Національний технічний університет України
«Київський політехнічний інститут»

Факультет (інститут) ННК “Інститут прикладного системного аналізу”
(повна назва)

Кафедра Системного проектування
(повна назва)

Рівень вищої освіти Перший(Бакалаврський)
(перший (бакалаврський), другий (магістерський або спеціаліста)

Спеціальність 7.05010102, 8.05010102 Інформаційні технології проектування
7.05010103, 8.05010103 Системне проектування
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри _____
А.І.Петренко
(підпис) (ініціали, прізвище)
«___»_____2016 р.

ЗАВДАННЯ на дипломний проект (роботу) студенту
Дудку Дмитру Івановичу
(прізвище, ім’я, по батькові)

1. Тема проекту (роботи) «Технології семантичного веб для задач бізнес-аналітики»

керівник проекту (роботи) Булах Богдан Вікторович, к.т.н., ст.в. ,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від 12 травня 2016 р. № 50-ст
2. Строк подання студентом проекту (роботи) _____
3. Вихідні дані до проекту (роботи) _____
    1. Мова програмування - python
    2. Форма реалізації - веб-додаток
    3. Можливість будувати SPARQL-запити та візуалізовувати дані.
4. Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити)
    1. Технології семантичного веб
    2. Технології Business Intelligence
    3. Проектування BI-платформи
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслеників, плакатів тощо)
    1. Порівняння можливостей BI-систем-плакат
    2. Основні задачі бізнес-аналізу-плакат

3. Архітектура додатку-плакат

6. Консультанти розділів проекту (роботи)*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 01.02.2016

Календарний план

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Отримання завдання	01.02.2016	
2	Збір інформації	15.02.2016	
3	Дослідження варіантів реалізації та вибір архітектури та програмних засобів для розробки додатку	01.03.2016	
4	Розробка і удосконалення алгоритму реалізації програмного продукту	30.03.2016	
5	Розробка основного программного модулю	29.04.2016	
6	Розробка плану тестування програмного продукту	10.03.2016	
7	Підбір тестових даних та тестування програмної реалізації	20.05.2016	
8	Оформлення дипломної роботи	31.05.2016	
9	Отримання допуску до захисту та подача роботи в ДЕК	05.06.2016	

Студент _____ Дудко Д.І.

(підпис) (ініціали, прізвище)

Керівник проекту (роботи) _____ Б.В. Булах

(підпис) (ініціали, прізвище)

* Консультантом не може бути зазначено керівника дипломного проекту (роботи).

АНОТАЦІЯ

бакалаврської дипломної роботи Дудка Дмитра Івановича

на тему «Технології семантичного веб для задач бізнес-аналітики»

Метою роботи є застосування технологій семантичного веб для вирішення задач бізнес-аналітики. В роботі розглянуто загальну архітектуру платформ для бізнес-аналіза, розроблено нову архітектуру зважаючи на необхідність інтеграції з BI, розроблено прототип семантичної системи для задач бізнес-аналітики. Дана робота демонструє переваги застосування нових технологій для бізнеса та зосереджена на оптимізації бізнес-процесів.

Загальний обсяг роботи: 72 сторінок, 14 рисунків, 7 таблиць, 16 бібліографічних посилань.

Ключові слова: семантичний веб, бізнес-аналіз, RDF, OWL, SPARQL, аналіз даних.

АННОТАЦИЯ

бакалаврской дипломной работы Дудко Дмитрия Ивановича

на тему «Технологии семантического web для задач бизнес-аналитики»

Целью работы является применение технологий семантического веб для решения задач бизнес-аналитики. В работе рассмотренная общая архитектура платформ для бизнес-анализа, разработана новая архитектура учитывая необходимость интеграции с BI, разработан прототип проектируемой системы. Данная работа демонстрирует преимущества применения новых технологий для бизнеса и сосредоточена на оптимизации бизнес-процессов.

Общий объем работы: 72 страница, 14 рисунков, 7 таблиц, 16 библиографических ссылок.

Ключевые слова: семантический веб, бизнес-анализ, RDF, OWL, SPARQL, анализ данных.

ABSTRACT

a bachelor's degree work of Dudko Dmytro Ivanovich
entitled "Semantic web technologies for business analysis"

The aim of this work is the application of semantic web technologies for solving problems of business intelligence. The paper discusses the General architecture platforms for business Analytics, designed a new architecture given the need to integrate with BI, developed a prototype of semantic system for problems business intelligence. This work demonstrates the benefits of applying new technology for business and focus on optimizing business processes.

Total volume: 72 pages, 14 figures, 7 tables, 16 bibliographic links.

Key words: semantic web, business analysis, RDF, OWL, SPARQL, data analysis.

Зміст

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

RDF	Спосіб подання знань в децентралізованій системі; основна технологія Семантичного Вебу, яка дозволить комп'ютерним програмам користуватися всією структурованою інформацією, розподіленої по вузлам Інтернету.
OWL	Мова опису онтологій для семантичної павутини.
SPARQL	Мова запитів до даних, представлених з допомогою моделі RDF; протокол передачі запитів та відповідей.
Онтологія	Концептуальна схема, яка формалізує деяку галузь знань. Інформація як ієархічна структура понять.
Business Intelligence(BI)	Процес перетворення даних в інформацію, тобто у нові знання, які можуть бути використані для збільшення ефективності та конкурентноздатності підприємства.

ВСТУП

Сьогодні Business Intelligence (BI) сформувалося в самостійний напрям індустрії інформаційних технологій (IT). BI є споживачем та сферою застосування різноманітних нових технологій.

За даними TAdviser, ринок BI-систем країн СНД (ліцензії та послуги) в 2012 році подолав позначку в 1404 USD, що на 15% більше, ніж роком раніше. Темп зростання ринку сповільнився (35% в 2011 році), але все ще залишився досить значним, випереджаючим темпи зростання ринку вцілому. Це свідчить про великий інтерес до цього напряму та його потенційний вплив на інші напрями IT.

Ціль BI – перетворення даних у знання, а знань – у бізнес-дії для отримання певної користі [1]. Тому доцільно проаналізувати перспективи інтеграції BI з сучасними технологіями керування розподіленими знаннями, а саме – з Semantic Web.

На жаль, промислові застосунки не завжди вчасно впроваджують наукові досягнення: в сучасних комерційних BI-застосунках технології та стандарти Semantic Web практично не використовуються, а замість цього здійснюються спроби заново розробити засоби подання знань та метаописів.

Як BI, так і Semantic Web нині належать до пріоритетних напрямів IT, проте розвиваються паралельно, незважаючи на значну подібність задач, які вони вирішують. Тому важливо проаналізувати перспективи їх інтеграції та можливість взаємного впливу методів та застосувань.

Метою дипломної роботи є дослідження способів інтеграцій технологій Semantic Web в BI-застосунках, проектування архітектури сучасної BI-платформи та реалізації її прототипу. Поставлена мета вимагає вирішення наступних наукових задач:

1. аналіз можливостей сучасних BI-платформ.
2. аналіз способів інтеграції технологій Semantic Web в BI-застосунках

3. створення прототипу використовуючи результати попереднього аналізу

Об’єктом дослідження є аналіз способів і засобів використання RDF-сховищ у BI системах. Предметом дослідження є бізнес-аналіз та семантичні технології.

Досягнення поставленої мети реалізовано з використанням мови програмування Python, існуючих для нього інструментів RDFlib(для здійснювання SPARQL-запитів до бази), веб-фремворк Flask(для обробки запитів користувачів системи).

Наукова новизна дипломної роботи полягає в тому, що було розроблено концепції побудови системи, яка об’єднує технології семантичного вебу та бізнес аналізу, що дозволяє робити нетипові запити до бази знань та візуалізовувати дані для подальшого їх аналізу.

Потенційні застосування та практична цінність результатів дипломної роботи:

1. Спроектована система може стати гарною альтернативою сучасним BI- системам.
2. Сформульовані основні засоби інтеграції технологій Semantic Web.

1. ТЕХНОЛОГІЇ SEMANTIC WEB

1.1 Вступ

Феномен World Wide Web став можливий тільки завдяки практичному використанню набору широко поширених стандартів на різних рівнях, що забезпечило інтероперабельність даних. Сучасна тенденція розвитку Інтернету полягає в переході від документів, "що читаються комп'ютером" (machine readable) до документів, які "комп'ютер розуміє" (machine understandable).

Web розроблявся, як інформаційний простір, корисний не тільки для комунікації людини з людиною, але і як простір, в якому зможуть ефективно співпрацювати і комп'ютери. Одне з головних перешкод на шляху до цього полягає в тому, що більша частина інформації в Web призначена для її розуміння людиною. Очевидно, що така структура даних не може бути зрозумілою для веб-робота, що її проглядає. Підхід Semantic Web базується на розробці мов, для вираження інформації у формі, придатній для машинної обробки.

Semantic Web являє собою мережу інформаційних вузлів, які пов'язані один з одним таким чином, щоб наявна інформація могла легко оброблятися комп'ютером. Його можна розглядати як ефективний спосіб представлення даних у Всесвітній павутині, або як глобально пов'язану базу даних. Даний проект пропонує реалізацію повної системи з автоматизованого створення та зберігання семантичного ядра контенту, наданого у Всесвітній павутині.

Проект Semantic Web - це спроба зібрати всі сталі ідеї і зробити так, щоб вони змогли працювати разом всередині мережі Інтернет. Для досягнення цієї мети використовуються стандарти, які розроблені не тільки консорціумом W3C, а й іншими організаціями. Мета проекту - дозволити взаємодіяти цим стандартам між собою, всередині децентралізованої системи, без втручання людини.

Semantic Web - це розширення поточного Web, в якому інформація надається з добре певним значенням, яке краще дозволить комп'ютерам і людям

працювати разом. Його ідея в тому, щоб мати дані в Web, визначені і пов'язані між собою таким чином, щоб їх можна було використовувати для більш ефективного дослідження, автоматизації, інтеграції та повторного використання в різних додатках ... ці дані можуть бути загальнодоступними і обробленими, автоматичними засобами так само, як і людьми [2]. У рамках даного проекту задіяні такі передові технології, як агентно-орієнтовний підхід у програмуванні [10] , онтології [15, 16], XML, RDF, та інші. В даний час поширюється використання Web-агентів (у спрощеному вигляді веб-сервісів), які розробляються як для окремих завдань, так і для створення ядра Semantic Web [2].

Як зазначив професор Джон Сова, - Semantic Web - багато-дисциплінарна тема, яка об'єднує теорії та методи трьох областей:

- логіка - формальні структури і правила логічного висновку;
- онтології - опис типів сутностей, які відносяться до предметної області;
- теорія моделей.

Інтернет - це мережа комп'ютерів, об'єднаних каналами, які використовують протоколи (TCP / IP) для зв'язку між собою. Web - це мережа сайтів, які використовують гіперпосилання для переходів між сторінками [9]. Традиційний Web базується на мові розмітки документів HTML. HTML-сторінка описує форму подання інформації в Web- браузері, а ця мова важко піддається автоматичному змістовному аналізу. Автоматизувати навіть такі тривіальні завдання, як пошук людей, проектів, програм в Інтернеті - неможливо. Наступний етап розвитку Інтернет - Semantic Web - представляє собою перехід на новий рівень представлення даних - рівень знань та автоматизованої обробки. Технологія Semantic Web дозволить комп'ютеру інтерпретувати інформацію, представлену в Web, нарівні з людьми, для чого й розроблена графова модель опису ресурсів RDF (Resource Description Framework).

У загальному вигляді Semantic Web (за Тіму Бернерс-Лі) - це:

- інтероперабельність даних між програмними додатками та організаціями;
- набір інтероперабельних стандартів для обміну знаннями;
- архітектура для взаємопов'язаних спільнот та словників [3].

Архітектура Semantic Web

З точки зору архітектури Semantic Web можна розглядати, як три яруси (рисунок 1.1): базис, який складається з унікальної глобальної ідентифікації ресурсу, метаданих для декларування фактів про ресурси, і спільної мови для вираження метаданих і знань, що реалізовані за допомогою онтологій, для загальнодоступного розуміння і загального словника метаданих, і правил для додавання нових метаданих та знань; базовий сервіс, наприклад, логічний висновок і запити до метаданих, і онтологія, роз'яснення таких висновків, управління довірою, агенти, пошукові системи, онтології; сервіси додатків, наприклад сервіс агентства подорожей.

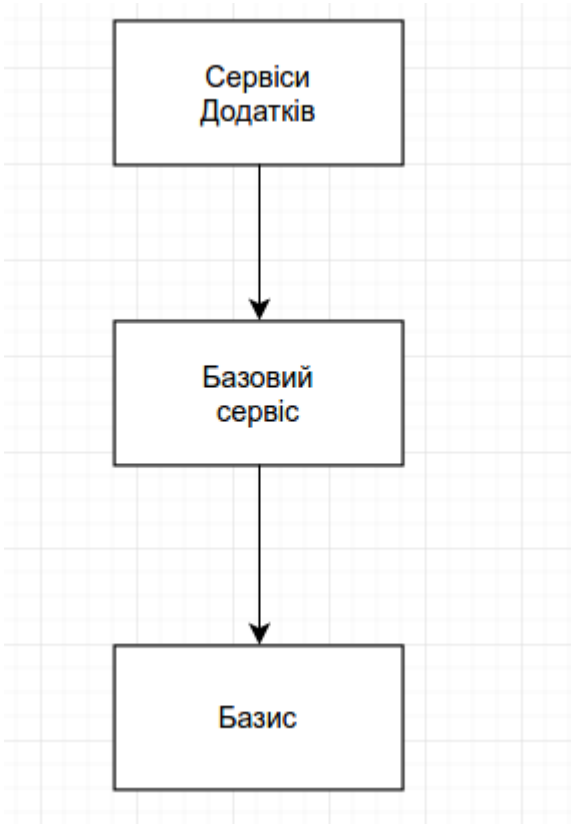

Рисунок 1.1. Три яруси мережі Semantic Web

Технології, які задіяні у розробці Semantic Web:

- Семантичний пошук;
- Питально-відповідні системи;
- Агенти;
- Об'єднання знань (інтеграція баз даних);
- Проникливі обчислення [2].

У 1998 році Тім Бернерс-Лі запропонував наступний логічний план побудови Semantic Web [1]:

1. Синтаксис для представлення знань, який використовує посилання на онтології
2. (RDF);
3. Мова опису онтологій (OWL);
4. Мова опису веб-сервісів (WSDL, OWL-S);
5. Інструменти читання / розробки документів Semantic Web (Jena, Haystack, Protege);
6. Мова запитів до знань, які записані в RDF (SPARQL);
7. Логічний висновок знань (знаходиться на етапі обговорення);
8. Семантична пошукова система (наприклад, SHOE).

Базова модель Semantic Web (пиріг Тіма) показана на рисунку 2 [4].

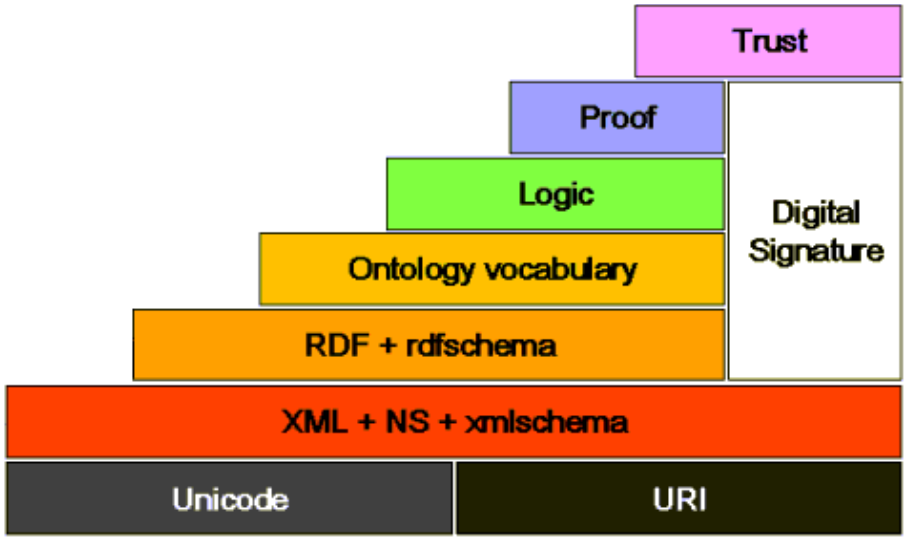

Рисунок 1.2. Базова модель Semantic Web

Фундаментальними основами Semantic Web є:

- графова модель представлення на пів структурованих даних (OEM, Lore);
- формальна логіка (логіка першого порядку, бази знань, фрейми);
- архітектура WWW (URI / IRI, Unicode, XML, HTTP);
- криптографія з відкритим ключем.
- Розглянемо структуру базової моделі Semantic Web більш детально в наступних пунктах.

1.1 URI – універсальний ідентифікатор ресурсів

В Web для ідентифікації елементів використовуються "Уніфіковані ідентифікатори ресурсів", або скорочено URI (Uniform Resource Identifier). URI можна присвоїти до чого завгодно, і якщо ця сутність має URI, то про неї можна говорити, що вона знаходиться "в Web": це може бути людина, книга, абстрактна концепція, тобто все, що має назву.

URI є базисом Web. URI - це компактний рядок символів, який використовується для ідентифікації абстрактних або фізичних ресурсів» [8]. Однією з форм URI є URL (Uniform Resource Locator), уніфікований покажчик ресурсу. URL - це адреса, за якою завантажується Web-сторінка.

Також необхідно вказати, що в початковій базової моделі в нижньому ярусі, було вказано ще й базове кодування – тобто, загальний для всіх принцип кодування всіх можливих символів багатьох мов - кодова таблиця UNICODE.

За синтаксисом URI стежить комітет IETF. Документ, який опублікований цим комітетом RFC 2396, є спільною специфікацією URI. Консорціум W3C підтримує список схем URI. У 2005 році на зміну URI був запропонований інтернаціоналізувати ідентифікатор ресурсу - Internationalized Resource Identifiers (IRI), що ідентифікує абстрактний або фізичний ресурс будь-якою мовою світу. URI можуть містити тільки латинські символи та знаки пунктуації з набору символів US-ASCII (в цілому близько 60 символів).

Для забезпечення принципів інтернаціоналізму, збереження «читабельності» для людини, в IRI було запропоновано, що ці ідентифікатори можуть містити будь-які символи Юнікоду (Unicode/ISO10646) у чистому вигляді, без будь-якого кодування. IRI не обмежують права інших мов і ведуть до більш високого ступеня рівноправності користувачів Інтернету. У майбутньому ідентифікатори IRI покликані замінити URI. Зазвичай посилання URI є відносною для будь-якого документа, в якому вона знайдена.

Якщо, наприклад, проглядається документ з базовим URI http://exslt.org/math/min/math.min.template.xsl, і в ньому виявляється URI-посилання .. / .. / random / random.xml, то вона призведе до даного документу з адресою http://exslt.org/random/random.xml. У форматі HTML є можливість винести базовий елемент в заголовок документа, щоб перекрити базовий URI. Базова специфікація XML (XML Base) забезпечує еквівалентну форму в XML.

1.2 Розширювана мова розмітки (XML)

XML [6] (eXtensible Markup Language) являє собою дуже простий і при цьому потужний, і гнучкий текстовий формат, для опису документів довільної структури. XML був розроблений і затверджений в якості стандарту в ProductID в 1998 р Консорціумом W3C, для спрощення реалізації, а також для забезпечення інтероперабельності між SGML і HTML. Він є підкласом мови SGML, однак більш простий для розуміння і обробки.

Опції XML:

- Подання синтаксису для інших мов розмітки;
- Семантична розмітка Web-сторінок. XML-представлення може використовуватися на Web-сторінці разом з таблицею стилів XSL, що визначає коректний вивід на екран різних елементів;
- Єдиний формат обміну даних. XML-представлення може передаватися між двома застосуваннями, як об'єкт даних.Мова XML дозволяє кожному створювати свій власний формат документів і потім писати документи в

цьому форматі. Ці формати документів можуть включати розмітку, яка уточнює зміст контенту документа. Документ з розміткою може "читатися" комп'ютером.

XML і RDF - сучасні Internet-стандарти, які служать для забезпечення семантичної інтероперабельності в Web. При цьому XML піднімає питання, пов'язані тільки зі структурою документів. RDF більше пристосований для забезпечення семантичної інтероперабельності, оскільки пропонує модель даних, яку можна розширити таким чином, щоб вона охоплювала більш досконалі методики подання онтології.

1.3 Загальна схема опису ресурсів RDF

Для опису предметної області ресурсів запропонований стандарт RDF (Resource Description Framework) [10], прийнятий у 1999 році консорціумом W3C і підтриманий багатьма провідними виробниками ПЗ, і постачальниками контенту. Початкове призначення RDF було в описі XML-ресурсів з різних точок зору. RDF представляє собою модель опису метаданих. Ця мова використовує XML-синтаксис.

У той час, як модель даних XML є графом з позначеними вершинами і не позначеними дугами (тобто без зв'язків), модель даних RDF є графом з позначеними, як вершинами, так і дугами, що дозволяє визначати зв'язки між сутностями.

Модель Resource Description Framework має мету: стандартизувати визначення та використання метаданих, які описують ресурси Web. Однак, RDF також добре підходить і для представлення даних [3].

Стандарт RDF (Resource Description Framework) включає дві основні частини - власне спосіб опису ресурсів, а також спосіб завдання схем, за якими ресурс описується.

Перша частина RDF [11] визначає просту модель для опису об'єкта, який розглядається, як ресурс, як зв'язок між ресурсами в термінах, найменованих властивостей і значень.

Друга (RDF Schema - RDFS) служить для завдання структури предметної області та аналогічно - діаграмі класів в UML.

На RDF можна описувати, як структуру ресурсу, так і пов'язану з ним предметну область. RDF описує ресурси у вигляді орієнтованого розміченого графа - кожен ресурс може мати властивості, які в свою чергу, також можуть бути ресурсами або їх колекціями.

Базовий будівельний блок у RDF - це трійка об’єктів «об'єкт - атрибут - значення», який часто записують у вигляді A (O, V), тобто «Об'єкт O має атрибут A зі значенням V».

Такий зв'язок можна також представити, як ребро з міткою A, яке об'єднує два вузли, O і V: [O] - A -> [V]. Така нотація досить корисна, оскільки RDF дозволяє міняти місцями об'єкти та значення. Таким чином, кожен об'єкт може грати роль значення, яке в графічному представленні відповідає ланцюжку з двох ребер з мітками.

Крім усього вищезгаданого, RDF допускає форму подання, в якій будь-який вираз RDF в трійці може бути об'єктом або значенням, тобто графи можуть бути, як вкладеними, так і лінійними. В Web це дозволено, наприклад, висловлювати сумнів або згоду з виразами, створеними іншими людьми.

Головна мета RDF - запропонувати базову модель даних «об'єкт - атрибут - значення» для метаданих. Окрім цієї семантики, що описана в стандарті лише неформально, RDF не містить будь-яких чітких правил, орієнтованих на моделювання даних. Також, як XML Schema використовується для визначення словника, RDF Schema дозволяє розробникамвизначати конкретний словник для даних RDF (такий, як authorOf) і вказувати види об'єктів, до яких можуть застосовуватися ці атрибути. Іншими словами, механізм RDF Schema надає базову систему типів для моделей RDF.

Таким чином, RDF надає можливість формулювати твердження у вигляді, придатному для обробки комп'ютером і це є основою Semantic Web.

Метадані – це дані,призначені для ідентифікації, опису або локалізації

інформаційних ресурсів, не залежно від фізичної природи ресурсу. А RDF – одна із стандартизованих форм представлення цих метаданих.

1.4 Метадані

У базовій моделі Semantic Web, представленої вище, запропонованої Тімом Бернерс-Лі, явно не виділено наявність засобів опису метаданихТим не менш, у своїх роботах, наприклад, [3], а також у роботах інших вчених вказується на важливість включення в концепцію Semantic Web поняття метаданих.

Метадані це дані про дані. Більш точно, це дані, призначені для ідентифікації, опису або локалізації (місця розташування) інформаційних ресурсів, не залежно від фізичної природи ресурсу.

Було розроблено безліч схем опису метаданих, серед яких слід згадати наступні:

Topic Maps (XMT) [7] - стандарт ISO (ISO / IEC 13250:2003) для представлення та обміну знаннями з точки зору пошуку інформації.

Text Encoding Initiative (TEI) [8] - міжнародний проект з розробки нормативів для розмітки (marking up) електронних текстів, таких як романи, пьєси, вірші; головним чином для підтримки досліджень у гуманітарній сфері.

Metadata Encoding and Transmission Standard (METS) [9] - стандарт кодування і передачі метаданих, був розроблений для задоволення потреби у стандартній структурі даних для опису складних цифрових бібліотечних об'єктів.

Metadata Object Description Schema (MODS) [5] - схема метаданих опису об'єктів, яка була виведена з MARC 21, і призначена для перенесення відібраних даних з існуючих записів метаданих MARC 21 або для створення оригінальної запису опису ресурсу.

Encoded Archival Description (EAD) [5] - закодований архівний опис, був розроблений, як спосіб розмітки даних, які містяться в пошукових коштах, для того, щоб вони знаходилися й показувалися в оперативному режимі.

Learning Object Metadata (LOM) [5] - стандарт IEEE 1484.12.1-2002 метаданих об'єктів навчального процесу для повторного використання ресурсів навчального характеру, таких, як: комп'ютерне та дистанційе навчання.

Online Information Exchange (ONIX) [5] - міжнародний стандарт схеми метаданих, який розроблений видавцями книжкової промисловості Сполучених Штатів і Європи.

Однак, базовими для Semantic Web в даний момент визнаються стандарти Dublin Core, FOAF, SIOC і DOAP [4].

FOAF (Friand-Of-A-Friend) [5 – 7] – це формат машинно-оброблюваних сторінок, що описують персональну інформацію про людей і їх діяльності (фотографії, календарі та інше) у форматі XML.

SIOC (Semantically-Interlinked Online Communities) [8] – документи, що описують онлайн-спільноти. SIOC забезпечує взаємозв'язок таких засобів обговорення інформації, як блоги, форуми і поштові розсилки, між собою.Description of a Project Description of a Project (DOAP) [11] - документи, що описують в мережі проекти з відкритим вихідним кодом.

Серед цих стандартів виділяється Dublin Core [6], як один з базових стандартів для представлення даних про інформаційні ресурси в Semantic Web. Dublin Core [6] - набір елементів (властивостей) для опису документів, який був розроблений в березні 1995 року. Мета Dublin Core - забезпечення мінімального набору елементів опису, які сприяють впровадженню опису та автоматичної індексації документоподібних мережевих об'єктів за принципом, подібного карткам бібліотечного каталогу.

Як наголошується в офіційному описі RDF, метадані можуть бути вбудованими (embedded) в сам ресурс, наприклад, в HTML сторінки або документи, наприклад, MsWord (це найпростіший підхід для опису сторінок), а можуть зберігатися і оновлюватися незалежно від ресурсів. Багато хто з виробників програмного забезпечення вже випускають ряд продуктів, які автоматично формують деякий невеликий блок RDF- опису, всередині

документа. Другий підхід є більш універсальним, так, як в цьому випадку метадані можуть бути створені для будь-якого ресурсу. В даний час вже розпочато проект на основі Open Directory (пошукова система Google) з автоматичним створенням репозиторії RDF-описів ресурсів Інтернет.

У разі розміщення метаданих окремо від ресурсу, самі метадані переважно зберігаються (і передаються) у форматі XML. При цьому максимально використовуються можливості моделі RDF та забезпечується вільний обмін інформацією (interoperability). Обмін метаданими зводиться до пересилання RDF / XML-файлів (тобто текстових файлів у форматі XML або просто посилань на ці файли), тобто може бути повністю автоматизований.

RDF Schema слугує для метаданих тим, що вона може представити конкретні дані(метадані) в RDF форматі, уже згідно з RDF Schema .

1.5 RDF Schema

Першим "пластом" Semantic Web над тільки, що обговорених синтаксисом, є проста модель типізації даних. Схема і онтологія - це кошти для опису змісту і зв'язку між термами.

На основі RDF 23 січня 2003 був запропонований робочий проект RDF Vocabulary Description Language 1.0: RDF Schema. Схема RDF була розроблена, як проста модель типізації даних для RDF. Як вказується в документі, RDF є мовою загального застосування для подання інформації в Інтернет. Дана специфікація описує як використовувати RDF для опису RDF-словників. Вона визначає базовий словник, призначений для цих цілей і прийняті угоди, які можуть бути використані при створенні додатків Semantic Web для підтримки більш складних словників RDF-описів. Мова опису словника RDF визначає класи і властивості, які можуть бути використані для опису інших класів і властивостей, а також робити деякі більш складні речі, такі, як створення діапазонів і областей для властивостей.

Три найбільш важливих поняття, які дає нам RDF і схема RDF - це "Ресурс" (rdfs: Resource), "Клас" (rdfs: Class) і "Властивість" (rdfs: Property). Ці

поняття є "класами" в тому розумінні, що цим класам можуть належати терміни.

Як вже було зазначено, RDF Schema визначається в термінах базової інформаційної моделі RDF - структури графа, який описує ресурси і властивості. Всі словники RDF використовують деяку базову структуру: вони описують класи ресурсів і типи зв'язків між ресурсами. Ця спільність дозволяє різнорідні словники, створені для машинної обробки, і відповідає вимогам, щодо створення метаданих, в яких твердження можуть бути отримані з безлічі різнорідних децентралізованих словників, створених різними спільнотами за різними принципами і різними методами.

Опис за допомогою RDF не обмежується тільки описом документів Інтернет. Цей стандарт досить універсальний і гнучкий для того, щоб описувати більшість типів структурованих даних. Наприклад, в RDF природно виражаються діаграмами сутній зв'язки, які широко застосовувані для проектування баз даних. Опис семантики ресурсу на RDF може бути як «зовнішнім», коли описується ресурс в цілому, так і «внутрішнім», коли описується внутрішня структура ресурсу - будь-то база даних, XML-документ, або цілий сайт.

Важливою особливістю стандарту RDF, який лежить в основі XML, є розширюваність. На RDF можна задати структуру опису джерела, використовуючи і розширюючи вбудовані поняття RDF-схем, такі як класи, властивості, типи, колекції. Модель схеми RDF включає спадкування; успадковуватися можуть як класи, так і властивості.

Крім опису структури, RDF дозволяє оперувати твердженнями. Вираз «ресурс R1, як властивість P має ресурс R2» можна проінтерпретувати і як предикат P (R1, R2), а потім використовувати це твердження як об'єкт інших тверджень. Така інтерпретація дозволяє описувати, з допомогою RDF, концептуальну інформацію.

Таким чином, RDF цілком підходить на роль універсальної мови опису

семантики ресурсів і взаємозв'язків між ними.

Однак, як стверджують самі автори стандарту, RDF має й ряд відсутніх властивостей, які вказують як наступні:

- неможливість вказати того, що подана властивість (наприклад, hasAncestor - має предка, прототип) є транзитивна, наприклад, що «якщо A hasAncestor B, і B hasAncestor C, тоді A hasAncestor C»;
- неможливість вказівки того, що два різних класи, визначені у різних схемах, фактично представляють одне і те ж поняття;
- неможливість вказівки того, що два різних примірника (instances), визначені окремо, фактично представляють один і той самий суб'єкт;
- неможливість визначення нових класів у термінах операцій (наприклад, об'єднання і перетин) над іншими класами.

Найбільш розвиненою мовою представлення онтологій в даний час є OWL (Web Ontology Language), яка розширює можливості XML, RDF, і RDF Schema. Онтології грунтуються на математичному апараті формальної логіки (descriptive logic, DL)- мала підмножина, якого охоплена RDF-схемою

1.6 Онтології

Онтології, в загальному вигляді, визначаються, як спільно використовувані формальні концепції конкретних предметних областей, вони дають загальне уявлення про поняття, інформацією, з яких, можуть обмінюватися люди та програми. Вони дозволяють скласти в концепцію домен фіксуванням сутностей і зв'язків у домені. Вказівка, в яких зв'язках бере участь сутність, частково дозволяє зрозуміти і її значення, оскільки це надає можливість бачити, де дана сутність входить у відносини з іншим доменом.

Онтології грунтуються на математичному апараті формальної логіки (descriptive logic, DL), мале підмножина, якого охоплена RDF-схемою. DL є підмножиною логіки першого порядку, яке обчислюваних.

Додаткові можливості, вище зазначені, в додатку до наявних в RDF, є

метою онтологічних мов, таких, як DAML + OIL [73] і OWL [75]. Дані дві мови засновані на RDF і RDF Schema. Мета даних мов - забезпечення ресурсів додаткової машинно-оброблюваної семантикою, тобто вони спрямовані на забезпечення машинного подання ресурсів у формі, який більш відповідає їх оригіналу з реального світу.

Розмітка документів Semantic Web, за допомогою онтологічних термінів, дозволить виробляти автоматичну обробку їх контенту. Таким чином, онтології визначаються, як ключова технологія для розвитку Semantic Web.

Онтології в змозі зіграти критично важливу роль в організації обробки знань на базі Web, їх загального використання та їх обміну між додатками.

Мова OWL. Найбільш розвиненою мовою представлення онтологій в даний час є OWL (Web Ontology Language), яка розширює можливості XML, RDF, і RDF Schema. Ця мова заснована на DAML + OIL. Проблеми, які виникли в DAML + OIL, були викликані постійною зміною ядра специфікацій RDF, на якому заснований DAML + OIL.

Як вказується в основному робочому проекті, OWL майже повністю схожий на DAML + OIL. Основні й істотні відмінності від DAML + OIL полягають у наступному:

- усунення деяких обмежень;
- здатність прямо вказувати, що властивість може бути симетричною;
- відміна деяких невикористовуваних конструкцій DAML + OIL, особливо обмеження з додатковими компонентами.

Існує також кілька незначних розбіжностей, які включають в себе деякі зміни імен деяких конструкцій, однак основна мета, яка ставилася при створенні OWL, полягала в тому, щоб максимально коректно зберегти імена DAML + OIL.

Онтологія OWL є послідовністю аксіом і фактів з додаванням посилань на інші онтології, які вважаються включеними в онтологію. Онтології OWL є Web-документами і на них можна посилатися. Онтології також мають не пов'язану з

логікою компоненту (поки ще не визначену), що може бути використана для запису авторства, і інша не пов'язана з логікою інформація, асоційована з онтологією. Фактично це словник, який розширює набір термінів, визначених у RDFS.

Онтології включають інформацію про класи, властивості і окремі випадки, кожен з яких може мати ідентифікатор ID, що є посиланням URI.

OWL має три модифікації:

- OWL Lite (простий);
- OWL DL (з повним доступом);
- OWL Full (з повною виразною потужністю).

Кожна з цих модифікацій (крім Lite) є розширенням попередньої. Як наслідок: будь-яка OWL Lite онтологія є OWL DL онтологією, а будь-яка OWL DL онтологія є OWL Full онтологією.

Головні характеристики мови веб-онтологій - OWL:

- OWL використовує синтаксис XML;
- OWL має інструкції для представлення дерева класів;
- OWL має інструкції для вказівки приналежності індивідів до класів;
- OWL має систему опису властивостей: область визначення, область значень;
- OWL може задавати характеристики властивостей: симетричність, транзитивність, функціональність;
- OWL має інструкції для вказівки еквівалентності (склеювання) класів. Використання готової онтології дозволить розробникам, безпосередньо, приступити до заповнення даних та побудови шаблонів і дизайну.

У разі відкритої публікації RDF-даних можлива реалізація програмних агентів для пошуку цих даних (наприклад, за допомогою спеціальних запитів системи Google), агрегація в єдиному сховищі та надання даних користувачеві (наприклад, абітурієнту) в єдиному інтерфейсі зі специфічними функціями.

Можуть бути просто інтегровані дані підрозділів і представництв вузу, які просто редагуються редактором онтологій на місці, та імпортуються з основного веб-сайту цього вузу. У разі інтеграції досить великих і часто мінливих розподілених даних (наприклад, для агрегації інформації про конференції регіону з веб-представництв вузів і наукових організацій), можливе використання RDF-сховищ з відкритими інтерфейсами для вибірки тільки необхідних даних.

1.7 Мови запитів до RDF сховищ

Говорячи про мови запитів, фактично мова йде про інтеграції різних мов (інформаційно- пошукових, баз даних, маніпулювання даними, обміну даними і т.п.) в єдину мову запитів Web. При цьому всі фахівці об’єднуються в думці, що це має бути декларативна мова, побудована на моделі не повністю структурованих даних (semistructured).

Документ "XML-QL: A Query Language for XML" [76] був підготовлений до семінару W3C по пошуковим мовам, який пройшов в кінці 1998 року і виявився далеко не єдиною спробою узагальнення такого роду.

В даний момент з'явилося декілька мов запитів до XML-джерел даних: XQL (1998) [7], XML QL (1998) [8]. Пошук в XML-документі полягає в знаходженні елементів, які задовольняють умови запиту, з подальшим перетворенням знайдених елементів у структуру, задану у запиті.

Мова запитів до RDF-джерел даних (RDF Query), запропонована в 1998 [12] і в даний час має вже практичну реалізацію в проекті Sesame [6]. У 2006 році консорціум W3C почав розробку мови запитів до RDF та OWL-сховищ SPARQL Query Language for RDF, який зараз має статус рекомендованого кандидата (candidate recommendation) [8].

SPARQL - мова запитів, яка базується на патерну графів.

SPARQL одночасно є, як мовою запитів, так і протоколом доступу до даних, також SPARQL є одною з ключових компонент додатків Web 2.0: в якості стандарту, для підтримки гнучкої моделі даних, він дає загальний механізм

запитів для всіх додатків Web 2.

1.8 Принцип "логічного висновку"

Принцип "логічного висновку" дуже простий: це можливість виводити нові дані з даних, які вже є. В математичному сенсі, виконання запиту є однією з форм логічного висновку (наприклад, можливість вивести з маси даних, деякий результат пошуку). Логічний висновок є одним з провідних принципів Semantic Web, так як він дозволяє дуже легко створювати SW-програми [8].

Для того, щоб Semantic Web став досить виразним і зміг допомагати людям у різних ситуаціях, виникає необхідність побудови потужної логічної мови, яка підтримує логічний висновок. Дискусії, щодо методів, і навіть можливостей виконання цього завдання, до цих пір ведуться дуже активно; звертається увага на те, що в RDF недостатні можливості квантифікації, і що ця область визначена недостатньо добре. Проблеми логіки предикатів докладно розглянуті в базовій монографії Джона Сова (John Sowa's)

Rule Interchange Format (RIF) - формат обміну правилами. Мета якого розробляється консорціумом W3C стандарту - визначення формату, який би дозволив транслювати правила між різними мовами і завдяки цьому забезпечити обмін правилами між системами, заснованими на правилах.

Системи, які грунтуються на правилах, одержали широке поширення в інформаційних технологіях. До їх числа відносяться, наприклад, експертні системи і системи дедуктивних баз даних. Розробки технологій Semantic Web забезпечують нове середовище використання таких систем. Тому консорціум

W3C приділяє окрему увагу цій галузі.

Специфікація RIF може розглядатися, як складова частина комплексу стандартів Semantic Web.

В даний час робочою групою, організованої за консорціумі для розробки цього стандарту, підготовлений, та обговорюється, робочий проект документа, який систематизує випадки використання RIF та вимоги до цієї мови. Найважливіша вимога до створюваного стандарту - забезпечення можливості

його використання не тільки при поточному стані технологій, заснованих на правилах, але і його гнучкості, достатньої для забезпечення його використання в процесі їх еволюції.

Робочий проект документа, який описує випадки використання, дасть можливість визначити функціональні вимоги до RIF і на цій основі розробити адекватні специфікації мови.

Правила виведення нових фактів SWRL. Завдяки доповненню OWL мовою RuleML (підмножина Datalog) у вигляді словника SWRL (A Semantic Web Rule Language) з'явилася можливість використовувати діз'юнкти Хорна (Horn-like rules) для явної вказівки способу виведення нових фактів з RDF-тверджень. Поки словник SWRL знаходиться в стадії стандартизації.

Хоча роботи над цим рівнем Semantic Web тривають, проте в нашому розпорядженні є вже достатній набір засобів для побудови Semantic Web: твердження, цитування (матеріалізація) у RDF, класи, властивості, області, документування у схемі RDF, непересічні класи, властивості однозначності та унікальності , типи даних, інверсії, еквівалентності, списки та інше

1.9 Висновки по розділу

У цьому розділі були описані та проаналізовані основні стандарти та засоби, що закладені в основу технології створення семантичних додатків.

2. ТЕХНОЛОГІЇ BUSINESS INTELLIGENCE

2.1 Вступ

Бізнес-аналітика, або BI(Business Intelligence) — це загальний термін, що має на увазі різноманітні програмні продукти і програми, створені для аналізу первинних даних організації. Вперше термін Business Intelligence запропонував американський вчений, співробітник IBM H.P. Luhn Ханс Петер Лун (1896-1964) у 1958 році[13] . Він був фахівцем в області information science, тобто займався інформатикою в докомпьютерную епоху, йому були доступні тільки електромеханічні табулятори.

BI – орієнтований на користувача процес, що забезпечує доступ і дослідження інформації, її аналіз, формування інтуїтивного розуміння, що ведуть до поліпшеного і неформального прийняття рішень. Нині BI – це інструменти для аналізу даних, побудови звітів і запитів, що можуть допомогти бізнес-користувачам обробляти великі обсяги і синтезувати з них значиму інформацію [12].

В основі технології BI лежить організація доступу кінцевих користувачів, аналіз структурованих кількісних даних і інформації про бізнес. BI забезпечує процес збору багатоаспектної інформації про досліджуваний предмет [3].

На жаль, адекватного перекладу словосполучення “Business Intelligence” українською та російською мовами немає, а ті терміни, що використовуються, тільки більше заплутують читача або є не зовсім коректною калькою з англійської («бізнесінтелект», «бізнес-аналіз»). Можливо, варто звернутися до перекладу слова Intelligence у словосполученні Artificial Intelligence, де під Intelligence розуміється зовсім не «інтелект» і тим більше не «інтелігентність», а просто здатність до логічного виведення. Таким чином, словосполучення “Business Intelligence” можна перевести як “засоби логічного виведення, орієнтовані на бізнес-додатки”.

Бізнес-аналіз як діяльність складається з кількох пов'язаних між собою

процесів:

- інтелектуальний аналіз даних (data mining),
- аналітичну обробку в реальному часі (online analytical processing),
- отримання інформації з баз даних (querying),
- складання звітів reporting).

BI – це не тільки технологічна платформа або набір програмних засобів для збору, збереження, аналізу і забезпечення доступу до даних складної структури, що допомагає працівникам підприємства приймати кращі бізнес рішення. Це й здатність підприємства ефективно використовувати свої людські й інформаційні ресурси. Основний напрям сучасних BIпроектів – це доставка інформації, але варто також приділити увагу аналізу нових відомостей і проблемі інтеграції з іншими сховищами даних. Визначаючи платформу BI, можна виділити наступні три групи базових можливостей BI:

- інтеграція: інфраструктура BI, керування метаданими, розвиток, потоки робіт і колективна робота;
- доставка інформації: звітність, інформаційні панелі, незаплановані запити, інтеграція з Microsoft Office;
- аналіз: OLAP, розширена візуалізація, прогнозуюче моделювання, протоколи результатів.

Компанії використовують BI для прийняття обґрунтованих рішень, скорочення витрат і пошуку нових перспектив для бізнесу. BI — це щось більше, ніж звичайна корпоративна звітність або якийсь набір інструментів для отримання інформації з облікових систем підприємства. IT-директора використовують бізнес аналітику, щоб виявити неефективні бізнес-процеси, які «дозріли» для перебудови.

Використовуючи сучасні засоби бізнес-аналізу, підприємці можуть почати аналізувати дані самостійно і не чекати, поки IT-відділ сформує складні і заплутані звіти. Така демократизація доступу до інформації дає користувачам

можливість підкріпити реальними даними свої бізнес-рішення, які у протилежному випадку були б засновані на припущиннях і випадковості.

Незважаючи на те, що системи BI досить перспективні, їх впровадження може бути ускладнене технічними проблемами. Менеджерам необхідно забезпечувати чіткі та узгоджені дані для BI додатків, щоб користувачі могли їм довіряти.

2.2 З чого має розпочатися впровадження BI?

Загальна залученість співробітників життєво необхідна для успіху BI-проектів, оскільки кожен, хто задіяний у процесі, повинен володіти повним доступом до інформації, щоб мати можливість змінити способи і методи своєї роботи. BI-проекти повинні починатися з вищого керівництва, а наступною групою користувачів повинні бути менеджери з продажу. Їх основний обов'язок — нарощувати продажі, і заробітна плата часто залежить від того, наскільки добре вони це роблять. Тому вони набагато швидше сприймуть будь-який інструмент, здатний допомогти їм у роботі, за умови, що цей інструмент легко використовувати і що вони довіряють одержуваної з його допомогою інформації.

Використовуючи BI-системи, співробітники коригують роботу над індивідуальними і груповими завданнями, що веде до більш ефективної роботи команд продавців. Коли керівники відділів продажів бачать суттєву різницю показників декількох відділів, вони намагаються довести «відстаючі» відділи до того рівня, на якому працюють «лідируючі».

BI забезпечує єдність метаданих для всього інструментарію та уніфіковані зручні способи пошуку, здобуття, збереження, повторного використання і публікації об’єктів метаданих. OLAP дозволяє кінцевим користувачам аналізувати дані за допомогою швидкого виконання запитів і розрахунків, а прогнозуюче моделювання і Data Mining дозволяє класифікувати та категоризувати інфор-мацію, видобуваючи закономірності.

Багато організацій використовує відразу кілька BI-продуктів, однак хотіли

б упровадити єдиний загалькорпоративний BI-стандарт, що забезпечує широку функціональність й підтримує одночасно велику кількість користувачів.

Класифікація продуктів BI

Програмне забезпечення BI підрозділяють на BI-інструменти та BI-додатки. BI-інструменти застосовуються кінцевими користувачами для доступу, аналізу і генерації звітів за даними, що розташовуються в сховищі, вітринах чи оперативних складах даних [4, 5]. Серед них виділяють:

- генератори запитів і звітів;
- інструменти OLAP;
- корпоративні BI-набори EBIS;
- BI-платформи.

Засоби генерації запитів і звітів у великому ступені поглинаються і заміщуються корпоративними BI-наборами. Багатомірні OLAP-механізми та реляційні OLAP-механізми є BI-інструментами й інфраструктурою для BI-платформ.

2.3 Впровадження BI-системи

Перед впровадженням BI-системи, компаніям слід проаналізувати механізми прийняття управлінських рішень та зрозуміти, яка інформація необхідна керівникам для більш обґрунтованого й оперативного прийняття цих рішень. Також бажано проаналізувати, в якому вигляді керівники воліють отримувати інформацію (у якості звітів, графіків, онлайн у паперовій формі). Уточнення даних процесів покаже, яку інформацію компанії необхідно отримати, аналізувати і консолідувати в своїх BI-системах.

Якісні BI-системи повинні надавати користувачам контекст. Недостатньо просто складати звіти про те, якими були продажу вчора і якими — рік тому в цей же день. Система повинна давати можливість зрозуміти, які чинники привели саме до такого значення обсягу продажів в один день і іншому — в той же день рік тому.

Подібно багатьом IT проектів, впровадження BI не окупиться, якщо користувачі будуть відчувати «загрозу» або скептично ставитися до цієї технології і в результаті відмовляться від її використання. BI, будучи впровадженою в «стратегічних» цілей, повинна, по ідеї, фундаментальним чином змінити функціонування компанії і процес прийняття рішень, тому керівникам IT-департаментів необхідно з особливою увагою підходити до думок і реакцій користувачів.

Деякі переваги від використання BI-рішень

Велика кількість BI-додатків допомогло компаніям з лишком відбити вкладені кошти. Системи бізнес-аналітики використовуються для вивчення способів скорочення витрат, виявлення нових можливостей для розвитку бізнесу, представлення ERP-даних у наочній формі, а також для швидкого реагування на зміну попиту і оптимізації цін.

Крім підвищення доступності даних, BI може надати компаніям більше переваг під час переговорів, спрощуючи оцінку відносин з постачальниками і клієнтами.

Компанії, що використовують BI-системи - Ресторанні мережі (наприклад, hardee's, wendy's, Ruby Tuesday і T. G. I. friday's) активно використовують системи бізнес-аналітики. BI вкрай корисний їм для прийняття стратегічно важливих рішень. Які нові продукти додати в меню, які страви виключити, які неефективно працюють точки закрити і т. д. Вони також використовують BI для таких тактичних питань, як перегляд договорів з постачальниками продуктів та виявлення шляхів удосконалення неефективних процесів. Оскільки ресторанні мережі сильно орієнтовані на свої внутрішні бізнес-процеси і оскільки BI займає в контролі цих процесів центральне місце, допомагаючи управляти підприємствами, ресторани, серед усіх галузей, які входять в елітну групу компаній, які отримують реальну вигоду від цих систем.

Бізнес-аналітика є одним з ключових компонентів BI. Цей компонент важливий для досягнення успіху компанії з будь-якої галузі.

У секторі роздрібної торгівлі Wal-Mart широко застосовує аналіз даних і кластерний аналіз для того, щоб зберігати своє домінуюче положення в секторі. Harrah's змінив основи своєї політики конкурентної боротьби в гральному бізнесі, зробивши акцент на аналіз лояльності клієнтів та рівня обслуговування, замість підтримки мега-казино. Amazon і Yahoo — це не просто великі веб-проекти, вони активно використовують бізнес-аналітику і загальний підхід «протестуй і зрозумій» для налагодження своїх бізнес-процесів. Capital One проводить більше 30 000 експериментів щорічно для виявлення цільової аудиторії та оцінки пропозицій за кредитними картками.

2.4 Історія розвитку технології

Business Intelligence 1.0

Термін бізнес-аналіз став широко використовуватись в кінці 1990-х і початку 2000-х років. У цей період були дві основні функції BI: отримання даних і звітів, організація та візуалізація даних. Проте, існувало дві суттєві проблеми які стримували розвиток технології: складність і час.

Користувачами Business Intelligence 1.0 були IT-працівники та вчені-аналітики найвищих ієрархічних рівнів, більша частина спеціалістів не знала про ці засоби. Тільки експертні технічні фахівці змогли використовувати програмне забезпечення для аналізу даних. Інструменти почали розвиватися, щоб задовільняти недосвідчених користувачів, але це відбувалось повільними темпами.

Business Intelligence 2.0

На початку 21-го століття відзначений особливий поворотний момент, оскільки технології вирішили основні проблеми: складність і швидкість. Нове покоління передбачало:

- значне розширення кола користувачів аналітиків за рахунок доступності базових інтерфейсів користувачів і функцій засобів BI 2.0 для працівників всіх корпоративних ієрархічних рівнів;

- високу швидкодію на великих обсягах даних і наявність функцій добування й активації не тривіальних знань у режимі реального часу;
- покращання візуалізації результатів аналізу;
- можливість побудови користувачами простих програмних тригерів, які сповіщають про вихід параметрів за визначені межі;
- інтеграцію з елементами мережевих KMS з можливістю ділитися знаннями з колегами у корпоративній мережі;
- використання елементів Text Mining;
- спрощення навчання користувачів за допомогою навчальних відеороликів;

Технології для користувачів, які не є експертами, а це означає, що співробітники тепер можуть виконувати проекти без втручання з боку IT-відділу.

Експоненціальне зростання Інтернету підтримав і висунув ці зміни, частково за рахунок генезису соціальних мереж. Facebook, Twitter, блоги і дав користувачам дуже прості і дуже швидкі способи обміну ідеями та думками.

Він також надав користувачам можливість переглянути методи і програмне забезпечення, а також в більш широкому плані поширення базового розуміння різних способів використання бізнес-аналітики. Чим більше люди спілкувалися, тим більше, що вони зрозуміли.

До 2005 року зростаюча взаємопов'язаність світу бізнесу означає, що компанії необхідну інформацію в режимі реального часу, за цілою низкою причин. В основному вони необхідні, щоб не відставати від конкурентів, і зрозуміти, що їх споживачі хотіли і що вони думають про свою компанію.

Business Intelligence 3.0

Після 2010 року почали визначити Business Intelligence 3.0. За оцінкою DSS-BI.com.ua, не дивлячись на наявність інших оцінок, парадигму Business Intelligence 3.0 доцільно зв'язувати, насамперед, з парадигмою Cloud Computing, яка визначає апаратно-програмне забезпечення як сервіс (Software as a service,

SaaS) у віртуальних кіберхмарах. Business Intelligence / Cloud computing (Business Intelligence 3.0) суттєво відрізняється від Business Intelligence 2.0. Адже парадигма Cloud Computing та (або) SaaS принципово відрізняється від попередніх парадигм за своєю філософською сутністю, принципом плати за використання, апаратно-програмною реалізацією і більш високою ефективністю функцій Business Intelligence/Cloud computing. Розробники Cloud Computing BI (Business Intelligence 3.0) можуть забезпечити значно більш високу функціональність користувачам залежно від плати, безперервне централізоване удосконалення, ін. Тому перехід до Cloud computing значної кількості аналітичних функцій багатьох організацій й окремих користувачів є невідворотною перспективою.

2.5 Порівняння застосунків Business Intelligence

Розглянемо програмні реалізації BI, щоб зрозуміти, чому саме ця сфера є цікавим та перспективним сектором для впровадження новітніх інтелектуальних засобів. У додатки Business Intelligence часто вбудовані BI-інструменти.

Відповідно до досліджень провідних ринкових аналітиків, таких як Forrester і Gartner, розробники програмного забезпечення BI можуть бути впорядковані за допомогою “магічного квадрата” : лідери – постачальники з широким функціоналом можливостей платформ BI (Cognos, Business Objects, Microsoft, Oracle, MicroStrategy, SAS); претенденти на лідерство (SAP, Information Builders) пропонують платформи із широкою функціональністю, але в них є обмеження в частині програмного чи оточення предметних областей або каналів збуту; провидці (QlikTech, Tibko Spotfire) мають гарне представлення про платформу BI, відрізняються відкритістю і гнучкістю архітектури і пропонують добре розвитого функціонала у своїх цільових областях, однак, діапазон пропонованих функцій не досить широкий; нишеві гравці (Actuate, arcplan, Board International, Panorama Software) – компанії, що досягли успіху тільки у конкретній області.

Розглянемо більш детально характеристики компаній-виробників BI-додатків і їхньої продукції. SAP Business Objects [6] пропонує широкий набір апробованих закінчених рішень BI, включаючи продукти з інтеграції та якості даних і аналіз тексту. Всі продукти SAP Business Objects переводяться на загальну платформу для підвищення безпеки і поліпшення адміністрування, інтеграція різних продуктів, забезпечення більш зручного обміну метаданими і переходу від одного продукту до іншого. IBM Cognos [7] – комплекс інтегрованих програмних продуктів для керування ефективністю діяльності підприємства й управлінського обліку (Corporate Performance Management, CPM). Це одне з найсучасніших і масштабованих середовищ BI. Cognos базується на єдиній платформі J2EE і надає єдиний інтерфейс для більшості продуктів. SAS – постачальник повного комплексу рішень BI та аналітичних інструментів [8]. SAS забезпечує інтеграцію даних, аналіз тексту, інструментарій для статистичного аналізу і проактивне моделювання, а також маркетинговий аналіз і керування ризиками. Oracle реалізувала свої ідеї щодо BI у Enterprise Edition Plus [9], що містить сервер аналітики з механізмом ROLAP (Relational OLAP) і технологією інтеграції корпоративних даних (EII). MicroStrategy – платформа, яка охоплює весь комплекс можливостей BI, з власним механізмом ROLAP, що ефективно оптимізує різні моделі даних. Долі цих платформ у загальній кількості зображено на рисунку 2.1.

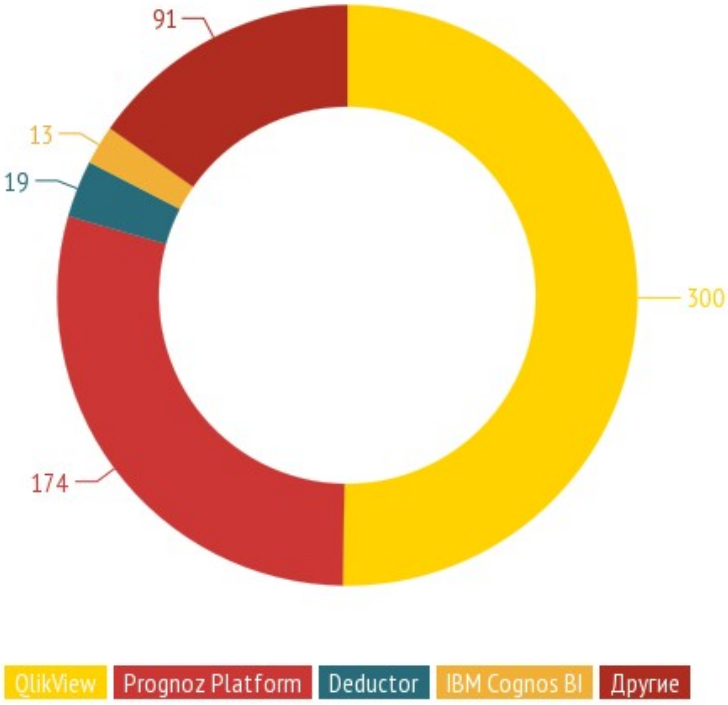

Рисунок 2.1 Долі BI-платформ в загальній кількості проектів

Більш наглядне порівняння можливостей сучасних BI-платформ наведено в таблиці 2.1.

Таблиця 2.1 Порівняльна таблиця засобів бізнес-аналітики

Назва платформи	Засоби розробки	Оповіщення	Колективна робота	Звіти	Dashboard (індикатори)	Власна мова запитів	Веб та мобільні додатки	OLAP	Візуальний аналіз	Аналіз через пошук	Прогноз та DataMining
Bime	V	+/-	+/-	V	V	V	V	V	V	X	V
Birst	V	V	V	V	V	V	V	V	V	X	V
IBM Cognos	V	V	V	V	V	V	V	V	V	V	V
IBM SPSS Decision	V	+/-	V	X	X	X	X	X	X	X	V
Jaspersoft	V	+/-	V	V	V	V	V	V	V	X	X
Microstrategy Express	V	+/-	V	V	V	V	V	V	V	X	X
QlikView	V	V	V	V	V	V	V	V	V	V	+/-
SAP Lumira	X	X	+/-	X	X	X	V	X	V	X	X
SAS	X	V	V	V	V	V	V	V	V	X	V
Tableau	+/-	V	V	V	V	V	V	V	V	X	+/-
Tibco Personal	X	X	X	V	V	V	V	V	V	X	V

2.6 Висновки до розділу

В цьому розділі оглянуто загальні відомості про BI системах і принципах їх роботи. Було класифіковано BI-інструменти на генератори запитів і звітів, інструменти OLAP, корпоративні BI-набори EBIS, BI-платформи.

Розглянуто процес та проблеми які можуть виникнути в разі вирішення компанії почати використовувати BI-інструменти. Надані поради щодо вибору та інтеграції цих технологій якомога швидше.

Проаналізована історія розвитку BI-систем. Оглянуто стан речей на сьогодні та можливий розвиток цього напрямку. Порівняно можливості сучасних систем та проаналізовано технології, що використовують сучасні гіганти цієї індустрії.

3. ПРОЕКТУВАННЯ BI-СИСТЕМИ

3.1 Вступ

Сьогоднішні засоби бізнес-аналітики (BI, OLAP) в основному сконцентровані на кількісному, статистичному аналізі бізнес-даних. Більшість подібних систем використовують жорстко задані в структурі бази даних набори параметрів і показників аналізу (для їх конфігурації потрібні дорогі фахівці). За допомогою таких засобів складно описувати і аналізувати причинно-наслідкові зв'язки, важко формалізуються і різноманітні взаємовідносини об'єктів (наприклад, ступінь впливу тих чи інших факторів на результати певних процесів). Семантична аналітика, побудована на принципі "відкритого світу", надає нові можливості вирішення подібних завдань. А в поєднанні з принципами нечіткої логіки, побудови симулятивних ("живих") моделей, її можливості стають практично безмежними.

Найпростішим засобом ілюстрації принципових відмінностей семантичної аналітики від інших засобів аналізу є інструмент Facebook Graph Search. Цей пошуковий інтерфейс, вбудований в соціальну мережу, може відповісти на запитання на кшталт "Які заклади подобаються моїм друзям?", "У яких містах бували мої родичі?". Зрозуміло, що жодна система повнотекстового пошуку (якими є сучасні пошукові машини) не зможе дати відповіді на ці запитання. Якщо для вирішення схожих завдань використовувати системи, побудовані на реляційних базах даних (якими є кошти BI і OLAP), то, очевидно, набір питань, на які система зможе відповісти, жорстко визначено набором сутностей, представлених в базі даних (закладу, міста, родичі повинні бути представлені у відповідних таблицях і полях БД). Принципова відмінність семантичної аналітики полягає в тому, що ми можемо вводити в модель будь-які нові сутності (не родичі, а колеги, не міста, а готелі, і т.д.), і використовувати їх при "формулюванні питань" системі, не змінюючи при це структури і змісту інших частин моделі.

Основні вимоги до функціоналу проектованої BI платформи:

- Можливість додавати sparql-endpoint.
- Додавання власних даних до rdf-сховища(import/export csv).
- Можливість робити sparql-запити до різних sparql-endpoint та до локального сховища.
- Створення графіків по створеним даним.
- Зручний веб-інтерфейс.

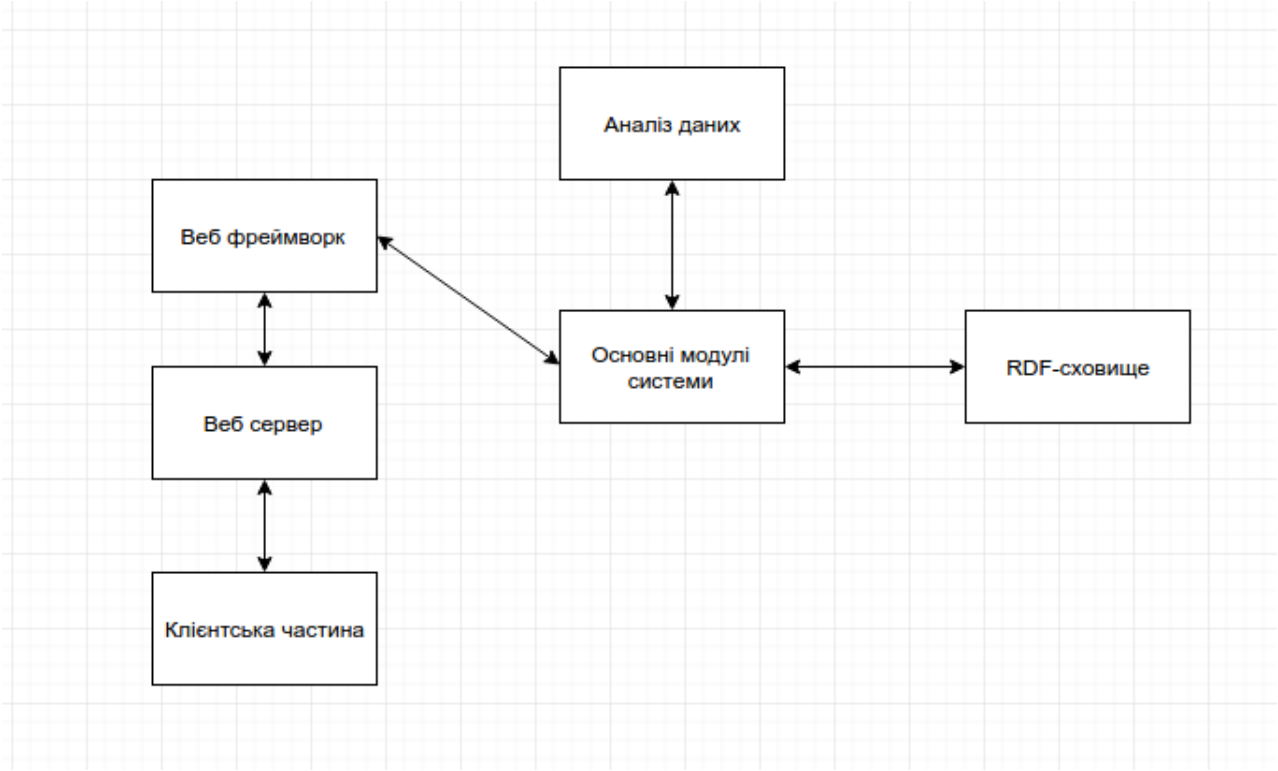

Рисунок 3.1 Архітектура додатку

3.2 Вибір інструментів

Мова програмування: Python

Python (найчастіше вживане прочитання — «Па́йтон») — інтерпретована об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою. Розроблена в 1990 році Гвідо ван Россумом. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням

роблять її привабливою для швидкої розробки програм, а також як засіб поєднання існуючих компонентів. Python підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій так і у вихідній формі на всіх основних платформах. В мові програмування Python підтримується декілька парадигм програмування, зокрема: об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована.

Бібліотека для аналізу даних: pandas

Pandas — вільна бібліотека Python, створена для проведення зручних маніпуляцій з даними та аналізу. Зокрема, дана бібліотека пропонує можливості роботи з числовими таблицями та часовими рядами. Назву бібліотека отримала від терміну з економетрії «panel data» — багатовимірного струкрутованого набору данних.

Можливості бібліотеки:

- DataFrame — тип данних з інтегрованою індексацією, створений для маніпуляцій над данними
- Інструменти для зчитування та запису данних таких розширень, як: CSV, Excel, JSON, SAS, FWF.
- Обробка відсутніх даних
- Підтримка операцій типу Group by
- Вставка та видалення строк/стовпців
- Злиття та приєднання датасетів
- Ієрархічна індексація осей для роботи з високорозмірними даними в низькорозмірних структурах даних
- Функціональність для роботи з часовими рядами
- Бібліотека Pandas оптимізована для отримання максимальної продуктивності при роботі, тому критичні частини коду написані на Cython та C.

Веб-фрейморк: Flask

Flask — мікрофреймворк для створення веб-додатків на мові програмування Python, використовує набір інструментів Werkzeug, а також шаблонизатор Jinja2.

Веб-сервер: nginx

nginx (engine x) — вільний веб-сервер і проксі-сервер. Є версії для сімейства Unix-подібних операційних систем (FreeBSD, GNU/Linux, Solaris, Mac OS X) та Microsoft Windows.

Розробляється Ігорем Сисоєвим з 2002-го року для компанії Rambler і постійно вдосконалюється. Восени 2004 року вийшов перший публічно доступний реліз.

З 2011 року розробкою програми опікується заснована Ігорем Сисоєвим компанія Nginx Inc., яка розвиває вільну та комерційні версії продукту.

Візуалізація даних: visjs

Динамічна бібліотека візуалізації. Бібліотека покликана бути проста у використанні, можливість обробляти великі обсяги динамічних даних.

RDF-сховище: allegrograph

AllegroGraph – це сховище триплетів із закритим кодом, яке призначене для зберігання RDF триплетів(стандартний формат для Linked Data).

Python драйвер для allegrograph: AllegroGraph Python Sesame API

Робота з RDF: RDFlib

Архітектура з використанням обраних технологі зображена на рисунку 3.2

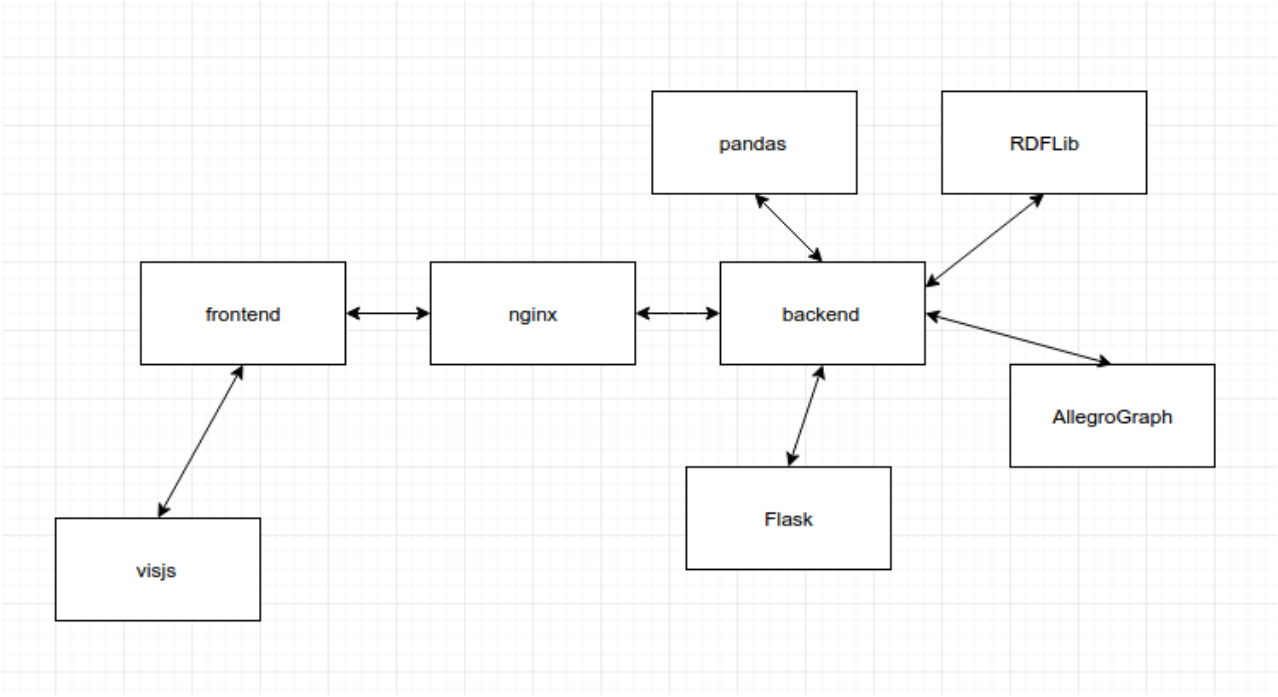

Рисунок 3.2 Оновлена архітектура додатку

3.3 Мова запитів SPARQL

Оскільки запити до системи здійснюються з допомогою SPARQL доречно оглянути синтаксис цієї мови.

Ймовірно, самі по собі мови представлення онтологій не були б так сильно потрібні, якщо б не виникало необхідності автоматично обробляти, поповнювати вміст онтологій і виконувати до них запити. Найбільш популярним серед мов запитів до RDF-сховищ на сьогоднішній день є мова SPARQL.

Елементи мови SPARQL – Суб'єкт-Предикат-Об'єкт

Розглянемо дещо спрощений синтаксис SPARQL-запиту. Припустимо, онтологія містить наступні RDF-триплети:

(Foo1, category, "Total Members");

(Foo1, rdf:value, 199);

(Foo2, category, "Total Members");

(Foo2, rdf:value, 200);

(Foo2, category, "CATEGORY X");

(bar, category, "CATEGORY X");

(bar, rdf:value, 358).

Простежимо за ходом виконання запиту

SELECT ?cat ?val WHERE {?x rdf:value ?val. ?x category ?cat.

FILTER (?val>=200).}

Семантика запиту: видайте всі об'єкти cat предиката category, суб'єкт якого (x) є також суб'єктом предиката rdf:value зі значенням val, не меншим 200. Разом зі значеннями cat видати відповідні значення val.

Хід виконання запиту:

На місце змінної x можуть бути підставлені Foo1, Foo2 та bar (з вихідною онтології), причому Foo2 може бути підставлений двічі, оскільки має 2 властивості category).

При підстановці Foo1 значення змінної val не задовольняє обмеження в реченні FILTER. У всіх інших випадках все умови запиту виконані (див. результат).

Результат виконання запиту 3 пари значень (cat, val)

[

["Total Members", 200],

["CATEGORY X", 200],

["CATEGORY X", 358]

]

3.4 Результати роботи прототипу

В роботі як було зазначено вище розроблявся прототип BI-платформи. Для цього нам потрібно зв’язати всі компоненти разом, та створити платформу. За основу взята мова програмування Python та веб-фреймворк Flask.

Зробимо з допомогою мови SPARQL запит до бази знань. В даному прототипу використовується загально відома dbpedia, але передбачена можливість інтеграції та подальшого використання. Зробимо тестовий запит до

бази(рисунок 3.3):

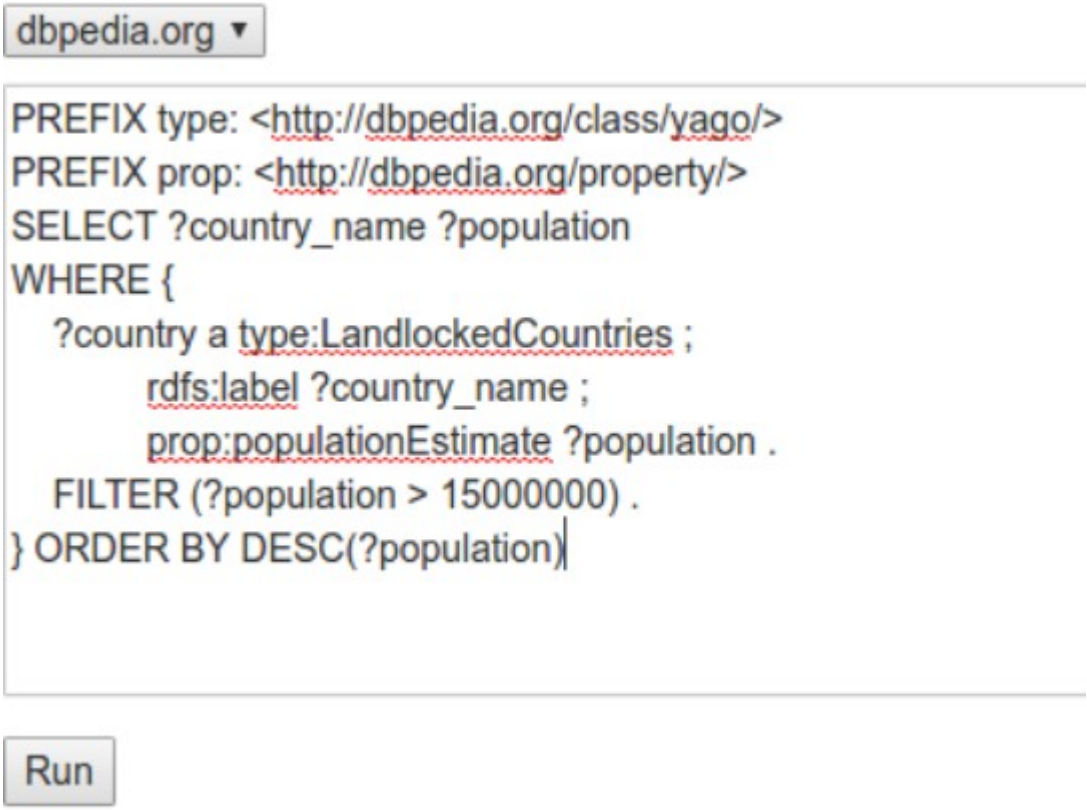

Рисунок 3.3 Тестовий запит до бази

Після надсилання запиту ми отримуємо відповідь у вигляді таблиці. Таблицю ми можемо редагувати, щоб мати змогу редагувати можливі неправильні данні. Приклад таблиці на рисунку 3.4:

Рисунок 3.4 Вихідна таблиця

Якщо вихідних даних забагато, і необхідна обробка за допомогою якихось сторонніх засобів є можливість скачати дані у вигляді CSV-файлу. Приклад вмісту на рисунку 3.4:

```
[Ethiopia , 90076012 ],
[Afghanistan , 32564342 ],
[Uzbekistan , 31025500 ],
[Kazakhstan , 17563300 ],
[Burkina Faso , 17322796 ],
[Malawi , 16407000 ],
[Zambia , 16212000 ],
```

Рисунок 3.5 Приклад експортованого файлу з вихідними даними

Якщо дані правильно відформатовані, то передбачена можливість створення графіків. Приклад створений на даних зазначених вище зображений на рисунку 3.6:

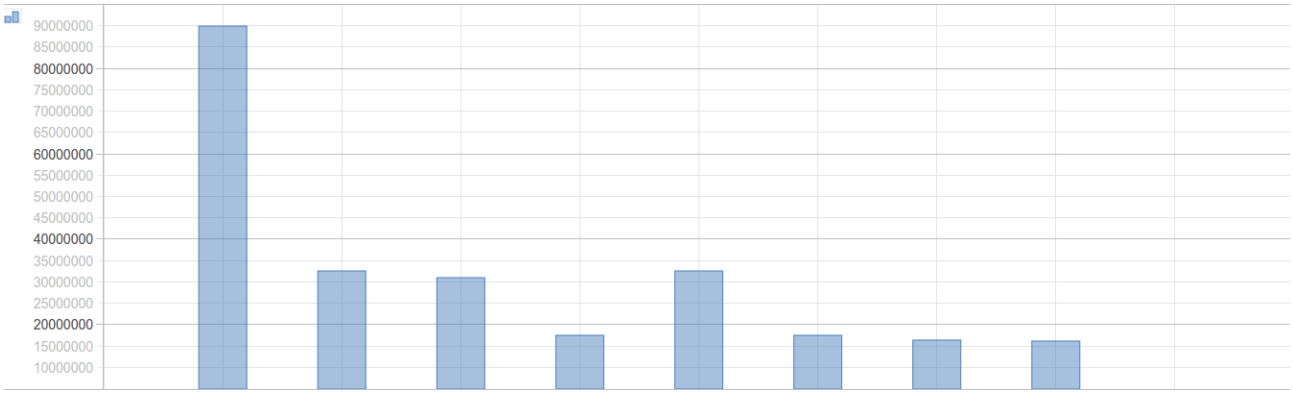

Рисунок 3.6 Візуалізація даних

3.5 Висновки до розділу

Було спроектовано BI-платформу, яка відповідає сучасним вимогам бізнесу. При проектуванні враховані можливі сценарії розвитку системи (наприклад додання можливості комбінувати різні джерела даних). Для розроблення прототипу використовувались мови програмування Python, Javascript. Python відіграє ключову роль в системі, адже відповідає за надання

доступу до даних, їх обробка, зберігання та т.п. JavaScript допомагає створити інтерактивний та зручний інтерфейс для кінцевого користувача, вирішуючи проблему редагування даних в браузері.

4. ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

4.1 Вступ

У даному розділі проводиться оцінка основних характеристик програмного продукту, призначений для керування системою бінес-аналітики. Бізнес-логіка системи розроблена за допомогою мови програмування Python. Інтерфейс користувача створений за допомогою технології HTML.

Програмний продукт призначено для використання на персональних комп'ютерах під управлінням операційної системи Windows, Linux.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимальної, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. Як прямі, так і побічні витрати розподіляються по продуктам та послугам у залежності від потрібних на кожному етапі виробництва обсягів ресурсів. Виконані на цих етапах дії у контексті метода ФВА називаються функціями.

Мета ФВА полягає у забезпеченні правильного розподілу ресурсів, виділених на виробництво продукції або надання послуг, на прямі та непрямі витрати. У даному випадку – аналізу функцій програмного продукту й виявлення усіх витрат на реалізацію цих функцій.

Фактично цей метод працює за таким алгоритмом:

• визначається послідовність функцій, необхідних для виробництва продукту. Спочатку – всі можливі, потім вони розподіляються по двом групам:

ті, що впливають на вартість продукту і ті, що не впливають. На цьому ж етапі оптимізується сама послідовність скороченням кроків, що не впливають на цінність і відповідно витрат.

• для кожної функції визначаються повні річні витрати й кількість робочих часів.

• для кожної функції наоснові оцінок попереднього пункту визначається кількісна характеристика джерел витрат.

• після того, як для кожної функції будуть визначені їх джерела витрат, проводиться кінцевий розрахунок витрат на виробництво продукту.

4.2 Постановка задачі техніко-економічного аналізу

У роботі застосовується метод ФВА для проведення техніко-економічний аналізу розробки.

Відповідно цьому варто обирати і систему показників якості програмного продукту.

Технічні вимоги до продукту наступні:

• програмний продукт повинен функціонувати на персональних комп’ютерах із стандартним набором компонент;

• забезпечувати високу швидкість обробки великих об’ємів даних у реальному часі;

• забезпечувати зручність і простоту взаємодії з користувачем або з розробником програмного забезпечення у випадку використовування його як модуля;

• передбачати мінімальні витрати на впровадження програмного продукту.

4.1.1. Обґрунтування функцій програмного продукту

Головна функція F_0– розробка програмного продукту, який аналізує процес за вхідними даними та будує його модель для подальшого прогнозування. Виходячи з конкретної мети, можна виділити наступні основні

функції ПП:

F_1 – вибір мови програмування;

F_2 – вибір веб-фреймворку;

F_3 – вибір бібліотеки аналізу даних.

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

а) мова програмування Python;

б) мова програмування Golang;

Функція F_2:

а) Flask.

б) Django;

в) Revel.

Функція F_3:

а) pandas;

б) scikitlearn.

в) awesome-machine-learning

4.1.2. Варіанти реалізації основних функцій

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1). На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 4.1).

Морфологічна карта відображує всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП.

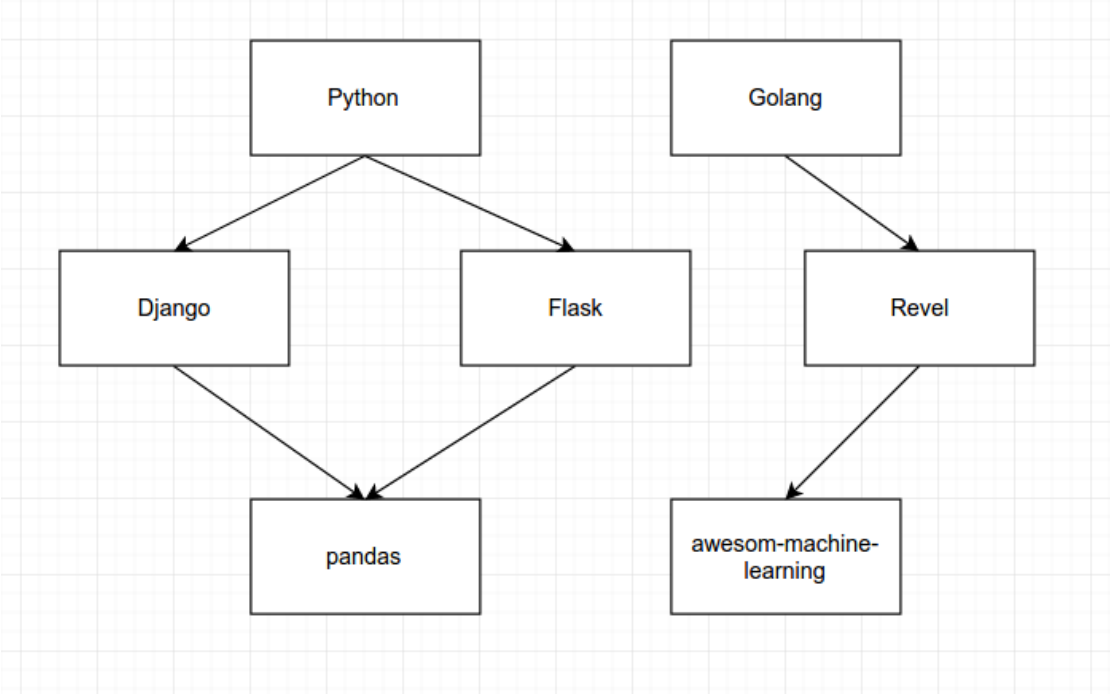

Рисунок 4.1 – Морфологічна карта

Таблиця 4.1 – Позитивно-негативна матриця

Основні функції	**Варіанти реалізації**	**Переваги**	**Недоліки**
F1	А	Займає менше часу при написанні коду	Низька швидкодія
	Б	Швидкість	Довгий час розробки
F2	А	Кількість готових рішень	Низька швидкодія
	Б	Популярність, простота	Низька кількість готових рішень
	В	Швидкодія	Значна кількість програмних помилок
F3	А	Готових рішень	Низька швидкодія
	Б	Простота створення	Незначна кількість реалізованих алгоритмів

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція *F1:*

Оскільки бюджет проекту мінімальний, реалізовуємо проект на мові Python, адже це дозволить зекономити на часу розробки.

Функція *F2:*

Реалізація проекту виконується з допомогою мови програмування Python, тому варіант В) відкидуємо.

Функція *F3:*

Оскільки, програмний продукт реалізується на мові Python, використовуємо варіант А як єдиний можливий.

Таким чином, будемо розглядати такі варіанти реалізації ПП:

1. F1а – F2а – F3а
2. F1а – F2б – F3а

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.1 Обґрунтування системи параметрів ПП

4.1.1 Опис параметрів

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

- *X1* – швидкодія відповіді клієнту;
- *X2* – кількість ресурсів на користувача;
- *X3* – завантаження даних до бази даних;
- *X4* – потенційний об’єм програмного коду.

X1:Швидкість відповіді на запит клієнта.

*X2:*Відображає необхідні ресурси для обслуговування одного користувача.

*X3:*Швидкість завантаження данних до бази даних.

*X4:*Показує розмір програмного коду який необхідно створити безпосередньо розробнику.

4.1.3 Кількісна оцінка параметрів

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 4.2.

Таблиця 4.2 – Основні параметри ПП

Назва Параметра	**Умовні позначення**	**Одиниці виміру**	**Значення параметра**		
			гірші	**середні**	**кращі**
Швидкодія відповіді клієнту	X1	мс	100	50	20
Кількість ресурсів на користувача	X2	Мб	32	16	8
Завантаження даних до бази даних	X3	мс	2500	1200	600
Потенційний об’єм програмного коду	X4	кількість строк коду	1500	1000	700

За даними таблиці 4.2 будуються графічні характеристики параметрів – рис. 4.2 – рис. 4.5.

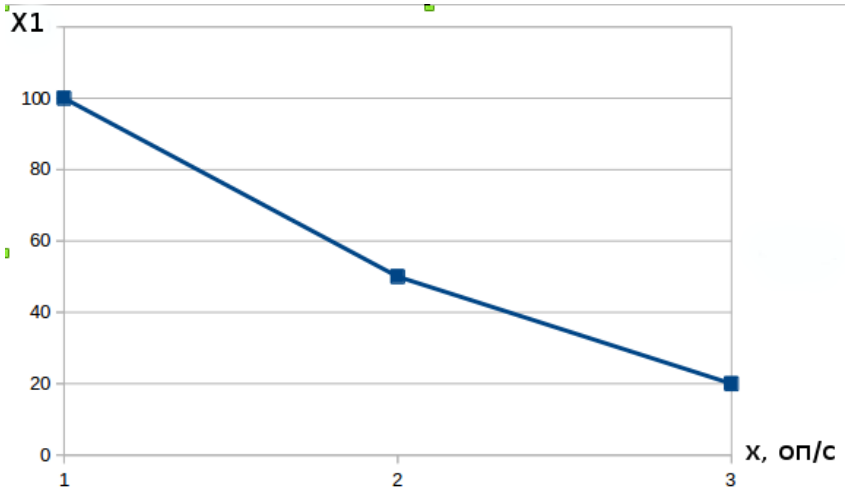

Рисунок 4.2 – X1, Швидкодія відповіді клієнту

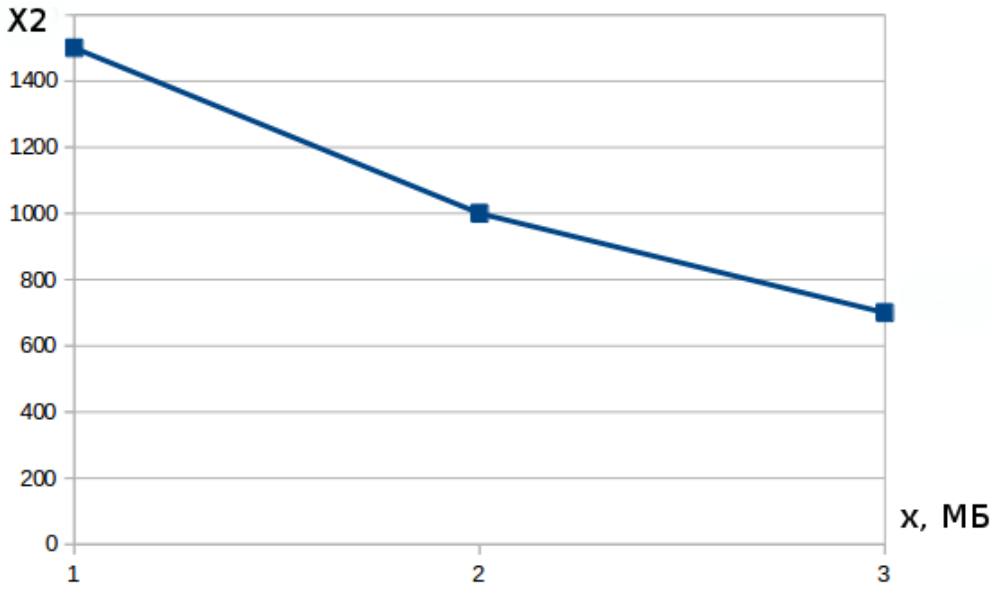

Рисунок 4.3 – X2, Кількість ресурсів на користувача

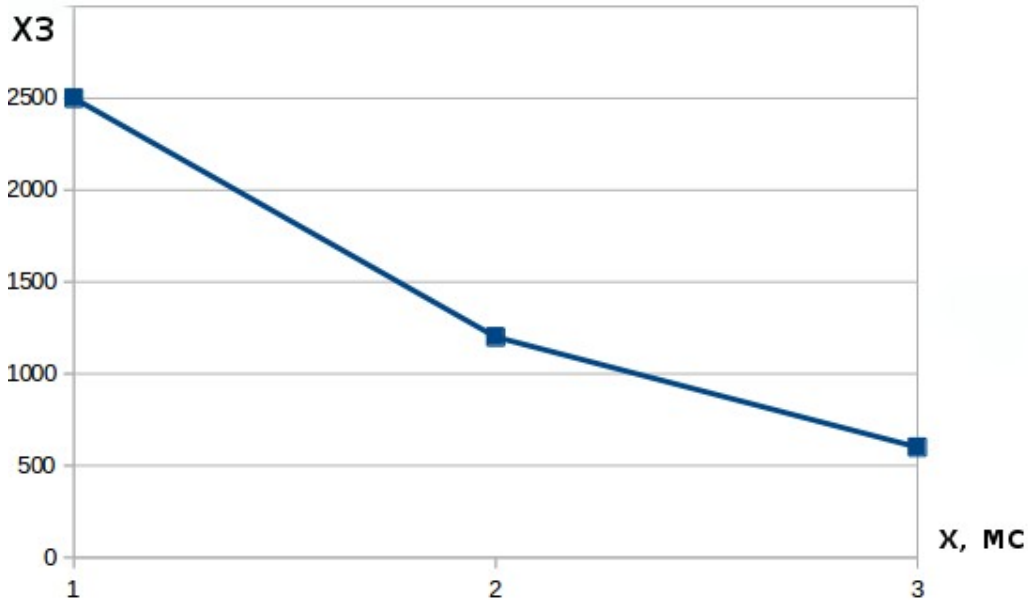

Рисунок 4.4 – X3, Завантаження даних до бази даних

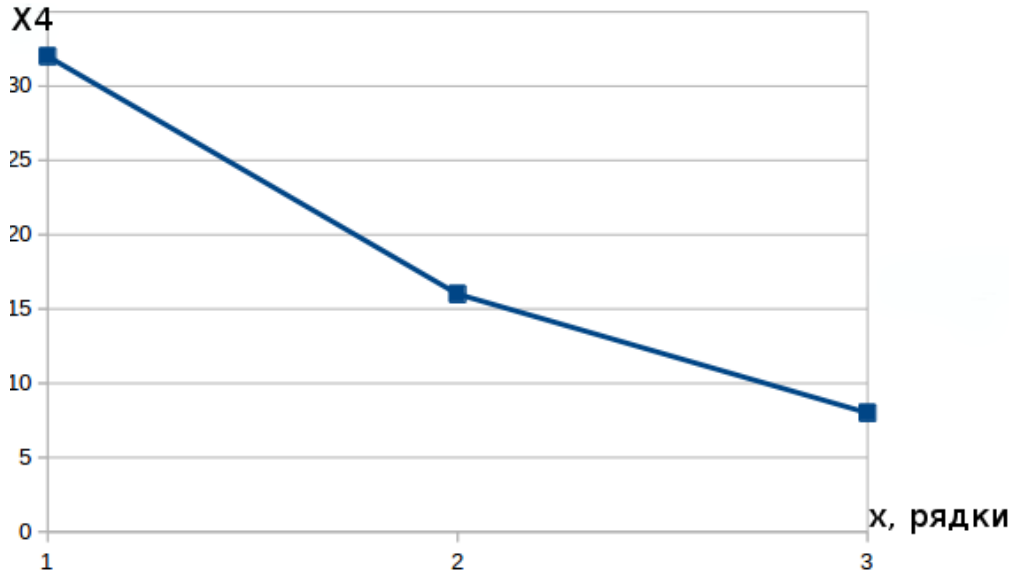

Рисунок 4.5 – X4, Потенційний об’єм програмного коду

4.1.2 4.1.4. Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

- визначення рівня значимості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;
- обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 4.3.

Таблиця 4.3 – Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія відповіді клієнту	мс	4	3	4	4	4	4	4	27	0,75	0,56
X2	Кількість ресурсів на користувача	Мб	4	4	4	3	4	3	3	25	-1,25	1,56
X3	Завантаження даних до бази даних	мс	2	2	1	2	1	2	2	12	-14,25	203,06
X4	Потенційний об’єм програмного коду	кількість строк коду	5	6	6	6	6	6	6	41	14,75	217,56
	Разом		15	15	15	15	15	15	15	105	0	420,75

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 105, \quad (4.1)$$

де N – число експертів, n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 26,25. \quad (4.2)$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T \tag{4.3}$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 420,75. \tag{4.4}$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 420,75}{7^2(5^3 - 5)} = 1,03 > W_k = 0,67 \tag{4.5}$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.4.

Таблиця 4.4 – Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	=	>	=	<	=	<	<	<	0,5
X1 і X3	<	<	<	<	<	<	<	<	0,5
X1 і X4	>	>	>	>	>	>	>	>	1,5
X2 і X3	<	<	<	<	<	<	<	<	0,5
X2 і X4	>	>	>	>	>	>	>	>	1,5
X3 і X4	>	>	>	>	>	>	>	>	1,5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \begin{cases} 1,5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j \\ 0.5 \text{ при } X_i < X_j \end{cases}$$

З отриманих числових оцінок переваги складемо матрицю $A = \| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{в\,i}=\frac{b_i}{\sum_{i=1}^{n} b_i}\ \text{, де}\quad b_i=\sum_{i=1}^{N} a_{ij}\ . \tag{4.6}$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%).На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{в\,i}=\frac{b_i^{'}}{\sum_{i=1}^{n} b_i^{'}},\quad \text{де}\quad b_i^{'}=\sum_{i=1}^{N} a_{ij} b_j\ . \tag{4.7}$$

Як видно з таблиці 4.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 4.5 – Розрахунок вагомості параметрів

Параметри x_i	Параметри x_j				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	$K_{в\,i}$	b_i^1	$K_{в\,i}^1$	b_i^2	$K_{в\,i}^2$
X1	1,0	0,5	0,5	1,5	3,5	0,219	22,25	0,216	100	0,215
X2	1,5	1,0	0,5	1,5	4,5	0,281	27,25	0,282	124,25	0,283
X3	1,5	1,5	1,0	1,5	5,5	0,344	34,25	0,347	156	0,348
X4	0,5	0,5	0,5	1,0	2,5	0,156	14,25	0,155	64,75	0,154
Всього:					16	1	98	1	445	1

4.4. Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2*(кількість ресурсів на користувача) та *X1* (швидкодія мови програмування) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X3 (*Завантаження даних до бази даних)

обрано не найгіршим (не максимальним), тобто це значення відповідає або варіанту а) 1000 мс або варіанту б) 80мс.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j}, \quad (4.8)$$

де n – кількість параметрів;

$K_{вi}$ – коефіцієнт вагомості i–го параметра;

B_i – оцінка i–го параметра в балах.

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1(X1)	А	50	8.367	0,185	1.548
F2(X2)	А	16	10.51	0,183	1,924
F3(X3,X4)	А	1200	4.8	0.284	1,67
	Б	1000	4.08	0,348	0,616

За даними з таблиці 4.6 за формулою

$$K_K = K_{ТУ}[F_{1k}] + K_{ТУ}[F_{2k}] + ... + K_{ТУ}[F_{zk}], \quad (4.9)$$

визначаємо рівень якості кожного з варіантів:

$K_{K1} = 1.548 + 1,924 + 1,67 = 5.142$

$K_{K2} = 1.548 + 1,924 + 0,616 = 4.08$

Як видно з розрахунків, кращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.2 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок

трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань. Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_M \cdot K_{СТ} \cdot K_{СТ.М}, \quad (4.10)$$

де T_P – трудомісткість розробки ПП;$K_П$ – поправочний коефіцієнт;$K_{СК}$ – коефіцієнт на складність вхідної інформації; K_M – коефіцієнт рівня мови програмування;$K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм; $K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення.

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: T_P =90 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П$ = 1.7. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $K_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $K_{СТ}$ = 0.8. Тоді, за формулою 5.1, загальна трудомісткість програмування першого завдання дорівнює:

$T_1 = 90 \cdot 1.7 \cdot 0.8 = 122.4$ людино-днів.

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $Т_Р$ =27 людино-днів, $К_П$ =0.9,$К_{СК}$ = 1,$К_{СТ}$ =0.8:

$Т_2$ = 27 · 0.9 · 0.8 = 19.44 людино-днів.

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$Т_I$ = (122.4 + 19.44 + 4.8 + 19.44) · 8 = 1328,64 людино-годин;

$Т_{II}$ = (122.4 + 19.44 + 6.91 + 19.44) · 8 = 1345.52 людино-годин;

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь три програмісти з окладом 14000 грн., один фінансовий аналітик з окладом 13500грн. Визначимо зарплату за годину за формулою:

$$СЧ = \frac{М}{T_m \cdot t} грн., \quad (4.11)$$

де М – місячний оклад працівників; T_m – кількість робочих днів тиждень; t – кількість робочих годин в день.

$$СЧ = \frac{3*14000+13500}{3 \cdot 21 \cdot 8} = 46,62 грн. \quad (4.12)$$

Тоді, розрахуємо заробітну плату за формулою

$$СЗП = C_ч \cdot T_i \cdot КД \quad (4.13)$$

де $С_Ч$– величина погодинної оплати праці програміста; T_i – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП}$ = 110.12· 1328.64 · 1.2 = 175571.8 грн.

II. $С_{ЗП}$ = 110.12· 1345.52 · 1.2 = 177802.659 грн.

Відрахування на єдиний соціальний внесок в залежності від групи професійного ризику (II клас) становить 22%:

I. $С_{ВІД} = С_{ЗП}$ · 0.22 = 175571.8*0.22 = 38625.796 грн.

II. $С_{ВІД} = С_{ЗП}$ · 0.22 = 177802.659*0.22 = 39116.58 грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 14000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 14000 \cdot 0,2 = 33600$ грн.

З урахуванням додаткової заробітної плати:

$С_{ЗП} = С_Г \cdot (1 + К_З) = 33600 \cdot (1 + 0.2) = 40320$ грн.

Відрахування на єдиний соціальний внесок:

$С_{ВІД} = С_{ЗП} \cdot 0.3677 = 40320 * 0,22 = 8870.4$ грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 10000 грн.

$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 10000 = 2875$ грн.,

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 10000 \cdot 0.05 = 575$ грн.,

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 8 – 16) \cdot 8 \cdot 0.9 = 1706.4$ годин,

де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день; $К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1706,4 \cdot 0,156 \cdot 0.2 \cdot 2.0218 = 107.639$ грн.,

де $N_С$ – середньо-споживча потужність приладу; $К_З$– коефіцієнтом зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$С_Н = Ц_{ПР} \cdot 0.67 = 10000 \cdot 0,67 = 6700$ грн.

Тоді, річні експлуатаційні витрати будуть:

$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$

$С_{ЕКС} = 177802 + 8870.4 + 2875 + 575 + 107.639 + 6700 = 196930.039$ грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 196930.039 / 1706,4 = 115.406$ грн/час.

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$С_М = С_{М\text{-}Г} \cdot Т$

I. $С_М = 115.406 * 1328.64 = 153333$ грн.;

II. $С_М = 115.406 * 1345.52 = 155281$ грн.;

Накладні витрати складають 67% від заробітної плати:

$С_Н = С_{ЗП} \cdot 0,67$

I. $С_Н = 153333 * 0,67 = 102733.11$ грн.;

II. $С_Н = 155281 * 0,67 = 104038.27$ грн.;

Отже, вартість розробки ПП за варіантами становить:

$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н$

I. $С_{ПП} = 175571.8 + 38625.796 + 153333 + 102733.11 = 470263.706$ грн.;

II. $С_{ПП} = 177802.659 + 39116.58 + 155281 + 104038.27 = 476238.509$ грн.;

4.3 Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$К_{ТЕРj} = К_{Кj} / С_{Фj}$,

$К_{ТЕР1} = 5.142 / 470263.706 = 0,11 \cdot 10^{-5}$;

$К_{ТЕР2} = 4.08 / 476238.509 = 0,85 \cdot 10^{-6}$;

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 0,11 \cdot 10^{-5}$.

4.4 Висновки до розділу 4

В даному розділі проведено повний функціонально-вартісний аналіз ПП, який було розроблено в рамках дипломного проекту. Процес аналізу можна умовно розділити на дві частини.

В першій з них проведено дослідження ПП з технічної точки зору: було визначено основні функції ПП та сформовано множину варіантів їх реалізації; на основі обчислених значень параметрів, а також експертних оцінок їх важливості було обчислено коефіцієнт технічного рівня, який і дав змогу визначити оптимальну з технічної точки зору альтернативу реалізації функцій ПП.

Другу частину ФВА присвячено вибору із альтернативних варіантів реалізації найбільш економічно обґрунтованого. Порівняння запропонованих варіантів реалізації в рамках даної частини виконувалось за коефіцієнтом ефективності, для обчислення якого були обчислені такі допоміжні параметри, як трудомісткість, витрати на заробітну плату, накладні витрати.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР} = 0,14 \cdot 10^{-4}$.

Цей варіант реалізації програмного продукту має такі параметри:

- мова програмування – Python;
- веб-фрейморк – Flask;
- бібліотека аналізу даних - pandas.

Даний варіант виконання програмного комплексу дає користувачу гарний функціонал і швидкодію.

ВИСНОВКИ

Дипломна робота присвячена дослідженню методам інтеграції технологій Semantic Web для Business Intelligence. Досліджено сьогоднішні тенденції розвитку цих технологій.

Насьогодні комп'ютери беруть досить обмежену участь у формуванні й обробці інформації в мережі Інтернет. Функції комп'ютерів в основному зводяться до збереження, відображення і пошуку інформації. У той же час створення інформації, її оцінка, класифікація й актулізація — усе це як і раніше виконує людина. Як включити комп'ютер у ці процеси? Якщо комп'ютер поки не можна навчити розуміти людську мову, то потрібно використовувати мову, що була б зрозумілою комп'ютеру. Тобто, в ідеальному варіанті, вся інформація в Інтернеті повинна розміщуватись двома мовами: людською мовою для людини і комп'ютерною мовою для розуміння комп'ютера. Семантична павутина — це концепція мережі, у якій кожен ресурс людською мовою був би доповнений описом, зрозумілим комп'ютеру.

Було названо і проаналізовано ряд існуючих BI-систем, що найбільше сьогодні використовуються для вирішення бізнес задач. І було зроблено висновок, що вони всі є досить великими, зручними, та мають звісно багато переваг, проте всі вони є дуже складними та дорогими рішеннями, що перешкоджає інтеграцію для середнього та малого бізнесу. Тому було вирішено створити власну BI-систему, що задовільняла б середній та малий бізнес.

В ході роботи було спроектовано систему, що задовільняє сучасним необхідностям бізнесу, використовує новітні технології, має логічну та масштабовану архітектуру. Основою реалізованого прототипу є мова програмування Python, який відповідає за backend додатку(з допомогою веб-фреймворку Flask), обробку та трансформацію даних, створення запитів до

RDF-сховищ.

В прототипі системі реалізовано наступний функціонал:

- Додавання нових знань до локального RDF-сховища
- Створення та обробка SPARQL-запитів до локального сховища та віддаленого.
- Передобробка отриманою інформації(видалення, редагування)
- Візуалізація даних(на даний момент реалізовано графік та стовпчата діаграма).

В подальшому розвитку система перспективною є автоматизація процесів введення інформації, можливо з допомогою технологій machine learning. Проте це не було метою роботи, тому залишимо це для майбутніх досліджень.

ПЕРЕЛІК ПОСИЛАНЬ

1. Sanjay Mehta BI 2.0 Technology – MAIA Intelligence Perspective. – Режим доступу: http://www.maia-intelligence.com/.– Дата доступу: 13.02.2016
2. Business intelligence definitions. – Режим доступу: http://searchdatamanagement.techtarget.com/definition/business-intelligence. – Дата доступу: 13.02.2016
3. Вэллс Д. Десять основных преимуществ Microsoft Business Intelligence, 2008, (Пер. с англ.) - Режим доступу: http://www.w3.org/TR/html4/strict.dtd. – Дата доступу: 13.02.2016
4. Neches R., Fikes R., Finin T., Gruber T., Patil R., Senator T., Swartout R. Enabling Technology for Knowledge Sharing / Neches R., Fikes R., Finin T., Gruber T., Patil R., Senator T., Swartout R. // AI Magazine. – 1999. – Vol. 12., № 3.
5. Уотсон Х., Викском Б. Современное состояние бизнес-аналитики (Business Intelligence, BI),- Режим доступу: http://www.osp.ru/os/2007/08/4489920/. – Дата доступу: 13.02.2016
6. SAP Business Objects Business Intelligence Solutions. – Режим доступу: http://www.sap.com/solutions/sapbusinessobjects/large/business-intelligence/ index.epx. – Дата доступу: 13.02.2016
7. Business Intelligence from IBM. –- Режим доступу: http://www-01.ibm.com/software/data/businessintelligence/. – Дата доступу: 13.02.2016
8. SAS Business Intelligence. – Режим доступу: http://www.sas.com/technologies/bi/. Дата доступу: 13.02.2016
9. Oracle Enterprise Performance Management and Business Intelligence. - Режим доступу: http://www.oracle.com/solutions/business_intelligence/index.html. Дата доступу: 13.02.2016
10. Semantic Web - Режим доступу: http://www.semanticweb.org. Дата доступу: 13.02.2016

11. Рогушина Ю.В., Гладун А.Я. Технологии Semantic Web и их использование при разработке интеллектуальных приложений / Рогушина Ю.В., Гладун А.Я. // Проблеми програмування. – 2008. – № 2-3. – С. 385–394.
12. Плескач В.Л., Рогушина Ю.В. Агентні технології / Плескач В.Л., Рогушина Ю.В. // Монографія. – К.: Київськ. нац. торг.-екон. ун-т, 2005. – 338 с.
13. Артемьев В. Что такое Business Intelligence? Режим доступу: http://www.osp.ru/os/2003/04/ 020.htm. – Дата доступу: 13.02.2016
14. Эверет Д. Web-сервисы на службе у Business Intelligence. – Режим доступу: http://www.iso.ru/journal/articles/ 572.html. – Дата доступу: 13.02.2016
15. Рогушина Ю.В., Гладун А.Я., Штонда В.М. Онтологический анализ Web-сервисов в интеллектуальных сетях / Рогушина Ю.В., Гладун А.Я., Штонда В.М. // Proc. of The XIII-th International Conf. "Knowledge-Dialogue-Solution", ITHEA, Sofia, V. 2, 2007. – С. 451– 459.
16. OWL-S: Semantic Markup for Web Services. The OWL Services Coalition. – Режим доступу: http://www.daml.org/services/owl-s/1.0/owls.html. – Дата доступу: 13.02.2016