Міністерство освіти і науки України
Національний університет «Львівська політехніка»

На правах рукопису

КУШНІРЕЦЬКА ІРИНА ІГОРІВНА

УДК 004.738.5 + 004.62

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ДИНАМІЧНОЇ ІНТЕГРАЦІЇ СЛАБОСТРУКТУРОВАНИХ ДАНИХ У WEB-СИСТЕМАХ

05.13.06 - інформаційні технології

Дисертація на здобуття наукового ступеня
кандидата технічних наук

Науковий керівних:
Берко Андрій Юліанович,
доктор технічних наук, професор

Ідентичність всіх примірників дисертації

ЗАСВІДЧУЮ:

Вчений секретар спеціалізованої
вченої ради *Батюк А.Є.*

Львів – 2017

ЗМІСТ

СПИСОК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ІТ	Інформаційна технологія
ІС	Інформаційна система
БД	База даних
ІД	Інтеграція даних
ІР	Інформаційний ресурс
СІД	Система інтеграції даних
EAI	Enterprise Application Integration (інтеграція корпоративних застосунків)
EDR	Enterprise Data Replication (тиражування корпоративних даних)
API	Application Programming Interface (набір визначень взаємодії різнотипного програмного забезпечення)
SOA	Service Oriented Architecture (підхід до проектування, розгортання та експлуатації програмних систем)
ODBC	Open Database Connectivity
JDBC	Java Database Connectivity
XML	Extensible Markup Language (стандарт побудови мов розмітки ієрархічно структурованих даних)
XSD	XML Schema (стандартна мова для опису XML документів)
Ajax	Asynchronous JavaScript + XML
STD	State Transition Diagram (діаграми переходів станів)
RDF	Resource Description Framework (уніфікована модель даних)
OWL	Web Ontology Language (мова web-онтологій)
SQL	Structured query language (мова структурованих запитів)
SPARQL	Семантична мова запитів для БД, що дозволяє маніпулювати даними, збереженими в RDF форматі
GUI	Graphical user interface (графічний інтерфейс користувача)

ВСТУП

Актуальність теми. Зростаюча кількість ресурсів та послуг, доступних у глобальній мережі та можливостей, які пропонуються Web 2.0 штовхають кінцевих користувачів еволюціонувати від пасивних споживачів інформації в інформаційних виробників, які в змозі отримати доступ до існуючих інформаційного контенту та маніпулювати ним для того, щоб генерувати новий зміст. З точки зору людино-машинної взаємодії, це вимагає нових парадигм комунікацій, які повинні дозволити людям отримати доступ до інформаційних даних, перемістити їх в особисті інтерактивні робочі області, де вони можуть інтегрувати їх, а також, при потребі створювати новий інформаційний контент, використовуючи наявний. Одним з рішень даної проблеми є розробка систем динамічної інтеграції слабоструктурованих даних. Найперспективнішими в наш час є системи інтеграції даних, що працюють використовуючи технологію Mashup. Mashup - це технологія для проектування web-систем, яка дозволяє користувачам об'єднувати різного роду дані з декількох джерел в один інтегрований інструмент. Mashup системи відкривають нові і широкі можливості використання інформаційних ресурсів, і тому при створенні програмних продуктів все більше розробників переходять на використання Mashup технології інтеграції даних. Значну частину представлених в Web 2.0 даних доцільно розглядати як слабоструктуровані - вони мають деяку семантичну структуру, однак та інформація про їх структуру, яка зазвичай асоціюється зі схемою даних, може міститися в них самих. Однак, існуючі методології та інструментальні засоби побудови програмних систем орієнтовані на добре структуровані задачі із достатньо формалізованими предметними областями і постійними локальними джерелами знань.

Розроблення технології динамічної інтеграції слабоструктурованих даних із врахуванням змісту даних, що знаходяться в різних web-системах та здатної враховувати особливості різних вхідних інформаційних систем актуально

беручи до уваги такі фактори як: недостатність теоретичного обґрунтування методів семантичного опрацювання даних систем динамічної інтеграції даних, а також, необхідність уніфікації програмних засобів динамічного опрацювання інформаційних ресурсів інтегрованих систем Mashup. Практичний чинник опрацювання даних різноманітної структури і форматів подання, що знаходяться в різних web-системах при їх динамічній інтеграції пов'язаний з вирішенням задач покращення пошуку інформації в Інтернет в сьогоднішню еру стрімкого зростання об'єму інформаційних даних різноманітного характеру та змісту, інтеграції гетерогенних інформаційних систем, що вміщають інформаційні ресурси різного роду і змісту та величезний попит на системи динамічної інтеграції даних на основі технології Mashup.

Теоретичний чинник опрацювання даних різноманітної структури і форматів подання, що знаходяться в різних web-системах при їх динамічній інтеграції пов'язаний із ІТ семантичного динамічного опрацювання даних при їх інтеграції. Стандартні технології для вирішення однотипних завдань інтеграції почали розроблятися ще на початку 90-х років. Дослідження в напрямку Mashup інтеграції даних активно почали розвиватися 8 років тому. .В наукових роботах Фішера (T. Fischer), Бакалова (F. Bakalov), Наєрца (A. Nauerz), Алтінела (M. Altinel), Брауна (P. Brown), Клайна (S. Cline), Маркла (V. Markl), May (L. Mau) та Сінга (A. Singh) наведено характеристику сучасних підходів створення Mashup систем та опис роботи деяких сучасних Mashup застосунків. Хендрік (Hendrik), Анйомшоа (A. Anjomshoaa) та ін. описують, в своїх роботах, Mashup засоби для аналізу різного об'єму даних. Основні нюанси моделювання Mashup простору наводять такі науковці, як: Абітбул (S. Abiteboul), Гріншпен (O. Greenshpan) та Мілоу (T. Milo). Предметно-орієнтовану мову для web-інтерфейсів та сервісів Mashup систем пропонують Максімільян (E. Maximilien), Вілкінсон (H. Wilkinson) і Тай (S. Tai). Згідно проведеним науковим аналізом встановлено, що даний сегмент ІТ є мало дослідженим. Кожний окремий проект реалізують практично з початку,

фактично на основі своїх ідей та рішень. У літературі надзвичайно мало висвітлені суттєві теоретичні обґрунтування, дослідження, висновки, рекомендації, узагальнення для проектування систем динамічної інтеграції даних на основі технології Mashup та опрацювання даних в таких системах, враховуючи їх семантику. Виникла потреба в аналізі, узагальнені та обґрунтуванні існуючих підходів створення і роботи систем динамічної інтеграції даних на основі технології Mashup. Актуальною є задача створення комплексу технологічних засобів на основі теоретичного обґрунтування методів, моделей і принципів семантичного опрацювання даних різноманітної структури і форматів подання, що знаходяться в різних web-системах при їх динамічній інтеграції.

Зв'язок роботи з науковими програмами, планами, темами. Тема дисертаційної роботи відповідає науковому напряму кафедри інформаційних систем та мереж Національного університету «Львівська політехніка» на тему «Дослідження, розроблення і впровадження інтелектуальних розподілених інформаційних технологій та систем на основі ресурсів баз даних, сховищ даних, просторів даних та знань з метою прискорення процесів формування сучасного інформаційного суспільства». Результати досліджень, відображені у дисертаційній роботі, отримані у рамках виконання науково-дослідної роботи кафедри інформаційних систем та мереж Національного університету «Львівська політехніка» за темою «Розроблення інтелектуальних розподілених систем на основі онтологічного підходу з метою інтеграції інформаційних ресурсів» № держреєстрації 0115U004228 (автор вдосконалила структурну модель Mashup системи, розробила метод опису структури і змісту вхідних даних та метод формування контенту об'єднаного динамічного набору даних, який має загальну структуру і єдиний зміст).

Мета і завдання дослідження. Мета дисертаційної роботи - підвищення якості результатів пошуку даних у web-системах шляхом розроблення

інформаційної технології динамічної інтеграції слабоструктурованих даних у web-системах.

Для досягнення зазначеної мети необхідно вирішити такі основні **завдання дослідження**:

- провести аналіз сучасного стану опрацювання даних web-систем в сенсі застосування технології Mashup;
- розробити метод опису структури і змісту вхідних даних Mashup систем;
- розробити метод формування об'єднаного динамічного набору вихідних даних, який має узгоджену структуру і єдиний зміст;
- удосконалити структурну модель системи динамічної інтеграції слабоструктурованих даних, що працює на основі технології Mashup;
- розробити програмні засоби динамічної інтеграції слабоструктурованих даних у web-системах.

Об’єкт дослідження – процеси динамічної інтеграції слабоструктурованих даних у web-системах.

Предмет дослідження – інформаційні технології динамічного об'єднання наборів даних на основі принципів Mashup.

Методи дослідження. Для вирішення проблеми аналізу та дослідження сучасного стану опрацювання даних web-систем в сенсі застосування технології Mashup було використано теорію систем та системний аналіз. Для розроблення методів та алгоритмів опрацювання даних інформаційних систем, що інтегруються було використано онтології та стандарт онтологічного моделювання IDEF5, теорію множин та математичну логіку. При побудові програмних засобів комплексного динамічного опрацювання даних різноманітної структури і різних форматів подання було використано методи об'єктно-орієнтованого програмування та методи сервісно-орієнтованої архітектури.

Наукова новизна одержаних результатів. У процесі теоретичних та експериментальних досліджень отримано такі нові наукові результати:

Вперше

- Розроблено метод опису структури і змісту вхідних даних Mashup системи шляхом застосування онтологій для класифікації інформаційних ресурсів, що дало змогу виконувати процеси інтеграції даних із врахуванням їх змісту.

- Розроблено метод формування об'єднаного динамічного набору даних через застосування процедур лінгвістичного аналізу змісту запиту користувача, що забезпечило підвищення релевантності та узгодженості результатів динамічної інтеграції слабоструктурованих даних.

Удосконалено

- Структурну модель системи динамічної інтеграції слабоструктурованих даних на основі технології Mashup, що відрізняється від відомих доданням функціональних компонентів визначення та опису семантики вхідних та вихідних наборів даних, що дало змогу підвищити якість результатів застосування таких систем.

Отримала подальший розвиток Mashup технологія динамічної інтеграції даних через запровадження додаткових можливостей опрацювання семантики інформаційних ресурсів, що дало змогу підвищити якість результатів інформаційно-пошукових web-систем.

Практичне значення одержаних результатів. Розроблено рекомендації проектування структури Mashup системи динамічної інтеграції слабоструктурованих даних, відмінної від існуючих через наявність підсистем динамічного семантичного опрацювання інформаційних ресурсів, що дає змогу покращити якість результатів роботи Mashup системи. Розроблено та впроваджено програмні засоби, які дають можливість підвищити якість даних, отриманих в результаті динамічної інтеграції: збільшення точності до 6 %, зменшення інформаційного шуму до 6%, покращення узгодженості результатів інформаційних даних до 10% у системі, що працює за технологією Mashup.

Одержані у роботі результати реалізовано в проектах: News Mashup from Social Networks - система моніторингу новин із соціальних мереж (http://snmashup.in), елементи якої використано при побудові проектів для: ТОВ «Торгова група «Тиса», ДП «Латориця» та ФОП Сидоряк Р.Б.; Shares and Discounts Mashup – система пошуку знижки на купівлю товару, отримання послуги тощо (http://discountsmashup.in.ua); системах аналізу та опрацювання даних, розроблених для КЗ ЛОР «Львівське обласне патологоанатомічне бюро» та стоматологічної клініки «Улюблений доктор», що підтверджено відповідними актами впровадження. Результати впроваджено також в навчальному процесі НУ «Львівська політехніка» під час викладання дисципліни «Об'єктно-орієнтоване програмування».

Особистий внесок здобувача. Усі наукові результати дисертаційної роботи отримані дисертантом особисто. У наукових працях, опублікованих у співавторстві, здобувачу належать: [1, 2, 10] – характеризовано проектування Mashup системи та формування контенту динамічного набору даних, який має загальну структуру і єдиний зміст; [4] – опис процесу та алгоритм побудови загальної структурної онтологічної інформаційної моделі даних систем, що інтегруються; [5] – концептуалізація та моделювання роботи Mashup системи, метод опису структури і змісту вхідних даних; [6, 14] – структурно-динамічна модель Mashup системи в певний дискретний момент часу; [7, 10, 11] – опис інтеграції слабо-структурованих даних на різних рівнях складності, принципи перетворення слабоструктурованих даних у структуровані; [14, 16] - досліджено використання онтологій для побудови Mashup систем; [3, 12, 13] – опис основних складових математичної моделі онтології інформаційної системи для семантичного пошуку і зберігання даних; [15] – характеристика трирівневої побудови Mashup системи; [8, 9, 17] - опис використання фасетного методу класифікації у системі підтримки прийняття рішень.

Апробація результатів дисертації. Основні результати роботи доповідалися на таких семінарах та конференціях: Міжнародна наукова

конференція «Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту» (Залізний Порт, 25-28 травня 2015); The 7, 8, 9-th International Scientific and Technical Conference «Computer Sciences and Information Technologies» (Lviv, November 20-24, 2012; Lviv, November 11-16, 2013; Lviv, November 18-22, 2014); The III International Scientific-Practical Conference «Information Control Systems and Technologies» (Odessa, September 23-25, 2014); II, III Міжнародна наукова конференція «Інформація, комунікація, суспільство 2014» (Львів-Славське, 16-19 травня 2013, 21-24 травня 2014); The 6-th International Conference of Young Scientists «Computer Science and Engineering» (Lviv, November 21-23, 2013); XI Відкрита наукова конференція ІМФН (Львів, 17-18 травня 2012, 13-14 червня 2013). Результати дисертаційних досліджень регулярно доповідалися на наукових семінарах кафедри «Інформаційні системи та мережі» НУ "Львівська політехніка" (2013-2015).

Публікації. Основні результати дисертації відображені у 17 наукових публікаціях, з них 6 статей у наукових фахових виданнях України, 1 публікація у науковому виданні України, яке входить до міжнародних наукометричних баз (Index Copernicus, Ulrich's Periodicals Directory, DRIVER, Bielefeld Academic Search Engine (BASE), РИНЦ, ResearchBib, Directory of Open Access Journals (DOAJ), WorldCat, EBSCO), 2 публікації у наукових періодичних виданнях інших держав (включені до міжнародних наукометричних баз BazTech, Cabell's Direc-tory, CNKI Scholar, ERIH PLUS, Index Copernicus, J-Gate, Google Scholar, TEMA Technik und Management) та 8 публікацій за матеріалами наукових конференцій.

Структура та обсяг дисертації. Дисертація, обсягом 164 сторінки, складається зі вступу, чотирьох розділів, висновків, списку використаної літератури та додатків. Основний текст роботи викладено на 137 сторінках, дисертація містить 29 рисунків та 13 таблиць. Бібліографічний список налічує 160 найменувань.

РОЗДІЛ 1 ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ ЗАСАДИ ДИНАМІЧНОЇ ІНТЕГРАЦІЇ СЛАБОСТРУКТУРОВАНИХ ДАНИХ У WEB-СИСТЕМАХ

1.1 Основні поняття динамічної інтеграції даних

1.1.1 Сценарії інтеграції даних

Інтеграція даних (ІД) в контексті інформаційних систем (ІС) розглядається як надання єдиного уніфікованого інтерфейсу до деякої множини незалежних неоднорідних джерел даних [17, 18]. Таким чином, для користувача інформаційні ресурси (ІР) всієї множини інтегрованих джерел представляються як новий єдиний сервіс [1]. Життєвий цикл інтеграції даних (рис. 4.1) містить наступні складові: доступність, відкритість, очищення, інтеграція, доставка, управління та аудит, моніторинг і звітність.

Рисунок 1.1 - Життєвий цикл процесу інтеграції даних

Доступність - дані надходять з багатьох джерел, в тому числі успадкованих систем, баз даних, сучасних систем, різних XML повідомлень і різних типів

документів. Ідентифікація та доступ до цих джерел є першим кроком на шляху до інтеграції даних.

Відкритість - передбачає об'єднання всіх джерел даних відкрито, і документування використання та структурування маловивчених або недостатньо описаних джерел.

Очищення – дані повинні бути очищені для точності і цілісності. Очищення може включати в себе виявлення та виправлення помилок, додаючи відсутні елементи і значення, забезпечення дотримання стандартів даних, перевірку даних і видалення повторюваних записів.

Інтеграція - цей крок включає в себе об'єднання даних всіх систем, доступ до фрагментів даних цих систем, створюючи точне і узгоджене відображення їх інформаційних ресурсів, а також використання цих активів для операцій управління бізнес-рішеннями.

Доставка - правильні, відповідні дані доступні в належній формі, своєчасно всім користувачам і системам, які потребують такого доступу. Це може означати отримання відповіді на запити, які вимагають результату у вигляді окремого запису або невеликих наборів відповідей, вхідними даними для яких є великі набори даних для звітності в масштабах підприємства.

Управління - це поєднання наборів інструментів на основі XML, які дають можливість управляти даними; бізнес-аналітиків, архітекторів, розробників і менеджерів для спільної роботи для створення набору правил інтеграції даних, процесів, методів і процедур, тим самим захоплюючи і реалізацію всіх основних робіт, виконаних в п'яти попередніх кроках.

Аудит, моніторинг і звітність - після того, як було визначено семантику і можливості використання, засоби захисту, виправлені помилки, і розглянуто питання якості потрібне постійне спостереження і аналіз, щоб зберегти дані правильно, надійно і доступно.

Система, що відповідає таким критеріям, називається системою інтеграції даних (СІД) [6, 7, 11]. Джерелами для здійснення інтегрування зазвичай є:

традиційні системи баз даних, web-сайти, репозиторії, файли структурованих даних, тощо [27, 87]. До програмних СІД відносять рішення, що надають інфраструктуру доступу і доставки даних для наступних сценаріїв інтеграції, які описано у таблиці 1.1 [44].

Таблиця 1.1 - Сценарії інтеграції даних

Сценарій	Опис сценарію
Отримання даних для роботи різних систем бізнес-аналітики	Вилучення даних з систем підтримки оперативної діяльності, трансформація і об'єднання цих даних, подання інтегрованих даних для вирішення аналітичних завдань.
Побудова інтегрованих сховищ даних	Забезпечення консолідації та раціоналізації даних, що мають важливе значення для бізнесу [88].
Міграція/перетворення даних	Виконання автоматизації переміщення і трансформації даних, затребуваних при заміні успадкованих додатків і для консолідації даних при злитті і придбанні компаній.
Синхронізація даних між системами	Забезпечення узгодженості між системами на рівні БД, включаючи як внутрішні, так і зовнішні БД.
Федеративне об'єднання даних	Забезпечення об'єднаного подання даних із сукупності різних джерел.
Сервіси даних в контексті сервіс-орієнтованої архітектури	Реалізація інтеграції даних в рамках SOA. Це не окремий сценарій, а, швидше, новий метод інтеграції, що набув активного розвитку у зв’язку із зростанням популярності сервісних архітектур і може бути застосований, в принципі, до будь-якого з перерахованих випадків [83].
Уніфікація структурованих і неструктурованих даних	Це також новий підхід до інтеграції, що відображає тенденцію до створення єдиної платформи управління, здатною охопити джерела даних довільних типів.

1.1.2 Області використання систем інтеграції даних

Основними областями використання систем інтеграції даних є [45]:

1) Інтеграція схем БД. Зважаючи на те, що схеми БД розробляються незалежно один від одної, вони часто мають різну структуру і термінологію. У підсумку виникають проблеми омонімії, синонімії, абсолютної невідповідності понять. Першим кроком на шляху до інтеграції даних є знаходження відповідностей між двома схемами даних. Після знаходження відповідностей елементи, що збігаються можуть бути об'єднані в єдину узагальнену схему (в разі консолідації) або узагальнене представлення, якщо обрана архітектура федералізованих баз даних.

2) Сховища даних (СД). Сьогодні, існує чимала кількість СД, що забезпечують базис систем підтримки прийняття рішень (СППР). Дані джерел мають принципово відмінну від сховищ структуру і тому оригінальні дані необхідно трансформувати. Для цього використовується технологія ETL (Extract, Transform, Load).

3) Простори даних. Якщо користувач зіштовхнувся із роботою у всесвітній мережі, де існує величезна кількість різних даних не має можливості визначити, які саме моделі даних використовуватимуться. В таких випадках використовують таке поняття, як простір даних (ПД). Простір даних – це сукупність БД, СД, статичних web-сторінок, засобів інтеграції, локальних сховищ та індексів, графічних матеріалів, засобів пошуку та опрацювання даних [2, 5, 8, 23, 29].

4) Електронна комерція. Електронна комерція - це набір технологій і сервісів, що надають можливість представити в Інтернеті свої товари і послуги, приймати замовлення, виставляти рахунки, а також отримувати оплату і переводити гроші контрагентам через Всесвітню мережу.

5) Семантична обробка запитів. Попередні приклади характерні тим, що вони виконуються на етапі проектування результуючої схеми даних або, принаймні, етапі її доопрацювання з причини приєднання нового джерела.

Зовсім інший сценарій спостерігається під час обробки запитів безпосередньо під час їх виконання (run-time). Користувач явно вказує очікуваний результат, який повинен бути отриманий запитом, а система повинна визначити найбільш достовірне і актуальне джерело даних для цього запиту, перетворити назви стовпців і граничні умови у адекватний для вибраного джерела формат, отримати результат і повернути його в термінах користувача.

1.1.3 Вимоги до інтеграції даних

При інтеграції інформаційних систем основоположну роль відіграє властивість інтероперабельності інформаційної системи. Інтероперабельність - це можливість побудови систем на основі довільних неоднорідних, розподілених компонентів, із уніфікованих інтерфейсів [9]. Також, під інтероперабельністю розуміється здатність ІС взаємодіяти з іншими ІС. Така взаємодія може виражатися у вигляді обміну даними, розподіленого виконання пошукових запитів, узгоджених змін БД, тощо. Необхідність забезпечення інтероперабельності виникає при зв'язуванні бізнес-процесів підприємств-партнерів, узгодженні роботи існуючої інформаційної системи з прийнятими стандартними рішеннями. Також властивість інтероперабельності використовується при інтеграції декількох інформаційних систем, доданні в створювану систему бази даних раніше використаних сховищ даних, при розробці комплексних автоматизованих систем, а також у багатьох інших випадках. Виділяється два аспекти інтероперабельності: структурний і семантичний [12].

Існуючі методи досягнення інтероперабельності головним чином стосуються її синтаксичних (структурних) аспектів, тобто спрямовані на узгодження та перетворення структур даних за рахунок стандартизації їх форматів і використання їх розширювань метамовою [19, 42, 52]. Універсальні підходи до забезпечення інтероперабельності інформаційних систем на семантичному рівні в даний час відсутні.

В останні роки в багатьох країнах в широко розгорнутих розробках електронних бібліотек (Digital Libraries) [6, 22, 75, 105] проблеми інтеграції неоднорідних даних стали грати ключову роль, причому виникає, також, задача інтеграції текстових інформаційних ресурсів з різних незалежних джерел [110, 151, 158].

Основні проблеми динамічної інтеграції даних можна поділити на чотири групи, які описано на рис. 1.2.

Рисунок 1.2 - Групи проблем динамічної інтеграції даних

1.1.4 Роль стандартів для систем інтеграції даних

Застосування стандартів відіграє важливу роль в проектуванні інформаційних систем - насамперед тому, що стандарти забезпечують можливість взаємодії різних компонент між собою [109]. Чим більш складною, розподіленою і тиражованою є система, тим визначальна й консолідуюча роль стандартів стає все більш актуальною. Стандарти є загальноприйнятими документами, що формалізують кращі практики. Строго кажучи, прийнято розрізняти стандарти де-юре, тобто розроблені і підтримувані офіційними органами стандартизації, такими як Міжнародна організація по стандартизації - ISO, і стандарти де-факто, засновані на існуючому широкому поширенні технології, методології або продукту.

У системах інтеграції даних широко використовується ряд офіційних міжнародних стандартів, а також індустріальних стандартів де-факто. Серед них:

- стандарти протоколів: RPC, ODBC, IIOP, SOAP;
- мови опису даних і інтерфейсів: HTML, XML, IDL, WSDL;
- мови опису бізнес-процесів: UML, BPEL;
- стандарти реляційних БД: SQL: 2008, SQL / MED, SQL / XML;
- об'єктні стандарти: ODMG, CORBA;
- стандарти метаданих: DC, UML, CWM;
- стандарти SOA: WSDL, UDDI, SOAP.

Основою стандарту ISO 15926 [101] «Інтеграція даних життєвого циклу установок безперервного виробництва» є модель даних, що описує все, що пов'язано з підтримкою заходів забезпечення життєвого циклу установок безперервного виробництва, і в єдиному контексті стандартизованої онтології, що відповідає за визначення значення відомостей про життєвий цикл. Пропонована стандартом онтологія підтримує всі групи описів, якими можуть володіти по відношенню до установки інженери-технологи, інженери з обладнання, інженери з технічного обслуговування та інші фахівці. На базі цієї моделі даних пропонуються механізми валідації, інтеграції, аналізу, тощо.

1.2 Сучасні методи та технології динамічної інтеграції даних

1.2.1 Основні методи інтеграції даних

В загальному виділяють такі основні методи інтеграції даних [160]:

1) Консолідація (централізація зберігання). При використанні цього підходу дані витягуються з декількох джерел і інтегруються в одне постійне місце зберігання (рис. 1.3.). Для даного підходу характерне пакетне вилучення даних з джерела, перетворення між вихідним і цільовим форматом зберігання і подальше завантаження у сховищі, що призводить до деякої затримки поновлення даних. Однак затримка в даному випадку не настільки критична, оскільки засоби аналізу і прогнозу оперують набагато більшими часовими

інтервалами. Технологія, що застосовується при такому підході, носить назву ETL (Extract-Transform-Load).

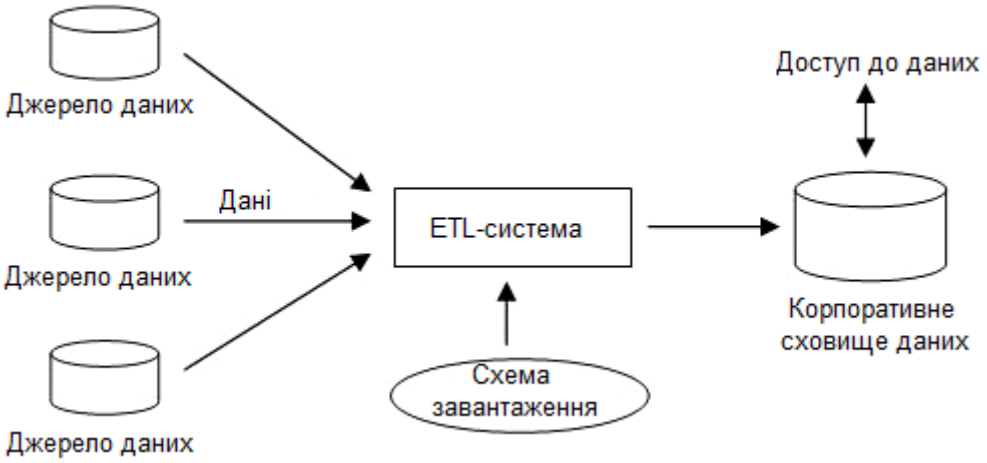


Рисунок 1.3 - Інтеграція даних на основі сховища даних

Процес інтеграції проходить в три етапи: вилучення даних з джерела (extract), перетворення між вихідним і цільовим форматом або структурою (transform) і завантаження в цільове місце зберігання (load). Найвагоміші переваги підходу - висока швидкість обробки запитів, якість збережених даних (що пройшли узгодження і очистку) простота і зручність процесу аналізу даних.

2) Федералізація (уніфікація доступу). Федеративний підхід забезпечує єдине віртуальне бачення різнорідних джерел даних. При цьому дані фактично зберігаються в різних за складом і структурою джерелах (прості файли, web-сайти, БД), інформація в яких може частково дублюватися. Джерела залишаються повністю автономними, а інтеграція даних зводиться до інтеграції схем зберігання і створення програмної компоненти (процесора федералізації), що забезпечує прозорий доступ до фізично розподілених даних. Процесор федералізації аналізує запити, що поступають від додатків, переадресовує їх відповідним джерелам, в яких ці дані можуть зберігатися, інтегрує отримані дані відповідно до віртуальної картини і повертає відповідь додатку. Між джерелом даних і процесором, як правило, міститься програмний компонент (адаптер), що приховує деталі реалізації при доступі до джерела (рис.1.4) [25].

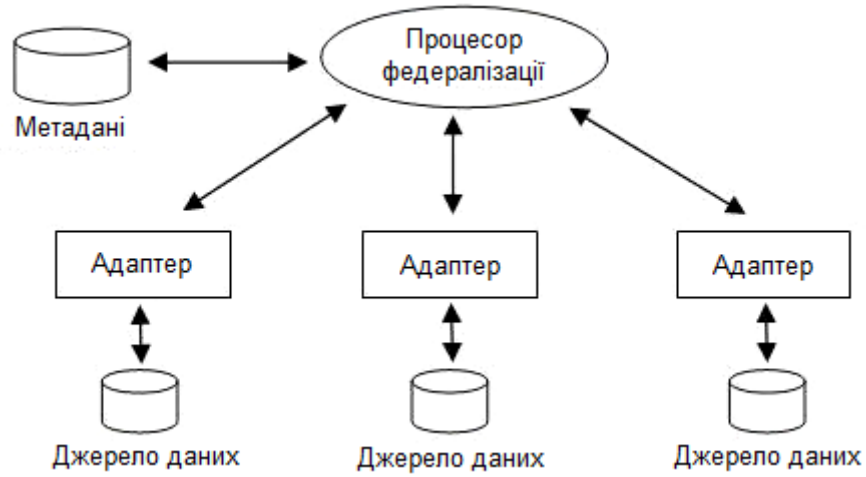


Рисунок 1.4 - Інтеграція даних на основі федералізації

Важливим елементом федеративної системи є метадані, які використовуються процесором федералізації даних для доступу до джерел інформації. У деяких випадках ці метадані можуть складатися виключно з визначень віртуальної картини, які ставляться у відповідність первинним файлам. У більш передових рішеннях метадані також можуть містити детальну інформацію про кількість даних, що знаходяться в первинних системах, а також про шляхи доступу до них.

3) Розповсюдження даних. Додатки розповсюдження даних здійснюють передачу даних від джерела до одержувача. Зміни в джерелі даних фіксуються і передаються одержувачу в оперативному режимі. Одержувач в свою чергу, отримавши повідомлення, фіксує зміни у своїй базі даних. Таким чином досягається висока оперативність в оновленні даних, фактично в режимі реального часу (рис. 1.5). Зв'язок між джерелами і одержувачами будується за допомогою посередника - як правило, це сервісна шина підприємства, яка бере на себе функції гарантованої доставки повідомлень. Джерела даних та одержувачі розробляються як сервіси, які реєструються в реєстрі сервісної шини. Прикладами технологій, що підтримують поширення даних є EAI (інтеграція корпоративних додатків) і EDR (тиражування корпоративних даних) [25].

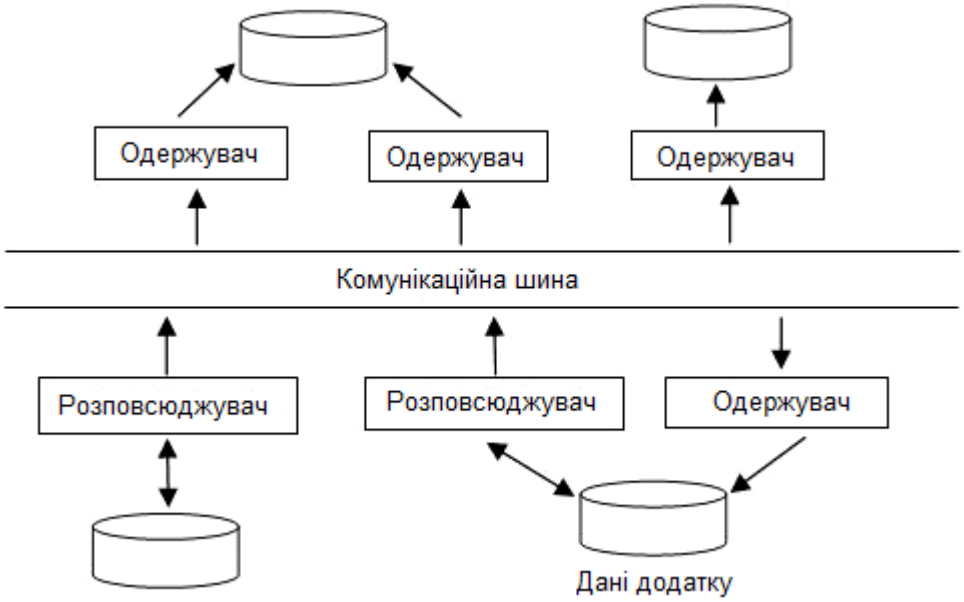


Рисунок 1.5 - Інтеграція на основі розповсюдження даних

4) Гібридна інтеграція – це поєднання можливостей трьох попередньо викладених методів [26]. Модель процесу інтеграції даних гібридним методом можна описати наступним чином:

$$GInt = \langle \{DI_i \mid i = 1,2,..N\}, Map_i(\{DI_i\}), DO \rangle, \quad (1.1)$$

де: $DI = \{di_1,..., di_N\}$ - множина вхідних даних; DO - множина вихідних даних; $Map_i(\{DI_i\})$ - відображення, вхідних множин даних у множину інтегрованих даних. Набір значень вихідної множини інтегрованих даних дістають, як результат процесу:

$$DO = DO_C \cup DO_F \cup DO_R \ , \quad (1.2)$$

де: DO_C - консолідація вхідних даних; DO_F - федералізація вхідних даних; DO_R - розповсюдження вхідних локальних даних.

1.2.2 Технології реалізації інтеграції даних

Для реалізації методів інтеграції даних потрібне застосування відповідних технологій інтеграції даних. Завдання технологій інтеграції даних полягає в подоланні численних проявів неоднорідності, властивої інформаційним

системам, які створювалися і створюються, керуючись чим завгодно, але не уніфікованим ставленням до даних [4, 26]. Дослідник з Цюріхського університету Клаус Діттріх запропонував схему класифікації технологій інтеграції даних з шести рівнів [160] (рис. 1.6):

Рисунок 1.6 - Класифікація рівнів інтеграції даних

1) Manual Integration (Інтеграція вручну). Користувач сам об'єднує дані, застосовуючи різні типи інтерфейсів і мови запитів. Користувачеві необхідно мати детальне уявлення про семантику отриманих даних і про їх структуру.

2) Common User Interface (Загальний інтерфейс користувача). Користувачеві надається єдиний користувацький інтерфейс, наприклад web-браузер, який дає доступ до гетерогенних даних. При цьому самі дані все ще залишаються неоднорідними, а їх структуризація та інтеграція повинна бути реалізована користувачем.

3) Integration by Applications (Інтеграція застосунками). Підхід заснований на використанні застосунків, які отримують доступ до різних неоднорідних джерел даних і надають користувачеві інтегрований результат [137].

4) Integration by Middleware (Інтеграція засобами ПО проміжного шару). Таке програмне забезпечення являє собою функціонал, який може бути

використаний самими додатками. Однак такий підхід не знімає повністю задачу інтеграції з додатків, а тільки робить її простішою, завдання може бути вирішеним в повному обсязі тільки при взаємодії з додатком. До всього іншого зазвичай потрібно цілий набір додаткових інструментів і проміжного ПЗ для створення загальної системи інтегрування.

5) Uniform Data Access (Уніфікований доступ до даних). На цьому рівні здійснюється логічна інтеграція даних, яка досягається на рівні доступу до даних. Різні додатки отримують однакове віртуальне бачення фізично розподілених і неоднорідних даних, самі фізичні дані при цьому не використовуються. Така віртуалізація даних має свої безперечні переваги, але структурування та уніфікація даних вимагає багато часу і значних ресурсів.

6) Common Data Storage (Загальні системи зберігання). На цьому рівні інтеграція здійснюється за рахунок злиття даних з різних систем зберігання даних в одну загальну. Такий підхід дає вигоду у швидкості роботи системи з єдиним джерелом даних, однак для перенесення даних потрібно додатковий час, а також цей підхід може викликати проблему дублювання інформації.

Співвіднесемо класифікацію технологій інтеграції даних за Клаусом Діттріхом з відомими технологіями інтеграції даних. Для реалізації нижніх рівнів, з другого по шостий, існує велика кількість різноманітних технологій, проте майже всі вони спрямовані на вирішення структурної складової задачі інтеграції, без урахування семантики даних, що інтегруються. Тим не менш саме ці технології на даний момент найчастіше використовуються інженерами в реальних проектах [93].

На сьогодні, найбільш поширеними технологіями реалізації інтеграції даних є: обмін на основі файлів; реплікація даних; технологія web-сервісів; сервіс-орієнтована архітектура (SOA); інтеграційні сервери [128].

Обмін файлами є найбільш поширеним способом інтеграції [85]. З точки зору реалізації це найпростіший спосіб, але він має недоліки. При необхідності

обміну складними структурами, необхідна розробка спеціалізованих форматів файлів, що призводить до великої залежності систем один від одного.

Реплікація - процес приведення даних електронних таблиць двох БД в ідентичний стан. Процес реплікації заснований на поняттях "видавець" і "передплатник". Видавцем є сервер публікації, тобто сервер, що відправляє дані. Передплатником є відповідно приймаючий сервер - сервер передплати.

В даний час використовується технологія Web-сервісів, яка є зручним засобом інтеграції додатків, що дозволяє легко реалізувати міжплатформенну взаємодію. Сервіси являють собою об'єкти, що знаходяться в певних відношеннях і володіють вкладеною в них функціональністю. Будь-який web-сервіс можна розглядати як чорний ящик, який має вхідний і вихідний порти.

Дуже поширеною є SOA (сервіс-орієнтована архітектура). Система, реалізована за принципами SOA, являє собою сукупність програмних компонентів, а саме сервісів, що мають стандартні інтерфейси для доступу до них за допомогою мережі та використання цих компонентів. Інтерфейси в SOA незалежні від платформ розгортання сервісів і технологій їх реалізації. SOA пропонує єдину схему взаємодії сервісів, незалежну від того, де знаходиться сервіс. В якості сервісу може бути цілий додаток, так і окремий його модуль. Передбачається, що сервіс виконує яку-небудь бізнес-дію, якусь окрему функцію [134]. SOA базується на наступних принципах:

- слабка зв'язність компонентів, яка дозволяє виконувати зміни всередині сервісу, що не зачіпає програмні модулі, що використовують його;

- «грубозерниста» (coarse-grained) структура сервісів, яка забезпечує, сервіси - модулі бізнес-логіки високого рівня, що не потребують наявності множини низькорівневих викликів, що враховують архітектуру сервісів, зниження навантаження на мережу і підвищення потужності;

- стандартні інтерфейси (standards-based) - завдяки нейтральності по відношенню до використовуваної апаратної платформи, забезпечується

універсальність взаємодії сервісів в різнорідному середовищі і зниження витрат на інтеграцію.

- Simple Object Access Protocol (SOAP) - для обміну даними. SOAP - це модель з'єднання, що забезпечує передачу повідомлення від відправника до одержувача, допускає наявність посередників, які можуть обробляти частину повідомлення або додавати до нього додаткові елементи. Багато в чому завдяки тому, що в SOAP використовується XML, він став настільки популярним засобом для вирішення завдань інтеграції, проте останнім часом все частіше замість SOAP стали використовувати RESTful web-сервіси. Від SOAP web-сервісів їх відрізняє те, що вони розробляються з урахуванням архітектурного стилю REST, запропонованого Роєм Філдінгом у своїй докторській дисертації [80]. При використанні такого стилю передбачається замість розробки нових протоколів, використовувати вже існуючі технології. Протокол http визначає методи GET / PUT / POST / DELETE, які й реалізують всі дії, необхідні в REST. Web-сервіси REST не перетворюють дані в XML або інші структури, а просто віддають їх як є.

Інтеграційні сервери являють собою проміжні сервери в інтеграційному середовищі, шлюзи, які здійснюють обробку потоків даних і повідомлень, а також розподіл даних між додатками, що мають різні інтерфейси. У ядрі інтеграційного сервера повинні зберігатися бізнес-правила, на основі яких, а також отриманих даних, виконуються обчислювальні операції, аналіз і прийняття рішень.

Сервісна архітектура динамічної інтеграції даних. Підтримка XML - перетворень, web-сервісів, зв'язку з джерелами даних з використанням таких специфікацій, як JDBC або JMS, підготували технології інтеграції даних до реалізації на базі сервісної архітектури (рис. 1.7). В ідеалі сервісна архітектура динамічної інтеграції даних повинна включати в себе три основних компоненти [21]: середовище універсального доступу до даних, репозиторій і сервіси метаданих, а також сервіси інтеграції.

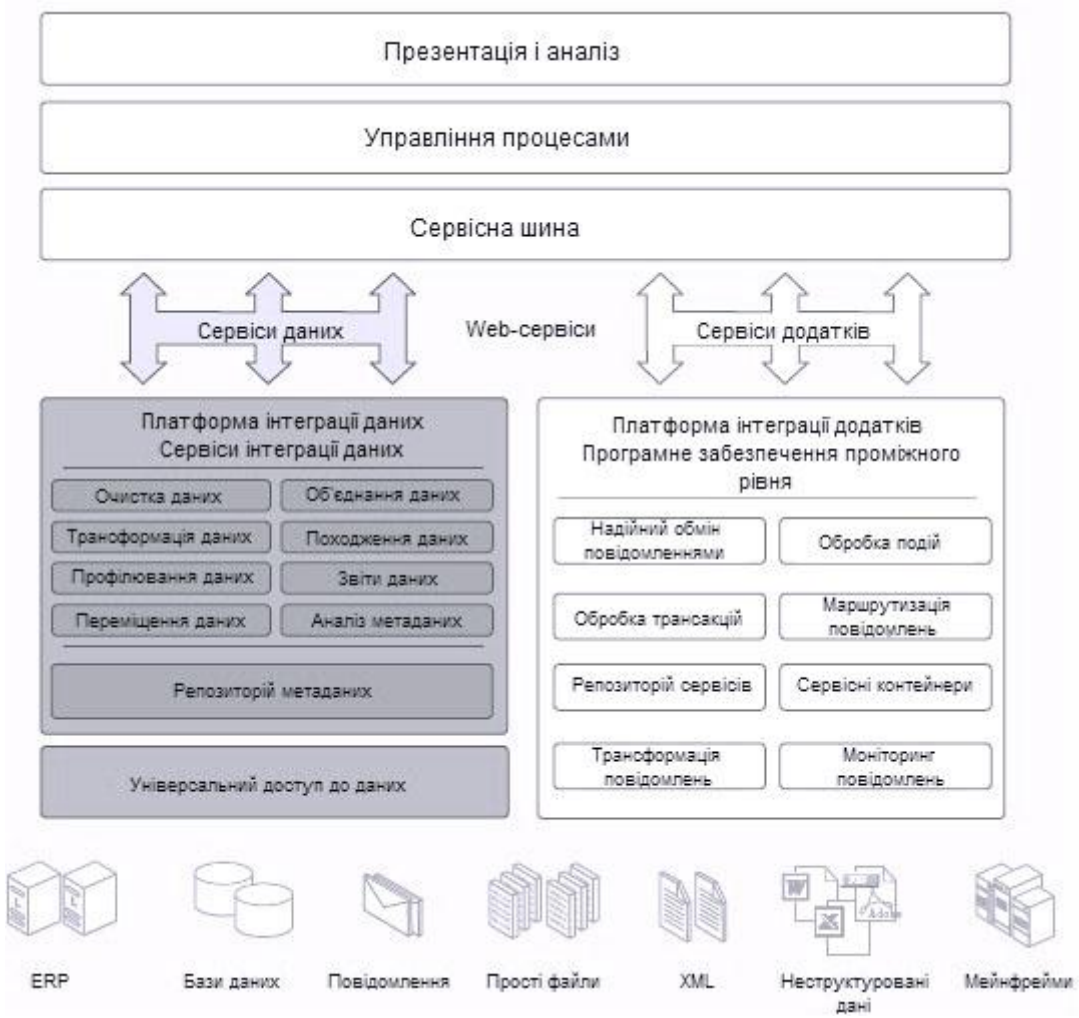


Рисунок 1.7 - Сервісна архітектура динамічної інтеграції даних

1.2.3 Використання онтологій в динамічній інтеграції даних

У контексті інформаційних систем, термін «онтологія» трактують наступним чином: формальна явна специфікація спільних концепцій із предметної області, використовуючи яку можуть взаємодіяти люди чи комп'ютерні системи [90].

Прийнято, що онтологія складається з термінів, їх визначень і атрибутів, відношень між цими термінами, а також пов'язаних з ними аксіом і правил виведення [89, 91].

Відомо таку типізацію онтологій: онтологія верхнього рівня (формальна онтологія), онтології предметних областей**,** онтології задач та онтології методів розв'язування класів задач [92].

При створенні онтології необхідно [91]:

- провести онтологічний аналіз: складається словник термінів, який включає в себе опис характеристик об'єктів і процесів, що входять в інформаційну систему, також описуються логічні взаємозв'язки між поняттями предметної області;
- виділити концепти (базові поняття);
- визначити кількість рівнів абстракції;
- розподілити концепти за рівнями;
- визначити відношення і взаємозв'язки з базовими поняттями;
- проконсультуватися з різними фахівцями для виключення протиріч і неточностей.

У дисертаційній роботі для моделювання та наочного подання онтологій використовується методологія IDEF5 [100]. У методології IDEF5 для побудови концептуальної моделі використовуються знання у вигляді сукупності понять, атрибутів і відношень. Таким чином, за допомогою методології IDEF5 можна наочно представити стан об'єктів протягом усього ходу процесу і ефективно розробляти і вивчати онтологію.

Формально онтологію даних зображають так:

$$O = \langle X, R, F \rangle, \tag{1.3}$$

де X — певна кількість понять предметної області з їх атрибутами (властивостями); R — певна кількість відношень між поняттями; F – певна кількість функцій інтерпретації понять.

Для забезпечення семантично коректного взаємозв'язку неоднорідних інформаційних систем необхідно зіставити онтології, що лежать в їх основі, і з'ясувати їх спільні і відмінні риси [40]. Це завдання вирішується за допомогою використання методів оцінки семантичної подібності концептів онтологій. В основу багатьох відомих методів знаходження міри подібності між концептами онтології покладено теоретико-множинний підхід Тверскі, заснований на

зіставленні властивостей концептів [28]. У роботі [137] розглядається структура шляхів між поняттями, а саме визначається довжина найкоротшого шляху як число концептів в ієрархії між двома розглянутими концептами в онтології. В роботі [129] подібність оцінюється як семантична відстань, вона обернено пропорційна семантичної подібності концептів. Недоліком описаних вище методів розрахунку семантичної подібності, заснованих на онтологічних структурах, є їх симетричність. Згідно з експертними оцінками міра подібності в більшості випадків не завжди симетрична.

Гібридні міри подібності, що поєднують кілька підходів, представляються найбільш перспективними. Гібридна міра, запропонована в роботі [123], містить три частини - таксономічну, реляційну і атрибутивну. Труднощі порівняння різноманітних онтологій полягають у відмінності імен понять і відношень, а також у підходах до визначення понять. При відображенні двох онтологій виконується пошук для кожного концепту однієї онтології подібного йому концепту іншої онтології.

Підходи на основі онтологій використовуються в різних областях від подання знань до інформаційної інтеграції [14, 15, 92]. Онтології застосовують в СППР з метою формального визначення модельованої частини світу у вигляді словника, що сформований фахівцями з обраної предметної області. На основі цього загального словника можна інтегрувати різні джерела знань. Таким чином, використовуючи загальний словник, можна розуміти і порівнювати різні інформаційні системи [24]. Для вирішення проблеми інтеграції даних в роботах [3, 10, 13, 155] використовувались онтології.

Для того щоб реалізувати онтологію, необхідно вибрати мову специфікації онтології, яка має достатню виразну потужність. Така мова дає можливість вказувати «машинну» семантику систем і наближає її до реального світу, що істотно підвищує можливості концептуального моделювання. Мови специфікації онтологій діляться на прості, на мови, засновані на дескриптивних логіках і на фреймах (OKBC, OCML, Flogic), а також на web-стандартах (XOL,

RDF(S), DAML, OIL, OWL, SHOE, UPML) [132, 133, 136]. Традиційні мови і web-мови специфікації онтології (Ontolingua, CycL) відрізняються виразними можливостями опису предметної області і деякими механізмами логічного висновку. Вони включають в себе конструкції для множинних ієрархій концептів, правил висновку, аксіом, а також можливість запису онтологій і відношень між ними. Існує цілий ряд інструментів (Ontolingua, OntoEdit, OilEd, WebOnto, ODE), що підтримують редагування, документування, візуалізацію, імпорт і експорт онтологій, а також об'єднання і порівняння [147].

Існуючі методики відображення онтологічних моделей не враховують їх специфіку. Застосовуючи існуючі методи, неможливо без участі експерта інтегрувати онтології, створені різними робочими групами.

1.3 Динамічна інтеграція даних на основі технології Mashup

1.3.1 Основні поняття динамічної інтеграції даних на основі технології Mashup

Одна з цілей Web 2.0 полягає в полегшенні створення, використання, опису, розповсюдження і повторного використання ресурсів в Інтернеті. Щоб добитися цього, створюються все нові і нові технології (наприклад, блоги, соціальні мережі). Можливості Web 2.0 ще більше розвиваються багатьма серверними провайдерами, які розвивають свої застосунки двома способами: перший - подання функціонування систем через web-API (наприклад, Google Map, Amazon.com, Youtube) і другий – подання через канали даних (наприклад, RSS і ATOM). Тому, відкриваються нові й захоплюючі можливості для споживачів і провайдерів, оскільки є можливість використання цих сервісів як «інгредієнтів», які можуть бути змішані і підібрані для створення нових застосунків [102].

Говорячи про динамічну інтеграцію даних у web-середовищі, то тут все більшої актуальності набувають дослідження роботи і розробки систем

інтеграції даних, що працюють використовуючи технологію Mashup [46, 50, 51, 76, 81, 97, 121, 126].

Mashup - це технологія проектування систем, яка дозволяє користувачам об'єднувати дані із певної сукупності різнорідних джерел в одне інтегроване представлення [103]. На відміну від композицій web-сервісів, де акцент робиться тільки на об’єднанні бізнес-сервісів, рамки Mashup є набагато ширшими, такі системи містять більше функціональностей і можуть інтегрувати різнорідні ресурси, такі як: сервіси передачі даних, UI-сервіси, тощо [104]. Формально опис Mashup системи динамічної інтеграції даних подають наступним чином:

$$H = \langle DI, Q, DO, C, T \rangle, \tag{1.4}$$

де: $DI = \{di_i \mid i = \overline{1, N_{DI}}\}$ - набори вхідних даних; DO - глобальний динамічний набір вихідних даних; $C = \{c_i \mid i = \overline{1, N_C}\}$ - множина умов інтеграції; $T = \{t_i \mid i = \overline{1, N_T}\}$ - час трансакцій оновлення даних; $Q = \{q_i \mid i = \overline{1, N_Q}\}$ - множина запитів користувачів; $H = \{h_i \mid i = \overline{1, N_H}\}$ - набір вихідних характеристик роботи системи.

У процесі динамічної інтеграції даних із множини вхідних даних виділяють певну підмножину значень DI^*, яку описують певною схемою S. Із даних підмножин, шляхом об’єднання, накладання різнотипних даних та створення глобальної схеми, формується єдиний глобальний динамічний набір вихідних даних DO для подальшого представлення користувачеві. Джерелами даних Mashup систем можуть бути: різноманітні web-сайти, портали, форуми, соціальні мережі, тощо [106]. Дані джерел інформаційних ресурсів Mashup систем подаються у слабоструктурованому вигляді (рис. 1.8). Слабоструктуровані дані (semistructured data) – дані, опис структури яких є частиною самих даних і має здатність змінюватися разом із їх змінами. При об'єднанні таких даних необхідно брати до уваги семантичний чинник подання інформації, а це буває досить складно, особливо у web-просторі.

Саме принципово інший підхід до ролі схеми даних є ключовою відмінністю слабоструктурованих даних від традиційних. Можна виділити наступні ключові положення, що характеризують особливості слабоструктурованих даних:

- не існує фіксованої схеми даних;
- немає чіткої відмінності між власне даними та їх схемою;
- відсутня чітка типізація;
- зміна схеми даних є рутинною операцією, яку можна порівняти з внесенням змін до даних;
- обсяг даних порівнюється зі складністю їх схеми;
- схема даних є певним описом, а не структурним розподілом, і може бути отримана з самих даних;
- абсолютне знання схеми даних не є необхідним для побудови запитів, можливі запити, що повністю ігнорують схему даних.

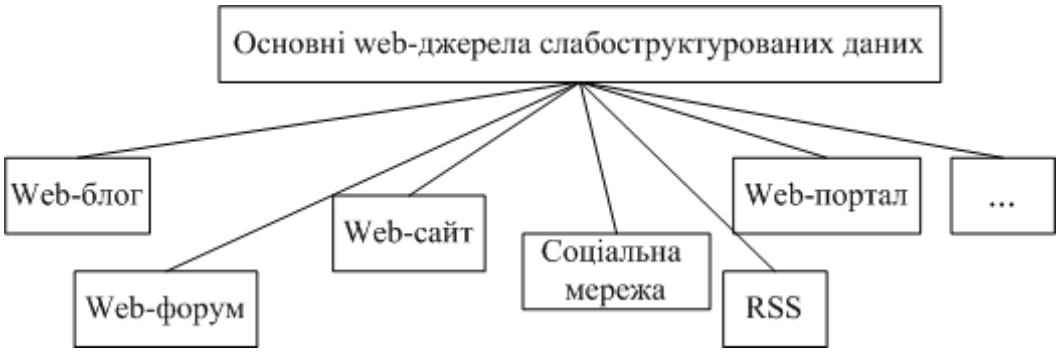


Рисунок 1.8 - Web-джерела слабоструктурованих даних

Як правило, схема Mashup інтеграції даних наступна: виконання завдання визначеного користувачем, перетворення даних до рівня сервісу та представлення готового колажу інформаційних ресурсів (рис. 1.9).

Програми, створені за допомогою технології Mashup називаються Mashups або Mashup застосунками (англ. Mashup applications) [118]. Згідно проведених досліджень існує велика зацікавленість в рамках Mashup [49, 78, 99, 154, 157]. А при стрімкому розвиткові IT-технологій особливої актуальності

набувають потреби в динамічній інтеграції різнорідних даних у web-середовищі. Технологія Mashup надає нові і широкі можливості для використання IP [107, 124, 131].

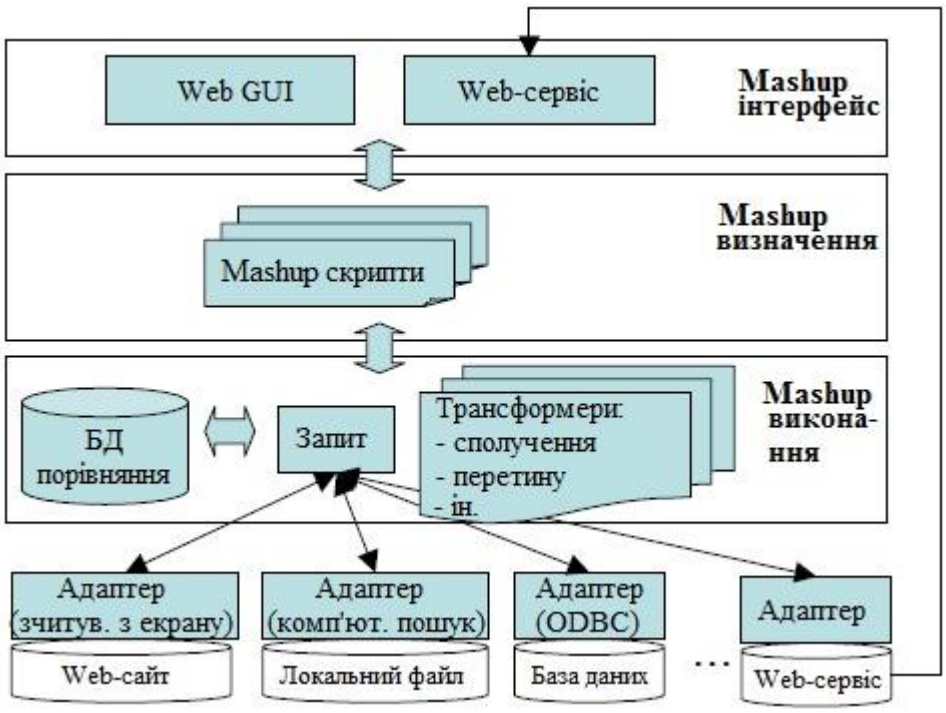


Рисунок 1.9 - Схема Mashup інтеграції даних

Архітектура web Mashup завжди складається з трьох частин [127]:

1. Провайдер вмісту - це джерело даних. Дані доступні через API та різні web-протоколи, такі як RSS, REST і web-сервіси [125].

2. Mashup сайт (рівень процесів) - це web-застосунок, що пропонує новий сервіс, який використовує джерела даних, що йому не належать [119].

3. Браузер клієнта (презентаційний рівень) - власне користувацький інтерфейс Mashup. У web-застосунках вміст може бути «замешаплено» клієнтським браузером з використанням клієнтської мови програмування, наприклад JavaScript [120, 143].

1.3.2 Mashup системи динамічної інтеграції даних

На сьогодні відомо ряд Mashup платформ, які допомагають користувачеві створювати Mashup [47, 53, 108, 159]. В роботі обрано наступні системи з трьох основних причин:

1) ці системи були найпопулярніші тоді, коли цей аналіз був виконаний (судячи з дискусій на форумах, блогах і т.д.);

2) інформація про деякі інші системи не могла бути повною через недоступність даних інструментів на певних стадіях дослідження, не дозволяючи повною мірою експериментувати із ними;

3) дана причина мотивується тим, що поставленою метою є не аналізувати всі системи, але, надати уявлення про поточний стан цих систем і зрозуміти їх загальний підхід до інтеграції даних.

IFTTT [47] ("if this, then that" - "якщо це, тоді то") - це Mashup сервіс, який дозволяє користувачам підключатися до різних web-застосунків (наприклад, Facebook, Evernote, LinkedIn, Dropbox і т.д.) за допомогою простих умовних операторів, відомих як "рецепти" і створювати просту автоматизовану послідовність операцій, яка запускається при виконанні певної дії. IFTTT дає можливість користувачам створювати і обмінюватися "рецептами", які відповідають судженню: "якщо це, тоді те", "це" - частина рецепта, яка називається тригером. IFTTT досить простий у користуванні та складається всього із трьох закладок:

- Tasks (Завдання) - це список активних завдань.
- Recipes (Рецепти) - це список найбільш популярних завдань, які можна використовувати як свої, тобто це щось типу заготовок завдань.
- Channels (Канали) - це список підтримуваних сервісів, на момент дослідження їх 54 (Twitter, LinkedIn, Gmail, Google Calendar, тощо).

Одним з гідних аналогів IFTTT можна назвати сервіс під назвою Zapier [159], який повністю успадкував схему тригер-дія і дозволяє будувати взаємозв'язки між різними web-застосунками. На відміну від IFTTT Zapier

підтримує 147 каналів, що в 2 рази більше, ніж у ifttt, а, також, його інтерфейс виглядає більш інтуїтивним і простим. Але на відміну від повністю безкоштовного IFTTT, Zapier має 4 пакети - один безкоштовний і три платних з обмеженою кількістю сервісів, задач і тригерних дій для кожного пакету.

Yahoo pipes [48] - web-інструмент, що надається Yahoo. Користувачі можуть створювати «мешапи» шляхом агрегування та обробки даних з web-каналів, web-сторінок та інших сервісів. Pipe складається з одного або декількох модулів, кожен з яких, виконує якусь певну задачу таку, як: отримання даних з web-джерела, фільтрування, сортування або об'єднання каналів. Результуючі дані системи можуть бути доступні клієнту за допомогою унікального URL як RSS або JSON, або через візуалізацію на Yahoo карті.

Google Alerts [41] - працює на основі ідеї моніторингу результатів пошукового запиту відповідно до часових змін. Фактично можна налаштувати «Алерти» на появу нових результатів за запитом. Система вміє фільтрувати результати і відбирати найбільш релевантні дані. У списку параметрів сповіщення:

- сам запит (підтримується також синтаксис пошукових запитів Google);
- тип запиту (все, новини, блоги, відео, обговорення, книги);
- частота оновлення повідомлень;
- фільтр кращих результатів або всіх;
- відправка результатів на e-mail або у вигляді RSS-фіду.

Wappwolf [53] - Mashup сервіс для роботи з файлами. Він схожий в ідеї з IFTTT, але з ухилом на обробку файлів. Єдина подія тут - додавання в папку хмарного сховища (підтримуються Dropbox, Google Drive, SkyDrive, Box) файлу, а ось дій тут може бути досить багато:

- синхронізація з іншими хмарними сховищами або з FTP-сервером;
- різноманіття простих операцій для зображень;
- операції для звукових файлів (конвертація в інший формат);

- операції для текстових файлів (конвертація в PDF, формати електронних книг, завантаження на Kindle, роздруківка через хмарний принтер Google).

- операції для всіх типів файлів (додавання в архів до архіву, перейменування, шифрування/дешифрування).

1.3.3 Технології реалізації Mashup динамічної інтеграції даних

Технологія на основі парадигми програмування [81]. Ряд інструментів, що створюють колажі даних на основі інтегрованого середовища розробки.

Технологія на основі скриптових мов. Створення колажів за цією технологією може бути занадто складно для не-програміста, тому що потрібно створити скрипти у відповідний час, а для більш складних колажів потрібно значну кількість досить складного скриптового коду.

Технологія на основі електронних таблиць. Засоби на основі електронних таблиць зосереджені на реміксі даних. На відміну від блочно-орієнтованих інструментів, тут дані безпосередньо вставлені в таблицю. Це означає, що вихідні дані для колажу із джерел даних записуються в комірках, які були попередньо обрані користувачем.

Технологія на основі блочної парадигми. Блочно-орієнтовані інструменти змішують і об'єднують дані, функціональність або презентації на основі вбудованих блоків. Це ручний зв’язок іноді називають провідниковим або трубчастим з різних модулів, з'єднувачів, компонентів або блоків. Доступні компоненти забезпечують різні функціональні можливості, і повинні бути з'єднані для досягнення бажаної координації Mashup.

Технологія на основі програмування через демонстрацію. Програмування через демонстрацію дозволяє користувачам вивчати систему через надання прикладів. Наприклад, користувач може мати можливість вказувати фрагменти з різних web-сторінок, які йому цікаві і агрегувати їх в персоналізовану Mashup сторінку. Витягнення даних відбувається на основі HTML-структури конкретної web-сторінки.

Технологія на основі автоматичного створення Mashups [148]. Автоматична генерація колажів додатків тільки недавно почала набирати інтерес в науковому співтоваристві [82]. Карлсон (Carlson) і ін. [66] пропонують методику для автоматичного компонування Mashup систем на основі Lotus Expeditor. У цій Mashup методиці, SOAPful або RESTful веб-сервіси знань об’єднуються автоматично через семантичну web-сервісну композицію на основі інтересів, завдань і досвіду користувача. Витягнуті дані об'єднуються і подаються у вигляді колажів.

Підсумовуючи аналіз сучасних технологій реалізації Mashup динамічної інтеграції даних, було виділено спільні риси опрацювання ними даних, які схематично відображено на рис. 1.10.

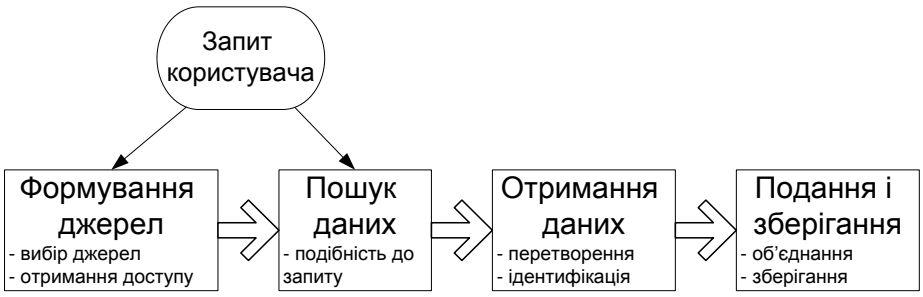


Рисунок 1.10 - Опрацювання даних існуючими методами і технологіями реалізації Mashup динамічної інтеграції даних

Існуючим технологіям для повноцінного вирішення завдання динамічного відображення вхідних наборів даних у вихідний інтегрований набір даних зі збереженням змісту даних недостатньо як масштабованості, так і функціонального охоплення. Ці підходи в основному концентруються на простому переміщенні даних з однієї системи в іншу і застосуванні трансформації та агрегації до елементів даних на рівні структур і форматів без врахування змісту даних. Вирішити цю проблему допоможе застосування методів та технологічних засобів підвищення якості пошуку та комплексного динамічного опрацювання даних із збереженням їх змісту, що знаходяться в

різних web-системах. Це обґрунтовує мету, актуальність, доцільність та напрями дослідження.

1.4 Висновки до розділу

1. Відповідно до аналізу основних вимог і стандартів до систем динамічної інтеграції слабоструктурованих даних у web-системах, при організації інтеграції даних у системі потрібно брати до уваги ряд факторів, які не завжди вдається врахувати: рівень інтеграції, який потрібно забезпечити, властивості окремих джерел даних, усю сукупність джерел в цілому, потрібні способи інтеграції.

2. Проаналізовано термінології та розглянуто сучасні методи та технології реалізації інтеграції даних для процесів проектування та моделювання систем динамічної інтеграції слабоструктурованих даних та виділено, що при створенні СДІД вагомою задачею є створення інтегруючої моделі даних, яка є основою єдиного користувацького інтерфейсу в системі інтеграції та потребують вирішення завдання розробки методів відображення моделей даних та побудови відображень в інтегруючу модель для конкретних моделей, підтримуваних окремими джерелами даних.

3. Зручність і корисність відображає перспективна в наш час Mashup технологія. Використовуючи дану технологію можна створювати системи, що об’єднують різного роду дані, розміщені у різних web-системах не затрачаючи надмірних зусиль. Та сучасні Mashup системи пропонують тільки досить прості сценарії інтеграції з використанням обмеженої кількості ресурсів.

4. Охарактеризовано процес динамічної інтеграції даних на основі технології Mashup та встановлено, що його недостатньо повно висвітлено у літературних джерелах та існує потреба у концептуалізації та розробленні методів та алгоритмів усунення неоднозначностей та достатнього обґрунтування роботи всіх функцій даного процесу.

5. Опрацювання даних в Mashup системі існуючими ІТ проходить наступним чином: згідно запиту користувача відбувається відображення вхідних даних у вихідний набір даних, застосовуючи методи простого переміщення даних із однієї системи в іншу і застосуванні агрегації до елементів даних на рівні структур і форматів, не беручи до уваги семантику даних.

6. Реалізація динамічної інтеграції слабоструктурованих даних у web-системах вимагає застосування ефективних методів та засобів опрацювання семантики інтегрованих інформаційних ресурсів на всіх етапах роботи із ними у Mashup системі. Сучасні ІТ реалізації Mashup систем не вирішують даного завдання. Тому існує потреба у розробленні ІТ, що давала б змогу врахувати зміст інтегрованих даних на всіх етапах роботи даної системи.

РОЗДІЛ 2 ВИЗНАЧЕННЯ ЗАГАЛЬНОЇ ДИНАМІЧНОЇ СТРУКТУРИ ДЛЯ МНОЖИНИ ВХІДНИХ ІНФОРМАЦІЙНИХ РЕСУРСІВ

2.1 Принципи роботи web-систем динамічної інтеграції слабоструктурованих даних

2.1.1 Формати і доступ до даних систем динамічної інтеграції даних

Користуючись Mashup застосунком, користувач може об'єднувати дані, описані в різних форматах. Наприклад, web-формат використовується для публікації часто оновлюваної інформації, наприклад, записів у блогах, сайтах новин і так далі; табличний формат підходить для опису моделей даних на основі таблиць, таких як CSV-файли або електронні таблиці; формат на основі розмітки (наприклад, HTML і XML), зазвичай, часто використовується для публікації даних; мультимедійний контент, такий як відео, аудіо та зображення, стають все більш поширеними [98]. Ці типи даних можуть бути доступні користувачеві з різних джерел даних. Найбільш поширеними джерелами даних можуть бути традиційні системи керування базами даних, локальні файли, які доступні у файловій системі власника, web-сторінки, web-сервіси і web-застосунки [146]. Для полегшення вилучення web-даних, провайдери часто дістають їх зміст через web-API [145]. Розглядаючи роль API, з точки інтеграції даних, вони пропонують конкретні типи і формати даних. Слід зазначити, що API може запропонувати кілька форматів даних, наприклад, CSV, XML і т.д.

API сприяють інтеграції між кількома системами, дозволяючи вилучення даних і обмін даними між системами. API допомагають отримати доступ і використовувати ресурси, не зосереджуючись на своїй внутрішній організації [149]. Прості й відомі приклади API включають ODBC і JDBC. Інтернет-провайдери такі як, як Microsoft, Google, eBay і Yahoo дозволяють отримати

доступ до даних, надаючи web-API, які, як правило доступні через стандартні протоколи, такі як REST / SOAP web-сервіси, Ajax або XML віддалений виклик процедур.

API, також, можуть бути використані для доступу до ресурсів, які не є URL адресацією, таких як приватні або корпоративні дані. Тим не менш, деякі загальні джерела даних не надають свій вміст через API. Тому потрібно використовувати якісь інші методи для того, щоб отримати дані.

2.1.2 Внутрішня модель даних систем динамічної інтеграції даних

Мета Mashup системи - об'єднання різних ресурсів (даних в нашому випадку) для створення нової системи. Ці ресурси приходять зазвичай з різних джерел, в різних форматах і транспортують різну семантику. Для підтримки своєї роботи, кожен Mashup інструмент використовує свою внутрішню модель даних.

Внутрішня модель даних є єдиною глобальною схемою, що являє собою уніфіковане представлення даних. Внутрішня модель даних Mashup системи може бути або у вигляді графу або на основі об'єктів.

Графічно-орієнтована модель - на основі графу будується модель, заснована на мові XML і всьому тому, що має безпосередній зв’язок з XML. Тобто модель може містити чистий XML, RDF, RSS і т.д. Більшість Mashup застосунків використовують модель у вигляді графу як внутрішню модель даних. Це, звичайно, мотивується тим, що більшість сьогоднішніх даних, в основному в Інтернеті, таких як RSS-канали, є доступні для цього формату моделі даних. Тобто, всі дані, які використовуються Mashup інструментами, в цій категорії, трансформують вхідні дані до вигляду XML перед їх обробкою.

Об'єктно-орієнтована модель - в цьому випадку внутрішня модель даних формується у вигляді об'єктів (у класичному сенсі об'єктно-орієнтоване програмування). Об'єкт є екземпляром класу, який визначає особливості елемента, в тому числі характеристик елемента (його атрибутів, полів або

властивостей) і поведінки елемента (методи). Слід зазначити, що в даному випадку немає ніякого явного перетворення, виконаного системою, як у випадку графічно-орієнтованої моделі, але програміст повинен визначити структуру об'єкту відповідно до її даних [135].

2.1.3 Робота з вхідним та вихідним потоком даних у системі динамічної інтеграції даних

Зіставлення даних. Щоб створити екземпляр внутрішньої моделі даних із зовнішнього джерела даних, Mashup застосунок мусить забезпечити стратегію по попередньо заданих відповідностях між внутрішньою моделлю даних і бажаними джерелами даних. Це досягається шляхом відображення даних. Відображення даних - це процес, що необхідний для визначення відповідності між елементами моделі даних джерела і внутрішньою моделлю даних [138]. Взагалі кажучи, відображення даних може бути:

- ручне, де всі відповідності між внутрішньою моделлю даних і моделлю даних джерела даних вручну вказано, одну за іншою, розробником програми. У цьому випадку система повинна потім надати деякі засоби для користувача, щоб розробити трансформацію.

- напівавтоматичне, коли система використовує метадані (наприклад, поля імена і типи) для пропозицій певних можливих конфігурацій відображення. Звичайно, користувач повинен підтвердити ці пропозиції, і зазвичай виправити деякі з них.

- автоматичне, де всі відповідності між двома моделями даних автоматично генеруються, без втручання користувача [138]. Це складна проблема в області інтеграції даних. Оскільки площа Mashup знаходиться в «початковій стадії», цей тип відображення не підтримується будь-яким інструментом Mashup. Слід зазначити, що процес відображення даних може зажадати проміжний крок, тобто крок загортання, для перетворення початкового формату у внутрішній формат, наприклад, від CSV для XML.

Цікаво також відзначити, що відображення в наявних в даний час засобах Mashup робиться тільки на рівні схеми даних, а семантичний аспект не розглядається й досі.

Оператори потоків даних. Оператори потоків даних дозволяють виконувати операції або на структурі даних, або на самих даних (змісті) (за аналогією до мови обробки даних/операторів в реляційній моделі). Розглянемо оператори і вирази мов програмування, що надаються інструментами для обробки та інтеграції даних. Оператори потоку даних дозволяють:

- реструктурувати схему вхідних даних, наприклад, додавати нові елементи, додавати нові атрибути до елементів;

- виконувати перетворення на наборах даних, таких як: витягнення певної частини інформації, поєднання конкретних елементів, які задовольняють задану умову, зміна значень деяких елементів;

- будувати новий набір на основі інших наборів даних, використовуючи операції перетину, об’єднання або різниці над даними.

Реалізація операторів потоку даних сильно залежить від головної мети інструменту, тобто інтеграції чи візуалізації. Деякі оператори, наприклад, Union (об’єднання), реалізуються в різних системах по-різному і надана інтерпретація результату також відрізняється. Основні оператори, орієнтовані на інтеграцію даних, реалізовані в декількох наступних операціях: Union (об’єднання), Join (з’єднання), Filter (фільтрування) і Sort (сортування) (таблиця 2.1).

Докладніший опис наведених операторів розглянуто в [152].

Оновлення даних. У деяких випадках, наприклад фондовий ринок, дані, зазвичай, динамічно генеруються і постійно оновлюються. Багато стратегічних рішень, особливо на підприємствах, як правило, приймаються у відповідності з останнім статусом/значенням даних.

Таблиця 2.1 - Основні оператори Mashup систем

Оператор	Опис
Union	Об'єднує два набори даних в один. Результуючий набір містить всі дані з усіх наборів, що беруть участь в об’єднанні.
Join	З’єднує різні набори даних відповідно до умови.
Filter	Вибирає конкретну підмножину (сутностей і атрибутів) з вихідної підмножини.
Sort	Представляє вибрані дані в певному порядку.

Саме тоді важливо, щоб система поширювала оновлені джерела даних для відповідного користувача(ів). Існують дві основні стратегії, що стосуються статусу даних у джерелі, в залежності від мети користувача:

1. Стратегія витягнення даних (pull strategy) - ця стратегія заснована на формуванні частих і повторюваних запитів від клієнта, акцент робиться на періодичності і частоті витягнення даних. Періодичність і частота витягнення даних, як правило, обираються, якомога нижчі, ніж середня частота оновлення даних в джерелі. Новизна даних залежить головним чином від періодичності і частоти витягнення даних, тобто чим вищою є періодичність і частота витягнення даних, тим новішими будуть дані і навпаки [142].

2. Стратегія вміщення даних (push strategy) – при даній стратегії клієнт не посилає запити, але повинен зареєструватися на сервері. Реєстрація необхідна, щоб вказати/ідентифікувати дані, що представляють інтерес. Отже, сервер посилає дані клієнту, коли відбувається зміна на стороні сервера. Основним недоліком цієї моделі є те, що клієнт може бути зайнятий виконанням інших завдань, коли відправляється інформація, і тому існує можливість затримки в момент обробки інформації.

Ще одним важливим параметром для оновлення даних є те, як система управляє інтервалом витягнення даних. Ми можемо визначити дві можливі стратегії для вирішення даної проблеми: глобальна стратегія і локальна стратегія.

Для глобальної стратегії, інтервал витягнення даних встановлюється для всієї програми. Це передбачає, що джерела даних мають такий ж самий інтервал оновлення даних. Тобто, до джерел даних посилається запит в той же проміжок часу, що відповідає одному з інструментів Mashup-системи. В результаті, користувач підтримує кращий зв’язок з деякими джерелами (ті, що мають низький інтервал оновлення в порівнянні з іншими), ніж з іншими (ті, що має високий інтервал оновлення в порівнянні з іншими).

У локальній стратегії, для кожного джерела даних є свій власний інтервал оновлення. Цей інтервал оновлення, як передбачається, відповідає одному з джерел даних. В результаті кращий зв’язок потрібно зберігати із кожним джерелом даних. На те, щоб встановити інтервал оновлення, кожна компонента джерела має параметр «інтервал оновлення». Після перевищення часу, дані з вказаного URL перезавантажуються.

Вихідні дані Mashup систем. Розглянемо вихідні дані, з точки зору вимірювання їх розміру, так як користувач може бути зацікавлений в експорті даних Mashup результату (потоку даних) до приведення їх до іншого формату, щоб використовувати їх для подальшої обробки, а не просто візуалізації. Тобто, ми можемо виділити дві основні категорії вихідних даних: людино-орієнтовані вихідні дані і вихідні дані, орієнтовані на обробку.

Людино-орієнтовані вихідні дані призначені для людської інтерпретації, наприклад візуалізація на карті, на HTML сторінці і т.д. Тобто, для даної категорії, вихідні дані можна розглядати як «кінцевий продукт» всього процесу.

Вихідні дані, орієнтовані на обробку - в основному орієнтовані на використання машинами. Це потрібно у випадку, коли необхідно використовувати дані, що підлягають подальшій обробці, наприклад, для

вилучення знань. Слід зазначити, що ця категорія може, на деякій стадії, включати першу категорію, наприклад, вихідні дані у вигляді RSS можуть бути як візуалізовані на HTML сторінці, так і використані в інших додатках для інших задач обробки даних.

Представлення вихідних даних залежить від головної мети системи. Фактично, більшість систем забезпечують багаті динамічні візуалізації колажів додатків. Але є й такі, які прагнуть до агрегації і маніпулювання даними, які можна використовувати з іншими додатками. Вихідні дані експортується в RSS, Atom або XML інтерпретацію шляхом додавання інформації заголовка і вмісту в ньому конкретної інформації, потім інформація перетворюється в кортежі послідовностей в зазначений тип вихідної подачі інформації.

2.1.4 Функціональні можливості систем динамічної інтеграції даних

Розширюваність. Розширюваність визначає здатність системи підтримувати додаткові, зазвичай, визначені користувачем, функціональні можливості. Може бути два можливі способи визначити і використовувати ці функціональні можливості. Функціональність може бути або вбудована всередині інструменту, тобто відповідний код цієї функціональності додається до інструменту з використанням конкретної мови програмування, або зовнішньою, тобто викликати відповідну послугу, яка містить таку функцію.

Розповсюдження. Mashup додатки засновані на технології Web 2.0, де люди можуть напрочуд легко створювати, коментувати і обмінюватися інформацією [144]. Питання безпеки та конфіденційності для обміну в цьому величезному мережевому середовищі, безумовно, є великою проблемою. Тому визначаючи модальність, яку система пропонує для включення спільного використання ресурсів, вона не може гарантувати конфіденційність і безпеку створених Mashup додатків. Це досить складна область у створенні Mashup систем і багато роботи ще належить виконати.

Представлення ресурсу може бути різним, наприклад: тільки для читання (користувач може читати всі дані, але не може їх змінювати чи дописувати щось), для читання/запису (користувач може читати і змінювати дані), без доступу (користувач не може читати чи змінювати дані). Користувачами, які можуть брати участь в процесі можуть бути, наприклад, будь-хто, певна група людей або конкретний користувач [73]. Слід зазначити, що для кожного користувача може бути різна політика доступу до розповсюдження інформації:

- повний доступ, тобто доступ читання вихідного коду, даних і виведених результатів;
- частковий доступ, тобто доступ на читання вихідного коду;
- без доступу, де Mashup дані не розповсюджуються.

2.1.5 Аналіз характеристик сучасних систем динамічної інтеграції даних на основі технології Mashup

Було проаналізовано всі описані вище характеристики сучасних систем динамічної інтеграції даних на основі технології Mashup та сформовано і подано їх опис у таблиці 2.2 та 2.3.

Таблиця 2.2 -Аналіз загальних характеристик сучасних систем динамічної інтеграції даних на основі технології Mashup

Назва характеристики		**IFTTT**	**Zapier**	**Yahoo pipes**	**Google Alerts**	**Wapp-wolf**
Загальні характеристики						
Формат даних	XML	+	+	+	+	-
	RSS	+	+	+	+	-
	ATOM	+	+	+	-	-
	JSON	+	+	+	-	-
	HTML	+	+	+	-	+
	CSV	-	-	+	-	-
	XLS	-	-	-	-	-
	RDF	-	-	+	-	-
	Text	-	-	-	-	+
	Image	+	+	+	+	+
	Video	+	-	-	+	-

Назва характеристики		IFTTT	Zapier	Yahoo pipes	Google Alerts	Wapp-wolf
Доступ до даних	HTTP	-	-	-	-	+
	REST	+	+	+	+	-
	SOAP	-	-	-	+	-
Внутрішня модель даних	Графічно-орієнтована	+	+	+	+	+
	Об'єктно-орієнтована	-	-	-	-	-
Функціональні можливості						
Розширюваність	Компоненти	+	+	+	+	+
	Дані	-	+	-	+	-
Поширення даних	Повне	+	+	+	+	+
	Часткове	-	-	+	-	-
	Відсутнє	+	+	+	+	+
	Тільки для читання	+	+	+	+	+
	Читання/запис	-	-	-	-	-

Таблиця 2.3 - Аналіз характеристик роботи із вхідними і вихідними даними сучасних Mashup систем

Назва характеристики		IFTTT	Zapier	Yahoo pipes	Google Alerts	Wapp-wolf
Робота із вхідним та вихідним потоком даних						
Зіставлення даних	Ручне	+	+	-	-	-
	Напівавтоматичне	-	-	+	+	+
	Автоматичне	-	-	-	+	-
Опрацювання семантики	Присутнє	-	-	-	+-	-
	Відсутнє	+	+	+	+-	+
Оновлення даних	Стратегія витягнення даних	+	+	+	+	+
	Стратегія вміщення даних	-	-	-	-	-
	Глобальний інтервал витягнення даних	+	+	+	+	-
	Локальний інтервал витягнення даних	-	-	-	-	-
	Інтервал оновлення	+	+	-	+	-
Вихідні дані	Людино-орієнтовані	+	+	+	+	+
	Машино-орієнтовані	-	-	+	-	+

2.2 Принципи визначення вхідних даних та формування вихідних даних у Mashup системі

2.2.1 Концептуалізація та моделювання роботи Mashup системи

Нехай процес опрацювання даних з (1.4) описано оператором:

$$do_j(t_{p+1}) = Mashup(di_i, q_d, c_k, t_p). \quad (2.1)$$

Для того, щоб зрозуміти основну суть роботи Mashup і як саме дана система працює з вхідним та вихідним потоком даних було використано STD діаграми переходів станів (State Transition Diagram) [74] (рис. 2.1). Діаграми переходів станів є інструментами для представлення пріоритетів взаємодій між процесами і станами системи, також, при необхідності, діаграми переходів станів легко перетворюються в об'єктно-орієнтований код.

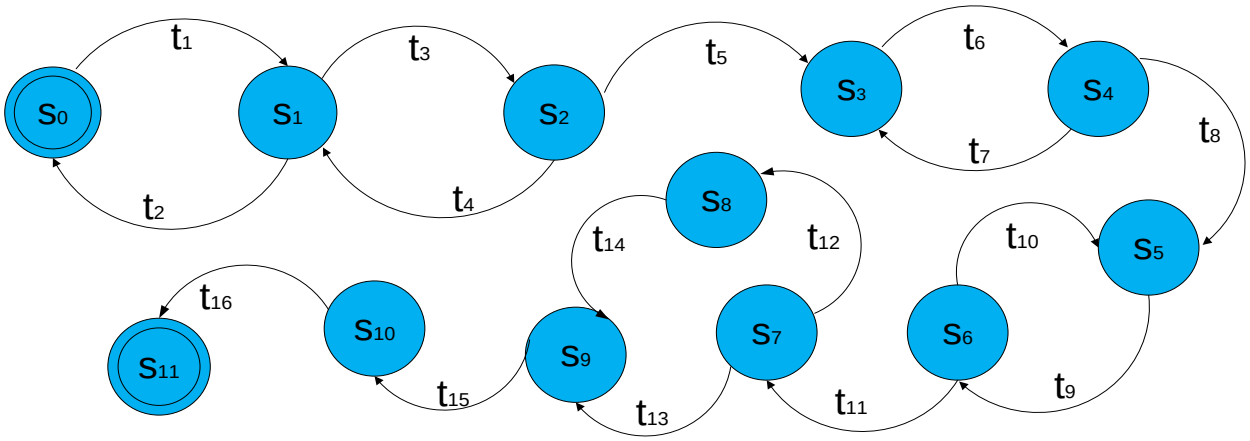


Рисунок 2.1 - Діаграма переходів станів Mashup системи

На діаграмі переходів станів, зображеній на рисунку 2.1 є наступні стани:

s_0 - стан початку (Start);

s_1 - стан, що відповідає за реєстрацію в системі (Registration);

s_2 - стан авторизації в системі (Authorization);

s_3 - стан очікування на формування завдання (Wait for a task);

s_4 - стан отримання завдання (Task receiving);

s_5 - стан очікування на формування джерел для Mashup (Wait for a source forming);

s_6 - стан формування джерел для Mashup системи (Sources forming for Mashup system);

s_7 - стан пошуку даних (Data searching);

s_8 - стан редагування пошукових критеріїв (Search with editing);

s_9 - стан витягнення відповідних запиту даних (Data extraction);

s_{10} - стан зберігання інформації у вигляді сервісу (Storing as a service);

s_{11} - стан візуального представлення готового Mashup (Visual presentation);

На даній діаграмі (рис. 2.1) зображено такі переходи станів:

t_1 - перехід, що відповідає за введення даних користувачем (User data input);

t_2 - перехід, що відбувається, якщо введені дані були не коректними (User data is incorrect);

t_3 - перехід, що відбувається, якщо введені дані були коректними (User data is correct);

t_4 - перехід, який відбувається у випадку не коректної авторизації (Authorization is not good);

t_5 - перехід, який відбувається у випадку коректної авторизації (Authorization is good);

t_6 - перехід, що відповідає за формування завдання (Forming a task);

t_7 - перехід, що відбувається, якщо завдання було сформовано не коректно (Task forming is incorrect);

t_8 - перехід, що відбувається, якщо завдання було сформовано коректно (Task forming is correct);

t_9 - перехід, що відповідає за вибір джерел для формування Mashup (Choice a sources for Mashup system);

t_{10} - перехід, що відбувається у випадку помилки при виборі джерел (Choice is not good);

t_{11} - перехід, що відбувається у випадку, коли джерела сформовано коректно (Source forming is good);

t_{12} - перехід, що відбувається, якщо потрібно відредагувати критерії для формування результату (Data searching need editing);

t_{13} - перехід, що відбувається, якщо пошук інформації відбувся вдало (Data searching is good);

t_{14} - перехід, що відбувається, якщо пошук інформації з редагуванням відбувся вдало (Data searching with editing is good);

t_{15} - перехід, що відбувається при коректному отриманні даних з обраних джерел (Data extraction is good);

t_{16} - перехід, що відбувається при зберіганні даних у вигляді сервісу (Data is stored as a service).

Для зручнішого та кращого аналізу роботи Mashup системи було інтерпретовано наведену вище діаграму переходів станів у модель діяльності системи використовуючи об’єктно-орієнтовану технологію моделювання роботи системи, таку як UML-діаграма діяльності [61, 111, 139].

Аналізуючи діяльність Mashup систем можна виділити наступні стани діяльності (рис. 2.2):

Перший стан – реєстрація. Якщо реєстрація пройшла успішно, переходимо до другого стану, якщо ж ні – повертаємося знову до початку реєстрації.

Другий стан – авторизація, якщо все пройшло успішно, рухаємося далі, якщо ні – повертаємося до початку авторизації.

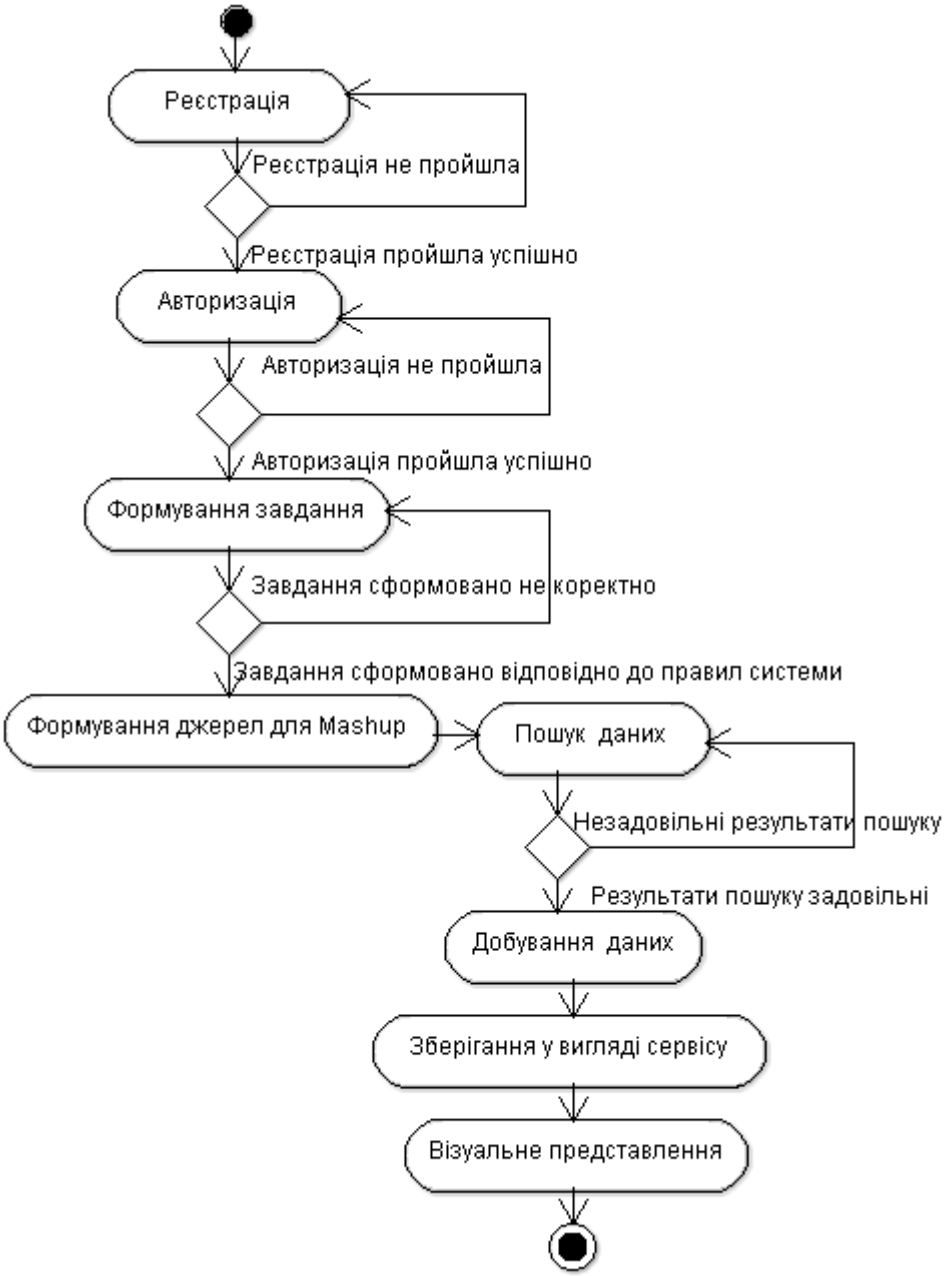


Рисунок 2.2 - UML-діаграма діяльності Mashup системи

Третій стан – формування завдання. Якщо завдання сформовано відповідно до правил системи, рухаємося далі, якщо ні – повертаємося до початку третього стану.

Четвертий стан – формування джерел для Mashup.

П’ятий стан – пошук даних у виділених джерелах. Якщо результати пошуку задовільні – йдемо далі, якщо ні – повертаємося назад до пошуку.

Шостий стан – добування даних і перехід до наступного стану.

Сьомий стан – зберігання даних у вигляді сервісу.

Восьмий стан – візуальне представлення готового Mashup.

Отже, було визначено 8 основних станів, які, зазвичай, наявні у будь-якій Mashup системі. Найважливішими станами при роботі Mashup системи є пошук даних (п'ятий стан), добування даних (шостий стан) та їх зберігання у вигляді сервісу (сьомий стан). При роботі з інформаційними ресурсами на цих станах досить часто може виникнути ряд проблем, які можуть вплинути не найкращим чином на результати роботи системи. Від коректної роботи системи на даних трьох станах залежить безпосередньо кінцевий результат системи. Стани пошуку та добування даних належать до процесу роботи системи із вхідними даними, а стан зберігання даних у вигляді сервісу – процесу відображення вихідних даних. Звідси, отримуємо, що процес відображення вхідних наборів у вихідний набір даних складається із: процесу визначення вхідних даних та процесу формування вихідних даних (2.1), який подано, як суперпозиція функцій

$$Mashup = \lambda \circ \beta \,, \tag{2.2}$$

де: λ - оператор визначення вхідних даних; β - оператор формування вихідних даних.

У дисертаційній роботі пропонується (рис. 2.3):

- розроблення методу опису структури і змісту вхідних даних для підвищення якості результату процесу визначення вхідних даних;

- розроблення методу формування об'єднаного динамічного набору даних, який має загальну структуру і єдиний зміст для підвищення якості результату процесу формування вихідних даних.

Mashup систему, враховуючи (2.2), подано як

$$H^{'} = \langle DI, Q, DO, C, T, \lambda, \beta \rangle . \tag{2.3}$$

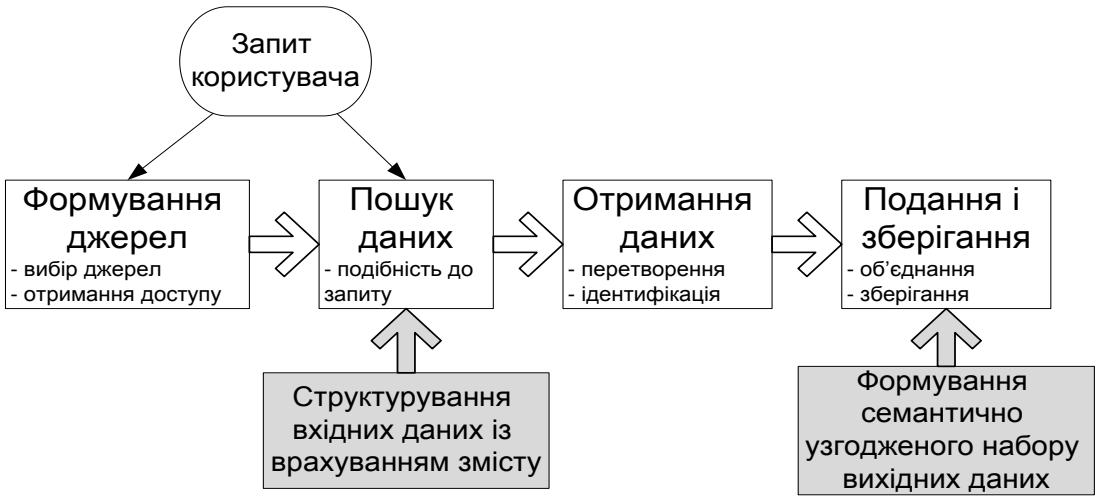


Рисунок 2.3 - Схема застосування методів для підвищення якості результатів опрацювання даних у Mashup системі

В основному джерелами Mashup застосунків є web-системи із даними, поданими у слабоструктурованому вигляді, адже сьогодні панівне становище в якості засобу представлення інформації в Інтернет займає мова гіпертекстової розмітки HTML, теги якої описують візуальне представлення документу, посилання, тощо, але не несуть інформації відносно семантичної структури документу. З розвитком Інтернет та збільшенням обсягів збережених даних необхідність автоматизованого пошуку семантично релевантної інформації стає все більш актуальною. Mashup застосунки, здатні інтегрувати дані із декількох різних джерел є досить корисними для вирішення цього завдання. Але документи HTML мало придатні як засіб подання і отримання семантично узгодженої інформації для таких систем, так як описана засобами HTML інформація може бути охарактеризована більш як для машинного читання, але не як машинного розуміння. Тому надзвичайно актуальною є проблема використання ефективних методів та ІТ забезпечення можливості подання семантичної структури документів в Інтернет, які дозволяли б опрацьовувати користувацький запит, враховуючи зміст інтегруючих даних.

Оператор визначення вхідних даних λ є відображенням слабоструктурованих вхідних даних di_i у структурований вигляд із збереженим змістом даних.

$$\lambda : (di_i, t_p, c_S) \to (di_{i+1}, t_{p+1}), \tag{2.4}$$

де: $C_S = \{c_{iS} \mid i = \overline{1, N_{C_S}}\}$, $C_S \subseteq C$ - множина умов визначення вхідних даних.

Нехай $di_i^* = di_{i+1}$ - структуровані дані, згідно дії λ на di_i. І їх подано за допомогою (2.4).

$$di_i^* = \left\{ \bigcup_{i=1}^{N_{di}} di_i \mid \begin{array}{l} \forall di_i \in DI_{c_S}, di_i \notin DI_{\overline{c_S}}, \exists c_S \in C_{di_i}, c_S \notin C_{\overline{di_i}}, \\ DI = DI_{c_S} \cup DI_{\overline{c_S}}, C_S = C_{di_i} \cup C_{\overline{di_i}}, s = \overline{1, N_c} \end{array} \right\}. \tag{2.5}$$

Де множина умов визначення вхідних даних подана як

$$c_S = \left\{ \bigcup_{j=1}^{K} c_{iS} \mid \begin{array}{l} \forall c_{iS} \in C_{di_i}, \exists di_i \in DI_{c_S}, c_{iS} \notin C_{\overline{di_i}}, \\ C_S = C_{di_i} \cup C_{\overline{di_i}}, DI_{c_S} \subseteq DI, s = \overline{1, N_c}, i = \overline{1, M} \end{array} \right\}. \tag{2.6}$$

Оператор формування вихідних даних β є відображенням структурованих вхідних даних di_i^* у набір вихідних даних із узгодженим змістом даних do_j.

$$\beta : (di_i^*, q_d, t_p, c_u) \to (do_j, t_{p+1}), \tag{2.7}$$

де: $C_u = \{c_{iu} \mid i = \overline{1, N_{C_u}}\}$, $C_u \subseteq C$ - множина умов формування вихідних даних; $q_d = \{q_{id} \mid i = \overline{1, N_{Q_d}}\}$ - множина запитів користувачів.

Формування контенту динамічного набору даних подано як

$$do_j = \left\{ \bigcup_{i=1}^{N_{di^*}} di_i^* \mid \begin{array}{l} \forall di_i^* \in DI_{q_d}, \exists q_d \in Q, \exists c_u \in C_{di_i^*}, di_i^* \notin DI_{\overline{q_d}}, c_u \notin C_{\overline{di_i^*}}, \\ DI = DI_{q_d} \cup DI_{\overline{q_d}}, C_u = C_{di_i^*} \cup C_{\overline{di_i^*}}, d = \overline{1, N_Q}, s = \overline{1, N_C} \end{array} \right\}. \tag{2.8}$$

Умови формування вихідних даних подано як

$$c_u = \left\{ \bigcup_{j=1}^{K} c_{iu} \mid \begin{array}{l} \forall c_{iu} \in C_{di_i^*}, \exists di_i^* \in DI_{c_u}, c_{iu} \notin C_{\overline{di_i^*}}, \\ C_u = C_{di_i^*} \cup C_{\overline{di_i^*}}, DI_{c_u} \subseteq DI, u = \overline{1, N_u}, i = \overline{1, L} \end{array} \right\}. \quad (2.9)$$

Розглядаючи четвертий стан має місце вирішення проблеми ідентифікації інформаційних ресурсів або деякого інформаційного об'єкту (далі - об'єкту), яка може включати в себе кілька етапів, серед яких можна виділити основні:

- завдання умови виділення деякого об'єкту серед всієї сукупності об'єктів - визначення пошукового образу;

- процес розпізнавання об'єкту - перевірка задоволення умовами пошукового образу характеристик об'єкту сканування.

Основою роботи Mashup сервісу є комбінація інформації з різноманітних джерел даних, як результат на запит користувача. Щоб ця комбінація була максимально вдалою потрібно застосувати правильну стратегію пошуку потрібної інформації у відібраних джерелах. Проаналізувавши всі плюси і мінуси роботи Mashup сервісу було виділено чотири конкретні стратегії пошуку:

1) паралельна – дана стратегія пошуку виконує всі запити всіх доступних генераторів запиту.

2) послідовна - ця стратегія виконує запити за фіксованим порядком запиту генераторів.

3) оптимістична - ця стратегія пошуку виконує запити відповідно з кількістю охоплених осіб і тим самим віддає перевагу запитам з великим охопленням, порівняно з іншими запитами.

4) попередньої оцінки - ця стратегія виконує найперспективніші запити, засновані на продуктивності (тобто, ефективність і дієвість) раніше виконаних запитів одного і того ж генератора запиту. Підхід заснований на попередній оцінці результатів пошуку для всіх запитів генераторів на загальному наборі підготовки вхідних даних.

Усі пошукові стратегії використовують функцію оцінки запиту і функцію вибору запиту. Функція оцінки присвоює певну оцінку кожному запиту. Всі запити ранжуються за їх оцінками (у порядку зростання) і функція вибору потім обробляє і фільтрує запити відповідно до їх рейтингу.

2.2.2 Метод опису структури і змісту вхідних даних

Процес опису вхідних даних забезпечує динамічне отримання структурованих даних із збереженим змістом даних для їх подальшого використання у Mashup системі.

$$S(di_i) \to di_i \to DI \to \lambda(di_i, t_p, c_S) \to di_i^* \to DI^* \to D(DI^*), \qquad (2.10)$$

де: $S(di_i)$ - джерело даних; $D(DI^*)$ - база даних.

Для підвищення якості результату процесу визначення вхідних даних $\lambda : DI \to DI^*$ було розроблено метод опису структури і змісту вхідних даних, який складається із певних етапів - кроків для опису структури і змісту вхідних інформаційних ресурсів, також, було описано завдання, які потрібно вирішити на кожному з цих кроків. Рішення завдань кожного кроку методу отримуються використовуючи такі семантично-орієнтовані технології, як онтології і дескриптова логіка [54, 79].

Метод опису структури і змісту вхідних даних – комплекс заходів забезпечення підвищення якості результату отримання структурованих даних з різних інформаційних джерел для їх подальшого використання у системі динамічної інтеграції слабоструктурованих даних. Метод подано, як послідовність кроків:

Крок 1. Визначення форми подання вхідних даних.

Крок 2. Встановлення границь та задоволення основних обмежень, що стосуються вхідного потоку web-інформації.

Крок 3. Класифікація вхідних даних відповідно до предметної області із виділенням і збереженням семантики даних шляхом формування глобальної метамоделі всіх систем, що інтегруються [30]. Даний крок поділяється на етапи:

- структурна онтологія системи, що інтегрується;
- загальна структурна онтологія всіх систем, що інтегруються;
- глобальна метамодель систем, що інтегруються.

Крок 4. Виділення у вхідному інформаційному ресурсі набору реквізитів (атрибутів), що відображають його основні характеристики і аспекти зазначеної предметної області.

Крок 5. Формування опису кожного вхідного інформаційного ресурсу загальної структури, визначеної предметною областю.

Проаналізуємо детально кожен із сформованих основних кроків методу опису структури і змісту вхідних даних для роботи Mashup системи.

Згідно першого кроку – першочерговим завданням роботи Mashup системи для опису структури і змісту вхідних даних є визначення форми подання вхідних даних: структуровані, слабоструктуровані чи неструктуровані дані. Mashup система, зазвичай, може об'єднувати дані, описані в різних форматах. Наприклад, web-формат використовується для публікації інформації, що здатна часто оновлюватися, наприклад, запис у блогах, сайтах новин і так далі; табличний формат підходить для опису моделей даних на основі таблиць, таких як CSV-файли або електронні таблиці; формат на основі розмітки (наприклад, HTML і XML), зазвичай, часто використовується для публікації даних; мультимедійний контент, такий як відео, аудіо та зображення, стають все більш поширеними. Ці типи даних можуть бути доступні користувачеві з різних джерел даних. Найбільш поширеними джерелами даних можуть бути традиційні системи керування базами даних, локальні файли, які доступні у файловій системі власника, web-сторінки, web-сервіси і web-додатки.

В дисертаційній роботі розглядалися певні наперед визначені системи для інтеграції (соц. мережа, web-сторінка, web-сайт, web-сервіс), тому вхідними

даними виступають інформаційні ресурси цих систем. Формати інформаційних ресурсів, які розглядалися: web-формат, табличний формат, формат на основі розмітки та мультимедійний контент.

$$DI_0^* = \lambda_1(DI, C_{1S}, T), \tag{2.11}$$

де: λ_1 - оператор визначення форми подання вхідних даних; C_{1S} - множина умов визначення форми подання вхідних даних.

Відповідно до другого кроку методу, а саме: встановлення границь та задоволення основних обмежень, що стосуються вхідного потоку web-інформації, існує необхідність визначити необхідні умови роботи із вхідними даними. Наприклад, межу кількості інформаційних джерел, межу кількості інформаційних ресурсів у визначених джерелах, пріоритетність джерел та ресурсів, тощо.

$$DI_1^* = \lambda_2(DI, \lambda_1(DI, C_{1S}, T), C_{2S}) \tag{2.12}$$

де: λ_2 - оператор встановлення границь та задоволення основних обмежень; C_{2S} - множина умов встановлення границь та задоволення основних обмежень.

Відповідно до третього кроку потрібно використовувати такі методи і технології класифікації вхідної інформації відповідно до предметної області, щоб можна було виділити і зберегти семантику даних.

Очевидно, що при використанні гетерогенних даних, які до того ж мають різну структуру або взагалі неструктуровані, які часто описують одну й ту ж проблемну область з використанням різних термінів і понять, для успішного вирішення завдання визначення структури і змісту отриманих вхідних інформаційних ресурсів їх семантика повинна бути явним чином виражена і збережена разом з цими даними. У зв'язку з цим актуально при вирішенні даної проблеми скористатися семантично орієнтованими технологіями, такими як онтології і дескриптова логіка [141, 150, 153]. При цьому використання

онтологій не тільки дозволяє створювати моделі даних, адекватні реальному світу, а й відповідає загальному напрямку робіт у галузі стандартизації World Wide Web в рамках проекту Semantic Web, що дозволяє розглядати проблеми інтеграції даних та інтелектуалізації WWW з єдиної точки зору. Розроблені для цих цілей уніфікована модель даних RDF і мова web-онтологій OWL [95, 132] надають багаті можливості семантичного опису розподілених в Інтернет ресурсів.

На даному кроці, при автоматичній класифікації інформації, пропонується в першу чергу автоматично моделювати структурну онтологію кожного з інтегрованих додатків. Цю роботу система повинна виконувати після того як обрані джерела для Mashup (третій стан). Завдяки побудові повноцінної структурної інформаційної метамоделі, яка зможе об'єднати в собі всі елементи систем та їх відношення, можна, таким чином, провести процедуру класифікації вхідних даних, зберігши при цьому семантику даних. Для вирішення даного завдання, як основу, було використано стандарт онтологічного моделювання IDEF5 [100]. Даний стандарт складається з п'яти основних кроків:

1. Організація і визначення меж. Знаходження призначення онтології, контексту використання;

2. Збір даних. Необроблена інформація збирається стандартними засобами збору даних;

3. Аналіз даних. Зібрані дані аналізуються експертами з предметної області;

4. Розробка первісної онтології. Така онтологія включає в себе деякі базові поняття і відношення між ними;

5. Збагачення онтології і перевірка. Шляхом додавання нових концептів і відношень базова онтологія розширюється і тестується.

Важливою перевагою стандарту є те, що він допускає покроковий розвиток онтологічної моделі, за рахунок додавання нових семантичних зв'язків. Такий підхід надзвичайно зручний при автоматизованому добуванні

онтології, тому саме методологія IDEF5 була обрана як основа для вирішення завдання автоматичної класифікації інформації. Важливо зазначити, що такі кроки, як визначення меж онтології і збір даних не мають сенсу в контексті інтеграції інформаційних систем, тому що кордони вже визначені предметною областю функціонування інтегрованих систем, а самі дані вже зберігаються всередині баз даних. Суть процедури автоматичної онтологічної класифікації вхідної інформації наступна:

1. Структурна онтологія системи, що інтегрується. Витяг інформації про структуру кожної з інтегрованих інформаційних систем в онтологічному форматі;

2. Загальна структурна онтологія всіх систем, що інтегруються. Об'єднання отриманих онтологій структури в загальну структурну онтологічну інформаційну модель;

3. Глобальна метамодель систем, що інтегруються. Створення на основі онтологій верхнього рівня, онтології предметної області та загальної структурної онтології глобальної метамоделі, яка описує семантичні відношення між інформаційними системами, що інтегруються.

$$DI_2^* = \lambda_3(DI, C_{3S}, T, DI_0^*, DI_1^*), \tag{2.13}$$

де: $\lambda_3 = \lambda_{31} \circ \lambda_{32} \circ \lambda_{33}$ - оператор формування глобальної метамоделі всіх систем, що інтегруються; C_{3S} - множина умов формування глобальної метамоделі всіх систем, що інтегруються.

Процес добування даних зі сторінки HTML відбувається наступним чином. Web-сторінка, яку повертає сервер, відформатована з використанням мови розмітки, для подальшого відображення в тому чи іншому вигляді за допомогою спеціальної програми (web-браузер). Відбувається процес записування даних у БД, згідно наперед визначеним особливостям опису інтегруючої web-системи.

Далі процес полягає в перетворенні структури ІС з табличного вигляду, використовуваного в схемах БД, в вигляд, призначений для опису онтологій. Відповідно до стандарту IDEF5 моделювати онтологію слід в два етапи:

1) первинне створення протосутностей онтології і наступні додавання семантичних зв'язків [96]. Під протосутностями розуміються об'єкти описуваної предметної області, з деякими базовими характеристиками або властивостями. Кожна така протосутність по стандарту повинна бути унікальною і описуватися в спеціальній формі, яка має, як мінімум наступні поля: унікальний ідентифікатор; ім'я; опис. Даний етап поділяється на:

- первинна трансляція структури таблиць баз даних в протосутності у форматі RDF, назви таблиць стають класами, а поля таблиці - властивостями. Кожен клас і властивість володіють унікальними ідентифікаторами, що відповідають первинному ключеві таблиці;

- отримана початкова онтологічна модель збагачується за рахунок зв'язування між собою класів і властивостей, засобами розширення RDFs. На виході отримуємо онтологічну модель структури одного із додатків, що інтегруються.

2) Розширення онтологічної моделі шляхом додавання нових відношень до протосутностей, перетворивши їх на повноцінні об'єкти онтології. Тобто, витягуються дані про структуру інформаційної системи з джерел у форматі онтології.

Формально, структурна онтологія системи, що інтегрується має вигляд:

$$O = \langle X, Att, f_{Att}(X), R \rangle, \tag{2.14}$$

де: $X = \{x_i \mid i = \overline{1, N_X}\}$ - множина класів (концептів), $x_i = \{name_i, .type_i\}$; $Att = \{att_{ij}, i = \overline{1, N_{Att}}, j = \overline{1, M_{Att}}\}$ - множина атрибутів концептів; $f_{Att}(X)$ - функція, що визначає для кожного концепту множину його атрибутів; $R = \{r_i \mid i = \overline{1, N_R}\}$ – множина відношень між концептами онтології.

Наступне завдання полягає в об'єднанні розрізнених мета-моделей в одну єдину загальну онтологічну модель структури інформаційної системи. Для цієї мети необхідно підготувати онтології додатків. По-перше, слід створити унікальний префікс системи, що інтегрується, використовуючи механізм простору імен. Використання префіксів і просторів імен, дозволяє істотно розширити описові можливості та зберігати декілька онтологічних моделей всередині одного RDF документу. По-друге, необхідно провести автоматичний аналіз всіх елементів усередині витягнутих онтологій з метою знаходжень зв'язків між ними. Це дозволить додати семантичних властивостей, що пов'язують онтології структури різних інформаційних систем між собою.

$$Os = \bigcup_{i=1}^{n} O_i \,, \tag{2.15}$$

За допомогою формули (2.15) зображено загальну метамодель структури всіх інтегрованих систем, однак її ще недостатньо для автоматичного логічного аналізу при витяганні семантичних метаданих інформаційних ресурсів.

Для цього метою третього завдання є ще більше розширення моделі і перетворення її на глобальну метамодель, що використовує інформацію про предметну область та світ поза нею. Для цієї мети слід імпортувати поняття, концепти і відношення з заздалегідь підготовленої онтології предметної області, а також із загальних онтологій верхніх рівнів. Більш того, слід скористатися програмними редакторами онтологічних моделей і послугами експерта в предметній області. З їх допомогою можливо ще більше збагатити модель семантичними зв'язками і створити онтологію, яка семантично описує всі структурні елементи систем, що інтегруються, з урахуванням предметної області в якій вони працюють.

Опис глобальної метамоделі систем, що інтегруються міститься у формулі (2.16):

$$DI_2^* = \langle X, Att, R, f_{Att}(X), f_W(X), W \rangle, \tag{2.16}$$

де: $W = \{w_{ij} \mid i = \overline{1, N_W}, j = \overline{1, M_W}\}$ - словник, в якому визначаються терміни предметної області; $f_W(X)$ - функція інтерпретації концептів, зіставляє концепту набір термінів зі словника *W*.

Аналізуючи третій крок – виділення у вхідному інформаційному ресурсі набору реквізитів (атрибутів), що відображають його основні характеристики і аспекти зазначеної предметної області здійснюється за допомогою онтологій, використовуючи при цьому розроблену на третьому кроці процедуру автоматичної онтологічної класифікації даних у вхідному інформаційному ресурсі.

$$DI_3^* = \lambda_4(DI_2^*, C_{4S}, T), \tag{2.17}$$

де: λ_4 - оператор виділення метаописів інформаційних ресурсів; C_{4S} - множина умов виділення метаописів інформаційних ресурсів.

Завдяки розробленій на третьому кроці структурній інформаційній метамоделі можна на її основі виділити в обраній системі для інтегрування потрібні метадані інформаційних ресурсів, що зберігаються в системі. Тобто витягнути з баз даних семантичні метаописи інформаційних ресурсів, що зберігаються в них за допомогою глобальної метамоделі і механізмів автоматизованого логічного прийняття рішень. В результаті отримано наступний вигляд глобальної метамоделі систем, що інтегруються:

$$DI_3^* = \langle X, Att, R, f_{Att}(X), f_W(X), W, Mir \rangle, \tag{2.18}$$

де: $Mir = \{Mir_i \mid i = \overline{1, N_{Mir}}\}$ - множина метаописів інформаційних ресурсів.

Розроблена метамодель грає, по суті, роль словника, що зберігає в собі інформацію про структуру інтегрованих систем. На основі цього словника можна отримати з джерел даних семантичні метаописи самих збережених ресурсів. Завдяки прописаним в загальній метамоделі властивостям, що зв'язує різні таблиці і їх поля в базах даних, зробити це набагато простіше за

допомогою механізмів логічного мислення. Такі механізми, використовуючи логіку першого порядку і загальну онтологічну модель структури додають потрібні семантичні зв'язки в отримані метадані. Синтаксис і семантика даних побудовані таким чином, що основні логічні проблеми є розв'язані, тому отримання нових знань можна здійснювати комп'ютерними засобами [20], автоматично. У підсумку, на виході отримується повна онтологія, що містить інформацію про структуру інформаційної системи, або іншими словами понятійну частину, і метаописи ресурсів, збережених в системах, тобто змістовну частину.

Відповідно до п’ятого кроку, завершальним етапом роботи Mashup системи для визначення структури і змісту вхідних інформаційних ресурсів є формування опису вхідного інформаційного ресурсу загальної структури, визначеної предметною областю. Для вирішення даної задачі хочеться зазначити, що Mashup системи працюють, використовуючи API додатків для інтеграції між кількома додатками, дозволяючи вилучення даних і обмін даними між додатками. API допомагають отримати доступ і використовувати ресурси, не зосереджуючись на своїй внутрішній організації. Оскільки у всіх ресурсів API є унікальні URI з HTTP-доступом, було створено математичну модель інформаційного ресурсу, де фігуруватиме URI ресурсу.

$$DI_4^* = \langle URI, IR, TR, M, El, P_{RM}, P_R, C, FC \rangle, \qquad (2.19)$$

де: *URI* - адреса Web-сервісу, представлена ідентифікатором URI даного сервісу; *IR* – ідентифікатор ресурсу; *TR* – тип ресурсу; *M* – подання ресурсу у метамоделі системи; $El = \{el_i \mid i = \overline{1, N_{El}}\}$ – множина елементів об'єктної схеми інформаційного ресурсу; $P_{RM} : El \to X$ - відображення, що ставить у відповідність елементу об'єктної схеми його концепт; $P_R : El \times El \to R$ - відображення, що ставить у відповідність зв'язкам між елементами об'єктної схеми відношення в онтології, і для будь-якого елементу $el \in El$ виконується умова: множина атрибутів елементу об'єктної схеми відповідає атрибутам його концепту, тобто

$\{att : \langle att, d \rangle \in el\} = P_X(P_{El}(el))$; *C* (Calls) - кількість викликів web-сервісу; *FC* (Failed calls) - кількість невдалих викликів.

Для представлення функцій інформаційних ресурсів пропонується використовувати web-сервіси з приписуванням їм семантики проблемної області. Представлена модель інформаційного ресурсу дозволяє встановлювати відображення між реалізованими в web-сервісах функціями і атрибутами класів прикладної онтології. Дані відображення показують, які ресурси можуть привласнювати значення яким атрибутам.

Було проаналізовано операції над онтологіями, що використовуються в різних підходах, і сформований набір операцій, що дозволяє отримати онтологічну модель контексту зі знань, представлених прикладною онтологією.

Набір операцій над онтологіями:

1) Формування сукупності ключових слів:

- тип ситуації;
- опціонально: завдання, що вирішуються, доповнення, уточнення.

2) Пошук:

- маркування елементів прикладної онтології, імена яких збігаються з ключовими словами.

3) Вибірка:

- маркування елементів прикладної онтології, релевантних для створення моделі поточної ситуації.

4) Добування:

- формування зрізів прикладної онтології на підставі маркованих елементів.

5) Інтеграція:

- об'єднання зрізів, отримання абстрактного контексту.

Перші три операції - пошуку, вибірки і вилучення - використовуються для виявлення знань прикладної онтології, які є релевантними поточній ситуації. У результаті цих операцій формується множина зрізів прикладної онтології, в

результаті інтеграції яких формується онтологічна модель абстрактного контексту. Ця модель розширюється множиною контекстно-залежних web-сервісів, яка формується на підставі відображень між web-сервісами та атрибутами класів прикладної онтології, включеними в абстрактний контекст.

Отже, було запропоновано підвищення якості результату процесу визначення вхідних даних Mashup системи, шляхом застосування процедури опису структури і змісту вхідних інформаційних ресурсів. Для цього було розроблено метод опису структури і змісту вхідних даних.

2.3 Формування моделі системи із динамічною структурою, враховуючи зміст інформаційних ресурсів

Модель даних розглядається як алгебраїчна система, тобто множина допустимих типів даних, визначених на них відношень та операцій, а також обмежень цілісності, які можуть накладатися на дані [156].

Було розглянуто, для порівняння, два типи моделі предметної області - структурно-статичний і структурно-динамічний тип. Нехай загальна структурно-статична модель системи матиме наступний вигляд:

$$M_{total} = \langle O, R, L \rangle, \tag{2.20}$$

де: O - сукупність типів об'єктів; R - сукупність відношень між типами об'єктів; L - сукупність обмежень; O, R, L незмінні в часі.

Тоді, процес формування моделі системи із динамічною структурою, враховуючи зміст інформаційних ресурсів має наступний вигляд:

Крок 1. Визначення набору об’єктів в дискретний момент часу t.

Нехай інформаційним об’єктом є інформаційний ресурс. Нехай O - і-тий інформаційний ресурс, де:

$$O_i = \langle URI, IR, TR, M, El, P_{RM}, P_R, C, FC \rangle, \tag{2.21}$$

Всі характеристики O описано після формули (2.19).

Крок 2. Визначення набору відношень між заданими об'єктами в дискретний момент часу t.

$$R = \{R_1, ..., R_n\}, \tag{2.22}$$

де: $R_1, ..., R_n$ – відношення між поняттями в моделі.

Крок 3. Визначення набору обмежень на задані об'єкти в дискретний момент часу t.

Множина обмежень на змінні визначається згідно рішенню задачі задоволення обмежень:

$$SCT = (V_{SCT}, D_{SCT}, C_{SCT}), \tag{2.23}$$

де: V_{SCT} – сукупність змінних, D_{SCT} – сукупність доменів змінних, C_{SCT} – сукупність обмежень на змінні.

Крок 4. Визначення набору виконуваних дій (Actions) над заданими об'єктами.

Було досліджено, що основними операціями, які можуть виконуватися над об’єктами Mashup системи є наступні:

$$Ac = \{new, union, join, filter, sort, refresh, delete\}. \tag{2.24}$$

Крок 5. Забезпечення оновлення даних, відповідно до події, що настала.

Перехід T_{ij} (Transition) від одного відображення P_i (Presentation) системи до іншого P_j може ініціюватися деякою подією E_{ij} (Event):

$$T_{ij}(t) = T_{ij}(P_i(t-1), E_{ij}(t) = truth) = P_j(t). \tag{2.25}$$

Також, перехід можна представити у наступному вигляді:

$$P_i(t-1) \xrightarrow{E_{ij}(t)} P_j(t). \tag{2.26}$$

Під настанням події розуміється, що в деякий момент часу t система умов-предикатів (Condition) істинна:

$$E_{ij}(t) = \begin{Bmatrix} C_{ij}^{1}(t) = truth, \\ ... \\ C_{ij}^{k}(t) = truth. \end{Bmatrix}. \quad (2.27)$$

Тоді матриця переходів відображень моделі предметної області має наступний вигляд (табл. 2.4).

Таблиця 2.4.

Матриця переходів відображень

Початкове відображення	Наступне відображення			
	$P_1(t)$	$P_2(t)$	…	$P_n(t)$
$P_1(t-1)$	E_{11}	E_{12}	…	E_{1n}
$P_2(t-1)$	E_{21}	E_{22}	…	E_{2n}
…	…	…	E_{ij}	…
$P_n(t-1)$	E_{n1}	E_{n2}	…	E_{nn}

При умові, що, якщо не існує E_{ij}, такої що $E_{ij}=E_{kp}$, якщо $i \neq k$, $j \neq p$, тобто одна і та сама подія може ініціювати тільки один перехід відображень. Причому можлива ситуація, коли $E_{ij}=false$, і це говорить про те, що ніколи не відбудеться перехід відображення P_i у відображення P_j за умови $i \neq j$.

Було проаналізовано, якого типу подія E_{ij} має статися, щоб не було переходу відображень, тобто $P_i(t-1)= P_i(t)$. Можливо кілька варіантів.

Варіант 1 - коли настає подія E_{ij} такого типу:

$$E_{ij} = truth, i = j, \quad (2.28)$$

Це означає, що подія E_{ij} не залежить ні від яких змінних, в тому числі і від часу t. Такий випадок якраз і описує структурно-статичну модель предметної області:

$$P_i(t-1) = P_i(t) = const. \quad (2.29)$$

Варіант 2 – якщо ніякої події не настало, то немає зміни відображень.

Враховуючи формулу (2.21) фазова траєкторія (стан) і-го інформаційного ресурсу в кожен дискретний момент часу t буде мати вигляд:

$$\varphi_i(t) = O_i(t)(URI_i(t), IR_i(t), TR_i(t), FM_i(t), El_i(t), P_{RMi}(t), P_{Ri}(t), C_i(t), FC_i(t)). \quad (2.30)$$

Враховуючи, що фазовий простір інформаційного ресурсу може бути поданим як множина станів інформаційного ресурсу, які в кожен момент часу t описують кортежем значень характеристик інформаційного ресурсу, структурно-динамічна модель Mashup системи в дискретний момент часу t матиме наступний вигляд:

$$\langle O(t), R(t), C_{SCT}(t), Ac(t)\rangle = \langle O(t), R(O(t)), C_{SCT}(O(t)), Ac(O(t))\rangle \ , \quad (2.31)$$

де: O - набір об'єктів, R - набір відношень між об'єктами, C_{SCT} – набір обмежень на об’єкти, Ac – набір виконуваних дій (Actions) над визначеними об'єктами.

2.4 Висновки до розділу

1. Згідно проведеного дослідження сучасних найбільш популярних Mashup систем виділено такі групи особливостей проектування і функціонування цих систем: формати і доступ до даних, організація внутрішньої моделі даних, робота з вхідним і вихідним потоком даних та функціональні можливості.

2. В загальному основний принцип функціонування системи динамічної інтеграції слабоструктурованих даних полягає у здійсненні процесу відображення вхідних наборів даних у вихідний набір. Для опису процесу опрацювання даних Mashup системою було здійснено концептуалізацію та моделювання роботи системи динамічної інтеграції слабоструктурованих даних, використовуючи принципи STD та UML діаграми діяльності.

3. Показано, що процес опрацювання даних системою динамічної інтеграції слабоструктурованих даних потребує деталізації на два підпроцеси: визначення вхідних даних і формування вихідних даних. Для підвищення якості виділених підпроцесів запропоновано метод опису структури і змісту вхідних даних та метод формування об’єднаного динамічного набору вихідних даних.

4. Розроблено метод опису структури і змісту вхідних даних, що дає змогу підвищити якість процесу отримання структурованих даних з різних інформаційних джерел для їх подальшого використання у системі динамічної інтеграції слабоструктурованих даних.

5. Згідно визначених принципів формування фазового простору інформаційного ресурсу, як множини станів інформаційного ресурсу, які в кожен момент часу t описують кортежем значень характеристик інформаційного ресурсу, отримано можливість сформувати структурно-динамічну модель Mashup системи в дискретний момент часу t.

РОЗДІЛ 3 МЕТОД ТА АЛГОРИТМИ ФОРМУВАННЯ ОБ'ЄДНАНОГО ДИНАМІЧНОГО НАБОРУ ДАНИХ, ЯКИЙ МАЄ ЗАГАЛЬНУ СТРУКТУРУ І ЄДИНИЙ ЗМІСТ

3.1 Принципи формування об'єднаного динамічного набору даних, який має загальну структуру і єдиний зміст

Перш ніж визначити принципи формування об'єднаного динамічного набору даних, який має загальну структуру і єдиний зміст було проаналізовано, з точки зору розробника, ряд факторів, що впливають на процес інтеграції інформаційних ресурсів для формування об'єднаного динамічного набору даних, який має загальну структуру і єдиний зміст. До таких факторів належать:

1) Швидкий розвиток. Прогресування інформатизації потребує все частіше і частіше змінювати структури даних, бізнес-процеси, не говорячи вже про дизайн і інтерфейс, які оновлюються настільки швидко, що не встигаєш відстежити. І, в цій динаміці, де «змінність» відображає саму природу та суть системи, має місце проблема інтеграції інформаційних ресурсів, враховуючи всю динаміку розвитку систем.

2) Гетерогенність. При розгляді великого проекту, не завжди є можливість забезпечувати відповідність платформам та інструментам, що належать одному виробнику, звідси, отримуємо проблему багатоплат-формленності.

3) Розподіленість. Підприємства збільшуються, а завдання все більш комплексно вирішуються – тому, має місце проблема вирішення розподіленості.

4) Страх новизни. Інколи досить важко абсолютно відмовитися від застарілих технологій, за допомогою використання яких ще можна отримати

досить непогані показники по надійності і продуктивності, але це аж ніяк не сприяє інтеграції.

5) Слабка зв’язність. Існують випадки, коли не можливо повністю формалізувати, специфікувати і структурувати дані – звідси, частина моделі може залишитися слабо зв’язаною, що може затруднити її комп’ютеру обробку.

6) Інтерактивність. Користувачі інформаційних систем постійно піднімають планку щодо очікуваної швидкодії системи.

7) Мобільність. Користувачі різного роду інформаційних застосунків стали рухатися швидше, а взаємозв’язок з ними тепер може існувати всюди: як в транспорті, так і вдома, на вулиці, тощо.

8) Безпека. Коли інформація зберігалася на носії всередині приміщення, що охороняється, не виникало потреби у захисті самого тексту інформації, а тепер, коли все стало доступним через використання Всесвітньої мережі, має місце проблема захисту даних, їх все можливому шифруванні, тощо.

9) Високонавантаженність. Важкість забезпечення інтеграції залежить від: кількості користувачів застосунку, можливостей опрацювання даних, обсягів даних, тощо.

Забезпечити загальну структуру і єдиний зміст інформації при формуванні об'єднаного динамічного набору даних можливо тільки, якщо використовувати семантичну інтеграцію даних (табл. 3.1).

На основі поданої таблиці 3.1 можна побачити відносні переваги семантичного підходу до інтеграції даних, порівняно із традиційним підходом.

Семантичний підхід, як можна побачити із табл. 3.1 вимагає використання, відповідно, семантично-орієнтованих технологій, таких, як, наприклад, онтології (рис. 3.1).

Рисунок 3.1 - Систематизація знань в області онтологій

Таблиця 3.1 - Порівняльна характеристика традиційного та семантичного підходів до інтеграції даних

	Традиційна інтеграція	Семантична інтеграція
Структура даних	Переважно реляційна, орієнтована на однакові набори даних.	Орієнтована на відношення між одиницями даних, не залежно від їх подібності.
Метод інтеграції	Дані витягуються із джерел, перетворюються, відповідно до заданих вимог і завантажуються в сховища.	Встановлюються зв'язки між одиницями даних, відповідно до визначень у спільних для них онтологіях.
Масштабування	Із кожним новим джерелом даних вартість росте експоненційно.	Збільшення кількості джерел практично не впливає на вартість.
Походження джерел	Внутрішнє.	Зовнішнє і внутрішнє.
Відношення до стандартів	Жорстке дотримання стандартів, порушення призводить до втрати контексту.	Гнучке дотримання стандартів, контекст зберігається в будь-яких умовах.

Інтегруючи дані із застосуванням онтологій можна отримати дані, що зберігаються у джерелі, об’єднати їх з логічними і глобальними онтологіями за допомогою відображення [141].

При розгляді систем на основі Mashup технології, має місце поняття «піраміди значень». «Піраміда значень» Mashup системи динамічної інтеграції даних має вигляд, як на рис. 3.2.

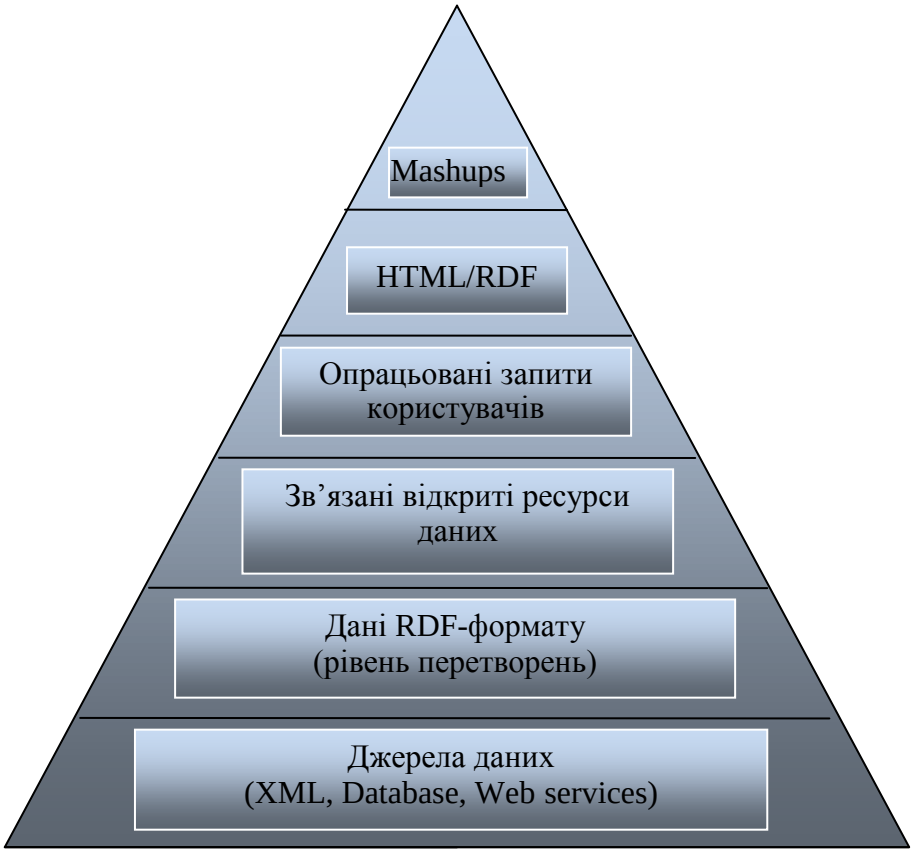


Рисунок 3.2 - «Піраміда значень» Mashup системи

На основі аналізу моделей контексту, що використовуються в контекстно-керованих системах, сформульовано вимоги до моделі контексту та контекстно-керованих систем формування об'єднаного динамічного набору даних, який має узгоджену структуру і єдиний зміст (табл. 3.2). Та на підставі цих вимог розроблено принципи побудови і функціонування контекстно-керованої системи формування об'єднаного динамічного набору даних, який має узгоджену структуру і єдиний зміст.

Таблиця 3.2 - Принципи побудови і функціонування контекстно-керованої системи

№ п/п	Специфікація вимог до контексту у контекстно-керованій системі	Принципи побудови і функціонування контекстно-керованої системи
1	Контекст описується стандартизованими способами.	Використання онтологій для опису контексту.
2	Модель представлення знань підтримує операції з контекстом.	Використання онтологічної моделі подання знань.
3	Контекст надає релевантну інформацію для вирішення задачі або розуміння ситуації.	Глобальна мета-модель контексту, що використовує інформацію про предметну область та світ поза нею.
4	Підтримка механізмів повторного використання контексту та отримання контексту більш високого рівня абстракції.	Дворівнева модель контексту.
5	Підтримка механізмів використання субконтекстів.	Подання завдань у вигляді ієрархії підзадач.
6	Контекстна інформація містить дані, історію їх отримання і знання.	Використання онтологічної моделі контексту, наявність архіву контекстів.
7	Контекст містить відому залученим в ситуацію об'єктам інформацію.	Доступ залучених у ситуацію об'єктів до контексту.

На основі аналізу моделей контексту, що використовуються в контекстно-керованих системах, були сформульовані вимоги до моделі контексту і контекстно-керованих систем формування об'єднаного динамічного набору даних, який має узгоджену структуру і єдиний зміст (табл. 3.1). На підставі цих вимог були розроблені принципи, покладені в основу концептуальної моделі побудови і функціонування контекстно-керованої динамічної системи формування об'єднаного динамічного набору даних, який має узгоджену структуру і єдиний зміст. Основоположними є перші три принципи.

3.2 Аналіз вхідних інформаційних ресурсів для визначення їх спільних рис та виявлення зв'язків між ними

3.2.1 Визначення подібності схем даних систем, що інтегруються

Існує декілька підходів відображення подібності елементів баз даних, описаних в роботах Ерхарда Рама і Філіпа Бернштейна [138, 139]. Їх можна розділити на: підходи на основі аналізу рядків, підходи на онові аналізу мови в контексті, підходи на основі опорних слів із словників і тезаурусів, підходи на основі обмежень (типи даних, властивості). Всі підходи, за винятком, заснованого на використанні словника, працюють тільки з інформацією, що зберігається в схемах [122, 140]. Тезаурусний підхід також оперує загальнодоступною інформацією про значення слів та можливих синонімів. Такий спосіб є більш ефективним, однак його не завжди можна реалізувати на практиці у зв'язку з відсутністю повноцінного тезауруса з синонімами конкретної предметної області. Тому було запропоновано проводити аналіз подібних елементів на основі гібридного підходу, що поєднує в собі аналіз рядків, природної мови і обмежень схем баз даних. Щоб формалізувати дану задачу введено деякі позначення понять.

Нехай інформація про деяку web-систему вже попередньо записана і міститься у певній схемі бази даних: *S*.

$$S = \{A_1, ..., A_m\}, \tag{3.1}$$

де: $A_1, ..., A_m$ - атрибути схеми бази даних: імена таблиць (*TN*), назви полів в таблиці (*TS*), типи даних (*DT*), первинні та зовнішні ключі (*PK*, *FK*), обмеження (*C*), збережені дані (*SD*).

Відповідністю між елементами двох інтегрованих схем S_1 і S_2 є множина:

$$Match(S_1, S_2) = \{A(S_1), A(S_2), f, w\}, \tag{3.2}$$

де: $A(S_1)$ та $A(S_2)$ - атрибути схем S_1 і S_2 відповідно; f – функція, що визначає ступінь подібності двох атрибутів. Можливі значення функції містяться в інтервалі $\{0,1\}$; w – вага подібності: $w \geq 0$.

При автоматизованому аналізі елементів систем, що інтегруються припускається, що всі текстові описи елементів в тій чи іншій мірі містять в собі деяку семантичну або смислову складову. Таким чином, завдання полягає в обчисленні декількох примітивних функцій, що визначають подібність між атрибутами. У даному підході пропонується порівнювати кожен атрибут однієї схеми з кожним атрибутом іншої схеми. Отже, нехай існують деякі схеми S_1 і S_2 з атрибутами $A(S_1)_i$ та $A(S_2)_j$ відповідно.

Визначення подібності схем полягає в отриманні ймовірності збігу кожного атрибута за деякою функцією. Для кожного виду атрибутів і ознак, за якими їх оцінюють, використовується окрема функція. Порівняння необхідно виконувати, як за ознаками з невеликою вагою, такими як збіг типів даних у двох атрибутів, так і за більш важливими ознаками. Для порівняння атрибутів схем потрібно, використовувати наступні порівняння:

- порівняння елементів за типом даних. У зв'язку з тим що, різні реляційні дані використовують різні типи даних, необхідно скласти таблицю відповідностей;
- порівняння за анотаціями і описами;
- порівняння з використанням опорних слів. Використовуючи онтологію предметної області, як словник опорних слів, можна знаходити відповідності між елементами за допомогою цього словника.

В результаті аналізу отримується набір коефіцієнтів відповідностей для різних атрибутів за деякими ознаками, що володіють різними вагами w:

$$Sim(S_1, S_2) = \{p_{u_1} w_{u_1}, ..., p_{u_9} w_{u_9}\}, \quad (3.3)$$

де: $Sim(S_1, S_2)$ – відповідності схем; p_{u_1} - коефіцієнт відповідності за деякою ознакою u_1; w_{u_1} – вага ознаки.

Таблиця 3.3 - Ознаки для порівняння

Ознака	Вид порівняння
u_1	порівняння за типом даних
u_2	порівняння за анотаціями і описами
u_3	порівняння на основі опорних слів
u_4	порівняння назв таблиць
u_5	порівняння назв полів таблиць
u_6	порівняння обмежень
u_7	порівняння на виявлення дублікатів
u_8-u_9	додаткові порівняння при необхідності

Ваги для кожної ознаки переважно визначаються вручну [130]. Для визначення загальної імовірності відповідності елемента схеми бази даних по сукупності відповідностей атрибутів пропонується використовувати лінійний метод:

$$Sim(S_1, S_2) = \{p_{u_1} w_{u_1} + p_{u_2} w_{u_2} +,...,+ p_{u_9} w_{u_9}\}, \quad (3.4)$$

А також логістичну регресію, яка володіє деякими перевагами перед лінійною функцією:

$$Sim(S_1, S_2) = \sigma\left(\sum_i p_i w_i - p_0\right), \quad 1 \le i \le 9, \quad (3.5)$$

де: p_0 – мінімальне значення відповідності (p_0=0.1); σ – функція з наступною інтерпретацією:

$$\sigma = \frac{1}{1 - e^{-x}}. \quad (3.6)$$

За допомогою описаних рекомендацій аналізу подібності елементів можна знайти відповідність структур систем, що інтегруються, а також дублікатів при витяганні самих інформаційних ресурсів з баз даних.

3.2.2 Визначення подібності інформаційних ресурсів систем, що інтегруються

При аналізі структурних елементів, необхідно аналізувати назви таблиць і полів з двох інтегрованих схем. При аналізі ж збережених ресурсів, слід порівнювати самі дані. І тут виникає завдання, що полягає в порівнянні рядків. Для його вирішення, в роботі пропонується використовувати метрики схожості рядків. На даний момент існує кілька найбільш часто використовуваних загальновизнаних метрик визначення подібності двох текстових рядків [84, 86]: відстань Левенштейна або дистанція редагування, коефіцієнт Жаккар-Вінклера, коефіцієнт Танімото, а також коефіцієнт Серенсена-Дайса. Найбільш часто вживаною метрикою є відстань Левенштейна або дистанція редагування. Полягає ця метрика в мінімальній кількості правок одного рядка, необхідних для приведення його до вигляду другого рядка. Існує три основних види правок, це - стирання символу, додавання символу та заміна символу. Також існує розширена метрика, під назвою відстань Дамерау-Левенштейна, в якій додається четвертий тип правок - транспозиція або перестановка символів.

Нехай X і Y - два рядки довжиною m і n. Для отримання відстані Левенштейна розраховується матриця відстаней D, в якій кожен елемент $D[i,j]$ містить дистанцію між першими i символами рядка X і першими j символами рядка Y. Відстань Левенштейна визначається за наступною формулою:

$$D(i,j)=\begin{Bmatrix}\max(i,j), if\ \min(i,j)=0\\ \min\begin{Bmatrix}D(i,j-1)+1\\ D(i-1,j)+1\\ D(i-1,j-1)+m(X[1],Y[j])\end{Bmatrix}\end{Bmatrix}, \quad (3.7)$$

де: $m(a,b)=0$, якщо $a=b$ і $m(a,b)=1$, якщо $a\neq b$.

Коефіцієнт Танімото або Жаккар-Вінклера також часто використовується для визначення подібності одного рядка до іншого. В даному випадку рядок розглядають, як множину символів, а метрика подібності визначає кількість однакових символів за такою формулою:

$$p = \frac{c}{a + b - c}, \tag{3.8}$$

де: p – коефіцієнт подібності рядків, $0 \le p \le 1$; c – кількість спільних символів в рядках; a і b – кількості символів у рядках A і B відповідно.

За таким самим принципом діє і коефіцієнт Серенсена-Дайса, що є бінарною мірою подібності рядків і має в загальному вигляді такий запис:

$$p = \frac{2 \times n(X \cap Y)}{n(X) + n(Y)}, \tag{3.9}$$

де: p – коефіцієнт подібності рядків, $0 \le p \le 1$; X і Y – рядки, що порівнюються; $n(a)$ – функція, що визначає кількість символів у рядку a.

Аналізуючи розглянуті коефіцієнти, хочеться зазначити, що жоден з них не може гарантувати хорошого результату при зміні порядку слів у реченні і використанні декількох мов в текстовому рядку [72]. У зв'язку з цим, в роботі пропонується не використовувати безпосередньо ці метрики, а використовувати модифіковані метрики, що працюють на основі методу розбиття слова на N-грами [16]. Такий підхід дозволить оперувати вже не окремими символами в стрічці, а набором символів, що дозволить знаходити збіги найбільш точно.

Нехай є деякий алфавіт $ABC = \{l_1, ..., l_n\}$, де l_i - символ алфавіту. Тоді мовою *L(ABC)* у алфавіті *ABC* є множина ланцюжків кінцевої довжини, що складаються з символів *ABC*. Окремий ланцюжок називатиметься висловлюванням на мові *L(ABC)*, а N-грамою в алфавіті *ABC*, таким чином, називатиметься ланцюжок довжиною *n*.

N-грама [16] може збігатися з деяким висловлюванням, входити в нього чи не входити взагалі в мову. При розбитті рядку, необхідно використовувати, так званий, метод Шінглінга [72], який дозволить створити пересічні N-грами. У таблиці 3.4 наведено приклад розбиття слів на N-грами довжиною 3, методом Шінглінга.

Таблиця 3.4 - Розбиття слів на N-грами методом Шінглінга

Слово	N-грама
Розбиття	Роз/ бит /тя
слів	слі/ в
методом	мет/ одо /м
Шінглінга	Шін/ глі /нга

Було модифіковано формулу обчислення коефіцієнта Серенсена-Дайса, застосувавши N-грами замість символів:

$$p = \frac{2 \times f_{NG}(X \cap Y)}{f_{NG}(X) + f_{NG}(Y)}, \tag{3.10}$$

де: p – коефіцієнт подібності рядків, $0 \le p \le 1$; X і Y – рядки, що порівнюються; $f_{NG}(a)$ – функція, що визначає довжину ланцюжка N-грам рядку a.

Використання не окремих символів, а цілих словосполучень, дозволяє зменшити кількість помилок при аналізі поточних текстових рядків. Даний метод пропонується використовувати, як для порівняння назв таблиць і їх полів, так і даних, що зберігаються в інтегрованих базах даних.

3.3 Алгоритми створення загальної динамічної структури для множини вхідних інформаційних ресурсів із врахуванням їх змісту

Відповідно до розробленого та представленого у другому розділі методу опису структури і змісту вхідних даних для кроку методу, що відповідає за подання систем, що інтегруються у вигляді загальної онтологічної моделі, весь процес класифікації вхідних даних відповідно до предметної області повинен складатися із трьох етапів:

1. Структурна онтологія системи, що інтегрується. Витяг інформації про структуру кожної з інтегрованих інформаційних систем в онтологічному форматі.

2. Загальна структурна онтологія всіх систем, що інтегруються. Об'єднання отриманих онтологій структури в загальну структурну онтологічну інформаційну модель.

3. Глобальна метамодель систем, що інтегруються. Створення на основі онтологій верхнього рівня, онтології предметної області та загальної структурної онтології глобальної метамоделі, яка описує семантичні відношення між інформаційними системами, що інтегруються.

Але, тільки перші 2 етапи можна повністю автоматизувати, використовуючи відповідні стандартні засоби та технології. А от виконання третього етапу вимагає участі експертів по інтеграції web-систем та фахівців з подання знань у вигляді онтологій. Тому важливим завданням є розроблення алгоритмів побудови глобальної метамоделі об'єднаного динамічного набору даних, який має загальну структуру і єдиний зміст. Відповідно для вирішення даного завдання було розроблено наступні алгоритми: алгоритм побудови онтологічної моделі всіх систем, що інтегруються та алгоритм отримання інформаційних ресурсів із системи, що інтегрується.

3.3.1 Алгоритм побудови онтологічної моделі всіх систем, що інтегруються

Для того, щоб інтегрувати певний застосунок до Mashup системи, потрібно, насамперед знати структуру даного застосунку чи системи, що інтегрується. В будь-якій web-системі майже вся інформація зберігається у базах даних. Це, так звані, реляційні бази даних. Для даних, що не збережені в БД було розроблено процедуру їх запису в БД, розподіляючи інформацію відповідно до її опису в тій чи іншій web-системі. В реляційних базах даних інформація про структуру і зв'язки між структурними елементами зберігається в схемах даних, саме ці схеми необхідно отримати в ході роботи алгоритму.

Однак, аналізу самої схеми досить тільки для забезпечення структурної інтероперабельності. Для досягнення семантичної інтероперабельності при витяганні схеми даних потрібно також враховувати смислове призначення цих елементів, тому необхідно використовувати онтологію предметної області. Така онтологія для отриманої моделі додасть зв'язку між поняттями в предметній області. Таким чином, кожна онтологічна модель, що отримана із БД системи, буде підмножиною онтології предметної області.

Перед використанням алгоритму побудови онтологічної моделі всіх систем, що інтегруються потрібно виконати процедуру запису слабо-структурованої html-формату інформації у попередньо створену базу даних.

Процес добування даних зі сторінки HTML відбувається наступним чином. Web-сторінка, яку повертає сервер, відформатована з використанням мови розмітки (в основному HTML), для подальшого відображення в тому чи іншому вигляді за допомогою спеціальної програми (web-браузер). На рис. 3.3 показано приклад візуалізації web-браузером деякої області даних (спільнота соціальної мережі) і вихідний код цих даних.

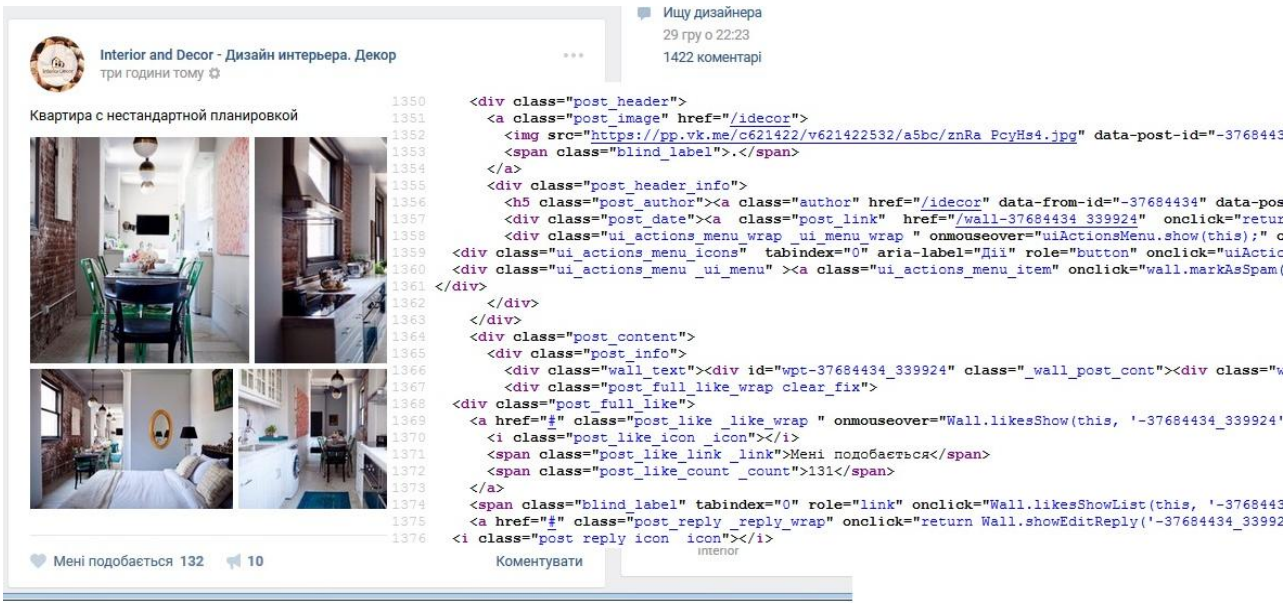


Рисунок 3.3 - Опис даних записів на стіні спільноти «Interior and Decor» соц. мережі vkontakte

Тут, наприклад, опис даних записів на стіні спільноти здійснюється наступним чином:

…

 …

Виділено певну кількість різноманітних класів, які відповідають за свою область даних. Так, клас "wall_text" описує підпис запису на стіні. Важливість тексту в записі визначається наявністю таких характеристик: жирний шрифт, курсив, підкреслення, хештег, тощо.

Для отримання таких даних здійснюється пошук назв тих класів, які відповідають за опис необхідної інформації і отримується їх зміст. Таким чином заповнюється одне з полів результатів таблиць БД. Для автоматизації цього процесу, програми витягнення даних використовується мова запитів до елементів мови розмітки (Xpath).

Наступним етапом є використання алгоритму побудови онтологічної моделі всіх систем, що інтегруються. При побудові будь-якого алгоритму, першочерговим завданням є визначення вхідних та вихідних даних. Вхідними даними для алгоритму побудови онтологічної моделі всіх систем, що інтегруються є:

- структурні схеми баз даних систем, що інтегруються;
- онтологія предметної області.

Нехай O_G (Generally Ontology) - загальна онтологія схем даних всіх систем, що інтегруються:

$$O_G = \{O_i, O_D\}, \tag{3.11}$$

де O_i - онтологія структури і-тої системи; O_D - онтологія предметної області.

Онтологія предметної області зазвичай розробляється попередньо, за участю експерта з предметної області та спеціаліста із представлення знань в онтологічному форматі. Процес створення такої моделі займає тривалий час, однак це потрібно тільки на первинному етапі інтеграції. При подальшому додаванні нових систем, що працюють в даній області, сама онтологія не вимагає додаткових змін.

Вихідними даними є загальна онтологічна модель, яка описує структуру інтегрованих систем в рамках їх предметної області і зв'язки між елементами різних систем. Така модель буде змодельована засобами мови RDF, її розширенням RDFs та із застосуванням мови OWL.

Опишемо роботу алгоритму побудови онтологічної моделі всіх систем, що інтегруються. І для цього введемо деякі позначення понять.

Нехай ми маємо деяку схему бази даних системи: S.

$$S = \{T_1,..., T_m\}, \tag{3.12}$$

де: $T_1,..., T_m$ - таблиці схеми бази даних системи S.

$$T_i = \{A_1,..., A_k\}, \qquad i = \overline{1,n}, \tag{3.13}$$

де: $A_1,..., A_k$ - атрибути таблиць схеми бази даних.

$$R = \{R_1,..., R_z\}, \tag{3.14}$$

де: $R_1,..., R_z$ - зв’язки між концептами онтології.

Алгоритм побудови онтологічної моделі всіх систем, що інтегруються містить в собі 6 основних кроків, а саме:

1) Представлення структури бази даних у вигляді RDF, тобто послідовне відображення схеми S в RDF формат.

$$T_i \to T(RDF)_i,\ A_j \to A(RDF)_j, \quad i = \overline{1,n},\ j = \overline{1,k}, \tag{3.15}$$

де: $T(RDF)_i$ - концепти онтології, описані за допомогою RDF; $A(RDF)_j$ - властивості концептів в онтології.

2) Додавання семантичних властивостей та створення онтології. Даний крок реалізується за допомогою використання процедури визначення спільних рис елементів бази даних і додавання зв'язків між ними.

3) Додавання онтологій верхніх рівнів та онтології предметної області. Реалізуємо даний крок за допомогою мови OWL, використовуючи команду owl: import. Завдяки правилу транзитивності в RDF, додаткові онтології розширюють предметні області і додають нові концепти і властивості.

4) Перевірка створеної онтології. Даний крок реалізується виконанням перевірки і аналізу витягнутої онтології на «зв'язність», тобто ми перевіряємо чи не бракує ніде семантичних зв'язків. Якщо так, тоді переходимо до п'ятого кроку, якщо ні – переходимо до шостого кроку.

5) Редагування витягнутої онтології за допомогою редактора онтології (Protégé) і додавання зв'язків між концептами. Далі повертаємося до кроку 4.

6) Зберігання отриманої загальної онтології структури системи у сховище метаданих у форматі RDF.

Зобразимо діаграму діяльності алгоритму побудови онтологічної моделі всіх систем, що інтегруються на рисунку 3.4.

Розглянемо детальніше всі кроки представленого алгоритму (рис. 3.4). На першому кроці відбувається послідовне відображення структурних елементів схеми даних в RDF формат. Основними елементами баз даних реляційного типу, які необхідно відобразити, є таблиці і їхні атрибути. Самі атрибути (поля таблиць), також володіють важливою структурною інформацією, такою як ім'я та тип атрибуту. Дана інформація отримується за допомогою використання стандартної мови запитів SQL і механізму зовнішніх ключів.

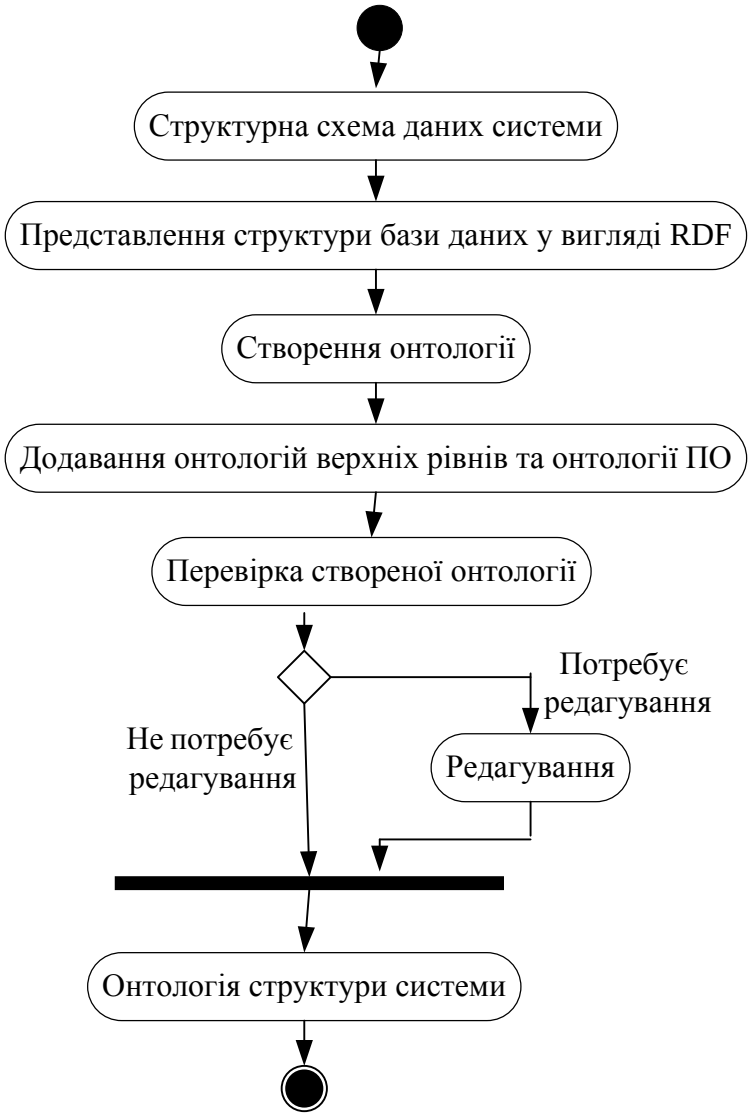


Рисунок 3.4 - Діаграма діяльності алгоритму побудови онтологічної моделі всіх систем, що інтегруються

При аналізі схеми даних, автоматично назви таблиць стають назвами нових класів, а назви полів таблиць властивостями, пов'язаними з їх класом. Також, можна окремо записувати інформацію про відповідність класу RDF таблиці бази в окремий документ у форматі XML або в інше сховище типу ключ-значення. Оскільки, згідно запропонованого методу, кожна витягнута протосутність повинна мати ідентифікатор, таблиця, що аналізується повинна мати первинний ключ. У випадку його відсутності, його необхідно створювати під час обробки таблиці. Таким чином унікальний ідентифікатор протосутності складатиметься з таблиці і первинного ключа.

Розглянемо наступну таблицю:

Таблиця 3.5 - Таблиця Tovar

Назва поля	Тип даних
ID	int
Name	varchar
Price	float
Quantity	int
Stock	int

Наведена вище таблиця 3.5 відповідає наступному запису:

```
TABLE Tovar (
ID INT UNSIGNED NOT NULL AUTO_INCREMENT,
Name VARCHAR(20) NOT NULL,
Price FLOAT,
Quantity INT,
Stock INT,
PRIMARY KEY (ID)
);
```

Для безумовної ідентифікації при відображенні таблиці у клас назви полів таблиці при перетворенні у властивості класу трансформуються за наступним принципом:

Property_of_class = назва_таблиці + _ + назва_поля

Звідси, клас, який є відображенням таблиці Tovar має наступний вигляд (відповідно до нотацій об’єктно-орієнтованого програмування):

```
class Tovar {
private int tovar_id;
private String tovar_name;
private double tovar_price;
```

```
private int tovar_quantity;
private int tovar_stock;
public String.getTovar_name ()
{    return tovar_name;
}  }
```

В даному класі визначено метод String.getTovar_name (), який повертає назву продукту. Також, бачимо, що трансформовано не тільки назви полів таблиці, а і їх типи. Це досить важливе завдання при аналізі схеми бази даних. Різні системи можуть використовувати абсолютно різні реляційні СУБД, які в свою чергу можуть використовувати різноманітну кількість типів даних для збережених ресурсів. При отриманні структури бази даних, необхідно також діставати інформацію про типи даних, що зберігаються в її елементах, і описувати їх засобами RDF. У зв'язку з тим, що використовувана для моделювання структура RDF заснована спочатку на мові розмітки XML, онтологічні властивості, описані таким чином, можуть мати різні XSD типи даних.

Результат після виконання першого кроку алгоритму - це RDF-документ, що містить твердження, що описують структурну інформацію зі схеми бази даних системи. Нижче показано приклад такого RDF-документу, який матиме один клас та відповідатиме таблиці Tovar, що описана вище.

В прикладі показано, як поля таблиці Tovar трансформуються в властивості owl: DatatypeProperty з XSD типами даних. Кожна властивість пов'язана з класом sys:tovar через властивість rdfs:domain.

На другому кроці алгоритму використовується процедура автоматичного визначення спільних рис елементів у структурі інтегрованих схем і виявлення зв'язків між ними. Основну мету, яку потрібно досягнути на другому кроці - це збільшити кількість семантичних зв'язків між онтологіями різних систем в рамках загальної глобальної онтології.

Приклад RDF-документу:

```
<sys:tovar>
a owl:Class;
rdfs:label "tovar"^^xsd:string.
<sys:tovar_id> a owl:DatatypeProperty;
rdfs:domain <sys:tovar>;
rdfs:label "ID"^^xsd:int.
<sys:tovar_name> a owl:DatatypeProperty;
rdfs:domain <sys:tovar>;
rdfs:label "Name"^^xsd:string.
<sys:tovar_price> a owl:DatatypeProperty;
rdfs:domain <sys:tovar>;
rdfs:label "Price"^^xsd:double.
<sys:tovar_quantity> a owl:DatatypeProperty;
rdfs:domain <sys:tovar>;
rdfs:label "Quantity"^^xsd:int.
<sys:tovar_stock> a owl:DatatypeProperty;
rdfs:domain <sys:tovar>;
rdfs:label "TovarStock"^^xsd:string.
```

Іншими словами, проаналізувавши автоматично всі структурні елементи в кожній системі, можна визначити подібні елементи і додати до них певну семантичну властивість, що дасть змогу зв'язати їх між собою. Отже, на другому кроці потребують вирішення два завдання:

1) визначення спільних рис елементів у структурі інтегрованих схем;

2) додання зв’язків між подібними елементами.

Вирішення першого завдання вже розглянуте нами у другому розділі дисертаційної роботи. Для вирішення другого завдання пропонується використовувати додаткові семантичні властивості, що реалізують підходи «один до одного» і «один до багатьох». В роботі [133] розглядаються деякі анотаційні властивості, які позначають класи та властивості в загальній онтології і пов'язують їх з іншими об'єктами в моделі.

Було виділено такі варіанти можливих семантичних властивостей між елементами онтології:

- Властивість узагальнення: $s_1 : P(x_1) = x_2$, $x_2 = \{x_{2i}\}$, якщо $b_1 < k(x_1, x_{2i}) < b_2$, де $P(x_1)$ – відображення концепту x_1; b_2 – максимальне значення коефіцієнта семантичної подібності $k(x_1, x_{2i})$, при якому будується відображення концепту x_1 в онтологію O_2; b_1 – мінімальне значення подібності для встановлення відсутності еквівалентності концептів.

- Властивість еквівалентності: $s_2 : P(x_1) = x_2$, якщо $k(x_1, x_{2i}) \geq b_2$, де b_2 – максимальне значення коефіцієнта семантичної подібності $k(x_1, x_{2i})$, при якому будується відображення концепту x_1 в онтологію O_2.

- Властивість часткової еквівалентності: $s_3 : P(x_1) = x_2$, якщо $b_1 < k(x_1, x_2) < b_2$, b_2 – максимальне значення коефіцієнта семантичної подібності $k(x_1, x_2)$, при якому будується відображення концепту x_1 в онтологію O_2; b_1 – мінімальне значення подібності для встановлення відсутності еквівалентності концептів.

- Властивість уточнення: $s_4 : P(x_1) = x_2$, $x_2 = \{x_{1i}\}$, якщо $b_1 < k(x_{1i}, x_2) < b_2$, де $P(x_1)$ – відображення концепту x_1; b_2 – максимальне значення коефіцієнта семантичної подібності $k(x_{1i}, x_2)$, при якому будується відображення концепту x_1 в онтологію O_2; b_1 – мінімальне значення подібності для встановлення відсутності еквівалентності концептів.

- Властивість відмінності: $s_5 : P(x_1) = 0$, $\exists x_1, \forall x_2 \in O_2$, $k(x_1, x_2) \leq b_1$, b_1 – мінімальне значення подібності для встановлення відсутності еквівалентності концептів.

На третьому кроці відбувається поповнення одержаної онтології додатковими онтологіями верхнього рівня і онтологією предметної області. Таке поповнення здійснюємо, використовуючи команду owl: import, що відповідає за імпортування понять і відношень із зовнішніх онтологій. На даний момент існує велика кількість онтологій різних предметних областей, які містять різноманітні множини понять і зв'язків між ними. Так, наприклад, існують онтології опису відношень людей, онтології для опису бібліографічних

документів, онтології організації і т.д. Імпортувавши такі властивості в отриману структурну онтологічну модель, можна отримати інформацію не тільки про системи всередині своєї предметної області, а й за її межами. Таким чином, можна отримати доступ до ще більших можливостей для автоматизованого логічного прийняття рішень, завдяки великій кількості інформаційних об'єктів всередині онтологічної моделі.

Четвертий крок передбачає виконання перевірки створеної онтології на відсутність зв’язків між об'єктами онтології. Адже при виконанні третього кроку при автоматизованому поповненні створеної онтології не завжди можуть бути знайдені всі закономірності між елементами системи і деякі зв’язки могли бути пропущені та не встановлені. Тому виконується перевірка, і якщо все гаразд, тоді переходимо зразу до кроку шостого, якщо ні – до кроку п’ятого.

П’ятий крок алгоритму передбачає додаткове ручне встановлення зв'язків між об'єктами онтології. Зв'язки можуть встановлюватися як між об'єктами онтологій систем, що інтегруються, так і між імпортованими онтологіями верхніх рівнів і онтологією предметної області. Для реалізації цього кроку необхідно скористатися програмними засобами редагування онтологій, такими як Protégé. На даному кроці ми додаємо необхідні зв'язки вручну, щоб створити повноцінну семантичну модель структури інтегрованих систем.

І, нарешті, останній шостий крок полягає у виведенні загальної онтології структури у форматі RDF. Результат може бути записаний у файл або в спеціалізоване сховище RDF даних. Отримана онтологічна модель, по суті, є ні чим іншим як понятійною частиною онтології. У ній містяться поняття і відношення між ними в системах, що інтегруються в рамках предметної області, в якій вони працюють, а також множини термінів з інших областей. На основі цієї загальної, глобальної моделі структури можна отримати метадані самих інформаційних ресурсів з систем. За допомогою такого підходу можна

автоматизовано описати семантику ресурсів ще на самому ранньому етапі їх отримання.

3.3.2 Алгоритм отримання інформаційних ресурсів із системи, що інтегрується

Відповідно до розробленого у другому розділі методу опису структури і змісту вхідних даних для процедури класифікації вхідних даних відповідно до предметної області, після того, як отримано глобальну онтологічну мета-модель, можна на основі даної моделі виділити у вхідній інформаційній системі потрібні метадані самих ресурсів, що зберігаються в системі. Для вирішення даного завдання було розроблено, відповідно, алгоритм отримання IP із системи, що інтегрується.

Алгоритм отримання інформаційних ресурсів із системи, що інтегрується передбачає використання, раніше отриманої, глобальної онтологічної метамоделі систем, що інтегруються, що включає онтології верхнього рівня і онтологію предметної області.

Вхідними даними алгоритму є:

- інформація про збережені у базах даних ресурси;
- глобальна онтологічна метамодель.

Вихідними даними, як результатом роботи алгоритму, є метамодель, що об'єднує в собі понятійну і змістовну частини онтології. Таким чином, в такій онтології об'єднується як інформація про структуру інтегрованих систем, так і метадані збережених в них об'єктів, описані термінами з понятійної частини. Отримана метамодель може бути використана, як єдиний інтерфейс для семантичної інтеграції даних розподілених систем та забезпечення їх інтероперабельності.

Для детального опису роботи алгоритму отримання інформаційних ресурсів із системи, що інтегрується скористаємося введеними раніше деякими позначеннями понять.

Нехай ми маємо деяку схему бази даних системи: *S* (3.12) і деяку таблицю T_i:

$$T_i = \{K_1,..., K_r\}, \qquad i = \overline{1,n}, \tag{3.16}$$

де: $K_1,..., K_r$ - кортежі таблиці T_i.

Робота алгоритму отримання інформаційних ресурсів із системи, що інтегрується складається із наступних кроків:

1) Отримання інформації про збережені інформаційні ресурси із системи, що інтегрується, використовуючи глобальну онтологічну метамодель. Тобто отримуємо кожен кортеж із кожної таблиці певної схеми бази даних.

2) Визначення спільних рис отриманих кортежів таблиць інформаційних ресурсів і додання семантичних зв'язків між ними.

3) Додавання семантичних властивостей, використовуючи семантичний аналізатор, який працює на основі дескриптових логік [20, 117] та імпортованій онтології.

Результатом є онтологія метаданих інформаційних ресурсів і структури системи, що зберігається у файл чи сховище метаданих у форматі RDF.

Зобразимо діаграму діяльності алгоритму отримання інформаційних ресурсів із системи, що інтегрується на рисунку 3.5.

Розглянемо детальніше всі кроки представленого вище алгоритму. На першому кроці відбувається отримання інформаційних ресурсів з бази даних, використовуючи глобальну онтологію структури систем, інтегруються.

$$DI_2^* = \langle X, Att, R, f_{Att}(X), f_W(X), W \rangle, \tag{3.17}$$

де: $X = \{x_i \mid i = \overline{1, N_X}\}$ - множина концептів; $Att = \{att_{ij} \mid i = \overline{1, N_{Att}}, j = \overline{1, M_{Att}}\}$ - множина атрибутів концептів; $R = \{r_i \mid i = \overline{1, N_R}\}$ – множина відношень між концептами; $W = \{w_{ij} \mid i = \overline{1, N_W}, j = \overline{1, M_W}\}$ - словник, в якому визначаються

терміни предметної області; $f_W(X)$ - функція інтерпретації концептів, зіставляє концепту набір термінів зі словника *W*.

Рисунок 3.5 - Діаграма діяльності алгоритму отримання інформаційних ресурсів із системи, що інтегрується

Дана онтологічна модель є своєрідним базисом для створення метамоделі інформаційних ресурсів систем, що інтегруються, а, також, надає множину понять і відношень, використовуючи які можна створити семантичні метаописи інформаційних ресурсів. Отже, всі збережені дані в таблицях баз даних витягуються через SQL-запит і записуються в онтологічну модель RDF-засобами. Звідси, кожен кортеж таблиці стає екземпляром класу онтології і має відповідні їй властивості. Екземпляр класу це і є самі дані, які необхідно інтегрувати. Разом з понятійною частиною онтології, екземпляри класів утворюють повну базу знань. Таким чином встановлюється зв'язок між структурою систем, що інтегруються і збереженими в них інформаційними ресурсами. Таким же чином XSD типи даних були успадковані екземплярами від об'єктів структури.

На другому кроці відбувається визначення спільних рис отриманих інформаційних ресурсів для додання ще більших семантичних зв'язків. На даному кроці, аналогічно до другого кроку алгоритму побудови онтологічної моделі всіх систем, що інтегруються, потребують вирішення два завдання:

1) визначення спільних рис отриманих інформаційних ресурсів;

2) додання зв’язків між подібними елементами.

Вирішення першого завдання описано в другій частині третього розділу дисертації. Щодо вирішення другого завдання – потрібно додати семантичні властивості, що зв'язують знайдені схожі елементи. Враховуючи той фактор, що подібність елементів між собою може бути відносною і не стовідсотковою - семантичних властивостей може знадобитися кілька. Для таких випадків можна скористатися командою owl: sameAs. Але, як зазначається у [95], дана команда не допоможе отримати результат на скільки елементи кортежів схожі один на одного. Для вираження ступеня схожості пропонується використовувати властивості зі словника верхнього рівня SKOS: skos: narrowMatch; skos: closeMatch; skos: exactMatch [95]. Цих властивостей достатньо для вираження різного ступеня схожості елементів, і саме їх доцільно додавати в онтологічну модель при виявленні подібних записів в інтегрованих схемах баз даних.

На третьому кроці алгоритму відбувається автоматизоване додавання семантичних властивостей за допомогою, так званого, семантичного аналізатора. Суть роботи якого полягає у наступному: нехай ми маємо деякий клас «Object1», який в понятійній частині онтології є підкласом класу «Object2», що володіє деякою властивістю «Property1OfObject2».

У змістовній частині онтології, шляхом вилучення кортежу із схеми бази даних, створюється екземпляр класу «Object1». Далі автоматично визначається, за допомогою використання правил семантичного аналізу та дескриптової логіки, те, що екземпляр класу «Object1» також є екземпляром класу «Object2» і володіє властивістю «Property1OfObject2». Таким чином до даних отриманого

кортежу додаються нові семантичні зв'язки. Дана дія виконується завдяки правилам транзитивності, використовуваним в глобальній онтології і дескриптовій логіці, що описує модель. Згідно використовуваних правил дескриптової логіки, основні логічні проблеми є розв'язуваними і мають відносно невисоку складність обчислювання, тому є можливим автоматизувати дану роботу.

Наведемо декілька правил, які лягли в основу роботи алгоритму на даному кроці.

Правило 1. Поняття A виконується в моделі M, якщо $A \neq 0$.

Правило 2. Поняття A виконується в моделі M, якщо існує деяка інтерпретація, в якій воно виконується.

Правило 3. Поняття A входить в поняття B, якщо у будь-якій моделі M виконується наступне: $A^M \subseteq B^M$.

Правило 4. Об'єкт a є екземпляром поняття A в онтології O, якщо в будь-якій моделі M виконується наступне твердження: $a^M \in A^M$.

Завдяки наведеним правилам і метаданим, що зберігаються в глобальній онтології, механізми автоматичного аналізу аналізують отримані метадані інформаційних ресурсів і додають нові семантичні зв'язки.

Результатом роботи алгоритму є повністю сформована онтологічна модель із семантичними метаописами інформаційних ресурсів систем, що інтегруються.

3.4 Метод формування об'єднаного динамічного набору даних, який має узгоджену структуру і єдиний зміст

Процес формування вихідних даних забезпечує, відповідно до запиту користувача, динамічне відображення структурованих вхідних даних di_i^* у набір вихідних даних із узгодженим змістом даних do_j.

$$User(q_d) \to q_d \to Q \to C_u(di_i^*, q_d) \to \beta(q_d, di_i^*, t_p, c_u) \to do_j \to DO \to User(DO), \quad (3.18)$$

де: $User(q_d)$ - формування запиту користувачем; $User(DO)$ - перегляд відповіді на запит.

Для підвищення якості результату процесу формування вихідних даних $\beta:(DI^*, Q) \to DO$ було розроблено метод формування об'єднаного динамічного набору даних, який має узгоджену структуру і єдиний зміст.

Метод формування об'єднаного динамічного набору даних, який має узгоджену структуру і єдиний зміст - комплекс заходів забезпечення підвищення якості результату відображення структурованих даних із врахуванням їх змісту, отриманих з різних інформаційних джерел для їх подальшого візуального подання у системі динамічної інтеграції слабоструктурованих даних. Метод подано, як послідовність кроків:

Крок 1. Виділення метаописів інформаційних ресурсів із метамоделі системи, що інтегрується.

Даний крок полягає у отриманні та деталізації метаописів інформаційних ресурсів певної визначеної кількості для кожного джерела.

$$DI_5^* = \beta_1(DI_4^*, C_{1u}, T), \quad (3.19)$$

де: β_1 і C_{1u} - оператор і множина умов виділення метаописів ІР.

Не потрібно розглядати абсолютно всі метаописи, адже їх семантичні зв'язки вже зазначено у метамоделі системи. І знайшовши, при порівнянні із запитом користувача, у виділеній кількості метаописів відповідні запиту – буде знайдено, також, всі інші семантично пов'язані метаописи із даними.

Нехай інформаційний ресурс описується, як інформаційний об'єкт згідно (2.21), тоді

$$DI_5^* = DI_4^* \setminus \langle URI, IR, TR, FM, El, P_{RM}, P_R, C, FC \rangle. \quad (3.20)$$

Крок 2. Виділення пошукового образу запиту.

На цьому кроці методу окремо виділяються терміни із пошукового запиту та семантично подібні терміни до термінів пошукового запиту. Останні отримуються за допомогою використання загальної онтології предметної області, створеної раніше. Наступним етапом є розставлення асоціативних зв'язків між термінами пошукового запиту та розставлення асоціативних зв'язків між семантично подібними термінами до термінів пошукового запиту.

$$Q_0 = \beta_2(Q, C_{2u}, T), \tag{3.21}$$

де: C_{2u} - множина умов виділення пошукового образу запиту.

Крок 3. Знаходження подібності запиту із метаописами.

Цей крок полягає у виконанні процедури порівняння запиту із виділеними метаописами. При порівнянні знаходяться подібні дані запиту. Умовою знаходження результату є певне обмеження кількості всіх даних у загальній Mashup моделі результуючих даних, що забезпечує високу релевантність результуючих даних та зниження, таким чином, інформаційного шуму. Дані обмеження на кількість результуючих даних введені для кожного джерела інформаційних ресурсів можуть бути різними – в залежності від джерела. Відповідну обмеженням кількість інформаційних ресурсів із кожного джерела дістається ще за допомогою додання подібних даних із метамоделі до знайдених даних відповідно запиту користувача.

$$DO_0 = \beta_3(\beta_2(Q, C_{2u}, T), DI_5^*, C_{3u}, T), \tag{3.22}$$

де: C_{3u} - множина умов знаходження подібності запиту із метаописами.

Розглядається питання пошуку мінімального порогу b семантичної близькості, при якій елемент запиту користувача і елемент семантичних метаданих приймаються еквівалентними.

$$b_1 = \max(k(el_i, el_j) \mid \forall el_i \in Q_0, \forall el_j \in DI_{k5}^*) \times (p_1/100), \tag{3.23}$$

де: Q_0 – запит користувача; DI^*_{5k} - метаописи інформаційних ресурсів; p_1 - відсоток, при якому b_1 приймається порогом подібності для встановлення еквівалентності та коректного відображення el_i і el_j.

Показано, що b_1 - мінімальний поріг, при якому зменшення цього значення призводить до неможливості повного відображення елементів порівняння. Знаходиться граничне значення, при якому елементи приймаються частково еквівалентними.

$$b_2 = \max(k(el_i, el_j) \mid \forall el_i \in Q_0, \forall el_j \in DI^*_{k5}) \times (p_2 / 100), \qquad (3.24)$$

де: p_2 - відсоток, при якому b_2 приймається порогом подібності для встановлення часткової еквівалентності елементів.

Показано, що b_2 - мінімальне значення в тому сенсі, що зменшення цього значення призводить до некоректного відображення елементів структур порівняння. Елементи вважаються різними, якщо мають значення міри семантичної подібності такої, що не перевершує поріг b_2.

Компонента системи, яка відповідає за аналіз елементів на подібність і отримання метаданих працює згідно застосування процедур лінгвістичного аналізу змісту елементів порівняння. Порівняння відбувається за кількома ознаками з урахуванням їх ваги. Найбільш висока вага у функції порівняння рядків. За замовчуванням ця функція працює на основі модифікованого лінгвістичного коефіцієнта Серенсена-Дайса з поділом рядків на N-грами. Довжину N-грами експериментально обрано рівну трьом символам.

Нехай X – запит користувача, який розглядається як рядок символів і Y – це метаопис ІР, який порівнюється із запитом. Тоді коефіцієнт подібності рядків обчислюється за допомогою модифікованої формули обчислення коефіцієнта Серенсена-Дайса, застосувавши N-грами замість символів:

$$k_p = \frac{2 \times f_{NG}(X \cap Y)}{f_{NG}(X) + f_{NG}(Y)}, \qquad (3.25)$$

де: k_p – коефіцієнт подібності рядків, $0 \le k_p \le 1$; X і Y – рядки, що порівнюються; $f_{NG}(X)$ – функція, що визначає довжину ланцюжка *N*-грам рядку *X*.

Однак у даній компоненті реалізовані, також, і інші лінгвістичні алгоритми порівняння рядків, зокрема коефіцієнт Танімото і відстань Дамерау-Левенштейна. Іншою функцією порівняння є порівняння за типами даних. Для її реалізації були створені таблиці відповідностей типів даних різних СУБД і XSD типів даних. Аналізатор порівнює вже не типи даних в СУБД, а відповідні їм типи XSD. Далі лінійно підраховується загальний коефіцієнт відповідності елемента або запису по формулі:

$$Sim(S_1, S_2) = \{p_{u_1} w_{u_1} + p_{u_2} w_{u_2} +,...,+ p_{u_9} w_{u_9}\}, \quad (3.26)$$

де: p - коефіцієнт відповідності з окремою ознакою; w - вага кожної ознаки.

У підсумку виходить значення в межах від 0 до 1. Відповідно до таблиці 3.6 проводиться додавання необхідної семантичної властивості з онтології верхнього рівня skos. Всі значення були виведені вручну, експериментальним шляхом.

Таблиця 3.6 - Зіставлення коефіцієнтів відповідності і семантичних властивостей

Коефіцієнт відповідності	Семантична властивість
$0.6 \le p < 0.75$	*skos:narrowMatch*
$0.75 \le p < 0.9$	*skos:closeMatch*
$0.9 \le p \le 1$	*skos:exactMatch*

Крок 4. Узгодження результату.

$$DO_1 = \beta_4(DO_0, C_{4u}, T), \tag{3.27}$$

де: C_{4u} - множина умов узгодження результату.

Узгодження результату полягає у ранжуванні результуючих даних відповідно до зв'язків подібності. Тобто, чим вищий коефіцієнт подібності одних даних результату іншим, тим ближче ці дані будуть знаходитися один біля одного у результуючому Mashup. Ступінь впорядкованості елементів розглядається використовуючи величину, що відображає число перестановок пари сусідніх елементів до впорядкованого положення від положення ранжування по спаданню релевантності метаописів до запиту, як:

$$d_{\max} = kmq_2(kmq_1 + kmq_0) + kmq_1 kmq_0 + ... + kmq_m(n - kmq_m) + kmq_{m-1}(n - kmq_m - kmq_{m-1}), \tag{3.28}$$

де: $d_{\max}$ – ступінь впорядкованості; $kmq_i, i = \overline{1,m}$ – коефіцієнт подібності метаопису до запиту; n_1 – номер першого релевантного метаопису.

Коефіцієнт узгодження результату має вигляд:

$$k_u = 1 - \frac{d}{\sum_{i,j=0,i>j}^{m} kmq_i kmq_j}. \tag{3.29}$$

Результатом методу є виведення сформованого контенту об'єднаного динамічного набору даних.

3.5 Висновки до розділу

1. Виявлено та проаналізовано, з точки зору розробника, ряд факторів, що впливають на процес інтеграції інформаційних ресурсів в Mashup системі, що дало змогу визначити принципи побудови об'єднаного динамічного набору даних, що має загальну структуру і єдиний зміст.

2. Охарактеризовано визначення подібності схем даних та інформаційних ресурсів систем, що інтегруються, використовуючи обчислення лінгвістичних метрик подібності для того, щоб визначити спільні риси інформаційних ресурсів інтегрованих систем і виявити зв'язки між ними.

3. Розроблено алгоритм побудови онтологічної моделі всіх систем, що інтегруються, яка містить поняття і відношення між ними в системах, що інтегруються в рамках предметної області, в якій вони працюють, а також множини термінів з інших областей. На основі цієї загальної, глобальної моделі структури можна отримати метадані самих інформаційних ресурсів з систем. За допомогою такого підходу можна автоматизовано описати семантику ресурсів ще на самому ранньому етапі їх отримання.

4. Розроблено алгоритм отримання інформаційних ресурсів із системи, що інтегрується, що дало змогу вдосконалити процес створення загальної динамічної структури для множини вхідних інформаційних ресурсів із врахуванням їх змісту.

5. Розроблено метод формування об'єднаного динамічного набору даних, що відрізняється від існуючих застосуванням процедур лінгвістичного аналізу змісту запиту користувача, що забезпечило підвищення релевантності та узгодженості результатів динамічної інтеграції слабоструктурованих даних.

РОЗДІЛ 4 РОЗРОБЛЕННЯ ТА ВПРОВАДЖЕННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ДИНАМІЧНОЇ ІНТЕГРАЦІЇ СЛАБОСТРУКТУРОВАНИХ ДАНИХ

4.1 Проект розроблення системи динамічної інтеграції слабоструктурованих даних

4.1.1 Характеристики проекту розроблення системи

Для реалізації інформаційної технології динамічної інтеграції слабоструктурованих даних у web-системах було спроектовано застосунки, які призначені для динамічної інтеграції інформаційних ресурсів web-систем і подання їх у вигляді пов'язаних даних, відповідно до запиту користувача. В даних застосунках відбувається автоматизоване структурування даних, що містяться в інтегрованих інформаційних web-системах, в онтологічну модель, яка далі використовується для знаходження даних, відповідно до користувацького запиту.

Основними етапами розроблення будь-якої програмної системи є: постановка завдання, де визначаються високорівневі параметри (функціональні і експлуатаційні вимоги, інтерфейс, вимоги до надійності і безпеки); розробка архітектури продукту і вимог до неї; кодування, при якому на основі архітектури та вимог до неї виходить вихідний код системи; отримання виконуваного коду шляхом компіляції і завантаження вихідного коду.

Систему динамічної інтеграції слабоструктурованих даних у web-системах реалізовано у вигляді набору завершених модулів, які можна використати для побудови інших систем.

Згідно ISO TS 18876 формально архітектура системи інтеграції даних виглядає відповідно до рисунку 4.1.

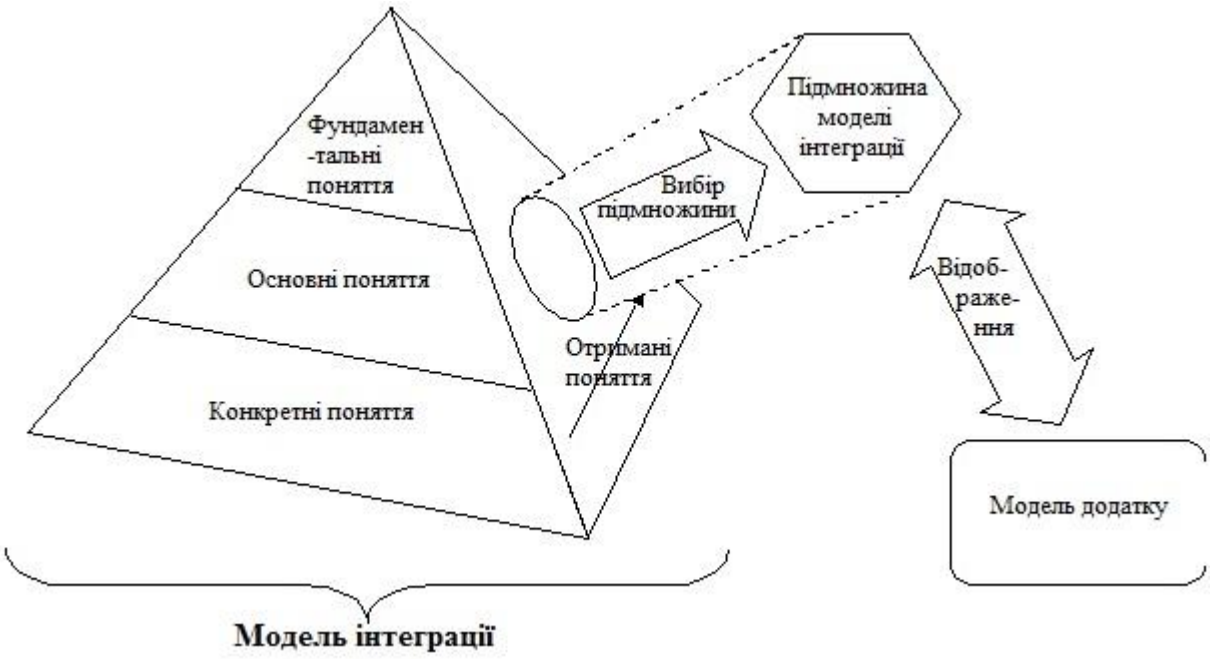


Рисунок 4.1 - Архітектура інтеграції даних (ISO TS 18876)

Підхід до проектування, що було визначено для системи динамічної інтеграції слабоструктурованих даних у web-системах зображено на рисунку 4.2.

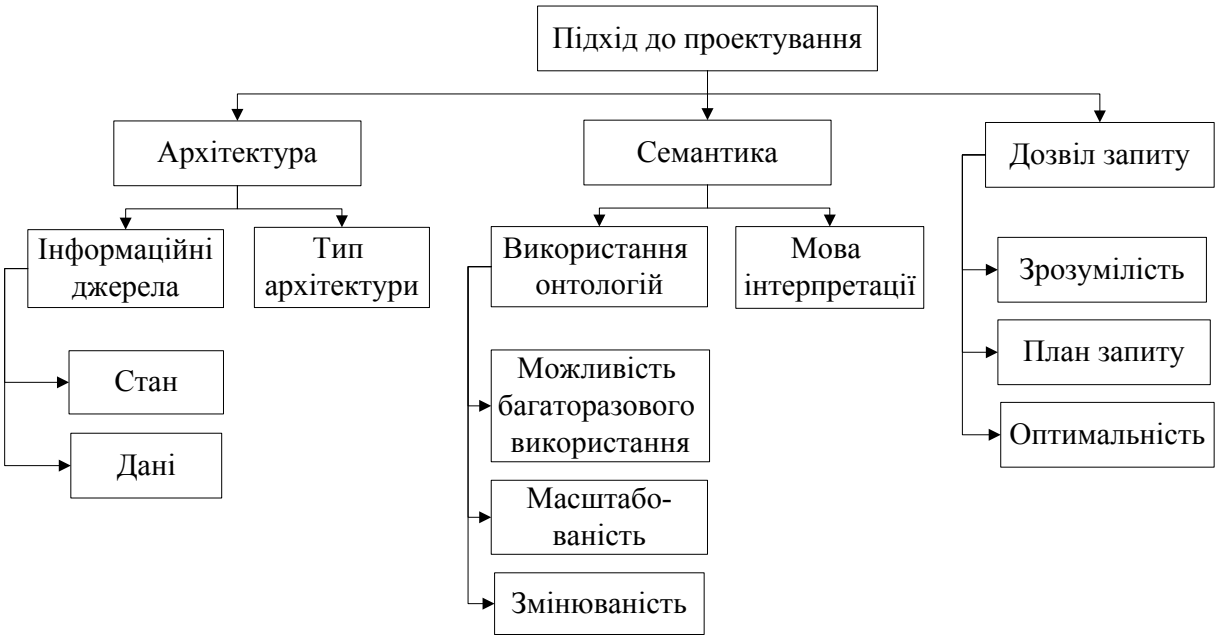


Рисунок 4.2 - Підхід до проектування системи динамічної інтеграції слабоструктурованих даних

Структурна модель системи динамічної інтеграції слабоструктурованих даних включає (рис. 4.3):

- Адаптери для роботи з інформаційними джерелами;
- Модуль інформації про систему;
- Модуль даних про інформаційні джерела;
- Підсистема формування завдання;
- Модуль отримання доступу до інформаційних джерел;
- Підсистема опису структури і змісту вхідних даних;
- Підсистема формування об’єднаного динамічного набору даних;
- Модуль зберігання даних;
- Модуль підтримки роботи системи;
- Підсистема візуалізації результатів роботи системи.

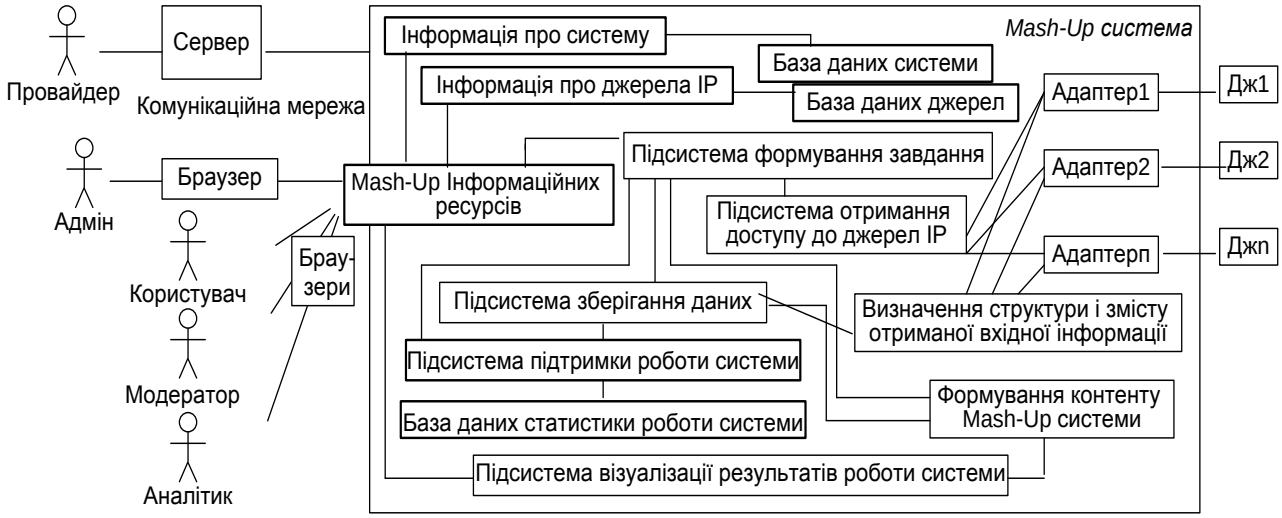


Рисунок 4.3 - Структурна модель системи динамічної інтеграції слабоструктурованих даних, що працює на основі Mashup

Згідно рис. 4.3 для роботи із кожним джерелом ресурсів потрібно використовувати таку компоненту, як об’єктний адаптер. На рис. 4.4 зображено об’єктний адаптер, що складається із:

- модуль опису ресурсу - застосовується для генерування структури даних ресурсу;

- модуль відображень - застосовується для отримання правил відображення;

- модуль перехоплень – застосовується для перехоплення операцій системи згідно ресурсних та картографічних даних, отримання доступу до нового ресурсу та для повернення результатів відображень до системи.

Рисунок 4.4 - Об’єктний адаптер Mashup системи динамічної інтеграції слабоструктурованих даних

Модуль опису структури і змісту вхідних даних та модуль формування об’єднаного динамічного набору вихідних даних спроектовано на основі розроблених методу опису структури і змісту вхідних даних та методу

формування об’єднаного динамічного набору даних, який має узгоджену структуру і єдиний зміст.

Підсистема опису структури та змісту вхідних даних забезпечує:

- експорт метаданих структури (підключення до інформаційної системи, отримання структурної інформації, аналіз на подібність елементів структури, додавання семантичних зв’язків на основі аналізу, експорт онтології);
- розширення онтології структури (додавання онтологій верхнього рівня, додавання онтологій предметної області, додавання семантичних зв'язків, експорт розширеної онтології);
- отримання метаданих інформаційних ресурсів (імпорт розширеної онтології структури, підключення до інформаційної системи, витяг метаданих, аналіз на схожість, додавання семантичних зв'язків на основі аналізу, додавання семантичних зв'язків на основі онтології структури, експорт онтології в сховище).

Підсистему опису структури та змісту вхідних даних реалізовано у вигляді наступних основних компонентів:

- модуль формування шаблону запиту на використання даних із джерел інформаційних ресурсів, затребуваних користувачем;
- модуль створення і зміни онтологічних описів, який відповідає за створення і заміну понять і ієрархій в базовій онтології;
- репозиторій метаданих, що забезпечує зберігання описів онтологій предметних областей;
- модуль зіставлення і інтеграції онтологій, що виконує об’єднання онтологій різних предметних областей;
- модуль вирішення конфліктів, що можуть виникнути при зіставленні концептів онтологій;

- модуль, що відповідає за формування моделі кожного вхідного інформаційного ресурсу загальної структури, визначеної предметною областю;
- додаткові компоненти (адаптери, конфігуратори, конектори).

Підсистема формування контенту Mashup забезпечує:

- аналіз семантичних метаданих вхідних інформаційних ресурсів (імпорт отриманих семантичних метаописів ресурсів, їх деталізація, виділення зазначеної кількості, експорт даних у сховище);

- визначення подібності запиту користувача із семантичними метаописами інформаційних ресурсів (імпорт метаописів інформаційних ресурсів, виділення пошукового образу запиту, аналіз на подібність запиту користувача та метаописів, додання подібних метаописів до знайдених, експорт даних у сховище);

- узгодження результату (отримання зі сховища даних, які відображають схожі до запиту користувача метаописи, ранжування результуючих даних відповідно до зв’язків подібності, запуск сервера точки доступу, відображення отриманих даних).

Підсистему формування контенту Mashup реалізовано у вигляді таких основних компонентів:

- модуль опрацювання запиту користувача, який реагує на некоректність вводу даних для запиту та відповідає за імпорт метаописів інформаційних ресурсів із репозиторію метаданих, виділення пошукового образу запиту, аналіз на подібність запиту користувача та метаописів, додання подібних метаописів до знайдених, експорт даних у сховище;

- модуль редагування даних, який дає можливість врахувати внесені зміни у запит користувача через постійний зв’язок із модулем опрацювання запиту;

- модуль відображення даних, який відповідає за ранжування результуючих даних відповідно до зв’язків подібності між ними та новизною їх опублікування у джерелі;
- додаткові компоненти (адаптери, конфігуратори, конектори).

4.1.2 Вхідні, вихідні дані

Робота програмної системи, що створена на основі інформаційної технології динамічної інтеграції слабоструктурованих даних у web-системах, полягає у представленні об’єднаного динамічного набору даних, отриманого із різних web-систем, відповідно до запиту користувача. Система автоматизовано виконує відображення інформації, що зберігається в гетерогенних, інформаційних web-системах, що інтегруються, в онтологічну модель, яка пізніше використовується для пошуку інформаційних ресурсів, згідно запиту користувача.

Вхідні дані - це запит користувача, інформація про джерела інформаційних ресурсів та самі інформаційні ресурси.

Вихідні дані – це множина метаописів інформаційних ресурсів із можливістю перейти до джерела ресурсу.

4.1.3 Обґрунтування розроблення та впровадження систем динамічної інтеграції слабоструктурованих даних

Проектування систем динамічної інтеграції слабоструктурованих даних з використанням технології Mashup вимагає врахування досить великої кількості різноманітних факторів, до яких, наприклад, можна віднести: організація задання можливих запитів користувача, динамічність отримання колажів даних, формування відповіді на запит користувача, визначення спільних рис інформаційних ресурсів, особливості систем, що інтегруються, тощо. Відомими розробниками систем інтеграції даних на основі технології Mashup можна назвати Microsoft, Google, Yahoo, тощо.

Завдання технологій інтеграції даних полягає в подоланні численних проявів неоднорідності, властивої інформаційним системам, які створювалися і створюються, керуючись чим завгодно, але не уніфікованим ставленням до даних. Системи мають різну функціональність, використовують різні типи даних (алфавітно-цифрові і медіа, структуровані і неструктуровані), їх компоненти розрізняються по автономності, мають різну продуктивність. Системи будуються на різних апаратних платформах, мають різні засоби управління даними, різне програмне забезпечення проміжного шару, моделі даних, призначені для користувача інтерфейси і багато іншого.

Інтеграція контенту в Mashup системах, як правило, динамічна, тобто вона відбувається під час виконання, на основі введених даних користувачем. Для досягнення мінімального часу виконання більшість Mashup систем підтримують тільки досить прості види інтеграції даних [55, 64, 65, 94]. Наприклад, вони часто використовують стандартизовані ідентифікатори об'єктів (наприклад, широта / довгота географічних позицій, або унікальні номера продукту, такі як EAN або ISBN) для легкого взаємозв'язку різних джерел або послуг [56, 67, 70]. Запит доступу до джерел даних або пошукових систем є, як правило, на основі ключових слів, тегів або категорій імен, без опрацювання семантики даних, для перегляду результатів з різних джерел [43, 57]. Звідси, можна зробити висновок, що сучасне використання Mashup технології потребує покращеної підтримки для динамічної інтеграції даних в гетерогенних об'єктах даних.

Зважаючи, на активний розвиток Mashup систем та на їх величезні функціональні можливості особливо актуальна задача розробки підходів, технологій, архітектурних рішень і гнучких інструментальних засобів проектування Mashup систем динамічної інтеграції слабоструктурованих даних, та надання рекомендацій опрацювання даних в таких системах.

4.2 Вимоги до системи динамічної інтеграції слабоструктурованих даних

Функціональні вимоги до інформаційної системи описують ту цінність, яку представляє система з точки зору реалізації функцій організації (рис. 4.5). Архітектура більшості автоматизованих застосунків, які забезпечують і реалізують такі функціональні вимоги, включаючи інтерфейси до бізнес-програм і інших прикладним систем описує структуру застосунків і те, як ця структура реалізує функціональні вимоги організації [60].

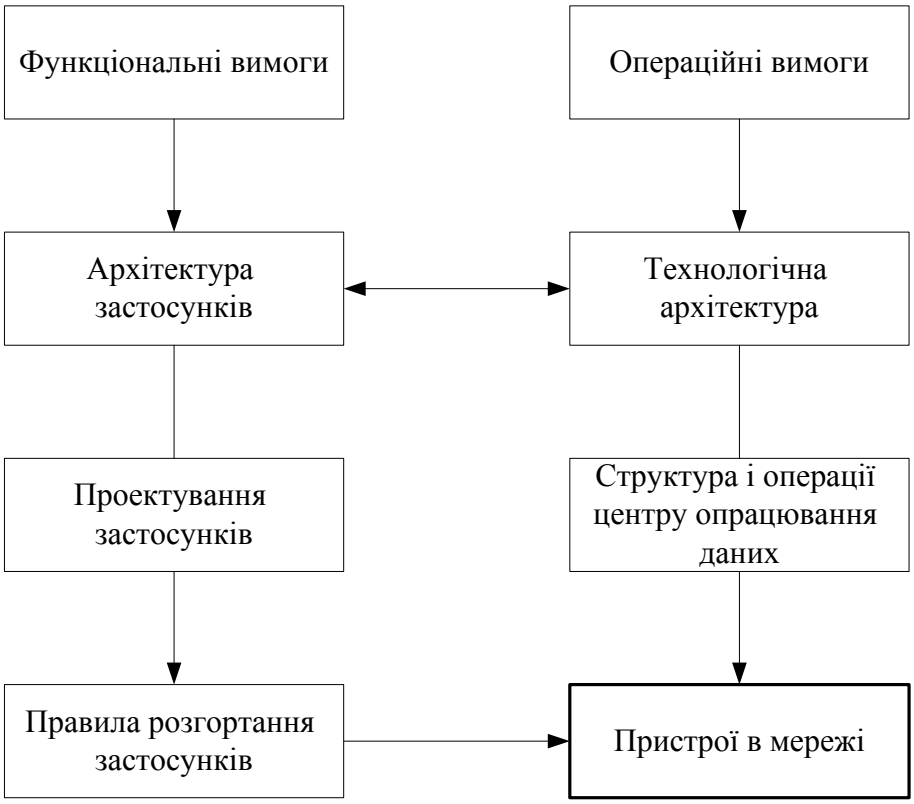


Рисунок 4.5 - Взаємозалежність функціональних та операційних вимог

Операційні (або експлуатаційні) вимоги до системи специфікують такі аспекти, як надійність, керованість, продуктивність, безпека, сумісність. І це далеко не повний список. Прикладами операційних вимог є можливість доступу

до системи тільки авторизованих користувачів, рівень доступності інформаційної системи 99, 99 % часу, тощо.

Технологічна архітектура є архітектурою інфраструктури апаратного та програмного забезпечення, яка забезпечує роботу прикладних систем і виконання операційних (не функціональних) вимог, що подаються до архітектури прикладних систем [63].

Хоча, звичайно, слід зробити одне зауваження. Хороша технологічна архітектура може забезпечувати безпеку, доступність, надійність і цілий список інших операційних вимог, але якщо застосунок спроектовано так, що він не використовує переваг технологічної архітектури, він все одно буде функціонувати не належним чином, і його буде складно впроваджувати і супроводжувати. Аналогічним чином, добре спроектована структура прикладної системи, яка точно відповідає вимогам бізнес-процесів і зібрана з багаторазово використовуваних компонент із застосуванням найсучасніших технологій, може не відповідати реальній конфігурації використовуваного апаратного та системного програмного забезпечення. Це показує, що все-таки є істотна взаємозалежність між архітектурою застосунків і технологічною архітектурою: хороша технологічна архітектура повинна бути побудована з урахуванням підтримки прикладних систем, що вмають важливу роль у роботі організації. Хороша архітектура застосунків повинна ефективно оперувати технологічною архітектурою, щоб забезпечити належний рівень відповідності всім операційним вимогам.

Згідно ISO 15926-7 якщо сервіс забезпечує процедуру обробки та обміну даними, він повинен виконувати наступні завдання:

- приймати потік даних, що архівуються у процесі інформаційного обміну між інформаційними системами та/або з федеральними інформаційними ресурсами в режимі реального часу;

- розархівовувати отримані дані;

- визначати приналежність даних, шляхом аналізу спільних ознак;

- обробляти розархівовані дані формату xml і розміщувати їх в реляційній базі даних web-сервісів;

- логувати інформацію на етапі передачі, розархівування, обробки та збереження;

- забезпечувати фіксацію часу передачі, цілісності та автентичності даних, зазначення їх авторства та можливості надання відомостей, що дозволяють простежити історію руху даних.

При проектуванні системи динамічної інтеграції слабоструктурованих даних у web-системах повинно бути враховано трирівневу організацію опрацювання даних, яку використовують при побудові Mashup систем (рис. 4.6.).

Рисунок 4.6 - Трирівнева організація опрацювання даних у системі динамічної інтеграції даних на основі технології Mashup

Було розроблено функціональні вимоги до системи динамічної інтеграції даних на основі технології Mashup, які відображено за допомогою рисунку 4.7 у вигляді діаграми прецедентів. На розробленій діаграмі можна побачити наступних дійових акторів:

- Модератор (управляє, підлагоджує і підтримує систему);
- Адміністратор (адмініструє побудованою системою);

- Аналітик (його участь необхідна для прийняття аналітичних рішень у системі);

- Користувач (власне для нього і розробляється система, щоб він міг отримати необхідну йому інформацію у вигляді взаємопов'язаного колажу даних на основі заданого пошукового запиту).

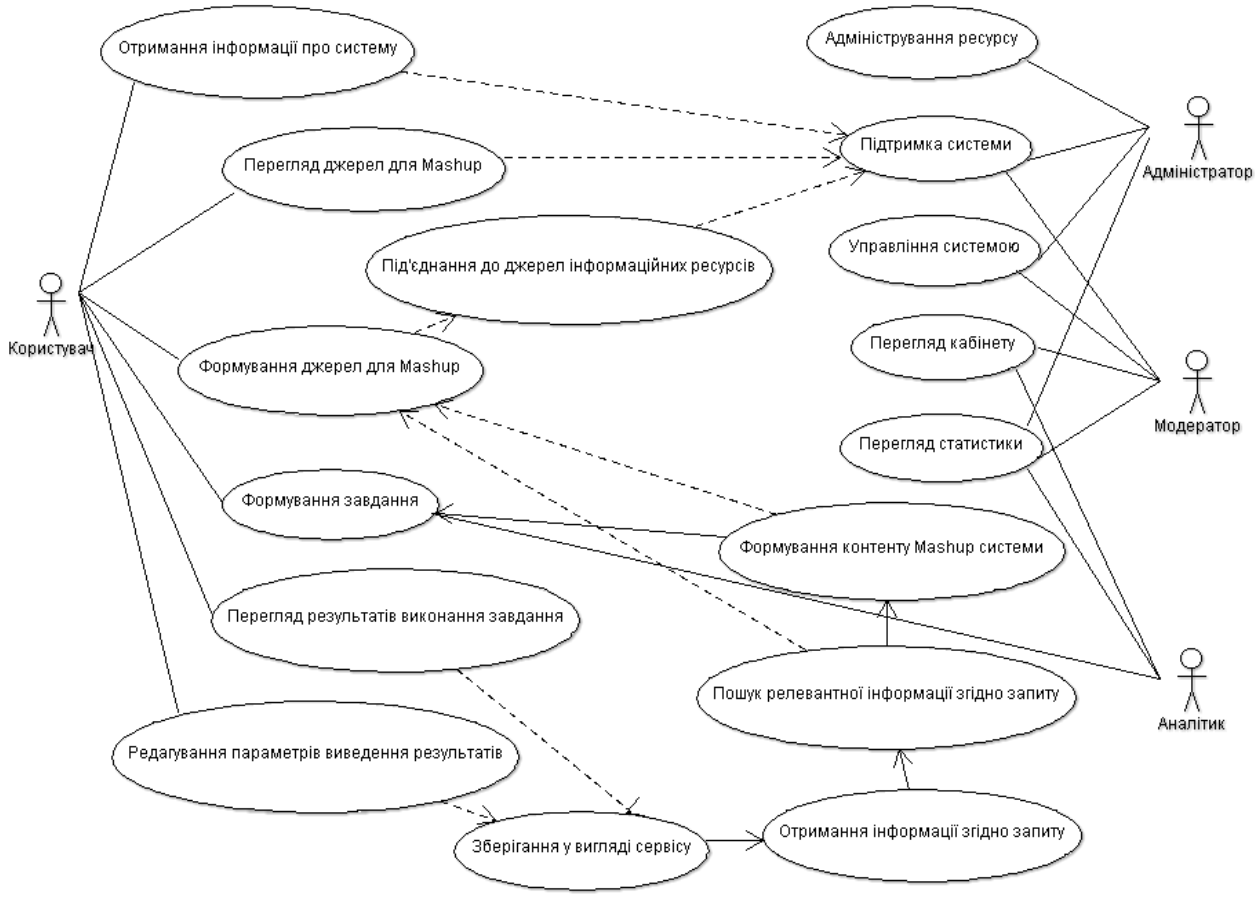


Рисунок 4.7 - Функціональні вимоги до системи динамічної інтеграції даних на основі технології Mashup

Адміністратор може виконувати такі дії:

- адміністрування ресурсу;
- підтримка системи;
- управління системою;
- перегляд статистики.

Модератор може виконувати такі дії:

- підтримка системи;
- управління системою;
- перегляд кабінету;
- перегляд статистики.

Аналітик може виконувати такі дії:

- перегляд кабінету;
- перегляд статистики;
- формування завдання.

Користувач може виконувати такі дії:

- отримання інформації про систему;
- перегляд джерел для Mashup;
- формування джерел для Mashup;
- формування завдання;
- перегляд результатів виконання завдання;
- редагування параметрів виведення результатів.

Щоб досягти правильної узгодженої роботи всіх модулів системи для формування результату роботи системи використано принцип динамічної інтерпретації метамоделі. Метамодель системи – це інформаційна модель більш високого рівня абстракції, ніж конкретна модель предметної області. Метамодель описує і покриває функціоналом не окрему задачу, а широке коло завдань з виділенням в цих завданнях загальних абстракцій, правил обробки даних і управління бізнес-процесами. До необхідної конкретики метамодель адаптується вже в момент виконання, перетворюючи метадані у динамічний код і забезпечуючи їх динамічне зв'язування з середовищем запуску. За допомогою метамоделі у системі інтерпретуються метадані інформаційних ресурсів та динамічно будується модель предметної області, яка вже інтерпретує дані, що надходять. Метадані дають змогу метамоделі скласти

динамічну модель розв'язуваної задачі, тобто, метадані доповнюють опис даних і доповнюють можливості метамоделі по інтерпретації отриманих даних.

Метамодель знаходиться в кожному компоненті інформаційної системи і містить абстрактну функціональність для широкого класу задач. При запуску компоненти метамодель розгортається в прикладній віртуальній машині, але для її динамічної інтерпретації, тобто для перетворення її в модель предметної області або розв'язуваної задачі, необхідно отримати дані і метадані. Динамічна інтерпретація метамоделі - процес створення моделі предметної області з метамоделі, метаданих і даних. Динамічна інтерпретація відбувається як на клієнтській, так і на серверній частині в момент виконання. При динамічній інтерпретації модель предметної області не будується повністю в пам'яті, а інтерпретується фрагментарно, у міру обробки вхідних даних і може кешуватися в прикладній віртуальній машині, щоб уникнути витрат обчислювальних ресурсів на багаторазову інтерпретацію.

Метамодель містить специфікацію метаданих, які необхідні для динамічної інтерпретації. Якщо це перший сеанс роботи компоненти і в доступному для нього постійному сховищі і машині станів ще немає метаданих, то їх одержання може бути зроблене за допомогою ідентифікатора (адреси) або запиту, введеного користувачем або переданого від іншої компоненти системи (але не зашитого в компоненті). Цим процесом керує сесійний компонент прикладної віртуальної машини. Надалі, метадані можуть кешуватися, як кешуються дані для мінімізації запитів, а оновлюватися тільки при зміні версії або контрольної суми в джерелі (системі зберігання або віддаленому компоненті) або при отриманні події від джерела. Отримання змінених метаданих призводить до перебудови моделі предметної області в оперативній пам'яті.

Метадані, що надійшли таким чином до компоненти системи, доповнюють метамодель своїми декларативними параметрами і імперативними

сценаріями, таким чином, що стає можливим динамічна побудова класів предметної області, їх наповнення отриманими даними і опрацювання за допомогою динамічно ж побудованої бізнес-логіки. У результаті, як функціональна компонента, так і інтерфейси взаємодії з іншими компонентами можуть змінюватися в широкому класі вирішуваних завдань без перекомпіляції системи, а багато змін можливі при простому заданні нових параметрів метаданих (навіть без модифікації сценаріїв).

Динамічно побудовані прикладні структури даних і прикладні сценарії (події, методи обробки даних) спираються на виклики API прикладної віртуальної машини, в якій запущена динамічна модель. Таке API, у свою чергу, «сідає» на API середовища запуску. Динамічна побудова моделі відбувається як в серверних, так і в клієнтських (користувацьких) компонентах інформаційної системи, тобто інтерфейс користувача будується динамічно, а роль API тут виконує використовувана бібліотека візуалізації (компоненти GUI).

Користувач ініціює події в GUI, які призводять до локальної обробки даних або мережевим викликам. Сенс динамічної інтерпретації в мережевій взаємодії полягає в тому, що ідентифікатори віддалених процедур, що викликаються, їх параметри і структура даних, що повертаються заздалегідь не зашиті у компоненті, яка викликається, а виходять з метаданих під час динамічної інтерпретації метамоделі. Дистанційна ж сторона так само не має фіксованого інтерфейсу, натомість, заздалегідь відомий тільки метапротокол (формати передачі даних, типи параметрів, обробка яких підтримується) і механізм інтроспекції, що дозволяє динамічно зчитувати структуру віддаленого мережевого інтерфейсу.

Роль машини станів в динамічній інтерпретації в тому, щоб зберігати сесію з необхідним набором даних як на клієнті, так і на сервері між мережевими викликами. Це необхідно для того, щоб забезпечити послідовну

або інтерактивну людино-машинну обробку даних, при якій дії користувача призводять модель в певний стан, а кожний наступний мережевий виклик залежить від попереднього в межах транзакції. Крім того, машина станів дозволяє істотно скоротити витрати трафіку і обчислювальних ресурсів, кешуючи метадані, дані і навіть саму динамічно побудовану модель в оперативній пам'яті.

Таким чином, багатошарова структура компонентів інформаційної системи з динамічною інтерпретацією метамоделей замикається знизу на сховище, яке обмежене рівнем абстракції метамоделі, а зверху замикається на користувача, який може модифікувати не тільки дані, але й метадані, переналагоджуючи інформаційну систему під постійно мінливі вимоги розв'язуваної задачі, в цьому і полягає гнучкість підходу і спрощення модифікації систем, зниження необхідного рівня кваліфікації користувача при внесенні змін в структуру і функції. Місце ж архітектора програмних рішень і програмістів у створенні програмного забезпечення з підвищеним рівнем абстракції, тобто в реалізації метамоделі (що є, природно, більш важкою інженерною задачею). Але в кінцевому підсумку, підхід окупає себе, завдяки підвищенню повторного використання коду з високою абстракцією, автоматизації багатьох завдань зв'язування компонентів, зниження впливу людського чинника при модифікації та інтеграції систем і спрощення інтеграції між компонентами програмних систем.

Для взаємодії з іншими системами бажано використовувати стандартизовані технології та методи. У зв'язку з цим було виділено вимоги, що стосуються стандартів:

1. В системі повинні використовуватися стандартні технології семантичної павутини, рекомендовані консорціумом W3C. Онтологічна модель повинна описуватися в форматі RDF із застосуванням мови OWL. Доступ до даних повинен здійснюватися за допомогою мови запитів SPARQL. Система

повинна підтримувати використання існуючих онтологій верхнього рівня, за рахунок механізму простору імен.

2. В системі повинні бути реалізовані рішення, пропоновані в рамках стандартів відображення реляційних баз даних.

3. Доступ до інтегрованих систем повинен здійснюватися за допомогою стандартизованих засобів, таких як SOAP web-сервіси та JDBC адаптери до баз даних.

4.2.1 Функціональні вимоги до підсистеми опису структури і змісту вхідних даних

До структури підсистеми опису структури і змісту вхідних даних входять наступні компоненти:

- модуль формування шаблону запиту на використання даних із джерел інформаційних ресурсів, затребуваних користувачем;
- модуль створення і зміни онтологічних описів, який відповідає за створення і зміну понять і ієрархій в базовій онтології;
- репозиторій метаданих, що забезпечує зберігання описів онтологій предметних областей;
- модуль зіставлення і інтеграції онтологій, що виконує об’єднання онтологій різних предметних областей;
- модуль вирішення конфліктів, що можуть виникнути при зіставленні концептів онтологій;
- модуль, що відповідає за формування моделі кожного вхідного інформаційного ресурсу загальної структури, визначеної предметною областю;
- додаткові компоненти (адаптери, конфігуратори, конектори).

У репозиторії метаданих зберігаються описи всіх концептів, фізичних і логічних зв'язків між інформаційними системами, а також посилання на всі процедури і сервіси, що забезпечують підтримку якісної інформації. Всі

триплети обробляються і зберігаються у вигляді записів в таблицях за схемою MDSYS. Сутності описуються шляхом зазначення імені, атрибутів і їх значень. Структура RDF-специфікації складається з опису всіх інформаційних об'єктів, а також URI посилання на описуваний інформаційний об’єкт і описів атрибутів (ім'я; його значення; посилання на інший інформаційний об’єкт). Кожен інформаційний об'єкт пов'язаний з яким-небудь поняттям з онтології. Цей зв'язок вказується у властивостях поняття. З метою відстеження поновлення примірників понять в реляційну базу даних додаються тригери.

Ключовими етапами роботи підсистеми є наступні:

1. Підключення до сервера баз даних джерел інформаційних ресурсів і витягнення онтологій цих інформаційних систем у форматі OWL. Одна з обраних онтологій вказується основною (головною), в яку необхідно провести відображення концептів другої онтології. Отримання даних, що не містяться в БД інтегруючої системи, відповідно до наперед встановленої специфіки опису тої чи іншої web-системи, та запис їх у спеціальну базу даних, яка теж використовується для побудови загальної онтології системи, що інтегрується.

2. Знаходження семантичних залежностей між онтологіями систем, що інтегруються, визначення семантичної близькості концептів двох онтологій, рішення різноманітних конфліктів.

3. Відображення онтологій. На цьому етапі виконуються наступні кроки:

1) інтеграція еквівалентних концептів:

- копіювання властивостей обох вихідних концептів;
- копіювання відношень обох вихідних концептів.

2) прив'язка концептів, які перебувають у відношенні R1 «клас-підклас» з обома вихідними концептами (якщо вони присутні в створюваній онтології):

- інтеграція еквівалентних властивостей концептів;
- інтеграція еквівалентних відношень між концептами;
- копіювання концептів в створювану онтологію;

- копіювання атрибутів в створювану онтологію;
- копіювання відношень в створювану онтологію.

4. Встановлення взаємозв'язку інформаційних систем на структурному рівні, ґрунтуючись на узгодженій семантиці предметної області.

Описання всіх концептів, фізичних і логічних зв'язків між інформаційними системами зберігаються в репозиторії метаданих. У ньому зберігаються також посилання на всі процедури і сервіси, що забезпечують підтримку якісної інформації. Система управління репозиторієм вирішує наступні завдання:

- створення нових концептів і редагування існуючих;
- визначення атрибутів концепту, встановлюється тип та обмеження;
- визначення властивостей концепту як відношення з іншими концептами;
- визначення зв'язку онтологічних специфікацій з об'єктними схемами інформаційних джерел.

Для забезпечення зручної роботи користувача із системою було визначено, що підсистема опису структури і змісту вхідних даних повинна відповідати певним функціональним вимогам (functional requirement):

- FR1: Підсистема повинна підтримувати всі найбільш поширені типи реляційних баз даних, а також бути розширюваною. Повинні підтримуватись такі СУБД, як Oracle, MSSQL Server, PostgreSQL, MySQL. Одночасно можуть інтегруватися системи, що використовують різні СУБД.
- FR2: Підсистема повинна мати можливість розподіляти дані, що не містяться в базі даних інтегруючої web-системи в наперед створену базу даних. Для автоматизації цього процесу, необхідно отримувати дані використовуючи мову запитів до елементів мови розмітки Xpath.

- FR3: Підсистема повинна мати можливість отримувати інформацію про структуру кожної з інтегрованих систем і представляти її у форматі RDF.
- FR4: Підсистема повинна забезпечувати автоматизоване об'єднання окремих метамоделей, що містять інформацію про інтегровані системи, в єдину онтологію. Об'єднання має відбуватися з урахуванням семантичних зв'язків між інтегрованими інформаційними системами.
- FR5: Підсистема повинна робити витяг семантичних метаописів інформаційних джерел даних на основі загальної онтології структури.
- FR6: Підсистема повинна автоматично визначати інформацію, що дублюється при витягненні метаданих ресурсів.
- FR7: Всі необхідні налаштування формування результату і вибір джерел для інтеграції повинні проводитися через користувацький web-інтерфейс.
- FR8: Користувачу необхідно надати вибір джерел для побудови Mashup результату, а також їх редагування, при необхідності.
- FR9: Підсистема повинна виводити чіткі і конкретні повідомлення про помилки, а також пропозиції їх уникнення при не коректному користуванні нею.

Дані вимоги були сформульовані із врахуванням розроблених методів та включають найкращі можливості сучасних Mashup систем.

4.2.2 Функціональні вимоги до підсистеми формування контенту Mashup

Підсистему формування контенту Mashup реалізовано у вигляді таких основних компонентів:

- модуль опрацювання запиту користувача, який реагує на некоректність вводу даних для запиту та відповідає за імпорт метаописів інформаційних ресурсів із репозиторію метаданих, виділення пошукового образу запиту, аналіз на подібність запиту користувача та метаописів, додання подібних метаописів до знайдених, експорт даних у сховище;

- модуль редагування даних дає можливість врахувати внесені зміни у запит користувача через постійний зв’язок із модулем опрацювання запиту;

- модуль відображення даних, який відповідає за ранжування результуючих даних відповідно до зв’язків подібності між ними та новизною їх опублікування у джерелі;

- додаткові компоненти (адаптери, конфігуратори, конектори).

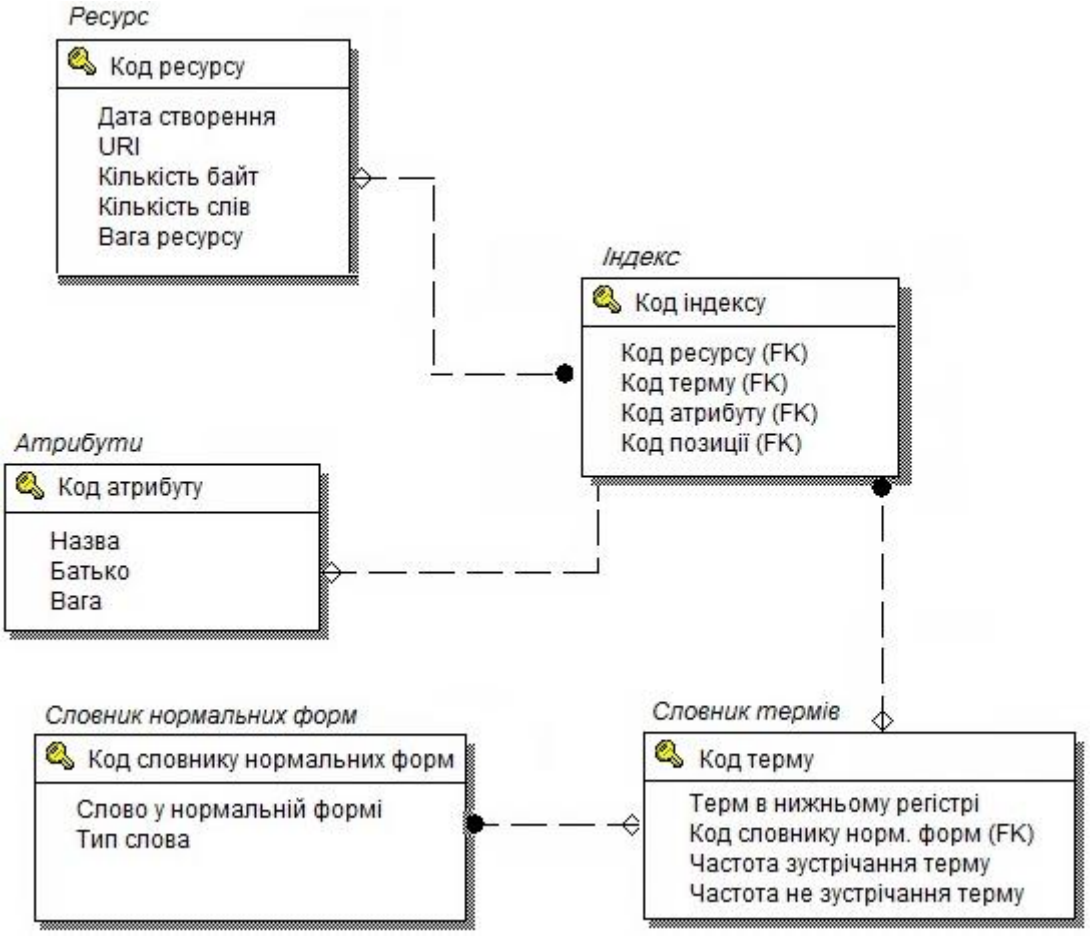


Рисунок 4.8 - ER-діаграма таблиць для зберігання індексів метаописів інформаційних ресурсів

Модуль опрацювання запиту користувача відповідає за виконання наступних завдань:

1) Аналіз коректності подання запиту – запит користувача повинен відповідати певним визначеним умовам системи.

Виділення для кожного джерела певної наперед заданої у системі кількості метаописів інформаційних ресурсів, яка далі використовуватиметься для аналізу на схожість із пошуковим образом запиту. Дана інформація повинна записуватися до бази даних системи за допомогою спеціально створених взаємопов’язаних таблиць окремо для кожного джерела (рис. 4.8).

2) виділення пошукового образу запиту – повинні бути виділені терміни із пошукового запиту та семантично подібні терміни до термінів пошукового запиту (отримуються за допомогою використання загальної онтології предметної області) та розставлені асоціативні зв’язки між термінами пошукового запиту та між семантично подібними термінами до термінів пошукового запиту;

3) аналіз на подібність запиту користувача та метаописів – для кожного виділеного метаопису потрібно знайти коефіцієнт подібності запиту і метаопису;

4) додання подібних метаописів до знайдених – для кожного джерела повинно бути наперед визначено максимальну кількість результуючих даних, яка, при ситуації недостачі даних результату визначеної кількості, повинна доповнюватися семантично подібними мета описами до знайдених, які повинні отримуватися за допомогою коефіцієнта подібності метаописів;

5) зберігання отриманої інформації у сховище, якщо у системі передбачено дану процедуру.

Для забезпечення зручної роботи користувача із системою було визначено, що підсистема формування контенту Mashup повинна відповідати певним функціональним вимогам (functional requirement):

- FR10: Всі необхідні налаштування формування результату і вибір джерел для інтеграції повинні проводитися через користувацький web-інтерфейс.
- FR11: Користувачу необхідно надати вибір джерел для побудови Mashup результату, а також їх редагування, при необхідності.

- FR12: Підсистема повинна виводити чіткі і конкретні повідомлення про помилки, а також пропозиції їх уникнення при не коректному користуванні нею.
- FR13: Підсистема повинна отримувати та виконувати деталізацію метаописів інформаційних ресурсів певної визначеної кількості для кожного джерела.
- FR14: Підсистема повинна ранжувати результуючі дані відповідно до визначених між ними зв’язків подібності.

Дані вимоги були сформульовані із врахуванням розроблених методів та включають найкращі можливості сучасних Mashup систем.

4.3 Тестування можливостей систем із впровадженою інформаційною технологією динамічної інтеграції слабоструктурованих даних у web-системах

Розроблені програмні засоби для підвищення якості процесів отримання даних, оброблення отриманих даних та їх подання у вигляді сервісу у системі, що працює використовуючи технологію Mashup динамічної інтеграції даних використовуються у наступних проектах: News Mashup from Social Networks - система моніторингу новин із соціальних мереж (http://snmashup.in) та Shares and Discounts Mashup – система пошуку знижки на купівлю товару, отримання послуги тощо (http://discountsmashup.in.ua). Подано програмні реалізації розроблених систем з підсистемами, що працюють на основі розробленої інформаційної технології динамічної інтеграції слабоструктурованих даних у web-системах.

Процес опрацювання даних у системах із впровадженою інформаційною технологією динамічної інтеграції слабоструктурованих даних у web-системах відбувається з використанням розроблених методу опису структури та змісту

вхідних даних та методу формування об’єднаного динамічного набору даних, який має загальну структуру і єдиний зміст (додаток Б).

Вхідними даними систем із впровадженою інформаційною технологією динамічної інтеграції слабоструктурованих даних у web-системах Shares and Discounts Mashup і News Mashup from Social Networks є джерела інформаційних ресурсів, до яких має доступ система та запит користувача. Користувач задає запит і отримує відповідь на нього у вигляді колажу із інформаційних ресурсів доступних джерел, які відповідають заданому запиту.

При проектуванні Інтерфейси, розроблених систем були спроектовані досить просто, так, щоб було зрозуміло будь-якому користувачеві. Інтерфейс написано з елементами синтаксису HTML, оскільки HTML якнайкраще підходить для оформлення web-сторінок. Описуючи систему Shares and Discounts Mashup видно, що назву системи ненав’язливо, але все ж досить помітно виділено в лівому верхньому куті сторінки (рис. 4.10). Основне місце фону займає надпис «Sale», що безпосередньо характеризує систему пошуку купонів знижок. Навігація по системі здійснюється завдяки використанню пунктів меню або просто гортаючи сторінку вниз. Так як система має вигляд web-сайту, передбачено деякий його опис (необхідність, призначення, функції, тощо). Для цього виділено пункт меню із назвою, що не потребує пояснень: «Про нас». Пункти меню розміщено горизонтально у верхній частині сайту зліва направо.

Основні пункти меню – це: «Як це працює?», «Пошук», «Партнери», «Про нас», «Контакти» (див. додаток В).

Натиснувши на пункт меню «Як це працює?» Ви перейдете у частину сайту з описом правил роботи із системою. Пункт меню «Пошук» відповідає стрічці пошуку, де користувач може ввести запит на пошук необхідного купону на знижку і система видасть Mashup результатів – інформаційні ресурси із доступних джерел, які найбільш повно відображають відповідь на

користувацький запит. Пункт меню «Партнери» містить перелік можливих джерел для пошуку інформаційних ресурсів із можливістю безпосередньо перейти на сайт-джерело і ознайомитися із ним детальніше. Даними джерелами виступають такі відомі сайти купонів на знижки: Pokupon, Biglion, Gara, Goodplace, Groupon, KupiSkidku, тощо. Загалом пошук здійснюється більше, ніж на двадцяти різних сайтах купонів на знижки. Пункт меню «Про нас» містить опис системи. Пункт меню «Контакти» - зворотній зв'язок для відгуків, пропозицій користувачів.

Приклад роботи із системою: користувач вводить у пошукову стрічку «Спа-процедуры». Так як майже всі українські сайти купонів на знижки описані російською мовою – умовою для введення запиту є російська мова. Результат на даний запит ми можемо побачити на рисунку 4.9.

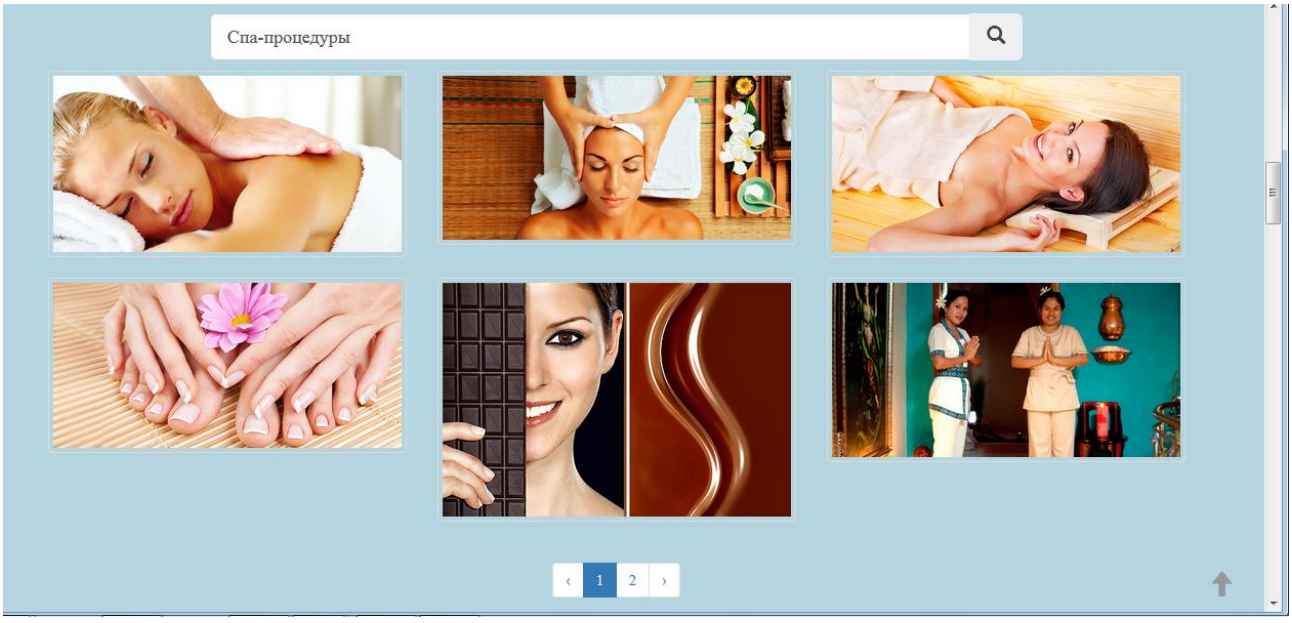


Рисунок 4.9 - Результат роботи системи Shares and Discounts Mashup

Надзвичайно проста і зрозуміла схема роботи даної системи дає змогу користувачу із будь-яким рівнем володіння комп'ютерними технологіями виконати пошук купону на знижку купівлі необхідного товару чи отримання певної послуги легко і швидко. Далі, знайшовши потрібну знижку, користувач

може безпосередньо перейти до джерела купону на знижку і ознайомитися із акцією більш детальніше.

News Mashup from Social Networks призначена для отримання інформаційних ресурсів із web-систем, що інтегруються і подання їх у вигляді пов'язаних даних, відповідно до запиту користувача. Система автоматизовано виконує відображення інформації, що зберігається в гетерогенних, інформаційних web-системах, що інтегруються, в онтологічну модель, яка пізніше використовується для пошуку інформаційних ресурсів, згідно запиту користувача. Із допомогою News Mashup from Social Networks можна отримати інформацію про будь-які події, явища, тощо із таких соціальних мереж, як: вконтакте, facebook, twitter.

Реалізація системи динамічної інтеграції слабоструктурованих даних у web-системах являє собою сукупність завершених модулів, які можна використати для побудови інших систем.

Основні етапи щодо користування системою будуть наступні:

1. Для роботи із системою потрібно перейти за посиланням: http://snmashup.in.

2. Перед нами на екрані інтерфейс системи News Mashup from Social Networks (додаток В).

3. Заповнюємо форму для пошуку новин і вибираємо соціальні мережі, де нам потрібно здійснити пошук новин та натискаємо на кнопку «Пошук».

4. Отримуємо результат пошуку у вигляді колажу із новин, що ранжуються відповідно до новизни їх опублікування у мережі та відображаються відповідно до зв’язків подібності результуючих даних.

5. При потребі здійснюємо корекцію пошукового запиту і натискаємо на кнопку «Пошук».

6. Отримуємо результат на відредагований запит пошуку.

7. Обравши будь-який запис у результуючому колажі ми можемо безпосередньо перейти до джерела даного запису для більш детального ознайомлення із ним.

News Mashup from Social Networks дає змогу користувачу із будь-яким рівнем володіння комп'ютерними технологіями виконати пошук новин одночасно у декількох найпопулярніших соціальних мережах абсолютно на будь-яку тематику легко і швидко. Далі, знайшовши цікаву його увазі новину, користувач може безпосередньо перейти до джерела даної новини і ознайомитися із її змістом та, можливо, автором запису більш детальніше.

Після проведення низки експериментів щодо подання результатів систем із впровадженою інформаційною технологією динамічної інтеграції слабоструктурованих даних у web-системах встановлено, що міра подібності інформаційних ресурсів, виданих системами, користувацькому запиту знаходиться в межах 86-90%, що свідчить про високу якісь пошуку інформації (табл. 4.1).

Таблиця 4.1 - Результати роботи системи News Mashup from Social Networks

Експе-римент	Характеристика			
	Кількість знайдених результатів	Співвідношення знайдених джерел до всіх можливих, %	Точність, %	Доступність до повної інформації, %
1	140	100	87	100
2	114	100	89	100
3	110	67	81	100
4	90	33	83	100
5	115	67	90	100

Для тестування роботи систем з підсистемами, що працюють на основі розробленої інформаційної технології динамічної інтеграції слабоструктурова-

них даних у web-системах було здійснено 424 експерименти (приклад результатів деяких експериментів відображено в табл. 4.1).

Для аналізу якості пошуку і подання даних (додаток Г, рис. 4.10, рис. 4.11), використано такі характеристики: критерій оцінки точності, критерій оцінки повноти, оцінка рівня інформаційного шуму, оцінка рівня узгодженості даних (семантична зв'язність об'єднаного набору даних). Звідси, якість пошуку має вигляд:

$$e_i = \langle p_i, r_i, il_i, n_i, rc_i \rangle, \tag{4.1}$$

де: p_i – точність (precision) пошуку; r_i - повнота (recall) пошуку; il_i - втрати інформації (information loss); n_i - пошуковий шум (search noise); rc_i – узгодженість результатів (results coordination).

Для обчислення цих показників використовуються стандартні формули:

$$r = \frac{To}{To + No}, \tag{4.2}$$

де: To – правильно додані об'єкти; No – кількість об'єктів, які відповідають запиту, але їх не було додано.

$$p = \frac{To}{To + Fo}, \tag{4.3}$$

де: b - кількість неправильно доданих об'єктів.

$$il = 1 - r, \tag{4.4}$$

$$n = 1 - p, \tag{4.5}$$

де: rc_i визначається за допомогою коефіцієнта подібності метаописів сусідніх інформаційних ресурсів k_u.

Гармонійною серединою між точністю і повнотою є F-міра.

$$F = \frac{(\beta^2 + 1) \cdot r \cdot p}{\beta p + r}, \tag{4.6}$$

де: p – точність (precision) пошуку; r - повнота (recall) пошуку; β - коефіцієнт, що виражає важливість або точності при $0<\beta<1$, або повноти при $\beta>1$, а при $\beta=1$ - точність і повнота мають однакову вагу.

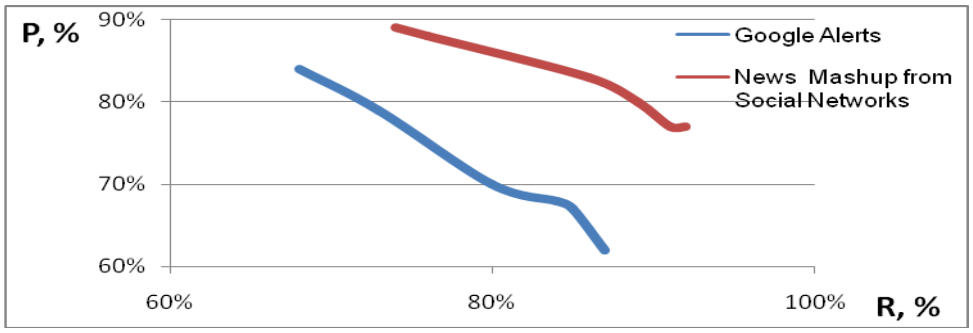


Рисунок 4.10 - Графік повноти-точності систем Google Alerts та News Mashup from Social Networks

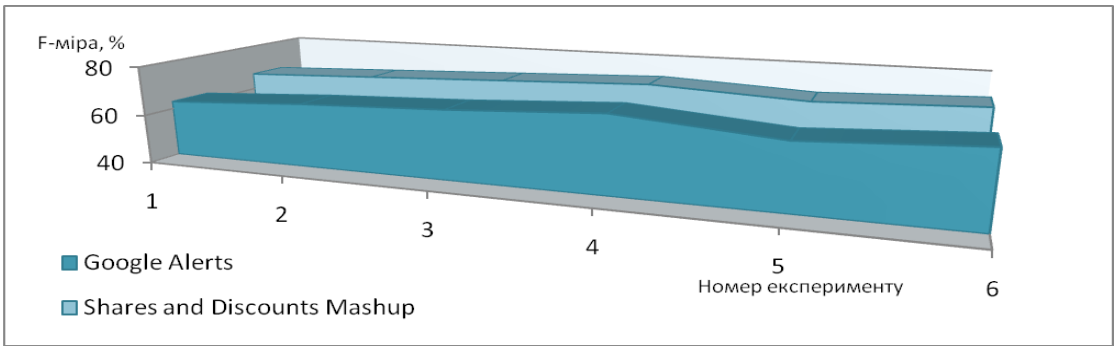


Рисунок 4.11 - Відображення точності системи із інформаційною технологією динамічної інтеграції слабоструктурованих даних та аналога

Визначено якість пошуку і подання інформаційних даних для різних методів. Для порівняння проаналізовано отримання результату трьома іншими методами для одних і тих самих запитів (табл. 4.2). Таким чином, результати роботи системи з підсистемами, що працюють на основі розробленої інформаційної технології кращі, порівняно із результатами інших методів.

Таблиця 4.2 - Порівняння формування результатів роботи Mashup системи, використовуючи різні методи опрацювання даних

Технологія	Точність	Інформ. шум	Узгодженість
Розроблена ІТ	91%	9%	79%
Технологія на основі парадигми програмування	73%	27%	69%
Технологія на основі електронних таблиць	77%	23%	57%
Технологія на основі блокового подання	85%	15%	62%

Наявність підсистем опису структури та змісту вхідних даних та формування об’єднаного динамічного набору даних, що має загальну структуру і єдиний зміст в Mashup системах сприяє збільшенню точності результуючих даних до 6%, зниженню інформаційного шуму вихідних даних до 6% та покращенню узгодженості результатів до 10%.

4.4 Висновки до розділу

1. Охарактеризовано проект розроблення системи динамічної інтеграції слабоструктурованих даних у web-системах. Було використано підхід до проектування, згідно якого робиться акцент на розробленні архітектури системи, семантиці інтегрованих даних і особливостях запиту користувача. Таким чином було удосконалено структурну модель Mashup системи динамічної інтеграції слабоструктурованих даних шляхом додання функціональних компонентів визначення та опису семантики вхідних та вихідних наборів даних, що дало змогу розширити функціональні можливості таких систем.

2. Розроблено функціональні вимоги до системи динамічної інтеграції даних на основі технології Mashup для того, щоб забезпечити узгодженість роботи всіх модулів формування результату роботи системи.

3. Розроблені програмні засоби для підвищення характеристик якості пошуку і подання даних у системі, що працює використовуючи технологію Mashup динамічної інтеграції даних.

4. Розроблені методи реалізовано у наступних проектах: News Mashup from Social Networks - система моніторингу новин із соціальних мереж (http://snmashup.in) та Shares and Discounts Mashup – система пошуку знижки на купівлю товару, отримання послуги тощо (http://discountsmashup.in.ua). Спроектовані застосунки працюють на основі технології Mashup з підсистемами опису структури та змісту вхідних даних та формування об'єднаного динамічного набору даних, що має загальну структуру і єдиний зміст. Подано програмні реалізації розроблених систем з підсистемами, що працюють на основі розробленої інформаційної технології динамічної інтеграції слабоструктурованих даних у web-системах, що дало змогу оцінити функціональні можливості систем із впровадженою ІТ та аналогами.

5. Дано опис таких характеристик, як: критерій оцінки точності, критерій оцінки повноти, оцінка рівня інформаційного шуму, оцінка рівня узгодженості даних, за допомогою яких можна оцінити якість пошуку і подання даних у Mashup системі динамічної інтеграції слабоструктурованих даних.

6. Проведено тестування можливостей систем із впровадженою інформаційною технологією динамічної інтеграції слабоструктурованих даних у web-системах, в результаті чого встановлено, що використання інформаційної технології динамічної інтеграції слабоструктурованих даних у web-системах сприяє збільшенню точності вихідних даних до 6%, зниженню інформаційного шуму результуючих даних до 6%, а також покращення узгодженості результатів до 10%.

ВИСНОВКИ

У дисертаційній роботі розв’язано актуальне наукове завдання розроблення інформаційної технології динамічної інтеграції слабоструктурованих даних у web-системах для підвищення якості пошуку даних web-систем. Отримані в дисертації результати у сукупності є теоретичною та практичною основою для проектування Mashup систем динамічної інтеграції слабоструктурованих даних із врахуванням семантичного чинника опрацювання даних для підвищення точності та повноти результату, зниження рівня інформаційного шуму та покращення узгодженості результуючого набору даних. Отримано такі основні наукові і практичні результати:

1. Дослідження та аналіз принципів побудови та функціональних можливостей сучасних Mashup систем динамічної інтеграції даних показав, що існуючі методи та технології, що використовуються для побудови таких систем не беруть до уваги семантичний фактор опрацювання даних при їх динамічній інтеграції, врахування якого б дозволило отримати більш якісні результати для Mashup систем, що здійснюють інформаційний пошук.

2. Шляхом застосування онтологій для класифікації інформаційних ресурсів розроблено метод опису структури і змісту вхідних даних Mashup системи, що дало змогу врахувати зміст даних у процесах їх інтеграції.

3. Через врахування змістовних вимог запиту користувача розроблено метод формування об'єднаного динамічного набору даних, що забезпечило підвищення семантичної коректності результатів динамічної інтеграції слабоструктурованих даних.

4. Через запровадження додаткових можливостей опрацювання семантики інформаційних ресурсів отримала подальший розвиток Mashup технологія динамічної інтеграції даних, що дало змогу підвищити якість результатів динамічної інтеграції слабоструктурованих даних.

5. Шляхом додання функціональних компонентів визначення та опису семантики вхідних та вихідних наборів даних удосконалено структурну модель системи динамічної інтеграції слабоструктурованих даних на основі технології Mashup, що дало змогу розширити функціональні можливості таких систем.

6. Розроблено та впроваджено програмні засоби динамічної інтеграції слабоструктурованих даних для підвищення характеристик якості пошуку і подання даних таких, як: збільшення точності до 6% та зниження інформаційного шуму до 6%, а також покращення узгодженості результатів даних до 10% для процесу отримання результуючої множини вихідних даних у Mashup системі динамічної інтеграції даних.

ЛІТЕРАТУРА

1. Андрейчиков А.В. Интеллектуальные информационные системы / А.В. Андрейчиков, О.Н. Андрейчикова // Учебник. Финансы и статистика, Москва, 2004. – 424 с.

2. Асадуллаев С. Архитектуры хранилищ данных. Цикл статей [Електронний ресурс] / IBM-2009. // Режим доступу: URL: http://www.ibm.com/developer-works/ru/library/sa-bir/axd_1/index.html.

3. Батоврин В.К. Разработка понятийной схемы (онтологии) для обеспечения единой семантики в среде открытой системы интеграции разнородных данных / В.К. Батоврин, М.Р. Когаловский, А.С. Королев, А.Б. Петров // Телематика'2006: материалы Всероссийской научно-методической конференции. СПб., июнь 2006 г. – СПб.: Из-во СПбГУ ИТМО, 2006. − С. 90−91.

4. Барсегян А.А. Технологии анализа данных / А.А. Барсегян, М.С. Куприянов, В.В. Степаненко, И.И. Холод // Data Mining, Visual Mining, Text Mining, OLAP – 2е изд., перераб. и доп. БХВ-Петербург, СПБ, 2007. – 384с.

5. Башмаков А.И., Башмаков И.А. Интеллектуальные информационные технологии: Учеб. пособие / А.И. Башмаков, И.А. Башмаков // Изд-во МГТУ им. Н.Э. Баумана, М., 2005. – 304 с.

6. Бездушный А.Н. Интегрированная система информационных ресурсов РАН и технология разработки цифровых библиотек / А.Н. Бездушный, А.Б. Жижченко, М.В. Кулагин, В.А. Серебряков // Программирование, 2004. - С. 3-14.

7. Бездушный А.А. Математическая модель интеграции данных на основе дескриптивной логики: автореф. дис. ... канд. физ.-мат. наук. – М., 2008. – С. 21.

8. Брайчевский С.М. Современные информационные потоки: актуальная проблематика / С.М. Брайчевский, Д.В. Ландэ // Научно-техническая информация. Сер. 1. Вып. 11. – 2005. - С. 21-33.

9. Брюхов Д.О. Интероперабельные информационные системы: архитектуры и технологии / Д.О. Брюхов, В.И. Задорожный, Л.А. Калиниченко, М.Ю. Курошев, С.С. Шумилов // СУБД. Москва, 1995, №4. – С.86-113.

10. Бубарева О.А. Адаптируемая система расчета себестоимости ОУ на основе онтологического подхода как часть автоматизированной системы управления вузом / О.А. Бубарева // Сборник научных работ Всероссийского конкурса научно-исследовательских работ в области технологий электронного обучения в образовательном процессе. − Белгород: Белгородский государственный университет, 2010. − С. 112−116.

11. Бубарева О.А. Использование интеграции информации для анализа несопоставимых источников данных в информационно-управляющих системах / О.А. Бубарева, Ф.А. Попов // Единая образовательная информационная среда: проблемы и пути развития: труды VIII Международной научно-практической конференции, Томск: ТГУ, 2009. – С. 136–137.

12. Бубарева О.А. Решение проблемы интеграции данных при построении интегрированной автоматизированной информационной системы вуза / О.А. Бубарева, Ф.А. Попов, Н.Ю. Ануфриева // Международный журнал экспериментального образования, №5, 2011. – С. 90–92.

13. Вагин В.Н. Разработка метода интеграции информационных систем на основе метамоделирования и онтологии предметной области / В.Н. Вагин, И.С. Михайлов // Программные продукты и системы. − 2008. – С. 22−26.

14. Гаврилова Т.А. Онтологический инжиниринг // Сб. докладов Восьмой научно-практической конференции «Реинжиниринг бизнес-процессов на основе современных информационных технологий. Системы управления знаниями» (РБП-СУЗ-2005). М.. , 2005. – с.79-82.

15. Гладун А.Я. Онтологии в корпоративных системах / А.Я. Гладун, Ю.В. Рогушина // Корпоративные системы. – 2006. − № 1. – Ч. II.

16. Гудков В. Ю. N-граммы в лингвистике [Текст] / В. Ю. Гудков, Е. Ф. Гудкова // Вестник Челябинского государственного университета. – 2011. – № 24 (239). – С. 69-71.

17. Добровольский А. Интеграция приложений: методы взаимодействия, топология, инструменты /А. Добровольский // Открытые системы №9, 2006.

18. Дубова Н. Интеграция приложений и бизнес-процессы / Н. Дубова // Открытые системы №9, 2006.

19. Эйзенберг Э. SQL:1999, ранее известный как SQL3 / Э. Эйзенберг, Дж. Мелтон // Пер. с англ. Открытые системы. – 1999. - № 1. – С. 52-57.

20. Ивашко А.Г. Применение дескрипционной логики для описания архитектуры информационной системы / А.Г. Ивашко, Е.И. Иванова, Е.О. Овсянникова, С.И. Коломиец // Вестник Тюменского государственного университета. – 2012. – №4. – С. 137-142.

21. Калиниченко Л.А. Логика отображения сетевой модели данных в реляционную / Л.А. Калиниченко, А.Е. Рамъялг // Программирование, 2, 1979.

22. Калиниченко Л.А. Проблемы создания предметного посредника для интеграции молекулярно-генетических информационных ресурсов / Л.А. Калиниченко, Н.А. Колчанов, Н.Л. Подколодный // Вторая Всероссийская научная конференция «Электронные библиотеки: перспективные методы и технологии, электронные коллекции», Протвино, 2000. - С. 174-184.

23. Дрюэк К. Хранилища данных: сходство и различия подходов Билла Инмона и Ральфа Кимболла [Електронний ресурс] / К. Дрюэк // 2005 Режим доступу: URL: http://www.b-eye-network.com/view/743

24. Клещев А.С. Математические модели онтологий предметных областей. Часть 3. Сравнение разных классов моделей онтологій / А.С. Клещев, И.Л. Артемьева // Научно–техническая информация, Сер. 2. Информационные процессы и системы, № 4, 2001. - С. 10–15.

25. Когаловский М.Р. Энциклопедия технологий баз даннях / М.Р. Когаловский // Финансы и статистика, Москва, 2002. – 800 с.

26. Когаловский М.Р. Перспективные технологии информационных систем / М.Р. Когаловский // ДМК Пресс, Москва, 2003. – 288 с.

27. Крупский В. Интеграция приложений на основе концепции Service Oriented Architecture (SOA) / В. Крупский // Connect №6, 2008.

28. Кузнецов О.П. Онтология как систематизация научных знаний: структура, семантика, задачи [Електронний ресурс] / О.П. Кузнецов, В.С. Суховеров, Л.Б. Шипилина // Режим доступу: URL: http://cmm.ipu.ru/proc

29. Кузнецов С. От баз данных к пространствам данных: новая абстракция управления информацией / С. Кузнецов, 2006 // Режим доступу: URL: http://www.citforum.ru/data-base/articles/from_db_to_ds

30. Кушнірецька І. І. Визначення структури і змісту вхідних інформаційних ресурсів для роботи Мешап-системи / І.І. Кушнірецька, О.І. Кушнірецька, А.Ю. Берко // Технологический аудит и резервы производства — № 6/3(20), 2014, С. 4-9.

31. Кушнірецька І.І. Аналіз інформаційних ресурсів системи динамічної інтеграції слабоструктурованих даних web-середовищі / І.І. Кушнірецька, О.І. Кушнірецька, А.Ю. Берко // Вісник Національного університету «Львівська політехніка». – 2014. – № 805: Інформаційні системи та мережі. – С. 162-169.

32. Кушнірецька І.І. Використання онтологій для інтеграції слабострук-турованих даних у web-системах / І.І. Кушнірецька, А.Ю. Берко // Матеріали XI Відкритої наукової конференції ІМФН (PSC-IMFS-11) — Львів: Видавництво НУ «Львівська політехніка», 2013 С. 149-150.

33. Кушнірецька І.І. Застосування онтологій і метамоделей для динамічної інтеграції слабоструктурованих даних / І.І. Кушнірецька, О.І. Кушнірецька, А.Ю. Берко // Вісник Національного університету "Львівська політехніка". – 2014. – № 783: Інформаційні системи та мережі. – С. 128–137.

34. Кушнірецька І.І. Методика «чорної скриньки» та агентно-орієнтованого підходу для забезпечення Mash-Up інтеграції інформаційних ресурсів / І.І. Кушнірецька, А.Ю. Берко // Матеріали III Міжнародної наукової конференції

«Інформація, комунікація, суспільство 2014» (ІКС-2014). — Львів: Видавництво НУ «Львівська політехніка», 2014. - С. 32-33.

35. Кушнірецька І.І. Отримання метаданих вхідного інформаційного ресурсу при його динамічній інтеграції / І.І. Кушнірецька, О.І. Кушнірецька, А.Ю. Берко // Матеріали XI Міжнародної наукової конференції «Інтелектуальні системи прийняття рішень і проблеми обчислювального інтелекту» (ISDMCI'2015). – Залізний Порт, 2015.

36. Кушнірецька І.І. Принципи семантичного пошуку науково-технічних інформаційних ресурсів / І.І. Кушнірецька, О.І. Кушнірецька, А.Ю. Берко // Мате-ріали III Міжнародної науково-практичної конференції «Інформаційні управляючі системи та технології» (ІУСТ-Одеса-2014). – Одеса: «ВМВ», 2014. – С. 253-255.

37. Кушнірецька І. І. Проектування системи динамічної інтеграції слабо-структурованих даних на основі технології Mash-Up / І.І. Кушнірецька, О.І. Кушнірецька, А.Ю. Берко // Вісник Національного університету "Львівська політехніка". – 2015. – № 832: Інформаційні системи та мережі. – С. 165-177.

38. Кушнірецька І.І. Проектування системи підтримки прийняття рішень виборця у вигляді веб-сайта / І.І. Кушнірецька, О.М. Верес // Вісник Національного університету "Львівська політехніка". – 2012. – № 743 : Інформаційні системи та мережі. – С. 44–53.

39. Кушнірецька І.І. Семантичний пошук і зберігання даних науково-технічної інформаційної системи / О.І. Кушнірецька, І.І. Кушнірецька, А.Ю. Берко // Вісник Національного університету "Львівська політехніка". – 2015. – № 814: Інформаційні системи та мережі. – С. 310-319.

40. Кушнірецька І.І. Система підтримки прийняття рішень для виборця з використанням фасетного методу класифікації / І.І. Кушнірецька, Л.В. Чирун // Вісник Національного університету «Львівська політехніка». – 2011. - № 699: Інформаційні системи та мережі. - С. 144-153.

41. Сповіщення. Слідкуйте за новим цікавим вмістом в Інтернеті [Електронний ресурс] / Google Inc. // Режим доступу: URL: https://www.google.com/alerts.

42. Хаав Х.-М.Х. Единый язык описания моделей данных / Х.-М.Х. Хаав // Финансы и статистика, Прикладная інформатика, № 2, Москва, 1986. - С. 130-142.

43. Черняк Л. EDA как очередная инкарнация SOA / Л. Черняк // Открытые системы №9, 2006.

44. Черняк Л. Интеграция данных: синтаксис и семантика / Л. Черняк // Открытые системы №10, 2009.

45. Черняк Л. Сервисы и сложные системы / Л. Черняк // Открытые системы №10, 2007.

46. Abiteboul S. Modeling the mashup space / S. Abiteboul, O. Greenshpan, T. Milo // In *WIDM*, 2008. – P. 87-94.

47. About IFTTT [Електронний ресурс] / IFTTT Inc. // Режим доступу: URL: https://ifttt.com/wtf.

48. About Pipes [Електронний ресурс] / Yahoo! Inc. // Режим доступу: URL: http://pipes.yahoo.com/pipes

49. Altinel M. Damia: a data mashup fabric for intranet applications / M. Altinel, P. Brown, S. Cline, R. Kartha, E. Louie, V. Markl, L. Mau, Y.-H. Ng, D. Simmen, A. Singh // In *VLDB '07*, VLDB Endowment, 2007. – P. 1370-1373.

50. Amer-Yahia S. What does web 2.0 have to do with databases? / S. Amer-Yahia, A. Y. Halevy // In *VLDB*, 2007. – P. 1443-1444.

51. Anjomshoaa A. Combining and integrating advanced it-concepts with semantic web technology mashups architecture case study / A. Anjomshoaa, A.M. Tjoa, A. Hubmer // In: Nguyen, N.T., Le, M.T., Świątek, J. (eds.) ACIIDS 2010. LNCS, vol. 5990, Springer, Heidelberg, 2010. – P. 13-22.

52. ANSI/NISO Z39.50-1995. Information Retrieval (Z39.50): Application Service Definition and Protocol Specification.

53. Automate your Dropbox [Електронний ресурс] / Wappwolf Inc. // Режим доступу: URL: http://wappwolf.com/dropboxautomator.

54. Baader F. The Description Logic Handbook: Theory, Implementation and Applications / F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P.F. Patel-Schneider // Cambridge University Press. – 2003. – P. 132-136.

55. Barbosa L. Siphoning Hidden-Web Data through Keyword-Based Interfaces / L. Barbosa, J. Freire // In Proc. Of SBBD, 2004

56. Barlow K. Like Technology from an Advanced Alien Culture: Google Apps for Education at ASU / K. Barlow, J. Lane // In Proceedings of Special Interest Group on University Computing Centers, Indiana, 2007. – P. 8-10.

57. Baumgartner R. Interactively adding web service interfaces to existing web applications / R. Baumgartner, G. Gottlob, M. Herzog, and W. Slany // Int. Symposium on Applications and the Internet, 2004. – P. 74–80.

58. Bizer C. Linked Data: Principles and State of the Art / C. Bizer, T. Heath et al. // Proceedings of the 3rd IEEE International Conference on WWW, 2008.

59. Blake M.B. A Web Service Recommender System using Enhanced Syntactical Matching / M.B. Blake, M.F. Nowlan // International Conference on Web Services (ICWS 2007), 2007.

60. Blake M.B. The EEE-05 Challenge: A New Web Service Discovery and Composition Competition / M.B. Blake, K.C. Tsui, A. Wombacher // Proceedings of the IEEE International Conference on E-Technology, E-Commerce, and E-Services, Hong Kong, 2005.

61. Bock C. UML 2 Composition Model / C. Bock // Journal of object technology. 2004 – N. 10(3) – P. 95–97.

62. Bosca A. Composing Web Services on the Basis of Natural Language Requests / A. Bosca, A. Ferrato, D. Corno et al. // Proceedings of the 3rd IEEE International Conference on Web Services (ICWS 2005), Orlando, Fl, 2005. – P. 817-818.

63. Brin S. Extracting patterns and relations from the world wide web / S/ Brin // In WebDB Workshop at 6th Intl. Conf. On Extending Database Technology, 1998.

64. Brusilovsky P. The Adaptive Web, Methods and Strategies of Web Personalization / P. Brusilovsky, A. Kobsa, W. Nejdl //volume 4321 of Lecture Notes in Computer Science. Springer, 2007.

65. Canfora G. Migrating interactive legacy systems to web services. / G. Canfora, A.R. Fasolino, G. Frattolillo, and P. Tramontana // IEEE CS Press, editor, European Conference on Software Mainteinance and Reengineering, 2006. – P. 23–32.

66. Carlson M.P. Automatic Mash Up of Composite Applications / M. P. Carlson, A. H.H. Ngu et al. // In Int. Conf on Service-Oriented Computing (ICSOC-08), 2008.

67. Cetin S. A Mashup-Based Strategy for Migration to Service-Oriented Computing / S. Cetin, N. I. Altintas, H. Oguztuzun, A. Dogru, O. Tufekci, S. Suloglu // IEEE International Conference on Pervasive Services, 2007.

68. Chang C. IEPAD: Information extraction based on pattern discovery / C. Chang, S. Lui // In Proc. of Intl. World Wide Web Conf., 2010. – P. 681-688.

69. Cheng T. C.-C.: Entity Search Engine: Towards Agile Best-Effort Information Integration over the Web / T. Cheng, K. Chang // In Proc. of CIDR, 2007.

70. Chiali S. A Dynamic Composition of an Adaptive Course / S. Chiali, Z. Eberrichi, M. Malki // In Proceedings of World Academy of Science, Engineering and Technology, USA, 2006. – P. 11-11.

71. Chong E. An efficient SQL based RDF querying scheme / E. Chong, S. Das, G. Eadon, J. Srinivasan // VLDB, 2005.

72. Computer science [Електронний ресурс] / Д. Барашев // Big Data’13. Лекция IX: поиск похожих документов. Режим доступу: URL: http://compscicenter.ru/sites/default/files/materials/2013_04_18_BigData_lecture_09.pdf.

73. Crescenzi V. ROADRUNNER: Towards automatic data extraction from large web sites / V. Crescenzi, G. Mecca, P. Merialdo // In Proc. of the 2001 Intl. Conf. on Very Large Data Bases, 2001. –P. 109–118.

74. Davis W. S. The Information System Consultant's Handbook: Systems Analysis and Design / W. S. Davis, D. C. Yen // CRC Press LLC, 2000, Florida. - P. 235-240.

75. Digital Libraries Initiative [Електронний ресурс] / NSF // Режим доступу: URL: http://www.dli2.nsf.gov.

76. Duda C. Ajaxsearch: crawling, indexing and searching web 2.0 applications / C. Duda, G. Frey, D. Kossmann, C. Zhou // *PVLDB*, 2008. – P. 1440-1443.

77. Ehrig M. The semantic web: Research and applications / M. Ehrig, Y. Sure // Proc. 1st European Semantic Web Symposium. LNCS. Berlin: Springer, 2004. – P. 3053.

78. Ennals R. User-Friendly Functional Programming for Web Mashups / R. Ennals, D. Gay / / ICFP, ACM, 2007. – P. 223-234.

79. Euzenat J. Ontology matching / J. Euzenat J., P. Shvaiko // Berlin: Springer-Verlag Heidelberg, 2007. – P. 332.

80. Fielding R. T. Architectural Styles and the Design of Network-based Software Architectures / R. T. Fielding // диссертация. д.т.н., University of Southern California, Irvine, 2010.

81. Fisher T. An overview of current approaches to mashup generation / T. Fischer, F. Bakalov, A. Nauerz // Proceedings of the International Workshop on Knowledge Services and Mashups, 2009, P. 157-158.

82. Fischer T. Towards an Automatic Service Composition for Generation of User-Sensitive Mashups / T. Fischer, F. Bakalov, A. Nauerz // In LWA 2008 - Workshop-Woche: Lernen, Wissen & Adaptivit¨at,Wu¨rzburg, Germany, 2008. – P. 157-158.

83. Fletcher W. Sharable Content Object Reference Model / W. Fletcher, M. Handwork, S. Herald // Technical Document, USA, 2004.

84. Fujima J. Clip Connect Clone: Combining Application Elements to Build Custom Interface for Information Access / J. Fujima // In Proceedings of User Interface Software and Technology, USA, 2004. – P. 175-184.

85. Fung B. C. M. Anonymizing classification data for privacy preservation / B. C. M. Fung, K. Wang, P. S. Yu // IEEE Transactions on Knowledge and Data Engineering (TKDE), 19(5), 2007. – P. 711-725.

86. Garofalokis M. XTRACT: A system for extracting document type descriptors from XML documents / M. Garofalokis, A. Gionis, R. Rastogi et al. // In Proc. of the 2000 ACM SIGMOD Intl. Conf. on Management of Data, 2000. – P. 165–176.

87. Godfrey M. Using Origin Analysis to Detect Merging and Splitting of Source Code Entities / M. Godfrey, L. Zou // IEEE Transactions on Software Engineering, vol. 31, 2005. - P. 166-181.

88. Gravano L. QProber: A system for automatic classification of hidden-Web databases / L. Gravano, P. Ipeirotis, M. Sahami // ACM Transactions on Information Systems, Volume 21 , Issue 1, 2003. – P. 1-41.

89. Gruber T. A Translation Approach to Portable Ontology Specifications / T. Gruber // Knowledge Acquisition, 1993. – P. 199-220.

90. Guarino N. The Role of Ontologies in Information Systems Design / N. Guarino // Proceedings of the First International Conference on Formal Ontologies, FOIS'08, 2008.

91. Guarino N. Formal ontology, conceptual analysis and knowledge presentation / N. Guarino // International Journal of Human and Computer Studies, 43(5/6). - P. 625−640.

92. Guarino N. The Ontological Level / N. Guarino // In: Casati R., Smith N. and White G.(eds.), Philosophy and the Cognitive Sciences, Vienna: Ho:lder-Pichler-Tempsky, 2004. – P. 115-119.

93. Guo H. Wrapping client-server application to web services for internet computing / H. Guo H., C. Guo et al. // International Conference on Parallel and Distributed Computing, Applications and Technologies, 2005. – P. 366–370.

94. Gurram R. A Web Based Mashup Platform for Enterprise 2.0 / R. Gurram, B. Mo et al. // In Web Information Systems Engineering Workshops, Volume 5176 of Lecture Notes in Computer Science, Springer-Verlag, 2008. - P. 141-151.

95. Halpin H. When owl:sameAs isn't the Same: An Analysis of Identity Links on the Semantic Web / H. Halpin, I. Herman, P.J. Hayes, D. Wood, S. Decker // RDF Next Steps Workshop, 2010.

96. Heflin J. Searching the Web with SHOE / J. Heflin // Defense Technical Information Center, 2010.

97. Hendrik. Towards Semantic Mashup Tools for Big Data Analysis / Hendrik, A. Anjomshoaa, A. Tjoa // Information and Communication Technology, Volume 8407, 2014. – P. 129-138.

98. Hoyer V. Market Overview of Enterprise Mashup Tools / V. Hoyer, M. Fischer // In International Conference on Service oriented Computing, Volume 5364 of Lecture Notes in Computer Science, Springer-Verlag, 2008. – P. 708-721.

99. Huynh D.F. Potluck: Data Mash-Up Tool for Casual Users / D. F. Huynh, R. C. Miller, D. R. Karger // In ISWC/ASWC, volume 4825 of Lecture Notes in Computer Science, Springer, 2007. – P. 239-252.

100. IDEF5 Ontology Description Capture method [Електронний ресурс]. Режим доступу: URL: http://www.idef.com/IDEF5.html, 2006.

101. ISO 15926-7: Data integration, sharing, exchange, and hand-over between computer systems. Part 7: Implementation methods for the integration of distributed systems: Template methodology, 2007.

102. Jackson C. Subspace: secure crossdomain communication for web mashups / C. Jackson, H.J. Wang // Proceedings of the 16th International Conference on World Wide Web, Banff, Canada, ACM Press, 2007. - P. 611-620.

103. Jhingran A. Enterprise information mashups: integrating information, simply / A. Jhingran // Proceedings of the 32nd international conference on Very Large Databases, 2006. – P. 3-4.

104. Jiang Y. Towards reengineering web sites to web-services providers / Y. Jiang and E. Stroulia // IEEE CS Press, editor, European Conference on Software Mainteinance and Reengineering, 2004. – P. 296–305.

105. Kalinichenko L.A. Infrastructure of the subject mediating environment aiming at semantic interoperability of heterogeneous digital library collections / L.A. Kalinichenko, D.O. Briukhov, N.A. Skvortsov, V.N. Zakharov // The Second All-

Russian Scientific Conference "Digital Libraries: Advanced Methods and Technologies, Digital Collections", Protvino, 2000. – P. 78-90.

106. Kaufmann E. How Useful Are Natural Language Interfaces to the Semantic Web for Casual End-Users / E. Kaufmann, A. Bernstein // ISWC, 2007.

107. Keukelaere D. F. SMash: Secure Component Model for Cross-Domain Mashups on Unmodified Browsers / D. F. Keukelaere, S. Bhola M. Steiner, S. Chari, S. Yoshihama // In Proceeding of the 17th International Conference on World Wide Web, Beijing, China, ACM Press, 2008. – P. 535-544.

108. Khizhnyak A. Apatar Connector Guide [Електронний ресурс] / A. Khizhnyak // Website, 2008. Режим доступу: URL: http://www.apatarforge.org/wiki/display/GUI/Apatar+Connector+Guides.

109. Koehler J. Web Service Composition: Current Solutions and Open Problems / J. Koehler, B. Srivastava // Proceedings of the Workshop on Planning for Web Services in conjunction with ICAPS03, 2003.

110. Kushmerick N. Wrapper induction for information extraction / N. Kushmerick, D. Weld, R. Doorenbos // In Proc. of the 1997 Intl. Joint Conf. on Artificial Intelligence, 1997. – P. 729–737.

111. Kushniretska I. I. Application of Mash-Up Technology for Dynamic Integration of Semi-Structured Data. / I. I. Kushniretska, A. Y. Berko // Матеріали VI Міжнародної конференції молодих вчених CSE-2013. – Львів, Видавництво НУ “Львівська політехніка”, 2013.

112. Kushniretska I.I. Application of faceted classification in a decision support system for voter / I.I. Kushniretska, O.M. Veres // Матеріали VII Міжнародної науково-технічної конференції «Computer Sciences and Information Technologies» **(**CSIT’2012**)**. — Львів, Видавництво НУ «ЛП», 2012 С. 181-182.

113. Kushniretska I.I. Designing of Structural Ontological Data Systems Model for Mash-UP Integration Process / I.I. Kushniretska, O.I. Kushniretska, A.Y. Berko // Applied Computer Science. - Volume 11, No. 1, Poland, 2015. - P. 39-50.

114. Kushniretska I.I. Forming of the Semistructured Data Dynamic Integration Mash-Up System Content / I.I. Kushniretska // Applied Computer Science. - Volume 11, No. 3, Poland, 2015. - P. 34-44.

115. Kushniretska, I.I. The ontological model of knowledge of scientific and technical information system / I.I. Kushniretska, O.I. Kushniretska, A.Y. Berko // Матеріали 9-ї Міжнародної науково-технічної конференції «Computer Sciences and Information Technologies» **(**CSIT'2014**)**. — Львів: Видавництво НУ «ЛП», 2014. - P. 47-48.

116. Kushniretska, I.I. The principles of Integration the Semi-Structured Data in Web-Systems using Ontologies / I.I. Kushniretska, A.Y. Berko // Матеріали VIII Міжнародної науково-технічної конференції «Computer Sciences and Information Technologies» **(**CSIT'2013**)**. — Львів: Видавництво НУ «Львівська політехніка», 2013. – P. 50-51.

117. Levy A.Y. Logic-Based Techniques in Data Integration / A. Y. Levy // Logic Based Artificial Intelligence. Kluwer Publishers, 2000. – P. 74-76.

118. Li S. Mashup: a New Way of Providing Web Mapping and GIS Services / S. Li, J. Gong // In ISPRS Congress Beijing, Proceedings of Commission IV, 2008.- P. 639-649.

119. Liu X. Towards Service Composition Based on Mashup / X. Liu, Y. Hui, W. Sun, H. Liang // IEEE Congress on Services, 2007. – P. 332-339.

120. Lorenzo G.D. Data integration in mashups / G.D. Lorenzo, H. Hacid, H.-Y. Paik, B. Benatallah // SIGMOD Record, vol. 38, no. 1, 2009. – P. 59-66.

121. Lorenzo G. Mashups for data integration: An analysis / G. Lorenzo, H. Hacid, H. Paik, B. Benatallah // Technical Report UNSW-CSE-TR-0810, 2008.

122. Machanavajjhala A. ℓ-diversity: Privacy beyond k-anonymity / A. Machanavajjhala, D. Kifer, J. Gehrke et al. // ACM TKDD, 1(1), 2007.

123. Maedche A. Clustering Ontology-Based Metadata in the Semantic Web / A. Maedche, V. Zacharias // Proc. 6th European PKDD Conf. LNCS, vol. 2431, Berlin: Springer, 2002. − P. 348-360.

124. Manolescu I. Answering XML Queries over Heterogeneous Data Sources / I. Manolescu et al. // Proc. Of the 27th VLDB Conference, Roma, Italy, 2011.

125. Maraikar Z. Building Mashups for The Enterprise with SABRE / Z. Maraikar, A. Lazovik et al. // In Int. Conf on Service-Oriented Computing (ICSOC-08), 2008.

126. Maximilien E. M. A domain-specific language for web apis and services mashups / E. M. Maximilien, H. Wilkinson, N. Desai, and S. Tai // ICSOC'07. - Berlin, Heidelberg, 2007. – P. 13–26.

127. Merrill D. Mashups: The new breed of Web app / D. Merrill // Website, 2006. Режим доступу: URL: http://www.ibm.com/developerworks/library/xmashups.html.

128. Michi H. The Rise and Fall of CORBA / H. Michi // ACM Queue, 4(5), 2006.

129. Nguyen H.A. Thesis for the Degree Master of Science / H. A. Nguyen // University of Houston−Clear Lake, 2006.

130. Nie Z. Object-level Vertical Search / Z. Nie, J.-R. Wen, W.-Y. Ma // In Proc. of CIDR, 2007.

131. Ort E. Mashup Styles, Part 1: Server-Side Mashups / E. Ort, S. Brydon, M. Basler // Sun Microsystems, 2007.

132. OWL Web Ontology Language [Електронний ресурс]. Режим доступу: URL: http://www.w3.org/TR/owl-features/.

133. OWL Test cases [Електронний ресурс] / OWL Web Ontology Language Test Cases // Режим доступу: URL: http://www.w3.org/TR/owl-test/.

134. Papazoglou M. Service-oriented computing: Concepts, characteristics and directions / M. Papazoglou // In Proceedings of WISE '03, 2003.

135. Parent C. About Complex Entities, Complex Objects and Object-Oriented Data Models / C. Parent, S. Spaccapietra // Info. System Concepts, 1989.

136. Protege. [Електронний ресурс]. Режим доступу: URL: protege.stanford.edu/download/registered.html.

137. Rada R. Development and Application of a Metric of Semantic Nets / R. Rada, H. Mili, E. Bicknell et al. // IEEE Trans. on Systems, Man and Cybernetics, vol. 19 (1), 1989. - P. 17-30.

138. Rahm E. A survey of approaches to automatic schema matching / E. Rahm, P. A. Bernstein // The VLDB Journal, 10(4):334–350, 2011.

139. Rahm E. Dynamic Fusion of Web Data / E. Rahm, A. Thor, D. Aumueller // In XSym, 2007. – P. 14-16.

140. Recchia G. A Comparison of String Similarity Measures for Toponym Matching // G. Recchia, M. Louwerse // Proceedings of The First ACM SIGSPATIAL International Workshop on Computational Models of Place. – 2013. – P. 54-62.

141. Rodrerguez M. Determining Semantic Similarity among Entity Classes from Different Ontologies / M. Rodrerguez, M. Egenhofer // IEEE Trans. On Knowledge and Data Engineering, (15), (2) 2003. – P. 442-456.

142. Sabbouh M. Web Mashup Scripting Language / M. Sabbouh, J. Higginson, S. Semy, D. Gagne // In WWW '07: Proceedings of the 16th Intl. Conf. on World Wide Web, ACM, New York, USA, 2007. – P. 1305-1306.

143. Schwarzkopf E. Towards intuitive interaction for end-user programming / E. Schwarzkopf, M. Bauer, D. Dengler // In IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces, ACM, New York, USA, 2003. – P. 287-289.

144. Sheth A. P. SAREST: Semantically Interoperable and Easier-to-Use Services and Mashups / A.P. Sheth, K. Gomadam, J. Lathem // IEEE Internet Computing, IEEE Educational Activities Department, 11(6), 2007. - P. 91-94.

145. Shevertalov M. A Case Study on the Automatic Composition of Network Application Mashups / M. Shevertalov // In ASE, IEEE, 2008. – P. 359-362.

146. Simmen D.E. Damia: data mashups for intranet applications / D. E. Simmen, M. Altinel, V. Markl, A. Singh // In SIGMOD Conference, ACM, 2008. – P. 1171-1182.

147. Sure Y. OntoEdit: Collaborative ontology development for the Semantic Web / Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer, D. Wenke // In Proc. of the Inter. Semantic Web Conference (ISWC 2002), Sardinia, Italia, 2002.

148. Sugiura A. Internet Scrapbook: Automating Web Browsing Tasks by Demonstration / A. Sugiura, Y. Koseki // In ACM Symp. on User Interface Software and Technology, 2008. – P. 9-18.

149. Thor A. Data Integration Support for Mashups / A. Thor, D. Aumueller, E. Rahm // Sixth Int. Workshop on Information Integration on the Web, IIWeb, Vancouver, Canada, 2007.

150. Thor A. Instance-based matching of hierarchical ontologies / A. Thor, T. Kirsten, E. Rahm // In Proc. of BTW, 2007.

151. Thor A. MOMA - A Mapping-based Object Matching System / A. Thor, E. Rahm // In Proc. of CIDR, 2007.

152. Tuchinda R. Building Mashups by example / R. Tuchinda, P. A. Szekely, C. A. Knoblock // Intelligent User Interfaces, ACM, 2008. – P. 139-148.

153. Uschold M. Ontologies: Principles, Methods and Applications / M. Uschold, M. Gruninger // Knowledge Engineering Review, 11(2), 1996.

154. Vikram K. Mashup component isolation via server-side analysis and instrumentation / K. Vikram, M. Steiner // In Web 2.0 Security & Privacy Workshop. IEEE Computer Society, Technical Committee on Security and Privacy, 2007.

155. Wache H. Ontology-Based Integration of Information. A Survey of Existing Approaches [Електронний ресурс] / H.Wache, T. Vogele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, S. Hubner // Режим доступу: URL: www.let.uu.nl/~Paola.Monachesi/personal/papers/wache.pdf, 2001.

156. Wagner G. The Agent-Object-Relationship Metamodel: Towards a Unified View of State and Behavior Information Systems G. Wagner, 2013.

157. Wong J. Making Mashups with Marmite: Towards End-User Programming for the Web / J. Wong, J. I. Hong // In Mary Beth Rosson and D. J. G., editors, CHI, ACM, 2007. – P. 1435-1444.

158. Wu P. Query Selection Techniques for Efficient Crawling of Structured Web Sources / P. Wu, J.-R. Wen, H. Liu, W.-Y. Ma // In Proc. of ICDE, 2006.

159. Zapier [Електронний ресурс] / Zapier Inc. // Режим доступу: URL: https://zapier.com.

160. Ziegler P. Three decades of data integration - All problems solved? / P. Ziegler, K.R. Dittrich // World Computer Congress - IFIP. – 2014. – P. 3-12.

ДОДАТОК А.
АКТИ ВПРОВАДЖЕННЯ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЙНОГО ДОСЛІДЖЕННЯ

ЗАТВЕРДЖУЮ
Проректор з наукової роботи
Національного університету
«Львівська політехніка»
д.е.н., професор Чухрай Н.І.
«02» 11 2015 р.

А К Т

про використання наукових результатів дисертаційної роботи Кушнірецької Ірини Ігорівни, представленої на здобуття наукового ступеня кандидата технічних наук, при виконанні науково-дослідної роботи кафедри інформаційних систем та мереж Національного університету «Львівська політехніка» за темою «Розроблення інтелектуальних розподілених систем на основі онтологічного підходу з метою інтеграції інформаційних ресурсів» (номер державної реєстрації 0115U004228)

Комісія у складі: голови комісії – начальника науково-дослідної частини, к.т.н., доцента Жук Л.В. та членів комісії – завідувача кафедри інформаційних систем та мереж, д.т.н., професора Литвина В.В., завідувача відділу науково-організаційного супроводу наукових досліджень, к.т.н. Лазько Г.В. та заступника начальника планово-фінансового відділу Чулой Т.М., цим актом підтверджують, що результати дисертаційної роботи асистента кафедри інформаційних систем та мереж Кушнірецької Ірини Ігорівни на тему «Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах» використано при виконанні науково-дослідної роботи кафедри інформаційних систем та мереж Національного університету «Львівська політехніка» за темою «Розроблення інтелектуальних розподілених систем на основі онтологічного підходу з метою інтеграції інформаційних ресурсів» (№ держреєстрації 0115U004228), у 2015 році.

Отримані автором результати використано:

- при розробленні структури Mash-Up системи динамічної інтеграції слабоструктурованих даних;
- при розробленні методу визначення структури і змісту отриманої вхідної інформації;
- при розробленні методу формування контенту об'єднаного динамічного набору даних, який має загальну структуру і єдиний зміст;
- при здійсненні концептуалізації та моделюванні роботи системи динамічної інтеграції слабоструктурованих даних у web-системах, що працює використовуючи технологію Mash-Up.

Голова комісії:
Начальник науково-дослідної частини
к.т.н., доцент Л.В. Жук

Члени комісії:
Завідувач кафедри інформаційних
систем та мереж, д.т.н., професор В.В. Литвин

Зав. відділу науково-організаційного
супроводу наукових досліджень, к.т.н. Г.В. Лазько

Заст. начальника
планово-фінансового відділу Т.М. Чулой

«ЗАТВЕРДЖУЮ»
Проректор з науково-педагогічної роботи
Національного університету
«Львівська політехніка»
_____ **О.Р. Давидчак**
"05" _____ 2015 р.

АКТ

про впровадження в навчальний процес результатів
кандидатської дисертаційної роботи
Кушнірецької Ірини Ігорівни

Цей акт складено про те, що результати кандидатської дисертаційної роботи Кушнірецької Ірини Ігорівни на тему «Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах», представленої на здобуття наукового ступеня кандидата технічних наук, використовуються у навчальному процесі кафедри «Інформаційні системи та мережі» Національного університету «Львівська політехніка». Матеріали дисертаційного дослідження використовуються під час написання студентами курсових робіт, кваліфікаційних бакалаврських та магістерських робіт, а також під час викладання дисципліни «Об'єктно-орієнтоване програмування».

Зокрема, у навчальному процесі використовуються запропоновані І.І. Кушнірецькою:

- принципи створення віртуальних функцій для забезпечення динамічного поліморфізму (дисципліна «Об'єктно-орієнтоване програмування» для студентів освітньо-кваліфікаційного рівня «бакалавр», що навчаються за напрямом 6.050101 «Комп'ютерні науки», тема 7 «Поліморфізм віртуальних методів»);
- методи моделювання діаграми класів з використанням множинного узагальнення класів (дисципліна «Об'єктно-орієнтоване програмування» для студентів освітньо-кваліфікаційного рівня «бакалавр», що навчаються за напрямом 8.05010104 «Системи штучного інтелекту», тема 15 «Діаграми класів»).

Директор ІКНІ,
д.т.н., професор М.О. Медиковський

Завідувач кафедри ІСМ,
д.т.н., професор В.В. Литвин

Професор кафедри ІСМ, д.т.н. Р.М. Камінський

ТОРГОВА ГРУПА
ТИСА
Офіційний представник Ужгородського коньячного заводу

Україна, 79035, м. Львів, вул. Зелена, 115б (юридична адреса)
Україна, 79026, м. Львів, вул. Лазаренка, 6Б (поштова адреса)

Телефон : (032) 240-54-40,
тел./факс: (032) 242-87-73

№_____від_____20__ р.

на №_____

А К Т

впровадження результатів дисертаційного дослідження Кушнірецької Ірини Ігорівни за темою "Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах"

Цей акт підтверджує, що результати кандидатського дисертаційного дослідження Кушнірецької Ірини Ігорівни за темою "Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах" використовуються у ТОВ «Торгова група «Тиса» для розроблення стратегій покращення якості продукції та визначення можливих напрямків розвитку підприємства за рахунок введення на ринок найбільш популярних новітніх продуктів виробництва.

Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах, автором якої є Кушнірецька І.І., дає змогу при її застосуванні ефективніше отримувати і використовувати дані із декількох різноманітних джерел, враховуючи при цьому зміст інформації. Дисертантом складено та сформовано повний комплект вимог для використання розробленої технології та інструкції по користуванню системою із впровадженою інформаційною технологією динамічної інтеграції слабоструктурованих даних у web-системах.

Впровадження інформаційної технології динамічної інтеграції слабоструктурованих даних у web-системах у проекті моніторингу новин ринку збуту продукції деревообробної промисловості, що розроблений для ТОВ «Торгова група «Тиса» на основі використання методів Кушнірецької І.І. дає змогу підприємству завжди отримувати достовірну та максимально «свіжу» інформацію щодо роботи галузі, яка подана у зручній формі для розроблення стратегій покращення якості продукції та введення на ринок нових продуктів, відповідно до сучасних потреб користувачів.

Генеральний директор ТОВ
«Торгова група «Тиса» С.В. Тарновецький
"08" 12 2015 р.

А К Т

впровадження результатів дисертаційної роботи «Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах»

Кушнірецької Ірини Ігорівни,

асистента кафедри інформаційних систем та мереж Національного університету «Львівська політехніка»

Цей акт підтверджує, що основні теоретичні та практичні результати дисертаційної роботи Кушнірецької І.І. за темою «Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах», представленої на здобуття наукового ступеня кандидата технічних наук, використовуються у стоматологічній клініці «Улюблений доктор» для удосконалення її роботи.

Розроблена Кушнірецькою І.І. інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах дає змогу при її застосуванні ефективніше отримувати і використовувати дані із декількох різноманітних джерел, враховуючи при цьому зміст інформації. Дисертантом складено та сформовано повний комплект вимог для використання розробленої технології та інструкції по користуванню системою із впровадженою інформаційною технологією динамічної інтеграції слабоструктурованих даних у web-системах.

Створені Кушнірецькою І.І. у процесі дослідження методи та програмні засоби динамічної інтеграції слабоструктурованих даних у web-системах використовуються при опрацюванні даних, отриманих із декількох різних джерел для пошуку новаторських рішень в імплантології і в стоматології загалом, для аналізу і опису 3D моделей, отриманих рентгенологічними дослідженнями світової практики, для вивчення методів використання парадонтологічних протоколів у світовій практиці, а також при веденні статистичних даних по обліку відвідувань стоматологічних хворих.

Це впровадження сприяє забезпеченню роботи клініки «Улюблений доктор» у відповідності до сучасних вимог.

Власник клініки,
доктор ортопед- стоматолог З.Я.Пігович

«23» 10 2015 р.

Печатка

КЗ ЛОР «ЛЬВІВСЬКЕ ОБЛАСНЕ ПАТОЛОГОАНАТОМІЧНЕ БЮРО»

А К Т

впровадження результатів дисертаційної роботи
“Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах”
Кушнірецької Ірини Ігорівни,
асистента кафедри інформаційних систем та мереж
Національного університету “Львівська політехніка”

Цей акт підтверджує, що основні теоретичні та практичні результати дисертаційної роботи Кушнірецької І.І. за темою “Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах”, представленої на здобуття наукового ступеня кандидата технічних наук, використовуються у КЗ ЛОР «Львівське обласне патологоанатомічне бюро» для опрацювання статистичних даних.

Розроблена Кушнірецькою І.І. інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах дає змогу при її застосуванні ефективніше отримувати і використовувати дані із декількох різноманітних джерел, враховуючи при цьому зміст інформації. Дисертантом складено та сформовано повний комплект вимог для використання розробленої технології та інструкції по користуванню системою із впровадженою інформаційною технологією динамічної інтеграції слабоструктурованих даних у web-системах.

Створені Кушнірецькою І.І. у процесі дослідження методи та програмні засоби динамічної інтеграції слабоструктурованих даних у web-системах мають можливість використовуватись для опрацювання даних, отриманих із декількох різних джерел, та опрацюванні статистичної інформації для роботи КЗ ЛОР «Львівське обласне патологоанатомічне бюро».

Начальник бюро Б.Й.Рібун

А К Т

впровадження результатів дисертаційного дослідження Кушнірецької Ірини Ігорівни за темою "Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах"

Цей акт підтверджує, що результати кандидатського дисертаційного дослідження Кушнірецької Ірини Ігорівни за темою "Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах" використовуються у ФОП Сидоряк Р.Б. для визначення можливих напрямків розвитку підприємства за рахунок розширення асортименту продукції та введення на ринок найбільш популярних новітніх продуктів виробництва.

Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах, автором якої є Кушнірецька І.І., дає змогу при її застосуванні ефективніше отримувати і використовувати дані із декількох різноманітних джерел, враховуючи при цьому зміст інформації. Дисертантом складено та сформовано повний комплект вимог для використання розробленої технології та інструкції по користуванню системою із впровадженою інформаційною технологією динамічної інтеграції слабоструктурованих даних у web-системах.

Впровадження інформаційної технології динамічної інтеграції слабоструктурованих даних у web-системах у проекті моніторингу новин галузі виробництва та продажу будівельних матеріалів, що розроблений для ФОП Сидоряк Р.Б. на основі використання методів Кушнірецької І.І. дало змогу підприємству завжди отримувати достовірну та максимально «свіжу» інформацію галузі виробництва та продажу будівельних матеріалів, яка подана у зручній формі для введення на ринок сучасних продуктів галузі виробництва та продажу будівельних матеріалів, відповідно до сучасних потреб користувачів.

Директор ФОП
Сидоряк Р.Б.

ФІЗИЧНА ОСОБА-ПІДПРИЄМЕЦЬ СИДОРЯК Роман Богданович 3037707956 УКРАЇНА м. ЛЬВІВ

Сидоряк Р.Б.

"15" листопада 2015р.

А К Т

впровадження результатів дисертаційного дослідження Кушнірецької Ірини Ігорівни за темою "Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах"

Цей акт підтверджує, що результати кандидатського дисертаційного дослідження Кушнірецької Ірини Ігорівни за темою "Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах" використовуються у ДП «Латориця» при виконанні процедури організації перевезення вантажів для залучення нових клієнтів та утримання вже існуючих.

Інформаційна технологія динамічної інтеграції слабоструктурованих даних у web-системах, автором якої є Кушнірецька І.І., дає змогу при її застосуванні ефективніше отримувати і використовувати дані із декількох різноманітних джерел, враховуючи при цьому зміст інформації.

Впровадження методів опрацювання даних, розроблених Кушнірецькою І.І., дає змогу оптимізувати виконання процедури організації перевезення вантажів ДП «Латорицею», отримувати своєчасно інформацію щодо перевезення вантажів, бути в курсі актуальних новацій у сфері вантажоперевезень, логістики, тощо.

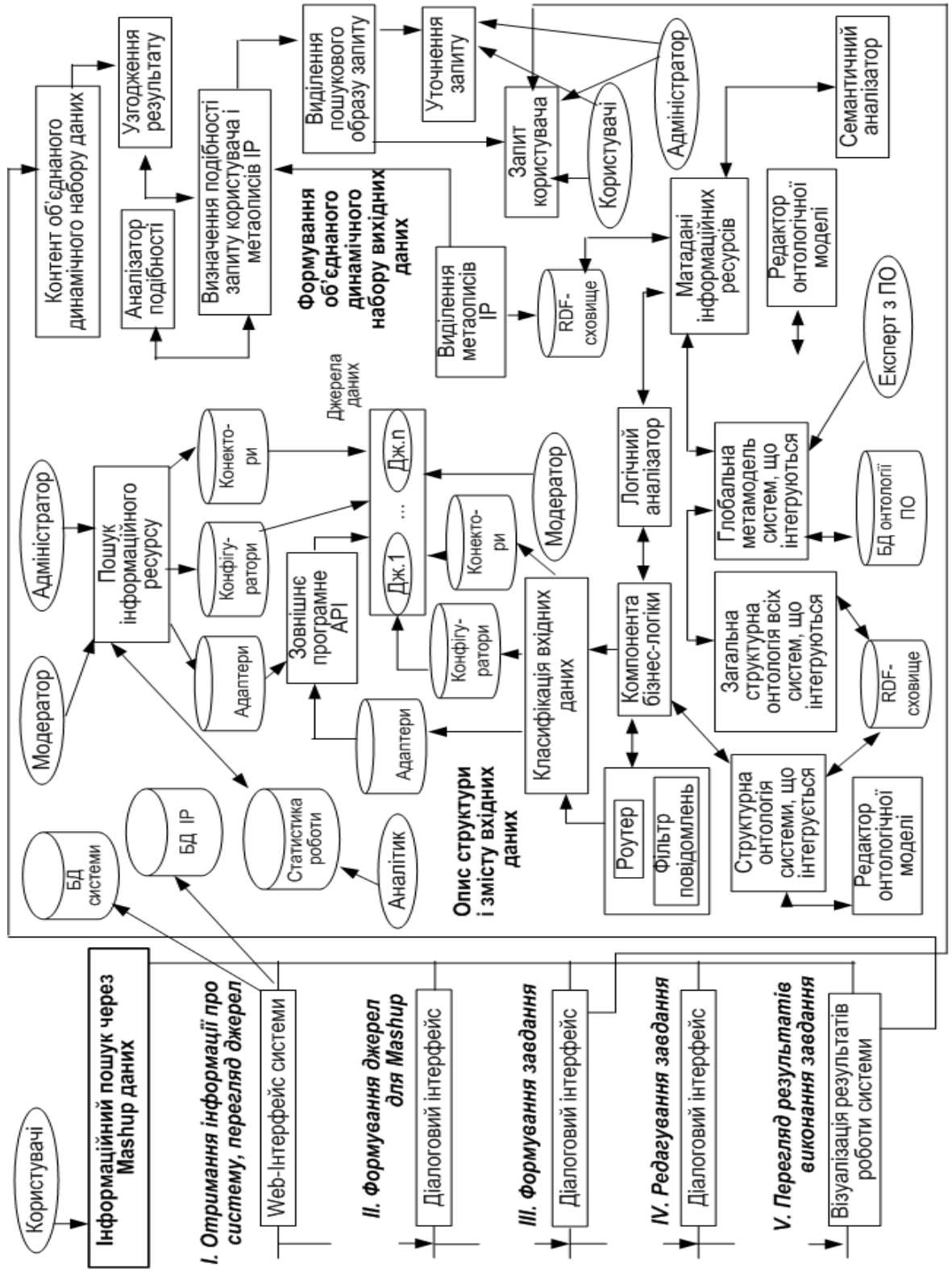
Директор ДП «Латориця» С.В. Тарновецький

ДП "ЛАТОРИЦЯ" №30698716

"16" вересня 2015 р.

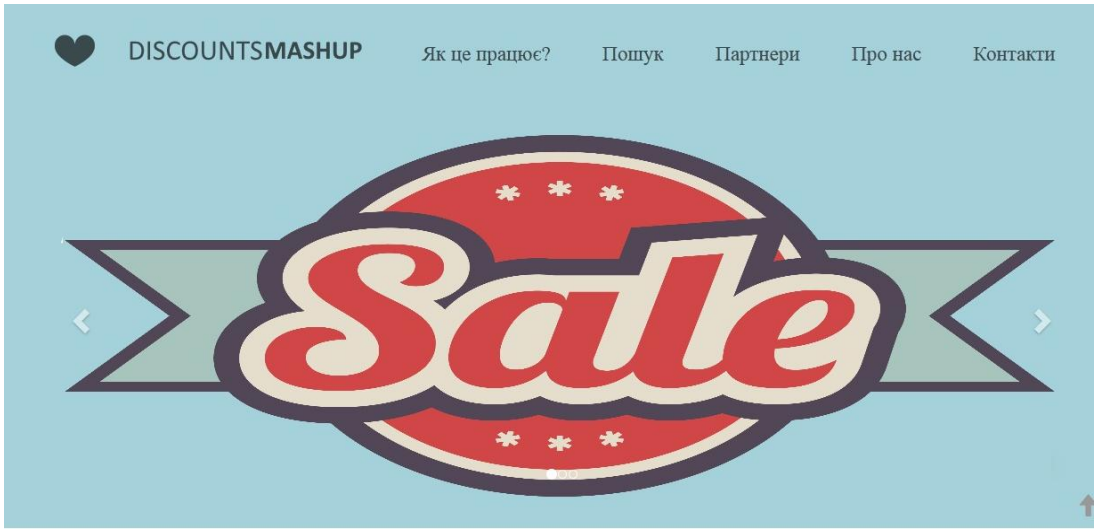
ДОДАТОК Б.

Схема процесу опрацювання даних у системах із впровадженою інформаційною технологією динамічної інтеграції слабоструктурованих даних у web-системах

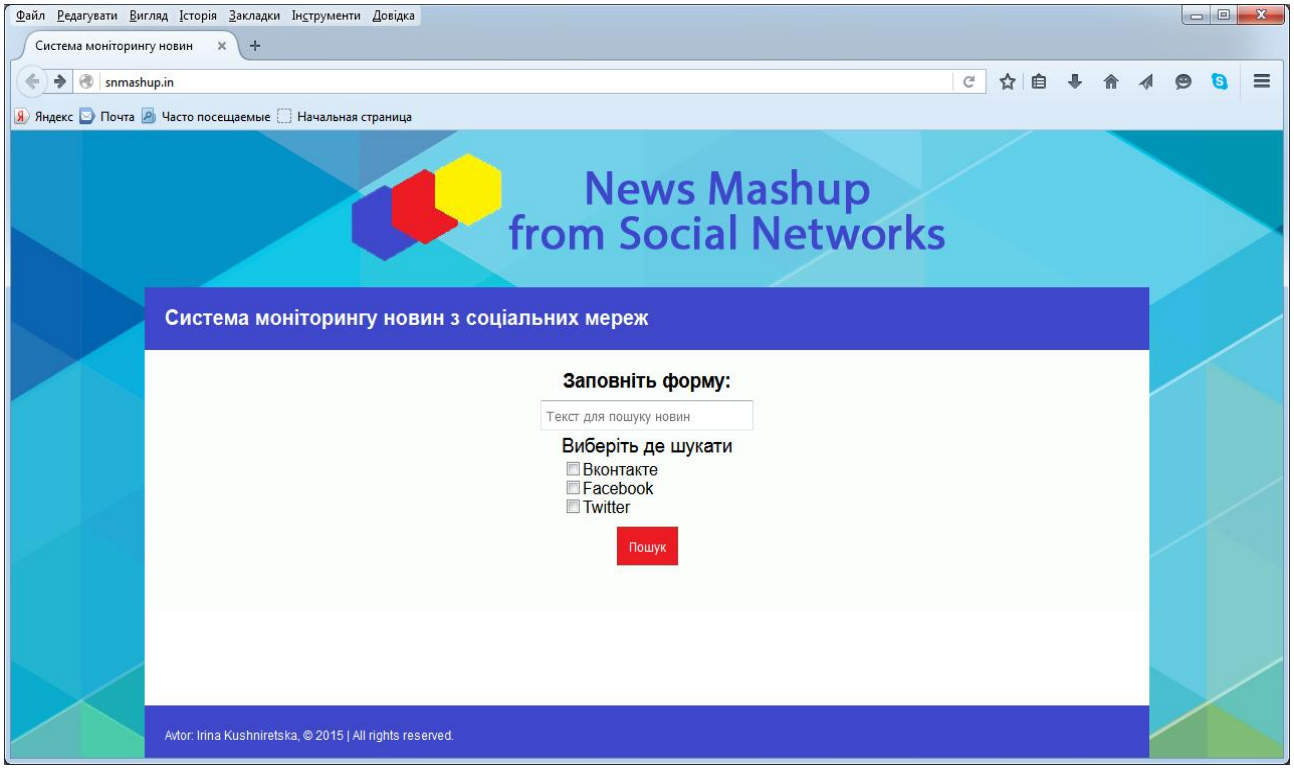


ДОДАТОК В.

Вигляд інтерфейсу системи Shares and Discounts Mashup



Вигляд інтерфейсу системи News Mashup from Social Networks



ДОДАТОК Г.

Порівняльний аналіз характеристик систем на основі інформаційної технології динамічної інтеграції слабоструктурованих даних із аналогами

Назва характеристики		IFTTT	Google Alerts	Shares and Discounts Mashup	News Mashup from Social Networks
Робота із вхідним та вихідним потоком даних					
Зіставлення даних	Ручне	+	-	-	-
	Напівавтоматичне	-	-	+	+
	Автоматичне	-	+	+	+
Опрацювання семантики	Присутнє	-	-	+	+
	Відсутнє	+	+	-	-
Якісні показники опрацювання результату	Точність	0,85	0,78	0,91	0,89
	Повнота	0,67	0,72	0,74	0,78
	Втрати інформації	0,33	0,28	0,26	0,22
	Інформаційний шум	0,15	0,22	0,09	0,11
	Узгодженість результатів, %	43,9	68,6	73,7	84,5
Оновлення даних	Стратегія витягнення даних	+	+	+	+
	Стратегія вміщення даних	-	-	+	+
	Глобальний інтервал витягнення даних	+	+	+	+
	Локальний інтервал витягнення даних	-	-	+	+
	Інтервал оновлення	+	+	+	+
Вихідні дані	Людино-орієнтовані	+	+	+	+
	Машино-орієнтовані	-	-	-	-