вдосконалення проекту, є можливість внесення нових помилок у проект. На веб-сайт проекту щодня відправляється кілька помилок, які можуть використовувати деякі інструменти керування конфігурацією (SCM) для керування версіями і випуском програмного забезпечення. Інструменти SCM можуть не надавати ніяких звітів про помилки, а також про хід виправлення помилок. Існує нагальна необхідність щодо планування і впровадження найкращої системи відстеження помилок і звітності. У середовищі з відкритим вихідним кодом, як правило, коли помилка відправляється, будь-яка людина може почати роботу щодо її виправлення. Але в той самий час інші люди можуть також почати роботу над виправленням тієї ж помилки. Тому модератор проекту буде вибирати, яке рішення реалізувати в системі. Асоціація Business Software Alliance провела дослідження, метою якого було порівняння кращих

постачальників програмного забезпечення для виявлення дефектів і помилок. Не існує визначеного графіку часу для виправлення помилки, ні особи / команди, відповідального за своєчасну фіксацію помилки [1].

У патентованому програмному забезпеченні виправлення помилки відбувається поетапно, оскільки на кожному етапі є окрема група, відповідальна за виконувану роботу. У той же час в проекті з відкритим вихідним кодом будь-хто може знайти помилку і почати працювати над нею. Залежність може стати основною проблемою при обчисленні часу для виправлення помилки. Якщо помилка виявлена до фази реалізації, вона може бути якісно проаналізована і виправлена командою розробників або власником модуля. Але як тільки програмне забезпечення випущено й впроваджено, стає дуже складно виправити помилку. Для модератора є громіздким завдання знайти відповідного розробника, який може виправити конкретну помилку, оскільки опис повідомлення про помилку може взагалі не надавати повної інформації.

Для звичайної команди розробників з двох або більше людей, системи керування проектами надають широкий спектр інструментів і методологій, які допомагають керувати проектом на різних стадіях життєвого циклу програмного забезпечення. Такі системи мають багато переваг, тому вони поступово стали популярними при розробці програмного забезпечення. Найважливішими їхніми перевагами є: виявлення помилок, покращення прозорості процесу розробки, допомога при плануванні випуску програмного продукту, керування плануванням ресурсів, відстеження стану помилок, взаємодія з командою проекту в режимі реального часу, коли кожен член команди може розв'язувати питання при їхньому виникненні.

Система відстеження помилок або дефекту — програма, яка відстежує програмні помилки, про які повідомляють, у проектах розробки програмного забезпечення. Це може бути розцінено як тип системи стеження проблеми [2].

Багато систем відстеження помилок надають кінцевим користувачам можливість проглядати звіти про помилки. Інші системи

використовуються тільки внутрішньо в компанії або організації. Як правило, системи відстеження помилок об'єднані з іншими додатками керування проектом програмного забезпечення.

Система відстеження помилок — необхідний компонент хорошої інфраструктури розробки програмного забезпечення, і послідовне використання помилки або системи стеження проблеми вважають однією з ознак хорошої команди програмного забезпечення.

Типові системи відстеження помилок підтримують поняття життєвого циклу для помилки, за якою стежать через статус, призначений на помилку. Система відстеження помилок повинна забезпечити адміністраторам можливість формувати дозволи, засновані на статусі, переміщати помилку в інший статус або видаляти її, формувати статуси помилки і встановлювати, до якої міри помилка в особливому статусі може бути переміщена [2].

Система звітності / відстеження помилок повинна надавати інтерактивну веб-платформу для звітності про помилки і відстеження прогресу. Система може включати в себе загальний процес або конкретний графік обробки повідомлень про помилки.

Як тільки помилка буде відправлена в систему, менеджер побачить її статус і подивиться деталі звіту і його параметри. Менеджер також може скинути або оновити деякі параметри і відповідно оновити статус і баг. У великих проектах з відкритим вихідним кодом середня швидкість передачі повідомлень про помилки досить висока. Наприклад, у проекті середня швидкість передачі повідомлень про помилки становить 50 випусків на день. Таким чином, менеджеру дуже складно переглядати кожну помилку, а потім оновлювати статус і присвоювати помилку певній людині [3].

Є багато систем відстеження помилок, доступних для використання. Систему відстеження помилок також називаються системою відстеження проблем, багтрекером, системою відстеження дефектів тощо. Такі системи розробляються співтовариством з відкритим вихідним кодом, а також організаціями з закритим кодом як пропрієтарне програмне забезпечення. Відкритий вихідний код

означає, що вихідний код надається всім відповідно до політики GPL. Кожен може зробити свій внесок в код добровільно. У спільнотах з закритим вихідним кодом вихідний код є власністю організації, і люди, які не входять в проект, можливо, не зможуть переглянути код. Існують організації, які також можуть надавати підтримку рішенням з відкритим вихідним кодом, наприклад, Redhat Inc. надає Red Hat Enterprise Linux (RHEL) і підтримує спільноти проектів з відкритим вихідним кодом Fedora.

Однією з доступних систем відстеження помилок є Bugzilla (1998 рік) — вільне програмне забезпечення відстеження помилок (багтрекінгу) з веб-інтерфейсом [4]. Вона досить проста, проте у ній є все потрібне для багтрекінгу типового проекту. Зараз її використовують більше тисячі компаній і організацій у всьому світі, серед яких: NASA, Id Software, IBM, Novell, Основа телеком і програмні проекти Mozilla Firefox, Linux, GNOME, KDE Software Compilation, Apache Project, OpenOffice.org, Wikimedia Foundation.

Ключовим поняттям системи Bugzilla є "баг" — деяке завдання, запит, рекламація щодо помилки в системі, або повідомлення, яке потребує зворотного зв'язку [4]. Система Bugzilla є інтегрованою системою з можливістю визначення безпеки на рівні продуктів. Вона відслідковує залежності помилок і виводить їх у графічному вигляді, має розвинену підсистему складання звітів, дуже зрозумілий і добре продуманий протокол рішення помилок, стабільний back-end на основі RDBMS, має API для електронної пошти, XML, HTTP і консолі. Можлива інтеграція системи Bugzilla з системами керування автоматичного конфігурування програмного забезпечення, у тому числі Perforce and CVS (через інтерфейс електронної пошти Bugzilla і скрипти для checkin/checkout) [4].

Головний компонент системи відстеження помилок — база даних, у яку записуються факти про відомі помилки. Факти можуть включати час, помилку, про яку повідомили, її серйозність, помилкову поведінку програми і деталі про те, як відтворити помилку; також

особу, яка повідомила про неї і будь-якого програміста, який може працювати над її фіксацією.

Програмна система Elasticsearch (ES) [5] — пошуковий движок з набором методів json rest api для взаємодії з сервером. Система використовує бібліотеку Lucene. Вона реалізує такі ж механізми, як і при складному пошукові в базі документів (наприклад, пошук з врахуванням морфології мови, пошук за geo-координатами), підтримує як шардінг, так і установку кількох мод у кластер. Як і будь-який пошуковий движок, Elasticserach залежить від оперативної пам'яті, тому треба при проведенні навантажувального тестування звертати особливу увагу на цей параметр.

Керування проектами для організацій є основним способом оптимізації процесів розробки й створення проектів і можливістю реалізувати проекти з максимальною результативністю й ефективністю. Відстеження знайдених помилок для виправлення є важливим напрямком досліджень у програмному забезпеченні. У даний час розробляється багато відкритих, безкоштовних і комерційних інструментів для відстеження помилок. Організаціям потрібні критерії вибору кращого інструменту серед доступних для фіксації і відстеження прогресу виправлення помилок.

Здійснено аналіз особливостей автоматизації відстеження помилок у програмному забезпечені. Також розроблено систему, яка дає можливість використовувати функції системи керування проектами, а також здійснювати пошук інформації швидше з використанням технології Elasticsearch.

Література:

1. Jalbert N. Automated Duplicate Detection For Bug Tracking Systems / N. Jalbert, W. Weimer. // International Conference on Dependable Systems & Networks: Anchorage, Alaska, June 24-27 2008. — P. 52-61. — https://web.eecs.umich.edu/~weimerw/p/weimer-dsn2008.pdf

2. Канер С. Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений / С. Канер, Д. Фолк, Е. Нгуен; пер. с англ. — К.: Издательство ДиаСофт, 2001. — 544 с.

3. Оценка систем отслеживания ошибок [Електронний ресурс]. — Режим доступу: sci-article.ru/stat.php?i=1494945305

4. The Bugzilla Guide [Електронний ресурс]. — Режим доступу: bugzilla.mozilla.org/

5. Elasticsearch [Електронний ресурс]. — Режим доступу: elastic.co/products/elasticsearch

Кублій Л.І., Костенко І.П.

Огляд ефективності паралельного генетичного алгоритму на базі теоретичної моделі “зірка”

Аналізуючи базові методи евристичної оптимізації повного перебору і подібні до них, включаючи такі, як метод мурашиного алгоритму, імітації відпалу гілок і меж, можна виявити, що доцільність й актуальність використання таких методів залежить від обсягу області пошуку можливих рішень і зменшується зі збільшенням цієї області [1-3]. Для випадку задачі комівояжера, на основі якої проводилося тестування, це область пошуку до 60 міст.

Порівнюючи результати роботи базових методів з результатами роботи генетичного алгоритму, можна виділити кілька суттєвих переваг:

— менша потреба часу;

— точніший розв’язок;

— можливість паралельної реалізації;

— можливість адаптації під конкретну задачу.

Відповідно до отриманих результатів проведеного обчислювального експерименту було проаналізовано поведінку паралельної реалізації генетичного алгоритму для аналогічної задачі.

Для тестування було обрано просту реалізацію паралельного генетичного алгоритму і реалізовану на базі нової теоретичної моделі

структуру типу “зірка” зі зміненим оператором кросинговеру. Для кожного алгоритму було проаналізовано 100 ітерацій запуску на кожному кроці в проміжку від 15 до 145 міст з кроком 10 міст. Вхідні параметри кожного генетичного алгоритму були однакові: розмір популяції — 1000 хромосом, ймовірність кросинговеру — 80%, ймовірність мутації — 2%, зупинка алгоритму відбувалася через 300 ітерацій. Кількість міст була також однаковою для кожного алгоритму, їхнє розташування на площині кожної ітерації було згенероване випадково.

Результати роботи паралельного генетичного алгоритму подано в таблиці 1.

Таблиця 1. Результати роботи паралельного генетичного алгоритму

Кількість міст	Довжина шляху	Час роботи (*мс*)
15	675,00	4907
25	904,51	6693
35	1218,92	9645
45	1378,06	10956
55	1570,68	12065
65	2163,21	15733
75	2106,91	17889
85	2639,37	20009
95	3421,04	20680
105	3869,60	21338
115	3992,80	23273
125	5117,01	24358
135	5496,20	33157
145	6122,91	34370

Відповідно також було проведено обчислювальний експеримент для паралельного генетичного алгоритму на базі теоретичної моделі “зірка”, запропонованого авторами [4] і модифікованого за допомогою використання нестандартного оператора генетичного алгоритму, а саме SBX-кросинговеру. Результати, отримані як наслідок роботи такого алгоритму, подано в таблиці 2.

Відповідно до отриманих результатів було побудовано графіки середньої довжини знайденого розв’язку (рисунок 1) і середнього часу, за який розв’язок було знайдено (рисунок 2).

Таблиця 2. Результати роботи модифікованого паралельного генетичного алгоритму

Кількість міст	Довжина шляху	Час роботи (*мс*)
15	675,000	4858
25	904,510	6559
35	1206,731	9751
45	1364,279	10846
55	1554,973	12186
65	2141,578	15769
75	2085,841	17710
85	2612,976	19609
95	3386,830	20473
105	3830,904	20271

115	3952,872	22342
125	5065,840	24114
135	5441,238	31168
145	6061,681	35401

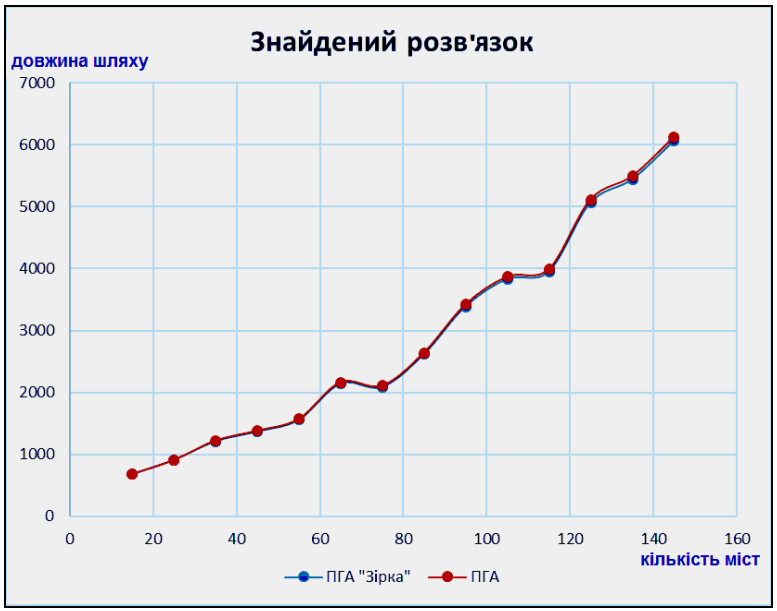


Рисунок 1. Залежність довжини шляху від кількості міст

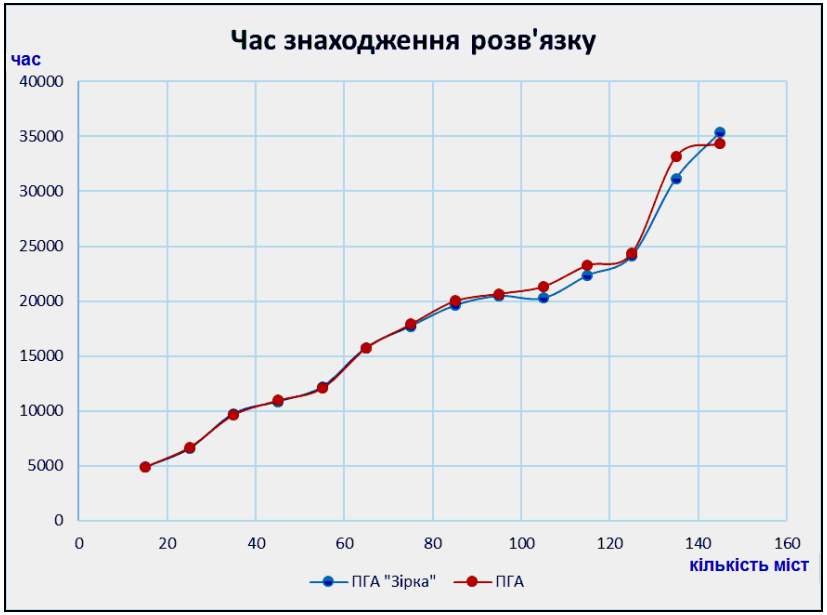


Рисунок 2. Залежність часу знаходження розв’язку від кількості міст

Потрібно зазначити, що модифікований генетичний алгоритм має свою специфіку роботи.

Це виражається в тому, що, по-перше, вдосконалена теоретична модель має певні переваги над звичайним паралельним генетичним алгоритмом. Перш за все вдосконалена модель надає можливість проводити ґрунтовні дослідження з метою оптимізації, можливість використання й випробування різних нестандартних операторів генетичних алгоритмів, можливості легкої оптимізації під потреби користувача, під кількість ядер на обчислювальному пристрої,

адаптації під мережеву чи GRID-інфраструктуру, можливість виконання й використання для хмарних обчислень.

По-друге, в результаті випробувань модифікованого паралельного генетичного алгоритму досягнуто таких результатів:

— точність розв’язку, отриманого в результаті випробування на тестовій задачі комівояжера, принаймні не гірша за точність результату, отриманого класичною реалізацією паралельного генетичного алгоритму;

— час, потрібний для знаходження розв’язку, у випадку модифікованого алгоритму в більшості випадків менший, ніж час, необхідний для розрахунків за класичною варіацією такого алгоритму. Це досягнуто за рахунок використання нестандартного кроссинговеру SBX, внаслідок роботи якого популяції менше схильні сходитися до локального оптимуму.

Література

1. Еремеев А.В. Генетические алгоритмы и оптимизация: Учеб. пособие / А.В. Еремеев. — Омск: ОмГУ, 2008. — 48 с.

2. Панченко Т.В. Генетические алгоритмы / Т.В. Панченко; под ред. Ю.Ю. Тарасевича. — Астрахань: ИД “Астраханский университет”, 2007. — 87 с/

3. Витковски Т. Исследование переменных и параметров генетического алгоритма для планирования производства / Т. Витковски, С. Эльзвай, А. Антчак // Проблемы управления и информатики. — 2004. —№1. —С.136-144.

4. Курейчик В.М. Параллельный генетический алгоритм. Модели и проблемы построения / В.М. Курейчик, Д.С. Кныш. //

Интегрированные модели и мягкие вычисления в искусственном интеллекте: Сб. науч. трудов V Междун. науч.-пр. конф. — М.: Физматлит, 2009. — С.41-52.

Кублій Л.І., Костенко О.П.

Проблеми і завдання взаємодії MATLAB і C#

На практиці в програмуванні в багатьох випадках виникає необхідність створення програмного забезпечення, яке розв'язує складну математичну задачу. Це передбачає використання математичних процедур, які не входять у набір стандартних засобів системи програмування. Розвиток інформаційних технологій і продуктивності сучасних електронно-обчислювальних машин зростає в дуже швидкому темпі. Ускладнюються технології, збільшуються обчислювальні можливості комп'ютерів як домашнього, так і промислового використання. Як наслідок відбувається розширення кола можливих задач, розв'язок яких покладають на електронно-обчислювальні машини.

Завдяки цьому забезпечується поява спеціалізованих інформаційних систем або окремих додатків для забезпечення збору, обробки й надання потрібних даних у контексті розв'язуваної проблеми або автоматизованого бізнес-процесу. Життєвий цикл таких систем може бути різним і залежить від сфери застосування, а також масштабу реалізації. Однак в динаміці побудови подібних систем можна виділити деякі загальні риси — з покоління в покоління ускладнюються алгоритми роботи з інформаційними потоками, а також зростає обсяг даних, які аналізуються та обробляються інформаційною системою.

Сучасні інформаційні системи в загальному випадку мають модульну побудову (рисунок 1), причому реалізація окремих складових частин не завжди здійснюється одним розробником.

Наприклад, реалізація серверів баз даних і СУБД, за допомогою яких будується сховище даних у багатьох інформаційних системах, як правило, здійснюється на основі вже розроблених

технологій компаній Microsoft, IBM, ORACLE та інших. Модульна структура забезпечує багато переваг, таких як: гнучкість розробки, масштабованість, ефективне використання ресурсів, конвеєрна обробка даних. Виходячи з цього, процес обробки даних за складними і ресурсномісткими алгоритмами доцільно виносити в окремий структурований блок. Дане рішення дає можливість оптимізувати роботу інформаційної системи в цілому. Також реалізація подібного блоку обчислень може здійснюватися стороннім розробником у разі паралельної розробки інформаційної системи. Можлива ситуація, коли частина підсистем і обчислювальних блоків вже реалізована. У даному випадку виникає проблема компіляції даних структурних одиниць в єдине ціле, або ефективне впровадження у вже існуючу інформаційну систему.

Рисунок 1. Структура інформаційної системи як сукупність підсистем [1]

Отже, при розв'язуванні задачі, коли системі програмування бракує стандартних засобів для спрощення процесу створення системи, можна піти багатьма шляхами. Або скористатися засобами будь-якого математичного пакету, наприклад, MATLAB і розв'язати математичну задачу, не створюючи ніякого спеціального програмного забезпечення, написавши код програми мовою пакету (MATLAB розв'язує задачі в режимі інтерпретації m-коду програми). Або самостійно написати весь програмний код, включаючи відповідні математичні процедури, скориставшись будь-якою мовою програмування (наприклад: C, C++, C#). Або шукати (наприклад, в Інтернеті) відповідну математичну бібліотеку, сумісну з середовищем програмування.

Пакет прикладних програм MatLab (MATrix LABoratory) фірми Math Works Inc. — це високопродуктивний математичний пакет для наукових і інженерних розрахунків, який дає можливість виконувати обчислення, візуалізацію результатів, програмування у зручній середовищі розробки.

Пакет MATLAB має унікальну колекцію реалізацій сучасних обчислювальних методів: матричні і логічні оператори, елементарні і спеціальні функції, поліноміальну арифметику, пошук коренів нелінійних алгебраїчних рівнянь, обчислення квадратур, оптимізацію функцій кількох змінних, одновимірну й багатовимірну інтерполяцію, аналітичні розрахунки та багато іншого. За допомогою пакету MATLAB можна створювати двовимірні й тривимірні графіки, виконувати візуальний аналіз даних. Пакет має інтерактивне середовище програмування, його мова програмування наближена до звичайної математичної нотації, має більш ніж 1000 вбудованих математичних функцій, підтримує роботу з текстовими і двійковими файлами, можна використовувати програми, написані мовами C, C++ та іншими. Засоби побудови графічного інтерфейсу користувача (GUI) полегшують взаємодію користувача з системою.

Існує три варіанти реалізації взаємодії між MATLAB і C#.

Перший — це завантажити і встановити збірку MATLAB Compiler [2], яка дає можливість обмінюватися програмами на базі MATLAB як окремими додатками. Використовуючи разом з MATLAB Compiler SDK, можна програми MATLAB інтегрувати з іншими мовами програмування. Усі додатки, створені з MATLAB Compiler, використовують MATLAB Runtime-бібліотеку, що дає можливість без додаткових ліцензійних відрахувань розробляти програмне забезпечення для користувачів, які не потребують MATLAB. MATLAB Compiler конвертує програми MATLAB в автономні програми. Щоб скористатися MATLAB Compiler, треба налаштувати з'єднання через командний рядок, написати потрібний код, скомпілювати і запустити на виконання.

Другий — це створити m-функцію, скомпілювати її в dll-бібліотеку, вказавши необхідні параметри, потім підключити до Visual Studio в проект, написаний мовою C#. Це не дуже зручний спосіб, бо у випадку помилок у коді m-функції після виправлення помилки доведеться перекомпілювати файл і повторно додати його у проект. Оскільки досить багато часу займає процес компілювання коду, написаного в MATLAB, то виправлення всіх помилок буде тривалим.

Третій — MATLAB-Deploy toolbox [3] — це пряма активізація MATLAB з активізації вікна мови C#. Для побудови бібліотек dll треба попередньо налаштувати компонувальник MATLAB, який на основі середовища MCR і встановленого в системі компілятора мови C++ дає можливість створювати окремі додатки MATLAB або компоненти DLL.

Реалізація динамічних бібліотек MATLAB здійснюється на основі функцій, написаних однойменною мовою і оформлених в окремі m-файли. Результатом роботи Deploy toolbox в даному випадку є файл *.dll, який реалізує необхідний клас для .NET. Методами даного класу забезпечується функціонування функцій MATLAB (функції можуть реалізовувати практично весь потенціал MATLAB залежно від контексту завдання). Для використання бібліотеки dll, реалізованої за допомогою MATLAB Deploy toolbox треба зробити

посилання на даний модуль з проекту .NET, а також підключити модуль MWArray.dll, який відповідає за обмін інформацією між компонентом MATLAB і середовищем .NET. При розробці кінцевих додатків мовою C # для .NET треба також використовувати відповідні простори імен, для використання підключених бібліотек у проекті. Для коректної інтеграції потрібні:

— простір імен, однойменний з dll-модулем, який реалізує потрібні m-функції;

— простір імен MathWorks.MATLAB.NET.Utility для коректної інтеграції;

— простір імен MathWorks.MATLAB.NET.Arrays для формування інфопотоків між модулями системи.

Дотримання вказаних вимог забезпечує повноцінну взаємодію MATLAB і .NET

Написана m-функція зберігається в каталозі MATLAB, а до створеного проекту і написаного коду C# через вікно Reference Manager підключається бібліотека. У цьому вікні треба вибрати пункт COM і підпункт Type Libraries. У списку бібліотек треба знайти бібліотеку MATLAB Application Type Library і відмітити її. Таким чином при запуску проекту Visual Studio також буде запускатися командний рядок MATLAB. Тому саме цей спосіб є найповільнішим. Але при цьому є можливість відразу виправити помилки, які можуть виникнути, не витрачаючи час на перекомпіляцію файлів і додавання бібліотек.

Результатом встановленої взаємодії буде програмний продукт, у якому недоліки C# в математичних обчисленнях будуть ліквідовані завдяки MATLAB і в свою чергу слабкості користувацького інтерфейсу MATLAB будуть усунені C#-частиною.

Література:

1. Информационные системы, базы знаний, базы данных [Електронний ресурс]. — Режим доступу:

sites.google.com/site/inftech11/home/sam/ informacionnye-sistemy-bazy-znanij-bazy-dannyh

2. MATLAB Compiler. Создание автономных приложений из программ MATLAB [Електронний ресурс]. — Режим доступу: matlab.ru/products/matlab-compiler

3. MATLAB Builder NE (for Microsoft .NET Framework) [Електронний ресурс]. — Режим доступу: matlab.ru/products/matlab-builder-ne

Кублій Л.І., Семенчук І.О.

Оцінка перспективності районів кроводач на основі аналізу статистики захворюваності населення

Необхідність збору крові завжди була і буде актуальною в медицині, оскільки завдяки цьому врятовано чимало людських життів. Збільшення ефективності роботи навіть одного із відділів станції дасть змогу збільшити обсяги отриманої крові. На даний час виїзні групи станції переливання крові виконують збори крові на прилеглих територіях за визначеним графіком, не враховуючи стан здоров'я донорів і можливість відбракування крові донорів на заданій території. Тому значна частина виїзних операцій приносить малу кількість зібраної крові порівняно з запланованою. Необхідність прогнозування захворюваності населення є однією із найбільших проблем, оскільки завдяки йому можливо оптимізувати чимало процесів і акцій профілактики стану здоров'я населення.

Оптимізація роботи виїзних груп, дасть можливість економити людські ресурси й ефективно використовувати час для збору крові. Правильна оцінка районів здачі крові на основі прогнозу захворюваності дає змогу спланувати виїзди робочих груп станцій в найбільш ефективному порядку. Оптимізація роботи бригади збору крові є одним з факторів, який може збільшити продуктивність роботи

станцій переливання крові. Оптимізації можливо досягнути за рахунок прогнозування перспективності донорів на територіальних районах.

Створене програмне забезпечення визначає донорів, які можуть здати кров. При цьому використовується наявна в базі даних інформація про них, враховується дата останньої дачі крові, перевіряється, чи не проживає донор в зоні карантину, з'ясовується, з якою метою донор звертався в певний період до лікувального закладу, чи знаходиться він в групі ризику. На основі інтенсивності здачі крові для цих донорів прогнозується найкращий виїзний район.

Розрахунок оцінки перспективності району виконується за допомогою рівняння множинної лінійної регресії. Важливим етапом побудови рівняння множинної регресії є відбір і подальше включення в нього факторних ознак. Велику роль відіграє кількість факторних ознак — чим більше факторних ознак включено в рівняння, тим більш реально воно описує досліджуване явище. З іншого боку, модель з понад ста факторними ознаками складно описується і вимагає великих часових затрат при побудові.

У розробленій моделі для обчислення оцінки використовуються такі факторні ознаки:

— кількість задовільних на даний час донорів;

— прогнозована кількість хворих донорів;

— середня інтенсивність здачі крові;

— середня доза крові для обраних донорів.

Узагальнена багатофакторна лінійна регресійна модель має вигляд:

$$y = a_0 + a_1 x_1 + a_2 x_2 + ... + a_p x_p + e,$$

де y — залежна змінна, $x_1, x_2, \ldots, x_p$ — незалежні змінні, $a_0, a_1, \ldots, a_p$ — параметри моделі, які потрібно оцінити, e — не спостережувана випадкова величина [1].

Значення залежної змінної залежить від цілі виїзду. Таким чином користувач надає пріоритет виїзду (обсяг крові, кількість донорів). Незалежні змінні представлені факторами.

Розроблена модель, яка враховує вказані основні фактори, забезпечує найкращий прогноз для вибору району для виїзної групи. Залежна змінна представляється як оцінка району.

Прогнозування кількості хворих донорів виконується на основі наявної в базі даних системи інформації про донорів. Необхідною умовою якісного прогнозу є якісні вхідні дані [2].

Для прогнозування кількості хворих донорів (як одного з факторів розробленої багатофакторної моделі) можна скористатися різними моделями. Так, дослідники Українського інституту стратегічних досліджень МОЗУ пропонують проводити прогноз захворюваності населення на середньострокову перспективу окремо залежно від класу хвороб на основі статистичних даних за роками; оскільки абсолютні прирости динамічних рядів захворюваності за основними класами хвороб дорослого населення України практично стабільні, то математичний апарат було обмежено лише лінійною однофакторною залежністю $y_t = at + b$ [3]. Для прогнозування порогових значень захворюваності на грип і ГРВІ теж запропоновано лінійну модель [4]. Проте деякі захворювання мають періодичні

епідемічні поширення і відрізняються сезонним характером. Також завдяки досягненням медицини і розв'язанню соціально-економічних проблем темпи прояву деяких захворювань можуть зменшуватися [5]. Внаслідок цього лінійна модель не завжди буде правильно відображати характер зміни захворюваності [6]. Крім того, якщо досліджувати не класи, а конкретні захворювання, то лінійної моделі для одержання достовірного прогнозу не завжди буде достатньо [7].

З метою підвищення точності прогнозу вхідні дані мають бути проаналізовані на наявність тенденції. Для виявлення наявності тенденції в розробленій системі було реалізовано метод кумулятивного t-критерію [8]. Аналіз і моделювання тенденції часового ряду доцільно починати з виявлення наявності тенденції в цілому. Кумулятивний t-критерій дає можливість визначити наявність не тільки самої тенденції, але й її математичного виразу — тренда.

Розрахункове значення кумулятивного t-критерію визначається як відношення накопиченої суми квадратів відхилень емпіричних значень рівнів часового ряду від середнього значення самих відхилень за формулою:

$$T_p = \frac{\sum_{i=1}^{n} z_i^{\,2}}{\sigma_y^2},$$

де z_i — накопичена сума відхилень емпіричних значень від середнього рівня вихідного ряду; σ_y^2 — загальна сума квадратів

відхилень, яка визначається за формулою $\sigma_y^2 = \sum_{i-1}^{n} (y_i - y_c)^2$ (тут n — довжина часового ряду (кількість рівнів); y_i — вихідні значення ознаки; y_c — середній рівень вихідного ряду динаміки).

Розрахункові значення кумулятивного t-критерію порівнюються з критичними при заданому рівні значущості α. Критичні значення беруться з таблиці критичних значень розподілу Стьюдента.

Гіпотеза перевіряється на основі t-критерію Стьюдента: якщо розрахункове значення T_p перевищує критичне значення критерію ($T_{кр}$), то гіпотеза про відсутність тренду відхиляється, отже, в вихідному часовому ряду існує тенденція, що описується трендом. В іншому випадку, якщо $T_p < T_{кр}$, визнається відсутність тенденції в ряду динаміки.

Оскільки прогнозування кількості хворих донорів здійснюється на незначний період (на один крок), то для прогнозування досить використати метод експоненційного згладжування [6]. Експоненційне згладжування дає можливість отримати оцінку, яка характеризує тенденцію на момент останнього спостереження. Модель має вигляд: $Y_i = \beta y_{i-1} + (1-\beta) Y_{i-1}$, де початковим значенням береться y_1, значення параметра згладжування β ($\beta \in [0;1]$) визначається простим перебором з кроком 0,001 так,

щоб порівняно з емпіричними даними прогноз був найточнішим (критерієм береться мінімум суми квадратів залишків).

У даному дослідженні поєднано прогнозування перспективності району на основі рівняння множинної лінійної регресії з прогнозуванням кількість хворих донорів (перспективності донорів) на основі визначення наявності тренду і застосування екпоненційного згладжування.

Розроблена і програмно реалізована модель, яка враховує вказані основні фактори, забезпечує найкращу оцінку району для виїзної групи, дає можливість максимально ефективно використовувати людські ресурси станції.

Література

1. Дрейпер Н. Прикладной регрессионный анализ: В 2-х кн. Книга 1 / Н. Дрейпер, Г. Смит. — М.: Финансы и статистика, 1986. — 366 с.

2. Ефимова Н.В. Опыт использования искусственных нейронных сетей при прогнозировании заболеваемости / Н. В. Ефимова, А. Ю. Горнов, Т. С. Зароднюк. // Экология человека. — 2010. — № 3. —С. 3-7.

3. Визначення проблемних регіонів України за станом здоров'я населення (2009) [Електронний ресурс]. — Режим доступу: uiph.kiev.ua/dawnload/2.pdf

4. МР 3.1.2.0005-10 Методика расчета эпидемических порогов по гриппу и острым респираторным вирусным инфекциям по

субъектам Российской Федерации [Електронний ресурс]. — Режим доступу: docs.cntd.ru/document/1200086225

5. Боев Б. В., Макаров В. В. Компьютерное моделирование и прогнозирование эпидемий птичьего гриппа // Ветеринарная патология. — 2005. — № 3. — С. 49-58.

6. Бокс Д., Дженкинс Г. Анализ временных рядов: прогноз и управление. Выпуск 1. — М.: Мир, 1974. — 406 с.

7. Бородулин А.И. Статистическая модель эпидемического процесса / А.И. Бородулин, Б.М. Десятков, А.Н. Шабанов, А. А. Ярыгин // Сибирский журнал индустриальной математики. — 2007. — Т. X, № 2 (30). — С. 23-30.

8. Шмойлова Р.А. Теория статистики / Р.А. Шмойлова, В.Г. Минашкин, Н.А. Садовникова, Е.Б. Шувалова. — М.: Финансы и статистика, 2004. — 656 с.

Кублій Л.І., Тобілко А.О.

Захист інформації в комплексі моделювання гідроакустичних процесів

У програмних система, які працюють із залежними від зовнішніх факторів масивами даних, важливу роль відіграє захист як на фізичному, так і на програмному рівнях. Гарантування безпеки цілої системи — досить складне завдання. Якщо систему розбити на логічні модулі й застосувати конфігурації безпеки окремо для кожного з них, то розв'язання цього завдання набагато спроститься.

Для забезпечення комп'ютерної безпеки при моделюванні гідроакустичних процесів [1] важливим є надання зручного й гнучкого інтерфейсу користувача. Це дасть змогу коректніше налаштувати безпеку системи відповідно до вимог користувача.

Розроблена система Hydroacoustic System Security збирає необхідну інформацію про систему, з якою буде працювати, і надає широкий спектр конфігурації для підтримки її безпеки. У розробленій системі передбачено можливість організації рівнів користувачів. За замовчуванням, у системі існує лише один рівень "Адміністратор". Даний рівень надає необмежений доступ до будь-якої конфігурації. Рівень "Адміністратор" дає змогу створювати нові групи й визначати набір прав, доступних для користувачів даної групи. Права контролюють доступ до певних даних системи (наприклад, дані за певний період часу) й певні дії користувача (наприклад, редагування чи видалення даних) [2]. Такий підхід дасть можливість правильно розподілити обов'язки між користувачами й побудувати певну ієрархію доступу.

Комплекс безпеки являє собою сервер, який має кілька режимів роботи. Програмний продукт може бути запущений для

одного користувача локально на його робочій машині. Також система може бути піднята як сервер для загального користування. Кількість користувачів може бути фіксованою до певної кількості. Це обмеження може бути вимкнене головним користувачем.

Перш ніж перейти до опису модулів системи, потрібно зрозуміти, як дані для моделювання гідроакустичних процесів отримуються й коригуються.

Отримання даних здійснюється за допомогою дистанційно-керованого підводного апарату (Remotely Operated underwater Vehicle, ROV), який переміщається в товщі води і (або) по дну й керується командою пілотів-техніків з борту надводного судна [3]. Апарат оснащено спеціальним обладнанням, приладами й інструментами для наукових досліджень, пошукових і аварійно-рятувальних операцій, виробничих робіт під водою, а також у військових цілях.

У цей клас підводних апаратів входять різноманітні за призначенням і конструкцією прив'язні, буксирувані й автономні телекеровані підводні апарати. Найскладнішим при роботі з даними апаратами є відстеження місцезнаходження. Коригування цих даних здійснюється за допомогою USBL tracking system — системи для відстеження підводних об'єктів, які змінюють своє положення [3].

Метод USBL (англ. Ultra-short baseline) є методом підводного акустичного позиціонування [4]. Повна система USBL складається з трансивера, встановленого під кораблем, а також транспондера або респондента на морському дні, або на поверхні апарату ROV. Комп'ютер або "верхній блок" використовується для обчислення позиції, виміряної трансивером.

Акустичний імпульс передається трансивером і виявляється підземним транспондером, який відповідає своїм акустичним імпульсом. Цей зворотний імпульс виявляється корабельним приймачем. Час від передачі початкового акустичного імпульсу до виявлення відповіді вимірюється системою USBL і перетворюється в діапазон (рисунок 1).

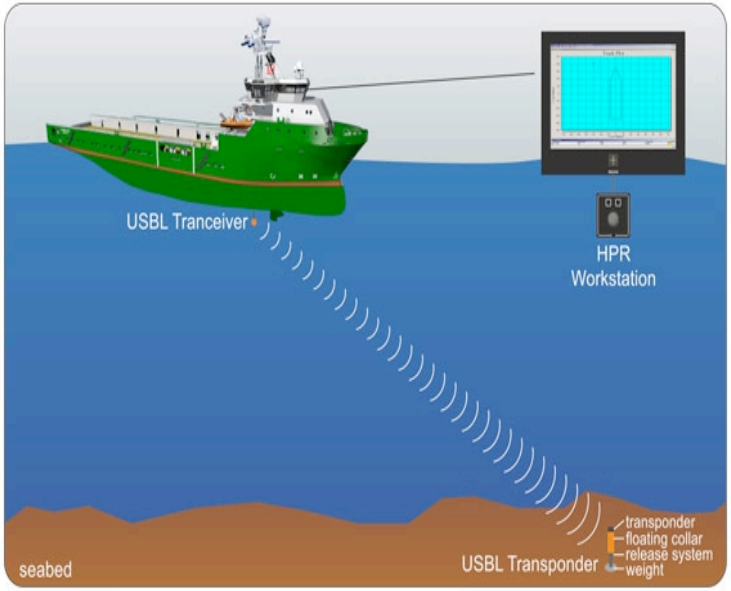


Рисунок 1. Принцип роботи системи USBL [4]

Зрозумівши принципи отримання й коригування даних, можна бути впевненим, що дані, отримані від ROV з коригуванням локації за допомогою USBL, будуть коректні й отримані безпечним шляхом. Залишається гарантувати їхню безпеку при збереженні й подальшому аналізі.

Складовими системи Hydroacoustic System Security є модулі, які надають конфігурації захист проти різних типів небезпек. Користувач самостійно вирішує, які модулі мають бути ввімкненими і які модулі справді потрібні системі. Кожен модуль містить опис детального переліку дій, які він виконує для забезпечення безпеки системи. Це слугує вбудованою документацією і користувачеві не потрібно шукати її в іншому місці.

Модуль фільтрування інформації дає можливість валідувати вхідні дані й відхиляти непотрібні на початкових стадіях моделювання гідроакустичних процесів. Він також бере участь у формуванні детального звіту роботи системи з використанням користувацьких фільтрів.

Важливим модулем розробленої системи є модуль моніторингу стану підконтрольної системи (в нашому випадку — системи моделювання гідроакустичних процесів). Це детально спроектована підсистема, яка ретельно слідкує за змінами і відразу повідомляє про мінімальні відхилення в її роботі або проблеми, які можуть виникнути під час її роботи.

Для написання модулів системи вибрано мову програмування Java, для роботи з базою даних — систему керування базами даних Microsoft SQL Server Express Edition, для забезпечення віддаленого підключення — мережеві технології Spring Boot. Вся розробка здійснювалась в середовищі JetBrains IDEA [5]. Для розробки графічних компонентів було обрано фреймворк Angular JS в поєднанні з мовою програмування TypeScript [6]. Для підключення додаткових бібліотек і сторонніх компонентів використано менеджер пакунків npm і систему bower керування пакунками на боці клієнта [7].

Розроблений програмний продукт враховує особливості одержання, зберігання й використання даних при моделюванні гідроакустичних процесів.

Література:

1. Hydroacoustics: Lakes and Reservoirs [Електронний ресурс]. — Режим доступу: stateofthesalmon.org/fieldprotocols/downloads/SFPH_p5.pdf

2. Web Application Security. A Beginner’s Guide [Електронний ресурс]. — Режим доступу: https://books.google.com.ua/books//w_app_sec.html

3. Емельянов В.В. Теория и практика моделирования / В.В. Емельянов, В.В. Курейчик, В.М. Курейчик. — М: Физматлит, 2003. — 432 с.

4. Underwater Positioning System [Електронний ресурс]. — Режим доступу: http://www.fcrov-piloting.com/company-equipment/usbl-underwater-positioning-system.aspx

5. Документація JetBrains IDEA [Електронний ресурс]. — Режим доступу: https://www.jetbrains.com/idea/docs

6. Гладков Л.А. Защита информационных систем / Л.А. Гладков, В.В. Курейчик, В.М. Курейчик. — 2-е изд. — М.: Физматлит, 2006. — 320 с.

7. Скобцов Ю.А. Основы эволюционных вычислений / Ю.А. Скобцов. — Донецк: ДонНТУ, 2008. — 326 с.

Левченко Л.О. Орел Д.С.

Інтеграція PDM та CAD систем

CAD (Computer-Aided Design) – це організаційно-технічна система для автоматизації процесу проектування, основними цілями якої є скорочення трудомісткості проектування та планування, скорочення собівартості виготовлення, зменшення затрат на експлуатацію тощо [1].

З метою забезпечення ефективного управління інформацією про технічний об’єкт використовується PDM-система (Product Data Management). Вона являє собою організаційно-технічну систему, що забезпечує управління всією інформацією про виріб. При цьому в якості виробів можуть розглядатися різні складні технічні об'єкти (кораблі і автомобілі, літаки і ракети, комп'ютерні мережі та ін.).

Впровадження PDM-системи на підприємстві дозволяє створити єдине інтегроване середовище управління інженерними даними та проектами, яке забезпечує [2]:

— колективну роботу проектно-конструкторських підрозділів підприємства над проектами з розмежуванням прав доступу до його складових частин;

— управління інженерними даними, технічними даними, та різною інформацією, пов’язаною з виробом.

— надійне зберігання і швидкий пошук інформації в електронних архівах;

— максимальне використання в нових розробках відпрацьованих і перевірених технічних рішень, що зберігаються в базі даних (БД);

— виключення помилок за рахунок усунення декількох джерел зберігання однієї і тієї ж інформації.

Будь-який підхід до керування професійними даними у виробництві вимагає інтеграції CAD та PDM. Технічно складні продукти не можуть бути оперативно оброблені та виведені на ринок своєчасно без PDM. Знання про продукти компанії в основному містяться в його моделях CAD, а також у відповідних даних та документах. Цей проект та інженерні знання повинні бути доступними на всій території підприємства. За допомогою PDM працівники всіх підрозділів можуть отримати доступ до будь-якої цієї інформації з відповідними дозволами. Всі дані, незалежно від того, є це моделі CAD, креслення, виготовлення та складання креслень, основні дані елемента, векселі матеріалів або документи з зображеннями, текстовими або розрахунковими таблицями, надійно зберігаються у сховищах даних системи PDM незалежно від типу CAD системи, з якої вони приходять. Це гарантує, що всі дані про продукт завжди оновлюються та узгоджуються між собою та між відділеннями. Перевагами PDM у керуванні даними CAD є наступні:

— безпечне зберігання та захищений доступ до інформації про продукт;

— інформація про продукт може зберігатися незалежно від основних систем авторизації, що використовуються в полях електроніки, Office та електронної пошти;

— інформація консолідується для створення файлів клієнта, машини та життєвого циклу;

— автоматичне надання дійсних креслень для виробничих цілей;

— автоматична синхронізація основних даних елемента та мікропроцесорів між проектуванням, розробкою та планування виробництва, виробництвом та сервісом.

Деякі розробники CAD-систем надають доступ до власно-створеної PDM-системи. Проте з використанням декількох CAD-систем виникає проблема у спільному збереженні всіх даних, що може призвести до помилок при зберіганні однієї і тієї ж інформації.

Підхід з використання однієї PDM-системи для збереження даних з різних CAD-систем має суттєві переваги над іншими. Інтеграція PDM-системи в різні CAD-системи допоможе членам конструкторського підрозділу більш ефективно працювати з CAD-системою і в ній мати можливість зберігати результат своєї роботи в PDM-систему.

Таким чином, інтеграція PDM та CAD-систем необхідна для скорочення термінів виходу нового виробу на ринок за рахунок зменшення часу на розробку технічної документації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

3. Phiri, Michael. Information Technology in Construction Design. London, 1999. 228 p.

4. Щеглов Д. К. Методы и средства выбора системы управления проектными данными конструкций летательных аппаратов [Електронний ресурс] // СПб.: ГНЦ РФ ЦНИИ «Электроприбор». – 2009. – Режим доступу до ресурсу: http://www.elektropribor.spb.ru/cnf/kmu10/text/s3-032.rar

Мажара О.О., Витвицький Д. А.

Формування навчальної вибірки в задачі класифікації музичних інструментів

Задача класифікації музичних інструментів має широке практичне застосування в прикладному програмному забезпеченні обробки аудіозаписів. Так, наприклад, вона може вирішуватися в рамках створення підсистем для формування баз даних аудіоматеріалів та в рамках рекомендаційних систем. Коректна класифікація музичних інструментів залежить від наступних виокремлених процесів [1]:

1. Вибору музичних інструментів для розпізнавання
2. Вибору характеристик
3. Математичного представлення характеристик в наборі даних
4. Зменшення розмірності набору даних
5. Використання методів класифікації

Перші чотири етапи відповідають за формування значимих характеристик та навчальної вибірки, на основі яких буде проводитися класифікація, та відіграють вирішальну роль в подальшій ефективності роботи алгоритму. Експерименти щодо створення класифікаторів музичних підтверджують, складність визначення оптимальної комбінації характеристик та методів розпізнавання [1, 2]. Остаточна точність будь-якого класифікатора явно залежить від кількості та типу обраних інструментів, а також від характеристик та підготовки прикладі. Це зумовлює актуальність аналізу сучасних репозиторіїв (бібліотек) записів музичних інструментів з метою формування навчальної вибірки для нейронної мережі розпізнавання музичних інструментів.

В даному дослідженні було проведено огляд як комерційних репозиторіїв музичних записів так і тих, які поширюються за вільною ліцензією. Для аналізу було також обрано такі характеристики як типи представлених музичних інструментів та формати даних (Таблиця 1).

Таблиця 1. Бібліотеки зразків музичних інструментів

Назва	Опис	Розробник	Типи музичних інструментів	Поширення за вільною ліцензією	Посилання
McGill University Master Samples (MUMS)	Бібліотека прикладів, представлених в WAV форматі	Opolko & Wapnick	Струнні, клавішні, духові, ударні	Так	[3]
Musical instrument sample (MIS)	Бібліотека музичних інструментів Університету Айови, представлених у форматі AIFF	Fritts	Струнні, клавішні (фортеп'яно), духові, ударні	Так	[4]
Vienna Symphonic Library	Бібліотека віртуальні інструменті в та відтворення	Vienna Symphonic Library GmbH	Струнні, смичкові, арфи, духові,	Ні	[5]

	акустик знаменитих концертів		латунь, саксофон, трубний орган, перкусія, хори		
London Philharmonic Orchestra dataset	Записи у виконанні оркестру філармонії, представлених у форматі mp3	London Philharmonic Orchestra	Струнні, клавішні, духові, ударні	Так	[6]
Free sound	Велика бібліотека музичних записів, представлених у різних форматах	Free sound community	Струнні, клавішні, духові, ударні	Так	[7]

Серед представлених репозиторіїв історично найбільш широкого застосування в академічній сфері набула бібліотека MUMS [8]. Бібліотека складається з трьох DVD-дисків та містить зразки більшості стандартних класичних, деяких нестандартних класичних та багатьох популярних музичних інструментів. Наразі у бібліотеці 6546 зразків записів, розділених між струнними (2204), клавішними (1595), перкусією (1087), дерев'яними (1197) та мідними духовими (463) сімействами музичних інструментів. Кожна нота з кожного інструменту була записана окремо (44,1 кГц, 24 біт). Для більшості інструментів представлено декілька стилів артикуляції. Як правило,

для кожного приладу є 29 зразків, що означає, що охоплюється не весь діапазон висот інструментів [8] Тривалість зразків переважно становить від 2 до 10 секунд та може значно відрізнятися в межах декількох інструментів. Це може зумовлювати проблеми при використанні темпоральних характеристик, оскільки вони безпосередньо залежать від часу. Бібліотека містить ряд помилок, які підлягають виправленню у нових ревізіях [3]. Крім того, існує ряд труднощів в процедурі доступу до репозиторію, адже його необхідно замовляти на сайті університету з обгунтуванням причин використання.

Віденська симфонічна бібліотека (Vienna Symphonic Library) надає віртуальні інструменти та цифрове відтворення акустики знаменитих концертних залів. З моменту створення було записано та опубліковано більше трьох мільйонів зразків практично всіх інструментів симфонічного оркестру, хору та інших. Це більш терабайт звуку з високою роздільною здатністю та найбільша база зразків, яку коли-небудь створювала одна компанія [5]. В той же час, записи поширюються на комерційній основі, що обмежує можливості їх використання в навчальних цілях.

Бібліотека Musical instrument sample (MIS) була створена в університеті Айови в 1997 році і з того часу доступна для вільного завантаження на його сайті [5]. Записи відбувалися для кожного інструменту нота за нотою на спеціалізованому обладнанні. На разі в репозиторії представлено 23 музичних інструменти. Бібліотеку поділено на два підрозділи відповідно до технології запису зразків: до 2012 та після 2012 року [5]. Недоліком бібліотеки є те, що кожен інструмент необхідно завантажувати окремо та відсутності опису характеристик записів.

Репозиторій Freesound має на меті створити величезну колективну базу даних аудіо фрагментів, зразків, записів, добірок [7]. Наразі користувачі завантажили тисячі записів, які поширюються за

вільною ліцензією Однак створення навчальної вибірки з такого репозиторію потребує кропіткої праці, щодо екстракції необхідних записів та перевірки їх на відповідність вимогам щодо якості.

На основі порівняння архівів зразків записів музичних інструментів навчальної вибірки для подальшого вирішення задачі класифікації було обрано бібліотеку Лондонського симфонічного оркестру. Вона містить у собі тисячі екземплярів мелодій, які поширюються за вільною ліцензією, та широко застосовується для наукових досліджень в області розпізнавання звукових сигналів [6]. В бібліотеці представлено приклади 20 різних інструментів з 4 різних класів – струнних, духових, ударних та клавішних. Зразки охоплюють всі набори тонів в кожній октаві та різній силі звучання (форте та піано). Крім того набори містять зразки різної техніки гри, такі як, наприклад, вібрато та тремоло [6]. Крім того, репозиторій має зручну структуру представлення записів. Користувач має змогу завантажити всі представлені дані або обрати музичний інструмент чи перкусію. В рамках одного музичного інструмента надається змога сортування та фільтрації за обраними характеристиками, такими як: висота звуку, довжина запису, динаміка та артикуляція. Представлена можливість завантажити всі зразки для даного інструменту або обрати лише декілька відповідно до необхідних характеристик.

Таким чином в дослідженні представлено огляд базових репозиторіїв зразків музичних інструментів. Виокремлено значимі характеристики для їх порівняння. На основі проведеного огляду для подальшого формування навчальної вибірки класифікатору музичних інструментів було обґрунтовано вибір бібліотеки Лондонського симфонічного оркестру (London Philharmonic Orchestra dataset).

Література

1. Loughran R. Musical Instrument Identification with Feature Selection Using Evolutionary Methods CLIPS [Електронний ресурс] / Roisin Loughran., The University of Limerick, 2009. – 281 с. — Режим доступу :\ http://ncra.ucd.ie/papers/LoughranThesis.pdf
2. Toghiani-Rizi B. Musical Instrument Recognition Using Their Distinctive Characteristics in Artificial Neural Networks [Електронний ресурс] / B. Toghiani-Rizi, M. Windmark // Cornell University Library. – 2017. – Режим доступу: https://arxiv.org/abs/1705.04971.
3. Opolko F. The McGill University master samples collection on DVD (3 DVDs). / F. Opolko, J. Wapnick. – Квебек, Канада: McGill University, 2006.
4. Fritts L. The University of Iowa Musical Instrument Samples [Електронний ресурс] / Lawrence Fritts // The University of Iowa. – 1997. – Режим доступу до ресурсу: http://theremin.music.uiowa.edu/MIS.html.
5. Vienna Symphonic Library [Електронний ресурс] – Режим доступу: https://www.vsl.co.at/en.
6. L. P. Orchestra. Sound samples. [Електронний ресурс]. – 2016. – Режим доступу до ресурсу: http://www.philharmonia.co.uk/explore/sound_samples.
7. Freesound samples [Електронний ресурс] – Режим доступу: https://freesound.org/.

Eerola T. Instrument library (MUMS) revised [Електронний ресурс] / T. Eerola, R . Ferrer Music Perception. – 2008 – с. 253 - 255 – Режим доступу: https://www.researchgate.net/publication/228092586_Instrument_library_MUMS_revised

Медведєва В. М., Панченко О.О.

Система фільтрації та децимації GPS-даних

Вступ

Кількість пристроїв з інтегрованою підтримкою системи глобального позиціонування (GPS) зростає швидкими темпами. Сьогодні ця технологія застосовується в різноманітних галузях промисловості, наприклад: авіація, сільське господарство, транспортні системи, аварійні системи, геодезія, екологічна захист, відпочинок та багато інших.

Деякі додатки, які застосовуються для визначення місця розташування пристрою, на ринку відносно прості, націлені на визначення грубої позиції користувача. Інші додатки надають послуги для більш точного визначення поточного місцезнаходження пристрою, використовуючи додаткові дані з gps пристроїв, таких як висота та кут повороту.

Відомо, що велика кількість транспорту значний період часу рухається прямолінійно, тому побудова пройденого ними маршруту потребує малу кількість даних. В таких випадках для заощадження трафіку, який використовує пристрій під час передачі даних, необхідно обробляти дані безпосередньо на самому пристрої. Точність самих координат може залежати від безлічі різних чинників, основними з яких зовнішнє середовище та апаратні помилки. Вплив даних помилок можна спостерігати під час моніторингу об'єкта, який перебуває на стоянці, тобто точок з нульовою або близькою до нуля швидкістю. В таких в місцях отримані координати зазвичай не постійні.

Для вирішення даних задач прийнято застосовувати алгоритми фільтрації та децимації навігаційних даних.

Аналіз літературних даних та постановка проблеми

На сьогодні найбільш поширеним алгоритмом передачі поточних GPS координат є алгоритм, що враховує в якості критеріїв швидкість руху об'єкта і кут його повороту. Принцип його роботи можна розглянути на прикладі технології DDF[1] системи LOCARUS. Даний алгоритм включає у себе мінімальний інтервал часу, через який необхідно передати поточні координати об'єкта, що рухається. Перевагою такого підходу є простота реалізації та швидкість обробки даних. До недоліків можна віднести інтервальну передачу. Навіть якщо об'єкт буде рухатись весь час по прямій з мінімальною швидкістю, трекер все одно буде передавати дані. Хоча для побудови маршруту було б достатньо всього двох точок.

Для мінімізації даних, що передаються можна використати математичні методи фільтрації даних, такі як алгоритми згладжування (фільтр Калмана, медіанний фільтр), У разі, коли кількість GPS помилок невелика, застосування різних алгоритмів згладжування дозволяє домогтися високої точності навігаційних даних. У ситуації, коли послідовні похибки складають значну частину траєкторії руху алгоритми згладжування не справляються (низька точність), тому необхідно застосувати методи, які здатні відкидати помилкову інформацію, тобто децимувати помилкові координати. Серед таких алгоритмів можна виділити адаптивний алгоритм обробки потоку навігаційних даних на основі методу діагностичної фільтрації [2]. В якості критерію децимації використовується відстань між двома точками, яка повинна бути не менше порогового значення ε. Радіус кола (сфери в разі використання висоти) відповідає граничному значенню ε і обчислюється на основі точності геопозиціювання, яка залежить від значень факторів втрати точності в поточний момент. До недоліків даного алгоритму можна віднести неврахування особливостей траєкторії руху об'єкта таких як кут його повороту та швидкість, що призводить до спотворень при побудові маршруту під час поворотів.

В даній роботі, в якості основного критерію для децимації gps–даних передбачається використання одного із алгоритмів спрощення полігонального ланцюга [3].

Розробка алгоритму фільтрації та децимації

Через неточність GPS модуля, під час стоянок трекер може передавати хибні дані, які показують рух об’єкта коли насправді він нерухомий. Дана проблема є актуальною для аграрної техніки, яка довгий час може перебувати на одному місці.

Для вирішення наведеної вище проблеми необхідно видалити хибні точки за допомогою попередньої фільтрації. Для тестового набору з мобільного пристрою були отримані 80 точок стандартним алгоритмом передачі GPS даних. Після застосування фільтрації їх кількість зменшилася до 63. Результат роботи показано на рисунку 1.

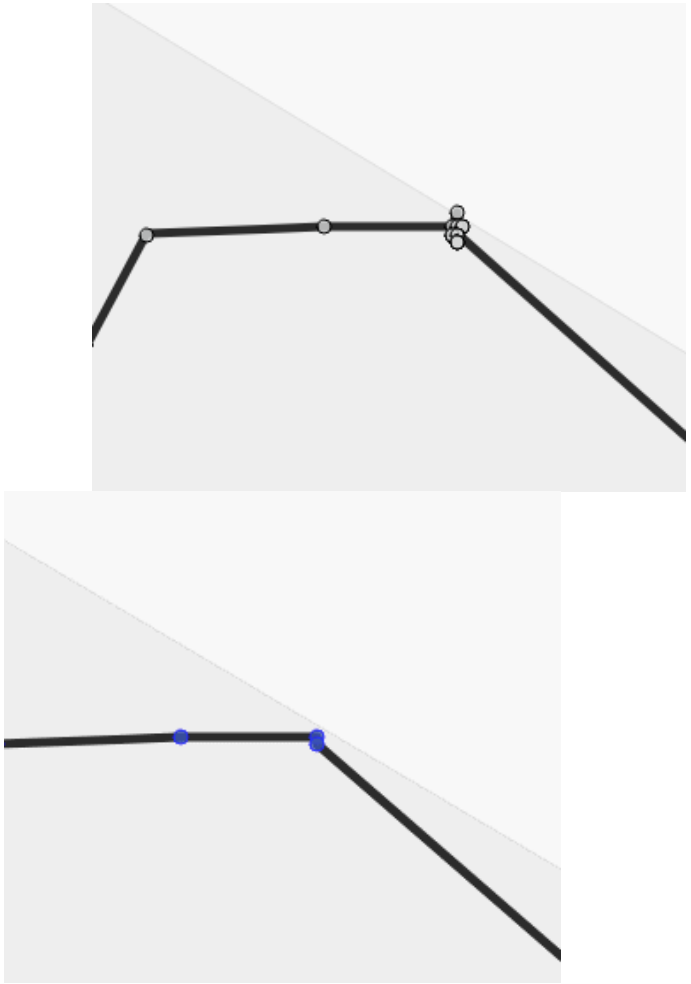


Рисунок 1. Координати об’єкта під час стоянки до та після фільтрації

Далі в якості алгоритму для децимації був застосований алгоритм Дугласа – Пекера. Після його використання кількість точок була скорочена до 30. До недоліків використання такого підходу можна віднести децимацію точок при повороті об’єкта. Також

врахування такого критерію як швидкість руху об’єкта дало б можливість збільшити точність при побудові маршруту, пройденого об’єктом.

Для усунення наведених вище недоліків було вирішено модифікувати алгоритм Дугласа – Пекера таким чином, щоб він враховував додаткові критерії руху об’єкта, такі як кут та швидкість повороту. Для цього алгоритм на кожному кроці починаючи з другої точки бере *i-1* , *i* та *i+1* точки та перевіряє кут нахилу між двома прямими, побудованими через ці 3 точки.

Якщо швидкість в точці *i*, яка є спільна для двох прямих, не рівна нулю та кут нахилу між цими прямими більше 15°, така точка не децимується і потрапляє у результуючий набір.

Після використання модифікованого алгоритму Дугласа – Пекера система отримала 41 із 80 початкових точок. На рисунку 2 можна побачити, що точки, які необхідні для побудови траєкторії руху об’єкта під час повороту збереглися.

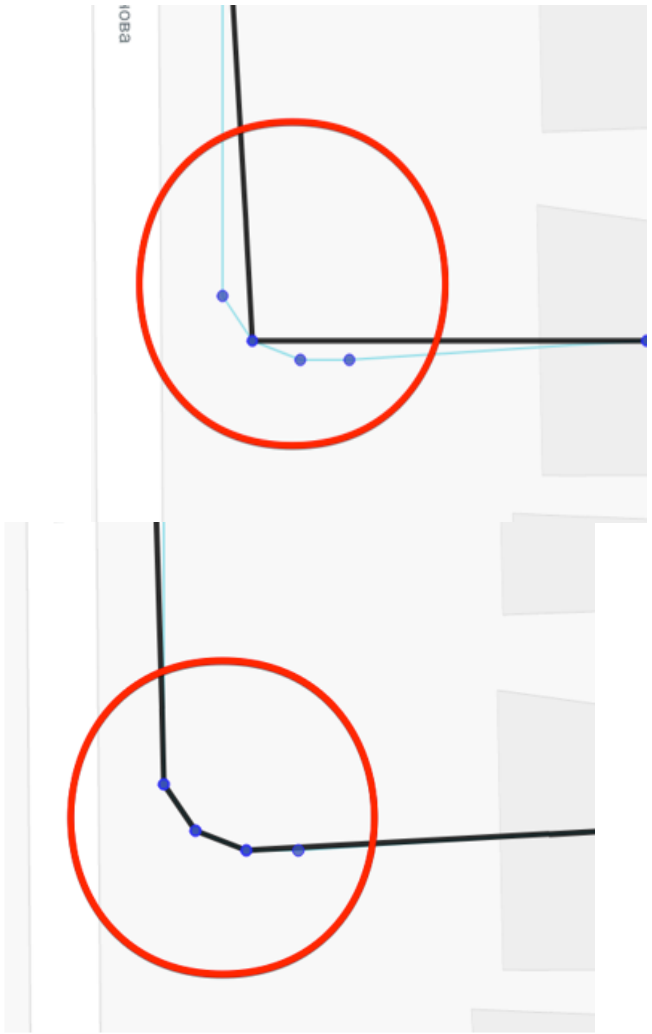


Рисунок 2. Результуюча траєкторія руху об’єкта до та після застосування модифікованого алгоритму Дугласа – Пекера

Таким чином розроблена система дає змогу відфільтровувати фізично неможливі GPS координати, джерелами яких можуть слугувати лінії електропередач, особливості зовнішнього середовища чи апаратні помилки.

Результуючий маршрут який побудовано по точках після фільтрації та децимації можна побачити на рисунку 3.

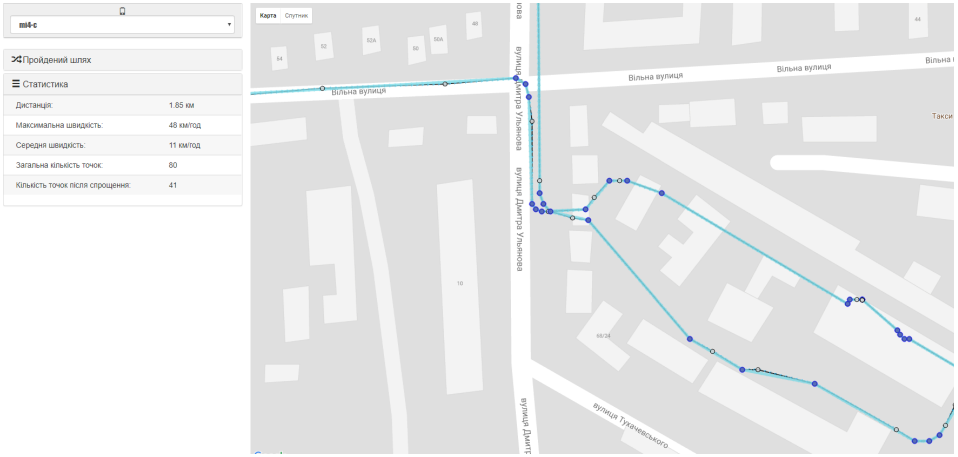


Рисунок 3. Спрощення пройденого шляху з попередньою фільтрацією точок та використанні модифікованого алгоритму Дугласа – Пекера

Висновки

Розроблено гнучкий алгоритм, який дозволяє децимувати навігаційні дані, що передаються GPS пристроєм, за рахунок використання алгоритму Дугласа-Пекера, з урахуванням додаткових критеріїв, серед яких час надходження даних, швидкість об’єкта та його кут повороту. Це дало змогу зберегти точки, необхідні для побудови траєкторії руху під час поворотів. Проведено тестування розробленого програмного продукту, яке показало двократне скорочення інтернет трафіку, який споживає GPS передавач при передачі координат на сервер у порівнянні із стандартним алгоритмом передачі навігаційних даних.

Список літератури:

1. DDF (Dynamic Data Flow — Динамический поток данных) [Электронный ресурс]. — Режим доступа: http://www.locarus.ru/pages/products/tehnologii/ddf

2. Хруль С. А. Адаптивный алгоритм обработки потока навигационных данных на основе метода диагностической фильтрации / С. А. Хруль, Д. М. Сонькин // Известия Томского политехнического университета [Известия ТПУ]. — 2012. — Т. 321

3. Упрощение полигональной цепи [Электронный ресурс] — Режим доступа: https://neerc.ifmo.ru/wiki/index.php?title=Упрощение_полигональной_цепи

Медведєва В.М., Симоненко Б.О.

Організація оптимального обчислювального процесу в корпоративній мережі на основі платформи .NET

Вступ. У власності підприємств, організацій, установ наявна велика кількість обчислювальних ресурсів, які ніяк не використовуються (наприклад комп'ютери працівників). Звичайний робочий графік працівників починається о 9 ранку і завершується о 6 вечора. Інший час великий парк ресурсів простоює. Незважаючи на розвиток хмарних технологій, їх використання не завжди доречне, особливо коли у підприємства вже є свій парк ресурсів.

Використання наявних ресурсів може зекономити установам кошти, особливо IT організаціям, де обчислювальні потужності потрібні не лише вдень, а й вночі (запуск автоматизованих тестів, збірка нових версій, тестування програмного забезпечення під навантаженням). Наприклад, вартість одного примітивного серверу, який може виконувати роль білд машини складає 29229 тис. грн [1]. Ці кошти можна зекономити за рахунок розподілення завдань на комп'ютери, що не використовуються в даний момент часу. Втрати ефективності в даному випадку будуть мінімальними через те, що потужність серверу еквівалентна потужності ПК середньостатистичного розробника. Втрати будуть лише в комунікації між машиною, що виконує задачу, та машиною що викликає виконання задачі. Але завдяки тому, що організація працює в межах LAN (Local area network), де затримки мінімальні і швидкість передачі даних сягає 1Гб/с – втрати на комунікацію і роботу розподілених ресурсів не повинні бути значними.

Для одночасного виконання задач на багатьох обчислювальних ресурсах використовуються розподіленні обчислювання. Розподілені обчислення – обчислювальне поняття, яке відноситься до кількох комп'ютерних систем, що працюють над вирішенням однієї проблеми. У розподілених обчисленнях одна проблема розділяється на кілька частин, і кожна частина вирішується за рахунок ресурсів різних комп'ютерів. До тих пір, поки комп'ютери об'єднані в мережу, вони можуть взаємодіяти один з одним, щоб вирішити цю проблему. Якщо взаємодія реалізована правильно, всі комп'ютери працюють як єдине ціле [2].

На сьогодні вже існує декілька пропріетарних рішень для розподілення обчислювальних задач. Найбільш розвиненими є інтернаціональні GRID інфраструктури. Вони використовуються для обрахунку складних наукових розрахунків, в тому числі для обрахунку даних CERN. На безоплатній основі туди допускаються наукові розрахунки, а для прибуткових організацій всі ресурси надаються на платній основі. Також GRID інфраструктура оптимізована для наукових обчислень і мало підходить для потреб компаній з розробки ПО, що спеціалізуються на програмному забезпеченні на основі .NET.

Для .NET інфраструктури існує 2 пропріетарних рішення – Selenium GRID та PNUnit (Parallel NUnit).

Selenium GRID – рішення для запуску автоматизованих браузерних тестів. Підтримує можливість одночасного виконання тестів на багатьох вузлах. Вузлами керує сервер. Це рішення дозволяє зменшити тривалість автоматизованого тестування від декількох годин до 15-30 хвилин, в залежності від кількості вузлів. Недоліком цього рішення є те, що воно може виконувати лише браузерне тестування. Також Selenium GRID не підтримує можливість моніторингу зайнятості вузлів, і всі задачі виконуються незалежно від поточного навантаження вузла.

PNUnit – розширення фреймворку для unit-тестування NUnit. Це розширення підтримує розподілене виконання задач. Принцип

роботи такий же як і у Selenium GRID, тільки PNUnit може виконувати будь-який код без обмеження лише браузерного тестування. Також, хоч і NUnit спочатку створювався лише як фреймворк для unit-тестування програмного забезпечення, його можна і використовувати для інших цілей таких як паралельне виконання задач, чи запуск невеликої DDoS атаки для перевірки власного програмного забезпечення на роботу під навантаженням. Для поточної задачі це рішення не підходить тим, що воно також не враховує можливої зайнятості ресурсів, і виконання тяжких задач може значно вплинути на продуктивність праці працівника що працює на вузлі.

Загальний принцип їх роботи зображено на рисунку 1. В задачах беруть участь вузли та один керуючий хаб, який знає мережеві адреси вузлів і при виконанні задачі розсилає програмний код для виконання на кожен із них. На вузлі встановлене програмне забезпечення яке «прослуховує» мережевий порт, і при отриманні команд через цей порт починає виконання програмного коду, який надійшов від хабу. Основними недоліками існуючих рішень є те, що вони не враховують зайнятість вузлів працівниками компанії, і виконання задач може викликати зменшення продуктивності праці працівників.

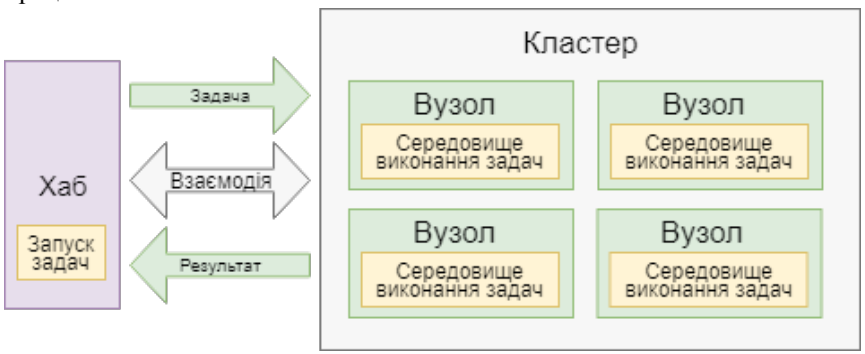


Рисунок 1 – типові розподілені обчислення

Мета дослідження. Удосконалити алгоритм розподілення задач з метою уникнення виконання задач на зайнятих користувачами

комп'ютерах за рахунок моніторингу діяльності користувачів, що дозволить перенести обчислення з хмарних ресурсів на комп'ютери працівників компанії як складових обчислювального кластеру. Досягнути зменшення тривалості виконання набору задач в декілька раз в порівнянні з виконанням на єдиному сервері.

Запропоноване рішення. Для задач підприємств здебільшого не потрібні високопродуктивні обчислення зі складними взаємодіями між кластерами. Для них краще підходить вид розподілених обчислень, який називається масово паралельними. При цьому керуючий кластер знає послідовність виконання частин великого завдання, і знає про всі доступні вузли. Ті частини завдання, які можна виконувати паралельно, кластер розсилає на вузли, і очікує їх завершення перед тим як почати виконувати іншу послідовну частину завдання. Вузли не знають один про одного і не взаємодіють між собою.

Важливою проблемою є те, що на відміну від серверів, робочі машини користувачів не можна використовувати тоді, коли на них працюють самі користувачі. Вони постійно виконують різноманітні задачі, вимикаються, навантажуються. Тому важливим є вибір на яких саме вузлах буде запускатись задача. Для цього запропоновано удосконалити традиційну модель розподілених обчислень (рисунок 1) додаванням кожному з вузлів модулю телеметрії (рисунок 2). За рахунок модулю телеметрії ведеться статистика доступності машин, їх поточний статус. Перед запуском задачі вузли повинні вибиратись на основі цих даних.

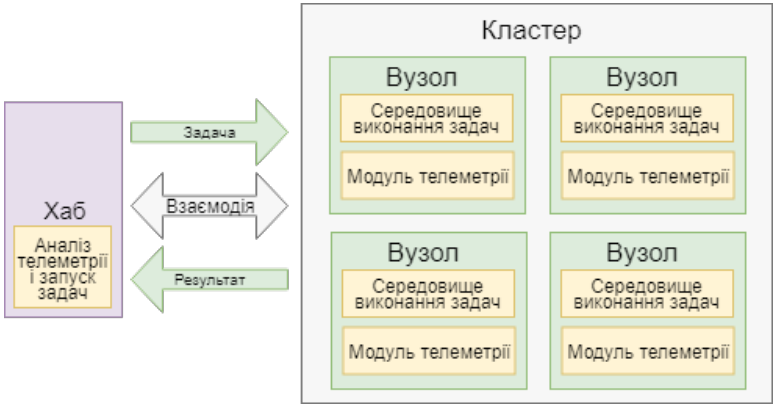


Рисунок 2 – запропонована модель розподіленої системи

Запропоновано такі основні компоненти хабу (управляючої машини):

- панель керування і моніторингу;
- модуль рішень;
- модуль, що відповідає за поширення задачі;
- ведення журналу подій.

Основні компоненти вузла (клієнтської машини):

- модуль збору даних (телеметрії);
- модуль взаємодії з хабом;
- модуль виконання задач.

На клієнтських машинах модуль збору даних відповідає за стан ресурсів. Наприклад, чи використовується ПК зараз користувачем і скільки ресурсів доступно в даний момент. Також постійно ведеться аналіз стану та об’єму доступних ресурсів. Можна брати фізичні характеристики обчислювального ресурсу, але на них не можна покластись. Наприклад, машина має ОЗП 8Гб. Якщо відняти витрати системи та встановленого ПО залишається приблизно 4Гб. І ця величина індивідуальна для кожного вузла і залежить від ревізії ОС, встановленого програмного забезпечення та програм, що працюють в даний момент.

Розробка основної системи. Для розробки програмного забезпечення використовувались такі технології:

- Для хабу в якості бази даних було вибрано SQL Server. Панель керування адміністратора реалізована на основі фреймворку ASP.NET MVC 5. Для можливості надсилання запитів із веб-додатку в браузер використовується технологія SignaR, що дозволяє миттєво обновлювати інформацію в панелі керування адміністратора. Модуль виконання задач є звичайним сервісом Windows і реалізований на платформі .NET.
- Всі вузли є також сервісами Windows і реалізовані на платформі .NET.

Висновок. Удосконалено алгоритм розподілення обчислювальних задач за рахунок збору і використання даних про активність працівників компанії. Раніше існуючі рішення розподіляли задачі на будь-які вузли, що могло викликати дискомфорт працівників.

Список літератури

1. Сервер Dell PowerEdge T130 (210-AFFS-PR) [Електронний ресурс] : [Веб-сайт]. – Режим доступу: https://elmir.ua/servers/server_dell_poweredge_t130_210-affs-pr.html (дата звернення 11.05.2018). – Назва з екрана.

Distributed Computing System [Електронний ресурс] : [Веб-сайт]. – Режим доступу: https://www.techopedia.com/definition/7/distributed-computing-system (дата звернення 11.05.2018). – Назва з екрана.

Сегеда І.В., Зіліцький В.Є.

Застосування QR-кодування в Україні

Технологія QR-кодування використовується сьогодні в рекламі, торгівлі, логістиці, туризмі, банківській сфері, інтернет-магазинах з метою максимальної економії часу клієнтів та онлайн-покупців. Однак поява в Україні нових технологій опрацювання інформації не означає, що вони відразу будуть затребувані. QR-код виконує дві функції: вміщує велику кількість інформації у невеликій картинці (більше двох друкованих сторінок) і дозволяє автоматично зчитувати закодовані дані. QR-код – простий, зручний та інтерактивний спосіб оперативного отримання і розповсюдження інформації.

Метою даної статті є дослідження застосування QR-кодування в Україні, оскільки сьогодні цей вид кодування найбільш розповсюджений в країні- розробнику – Японії, в Азії, Північній Америці та Європі.

QR- код є двовимірним представленням звичайного штрих кода, що поміщається практично на будь-яку вироблювану продукцію". QR" символізує миттєвий доступ до інформації, що зберігається в коді [1]. QR- код формується по строго визначеному алгоритму, який в спрощеному виді можна розділити на декілька етапів :

- Кодування даних.
- Додавання службової інформації.
- Розділення інформації на блоки.
- Створення байтів корекції.
- Розміщення інформації на QR- коді.

Заповнення QR- коду даними розпочинається з правого нижнього кута, від низу до верху, біт за бітом.

Як і інші технології, QR- коди мають свої плюси і недоліки. Позитивні сторони QR- коду: Код відноситься до відкритих технологій є доступним кожному; QR- код вміщує в себе набагато більше інформації і стійкіший до ушкоджень графічного малюнка; Недоліки: При використанні QR- коду необхідно бути упевненим, що адресат зможе його прочитати. Для читання такого коду потрібні гаджети, що уміють розпізнавати зображення QR- коду і розшифровувати його; Вміщує в себе відносно мало інформації, закодувати цілу книгу в один стандартний QR- код не представляється можливим; QR- код є загальнодоступною технологією, отже, не можна зберігати важливу інформацію у вигляді QR- коду, оскільки код не надає відповідний рівень захисту інформації;

В Україні багато сервісів використовують QR-коди. Так наприклад:

- Маркування деталей у виробництві.
- Зашифровані повідомлення, розміщені на рекламі, плакатах, листівках; зокрема реклама фірми із її закодованими логотипом, назвою, слоганом, адресою сайту, візитною карткою, датою ювілейної події.
- Кодування товарів і послуг, розміщення QR-кодів на касових чеках.
- Логотипи відомих брендів, які можуть містити адресу веб-сторінки.
- Розміщення кодів у друкованій рекламній літературі, зокрема у вигляді оголошень.
- Знижка при покупці товарів після сканування QR-коду, розміщеного в пам’яті мобільного телефона.
- Виготовлення електронних візиток (vCard, MeCard) для їх розміщення на паперовій візитці, сайті чи блозі.
- URL-адреси для «скачування» програмного забезпечення, файлів, мелодій, відео, афіш.

- «Запрошення» в соціальну мережу.

- Google-mapping в туризмі.

- Автоматичне внесення контактних даних з QR-коду візитної картки особи в адресну книгу мобільного пристрою.

- Електронний квиток, який можна отримати на залізничній станції чи в аеропорту після пред'явлення отриманого попередньо QR-коду у вигляді

SMS на мобільний пристрій.

- Доступ до сервісів інтернет-банкінгу.

- Особисті дані особи, які можуть бути вхідними даними для баз даних.

- Суміщення об'єктів реального і віртуального світу через присвоєння об'єкту QR-коду, який пов'язаний зі статтею у Вікіпедії (семапедія, QRpedia).

- Дизайн одягу, аксесуарів, будівель.

Приклад використання QR- кодів наведено на рисунку 1.

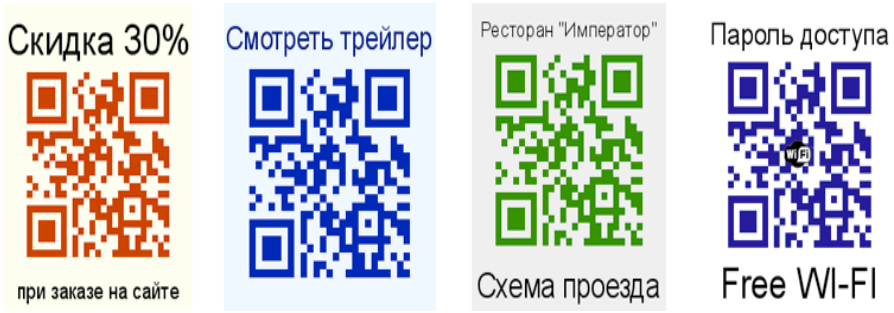


Рисунок 1. Приклади закодування з використанням QR-кодів

На сьогодні вбудованого механізму зчитування QR кодів в системі Android не існує, а велика кількість пропонованих додатків потребує чіткої картинки та якогось певного розташування коду. Кількість пристроїв на платформі Android збільшується з кожним днем, тож проблема створення зручного сканеру для цієї платформи є дуже й дуже актуальною. Наприклад, для підприємств, які планують

використовувати QR коди для швидкої авторизації працівників чи сканувати продукцію з ними в великих масштабах це дуже вагомий недолік, що стримує поширення таких прийомів там, де вони були б необхідні.

Створення саме такого програмного продукту обумовлено його перевагами в порівнянні з уже існуючими аналогами які не використовують технологію QR-коду.

- відновлення QR не потребує коштів і не займає багато часу;
- переносити код можна на любому пристрої;
- сканування QR коду не потребує ніякого спеціального обладнання;
- трекінг система може доповнюватись модулями, які потрібні адміністратору (підрахунок зп, тощо.);
- також трекінг система може бути доповнена додатковими засобами безпеки (напр. робити фото працівника що відсканувався).

Удосконаливши алгоритм сканування QR-коду за рахунок того що алгоритм буде знаходити пошукові маркери у будь-якому положенні коду вирівнюючи код за допомогою маркеру корекції відносно скануючого пристрою та модифікувавши його таким чином що алгоритм розшифровки QR-коду буде проходити масив даних що був створений на попередньому кроці у два цикли сортування, по вертикалі і по горизонталі, на відміну від стандартного алгоритму що працює у більшості скануючих пристроїв та проходить масив тільки по горизонталі.

Перевороти та підвищення якості зображення в умовах неякісного освітлення стали можливі завдяки підключенню до сканеру сторонньої бібліотекі Luminar Source що наразі використовується як фільтр для фотографій. Налаштувавши взаємодію сканера і бібліотеки можна досягти того що сканер буде швидко і точно працювати в умовах слабкого освітлення і нечіткого зображення. Поєднання цих двох технологій значно зменшує час сканування одного об'єкту.

Зміни часу який потрібен программі для сканування з використанням того чи іншого алгоритму, з та без використання Luminar Source приведено у таблиці 1.

Таблиця 1

Алгоритм	Час сканування
Звичайний Алгоритм	25мс
Звичайний Алгоритм з поліпшенням зображення.	31мс
Удосконалений Алгоритм	6мс
Удосконалений Алгоритм з поліпшенням зображення	10мс

Висновки з даного дослідження. Використання QR-кодів не обмежується ніякими ліцензіями, вони також описані та опубліковані в якості стандартів ISO. Найбільше визнання QR-код отримав серед користувачів мобільного зв'язку: встановивши програму-розпізнавач, абонент може після сканування коду заносити в свій мобільний пристрій текстову інформацію, додавати контакти в адресну книгу, переходити по web-посиланнях, відправляти SMS-повідомлення та ін.

Перевагою QR-кодів є висока ймовірність розпізнавання інформації навітьу випадку його пошкодження. QR-кодування є зручним, використовується в багатьох сферах, набуває популярності і лише почало використовуватися в банківській справі та електронній комерції.

Література:

1. Ковалёв А. И. QR-коды, их свойства и применение // Молодой ученый. — 2016. — №10. — С. 56-59. — URL https://moluch.ru/archive/114/29398/.

2. Электронная книга о QR-кодах. \\ Полное руководство по маркетингу с применением QR-кодов. — [Электронный ресурс]. — http://ru.qr-code-generator.com/qr-code-marketing/qr-codes-basics/.

Сич М. В.

Контроль енергоефективності систем вентилювання: аналіз, проблеми та альтернативні рішення

Вступ

XXI сторіччя відзначилося активним розвитком програм з енергозбереження. Одними з головних подій стали підписання Паризької угоди у рамках Рамочної конвенції ООН про зміну клімату та активний розвиток програми nZEB в ЄС. Метою даної програми є зменшення енергоспоживання будівель до <30% від існуючого, причому це споживання має покриватися локальними джерелами відновлюваної енергії.

Підвищення рівня енергоефективності, зокрема у житловому секторі, залежить від трьох основних напрямків:

- ефективне утеплення будинків з використанням сучасних технологій;
- використання альтернативних (відновлюваних) джерел енергопостачання;
- застосування енергоефективних інженерних систем;

Ефективне утеплення будівель лежить виключно у площині конструктиву самих будівель, ним неможливо керувати протягом року (за виключенням технологій по зміні пропускної здатності вікон, які дозволяють зменшувати інсоляцію в теплий період року).

Два останні пункти мають пряме відношення до інженерних мереж будівель.

Відомо, що загальні тепловтрати будівель складаються з тепловтрат через огороджувальні конструкції та тепловтрат на вентиляцію, але, на сьогодні, не достатньо уваги приділяється

системам вентилювання будівель. Залежно від функціонального призначення будівлі, тепловтрати на вентилювання можуть бути співвимірними з тепловтратами через огороджувальні конструкції і підвищення енергоефективності систем вентилювання принесе неменший економічний ефект.

Системи рекуперації тепла відіграють основну роль у розрахунках енергоефективності вентиляційного обладнання. Людина за добу споживає 30-40 кг свіжого повітря, для порівняння – усього близько 2 кг води та їжі.

На сьогоднішній день країни СНГ, у рамках виконання Паризької конвенції, здебільшого проводять модернізацію обладнання теплових пунктів, що у короткостроковій перспективі з невеликими капіталовкладеннями дає підвищення енергоефективності на 10-30%. Даний ресурс себе вичерпає вже за кілька років і настане час теплових насосів та систем вентилювання з рекуперацією тепла та холоду.

У даному матеріалі розглянуті системи вентиляції з рекуперацією тепла, політика найбільших виробників та головні проблеми даної сфери сьогодення.

Основна частина

Що під собою приховує термін: система вентиляції з рекуперацією тепла? Після зовнішнього утеплення будинків мешканці стикаються з проблемою відсутності свіжого повітря, оскільки порушується процес вентилювання через відсутність притоку свіжого повітря. Наступним кроком є облаштування системи механічної вентиляції, яка примусово подає повітря в приміщення і забирає відповідну ж кількість.

Особливістю програми nZEB є те, що маючи певну загальну ціль, кожна європейська країна впроваджує національні стандарти відповідно до своїх реалій. За приклад можна взяти Німеччину, чиї норми EnEV мають чи не найсуворіші вимоги до енергоефективності серед світових. (рис.1):

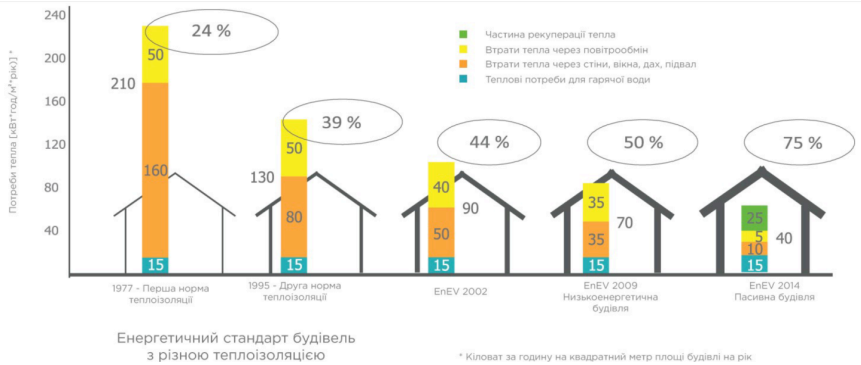


Рис.1. Частка вентиляційних втрат тепла у загальних теплових потребах будівель

Одним зі способів підвищення енергоефективності є застосування рекуперації тепла вентиляційного повітря (рис.2).

Основний параметр на який звертаємо увагу – це температурна ефективність.

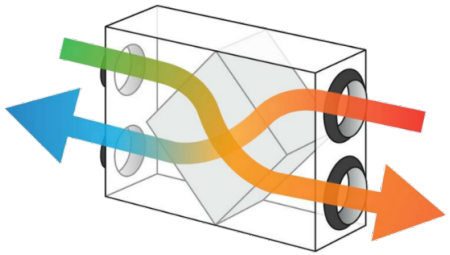


Рис.2. Принцип дії припливно-витяжної вентиляційної установки з рекуперацією тепла

Ефективність роторних рекуператорів може досягати 60-90%! Наприклад, температура повітря на вулиці -20°C, в приміщенні +22°C, після того, як повітря з вулиці пройде рекуператор, воно нагріється

приблизно до +16°C. На догрів 500 м³ такого повітря до +22°C потрібно затратити 0,6 кВт·год теплової енергії, для порівняння – щоб нагріти той же об'єм повітря до температури +22°C без рекуператора, знадобиться 6,7 кВт·год [1]. Ефективність рекуперації залежить від багатьох факторів, серед яких швидкість повітря в агрегаті, матеріал рекуператора, якість обладнання (а відповідно і його вартість) і багато інших.

Провідні європейські (і не тільки) виробники приєднуються до добровільної програми сертифікації EVROVENT, яка і визначає чи відповідає обладнання сучасним вимогам енергоефективності (таким к nZEB та ін.) (табл.1).

CLASS	Max. air velocity v_{class} [m/s]	Min. efficiency of HRS η_{class} [%]	Max. pressure drop of HRS Δp_{class} [Pa]	Fan Efficiency Grade $NG_{ref\text{-}class}$ [-]
A+	1.4	83	250	64
A	1.6	78	230	62
B	1.8	73	210	60
C	2.0	68	190	57
D	2.2	63	170	52
E	No calculation required			No requirement

Table 1: Definition for the Energy Efficiency classification

This new classification is in line with the latest Eurovent Certita Certification policy regarding Energy Efficiency classification, that is:

- A+ class represents less than 1% of the certified products;
- A class represents less than 5% of the certified products;
- B class represents less than 15% of the certified products.

Відповідно до таблиці, не так багато виробників можуть заявити про клас ефективності А та тим більше А+, хоча ви знайдете логотип EVROVENT на продукції більшості виробників: IV Produkt, Wolf, System Air, Swegon, GEA, Flakt Group та ін. Справа у тому, що така сертифікація може відноситися не до усього агрегата в цілому, а до його окремих частин чи навіть програми підбору обладнання.

Ще однією проблемою виявлення фактичної ефективності рекуперації тепла є політика компаній виробників. Не так багато з них постачають спеціальне програмне забезпечення, чи воно коштує досить дорого через наявність багатьох функцій зазвичай

надлишкових для дрібних клієнтів, або приватного сектора (IV Produkt Cloud, OJ Air Cloud та ін). Враховуючи, що більшість компаній виробників комплектує свої агрегати дорогим обладнанням для автоматизації, будівельні компанії користуються послугами локальних постачальників систем автоматизації, котрі концентрують увагу на функціоналі системи, а не на енергетичній ефективності. Ще гірше справи у виробників обладнання нижчого цінового діапазону Vesa, Ned, Aero Star, MyCond та більшість CEO бреднів поставляють на об'єкти з мінімальним бюджетом, тож питання енергоефективності там далеко не на першому місці. То ж усі намагання реалізувати у майбутньому контроль за енергоефективністю тієї ж вентиляції для отримання національних сертифікатів аналогічних nZEB потягнуть за собою чималі витрати на модернізацію, чи то обладнання, чи програмного забезпечення. Немає сумнівів, що країни СНГ підуть шляхом енергоефективності, як пішли Польща та Чехія.

Особливо гостро постає питання такої модернізації на невеликих підприємствах і у житловому секторі. Контроль за ефективністю рекуперації тепла сучасних систем вентиляції ускладнений не лише політикою компаній виробників (промислова таємниця), а й складністю технічних рішень та вартістю обладнання від сторонніх брендів. Такі флагмани як Siemens, MITSUBISHI ELECTRIC, Schneider Electric, CAREL та інші, здебільшого виробляють рішення для промислової автоматизації і просто не зацікавленні у виготовленні дешевих рішень, що склали б конкуренцію для їхньої власної продукції.

Частково цю ситуацію виправляє ринкова економіка, оригінальні рішення з децентралізованої вентиляції в останні кілька років дали поштовх до розвитку таким компаніям як Ventomaxx, Vaillant, Reventa, Eco Vent, Prana, Blauberg та багато інших. Ці невеликі вентиляційні установки продуктивністю 30…100 м³/год вирішують питання вентиляції в сучасних будинках і є надзвичайно енергоефективними.

Увесь будівельний ринок, а зокрема ретейлери та виробники обладнання орієнтуються лише на один параметр успішності компанії - маржі. Немає сенсу інвестувати в розробку дешевих продуктів, які не дадуть достатнього прибутку.

Ця проблема вирішиться сама по собі з розвитком такого напрямку, як IoT. Починаючи з ентузіастів та дрібних розробників, дані моделі автоматизації постійно захоплять більшість технологічного ринку і повна автоматизація усього невеликого виробництва виконуватиметься одним спеціалістом на рівні служби експлуатації. Як приклад можна привести рішення для контролю за енергоефективністю будь якого рекуператора на базі чіпа з WiFi модулем ESP8266 та датчиками температури і вологості DHT (рис. 3).

Проста схема підключення компонентів, автономна робота, невелике енергоспоживання, простота програмування керуючого блока у безкоштовній IDE для платформи Arduino, а головне – невелика вартість готового продукту (в районі 10 USD для одного агрегата). Така система монтується безпосередньо на корпусі, чи поруч з вентиляційною установкою, працює зі звичайною WiFi мережею, передаючи дані на локальний сервер або на віддалений сервер через мережу Інтернет. Така комплектація датчиків дозволить контролювати не лише ефективність збереження тепла, а й ефективність рекуперації вологи. Так як усі розрахунки проводяться на клієнт/серверній частині – цей же пристрій і алгоритм з мінімальними змінами можна застосовувати для теплових насосів, чиллерів, джерел відновлюваної енергії і т.д. (рис.4).

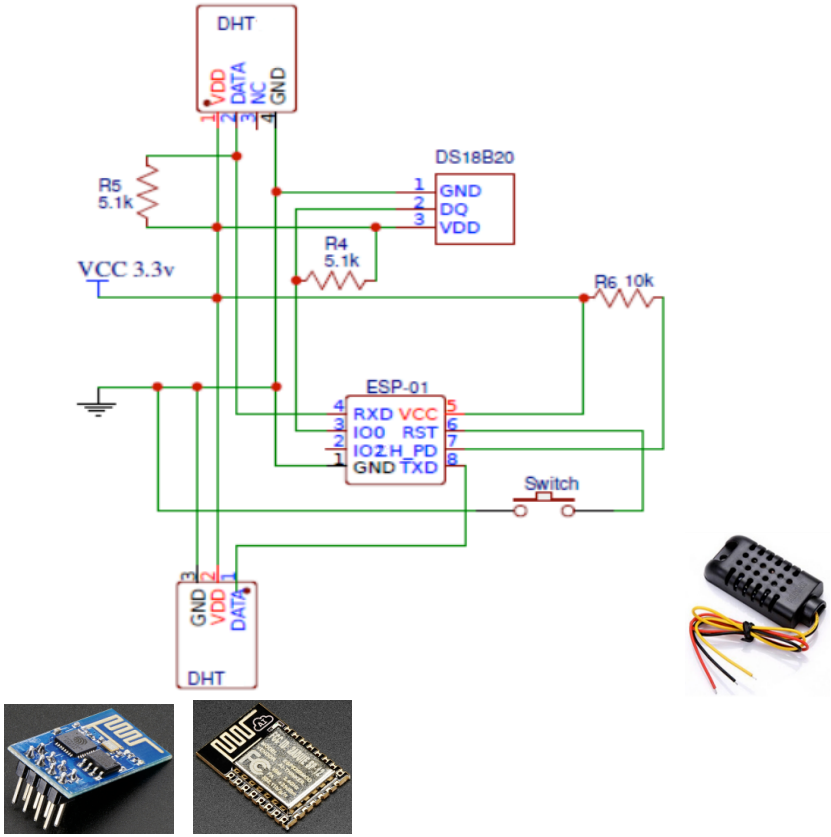


Рис.3. Схема та компоненти модуля моніторингу енергоефективності рекуператора

Data from the AHU unit:

ID	Unit	Temp1	Hum1	Temp2	Hum2	Temp3	Efficiency	Timestamp
7	l5u3m5e0s1	-14 C	84 %	11 C	32 %	21 C	71 %	2018-01-01 12:00:00 UTC
6	l5u3m5e0s1	-15 C	85 %	10 C	32 %	22 C	67 %	2018-01-01 11:00:00 UTC
5	l5u3m5e0s1	-12 C	87 %	12 C	32 %	20 C	75 %	2018-01-01 10:00:00 UTC
4	l5u3m5e0s1	-11 C	92 %	14 C	31 %	23 C	73 %	2018-01-01 09:00:00 UTC
3	l5u3m5e0s1	-17 C	95 %	12 C	38 %	20 C	78 %	2018-01-01 08:00:00 UTC
2	l5u3m5e0s1	-12 C	94 %	11 C	36 %	21 C	69 %	2018-01-01 07:00:00 UTC
1	l5u3m5e0s1	-17 C	90 %	10 C	33 %	22 C	69 %	2018-01-01 06:00:00 UTC

Рис.4. Web інтерфейс системи моніторингу ефективності рекуперації

Можливості подібних систем дозволяють будувати системи моніторингу та управління використовуючи лише WiFi з'єднання, базу даних та Web-інтерфейс. Сучасні клієнт-серверні додатки за функціоналом не поступаються сучасним SCADA системам і зможуть бути впровадженні на будь яких підприємствах на порядок дешевше ніж аналоги великих гравців ринку автоматизації.

Сучасні технології програмування дозволяють реалізовувати взаємодію з такими системами не лише через Web-інтерфейс, а й через додатки на планшетах та смартфонах, безпечно шифруючи дані.

Розвиток сучасних технологій згодом призведе до децентралізації рішень з автоматизації обладнання, вже сьогодні потужніші моделі контролерів оснащені різними засобами зв'язку з мережею та мають досить потужностей аби не просто передавати дані, а й керувати досить складним обладнанням.

Технології Web-розробки дають можливість пришвидшувати взаємодію з серверною частиною та робити інтерактивну клієнтну частину, тож увесь функціонал SCADA можна реалізовувати навіть через Web-інтерфейс.

Європейські норми ставлять досить амбітні задачі по енергозбереженню для будівельного сектору, зокрема вимоги деяких окремих членів ЄС, виглядають дуже суворими [2] (рис.5, 6).

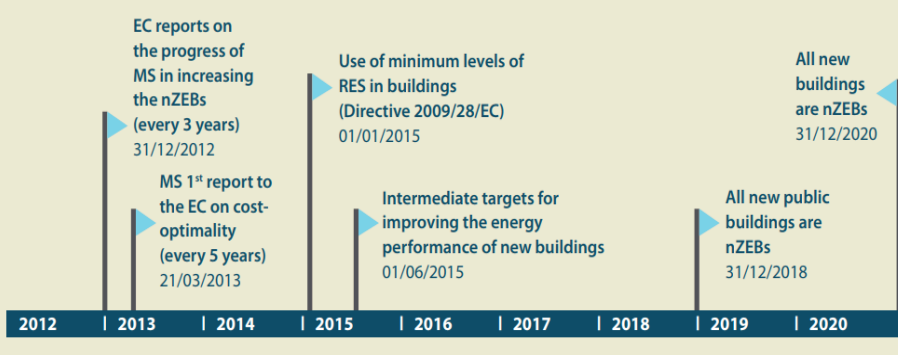


Рис.5. Основні роки для майже нульових енергетичних будівель (Directive 2010/31 /EC) (Джерело: EPISCOPE3)

Country	Maximum required primary energy consumption in new buildings		
	Before 2015	2015	
Denmark	52,5 + 1650/ (heated gross floor area)	30 + 1000/ (heated gross floor area)	
Slovak Republic	109-216 (Energy Class B)	55-108 (Energy Class A1)	(nZE

Рис.5. Шлях до nZEB для Данії та Словацької Республіки: максимальне споживання первинної енергії [кВт·год/(м²·рік)] для одноповерхових будинків (Джерело: BPIE, 2014)

Подібні вимоги можна реалізувати лише вводячи систему енергоменеджменту та автоматизованого контролю за роботою інженерних мереж.

Висновки

Країни СНГ продовжуватимуть традиції наслідування технологій і принципів енергоефективності, проте брак коштів в будівельній сфері дасть поштовх до розвитку дешевших рішень з тим же функціоналом. За кілька років ми станемо свідками активного розвитку розробок заснованих на IoT обладнанні.

Уже сьогодні для реалізації таких систем, як Smart House досить потужностей платформ Arduino та Raspbery PI, завтра їх буде достатньо й для промисловості.

Література

1. Ковальчук А.М., Листопадська Т.В.Ефективність використання вентиляційних установок рекуперативного

типу.// Энергосбережение. Энергетика. Энергоаудит – 2015.- №4 (135). – С. 7 – 14;

2. NEARLY ZERO ENERGY BUILDINGS DEFINITIONS ACROSS EUROPE - Buildings Performance Institute Europe (BPIE), 8c. (http://bpie.eu/uploads/lib/document/attachment/128/BPIE_factsheet_nZEB_definitions_across_Europe.pdf);

Смаковський Д.С., Соломкін М.В.

Підходи до побудови відмовостійких розподілених систем

Сучасні програмні системи досягли такої складності, що жодна людина у світі не може повністю зрозуміти усі аспекти роботи. Тому, розробникам довелося вигадувати нові методології розробки систем.

З’явився новий тип архітектури для побудови великих систем, що називається мікросервісною. Ця архітектура полягає в тому, що система будується з декількох маленьких сервісів, кожен з яких добре виконує свою частину роботи. Для комунікації один з одним можуть виконуватися як синхронні REST запити, так і асинхронна обробка та відправка повідомлень за допомогою черг.

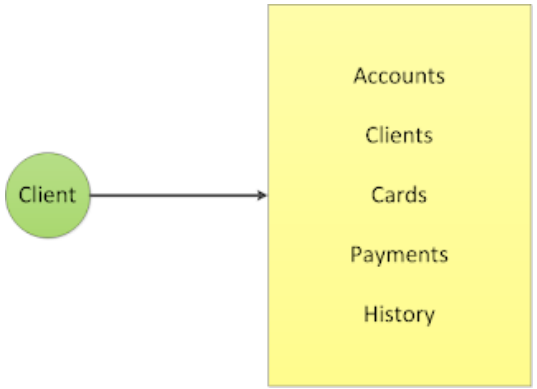


Рисунок 1 – Монолітна система

Мікросервісна система є розподіленою за своєю природою. Отже, система повинна бути розділена на логічні сервіси. Мікросервісна система виділяє функціональні одиниці у окремі сервіси та масштабується через багатократне підняття необхідних функцій на декількох серверах

Розподілені сервіси забезпечують можливість масштабування самостійно. Кожен сервіс може бути розгорнутий окремо зі збільшенням кількості екземплярів тих сервісів, що потребують більшої обчислювальної потужності (наприклад, для підтримки більшої кількості запитів). Це називається горизонтальним масштабуванням.

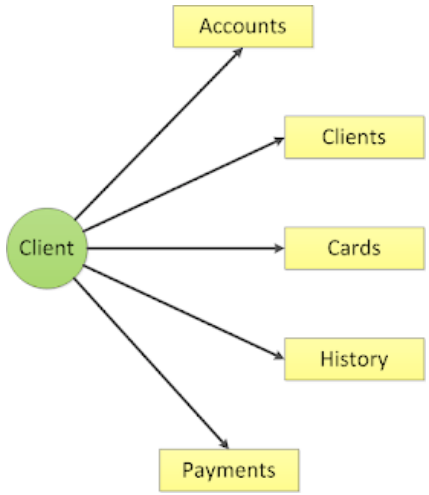


Рисунок 2 – Програма, яка розподілена на сервіси

З увімкненою опцією горизонтальної масштабованості ми можемо ввести шаблон балансу навантаження на стороні клієнта (client side load balancing) [1]. Коли клієнт хоче відправити запит до

сервісу, він може вибрати будь-який екземпляр із списку доступних екземплярів сервісу. Це дозволяє розподілити навантаження зі сторони клієнтів по всім екземплярам сервісу рівномірно. Будь-який сервіс в мережі може бути клієнтом. Зазвичай клієнт робить запити та отримує відповіді, а сервіс може бути клієнтом для іншого сервісу і робити запити до нього, і це може статися каскадно.

Загалом, все, що обговорювалося вище, може бути найпростішим способом застосування мікросервісної архітектури. Однак у такому випадку існує проблема. Як клієнт отримає інформацію про наявні екземпляри інших сервісів? Для цього можуть бути використані файли конфігурації, але це додає ще більшу складність, оскільки потрібно оновлювати конфігурацію після кожної зміни топології. Використання шаблону пошуку сервісу (Service discovery) вирішує проблему. Цей шаблон представляє собою окремий сервіс, який відіграє роль адресної книги (реєстру) для інших сервісів. Коли сервіс запускається, він повідомляє адресу (наприклад, хост і порт) до реєстру та надсилає повідомлення (наприклад, кожні 30 секунд), поки він є робочим. Отже, реєстр знає, які сервіси ще живі. Більше того, цей шаблон дозволяє реалізувати гнучку масштабованість. Таким чином, можливо масштабувати сервіси під час роботи системи, динамічно додаючи або зменшуючи кількість екземплярів, коли це потрібно.

Коли клієнт хоче відправити запит до сервісу, єдине, що він повинен знати – це ім'я, яке використовується для запису сервісу у реєстрі. Таким чином, клієнт робить запит до реєстру, щоб отримати

список доступних екземплярів сервісів, а потім робить прямий запит до обраного екземпляра сервісу. Екземпляр, зазвичай, обирається за допомогою логіки балансування навантаження.

Ви можете помітити, що ця архітектура вимагає від клієнтів знання про місцезнаходження реєстру та інтеграцію з ним. Це поширена практика для мікросервісних систем. Однак зовнішні клієнти не повинні нічого знати про внутрішню архітектуру системи. У мікросервісній архітектурі застосовується шаблон API шлюзу (API Gateway), який є єдиною точкою доступу до систему для зовнішніх систем [2]. Застосування шлюзу дозволяє приховати внутрішню складність системи для зовнішніх клієнтів.

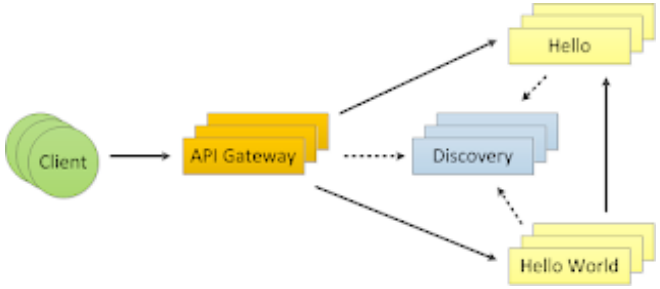


Рисунок 3 – API для сторонніх клієнтів

API шлюзи надають функціональність за допомогою наявних мікросервісів. Шлюзи звертаються до реєстрів, щоб отримати список доступних сервісів та звертаються до них застосовуючи балансування навантаження. Існує два види шлюзів: проксі-сервер і логічний. Проксі-шлюзи просто передають виклик до потрібного сервісу, а логічні шлюзи можуть виконувати декілька запитів для сервісів, щоб надавати інший API своїм клієнтам.

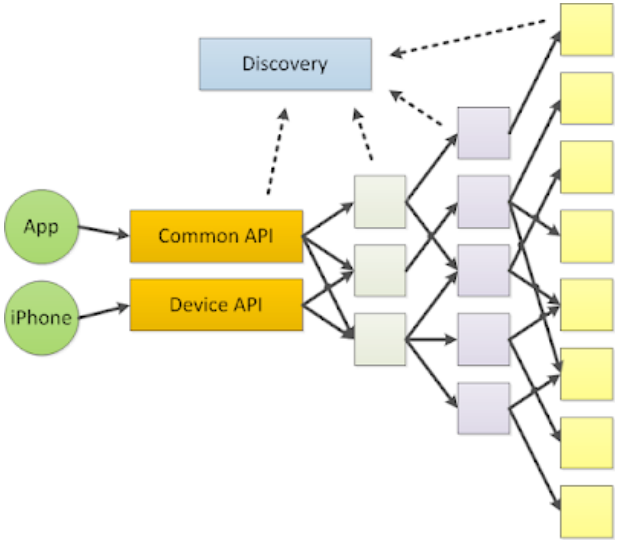


Рисунок 4 – Мікросервісна система

Це основні шаблони проектування, без яких не існує жодної реальної мікросервісної системи. Але існує ще безліч різних шаблонів, які націлені на покращення відмовостійкості системи та полегшення її конфігурації. До таких шаблонів відносяться автоматичний переривач запитів (Circuit breaker) та єдиний сервіс конфігурації для всіх сервісів.

Застосування мікросервісної архітектури дає великий простір для масштабування та гнучкості, але привносить додаткові навантаження на сервери та мережу, ускладнює розробку та тестування. Тому перед застосуванням мікросервісної архітектери потрібно уважно оцінити всі плюси і мінуси для конкретно взятою системи.

У більших додатках може бути застосовано більше шаблонів, а також деякі шаблони на стороні сервера та на стороні клієнта можуть бути змішаними. Загальна картина полягає в тому, що може існувати багато примірників багатьох служб, розгорнутих по мережі. Для підтримки цієї інфраструктури потрібна команда операцій. Саме тут підхід мікросервісу відіграє свою роль у життєвому циклі розгортання та підтримки додатків шляхом введення вказівок щодо організації процесів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Microservices: Approach or Architecture? [Електронний ресурс] — Режим доступу: http://www.ogasimov.com
2. Newman, Sam Building Microservices [Text] / Sam Newman. -O'Reilly Media, 2015. - P. 53-58.

Тарнавський Ю.А., Горб І.

Роль аналітичних систем в організації оперативного енергетичного менеджменту

Як відомо, кожне виробниче підприємство так чи інакше вирішує питання енергозбереження та раціонального використання паливно-енергетичних ресурсів. Енергетичний менеджмент є обов'язковим елементом в структурі підприємства, яке поставило собі за мету скорочення споживання енергетичних ресурсів шляхом їх ефективного використання [1].

Головною метою енергетичного менеджменту є скорочення витрат підприємства на виробництво продукції за рахунок скорочення витрат на енергетичні ресурси [2].

Впровадження енергетичного менеджменту дозволяє відстежувати картину споживання енергії, аналізувати та планувати впровадження енергозберігаючих заходів. Саме тому актуальною є задача оптимізації споживання енергетичних ресурсів та підвищення енергоефективності промислового підприємства. Вирішенням цієї задачі є впровадження на підприємстві системи оперативного енергетичного менеджменту.

Система оперативного енергетичного менеджменту - це система керування, заснована на проведенні типових вимірювань і перевірок, що забезпечують таку роботу підприємства, при якій споживається тільки цілковито необхідна для виробництва кількість енергії [3].

Основними задачами аналітичних систем в енергетичному менеджменті є:

- прийняття рішень з питань енергоспоживання;
- оцінювання результатів підвищення рівня енергоефективності;

- аналіз ефективності впровадження політики в сфері енергозбереження;
- планування заходів з питань поліпшення роботи системи енергетичного менеджменту на підприємстві

Функціонування системи розглядається як безперервний процес, здатний ураховувати зміни, що відбуваються на об'єкті, конкретні особливості його експлуатації, мінливість параметрів режиму в часі. Саме тому мова йдеться про нормалізацію енергоспоживання як «індикатор, здатний ураховувати різні обставини використання енергії на об’єкті» [3].

Основними функціями системи є:

- зіставлення фактичних показників з нормативними або граничними;
- зіставлення фактичних показників з показниками минулих місяців/років;
- зіставлення результатів діяльності до і після зміни якого-небудь фактору;
- створення системи моніторингу рівня енергетичної ефективності підприємства або його окремих структурних підрозділів;
- моніторинг тарифів на енергетичні ресурси.

Реалізація цих функцій сприятиме досягненню реального поліпшення енергетичної ефективності промислового підприємства, підвищенню рівня контролю використання енергії та зменшенню питомих витрат в технологічних процесах.

Перелік посилань:

1. Системи енергетичного менеджменту [Електронний ресурс] — Режим доступу: http://www.reee.org.ua/energy-efficiency/

2. Енергоменеджмент підприємств [Електронний ресурс] — Режим доступу: http://ukrenergoefekt.com.ua/ua/
3. Впровадження систем енергоменеджменту [Електронний ресурс] — Режим доступу: http://kyrator.com.ua/

Тарнавський Ю.А., Крижанівська Ю.В.

Моделювання режимів роботи інтегрованих систем енергозабезпечення

За недостатності та невідповідної якості власних та високої вартості імпортованих паливно-енергетичних ресурсів в Україні, підвищеного попиту на електричну енергію в години пікового споживання (особливо в опалювальний період), за недостатності наявних електро-генерувальних потужностей в Україні та періодичної неузгодженості графіків вироблення і споживання електричної енергії, з метою зменшення навантаження на енергосистему України в сучасних умовах надзвичайно актуальним постає питання оптимізації режимів роботи систем енергозабезпечення.

Зважаючи на актуальність поставленого питання, за останні роки проведено низку досліджень з ефективності застосування інтегрованих систем енергозабезпеченняя. Метою даного дослідження є оцінка енергетичної ефективності інтегрованих систем енергозабезпечення, визначення ефективних режимів роботи інтегрованих систем енергозабезпечення з урахуванням комплексного впливу змінних режимів роботи, використання у системі джерел альтернативної енергії різних рівнів потужності, з урахуванням втрат енергії під час генерування, постачання й перетворення електричної енергії.

Об’єктом дослідження виступають інтегровані системи енергозабезпечення. Мета роботи полягає в створенні комплексу математичних моделей і методів управління інтегрованими системами енергозабезпечення в умовах запровадження сучасної концепції реформування енергетичної галузі на основі інтелектуалізації електричних мереж, застосування сучасних інформаційних технологій та математичного апарату, використання потенціалу джерел генерації,

розробки та впровадження гнучкої системи тарифів на електричну енергію, а також побудови дієвої системи контролю ефективності її використання в суспільному виробництві.

Дана пропозиція передбачає застосування комбінованих систем енергопостачання, що включають альтернативні джерела енергозабезпечення, це дозволить знизити споживання електричної енергії від мережі, а також одержати дешевшу за собівартістю електроенергію порівняно з мережевою. Комбінована система може бути включати такі джерела альтернативної електроенергії як сонячні батареї, а також вітрові електрогенератори, які випускають українські та закордонні підприємства.

Використання розроблених пропозицій щодо застосування альтернативної електричної енергії дасть змогу удосконалити діючу в Україні методику встановлення відповідних тарифів на електроенергію, стимулювати споживачів до здійснення контролю та керування потоками альтернативної енергії, що, в свою чергу, дозволить підтримувати окремі показники якості електроенергії в мережі на необхідному рівні.

Перелік посилань:

1. Енергозбереження в промисловості. Частина1. Технології. Навчальний посібник / Укладачі: Праховник А.В., Суходоля О.М., Денисюк С.П., Прокопенко В.В. – К.: НТУУ «КПІ», 2011. – 417 с.

2. Попов В.А. Пути повышение эффективности моделирования электрических нагрузок в современных системах электроснабжения. / В.А. Попов, А.А. Журавлев, В.Н. Сидоренко, В.И. Степанова // Енергетика: економіка, технології, екологія / Наук. журнал. – К.: НТУУ «КПІ», 2010. – № 2. – С. 97 – 103. – Бібліогр.: с. 103. – 150 екз. – ISSN 1813-5420

3. http://www.lib.ua-ru.net/diss/cont/349169.html

Тарнавський Ю.А., Смоліженко Д.П, Михайлюк А.В.

Захист веб-систем на основі комп'ютерних тестів

На даний час сучасні веб-системи не здатні запобігати атакам автоматичних систем. Проблема автоматичних реєстрацій та інших операцій, якими можуть атакуватись сайти, зростає. Засобами виявлення атак автоматичних систем є різновиди оберненого тесту Тюрінга[1] (наприклад, CAPTCHA). Саме тому удосконалення та створення нових подібних алгоритмів є важливим напрямком захисту веб-систем. Це зумовлює актуальність розробки нових засобів та удосконалення існуючих.

Програмний засіб має забезпечити користувачу можливість динамічно генерувати та конфігурувати різні види комп'ютерних тестів. Користувачами системи можуть бути розробники програмного забезпечення. Програмний засіб може використовуватися у програмних системах.

Головною метою роботи є розробити засоби захисту інформаційних систем за допомогою комп'ютерних тестів. Завдяки цьому кожен користувач зможе додати капчу на веб-сайт та легко зконфігурувати її для роботи.

[1] Рассел Дж. Тест Тьюринга / Дж. Рассел – Красноярск: СибГАУ, 2012. – 98 с.

Завданнями роботи є:

- Проаналізувати існуючі методи захисту інформаційних систем на основі комп'ютерних тестів і інструментальних засобів з їх розробки;
- Дослідити можливості використання комп'ютерних тестів для захисту веб-систем;
- Розробити інструментальні засоби для використання нових методів захисту інформаційних систем на основі тестів.

У ході розробки було створено засіб діаграма класів[2] якого зображена на рисунку 1.

Клас Repository відповідає за роботу з базою даних та реалізує інтерфейс IRepository. Має функції видалення, вставки, отримання запису з бази даних та створення унікального ідентифікатора. Клас CacheProvider відповідає за роботу з оперативною пам'яттю та реалізує інтерфейс ICacheProvider. Має функції видалення, вставки, отримання об'єкту з оперативної пам'яті та створення унікального ідентифікатора. Клас CacheStorageProvider інкапсулює роботу з оперативною пам'яттю та реалізує інтерфейс ICaptchaStorageProvider. Інкапсулює функції класу CacheProvider. Клас DatatbaseStorageProvider інкапсулює роботу з базою даних та реалізує інтерфейс ICaptchaStorageProvider. Інкапсулює функції класу Repository.

[2] Шилдт Г. C# 4.0: полное руководство / Г. Шилдт. – К.: Вильямс, 2011. – 1056 с.

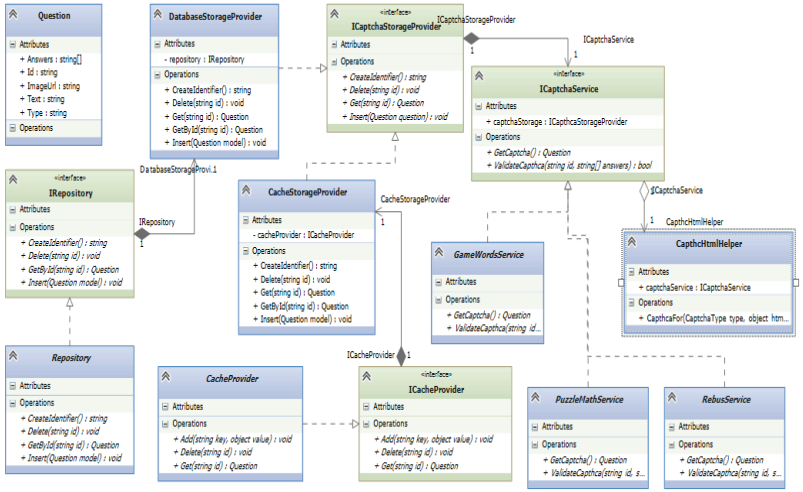


Рисунок 1 – Діаграма класів розробленої системи

Основними класами програмного засобу є класи: GameWordsService, PuzzleMathService, RebusService та CaptchaHtmlHelper. Класи GameWordsService, PuzzleMathService та RebusService реалізують роботу алгоритмів комп'ютерних тестів. Клас GameWordsService реалізує роботу алгоритму тесту Мюнстерберга та інтерфейс ICaptchaServie. Відповідає за функції генерації набору символів з словами серед них та збереженням зображення зі згенерованим текстом. Клас PuzzleMathService реалізує роботу алгоритму математичних головоломок та інтерфейс ICaptchaServie. Відповідає за функції генерації капчі на основі зображень та збереженням зображення зі згенерованою капчею. RebusService реалізує роботу алгоритму головоломок, а саме ребсів та інтерфейс

ICaptchaServie. Відповідає за функції генерації капчі на основі гри з визначення відповідностей. Клас CaptchaHtmlHelper відповідає за генерацію html розмітки та відображення капчі на веб-сторінці.

Після дослідження тестів Тюрінга було визначено такі види для реалізації:

1. Тест Мюнстерберга. Серед літерного тексту є слова. Завдання – знайти ці слова та ввести (рисунок 2).

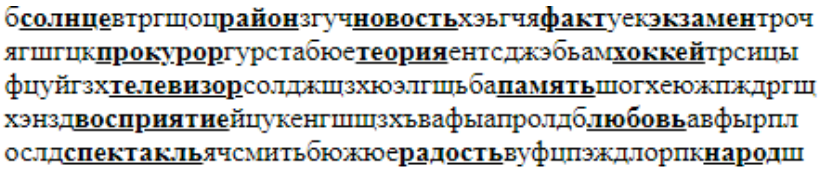


Рисунок 2. Тест Мюнстерберга

2. Клас головоломки. Тут можна виділити такі підкласи:

a) Ребуси;

b) Математичні. Вид тестів в якому потрібно визначити відповідь за допомогою розрахунків;

c) Лабіринти. Вид тесту в якому потрібно визначити правильних шлях від початкової точки до кінцевої.

Користувачами програмного засобу можуть бути розробники програмного забезпечення. Для того, щоб додати капчу, потрібно знайти її в Nuget Package Manager (рисунок 3).

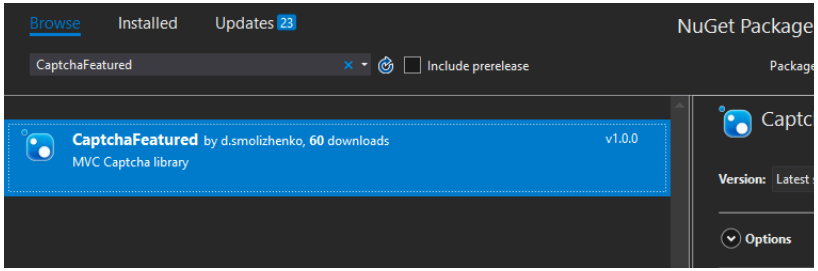


Рисунок 3 – Приклад додавання капчі в проект

Далі представлено приклад використання капчі для динамічної генерації на сторінці:

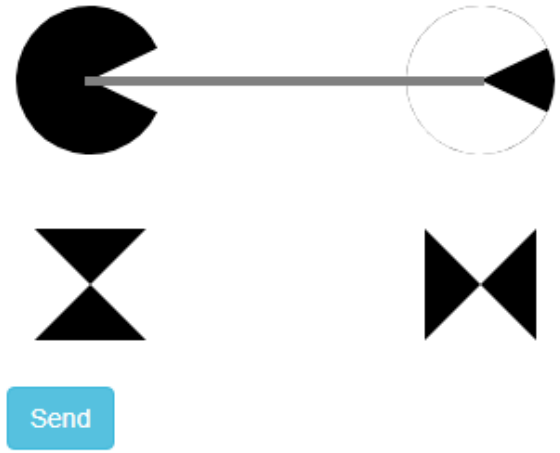
@Html.CaptchaFor(m => m.Captcha, new {@class = "captcha-text-field"})

Нижче представлено приклад використання капчі де користувач вказує вид тесту, який буде генеруватись на сторінці.

@Html.CaptchaFor(m => m.Captcha, CaptchaTypes.Rebus, new {@class = "form-control captcha-text-field"})

При завантаженні сторінки засіб сам генерує обрану капчу та розміщує на обраній сторінці (рисунок 4, а та б). Якщо капчу не обрано, то при завантаженні сторінки засіб буде динамічно генерувати різні види капчі.

а)



б)

Рисунок 4 — Приклад відображення різних видів капчі

Розроблено засіб генерації капчі на основі використання програмного пакету реалізації алгоритмів комп’ютерних тестів.

Розроблена система забезпечує динамічну генерацію чи генерацію конкретного виду качі на веб-сторінці.

Титенко С. В., Астахов А.Г.

Онтологічно-орієнтована навчальна система для вивчення дисциплін історичного спрямування

Вступ

Швидкий розвиток сучасних технологій призводить до того, що електронні пристрої завдяки зручності та доступності стають головним джерелом отримання інформації. Це стосується зокрема і області навчання, де переведення освітнього процесу в формат дистанційного навчання є актуальною задачею сучасності. Основним етапом цього процесу є формалізація навчальних матеріалів, тобто представлення даних таким чином, щоб їх можна було обробляти і представляти у зручному вигляді, та застосовувати елементи автоматизації задля більшої ефективності.

Розглянемо вже існуючі рішення представлення історичних даних для вивчення на дистанційній основі. Система *GoConqr* [1] представляє собою сучасний портал знань, на якому зосереджена велика кількість матеріалів по різноманітних напрямках. Матеріали представляються у вигляді як звичайних статей, так і у вигляді цікавих презентацій чи відео-роликів. В системі можна задати напрямок, за яким ви хочете отримати матеріали для вивчення. Матеріали за запитом виводяться у стрічці, де зазначається назва матеріалу, його

рейтинг за думкою користувачів та автор цього матеріалу. Кожен матеріал можна оцінити та занести до власної бібліотеки, для того, щоб потім можна було до нього повторно звернутись. Головним недоліком цієї навчальної системи є те, що всі матеріали знаходяться в загальному списку, що незручно з точки зору структурування та навігації. Єдиний спосіб згрупувати дані – вибрати одну з міток, які представлені у вигляді вже сформованого списку тем, наприклад, “історія мистецтв”, “модерн”, “війна” тощо. Така функція дещо полегшує пошук необхідних даних, але все одно унеможливлює пошук та групування певних історичних подій. Також тут є можливість обирати навчальні курси за певною тематикою, але це платно.

Прикладом системи для вивчення історії є портал [2]. Окрім стандартного розбиття матеріалів за розділами, та додавання до них відповідних тегів, присутня досить цікава функція «Стрічка часу». *Ця стрічка дозволяє пересуватись від події до події, від десятиріччя до десятиріччя, та таким чином візуально спостерігати за подіями, які відбувались в певній період. Безпосередньо зі стрічки можна обрати подію яка цікавить, та перейти до вивчення відповідного матеріалу. До недоліків можна віднести відсутність карт подій чи понять, що не дозволяє в широкому обсязі розглянути певну область історичних подій.*

Іншим прикладом системи дистанційного навчання є *EdEra* [3]. У бібліотеці цього порталу можна знайти підручники за багатьма

напрямками такими як фізика, географія, історія України та інші. Кожен тематичний підручник розбитий на теми, які в свою чергу розбиті на невеличкі підтеми. Навчальні курси являють собою звичайні статичні підручники, переведені у інтернет-версію. У користувача немає жодної можливості шукати потрібні йому дані.

У якості формальної моделі контенту системи для вивчення дисциплін історичного спрямування було обрано понятійно-тезисну модель (ПТМ) [4]. Дана модель надає широкі можливості автоматизації ряду функцій навчальних систем. Цей підхід дозволяє формалізувати навчальний контент та будувати інформаційно-пошуковий тезаурус, тобто словник термінів і даних, доповнений інформацією про зв'язки між ними. Але при застосуванні згаданої моделі для формалізації історичних даних виникає потреба її модифікації із-за відсутності компонентів та властивостей, необхідних для використання в даній предметній області.

Для кращого сприйняття навчальних матеріалів, необхідно правильно та ефективно їх подавати. Як було зазначено в статті [5], правильна візуалізація навчальних матеріалів дозволяє краще аналізувати та засвоювати більшу кількість інформації, розвиває критичне мислення, та допомагає зв'язувати отриману інформацію в цілісну картину. Добрим способом візуалізації історичних даних є побудова стрічки часу, на якій можна переглядати історичні події в хронологічному порядку. Прикладом такої візуалізації є сторінка

інформаційного ресурсу CNN, на якій розповідається про загострення ядерної проблеми у Північній Кореї (рисунок 1).

Рис. 1 – Представлення послідовності історичних подій за допомогою стрічки часу

У випадку візуалізації даних історичного спрямування, з оглядом на обрану модель формалізації контенту, найбільш ефективним способом представлення інформації є карти понять (concept map) [6]. Це ідея побудови графу асоціацій, де в якості вершин виступають певні поняття (історичні особистості, події, тощо), а в якості ребер – зв'язки між поняттями. Також варіантом представлення даних є тематичні (topic maps) [7] та інтелектуальні карти (mind maps) [8], які дозволяють у більшій мірі розкрити певну навчальну тему.

Отже проблема формалізації навчальних матеріалів історичного спрямування в автоматизованих навчальних системах набуває особливої актуальності. Важливою тут є також зручна візуалізація історичних даних, що дозволить користувачам краще

сприймати інформацію та надасть їм зручні інструменти дослідження навчальних матеріалів.

Постановка задачі

Метою роботи є створення формальної моделі та програмного забезпечення для формалізації навчального контенту системи для вивчення дисциплін історичного спрямування. Формалізація повинна відбуватись на основі понятійно-тезисної моделі [4], яку в свою чергу необхідно модифікувати для застосування в предметних областях історичного напрямку. Ставиться завдання забезпечити автоматизовану побудову карт понять на базі синтаксичного аналізу елементів понятійно-тезисної моделі. Модель структурування навчального контенту історичних дисциплін повинна забезпечити можливість гнучкого вибору історичного періоду для подальшого вивчення користувачем.

Модель формалізації навчальних матеріалів історичного спрямування

Понятійно-тезисна модель (ПТМ) – це модель подання знань, яка формалізує зміст навчального контенту і була розроблена спеціально для систем дистанційного навчання з урахуванням специфічних вимог навчального процесу [4]. Понятійно-тезисна

модель формалізує дидактичний текст і мультимедіа, які є вмістом інформаційно-навчальної програмної системи. ПТМ надає засоби опису понятійної складової контенту і забезпечує основу для програмного інструментарію редагування і використання бази даних та знань

Поняття – це певний об'єкт обговорення деякої області, який представляється для вивчення. Формально поняття – це одне або кілька слів, які виражають предмет розгляду деякого фрагменту навчального матеріалу [9]. **Теза** – це деяка відомість або твердження про поняття. Якщо поняття вказують на предмет курсу, то тези являють собою смислове наповнення бази знань.

В ПТМ, кожне поняття чи тезу можна класифікувати, тобто з'являються такі сутності, як клас поняття та клас тези. Ці сутності дозволяють розрізняти дані та формувати корисні вибірки матеріалів.

Для предметної області історичного спрямування, ПТМ не має необхідних класів понять, тому класи були розширені наступними значеннями:

$$CClasses = \{Cp, Ci, Ce\},$$

де Ср – клас "особистість", Сі – клас "інституція", Се – клас "подія".

Для використання понять та тез в області історичного спрямування необхідно, щоб ці сутності мали властивості часового періоду, що дозволить встановити їх місце у хронології подій. Але ПТМ в своїй початковій реалізації не має таких властивостей, тому було запропоновано вдосконалити понятійно-тезисну модель властивостями часового періоду для забезпечення її використання в навчальних системах історичного спрямування.

Для кожного елемента понятійно-тезисної моделі введено свій період часу, до якого він відноситься: $tpk = (t1, t2)$, де t_1=<D,M,Y> – день, місяць, рік початку, t_2=<D,M,Y> – день, місяць, рік кінця часового періоду.

Схематичне зображення ділянки ПТМ у співвідношенні з навчальним матеріалом (рисунок 2):

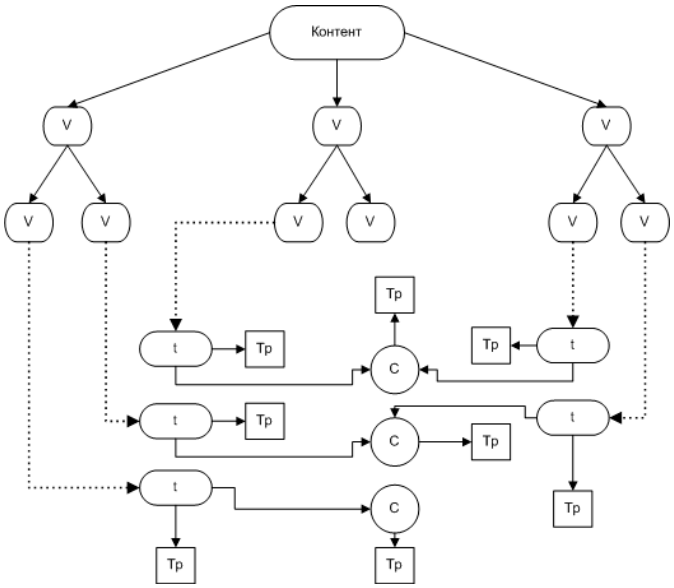


Рис. 2 – Ділянка ПТМ у співвідношенні з навчальним матеріалом, де *v* – навчальний матеріал, c – поняття, t – теза, tp – часовий період

Кожен період має свою дату початку: $tpStart(tpk) = t1$, та дату завершення $tpEnd(tpk) = t2$.

Як вже зазначалося, кожне поняття має свій часовий період:

$$CTP: C \rightarrow TP.$$

Так само до певного часового періоду відносяться тези та навчальні матеріали:

$$TTP: T \rightarrow TP,\ VTP: V \rightarrow TP.$$

Внесені модифікації по понятійно-тезисної моделі дають можливість гнучко обирати необхідні навчальні матеріали. Алгоритм отримання множини понять за заданим часовим проміжком описується наступним оператором:

$$\begin{aligned}\mathrm{CofTP(tpu)} = \{\mathrm{c}:\ & \mathrm{tpSart(CTP(c))} \geq \mathrm{tpStart(tpu)} \wedge \\ & \mathrm{tpStart\big(CTP(c)\big)} \leq \mathrm{tpEnd(tpu)} \vee \mathrm{tpEnd\big(CTP(c)\big)} \\ & \geq \mathrm{tpStart(tpu)} \wedge \\ & \mathrm{tpEnd\big(CTP(c)\big)} \leq \mathrm{tpEnd(tpu)} \vee \mathrm{tpSart\big(CTP(c)\big)} \\ & < \mathrm{tpStart(tpu)} \wedge \\ & \mathrm{tpEnd(CTP(c))} > \mathrm{tpEnd(tpu)}\}.\end{aligned}$$

Аналогічно робиться запит для отримання навчального контенту заданого періоду:

$$\begin{aligned}\mathrm{VofTP(tpu)} = \{\mathrm{v}:\ & \mathrm{tpSart\big(VTP(v)\big)} \geq \mathrm{tpStart(tpu)} \wedge \\ & \mathrm{tpStart\big(VTP(v)\big)} \leq \mathrm{tpEnd(tpu)} \vee \mathrm{tpEnd\big(VTP(v)\big)} \geq \\ & \mathrm{tpStart(tpu)} \wedge \mathrm{tpEnd\big(VTP(v)\big)} \leq \mathrm{tpEnd(tpu)} \vee \\ & \mathrm{tpSart\big(VTP(v)\big)} < \mathrm{tpStart(tpu)} \wedge \mathrm{tpEnd\big(VTP(v)\big)} > \\ & \mathrm{tpEnd(tpu)}\}\end{aligned}$$

Автоматична побудова відношень між поняттями

Для інтелектуалізації системи дистанційного навчання, необхідно певним чином пов'язувати ті чи інші поняття між собою. Це необхідно робити для того, щоб в подальшому користувачу можна було надавати інтерактивні дані про поняття, та зв'язки між ними.

Система має аналізувати поняття та тези, і знаходити в них синтаксичне входження інших понять, тоді на основі входжень одних понять до означень та тез інших понять будуються відношення між поняттями.

В деякому понятті *c* може зустрітись довільна кількість інших понять:

$$CinC = C \to 2^c.$$

Аналогічно в тезі *t* можна зустріти будь-яку кількість понять:

$$CinT = T \to 2^c.$$

Таким чином відношення між поняттями можна описати наступним чином:

$$c_1 \in CinC(c_2) \to assoc_rel(c_1, c_2),$$

де c_1, c_2 – деякі поняття, $assoc_rel(c_1, c_2)$ – відношення асоціативності між двома поняттями.

Графічне відображення понять

Для більшої наочності сукупність понять та зв'язків між ними можна представити у вигляді графу асоціацій. Так, якщо певне поняття якимось чином пов'язане з іншим, тобто поняття №2

зустрічається у визначенні або тезах поняття №1, то між цими поняттями будується зв'язок асоціації, що вказує на те, що ці поняття пов'язані між собою (рисунок 3).

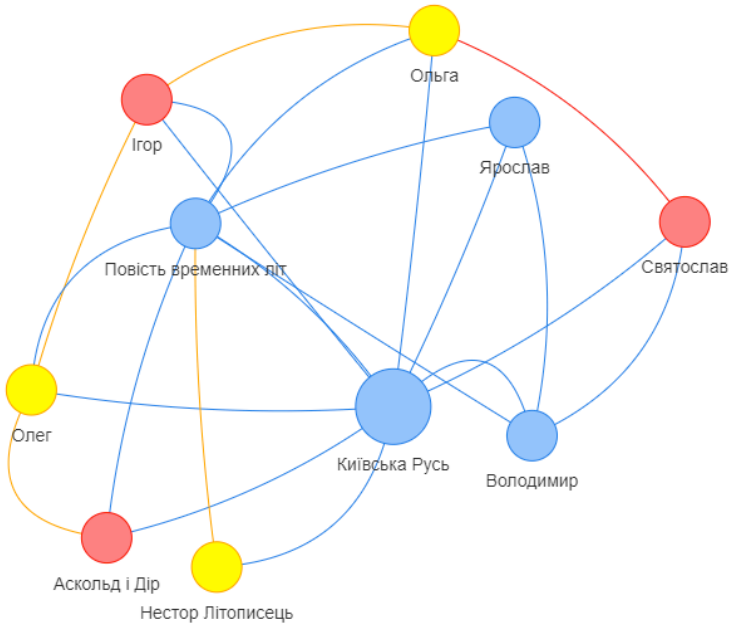


Рис. 3 – Граф асоціацій понять

Так само можна будувати карти понять, в яких зв'язки між поняттями будуть виділені через класи понять (рисунок 4).

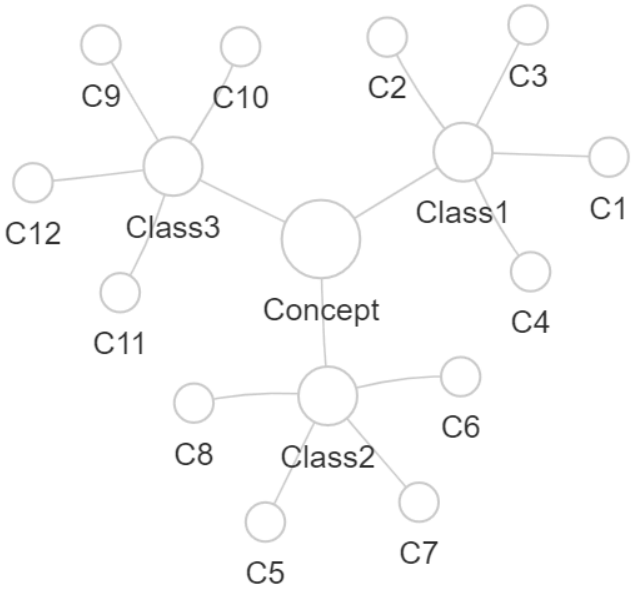


Рис. 4 – Карта понять

Програмна реалізація

Розроблена система являє собою веб-сайт з вільним доступом до нього.

На головній сторінці користувачу надається таблиця зі статтями, які присутні на сайті, а також список понять. За допомогою спеціальних інструментів, користувач може фільтрувати дані за часовим періодом.

Сторінка навчального матеріалу складається з трьох блоків: блок змісту навчального матеріалу, блок понять, які зустрічаються в змісті, та блок матеріалів, які запропоновані до вивчення. В змісті

виділяються поняття, які там зустрічаються. Ці поняття виводяться в блоці понять з коротким описом та посиланням на сторінку поняття.

Сторінка поняття являє собою список тез, які відносяться до поняття, та список навчальних матеріалів, в яких це поняття зустрічається. Схема взаємодії модулів системи відображена на рисунку 5.

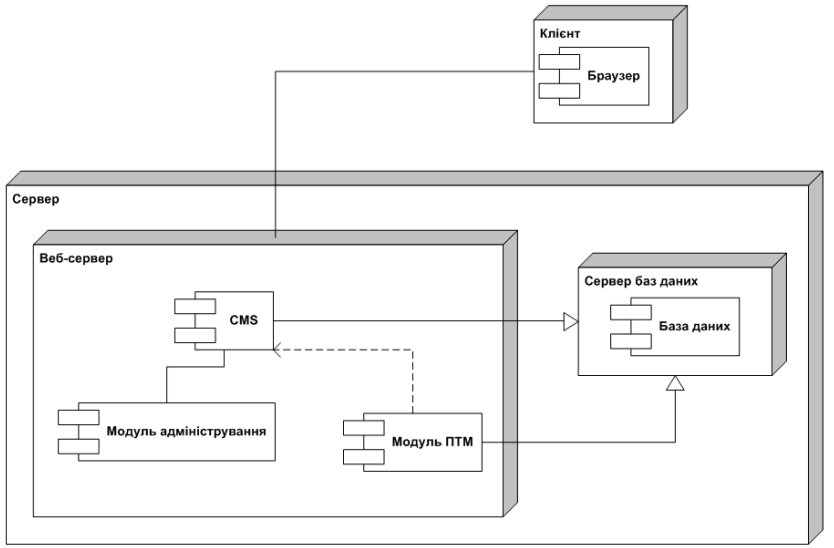


Рис 5 – Схема взаємодії модулів системи

Всі дані про навчальні матеріали, поняття, тези, та зв'язки між ними зберігаються в базі даних. Схему бази даних можна побачити на рисунку 6.

При створенні системи використовувались такі технології: PHP (Slim Framework), JavaScript (JQuery), HTML5, CSS3, MySql.

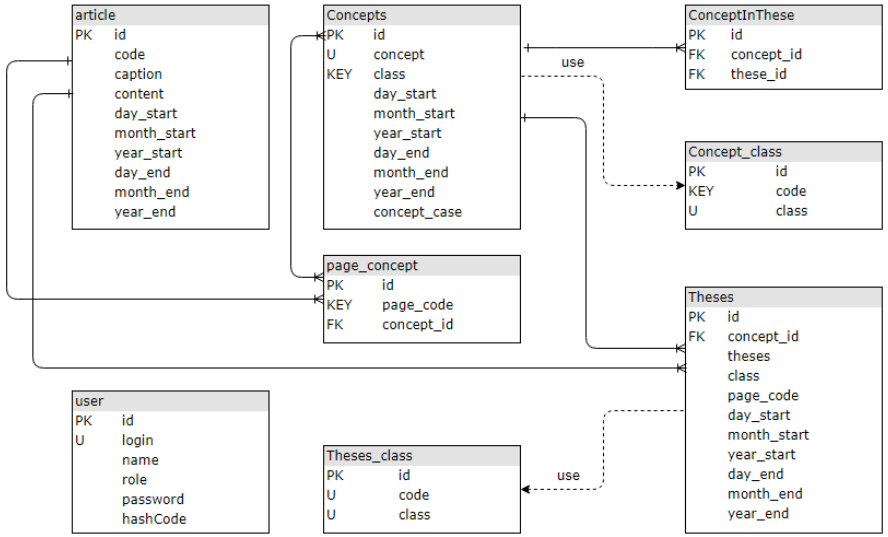


Рис 6 – Схема бази даних системи

Висновки

Створено формальну модель та здійснено на її основі програмну реалізацію системи формалізації контенту дисциплін історичного спрямування.

Набуло подальшого розвитку застосування понятійно-тезисної моделі для формалізації навчального контенту. Здійснено модифікацію ПТМ шляхом додавання до структурних елементів ПТМ властивостей часового періоду. Це дозволило формалізувати дані, характерні для навчальних матеріалів історичного напрямку. Модифікована ПТМ забезпечує гнучку вибірку необхідних навчальних ресурсів відповідно до історичного періоду та побудову інтерактивних карт понять для візуалізації навчальної інформації.

Реалізовано алгоритми побудови асоціативних зв'язків між поняттями на основі синтаксичного аналізу елементів ПТМ.

Розроблена система функціонує за адресою https://history-time.in.ua/.

Література

1. GoConqr. (2017). GoConqr - Changing the way you learn. [online] Available at: https://www.goconqr.com/
2. Histrf.ru. (2017). History of Russia - federal portal History.RF. [online] Available at: https://histrf.ru/
3. EdEra. (2017). EdEra – interactive online education. [online] Available at: https://www.ed-era.com/
4. Tytenko, S. (2010). Construction of didactic ontology based on the analysis of the elements of the conceptual-thesaurus model. Naukovi Visti NTUU KPI, 1(69), pp.82-87. (in Ukrainian).
5. Soroka, O. and Vasilyeva, I. (2017). Visualization of educational information. [online] Elib.bspu.by. Available at: http://elib.bspu.by/handle/doc/10693 [Accessed 20 Dec. 2017]. (in Russian).
6. Hollingsworth, M. (2016). Building a better eTextbook. Bulletin of the IEEE Technical Committee on Learning Technology, 18(2/3), pp.14-17.
7. xml.com. (2017). What Are Topic Maps. [online] Available at: https://www.xml.com/pub/a/2002/09/11/topicmaps.html.
8. Tonybuzan.com. (2017). Mind Mapping | Tony Buzan. [online] Available at: http://www.tonybuzan.com/about/mind-mapping/
9. Tytenko, S. (2009). Generation of test tasks in the system of distance learning based on the model of formalization of didactic text. Naukovi Visti NTUU KPI, 1(63), pp.47-57. (in Ukrainian).

Титенко С.В., Білоус А.О.

Інтерактивний доступ до професійно-навчальної інформації

Люди різних сфер діяльності потребують доступу до нової інформації. Зазвичай для цього використовуються пошукові системи. Найбільш поширеними проблемами, що можуть виникати під час пошуку, є отримання протилежних результатів через існуючі слова-омоніми та затрати часу для аналізу отриманої інформації (ознайомлення з нею, визначення чи відноситься дана інформація до необхідної галузі, поглиблений пошук за певним напрямом). Для більш ефективного пошуку необхідно виконувати аналіз запиту та результатів для представлення найбільш відповідної інформації, відображення результатів у більш зручному форматі для її сприйняття.

Професійно-навчальна інформація – це частина інформаційного простору, що містить впорядковані дані (визначення, статті, література, тощо), які можуть бути використані для навчання, роботи та досліджень у різних професійних галузях. Людина має проаналізувати отриману інформацію щоб зрозуміти яке з визначень відповідає її потребі, на що буде витрачено певний час.

Візуалізація - це процес представлення даних у вигляді зображення з метою максимальної зручності їх розуміння [1]. Згідно Б. Вюнше [2] процес візуалізації можна розділити на два етапи: кодування інформації за допомогою перетворення даних в візуальний

формат шляхом застосування різних візуальних атрибутів (форма, розмір, розташування, колір та ін.) і декодування інформації, що включає перетворення візуальних атрибутів в ментальний образ мозком, при цьому відбувається зіставлення шаблонів або патернів отриманих в процесі сприйняття зі знаннями, що зберігаються в довготривалої пам'яті, і подальша інтерпретація зображення (рис. 1).

Рисунок 1 – Процес візуалізації інформації (за Б. Вюнше [2])

Для представлення даних, що надаються інформаційним ресурсом, було обрано графовий метод візуалізації [3]. Перевагами такого способу представлення даних є можливість надати ключову інформацію та зв’язки у ній наочно. Так як робота заснована на інформаційних ресурсах, що містять професійно-навчальну інформацію – передбачається що дані мають певну структуру та є пов’язаними між собою. Тому при використанні графового способу візуалізації достатньо зручно буде відобразити структурні елементи (поняття, розділи) у якості вузлів графу, а зв’язки між елементами (посилання до уточнюючої інформації) як ребра графу.

У роботі розглядається професійно-навчальна інформація, що міститься на Web-ресурсах. Тобто передбачається що дані попередньо

структуровані за певними поняттями та мають між собою зв'язки, які проявляються у наявності активних посилань на сторінки, що містять роз'яснення або розширену інформацію про певні поняття, використані на сторінках ресурсу.

У зв'язку з цим завданням обробки інформації є:

– визначення зв'язків між елементами інформаційного простору;

– збір даних про зв'язки та їх кількість, що визначає вагомість зв'язку між елементами інформаційного простору;

– обробка зв'язків для оптимального їх представлення.

Так як інформаційні одиниці представлені у вигляді сторінок Web-ресурсу – для пошуку зв'язків між елементами HTML сторінка розбирається на посилання.

Посилання можуть бути не лише в межах системи, а й зовнішні – тому необхідно відфільтрувати лише внутрішні посилання і працювати з ними.

Інформаційний ресурс може містити безліч даних, які можуть бути розділені на певні групи за сенсом. Але побудова звичайного графу не передбачає поділу інформації на групи. Тому запропоновано спосіб пошуку тематичних груп серед елементів системи, що базується на зв'язках між ними.

Для поділу даних необхідно в першу чергу провести розбір системи та знайти усі структурні елементи та їх посилання на інші сторінки.

Серед знайдених елементів проходить пошук спільних посилань для кожного з них та, зрештою, знаходиться ступінь залежності одного вузла від іншого за формулою 1.

$$p_{ij} = \frac{intersection(re_i, re_j)}{ren_i + ren_j}, \quad (1)$$

де

pij – ступінь залежності вузлів i, j;

rei, rej – списки унікальних внутрішніх посилань;

intersection(rei, rej) – кількість співпадаючих посилань у списках унікальних внутрішніх посилань вузлів i, j.

reni, renj – кількість усіх унікальних посилань вузлів i, j.

Далі усі списки ступенів залежності для кожного структурного елементу перевіряються на вміст для того, аби відсіяти загальні посилання (ті посилання, що знаходяться у кожному структурному елементі). Загальні посилання у більшості випадків є посиланнями на службові сторінки. Так як головна сторінка, сторінка авторизації, інформація про компанію, контакти і таке інше. Для кожного вузла знаходиться найбільш вагоме посилання – для цього для кожного елемента відсортовується його список ступенів залежності від споріднених вузлів та виділяється елемент з найбільшим ступенем залежності. Список таких посилань обирається як перелік основних сторінок системи. Серед основних сторінок виділяються ті, що мають менше 50% спільних посилань з рештою зі списку основних сторінок – ці елементи і будуть «коренями» графа. Для кожного кореневого елемента створюється група. Решта сторінок розділяється на групи за

найбільш пов’язаним елементом з «кореневих». Якщо сторінка не має посилання на одну з кореневих сторінок – для неї встановлюється та ж група що й для найбільш пов’язаної з нею сторінки за ступенем залежності.

Система може опрацьовувати різноманітні Web-ресурси та генерувати для них граф контенту. Приклад опрацювання Web-ресурсу представлений на рисунку 2.

Рисунок 3.5 – Граф контенту ресурсу https://history-time.in.ua/

Для спрощення сприйняття та перегляду пов’язаної інформації – графовий метод представлення було вдосконалено поділом інформації на тематичні групи за пов’язаністю сторінок між собою. Такий граф контенту є більш наглядним способом демонстрації змісту

інформаційного ресурсу, ніж контекстне меню, яке зазвичай використовується у WEB-сайтах.

Список літератури

1. Темуров С. Й., Алимов Б. Н., Халимов У. Х. Основные визуальные способы представления и обработки учебной информации по математическим дисциплинам // Молодой ученый. — 2013. — №6. — С. 733-736.

2. Wünsche B. A survey, classification and analysis of perceptual concepts and their application for the effective visualisation of complex information // APVis '04 Proceedings of the 2004 Australasian symposium on Information Visualisation. 2004. Vol. 35. P.17–24.

3. Sarrafzadeh B., Vtyurina A., Lank E., Vechtomova O. Knowledge Graphs versus Hierarchies: An Analysis of User Behaviours and Per-spectives in Information Seeking / B. Sarrafzadeh, A. Vtyurina, E. Lank, O. Vechtomova // ACM SIGIR Conference on Human Information In-teraction and Retrieval. – 2016. – 337. – P. 91–100.

Шалденко О.В., Байда Д.В.

Нейронні мережі як засіб для семантичної сегментації зображення

У наш час все більше, для вирішення задач обробки зображень, використовується технології машинного навчання. Одним із основним напрямків роботи і досліджень є задача сегментації зображень. Зокрема можна виділити використання такого потужного інструменту як нейронні мережі для семантичної сегментації зображень.

Завдання сегментації - розбиття зображення на області, які відповідають різним об'єктам [1]. Семантична сегментація зображень - це поділ зображення на окремі групи пікселів, області, відповідні одному об'єкту з одночасним визначенням типу об'єкта в кожній області.

Нейронні мережі широко застосовуються для розпізнавання зображень без апріорного знання про сам предмет. Існує безліч модифікацій нейронних мереж, які оптимізовані для певних класів задач. Серед них можна виділити клас згорткових нейронних мереж, який можна ефективно застосувати для семантичної сегментації.

Згорткова нейронна мережа - спеціальна архітектура штучних нейронних мереж, націлена на ефективне розпізнавання зображень. Назву архітектура мережі отримала через наявність операції згортки, суть якої в тому, що кожен фрагмент зображення множиться на

матрицю (ядро) згортки поелементно, а результат підсумовується і записується в аналогічну позицію вихідного зображення.

На вхід згорткової нейронної мережі подається зображення, як масив значень освітленості кожного пікселю. На другому кроці, за допомогою різних ядер згортки формуються карти ознак ознак. Навчання нейронної мережі полягає у формуванні ваг ядер згортки, а ознаки самі формуються під час навчання.

Робота згорткової нейронної мережі зазвичай інтерпретується як перехід від конкретних особливостей зображення до більш абстрактних деталей, і далі до ще більш абстрактних, аж до виділення понять високого рівня [3]. При цьому мережа самоналаштовується і створює сама необхідну ієрархію абстрактних ознак, фільтруючи незначні деталі і виділяючи необхідні.

Для семантичної сегментації зображення можна застосувати FNN (Fully Convolutional Network). Fully Convolutional означає, що нейронна мережа складається з згорткових шарів, які не перекриваються між собою [2]. Додавання повністю підключеного шару дає змогу мережі сегментувати зображення, використовуючи загальну інформацію, без використання просторової інформації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

2. https://www.quora.com/How-is-Fully-Convolutional-Network-FCN-different-from-the-original-Convolutional-Neural-Network-CNN
3. https://habr.com/post/309508/

Шаповалова С.І., Бараніченко О.М.

Встановлення зв'язків між об'єктами логічної задачі

Існує широкий клас логічних задач, які необхідно розв'язувати у різноманітних сферах науки і техніки. Однак існуючі методи їх розв'язання або не задовольняють прийнятній швидкості, або не дають гарантовано точного результату. Використання нейронних мереж – найновіший підхід для розв'язання логічних задач. Через новизну цього підходу, існує мало результатів досліджень в цій області і досі не досягнуто рівня нейронних мереж, який задовольняє їх використанню в промислових системах. Тому створення концепції нейронних мереж для розв'язання логічних задач є актуальною задачею та має практичне застосування.

В роботі [1] наукова група корпорації Facebook надала класифікацію найбільш розповсюджених лінгвістичних задач, довела необхідність їх розв'язання та привела результати роботи системи, яка базується на використанні нейронних мереж з пам'яттю. При цьому було запропоновано набір прикладів [2], на яких проводилося навчання та тестування системи. В статті [3] було показано подібність логічних задач до задач типу XOR, приведено їх формалізацію та показано можливість їх розв'язання за допомогою елементів булевої алгебри. Однак недоліком такого підходу є більш складний, ніж у нейронних мереж, механізм модифікації постановки задачі, який

полягає у додаванні нових логічних термів, їх властивостей та значень. Тому необхідне вдосконалення нейронних мереж для розв'язання логічних задач.

Логічну задачу можна представити у вигляді об'єктів, які мають набір властивостей. Нехай, X – множина всіх властивостей об'єкта логічної задачі:

$$X = \{X_1, X_2, X_3, \dots, X_n\}, \quad (1)$$

де X_i (i=1..n) – і-та властивість об'єкта задачі.

В свою чергу, X_i має набір значень:

$$X_i \in V_i, \quad (2)$$

$$V_i = \{V_{i1}, V_{i2}, V_{i3}, \dots, V_{ik}\}, \quad (3)$$

де V_{ij} (j=1..k) – можливі значення властивості X_i.

Початкові умови задаються у вигляді відношень між значеннями властивостей.

Тоді розв'язок задачі для кожного об'єкту можна представити у вигляді

$$\{\langle X_1, \tilde{V}_1\rangle, \langle X_2, \tilde{V}_2\rangle, \dots, \langle X_n, \tilde{V}_n\rangle\},$$

де $\tilde{V}_i$ – значення і-ї властивості, яке не суперечить всім умовам.

Запропонована для розв'язання логічних задач нейронна мережа, за класифікацією [4], є неповною багатошаровою мережею прямого розповсюдження.

Наведемо приклад скороченої задачі Ейнштейна: X_1 – колір будику, X_2 – напій, відповідно V_1 = {синій, білий, жовтий}, V_2 = {кава, чай, вода}. Мережа для розв'язання такої задачі з 2 характеристиками,

кожна з яких має по 3 значення наведена на рисунку 1. Товщина ліній відповідає силам міжнейронних зв'язків. Для даної задачі це означає, що мешканець будинку білого кольору п'є каву.

Нейронна мережа складається з 3 шарів:

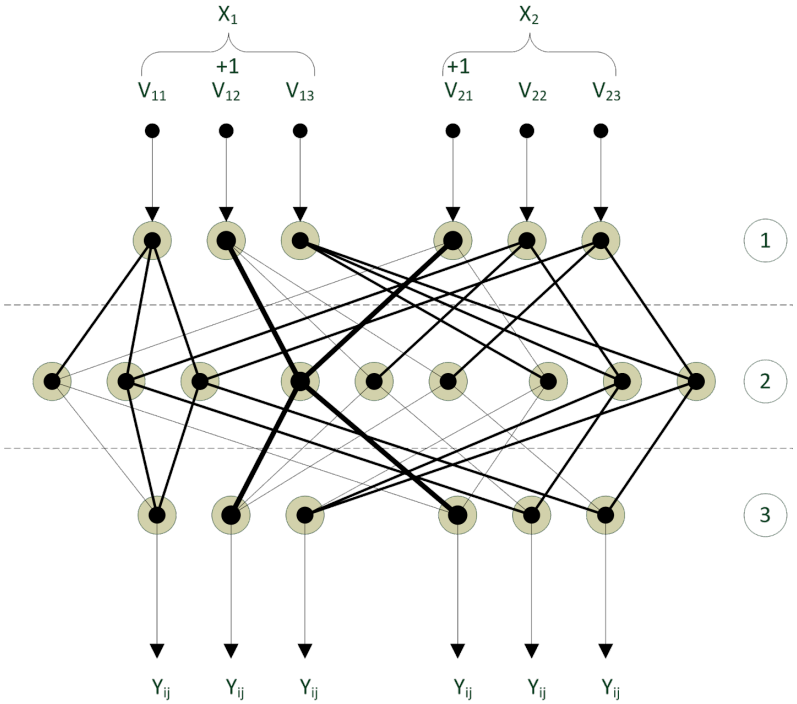


Рисунок 1. Архітектура нейронної мережі

1) Шар вхідних нейронів.

Призначення – передача сигналу до відповідних нейронів наступного шару.

2) Шар логічних нейронів.

Призначення – здійснення процесу навчання. Кожен нейрон цього шару зв'язує два нейрони з першого шару, що фактично означає зв'язок між двома властивостями різних класів.

3) Шар вихідних обчислювальних нейронів.

Призначення – визначення нейронів-переможців, які визначають зв'язки між властивостями логічної задачі. Кожен вихід Y_{ij} цього шару означає індекс k, який позначає значення властивості V_{mk}, що має найбільшу силу зв'язку з відповідним йому значенням V_{ij} властивості.

Навчання нейронної мережі має такий алгоритм:

1) Після того, як на вхідний шар поступили сигнали з розрідженого вектора значень, відбувається активізація тих нейронів, на які поступили сигнали високого рівня.

2) Активізовані нейрони подають сигнал на шар логічних нейронів. Для активізації нейрону цього шару необхідна подача на нього двох сигналів з високими рівнями.

3) Після спрацювання логічних нейронів, активізовані нейрони збільшують вагові коефіцієнти відповідних зв'язків, і таким чином укріплюють їх. Ті нейрони, які не були активізовані, послаблюють свої зв'язки з нейронами вхідного шару, імітуючи при цьому процес відмирання зв'язку.

Для виведення результатів після навчання лінійна комбінація векторів вхідних сигналів та встановлених сил зв'язків подається на шар вихідних обчислювальних нейронів. Вони, в свою чергу, визначають найсильнішу групу зв'язків та, в залежності від

постановки задачі, видають певний результат, наприклад індекс елементу з відповідної групи.

Висновки:

1. Проведено формалізацію логічних задач.

2. Запропоновано архітектуру нейронної мережі, яка здатна розв'язувати логічні задачі.

3. Показано алгоритми навчання та виведення результатів мережею.

Список літератури:

1. Weston J. Towards AI-complete questions answering : a set of prerequisite toy asks[Text] / J.Weston, A. Bordes, S. Chopra, A.M. Rush, B. van Merrienboer, A. Joulin, T. Mikolov// - Conference paper at ICLR, 2016.-P.14.

2. The bAbI project [Electronic resource] / [H. Miller, S. Chopra, M. Ranzato та ін.] // Facebook research. – 2015. – Mode of access: https://research.fb.com/downloads/babi/.

3. Filho W. Solving the Zebra Puzzle with Boolean Algebra [Електронний ресурс] / Wladston Filho. – 2017. – Режим доступу до ресурсу: https://code.energy/solving-zebra-puzzle/.

4. Хайкин С. Нейронные сети. Полный курс. / Саймон Хайкин. – Москва: Издательский дом "Вильямс", 2006. – 1104 с.

Шаповалова С.І., Колот С.С.

Системи моделювання Convolutional Neural Networks

Задача розпізнавання об’єктів, зокрема облич, є затребуваною в прикладних програмних системах. Для її розв’язання запропоновані спеціальні нейронні мережі Convolutional Neural Networks (CNN) [1]. Поширенню та вдосконаленню CNN сприяла велика кількість даних для їх навчання, наприклад, фотографій із соцмереж. Успішне використання цих нейронних мереж сприяло тому, що в останній час з’явився новий напрямок досліджень з їх обману шляхом маскування об’єктів. Тому задача створення середовища моделювання перешкод розпізнавання об’єктів є актуальною та має практичне застосування.

На сьогоднішній день розроблено велику кількість середовищ моделювання та навчання нейронних мереж, в яких реалізовані усі основні концепції, зокрема CNN. Найбільш відомими середовищами моделювання нейронних мереж є: Tensorflow [2], Caffe [3], CNTK [4], MXNET [5]. Кожна з цих систем має особливості з точки зору вимог до комп’ютерної техніки.

Для моделювання маскування об’єктів необхідно визначити найбільш відповідну систему моделювання CNN.

Мета статті – вибір системи моделювання Convolutional Neural Networks для тренування розпізнавання замаскованих об’єктів.

Система моделювання Tensorflow - це система машинного навчання, яка може працювати як на локальному комп'ютері або в хмарі, так і на смартфоні або навіть на мікрокомп’ютерах , таких як Rasberry Pi. Система Tensorflow розроблена командою Google Brain. Метою розробників є створення гнучких моделей нейронних мереж, які здатні самостійно отримувати ознаки, а також ефективно використовувати дані та обчислювальні потужності.

Середовище для глибинного навчання Caffe, розроблено Yangqing Jia в університеті Берклі. Середовище Caffe підтримує багато типів машинного навчання, націлених в першу чергу на вирішення задач класифікації та сегментації зображень. Caffe забезпечує моделювання Convolutional Neural Networks, Recurrent Convolutional Neural Networks, довгу та короткострокову пам'ять. Для прискорення навчання застосовується система графічних процесорів (GPU), підтримувана архітектурою CUDA, використовується бібліотека CuDNN від компанії Nvidia. До переваг Caffe можна віднести: велику кількість туторіалів, легкість освоєння, легкість розгортання.

Microsoft Cognitive Toolkit (CNTK) - безкоштовний загальнодоступний набір інструментів з відкритим кодом для моделювання нейронних мереж, розроблений компанією Microsoft. Це третій за популярністю спеціалізований пакет для глибинного навчання після TensorFlow і Caffe.

Apache MXNet - це середовище глибинного навчання, є портативним та легким, підтримує обчислення на декількох GPU.

Бібліотека MXNet надає не тільки середовище для глибинного навчання, але також і колекцію прикладів і інструкцій для побудови глибоких навчальних систем.

Дослідники компанії Microsoft для порівняння систем моделювання нейронних мереж відокремили такі характеристики: швидкість, точність, структура API, масштабованість, скоринг, можливість розширення, вбудовані модулі зчитування. В статті [6] доведено, що CNTK в порівнянні з TensorFlow показує кращі результати за більшістю представлених характеристик.

Дослідник компанії Microsoft A. Trivedi в роботі [7] представив формальне порівняння систем навчання нейронних мереж: MXNET, TensorFlow, Theano.

На основі цього представлення в даному дослідженні було порівняно системи моделювання нейронних мереж: Tensorflow, Caffe, CNTK, MXNET (Таблиця 1).

Таблиця 1 Порівняння систем моделювання CNN

Характеристика	**TensorFlow**	**Caffe**	**CNTK**	**MXNET**
Автор	Команда Google Brain	Yangqing Jia	Microsoft	Distributed (Deep) Machine Learning Community
Ліцензія на програмне забезпечення	Apache 2.0	BSD	MIT	Apache 2.0

Відкритий вихідний код	Так	Так	Так	Так
Платформа	Windows, Linux, Mac OS X	Windows, Linux, Mac OS X	Windows, Linux	Ubuntu, OS X, Windows, AWS, Android, iOS
Мова програмування	C++, Python	Matlab, Python C++	C#, C++, Python, Java	C++, Python, Julia, Matlab, JavaScript, Go, R, Scala, Perl
Підтримка CUDA	Так	Так	Так	Так
Підтримка паралельних і розподілених обчислень	Так	Так	Так	Так
Наявність попередньо навчених моделей	Ні	Так	Ні	Так
CNN	Так	Так	Так	Так

В дослідженні [6] перевагу віддано CNTK в порівнянні з TensorFlow однак CNTK має гіршу документованість ніж TensorFlow. Дослідження даного проекту показали, що з точки зору швидкості

розв'язання задачі MNIST [8] система моделювання нейронних мереж Tensorflow має кращі показники ніж Caffe. Завдяки цьому для дослідницьких цілей було обрано TensorFlow.

Список літератури

1. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel: Backpropagation Applied to Handwritten Zip Code Recognition, Neural Computation, 1(4):541-551, Winter 1989.
2. Caffe – Access mode http://caffe.berkeleyvision.org/
3. CNTK – Access mode https://github.com/Microsoft/CNTK
4. Deeplearning4j – Access mode https://deeplearning4j.org/
5. TensorFlow – Access mode http://www.tensorflow.org/
6. Reasons to switch from Tensorflow to CNTK. – Access mode https://docs.microsoft.com/en-us/cognitive-toolkit/reasons-to-switch-from-tensorflow-to-cntk
7. A. Trivedi. Deep Learning Part 1: Comparison of Symbolic Deep Learning Frameworks. – Access mode http://blog.revolutionanalytics.com/2016/08/deep-learning-part-1.html
8. THE MNIST DATABASE of handwritten digits. – Access mode http://yann.lecun.com/exdb/mnist/

Шаповалова С.І., Терпіль Д.О.

Вдосконалення графічної симуляції навколишнього середовища

Штучний інтелект все частіше використовується в прикладних програмних системах для миттєвого прийняття рішень в складних ситуаціях. Нейронні мережі використовують, наприклад, в бортових системах для аналізу зображення навколишнього середовища і визначення поточного кроку для досягнення кінцевої цілі.

Навчання нейронних мереж в реальних умовах дороге, повільне, а в випадках екстримальних ситуацій неможливе. Навчання доцільно проводити у штучностворених віртуальних середовищах, що імітують обмежений реальний фізичний світ. Тому створення та вдосконалення подібних специфічних середовищ є актуальною задачею та має практичне значення.

Враховуючи те, що штучний інтелект аналізує саме графічну інформацію для прийняття рішень, може бути використано спеціальний тип нейронної мережі – загорткові нейронні мережі. Інтелектуальний агент, який приймає покрокові рішення для досягнення кінцевої цілі, доцільно навчати за концепцією самонавчання. До нейронних мереж за цією концепцією навчання відносяться Deep Q-Network (DQN), які містять згорткову нейронну мережу для розпізнавання зображення та глибокі нейронні шари для

прийняття рішень. Deep Q-Network добре зарекомендував себе як механізм покрокового досягання цілі.

Концепція Deep Q-Network запропонована A. Mnih у 2012 році [1], полягає у поєднанні загорткової нейронної мережі та внутрішніх шарів нейроннів, які навчані з підкріпленням у внутрішніх шарах для прийняття інтелектуальним агентом покрокових рішень для досягнення цілі. Найбільш відомими прикладами програмних систем на основі Deep Q-Network є:

- AlphaGo для гри в го (розробленна Google) [2],
- OpenAI Gym платформа для розробки штучних інтелектів на основі навчання з підкріпленням та стала основою реалізації інтелектуального агенту бортової системи управляння автомобілю (розроблена компанією OpenAI) [3].

DeepMind – дослідницький підрозділ компанії Google, випробовував Deep Q-Network для створення інтелектуальних агентів, що приймають рішення в середовищах ігор Atari. Однак інтелектуальний агент у цих іграх знаходиться у середовищах, що засновані на обмежених формальних законах фізики. Крім цього візуалізація графічного середовища не відповідає реальним зображенням фізичного світу. Для деяких ігор, наприклад, Video Pinball, Ms. Pac-Man, BattleZone, IceHockey, представлення графічної інформації є дуже спрощеним, а фізичні властивості, відтворювані в грі, є неточними та неповними.

Для реалізації середовища навчання інтелектуальних агентів, доцільно використовувати існуючі ігрові рушії, які відтворюють базові функції реального середовища.

Наразі розроблено багато ігрових рушіїв для реалізації моделювання реального середовища. Однак їх функціональних можливостей недостатньо, адже відтворюються лише базові графічні властивості та закони фізики. Найбільш затребуваними ігровими рушіями є Unity [4] та Unreal Engine 4 (UE4) [5].

В обох рушіях є:

- статичні меши (static meshes) – єдині обєкти які можна рухати, повертати, і масштабувати;
- скелетні меши (skeletal meshes) - геометричні об'єкти, що прив'язані до кісток скелета і використовуються для анімації персонажів.

Меші однаково добре передають форму об’єктів відповідно до реального середовища та дозволяють відтворювати рухомі об’єкти.

Анімації, включені для скелетних мешів, також можна імпортувати. У Unity вони прикріплюються до імпортованого об'єкту, як кліпи анімації (animation clips). В Unreal Engine анімації прикріплюються до ігрового об’єкту та керуються послідовностями анімації (animation sequences). У першому рушії анімації керуються за допомогою контролерів анімації (animation controllers), а в другому за тим же принципом діють анімаційні Blueprints.

В обох рушіях є стейт-машини, що визначають переходи з одного стану ассету (налаштованого об’єкту) в інші. У UE4 система

називається Persona, а в Unity - Mecanim. Також можливе застосування скелетних мешів одного скелета до інших, але в Unity це в основному використовується для анімації гуманоїдів.

У рушії Unreal Engine є функції редагування анімації. Також в Unreal Engine вбудований дуже потужний редактор матеріалів, заснований, як і система Blueprints, на нодах.

Однак Unreal Engine через можливість працювати з полігонами, особливих алгоритмів випромінення променів потребує багато оперативної памяті пристрою, а також обрахункових можливостей процесора, що призводить до високих потреб потужностей пристрою моделювання.

Недоліком рушія Unity є те, що в ньому практично немає можливості редагування анімацій. Особливо це стосується рухів гуманоїдів. Прикріплений до об'єктів матеріал (наприклад, властивості та текстура дерева для об’єкту “стол”) додається в проект, але його властивості на зразок шейдера або текстур доведеться призначати вручну.

Для цього в Unity потрібно задати матеріалу шейдер і додати в його слоти текстури - карти шорсткостей, нормалей або дифузії. Власні шейдери доводиться розробляти самостійно або за допомогою сторонніх інструментів на кшталт Shader Forge або ASE.

Через гнучкість користування та набагато менше навантаження на систему, завдяки своїй архітектурі застосунків, Unity більш підходить для реалізації складних систем і середовища максимально наближеного до реального фізичного світу.

Однак в Unity представлено спрощені механізми для моделювання фізичних сил. Базовими реалізованими силами є сили тяжіння, тертя та прежності. Графічні можливості ігрового рушія дозволяють створювати тіні в реальному часі, відображати форми об’єктів та властивості матеріалів, проте не мають реалізації таких фізичних ефектів як пошкоджених поверхонь, та таких оптичних ефектів як відблиски та засвічення.

Тому необхідно доробити ігровий рушій Unity можливостями моделювання фізичного світу.

В роботі Unity вдосноаленно за рахунок:

- Розробки скриптів для вдосконалення моделювання сил тертя,
- Розробки скриптів, що дозволяють моделювати гідростатичні сили,
- Розробка шейдерів для створення відображення ефекту відблисків,
- Розробка шейдерів для відтворення графічних особливостей матеріалів.

Тому необхідно доробити ігровий рушій Unity можливостями моделювання гідростатичних сил; вдосконалити моделювання сили тертя; розробити шейдери для створення графічних перепон у вигляді відблисків, графічних особливостей матеріалів, та інших.

Література:

1. Google Research Blog From Pixels to Actions: Human-level control through Deep Reinforcement Learning . Google Inc. 2016 [Electronic resource]. - Access mode:

https://research.googleblog.com/2015/02/from-pixels-to-actions-human-level.html
2. DQN Human-level control through Deep Reinforcement Learning, 2016 [Electronic resource]. - Access mode: https://deepmind.com/research/dqn/
3. Geektimes GTA V подключили к платформе OpenAI Universe для обучения ИИ автопилота, 2016 [Електронний ресурс]. - Режим доступу: https://geektimes.ru/post/284522/
4. Unity 2018.1, 2018 [Electronic resource]. - Access mode: https://unity3d.com/ru
5. Unreal Engine 4 What is Unreal Engine 4 [Electronic resource]. - Access mode: https://www.unrealengine.com/en-US/what-is-unreal-engine-4

Щепак А. С.

Моделювання динаміки розповсюлження забруднень

Моніторинг стану навколишнього середовища є важливою складовою розвитку держави. Однією з головних його частин є спостереження за антропогенним впливом людини, особливо на територіях великих промислових об'єктів. Оскільки уникнути шкідливих результатів їх діяльності немає можливості, необхідно контролювати кількість викидів забруднюючих речовин у повітря, воду та землю. Крім того, важливо вживати заходів для мінімізації їх та шкоди, що вони несуть. Враховуючи те, що більшість таких підприємств знаходяться у приватній формі власності, держава має проводити підрахунок економічної шкоди від їх роботи для подальшого стягнення податків. На даний час на загальнодержавному рівні не існує автоматизованих систем, що б виконували такі дії, що ускладнює ці функції.

Ціллю цієї роботи є створення такого програмного продукту, який би був універсальним для всіх підприємств київської області. Інформаційна система має опрацьовувати всі форми викидів, бути єдиним інструментом оцінки для штатного та надзвичайного станів роботи підприємств. Дані для податкових та інших зборів беруться з державних документів, постанов Кабінету Міністрів України, що робить результати, опрацьовані в програмі, повністю відповідними до закону та робить можливим їх використання в якості офіційних.

Вибір території Київської області не випадковий – це одна з найбільших промислових агломерацій в Україні. Повітря області забруднюють десятки великих підприємств, підземні та наземні води не придатні для використання людьми, землі не придатні

для ведення сільського господарства на десятиріччя вперед. Також, значним фактором, що впливає на життя людей, що проживають на цій території, є аварія на ЧАЄС. Оскільки період напіврозпаду деяких речовин, що потрапили в навколишнє середовище під час трагедії значно перевищує сотні років, шкода для здоров'я вже народжених і майбутніх поколінь дуже значна. Враховуючи всі ці фактори, необхідно всіма можливими засобами покращити умови проживання людей, збільшити середню тривалість життя, зменшити кількість хвороб. Для цього необхідно значно обмежувати нові викиди та максимально ліквідувати наслідки старих. Це потребує значних грошових витрат.

Всі вхідні дані обраховуються за стандартними для державних органів методиками, що необхідно для можливості їх використання у якості офіційних. Розрахунки ведуться для кожної з форм викидів – у повітря, воду та землю; для надзвичайних та штатних ситуацій; понаднормових та не перевищуючих законних вимог. Ці та багато інших факторів мають бути розраховані окремо, проте кінцевий результат має бути узагальнено для легкості подальшого використання.

Розрахунки дозволяють отримати дані, що корисні як державним органам, так і власникам підприємств, що було проаналізовано. Першим це дозволяє з легкістю проводити збір податків; контролювати кількість забруднюючих речовин, які потрапляють до навколишнього середовища; знаходити підприємства, що більше за всіх шкодять екології; розрахувати для кожної людини шкоду для здоров'я залежно від зони проживання; вирахувати економічні втрати для держави від скорочення середньої тривалості життя людей, що піддаються шкідливому впливу та інше. Власникам підприємств будь-якої форми власності це дозволяє побачити можливі зміни на підприємстві – переобладнання, модернізація, відкриття нових ліній виробництва. По результатам, отриманим із програми можна буде оцінити, який ефект це спричинить як для збільшення

прибутків підприємства, так і з точки зору покращення екологічної ситуації в регіоні.

З усього вищевказаного зрозуміло, що реалізація інформаційної системи буде корисна і державі, і приватним компаніям. Зменшення годин для виконання стандартних операцій, збільшення економічної вигоди, стимулювання інновацій та інше. Але значно головнішим за все це є можливість покращення екологічного стану навколишнього середовища, здоров'я значної кількості жителів Києва та області. В перспективі це значно покращить якість та тривалість життя людей.

Вирахувавши ефект від зміни тривалості життя людини, програма дозволяє зрозуміти, як ефективно використовуються ресурси та наскільки вигідніше подовжувати тривалість життя людей, ніж отримувати збитки від поганої екології.

Таким чином розроблено додаток для моніторингу екологічних показників як у режимі реального часу, так і при загрузці даних зі сховища. Програмний комплекс здатен опрацьовувати як штатні, так і надзвичайні ситуації, швидко їх аналізувати та видавати висновки і рекомендації.

Перелік використаних джерел

1. GIS словник [Електронний ресурс]. – Режим доступу: http://support.esri.com/en/knowledgebase/GISDictionary/term/kriging

2. ESRI – ArcInfo [Електронний ресурс]. – Режим доступу: http://www.esri.com/software/arcgis/arcinfo.

Iryna Husyeva

Information and analytical system for innovation analysis

The main goal of the system is support of scientific decision making to manage the innovative development of regions, industries, enterprises of Shandong Province on the basis of macroeconomic analysis of innovative indicators.

On the early stage of research is necessary to identify and classify the following indicators:

• target parameters (macro-indicators), necessary for analysis the level of innovative development of science;

• criteria for the level of innovative development of regions, sectors of the national economy, enterprises of Shandong Province;

• indicators of economic, financial, scientific, demographic and innovative development of Shandong Province;

• information technologies and standards for the allocation, storage and processing of analytical data in a distributed environment;

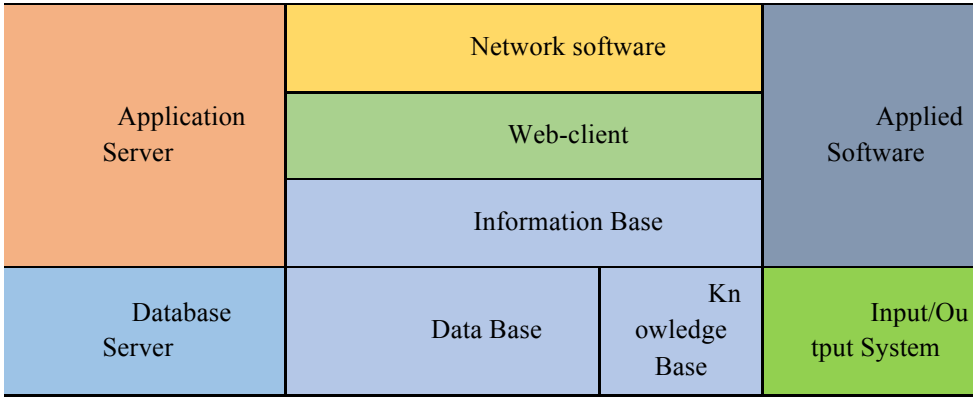
• potential users of the system (government departments of Shandong Province) processing of analytical data of innovation development.

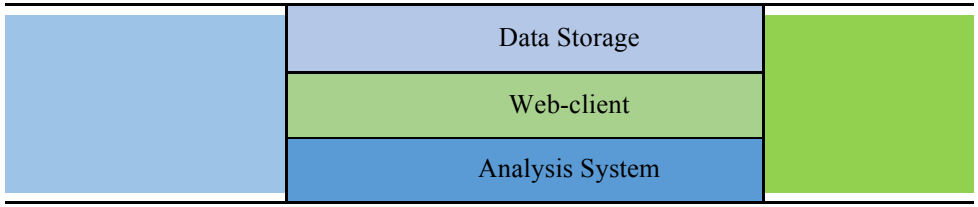
According to the goal of the System the Functions should be divided as followed:

1. Collection, registration of indicators of all kinds of innovation activities in Shandong Province
2. Monitoring of indicators of economic, financial, scientific, demographic and innovative development of Shandong Province
3. Collection, integration of information relating to the innovation activities of both Shandong and China

4. Classification of key innovative directions of science, the dynamics of their development, taking into account the level of novelty
5. Analytical studies of the impact of innovative projects on various areas of activity (economy, defense, health, technology, etc.)
6. Definition of interrelations of indicators of innovative activity with indicators of economic, financial, scientific, demographic development of a province of Shandong
7. Construction of ratings of innovation development in different areas of activity
8. Forecasting the modeling of innovative development of the Shandong Province
9. Visualization of monitoring results
10. Administration and management of the system

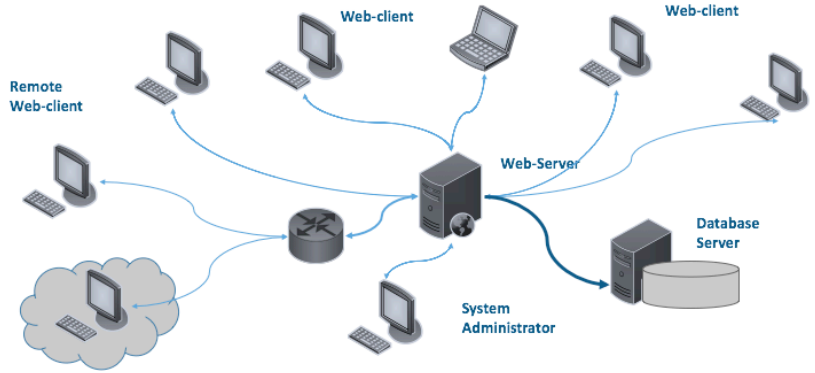
To fulfill the necessary functions of the System the modules of Informational and analytical system are Data Collection Module, Data Storage and Analysis Module and Visualization Module. All data is stored in the Information Base which contains of Data Base, Knowledge Base and Data Storage. System architecture (see Picture 1) describes structuring principles that enables a system to be comprised of a set of simpler systems each with its own local context that is independent of but not inconsistent with the context of the larger system as a whole.





Picture 1 – System architecture

Physically the System is built as showed on the Picture 2.



Picture 2 – System prototype

It is a three-tier architecture which provides the remote access to the system, tools for collaboration inside the enterprise and perfect means of communication.

The methodological base of the project include such methods as: system analysis, complex information network theory, Big Data analysis, DataMining tools for searching hidden dependencies between indicators of innovation activity (Correlation Analysis, Regression Analysis, Factor Analysis, Cluster Analysis, Discriminant Analysis, Tree Decision and Neural networks and deep learning).

Besides for qualitative analysis it is decided to use such frameworks as Grounded Theory, Distributed Cognition, Activity Theory.

Grounded theory[3] involves the progressive identification and integration of *categories of meaning* from data. It is both the process of category identification and integration (as *method*) and its product (as *theory*). Grounded theory *as method* provides us with guidelines on how to identify categories, how to make links between categories and how to establish relationships between them. Grounded theory *as theory* is the end-product of this process; it provides us with an explanatory framework with which to understand the phenomenon under investigation. To identify, refine and integrate categories, and ultimately to develop theory, grounded theory researchers use a number of key strategies, including *constant comparative analysis, theoretical sampling* and *theoretical coding*.

The Distributed Cognition[4] approach emphasizes the distributed nature of cognitive phenomena across individuals, artefacts and internal and external representations in terms of a common language of 'representational states' and 'media'. In so doing, it dissolves the traditional divisions between the inside/outside boundary of the individual and the culture/cognition distinction that anthropologists and cognitive psychologists have historically created. Instead, it focuses on the interactions between the distributed structures of the phenomena that is under scrutiny.

In research that studies the complexities of real world situations, such as modern workplaces, communities groups or places of learning, Activity Theory[5] provides a language and a set of frameworks for making sense of what is discovered about the situation through observation, interviews and other methods. Using the Activity Theory lens for research takes activity as the unit of analysis, where activity is defined by the 'dialectic relationship between subject and object' , in other words, 'who is doing what for what purpose'. In most complex situations, there are many dynamic inter-related activities forming what could be seen as a system of activities.

[3] http://www.mheducation.co.uk/openup/chapters/9780335244492.pdf
[4] Yvonne Rogers A Brief Introduction to Distributed Cognition http://mcs.open.ac.uk/yr258/papers/dcog/dcog-brief-intro.pdf
[5] http://ro.uow.edu.au/cgi/viewcontent.cgi?article=1405&context=buspapers

Summing everything up it is necessary to accent that the further development of the System is up to a class of advanced analytics systems using predictive modeling methods, as well as methods for generating dynamic scenarios for analyzing information in real time.

Наукове видання

СУЧАСНІ АСПЕКТИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Збірник наукових праць V науково-практичної дистанційної конференції молодих вчених і фахівців з розробки програмного забезпечення

15 травня 2018 р.

Формат 60x84 1/16. Папір офсетний,
Умов. друк. арк. 32,5. Гарнітура Times New Roman.
Зам. No 1657.
Тираж 300 прим.

Видавець: Чабаненко Ю. А.
Свідоцтво про внесення
до Державного реєстру видавців
серія ДК No 1898 від 11. 08. 2004 р.
Україна, м. Черкаси, вул. О. Дашкевича, 39.
Тел: 0472/56-46-66; 093/788-99-99
E-mail: office@2upost.com

Друк ФОП Чабаненко Ю.А.
Україна, м.Черкаси, вул. О. Дашкевича, 39.
Тел: 0472/56-46-66; 093/788-99-99
E-mail: office@2upost.com