Хмельницький національний університет
Факультет інформаційних технологій
Кафедра кібербезпеки

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

Метод побудови композитних вебдодатків на основі інтеграції слабоструктурованих даних

Галузь знань 12 – Інформаційні технології

Спеціальність 123 –Комп'ютерна інженерія

КвРКІ. 170152.21.01.09 ПЗ

Виконав: студент 2 курсу, група КІ1м-21-1 Підпис Пічура В.Ю.

Керівник: к. т. н, доцент кафедри кібербезпеки Підпис Тітова В.Ю.

Нормоконтролер ст. викладач кафедри кібербезпеки Підпис Мостовий С.В.

До захисту допускаю:
Зав. кафедри кібербезпеки, к.т.н., доц Підпис Кльоц Ю.П.

7 12 _____ 2022 р.

Хмельницький, 2022

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет ІНФОРМАЦІЙНИХ ТЕХОЛОГІЙ

Кафедра КІБЕРБЕЗПЕКИ

Освітній рівень МАГІСТР

Галузь знань 12 ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Спеціальність 123 КОМП'ЮТЕРНА ІНЖЕНЕРІЯ

Освітня програма ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА ПІДГОТОВКИ МАГІСТРА

ЗАТВЕРДЖУЮ

Зав. кафедри Ю.П. Кльоц

"4" 09 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Пічурі В.Ю.

Прізвище, ім`я, по батькові студента

1. Тема проекту (роботи) Метод побудови композитних вебдодатків на основі інтеграції слабоструктурованих даних

2. Керівник проекту (роботи) к.т.н., доц. Тітова В.Ю.

Прізвище, ім`я, по батькові, науковий ступінь, вчене звання

Затверджена наказом № 83 ректора університету додаток №25до наказу від 01.07.2022 р.

2. Строк подання студентом проекту (роботи) на кафедру 3.12.2022

3. Вихідні дані до проекту (роботи) веб онтології, семантична мережа, фреймворки, Mashup системи

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)

Аналіз технологій інтеграції слабоструктурованих даних у вебзастосунки; розроблення семантичної моделі профілю системи збору інформації; досліджено процес композитної збірки вебдодатків; вдосконалено метод композитної збірки вебдодатків; запропоновано інформаційнутехнологію композитної збірки вебзастосунків

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень) Загальна характеристика кваліфікаційної роботи, класифікація рівнів інтеграції даних, традиційна модель на остнові технологі Mashup, алгоритм побудови онтологічної моделі, метод композитної збірки вебдодатків, алгоритми роботи та архітектура системи динамічної інтеграції компонентів, висновки

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання пр
Відповідальний за оформлення ДП	Мостовий С.В.		

7. Дата видачі завдання «1» лютого 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів (розділів) кваліфікаційної роботи	Термін виконання етапів проекту (роботи)	Пр
1	Вибір напряму дослідження та узгодження тематики КРМ з керівником	3.02.2022	
2	Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення предмета та об'єкта дослідження	3.03.2022	
3	Робота над розділом 1 – аналіз відомих моделей, методів за темою; постановка задачі	2.04.2022	
4	Робота над розділом 2 – розробка моделей і методів для вирішення поставленої задачі	2.05.2022	
5	Робота над науковою статтею	2.06.2022	
6	Робота над розділом 3 – розробка алгоритмів та технологій, їх аналіз	3.09.2022	
7	Робота над розділом 4 – проектування ПЗ для вирішення поставленої задачі	01.10.2022	
8	Узгодження отриманих результатів; оформлення пояснювальної записки згідно вимог	02.11.2022	
9	Оформлення графічної частини	12.11.2022	
10	Попередній захист КРМ	25.11.2022	
11	Захист ДРМ на засіданні ЕК	8.12.2022	

Студент Підпис Ю.Ю. Дідура Ініціали, прізвище

Керівник проекту (роботи) Підпис В.Ю. Тітова Ініціали, прізвище

АНОТАЦІЯ

Тема кваліфікаційної роботи: Метод побудови композитних вебдодатків на основі інтеграції слабоструктурованих даних

Автор роботи: Пічура В.Ю.

Керівник роботи: к.т.н., доц. Тітова В.Ю.

Пояснювальна записка: 76 с., 31 рис., 3 табл., 2 дод., 29 джерел.

ВЕБ ОНТОЛОГІЇ, СЕМАНТИЧНА МЕРЕЖА, ФРЕЙМВОРКИ, MASHUP СИСТЕМИ, МЕТАДАНІ.

Мета кваліфікаційої роботи полягає в розробці підходу до побудови композитних вебдодатків шляхом вдосконалення інформаційної технології адаптивної інтеграції слабоструктурованої інформації у вебсистемах.

Дана кваліфікаційна робота присвячена вдосконаленю методу композитної збірки вебдодатків, що відрізняється можливістю отримувати набори компонент по їх частковому опису.

28.11.2022

ANNOTATION

a master's degree work of Pichura Vadym

entitled «Method of building composite web applications based on the integration of loosely structured data».

Mentor: Vira Titova

Total volume of work: 76 pages, 31 figures, 3 tables, 2 appendices, 29 references.

WEB ONTOLOGIES, SEMANTIC NETWORK, FRAMEWORKS, MASHUP SYSTEMS, METADATA.

The purpose of the qualification work is to develop an approach to building composite web applications by improving the information technology of adaptive integration of loosely structured information in web systems.

This qualification work is devoted to the improvement of the method of composite assembly of web applications, which is distinguished by the possibility of obtaining sets of components based on their partial description.

28.11.2022

ЗМІСТ

ВСТУП

Кожна людина користується Інтернетом, не замислюючись про те, як він працює і що буде в майбутньому. Проте все розвивається, а тому з часом змінюватиметься і Інтернет. На даний момент ми використовуємо інтернет-версію Web 2.0, а найближчим часом будемо використовувати Web 3.0. Важливим аспектом Web 2.0 є зміна акцентів у використанні технологій на задоволенні потреб користувачів. До основних принципів Web 2.0 можна віднести право користувачів самостійно створювати контент, керувати зв’язками між своїми та чужими матеріалами.

Різниця між Web 2.0 і 3.0 полягає в тому, що Web 3.0 більше зосереджений на використанні таких технологій, як машинне навчання та штучний інтелект для доставки відповідного вмісту кожному користувачеві, а не лише на вміст, який надають інші кінцеві користувачі. Web 2.0 по суті дозволяє користувачам робити внесок, а іноді й співпрацювати над вмістом сайту, тоді як Web 3.0, ймовірно, передасть цю роботу Semantic Web і технологіям штучного інтелекту. Web 3.0 також має значний акцент на децентралізованих послугах і повноваженнях, що різко контрастує з централізацією Web 2.0.

На сьогодні системи інтеграції даних в основному працюють використовуючи технологію Mashup. Mashup - це технологія для розробки вебзастосунків, яка допомагає користувачам об'єднувати різнотипні дані з декількох джерел в одну інтегровану систему. Mashup — це програма, яка поєднує вміст із різних джерел, свого роду гібридний вебдодаток. Зараз широко використовується синдикація інформаційних каналів (обмін RSS-потоками даних), і це mashup. Іншим прикладом є використання Google Map на вашому вебсайті. Однак сьогодні серед лідерів ринку існує мода «бути відкритим». Все, що вам потрібно, це бажання інтегрувати будь-що на вашому сайті, і до ваших послуг величезний список інструментів для безкоштовного використання (A9, Amazon, BBC, Blogger, Bloglines, FeedMap, Flickr, MapPoint, eBay, YahooTraffic, YahooMaps, WeatherBug, NASA, hostip, GoogleMaps, Livejornal, Shopping.com, YouTube, Zillow та інші).

Розробка технології для динамічної інтеграції неструктурованих даних, яка здатна враховувати характеристики різних вхідних інформаційних систем є актуальною задачою, враховуючи такі фактори, як: недостатня теоретична обґрунтування методів семантичної обробки даних, а також необхідності уніфікації програмних засобів динамічної обробки інформаційних ресурсів інтегрованих систем. Практичний фактор обробки даних різних структур і форматів представлення, які зустрічаються в різноманітних вебсистемах з їх динамічною інтеграцією, пов'язаний із вирішенням проблем удосконалення пошуку інформації в мережі в сучасну епоху швидко зростаючого обсягу даних різного характеру та змісту, інтеграція різнорідних інформаційних систем, що містять інформаційні ресурси різного типу та змісту, а також величезний попит на системи динамічної інтеграції даних для Web 2.0 на основі технології Mashup, з перспективою їх виколристання з настанням ери Web 3.0. побудови композитних вебдодатків на основі інтеграції слабоструктурованих даних

Мета і завдання дослідження: побудова композитних вебдодатків шляхом вдосконалення інформаційної технології адаптивної інтеграції слабоструктурованої інформації у вебсистемах.

Для досягнення поставленої мети в кваліфікаційній роботі були вирішені наступні завдання:

– проаналізувано Semantic Web концепцію при побудові вебзастосунків, її переваги та недоліки;

– вдосконалено модель стандартизованого профілю системи збору інформації;

– запропоновано алгоритм побудови онтологічної моделі композитного вебзастосунку;

– розроблено семантичну модель профілю системи збору інформації;

– вдосконалено метод композитної збірки вебдодатків;

– удосконалено структурну модель системи композитної збірки, яка використовує технологію Mashup;

– досліджено пошук та вибір сервісів для компонування і реалізації розроблених алгоритмів композитної збірки вебдодатку.

Об’єктом дослідження є процес композитної збірки вебзастосунків на основі інтеграції слабоструктурованих даних.

Предмет дослідження є методи агрегації та впорядкування наборів слабоструктурованих даних.

Наукова новизна отриманих результатів:

– вдосконалено структурну модель стандартизованого профілю системи збору інформації на основі технології Mashup, що відрізняється від відомих доданням функціональних компонентів для визначення та опису семантичних процесів вхідних та вихідних наборів даних, що дає можливість підвищити якість результатів використання таких систем;

– вдосконалено метод композитної збірки вебдодатків, що відрізняється можливістю отримувати набори компонент по їх частковому опису.

Практична реалізація. В роботі розроблено інформаційну технологію та програмні застосунки, які дають можливість зменшення інформаційного шуму до 7-8 % та покращення узгодженості результатів у системі композитного формування вебзастосунку за технологією Mashup.

Публікації. За матеріалами кваліфікаціної роботи опубліковано 1 стаття та 1 теза.

1 АНАЛІЗ ТЕХНОЛОГІЙ ІНТЕГРАЦІЇ СЛАБОСТРУКТУРОВАНИХ ДАНИХ У ВЕБЗАСТОСУНКИ

1.1 Еволюція технології Web

Потік нових інформаційних технологій збільшується з кожним днем. Розвиваються послуги, побудовані на різноманітних сервісах Інтернету, використання різноманітних інтерактивних технології це вже вимушений крок вже для отримання серйозної конкурентної переваги [12].

Технологія Web 1.0 з'явилася в 1990 році і проіснувала до початку 2000-х.

Web 1.0: дозволяє користувачеві переглядати сторінки. Тобто взаємодії з ними немає, і всі сайти та інформаційні портали є звичайними статичними сайтами. Ви не можете залишити коментар або якось вплинути на те, що ви бачите, тому що тільки власник сайту контролює все [17].

Після 2000 року з'явилася нова версія технології, якою всі користуються досі, і це Web 2.0

Технологія Web 2.0 – на даний момент все ще актуальна і, швидше за все, ще довго буде поточною версією Інтернету. У цій версії користувачі мають можливість взаємодіяти з порталами, сайтами, тощо. Ви можете залишати коментарі, переглядати оголошення на різних форумах, формувати свій контент у соціальних мережах, впливати на рейтинг вебзастосунків і багато іншого [15].

Ця версія Інтернету дає користувачам можливість наповнювати сайти контентом на власний розсуд. Все це відбувається або на чистому ентузіазмі людей, або за винагороду у вигляді «впродобайок», донатів і за принципом дії – «де цікавіший контент, там більше користувачів».

Це цілком логічно, тому що ми всі намагаємося потрапити туди, де точаться дискусії і вирує життя.

Крім того, логічно очікувати оновлення функцій Інтернету у версії Web 3.0.

Web 3.0 – майбутнє Інтернету. З цією версією користувачі будуть виконувати всі ті ж дії, але отримуватимуть за це якийсь прибуток. Тобто сайти і так далі будуть мати взаємовигідні відносини з користувачами сайтів. Тепер

замість чистого ентузіазму користувачів «продвигати» сайт буде обмін з ними, наприклад, криптовалютою. Це чудова новина для багатьох користувачів, але не всі знають про такі можливості [7].

Розглянемо відмінності Web 2.0 та Web 3.0.

Дійсно, відмінностей багато. Однією з ключових є те, що версія 2.0 керується людським розумінням інформації в Інтернеті. А версія 3.0 рухається шляхом взаємодії та розуміння даних комп'ютерними системами. Це означає, що в цій версії користувачі будуть більш захищені, тому що штучний інтелект допоможе у всьому [15].

Web 2.0 – керується користувачами (рис.1.1). Власники сайтів створюють зручні умови для того, щоб користувачі залишалися на їх сайтах якомога довше в майбутньому. Користувачі, перебуваючи на сайтах, просувають його з ентузіазму, наприклад, з бажання показати друзям цікаве відео (YouTube) або транслювати його (Twitch). Але у ньому багато мертвих сторінок, і досить легко потрапити на шахрайство. Інтернет дуже завантажений через легкість створення вебзастосунків. Крім того, сайти розміщуються дуже дешево, тому будь-хто може створити власний сайт, чим і користуються багато шахраїв. Web 2.0 – вебпереглядачі та вебсайти містять багато реклами. Іноді це оголошення дуже дратує, і з цієї причини користувачі залишають сайт. Щоб заблокувати рекламу, вам потрібно завантажити блокувальники чи спеціальні плагіни у свій браузер.

Web 2.0 – браузери не забезпечують блокування файлів cookie. Саме завдяки цим файлам сайти запам'ятовують, що ви їх відвідували, і дані, які ви на них ввели [21]. Іноді ви не хочете, щоб хтось знав, які сайти ви відвідували, але саме файли cookie видають вас, оскільки їх не завжди можливо заблокувати.

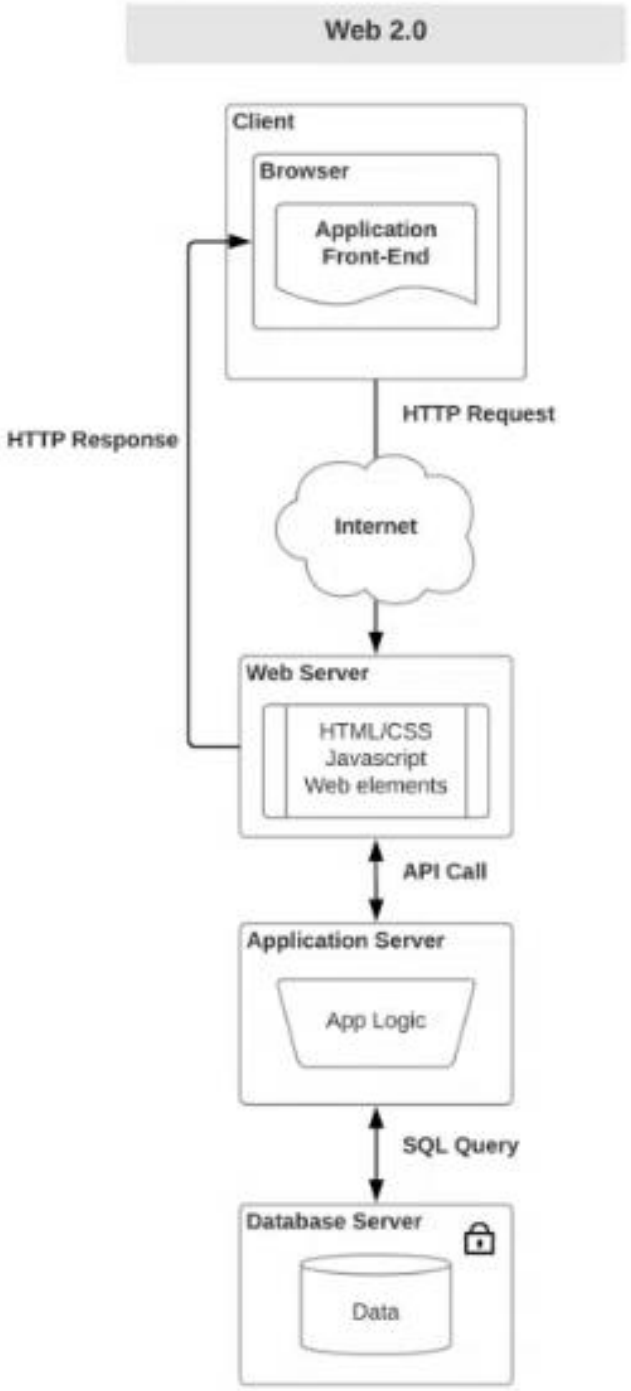


Рисунок 1.1 - Архітектура додатків Web 2.0

Web 3.0 – керується взаємодією користувача зі штучним інтелектом. Власники сайтів створюють умови, за яких користувачі будуть заробляти за те, що вони роблять. Наприклад, користувачі заходять на сайт і спільними зусиллями створюють там певний продукт, на цьому ж сайті цей продукт продається, і кожен, хто до нього долучався, отримує прибуток. Web 3.0. Очікується підвищена безпека,

і створювати веб-сайти буде нелегко. Так, ви все ще можете розміщувати свої сайти, але тепер штучний інтелект буде більш детально перевіряти всі сайти на предмет шахрайства. Завдяки цьому відбудеться підвищення рівня безпеки в Інтернеті. У Web 3.0 браузери мають вбудовані засоби блокування реклами, які приховають від ваших очей нав'язливу рекламу та дозволять вам переглядати Інтернет у приємній обстановці. У Web 3.0 присутнє блокування файлів cookie. Ви можете ввімкнути його на тих сайтах, де ви не хочете, щоб ваші дані залишалися (рис.1.2).

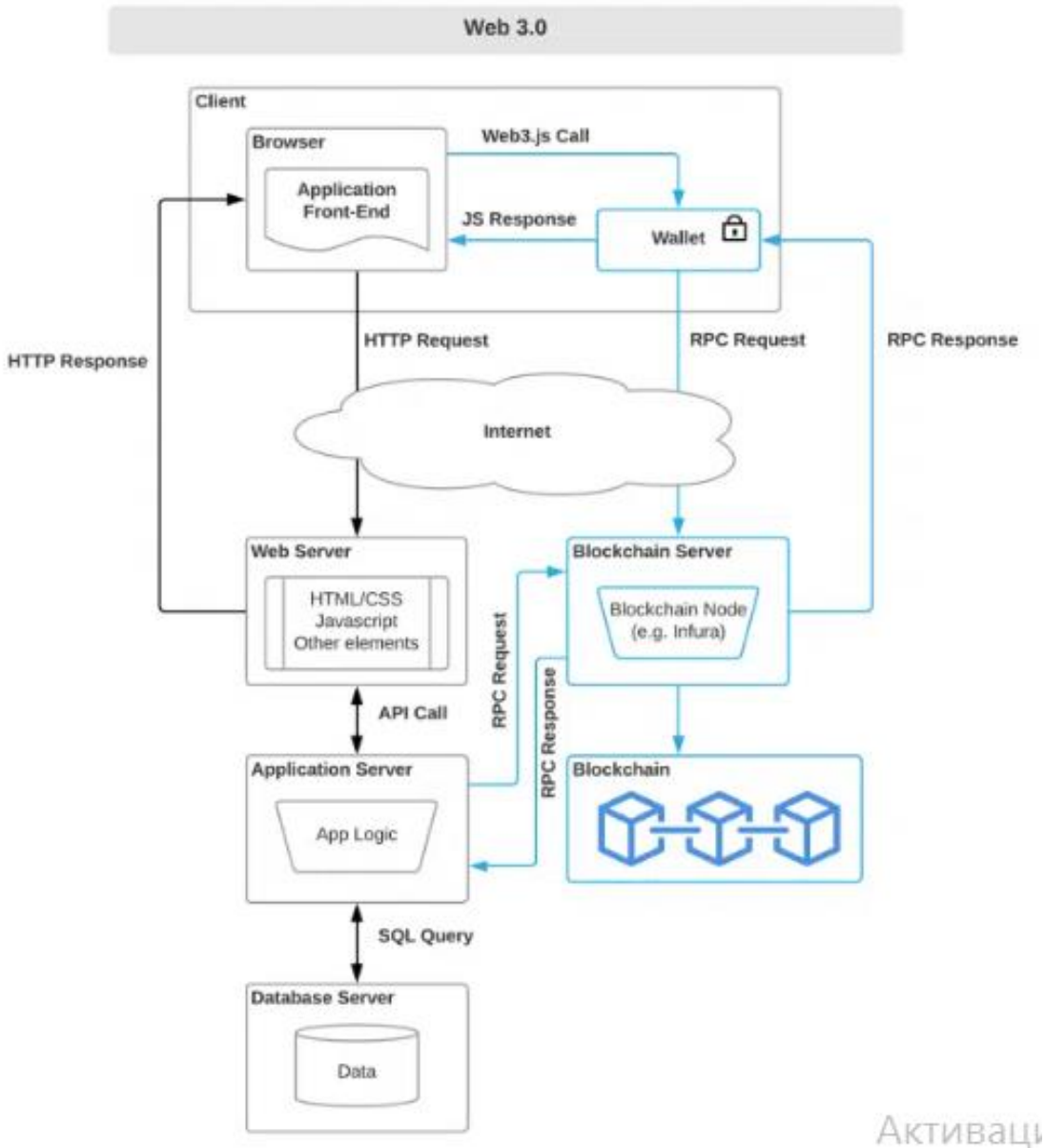


Рисунок 1.2 - Архітектура додатків Web 3.0

Приблизно цього очікують користувачі, користуючись новою версією Інтернету.

Користувачі отримають прибуток за роботу, виконану на чистому ентузіазмі.

Чому власники сайтів платитимуть вам? Справа в тому, що користувачі втомилися від безкоштовної роботи (показу реклами і т.д.). Люди шукають, де і як заробити, а головне, щоб це було легко і весело. Коли ви переглядаєте рекламу, ви маєте отримуєте прибуток від власників бізнесу. Наприклад, переглядаючи рекламу на YouTube, ви отримуєте прибуток як від автора ролика, так і від самої компанії. І якщо автор відео отримує 1 долар за 1000 переглядів, то компанія отримує сотні доларів за ті ж 1000 переглядів.

І користувач взагалі не заробляє ні копійки, це нечесно, чи не так? Саме для того, щоб цього не повторилося, у Web 3.0 створені взаємовигідні відносини між власниками сайтів і звичайними користувачами [3].

Крім того, саме в третій версії Інтернету буде підвищена безпека персональних даних користувачів. Ви зможете надавати дані самі, але в цьому випадку ви отримуватимете прибуток від показу реклами, і якщо ви не хочете надавати дані компаніям, ви не зможете. Перегляд Інтернету стане набагато приємнішим, ніж раніше, оскільки вас більше не турбуватимуть спливаючі сповіщення тощо.

Швидкість вашого браузера збільшиться в кілька разів за рахунок відключення реклами та інших непотрібних плагінів. Якщо ваш улюблений сайт зараз завантажується за кілька секунд, у майбутньому він завантажуватиметься миттєво.

Але, враховуючи сучасні реалії, технологія Web 3.0 ще не скори буде домінуючою на ринку. Тому розробники вебзастосунків намагаються при написанні програмного забезепечення під технологію Web 2.0 вже зараз використовувати елементи штучного інтелекту та різних технологій інтеграції слабоструктурованих даних для формування контенту застосунку. Це і швидше по часу, і зменшує потребу в людському ресурсі.

Mashup, або змішування, - це можливість створити композитний вебдодаток шляхом інтегрування програмних компонент декількох інших вебсервісів.

Методика використання асинхронних запитів Ajax дозволяє в застосунках використовувати JavaScript і XML, щоб відобразити потрібні дані у відповідь на дії користувача, не перезавантажуючи вебдодаток повністю. Завдяки використанню такого підходу можна значно покращити взаємодію користувача із сайтом та прискорити роботу застосунку.

Наприклад RSS - це технологія, заснована на мові XML, яка дозволяє користувачам за допомогою спеціального агрегатора переглядати новини з цікавлячих сайтів як єдиний потік - стрічку новин (читати нові публікації багатьох сайтів в одному місці) [15].

Використання тегів (спеціальних міток), дозволяє більш зручно ідентифікувати й тематично сортувати контент (статті, зображення, мультимедіа файли) [6].

Множина Wiki-сайтів, серед яких найбільш яскравим прикладом є Wikipedia, дає можливість своїм користувачам самим редагувати, додавати чи видаляти інформацію на сайті, формувати нові сторінки, навіть ведуться рейтинги найбільш активних користувачів. У такий спосіб пересічні користувачі більш активно беруть участь у наповненні ресурсів потрібною їм же самим інформацією, але це відкриває шлях до зловживань та шахрайства [14].

Ведення особистих мережних «сторіс», блогів, відеоблогів - от показовий приклад соціалізації в Web 2.0[14] . За допомогою блога кожний користувач може виділитися з сірою маси, персоналізувати свою певну частину ресурсу - додати Також соціалізації сприяє активне створення груп по інтересам (віртуальних спільнот), у яких кожний користувач може залишити комунікувати з іншими зацікавленими користувачами. Використання технологій інтеграції дозволить автоматизувати процес створення вебпорталів для зацікавлених віртуальних спільнот з різноманітним контентом, цікавим саме цій спільноті.

1.2 Огляд сучасних технологій інтеграції інформаційних ресурсів

За ДСТУ 2392-94: Інформаційна система - це комунікаційна система, яка забезпечує збирання, пошук, оброблення та пересилання інформації [8].

Інтеграція даних (Data integration) — це процес об’єднання даних із кількох різних джерел для надання користувачам єдиного централізованого перегляду [18]. Інтеграція — це акт об’єднання невеликих компонентів у систему, щоб вони могли функціонувати як єдине ціле. У контексті IT це об’єднання різних підсистем даних для створення більшої, складнішої та стандартизованої системи для кількох команд. Це допомагає створити єдину аналітику для всіх.

Інтеграція даних допомагає значно консолідувати всі типи даних, враховуючи їх зростання, обсяг і всі різні формати. Об’єднання їх для роботи з єдиним набором даних дає змогу організаціям допомагати внутрішнім відділам безпосередньо візуалізувати бізнес-стратегії та рішення та створювати ефективну та переконливу бізнес-аналітику для коротко- та довгострокового успіху. Поєднання інтеграції та збору даних, обробки, перетворення та зберігання в конвеєрі даних дозволяє вашому бізнесу агрегувати дані незалежно від типу, структури чи обсягу.

Застосування стандартів відіграє суттєву роль у розробці вебдодатків, головним чином тому, що стандарти пропонують можливість взаємодії різних компонентів один з одним і регламентують її [19]. Чим складніша і розгалуженіша система, тим вирішальнішу роль відіграють правила. Стандарти — це загальноприйняті документи, які формалізують передовий досвід.

У системах інтеграції слабоструктурованих даних широко використовується ряд офіційних міжнародних та індустріальних стандартів де-факто. Серед них:

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

Теорія інтеграції даних становить підмножину теорії баз даних і формалізує загальні поняття. Застосування теорії дає рекомендації щодо можливостей і труднощів інтеграції даних. Підключення до конкретних систем керування баз даних, таких як Oracle чи DB2, забезпечуються технологіями рівня реалізації, наприклад такими як JDBC [4].

Ці системи можуть забезпечити інтеграцію даних на логічному, семантичному та фізичному рівнях. На логічному рівні інтеграція даних передбачає можливість доступу до інформації, що міститься в різних джерелах, у термінах єдиної глобальної структури яка описує їх спільне представлення, враховуючи структурні та, можливо, поведінкові властивості (при використанні об’єктних моделей) даних. Підтримка уніфікованого представлення даних з урахуванням їх семантичних характеристик, враховуючи специфіку єдиної онтології предметної області забезпечується структуруванням даних на семантичному рівні [11].

Інтеграція даних на фізичному рівні, якщо розглядати з теоретичної точки зору є найпростішим завданням. Вона зводиться до перетворення даних з різних джерел у необхідний уніфікований формат їх фізичного представлення.

Реалізація методів інтеграції даних вимагає використання адекватних технологій інтеграції даних. Завдання існуючих технологій інтеграції даних полягає в тому, щоб подолати численні прояви неоднорідності, притаманні інформаційним системам, які створюються без уніфікованого ставлення до даних [4, 16]. Клаус Дітріх, дослідник Цюріхського університету, запропонував шестирівневу схему класифікації технологій інтеграції даних [16] (рис. 1.3):

Рисунок 1.3 – Шестирівнева схема інтеграції даних

На сьогодні розповсюдлженими технологіями впровадження інтеграції даних є: реплікація даних; обмін на основі файлів; технологія вебсервісів; інтеграційні сервери; сервісно-орієнтована архітектура (SOA) [18].

SOAP (Simple Object Access Protocol) — це стандартизований протокол для передачі повідомлень між клієнтом і сервером. Зазвичай він використовується в поєднанні з HTTP (S), але також може працювати з іншими протоколами прикладного рівня (такими як SMTP і FTP) [15].

SOAP працює лише з форматом XML. Під час роботи завжди зручно мати стандартизований опис можливих XML-документів і перевіряти правильність їх компіляції. Для цього існує визначення схеми XML.

Існує багато платформ інтеграції даних. Diffbot – це онлайн-сервіс, що включає набір інструментів (Knowledge Graph, Extraction APIs, Crawlbot) для перетворення неструктурованих веб-даних на структуровану і корисну для бізнесу інформацію [13].

Import.io – це онлайн-сервіс, що надає зручний інструментарій для вилучення зі сторінок веб-сайтів, зберігання, об'єднання, інтеграції у власні БД та візуалізації метаданих [19].

ParseHub – це програмний інструмент з нескладним графічним інтерфейсом, що дозволяє захоплювати та отримувати дані з інтернет-сайтів [12].

Octoparse – це хмарне програмне забезпечення, призначене для збору, зберігання та аналізу вебданих, парсингу сайтів [13]. Octoparse застосовує розширений алгоритм машинного навчання, щоб точно знаходити дані в момент, коли ви натискаєте на них.

1.3 Дослідження Semantic Web

Semantic Web (він же Web of Data, Linked Data, Linking Open Data) — напрямок розвитку Всесвітньої павутини, який дозволяє машинам не тільки відображати інформацію в Інтернеті, але й розуміти її значення. Семантичний веб є розширенням поточної мережі, де інформація надається з чітко визначеним

значенням, що дозволить комп’ютерам і людям краще працювати разом. Його ідея полягає в тому, щоб дані в Інтернеті були визначені та зв'язані таким чином, щоб їх можна було використовувати для більш ефективного дослідження, автоматизації, інтеграції та повторного використання між програмами, щоб дані могли спільно використовуватися та оброблятися за допомогою автоматичного засобами, а також людьми» [2].

Рисунок 1.4 - Semantic Web

З архітектурної точки зору семантичний веб можна розглядати як три рівні:

– основа, що складається з унікальної глобальної ідентифікації ресурсів, метаданих для оголошення фактів про ресурси та спільної мови для вираження метаданих і знань, яка реалізована за допомогою онтологій для загального розуміння та спільного словника метаданих і правил для додавання нових метаданих і знань ;

– основні послуги, такі як вивід та запит до метаданих і онтологій, , довірче керування, агенти, пошукові системи, сервери онтології;

– прикладні послуги, наприклад послуга туристичного агентства.

Технології, задіяні в розробці семантичного вебу:

– системи запитань і відповідей;

– семантичний пошук;

– агенти;

– інтеграція баз даних;

– всеосяжне обчислення (ubiquitous / pervasive computation) [19].

У 1998 році Тім Бернерс-Лі запропонував наступний логічний план побудови семантичної мережі [21]:

– синтаксис для представлення знань, що використовує посилання на онтології (RDF);

– мова опису онтології (OWL);

– мова опису ве,сервісів (OWL-S,WSDL);

– sнструменти для читання / розробки документів (Protege, Jena, Haystack);

– мова запитів знань, збережених в RDF (SPARQL);

– логічний вивід знань;

– пошукова система на семантиці (SCARPA).

Базова модель Semantic Web наведено на рисунку 1.4

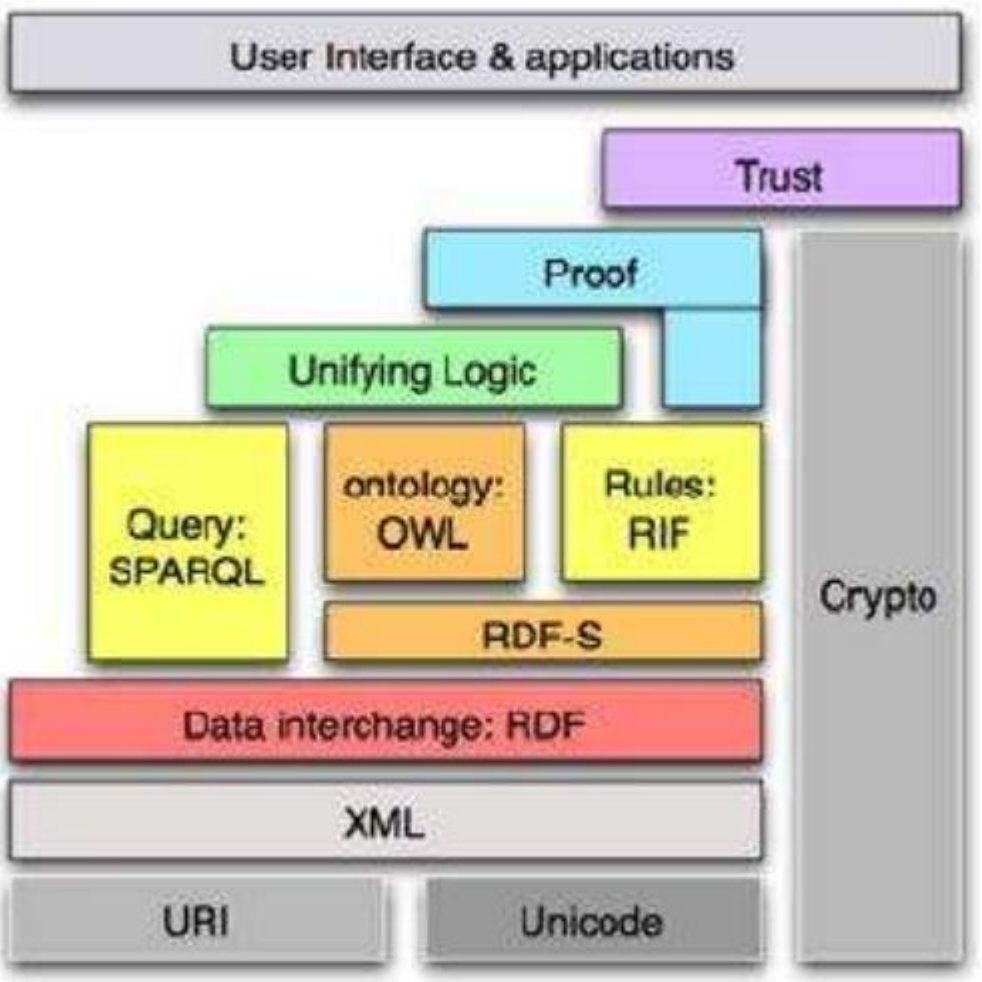


Рисунок 1.5 – Архітектура Semantic Web

Основи семантичної мережі:

– графічна модель для представлення напівструктурованих даних (OEM, Lore);

– формальна логіка (бази знань, фрейми, логіка першого порядку);

– WWW архітектура (URI / IRL, Unicode, XML, HTTP);

– криптографія з відкритим ключем.

Для опису тематичної області ресурсів був запропонований стандарт RDF (Resource Description Framework) [5], підтриманий багатьма основними виробниками програмного забезпечення та контент-провайдерами. Початковою метою RDF було описати ресурси XML з різних точок зору. RDF — це модель опису метаданих, на базі синтаксису XML.

У той час як модель даних XML являє собою граф із позначеними вершинами та непозначеними ребрами (тобто без зв'язків), модель даних RDF представляє собою граф із позначеними вершинами та ребрами. Цей підхід дозволяє визначати зв'язки між сутностями.

Метою Resource Description Framework є стандартизація процесу визначення та використання метаданих, що описують вебресурси. Однак RDF також підходить для представлення даних [4].

Стандарт RDF складається з двох основних блоків: фактичного методу опису ресурсу, і також методу визначення структури, за допомогою якого описується ресурс.

Перша частина RDF [4] визначає просту модель, яка використовується для опису об'єкта, що вважається ресурсом, так само як зв'язки між ресурсами з точки зору їх властивостей та іменованих значень.

Друга схема (RDF Schema - RDFS) [6] служить для опису структури предметної області і схожа на діаграму класів в мові UML.

За допомогою RDF можна описувати як структуру ресурсу, так і пов'язану з ним тематичну область.

RDF описує ресурси як орієнтований граф: кожен ресурс може мати властивості, які, у свою чергу, також можуть бути ресурсами або їх колекціями.

Головна мета RDF – запропонувати базову модель даних «об'єкт – атрибут –

значення» для опису метаданих.

У базовій моделі семантичного вебу, представленій вище, наявність засобів для опису метаданих явно не виділяється. Проте в роботах, наприклад [21], а також у роботах інших вчених вказується на важливість включення поняття метаданих у поняття семантичної мережі.

Метадані - це інформація про дані. Точніше, це інформація, призначена для ідентифікації, опису або локалізації інформаційних ресурсів, незалежно від фізичної природи ресурсу.

Для опису метаданих запропоновано багато схем, серед яких варто виділити наступні:

– Тематичні карти (XMT) [7] є стандартом ISO для представлення та обміну знаннями для пошуку інформації.

– Text Encoding Initiative (TEI) [8] - міжнародний проект, який пропонує стандарти для розмітки електронних текстів, таких як романи, вірші; в основному для підтримки досліджень у гуманітарних науках.

– Схема опису об’єкта метаданих (MODS) [6] — це схема метаданих для опису об’єкта, яка призначена для використання для переносу вибраних даних із існуючих записів метаданих MARC 21 або для створення запису унікального опису ресурсу.

– Стандарт кодування та передачі метаданих (METS) [9], стандарт кодування та обміну метаданими, був запропонований для задоволення потреб у стандартній схемі даних для опису складних об’єктів цифрової бібліотеки.

– Закодований архівний опис (EAD) [11] –був розроблений як спосіб позначення тегами інформації пошукової системи, для щоб їх можна було знайти та переглянути онлайн.

– Онлайновий обмін інформацією (ONIX) [3] — це міжнародний стандарт схеми метаданих, розроблений для книжкових видавців в США та Європі.

Однак наразі визнані основою для семантичного вебу стандарти FOAF, SIOC і DOAP [15].

– FOAF (Friand-Of-A-Friend) [5] — це машинозчитуваний формат сторінки, який описує особисту інформацію про користувачів та їх діяльність (фотографії, «сторіс», блоги тощо) у форматі XML.

– SIOC (Semantically Interconnected Online Communities) [5] - документи, що описують віртуальні спільноти. SIOC забезпечує взаємозв'язок між собою таких засобів комунікації, як форуми, блоги та списки розсилки.

– Опис проекту Опис проекту (DOAP) [9] - документи, що описують проекти з відкритим кодом викладені в мережі.

Онтології, найчастіше визначаються як формальні спільні концепції конкретних предметних областей, які забезпечують загальне уявлення про концепції, за допомогою яких можна здійснювати обмін інформацією між людьми та програмами. Вони дозволяють систематизувати домен, виправляючи сутності і зв’язки в домені. Посилання на відносини, в яких бере участь сутність, дозволяє частково осмислити їх значення (зміст), оскільки дає можливість визначити, де ця сутність відноситься до іншої області.

Онтології використовують математичний апарат формальної логіки (DL), невелика частина якої охоплюється схемою RDF. Дескриптивна логіка є підмножиною обчислюваної логіки першого порядку.

Додаткові функції, згадані вище, і ті, що містяться в RDF, є базисом онтологічних мов, таких як DAML + OIL і OWL [17]. Ці дві мови засновані використанні формату RDF. Завданням цих мов є є надати ресурсам додаткову машиночитану семантику, тобто вони прагнуть забезпечити машинне представлення ресурсів у формі, що більш відповідає їх оригіналу в реальному світі.

Розмітка семантичних вебдокументів за допомогою онтологічних термінів дозволить автоматизувати процес опрацювання їх вмісту. Тому онтології являються ключовоюу технологію для розвитку семантичної мережі.

Онтології відіграють важливу роль в організації обробки, спільного використання та оюміну знань між програмами.

Найдосконалішою мовою представлення онтології на теперішній час є мова веб-онтології (OWL), яка розширює можливості XML, RDF і схеми RDF. Ця мова

заснована на пвдході DAML + OIL. Проблеми, з якими стикалися DAML + OIL, були спровоковані постійними змінами в базовій специфікації RDF, яка основою для DAML + OIL.

Істотні відмінності OWL порівняно з DAML + OIL наступні:

– усунення певних обмежень;

– здатність явно стверджувати, що властивість може бути симетричною;

– усунення деяких невикористаних конструкцій, зокрема обмеження додатковими компонентами.

Є також несутєві відмінності, які включають деякі зміни імен для певних конструкцій.

SPARQL є як мовою запитів, так і протоколом доступу до даних, це один із ключових компонентів програм Web 2.0: як стандарт для підтримки гнучкої моделі даних, він забезпечує загальний механізм запитів для всіх програм [23]. Rule Interchange Format (RIF) - формат обміну правилами [25]. Системи, засновані на правилах, поширилися на інформаційні технології. До них відносяться, наприклад, експертні системи та дедуктивні системи баз даних. Розвиток семантичних вебтехнологій створює нове середовище для використання таких систем. Додавши RuleML до OWL у формі Semantic Web Rule Language (SWRL) [10], стало можливим використовувати правила, подібні до диз'юнкту Горна, щоб явно вказати, як нові факти повинні бути отримані з RDF правила.

Наступним кроком у розвитку семантичної мережі є довіра та докази. Для забезпечення цілісності та узгодженості інформації, представленої в семантичному вебі, потрібно переконатися, що додатки пов'язані з контекстом, а також існують механізми для перевірки доказів і цифрових підписів. У зв'язку з цим Semantic Web якраз і пропонує використовувати цифрові підписи для визначення джерела інформації [3].

Цифрові підписи - це невеликі фрагменти коду, які можна використовувати для однозначної перевірки того, хто написав певний документ.

Програмні агенти повинні відігравати провідну роль у семантичній мережі [24]. При описаній вище схемі інформаційного простору передбачається, що агенти з інтелектуальними здібностями зможуть самостійно виконувати

запропоновані користувачами цілі та завдання. Наприклад, знайти потрібну інформацію, підібрати оптимальні варіанти і т. д. У майбутньому (можливо в технології Web 3.0) це мобільні та розумні агенти, здатні ставити цілі, планувати, взаємодіяти з іншими агентами для досягнення мети, володіючи знаннями про себе та зовнішній світ. Щоб досягти цієї мети, вони повинні мати можливість використовувати деякі зі стандартних наборів послуг, представлених в Інтернеті як вебсервіси.

Вебсервіс - це програмна система, яка надає деякі послуги та забезпечує взаємодію через мережу інтернет. Як правило, це певний вебресурс, що характеризується абстрактним набором деяких функціональних можливостей, які в ньому реалізовані. Функціонально вебсервіс може бути інтелектуальним агентом або звичайною програмою.

Використання технологій інтеграції дозволить автоматизувати процес створення вебсервісів та вебпорталів зі спеціалізованим контентом, наприклад наукових досліджень, по обраній предметній області. Таким чином, завдання розвитку інформаційно-пошукових систем та алгоритмів для компонентної збірки вебдодатків є досить актуальним.

1.4 Постановка задачі

Реалізація динамічної інтеграції неструктурованих даних у вебсистемах потребує використання ефективних методів і засобів обробки семантики інформаційних ресурсів, інтегрованих на всіх етапах роботи з ними в системі Mashup. Проведений аналіз свідчить про актуальність завдання розробки ІТ, яка б дозволила врахувати зміст інтегрованих даних на всіх етапах функціонування даної системи. Мета і завдання дослідження: побудова композитних вебдодатків шляхом вдосконалення інформаційної технології адаптивної інтеграції слабоструктурованої інформації у вебсистемах.

Для подальшого дослідження була сформульована мета кваліфікаційного магістерського дослідження - побудова композитних вебдодатків шляхом вдосконалення інформаційної технології адаптивної інтеграції

слабоструктурованої інформації у вебсистемах.

Сформулюємо основні завдання дослідження:

– проаналізувати Semantic Web концепцію при побудові вебзастосунків, її переваги та недоліки;

– вдосконалити модель стандартизованого профілю системи збору інформації;

– розробити алгоритм побудови онтологічної моделі композитного вебзастосунку

– розробити семантичну модель профілю системи збору інформації;

– вдосконалати метод композитної збірки вебдодатків;

– вдосконалати структурну модель системи композитної збірки, яка використовує технологію Mashup;

– дослідити пошук та вибір сервісів для компонування і реалізації розроблених алгоритмів композитної збірки вебдодатку.

2 ФОРМУВАННЯ МОДЕЛІ СИСТЕМИ ІЗ ДИНАМІЧНОЮ СТРУКТУРОЮ, ВРАХОВУЮЧИ ЗМІСТ ІНФОРМАЦІЙНИХ РЕСУРСІВ

2.1 Семантична модель профілю системи збору інформації

Композитний вебзастосунок - це вебдодаток, який використовує дані з більше ніж одного джерела для генерації нового сервісу, що має єдиний графічний інтерфейс [6]. Наприклад, комбінуючи картографічну інформацію з Google Maps з відгуками користувачів про місця відпочинку, можна створити унікальний вебсервіс з унікальною функціональністю ,яку важко досягти використовуючи єдине джерело. Формування семантичної павутини буде можливим лише за умови вищого рівня ітерабільності. Проте для реалізації цього проекту вже зроблено багато практичних кроків. Новий проект, заснований на пошуковій системі Google, нещодавно надав свої ресурси, щоб вимагати від агентів виконання функцій пошуку та перевірки орфографії [10].

Стандартизований профіль будь-якої системи, включаючи профіль Інформаційної системи, має перелік певних характеристик. У предметній області можуть без проблем працювати кілька альтернативних інформаційних систем, спрямованих на вирішення одних і тих же задач, але такі системи мають різні архітектурні рішення та компоненти. Системи, які автоматизують подібні процеси, можуть мати різні характеристики апаратних і програмних компонентів і в кінцевому підсумку відрізнятися за структурними і функціональними аспектами. Процеси та компоненти інформаційних систем можуть бути добре формалізовані.

Важливість використання семантичних мереж у нашому проекті дуже велика і визначається тим, що такі мережі дозволяють формалізувати тематичні області як набір автоматизованих процесів за допомогою компонентів. На основі семантичної моделі можна побудувати інформаційне середовище для зберігання та модифікації інформації про процеси та компоненти інформаційної системи.

Основою для побудови онтологічної бази знань є семантичні моделі, які визначаються формулою:

$$S^d = \{O, M, R\}, \qquad 2.1$$

де O- набір взаємопов'язаних онтологій, які об'єднують класифікатори, сервіси, апаратні системи та їх специфікації. M - семантичні метадані, R - набір (логічних) правил.

Архітектура будь-якого mashup складається з трьох основних частин, фізично або логічно пов'язаних між собою:

– Постачальники API даних — це постачальники контенту, від яких надходить інформація

– Мешап-сайт — це вебзастосунок, який збирає та розміщує інформацію від постачальників контенту, отриману за допомогою відкритих API.

– Клієнтський браузер, який за допомогою певних налаштувань програмного забезпечення на стороні клієнта може генерувати інформацію про мовні та регіональні налаштування, а також останні пошукові запити.

З цієї причини при запуску браузер самостійно вибирає необхідні конфігурації вебсервісів, позбавляючи користувача від непотрібних дій та інформації, яка не стосується його мови, регіону та його пошукових запитів. Приклад реалізації такого налаштувань браузера чудово відображається в Google Maps API. Коли браузер запускається, користувачеві відображається інформація його мовою. Таким чином, клієнтський браузер - це середовище, в якому програма графічно інтерпретується та організовується взаємодія користувача.

Розвиток Semantic Web створює нове середовище для використання таких систем. Додавши RuleML до OWL у формі Semantic Web Rule Language (SWRL). Це дало можливість використовувати правила, подібні до диз'юнкту Горна, щоб явно вказати, як нові факти повинні бути отримані з RDF правила. В нашому розпорядженні є вже достатній набір засобів для побудови Semantic Web: ствердження, цитування у RDF, класи, властивості, області, документування у схемі RDF, класи, що не перетинаються, типи даних, інверсії, властивості однозначності та унікальності, еквівалентності, списки та інше.

Метою побудови композитного вебзастосунку на основі технології Mashup є об'єднання різних ресурсів (у нашому випадку даних) для його створення. Ці

ресурси можуть надходити з різних джерел, у різних форматах і мають різну семантику. Для підтримки роботи кожен інструмент Mashup використовує власну внутрішню модель даних. Внутрішня модель даних — це єдина глобальна схема, яка представляє уніфіковане представлення даних. Внутрішня модель даних системи Mashup може базуватися на об’єктах або графах.

Графічно-орієнтована модель – використовує теорію графів на основі якої будується модель, заснована на мові XML, RSS, RDF і т.д. При використані об'єктно-орієнтованої моделі внутрішня модель даних формується у вигляді об'єктів (у класичному розумінні об'єктно-орієнтованого програмування). Об’єкт - це екземпляр класу, який визначає характеристики елемента, включаючи його атрибути, поля або властивості і поведінку елемента (методи).

Для створення екземпляру внутрішньої моделі даних із зовнішнього джерела даних, програма Mashup має надати стратегію для збігів за умовчанням між внутрішньою моделлю даних і потрібними джерелами даних. Це досягається використовуючи відображення даних. Відображення даних — це метод, який використовується для визначення відповідності між елементами моделі вихідних даних і внутрішньої моделі даних [2]. Воно може виконуватися вручну, напівавтоматично та автоматично.

Проаналізувавши ці ключові моменти, можна додати нові завдання до списку вимог до майбутньої ІС:

1. Компоненти, які використовуються, повинні бути сумісні з усіма вимогами основних стандартів.

2. Наявність комбінацій стандартів у проектах ІБ, актуальних для користувачів системи, розробників системи та організацій, які їх використовують.

3. Чітке визначення класів і параметрів базових стандартів, щоб надалі гарантувати взаємодію внутрішніх елементів програмного забезпечення в межах єдиної системи. Узагальнений профіль IS складається з трьох наборів:

$$P_S(D_S, V_{D_S}, L_{V_{D_S}}), \tag{2.2}$$

D_S - множина всіх нормативних документів, V_{D_S} - це множина параметрів, які слугують для визначення документів і $L_{V_{D_S}}$ є множиною обмежень для параметрів.

Це базовий фрейм для відкритої обчилсювальної системи, яка базується на основі онтологій, та може адаптуватися для використання у розробці різних систем, для різних предметних областей.

У нашому досліджені використовується профільна модель інформаційної системи (на основі онтологічного фрейму). Базовий шар фрейму буде формуватися за рахунок класифікаторів і специфікацій. На верхньому рівні фрейму буде концепт Document, буде базовим для подальшого наслідування специфікацій і стандартів. Усі документи поділяються на два рівні: рівень довідкових документів і рівень документів прикладних.

Перший рівень включає кілька понять: Classifier, ModelClassifier, Reference, InteroperabilityModelClassifier. Таку модель можна розширити до будь-якого рівня розвитку, додавши до її складу класифікатори кожної групи, а також можна додати самі групи, що не внесе серйозних змін у структуру.

На другому рівні з'являється група понять, яка здійснює формалізацію прикладних документів. Концепція AppDocument використовується для визначення набору всіх документів, а потім InterfaceDocument, який вибирає лише документи, які мають регламентовану поведінку для кожного інтерфейсу. Кожен прикладний документ має модульну структуру, яка визначається за допомогою концепції InterfaceDocumentPart, кожен модуль може містити ряд логічних груп параметрів ParameterSet (кожна група має власні параметри). Якщо частини документів можуть класифікувати на основі принаймні однієї характеристики, вони додаються до ClassifiedDocumentPart. Параметри, які визначаються в нормативній документації, мають певний діапазон значень, заданий доменом. Він описується за допомогою поняття DomainMode. При виконанні опису домену встановлюється базовий системний тип SystemDataType. Він буде визначати параметри та обмеження (Restriction). Для одного параметра може бути визначено кілька екземплярів DomainMode. Існує обмеження InterfaceProfile, яке використовується для формування вибірки параметрів нормативних документів і

модальностей їх використання. Ця концепція лежить в основі інших концепцій, що моделюють роботу функцій введення-виведення та методів взаємодії з системою. Залежно від системи інтерфейси визначаються форматами та структурою даних, що передаються через них.

Насправді більшість систем забезпечують широку динамічну візуалізацію набору компонент. Але є й такі, які використовують агрегації та маніпулювання даними, які можна компонувати з іншими програмами. Вихідна інформація експортується в формат RSS, Atom або XML шляхом додавання метаданих, інформації заголовка та вмісту конкретної інформації, потім інформація перетворюється в послідовність, прийнятну для заданого типу вихідного каналу.

Мікроформати є спробою створити семантичну розмітку різноманітних сутностей на Web-сторінках, що однаково добре сприймається як людиною так і машиною. Інформація у певному мікроформаті не вимагає застосування додаткових технологій або просторів імен. Специфікація мікроформату, це просто домовленості на стандарти найменування класів елементів оформлення сторінки, чкі дозволяють зберігати в кожному з них відповідні дані (рис. 2.1).

Наприклад розберемо формат hCalendar.

```
<div class="vevent"><br>  <a class="url" href="http://www.web2con.com/"><br>    ht
tp://www.web2con.com/ <br>  </a><br>  <span class="summary"> <br>    Web 2.0
Conference <br>  </span>: <br>  <abbr class="dtstart" title="2022-10-
05"> <br>    October 5 <br>  </abbr><br>  -<br>  <abbr class="dtend" title="2007-
10-20"><br>    19<br>  </abbr><br>  ,at the
<br>  <span class="location"><br>    Argent Hotel, San Francisco, CA
<br>  </span><br> </div><br><br>* This source code was highlighted with Source
Code Highlighte.
```

Рисунок 2.1- Приклад мікроформату зберігання інформації для онтологій

Даний мікроформат є підмножиною формату iCalendar (RFC 2445). Він призначений для опису дат майбутніх або минулих подій для надання можливостей їх автоматичної агрегації пошуковими агентами.

На даний момент найпоширенішими мікроформатами є [16]:

– hAtom - формат розсилок новин;

– hCard - опис людей, компаній, місць;

– hCalendar - складання календаря та опис подій;

– hResume - формат опису резюме;

– XFN - спосіб вказівки відносин між людьми;

– hReview - використання оглядів.

Для того, щоб зрозуміти основну суть роботи системи збору інформації і як саме дана система працює з вхідним та вихідним потоком даних скористаємося діаграмою переходів станів (State & Transition Diagram) [14]. State & Transition Diagram – схема станів та переходів. Техніка для візуалізації ТЗ. Вона наочно показує процес, як об'єкт переходить із одного стану до іншого. (рис. 2.2). 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.

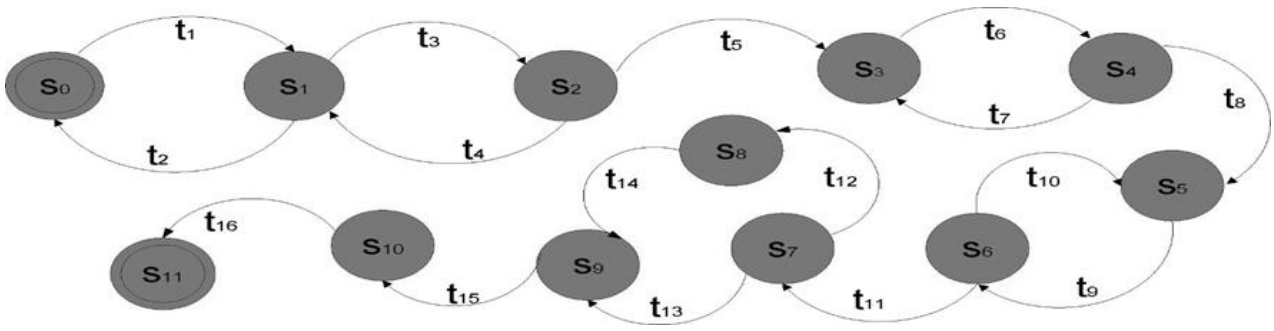


Рисунок 2.2 - Діаграма переходів станів системи збору інформації

На S&TD діаграмі, зображеній на рисунку 2.2 є наступні стани:

– s_0 - початок;

– s_1 - реєстрацію в системі;

– s_2 - авторизація;

– s_3 - очікування на формування завдання;

– s_4 - отримання завдання;

– s_5 - очікування на формування джерел даних;

– s_6 - формування джерел для системи збору інформації;

– s_7 - пошук даних;

– s_8 - редагування пошукових критеріїв;

– s_9 – отримання даних відповідних запиту;

– s_{10} - зберігання інформації у вигляді сервісу;

– s_{11} - стан візуального представлення.

На S&TD діаграмі (рис. 2.2) зображено такі можливы переходи між станами:

– t_1 - відповідає за введення даних користувачем;

– t_2 - введені дані були не коректними;

– t_3 - введені дані були коректними;

– t_4 - не коректна авторизація;

– t_5 - коректна авторизація;

– t_6 - формування завдання ;

– t_7 - завдання сформовано не коректно;

– t_8- завдання було сформовано коректно;

– t_9 - вибір джерел для формування композитного додатку;

– t_{10} - помилка при виборі джерел;

– t_{11} - джерела сформовано коректно;

– t_{12} - редагування критеріїв для формування результату;

– t_{13} - пошук інформації відбувся вдало;

– t_{14} - пошук інформації з редагуванням відбувся вдало

– t_{15} - отримання даних з обраних джерел;

– t_{16} - зберіганні даних у вигляді композитного вебзастосунку.

Для візуалізації та аналізу роботи системи збору інформації було інтерпретовано наведену вище S&TD діаграму у модель діяльності системи використовуючи UML-діаграму діяльності [6].

Аналізуючи діяльність систем можна виділити наступні стани: (рис. 2.3):

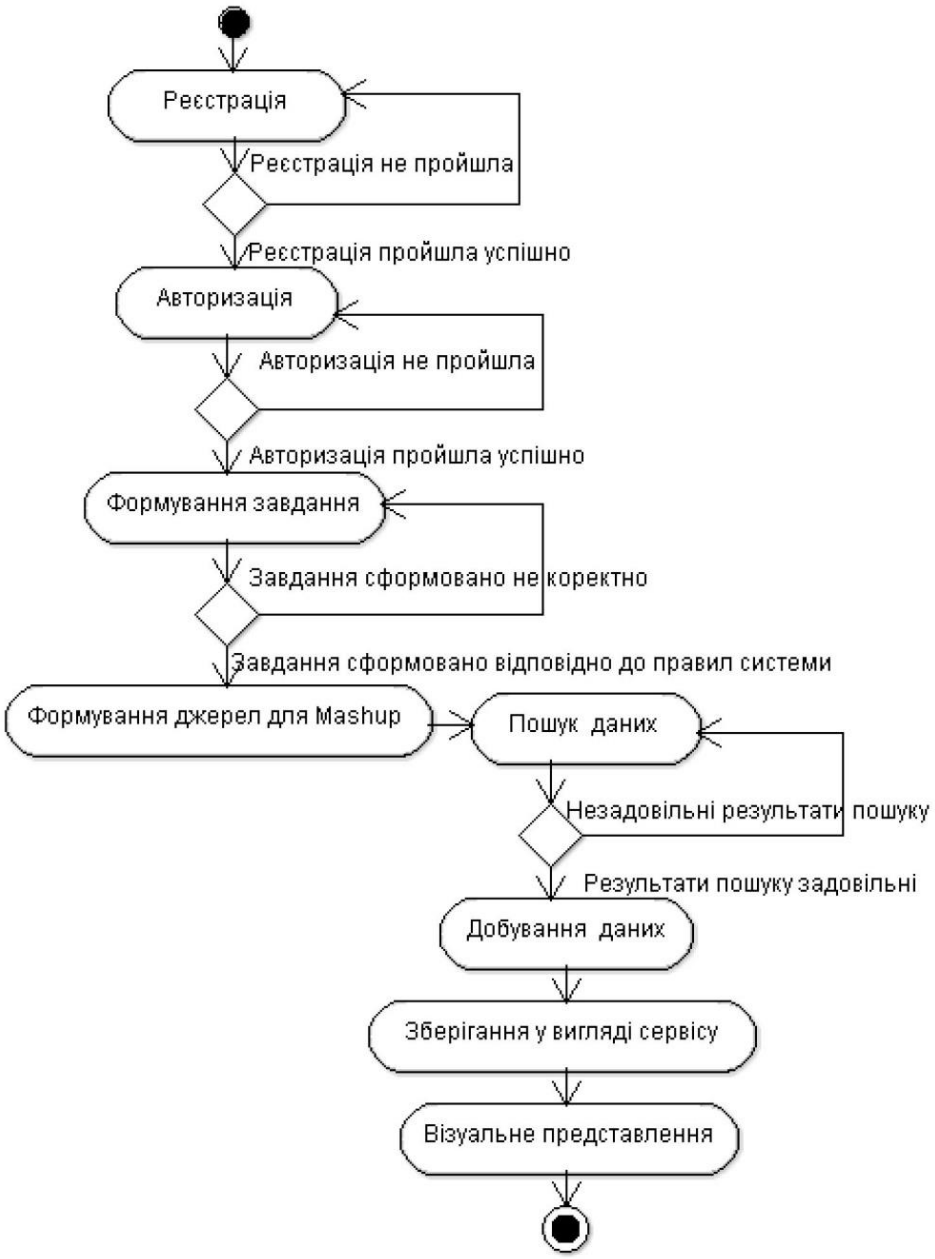


Рисунок 2.3 - UML-діаграма діяльності системи збору інформації

Отже, було визначено 8 основних станів, які притаманні будь-якій системі на основі технології Mashup. Найважливішими станами при використанні системи збору інформації є пошук даних (п'ятий стан), їх добування (шостий стан) та зберігання у вигляді певного сервісу, доступного для композитного вебдодатку (сьомий стан). Розглянемо детальніше ці стани. Стани пошуку та добування даних належать до методів роботи системи з вхідними даними, а стан зберігання даних до методів відображення вихідних даних. Тодіо процес відображення наборів вхідних даних до набору вихіднихо даних складається з: етапу визначення вхідних даних та

етапу генерації виводу. Тоді стандартизований профіль системи збору інформації можна подати:

$$H = \langle DI, Q, DO, C, T, \lambda, \beta \rangle, \quad (2.3)$$

де: DI – набір вхідних даних; Q – множина запитів користувачів; DO – глобальний набір вхідних даних; C – множина умов інтеграції; T – час транзакцій на оновлення даних; λ - оператор визначення вхідних даних; β - оператор формування вихідних даних.

Схема процесу інтеграції даних наведено на рисунку 2.4

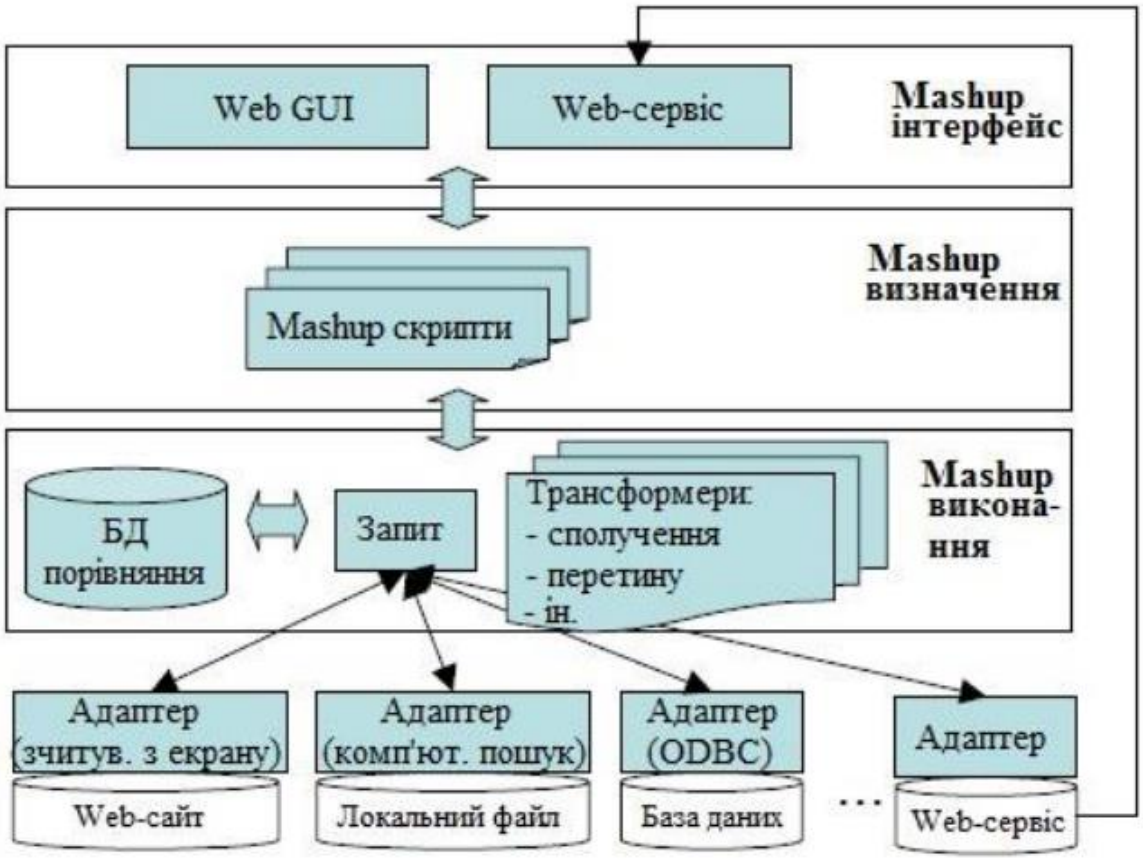


Рисунок 2.4 – Схема інтеграції даних

В основному джерелами комозитних додатків є вебсистеми з даними, представленими у вільно структурованому вигляді, оскільки на сьогоднішній день домінуючу позицію як засобу подання інформації в Інтернеті займає мова HTML, теги якої описують візуальне представлення документа, посилання тощо, але не містять інформації про семантичну структуру документа. Пошукові стратегії

включають функцію вибору запиту і функцію оцінки запиту. Всі запити ранжуються за отриманими оцінками (у порядку зростання) і функція вибору надалі обробляє та фільтрує запити відповідно до їх коефіцієнтів ранжування.

2.2 Процес композитної збірки вебдодатків

Першочерговим завданням композитної збірки вебдодатків для опису структури і змісту вхідної інформації є визначення форми подання вхідних даних: неструктуровані, структуровані, слабоструктуровані. Формати інформаційних ресурсів, які розглядалися: вебформат, табличний формат, XML та мультимедійний контент.

Представлено загальну схему автоматизованої системи композитної збірки. У цій схемі комбінація систем представлена у вигляді блоку первинного аналізу та форматування даних, блоку, що формує остаточний склад вебдодатків, і блоку керування збереженням даних про процеси та компоненти.

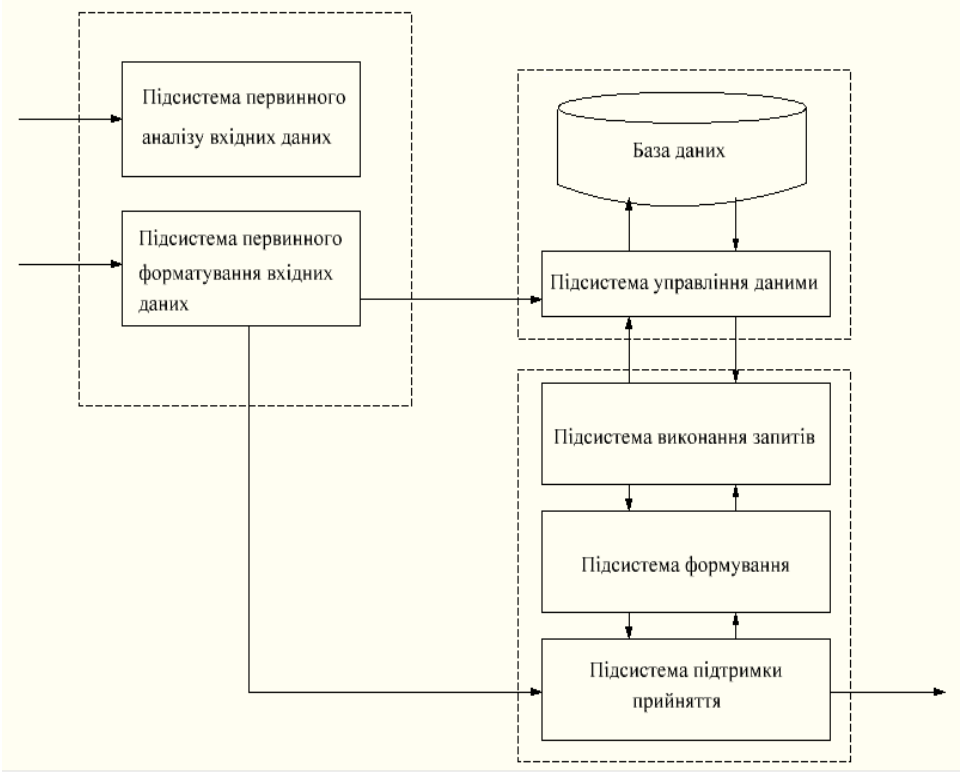


Рисунок 2.4 – Схема системи композитної збірки вебдодатків

Побудована за такою схемою семантична модель повинна обчислювати та оцінювати окремі властивості компонентів і процесів, що входять до неї. Також потрібно дотримуватися ієрархії процесів і вести списки готових рішень, які успішно застосовані.

Вибір процесів відбуватиметься ітераційно і на кожному кроці буде формуватися набір процесів, які претендуватимуть на включення в структуру веб-додатку. Компоненти можна умовно розділити на зовнішні та внутрішні.

Зовнішні компоненти забезпечують гнучкість системи, оскільки вона може використовувати компоненти в режимі віддаленого доступу за допомогою онлайн-ресурсів. При успішному підключенні до Інтернету ви можете завантажити необхідні компоненти для автоматизованої системи складання. Внутрішні компоненти вимагають більшої уваги, оскільки їх неможливо завантажити онлайн. Ці компоненти додаються вручну під час формування збірки. Вони реалізовані у вигляді різноманітних класів і бібліотек.

Процес композитного складання відбувається шляхом реалізації трьох основних фаз, які пов’язані з рівнем представлення: процесна (формування графа, який визначає функціональні характеристики програми), компонентна (фаза формування екземплярів із їхніми залежностями, які реалізують процеси на логічному рівні), структурний (складає план, на основі якого потрібні компоненти будуть імпортовані та вставлені в структуру проєкту). Компоненти діляться на два типи: внутрішні і зовнішні. Внутрішні компоненти розташовані в структурі самого проекту. Зовнішні компоненти використовуються в режимі віддаленого доступу.

Для зберігання інформації про набори процесів використовується префіксне дерево, яке дозволяє створювати швидкі алгоритми пошуку правил. Фрагменти семантичної моделі наведені на рисунку 2.5.

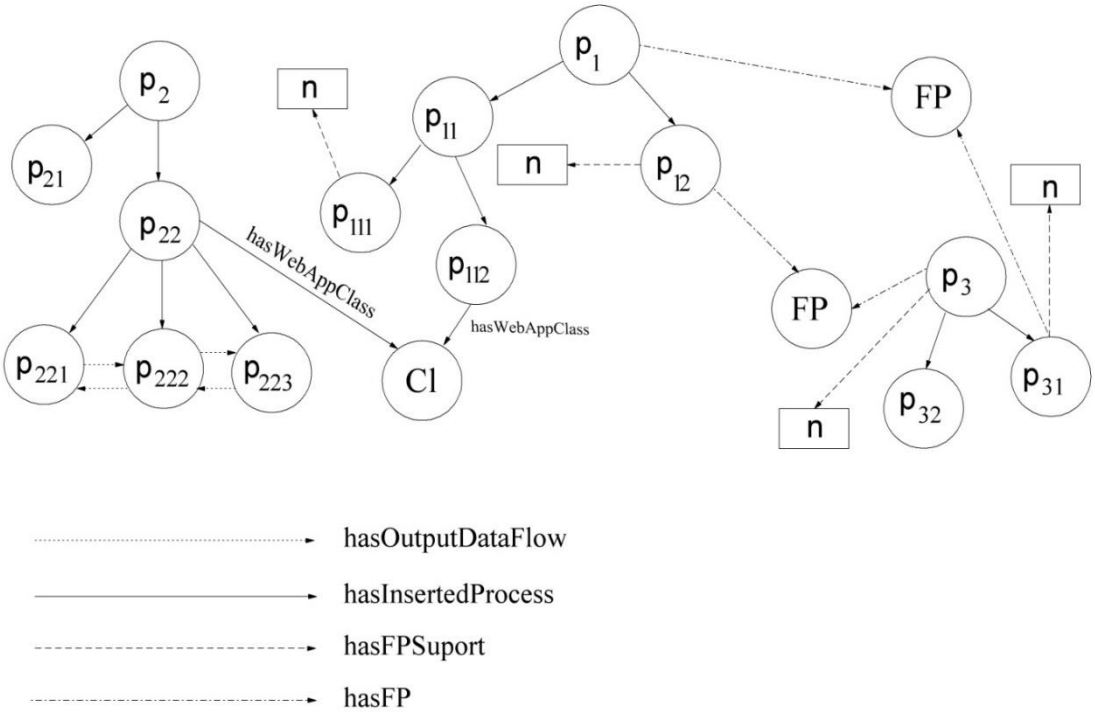


Рисунок 2.5 - Фрагмент семантичної мережі

Опис процесів (*p*) відбувається за допомогою вектора базових властивостей, який складається з ім'я (*name*) та ідентифікатор (*PrId*). Процесом може виступати будь-яка взаємодія, наприклад p_1 - взаємодія клієнт-сервер, p_{111}, p_{112} - процеси обміну даними. *hasFP* - описує склад процесів у форматі префіксного дерева, які найчастіше зустрічаються.

Продукційні правила будуть мати вигляд:

$$(p_1\ hasFP\ fp_1) \rightarrow fp_1\ fphasProcess\ p_1 \tag{2.4}$$

Продукційні правила потрібні для автоматизації процесу створення додаткових характеристик, які використовуються в пошукових алгоритмах.

Онтологія OWL — це послідовність аксіом та фактів із додаванням посилань на інші онтології, що вважаються включеними в онтологію. Онтології у OWL є вебдокументами, і на них можна посилатися. Онтології також можуть мати ще не

визначений компонент, який можна застосовувати для запису авторства та іншої нелогічної інформації, пов’язаної з онтологією. Іншими словами це словник, який розширює набір термінів, визначених у RDFS (рис. 2.6).

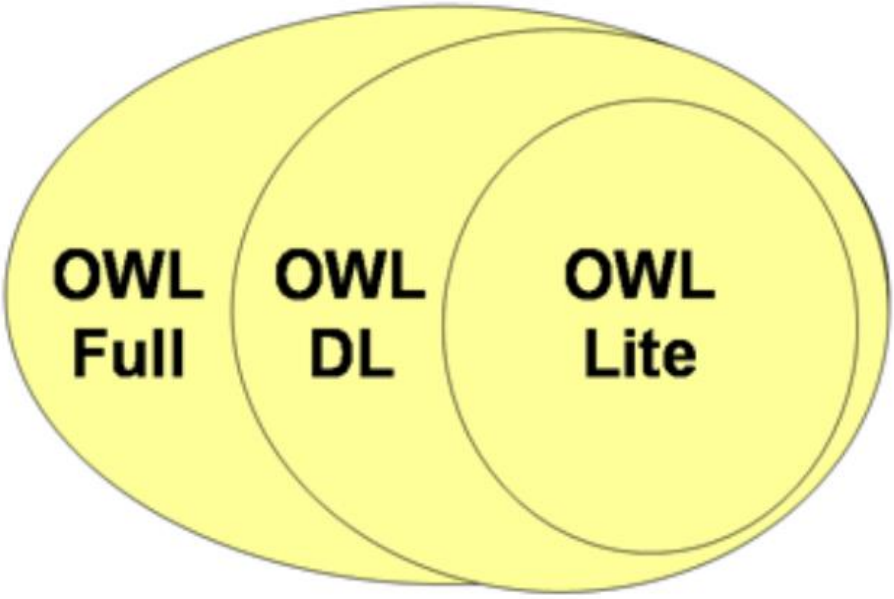


Рисунок 2.6– Відношення вкладеності між підмножинами мови OWL

Онтології містять інформацію про класи, властивості та деякі спеціальні випадки, кожен з яких може мати ідентифікатор, що є посиланням URI.

OWL має три модифікації:

– OWL Lite (простий);

– OWL DL (компроміс між обчислювальними можливостями і виразністю мови)

– OWL Full (доступні засоби максимальної виразності).

Основними особливостями мови веб-онтології OWL є:

– OWL містить інструкції для представлення дерева класів;

– OWL використовує синтаксис XML;

– OWL містить інструкції щодо визначення належності окремих осіб до класів;

– OWL може вказати характеристики властивості: симетричність, транзитивність, функціональність;

– OWL має систему опису властивостей: область визначення, область значень;

– OWL має інструкції для визначення еквівалентності (склеювання) класів (рис 2.7).

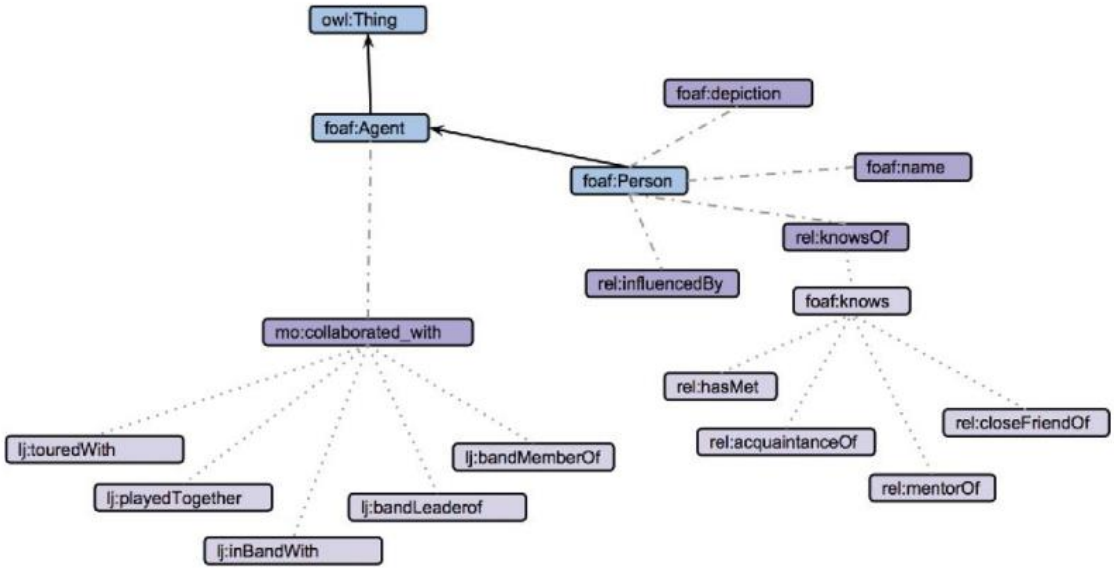


Рисунок 2.7 - Представлення онтології предметної області

Принцип «виводу» дуже простий: це здатність виводити нові дані з існуючих. У математичному сенсі виконання запиту є формою висновку (наприклад, здатність вивести результат пошуку з маси даних). Висновок є одним із керівних принципів семантичної мережі, оскільки він дуже полегшує створення програмних програм [8].

Для того, щоб семантична павутина стала достатньо виразною та могла допомогти людям у різних ситуаціях, стає необхідним створити потужну логічну мову, яка підтримує логічний висновок. Звертається увага на той факт, що RDF не має можливості кількісного визначення та ця область не є чітко визначеною.

2.3 Висновок

У другому розділі аналізується процес побудови стандартизованого профілю інформаційної системи.

Було формалізовано семантичну модель стандартизованого профілю ІС, а також формалізовану модель онтологічної бази знань на її основі. Запропонована модель включає 3 рівні: рівень специфікації, рівень домену, рівень інформаційної системи. Модель визначає загальну структуру, в межах якої можуть бути побудовані прикладні семантичні моделі, які застосовуються безпосередньо на практиці.

В розділі було запропоновано підхід до підвищення якость результату процесу визначення вхідних даних системи збору інформації шляхом застосування процедури опису структури та змісту вхідних інформаційних ресурсів, що дає змогу покращити якість процесу інтеграції даних з різних джерел для подальшого їх використання в системі збірки вебзастосунку. Для цього була розроблена модель процесу композитної збірки вебдодатків у вигляді семантичної мережі з використанням продукційних правил.

3 МЕТОД ТА АЛГОРИТМИ ФОРМУВАННЯ КОМПОЗИТНОГО ВЕБЗАСТОСУКУ

3.1 Формування структурованого набору данних

Загальна структура та уніфікований зміст інформації можуть бути забезпечені при формуванні комбінованого динамічного набору даних лише за умови використання семантичної інтеграції даних

Семантичний підхід вимагає використання семантично-орієнтованих технологій, однією з яких є онтології (рис. 3.1)

Рисунок 3.1 – Класифікація даних в області онтологій

Інтегруючи дані із застосуванням технології онтологій можна одержати дані, які зберігаються у іншому джерелі, об’єднати їх з логічними та глобальними онтологіями використовуючи методи відображення [14].

Доцільно застосувати різні методи для підвищення якості результатів (рис.3.2.)

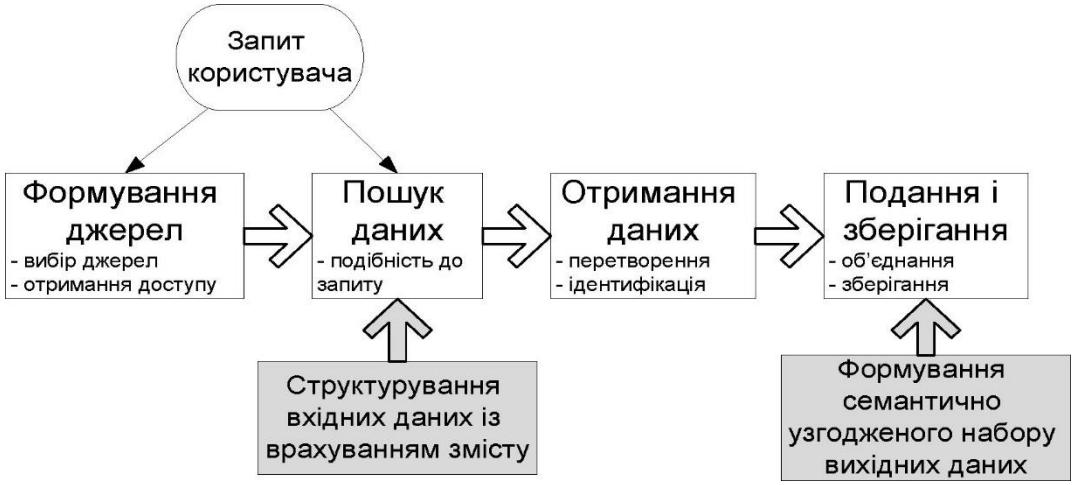


Рисунок 3.2 - Методи підвищення якості результатів

При розгляді систем динамічної інтеграції на основі Mashup технології, має місце поняття «піраміди значень», тобто різних рівнів, на яких відбувається інтеграція [23] (Рис. 3.3).

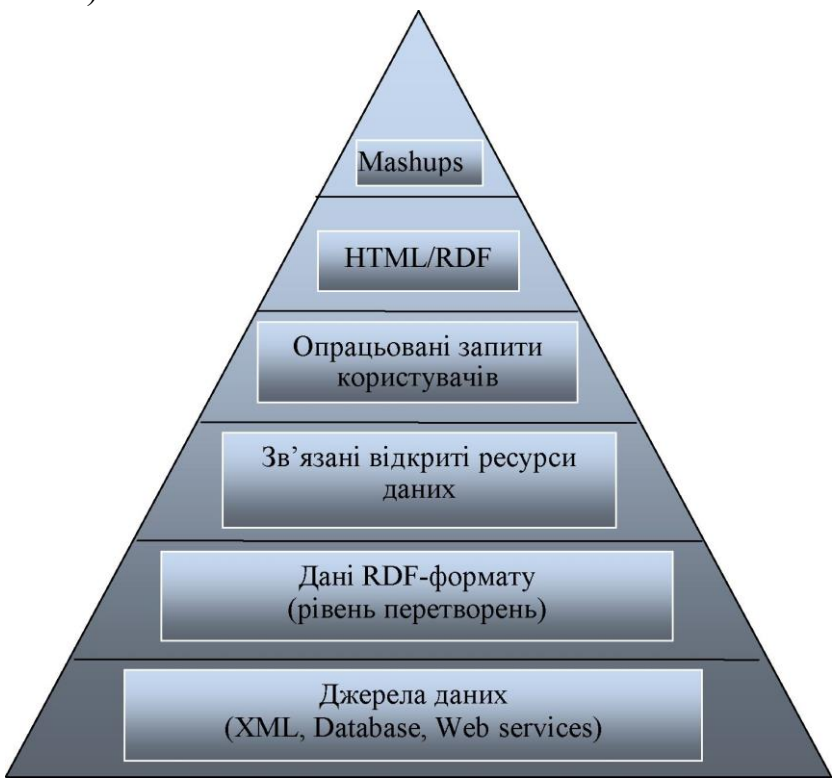


Рисунок 3.3 - «Піраміда значень»

Запропонуємо принципи побудови контекстно-керованої системи для формування об'єднаного наборуінформації з слабостротурованих даних, який має узгоджену структуру і єдиний зміст.

Таблиця 3.1 - Принципи побудови контекстно-керованої системи

№ п/п	Перелік вимог до контексту	Принципи побудови контекстно-керованої системи
1	Контекст має описуватися стандартизованими способами	Використання технології побудови онтологій для опису контексту
2	Модель знань має підтримувати операції з контекстом	Використання онтологічної моделі представлення знань
3	Контекст має надавати релевантну інформацію	Використання глобальної мета-моделі контексту, яка використовує інформацію про предметну область
4	Підтримка механізмів повторного використання та абстракції контексту	Використання дворівневої моделі
5	Підтримка механізмів використання підконтекстів	Ієрархічне представлення підзадач
6	Контекстна інформація містить метадані і знання	Онтологічна модель контексту

На основі аналізу моделей контексту були розроблені принципи, покладені в основу моделі побудови і функціонування динамічної контекстно-керованої системи формування узагальненого динамічного набору даних, що має узгоджену структуру і уніфікований зміст.

Збір інформації за предметною областю повинен складатися з трьох етапів:

1. Структурна онтологія системи інтеграції. Отримання інформації про структуру кожної з компонент інформаційних систем предсавлена в онтологічному форматі.

2. Загальна структурна онтологічна інформаційна модель.

3. Метамодель інтегрованих систем, в яку входить онтології домену та загальної структурної онтології глобальної моделі. Вона описує семантичні відносини між інтегруючими компонетами системи.

Щоб інтегрувати певну компоненту в композитний вебдодаток, нам повинні спочатку знати структуру програми або системи, яку потрібно інтегрувати. На більшості ресурсів майже вся інформація зберігається в базах даних. У реляційних базах даних інформація про структуру таблиць та зв'язки між ними зберігається в схемах даних, і ці схеми повинні бути отримані під час виконання алгоритму.

Однак аналізу самої схеми достатньо лише для забезпечення структурної сумісності. Для досягнення семантичної сумісності під час вилучення схеми даних необхідно також враховувати семантичне призначення цих елементів, тому потрібно використовувати онтологію домену. Така онтологія сформує зв'язки між поняттями в предметній області. Таким чином, будь-яка онтологічна модель, отримана з бази даних системи, буде підмножиною онтології домену.

Перед використанням алгоритму для побудови онтологічної моделі всіх інтегрованих систем необхідно виконати процедуру реєстрації в напівструктурованому форматі html інформації в попередньо створену базу даних.

Процес оримання даних із сайту HTML полягає в наступному. Ми шукаємо назви класів, які відповідають за опис потрібної інформації та отримують їх зміст. Таким чином, заповнюється інформаційні поля таблиць БД. Для автоматизації цього процесу інтегратор даних використовує мову спеціалізовану мову запитів.

Наступним етапом є використання алгоритму для побудови онтологічної моделі композитного вебзастосунку. Вхідними даними для цього алгоритму є:

– структурні діаграми схем данихінтегрованих систем;

– онтологія домену.

Онтологія кожної предметної області розробляється заздалегідь за участі фахівця з онтологічних знань. Процес побудови такої моделі займає певний час, але це відбувається лише на початковому етапі інтеграції.. Ця модель буде змодельована RDF та мовою OWL.

Алгоритм побудови онтологічної моделі композитного вібзастосунку, містить 6 основних кроків (рис. 3.4).

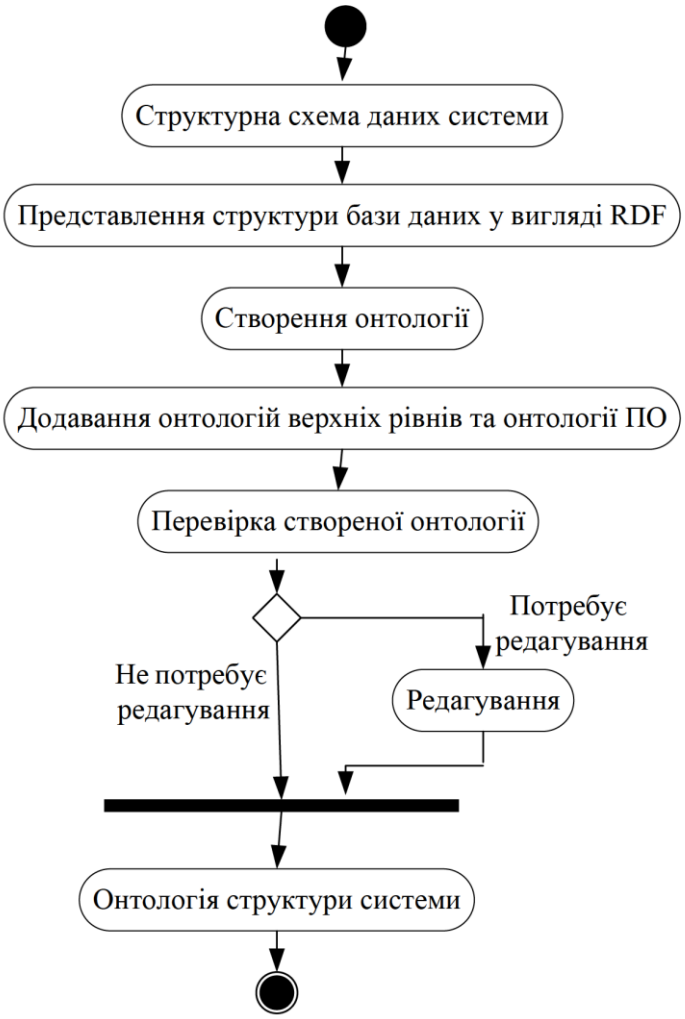


Рисунок 3.4 - Алгоритм побудови онтологічної моделі композитного вебзастосунку

На першому етапі відбувається послідовне відображення структурних елементів бази даних у формат RDF. Ця інформація отримується за допомогою мови запитів сімейства SQL та механізму зовнішнього ключа. Імена таблиць автоматично перетворюються в імена нових класів, а імена полів таблиці у методи, пов'язані з їх класом. Унікальний ідентифікатор прототипу буде містити таблицю і первинний ключ.

Результатом виконання першого етапу алгоритму є документ RDF, що містить інформацію, яка описує структурні дані зі схеми системної бази.

Другий етап алгоритму застосовує процедуру збільшенн кількість смислових зв’язків між різними онтологіями у межах загальної онтології.

Відбувається визначення схожості запиту користувача з семантичними метаописами інформаційних ресурсів.

На наступному етапі онтологія розширюється онтологіями домену та додатковими онтологіями верхнього рівня. Це робиться за допомогою команди функції *import.* На сьогодні існує велика кількість онтологій у різних тематичних областях, тому імпортуючи їх в отриману структурну онтологічну модель, ми можемо отримати данні не тільки про системи, які знаходяться у вашому домені..

Четвертий крок включає перевірку створеної онтології на наявність відсутніх зв’язків між об'єктами онтології. Автоматичне завершення створеної онтології на третьому етапі не завжди може знайти всі залежності між елементами онтології, тому на наступному етапі виконуться перевірка, в разі її успішності ми переходимо відразу до шостого етапу.

На п'ятому кроці можливе додаткове ручне втручання фахівця для встановлення зв’язків між об'єктами онтології.

Нарешті, завершальним кроком є отримання загальної онтології структури у форматі RDF. Цей результат може бути записаний у файл чи локальне сховище. Він містить зв'язки між компонетами в системах, які інтегруються в домен, і багато термінів з інших полів.

3.2 Вдосконалений метод композитної збірки вебдодатків

Взявши за основу семантичну модель, описаної у другому розділі, ми розробили метод копозитної збірки вебзастосунку.

В запропонованому методі композитної збірки вебдодатків на вхід надсилається опис складу компонентів вебзастосунку, на основі якого будується формалізований список цільових елементів із залежностями. Згодом цей список використовується для пошуку компонентів з одного або кількох репозиторіїв з подальшим формуванням структури цільового вебзастосунку [12].

Процес композитного складання відбувається шляхом реалізації трьох основних фаз, які пов’язані з рівнем представлення: процесна (формування графа, який визначає функціональні характеристики програми), компонентна (фаза

формування екземплярів із їхніми залежностями, які реалізують процеси на логічному рівні), структурний (складає план, на основі якого потрібні компоненти будуть імпортовані та вставлені в структуру проєкту) (рис. 3.5).

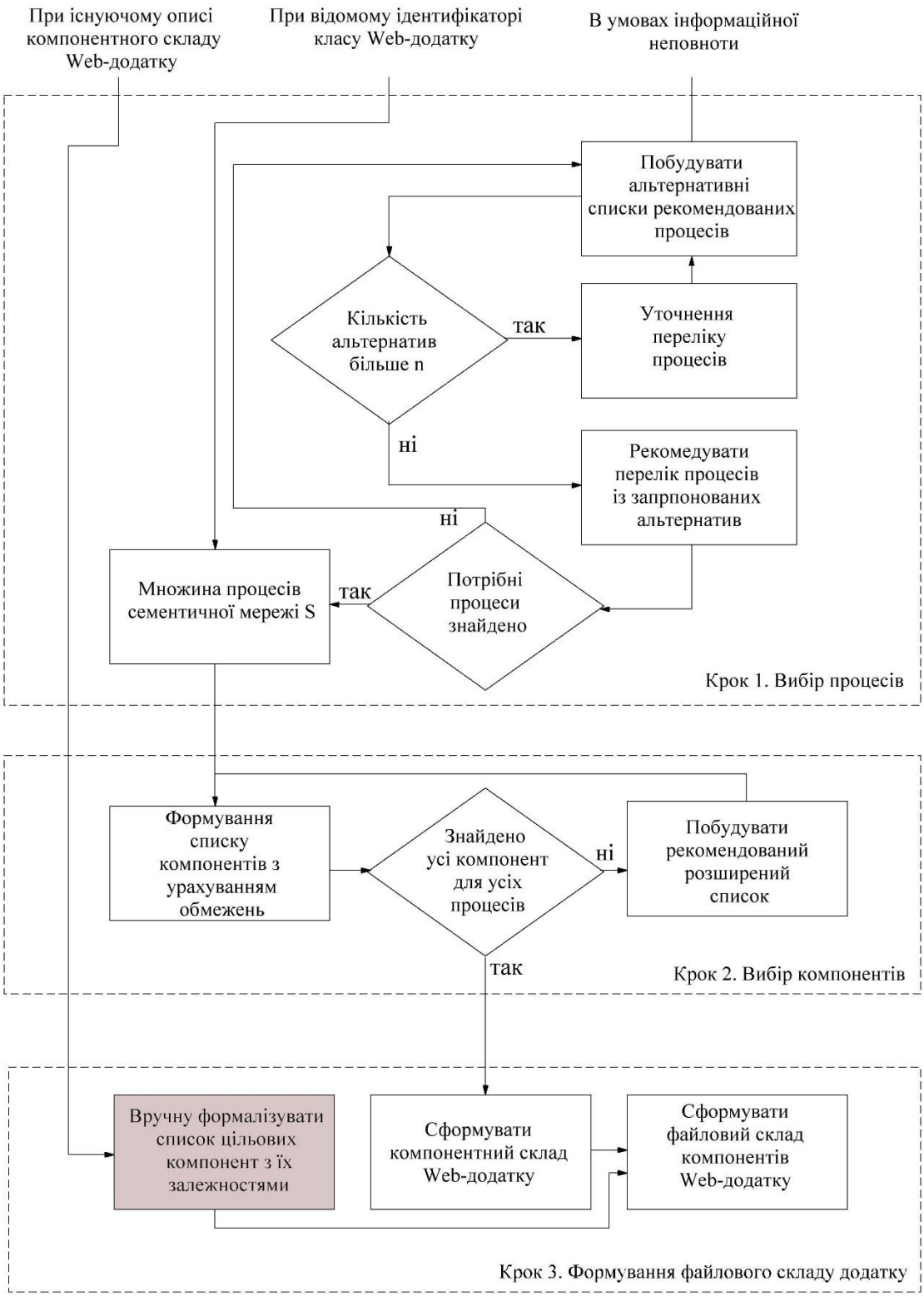


Рисунок 3.5 - Вдосконалений метод композитної збірки вебдодатків

Вдосконалений метод дозволяє відновлювати набори компонентів відповідно до їх часткового опису.

Він складається з трьох етапів: вибір процесів, компонент та формування складу композитного вебзастосунку При виборі процесів виконується перевірка типу запиту. На кожному кроці ітерації відбувається "нарощування наборів" і вибору асоціативних правил шляхом модифікації алгоритму генерація часткових шаблонів FPG з побудовою префіксного FP-дерева, адаптованого до роботи з семантичною мережею певної предметної області. При цьому постійно уточнюється множина рекомендованих наборів процесів і формуєтьсясемантична мережа з ієрархією вкладеності і потоками даних.

На наступному етапі відбувається вибір компонентів семантична мережа розбивається на непересічні підмножини, для компонентів яких визначені потоки передачі даних. На кожній ітерації формується список компонент, який може регулюватися певними обмеженями: операційна система, вартість, платформа розробки. Перелік обмежень за потреби можна розширити. При формувані списку сумісних компонентів, здійснюється перехід до третього кроку, де формується файлова структура композитного вебзастосунку.

На етапі проетування архітектури системи онтологічна модель буде спрощуватись. Замість опису кожного параметра, режиму тощо, пропонується описувати відповідні розділи документів і прив’язувати до них набір характеристик відповідно до системи класифікаторів. Репозиторій містить: електронні документи стандартів, специфікацій, файли з описом ресурсів і процесів, файли з атрибутами електронних документів, файли з описом класифікаторів.

У репозиторії кожен електронний документ однозначно ідентифікується URI та може бути розміщений на будь-якому сервері в мережі. Класифікатори використовуються для групування ресурсів та їх частин за різними ознаками.

Для організації механізмів пошуку та обміну інформацією пропонується застосувати мультиагентну схему керування запитами до бази знань стандартизованого профілю. Агентів можна поділити на 4 групи:

1. Агенти для формування інтерфейсу користувача (ClientUI).

2. Агенти планувальники (TaskPlanTranslator).

3. Агенти, відповідальні за виконання запиту (TaskAgent).

4. Агенти, які витягують дані з бази знань (KnowledgeAgent).

Агенти, відповідальні за прийом і попередню підготовку клієнтського запиту, постійно прослуховують спеціальний порт і переходять на етап «Попередня підготовка запиту», якщо є хоча б один запит в системній черзі. Згодом запит формалізується та форматується відповідно до вимог агентів у другій групі TaskPlanTranslator і надсилається їм на опрацювання. Далі відбувається оформлення доручення для відповідального виконавця (TaskAgent). Агенти KnowledgeAgent витягують дані з бази знань і формується глобально унікальний дескриптор діяльності.

На рисунку 3.6 показана блок-схема алгоритму перетворення інформації із загальної GENERIC-моделі в онтології, представлені в форматі OWL-DL.

Онтологія OWL — це послідовність аксіом та фактів із додаванням посилань на інші онтології, що вважаються включеними в онтологію. Онтології у OWL є вебдокументами, і на них можна посилатися. Онтології також можуть мати ще не визначений компонент, який можна застосовувати для запису авторства та іншої нелогічної інформації, пов’язаної з онтологією.Вона реалізована у вигляді словника.

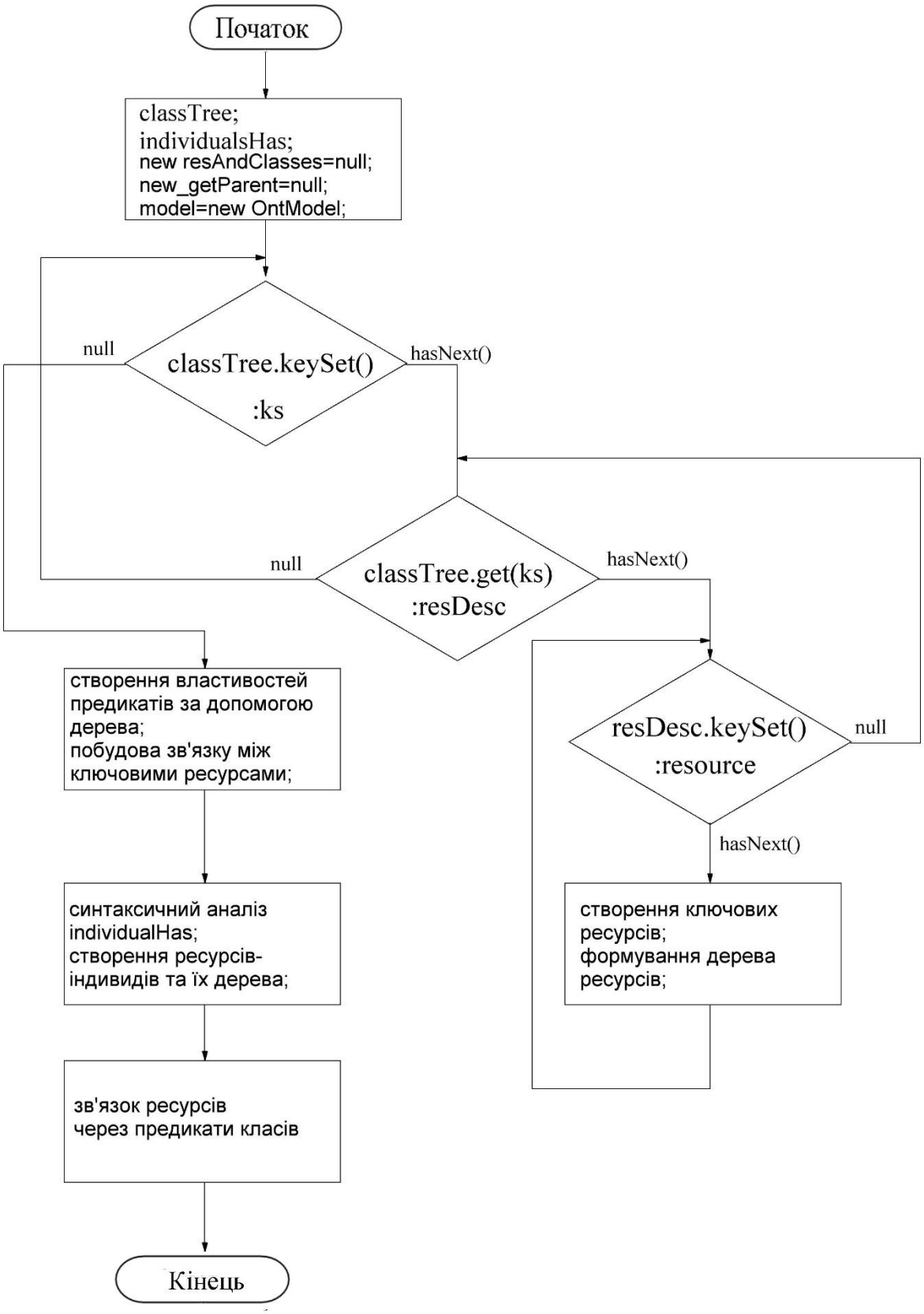


Рисунок 3.6 - Блок-схема алгоритму перетворення інформації із загальної GENERIC-моделі в онтології

При побудові системи композитної збірки вебдодатків будемо базуватися на еталонній моделі OSE/RM [12].

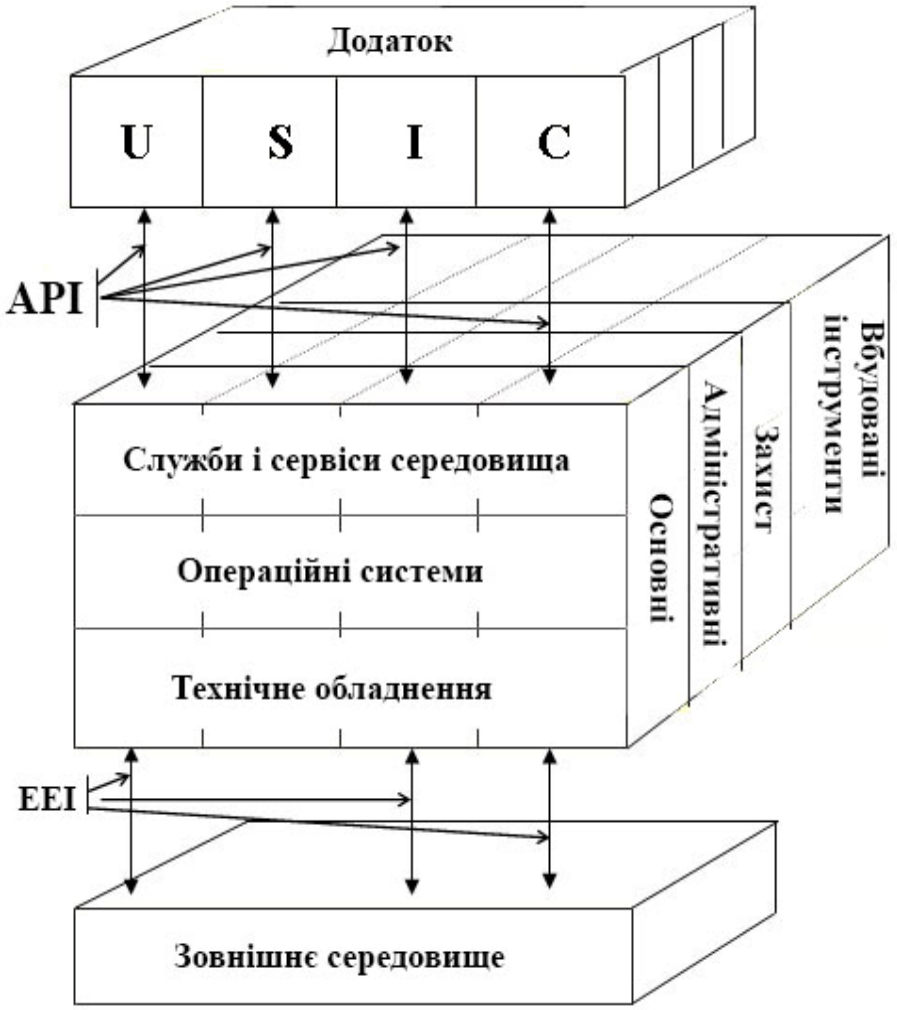


Рисунок 3.7 - Еталонна модель OSE/RM

Open System Environment Reference Model – OSE/RM модель використовує Application Programming Interface (API) - опис способів взаємодії однієї комп'ютерної програми з іншими та інтерфейс платформа - зовнішнє середовище (EEI).

API є інтерфейсом між прикладним програмним забезпеченням і платформою додатків. Його основною функцією є підтримка переносимості прикладного програмного забезпечення. API класифікується відповідно до типів послуг, доступних через цей API. Існує чотири типи служб API в OSE/RM:

- служби обміну інформацією;

- послуги зв'язку;
- служби інтерфейсу людина/комп’ютер;
- внутрішні системні служби.

Дана модель може бути модернізована в залежності від класу системи. Наприклад, для телекомунікаційних систем використовується 7-рівнева модель взаємозв’язку відкритих систем ISO/IEC 7498. Модель OSE/RM виросла як розширення моделі взаємозв’язку відкритих систем OSI із деталізацією верхнього прикладного рівня.

Модель OSE/RM була запропонована робочою групою POSIX Інституту інженерів з електротехніки та електроніки. Він передбачає поділ середовища на три складові:

- платформа додатків;
- програмне забезпечення;
- зовнішнє середовище.

Взаємодія між прикладним програмним забезпеченням і платформою додатків здійснюється за допомогою чотирьох програмних інтерфейсів, між платформою додатків і зовнішнім середовищем - за допомогою трьох типів інтерфейсів.

Еталонна модель має три виміри для класфікації. Послуги, які визначаються OSE/RM можна згрупувати за такими рівнями інтероперабельності: організаційний, технічний, семантичний.

По вертикалі можна виділити такі компоненти:

- платформа (прикладна платформа складається з апаратної платформи та програмного забезпечення. Це включає операційну систему, СУБД та графічні системи);

- прикладні системи, програми (прикладне програмне забезпечення включає прикладні програми, дані, документацію та засоби навчання користувачів);

- зовнішнє середовище (це зовнішні по відношенню до прикладної платформи та прикладного програмного забезпечення елементи системи, включаючи всі периферійні пристрої. Перевагою даної моделі є виділення зовнішнього середовища в незалежний елемент, який має певні функції та

відповідний інтерфейс, і можливість його застосування для опису систем, побудованих на основі архітектури «клієнт-сервер»);

– інтерфейс програми з платформою;

– інтерфейс платформи із зовнішнім середовищем.

По горизонталі виділяють наступні компоненти (функціональні зони):

– служби операційної системи (вони є корінними для забезпечення функцій платформи додатків);

– служба керування даними (є центральною для більшості систем щодо даних, які можуть бути визначені незалежно від процесів, які створюють ці дані та обмінюються ними);

– сервіси людино-машинного інтерфейсу (визначають спосіб взаємодії людини з прикладною програмою);

– служба обміну даними (забезпечує специфічну підтримку обміну інформацією, включаючи формат і семантику елементів даних між прикладними програмами на одній або різних платформах);

– служба забезпечення мережі (створює для розподілених прикладних програм можливості та механізми для доступу до даних і взаємодії між ними в неоднорідному мережевому середовищі)

– служба комп'ютерної графіки (забезпечує функції, необхідні для створення та обробки зображень на дисплеї).

Третій вимір:

– послуги підтримки розробки програмного забезпечення (що охоплюють стандартні мови програмування та засоби розробки програмного забезпечення);

– інтернаціоналізація (забезпечує мовну сумісність);

– послуги захисту інформації (призначені для забезпечення безпечного поширення інформації, цілісності інформації та захисту обчислювальної інфраструтури від несанкціонованого доступу);

– служба підтримки розподіленої системи (невід'ємна частина будь-якої операції, що виконується у функціональному середовищі відкритих систем). Він забезпечує механізми контролю та управління операціями, що здійснюються

окремими прикладними програмами в базі даних, системах, платформах, мережах, а також засоби взаємодії користувача з цими компонентами.

Профіль складається з вибраного списку стандартів та інших специфікацій, які визначають набір послуг, доступних для програм у певному домені. Приклади доменів можуть містити середовище робочої станції, середовище обробки транзакцій або середовище автоматизації офісу, вбудоване середовище керування процесом, розподілене середовище, щоб назвати декілька. Кожне з цих середовищ має різний перетин вимог до послуг, які можна вказати незалежно від інших. Кожна служба, однак, визначена в стандартній формі для всіх середовищ.

По суті, модель середовища відкритої системи є основним будівельним блоком кількох технічних еталонних моделей та технічної архітектури. Технічна архітектура ідентифікує та описує типи програм, платформ і зовнішніх об’єктів; їхні інтерфейси; та їхні послуги; а також контекст, у якому сутності взаємодіють.

Технічна архітектура є основою для вибору та подальшої реалізації інфраструктури для встановлення цільової архітектури.

Технічну еталонну модель можна визначити як множину послуг, організованих відповідно до концептуальної моделі, такої як модель середовища відкритої системи.

3.3 Висновок

У третьому розділі розроблено алгоритм побудови онтологічної моделі композитного вебзастосунку, що дозволило вдосконалити процес автоматичного створення загальної динамічної структури вхідних інформаційних ресурсів з можливістю урахуванням їх змісту.

Було запропоновано вдосконалений метод композитної збірки вебдодатків, який включає три етапи: вибір процесу; підбір компонентів; формування складу композитного вебзастосунку.

Для організації механізмів збору інформації використовувалася схема агентів керування запитами до бази знань ІС, на основі якого будується автоматизована

система композитної збірки вебдодатків. Для цього також запропоновано алгоритм перетворення інформації про процеси та компоненти вебзастосунку із загальної моделі, у модель бази знань онтологічного типу.

4 ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ КОМПОЗИТНОЇ ЗБІРКИ ВЕБДОДАТКІВ

4.1 Архітектура системи динамічної інтеграції компонентів

Для реалізації інформаційних технологій динамічної інтеграції інформації у вебсистемах розроблено технологію для динамічної інтеграції інформаційних ресурсів та їх подачі у вигляді пов’язаних даних на основі запиту користувача. У таких застосунках автоматизовано процес структурування даних, що містяться в вебінтегрованих інформаційних системах, в онтологічну модель, яка потім застосовується для пошуку даних на основі запиту користувача.

Основними етапами розробки будь-якої програмної системи є: встановлення завдання, яке визначає параметри високого рівня (функціональні та операційні вимоги, інтерфейс, вимоги до безпеки та надійності); розробка архітектури та вимог до продукту; кодування, при якому програмний код системи отримують на основі архітектури та її вимог; компіляція для отримання вихідного коду та завантаження вихідного коду.

Система динамічної інтеграції інформації у композитний вебдодаток реалізована як набір повних модулів, які можуть бути використані для побудови інших систем.

Структурна модель системи динамічної інтеграції інформації в композитний вебдодаток включає наступні блоки (рис. 4.1):

- блок інформації про систему;
- блок даних про інформаційні джерела;
- адаптери для роботи з інформаційними джерелами;
- підсистема формування завдання;
- блок отримання доступу до джерел з інформацією;
- підсистема формування агрегованого динамічного набору даних;
- блок опису структури вхідних даних і їх і змісту;
- блок зберігання даних;
- блок підтримки роботи системи;

– модуль візуалізації результатів роботи системи.

Рисунок 4.1 - Структурна модель системи на основі Mashup

Як бачимо з рис. 4.1 для роботи із будь-яким джерелом ресурсів необхідно використовувати такий елемент, як об’єктний адаптер. Структуру об’єктного адаптеру наведено на рисунку 4.2. Він складається із:

– блоку опису ресурсу, який служить для генерування структури даних ресурсу;

– блоку відображень, який використовується для отримання правил відображення;

– блоку перехоплень, що застосовується для перехоплення операцій системи відповідно до ресурсних та картографічних даних, одержання доступу до нового ресурсу і для повернення результатів до системи.

Рисунок 4.2 - Об’єктний адаптер системи динамічної інтеграції

Модуль опису структури та змісту вхідних даних має такі функції:

– експорт метаданих для структури (здійснює підключення до інформаційної системи для отримання структурної інформації та аналізу ступіння тотожності елементів структури, додавання смислових зв’язків між компонентами на основі аналізу, експорту онтології);

– розширення онтології для структури (додавання онтологій домену, онтологій верхнього рівня, смислових посилань, експорт в необхідному форматі розширеної онтології);

– одержання метаданих інформаційних ресурсів (імпорт попередньо розширеної онтології структури, зєднання з інформаційною системою, видалення метаданих, аналіз ступіння схожості, додавання необхідних семантичних зв’язків на основі аналізу, додавання отриманих семантичних зв’язків на основі онтології, експорт онтології до репозитарію).

Блок опису структури вхідних даних і їх і змісту реалізовано у вигляді таких основних компонентів:

– підсистема для генерації шаблону для запиту даних із інформаційних ресурсів;

– підсистема створення та зміни онтологічних описів, що відповідають за створення та модифікацію понять та ієрархій у початковій онтології;

– репозитарій метаданих, забезпечує зберігання описів доступних онтологій предметних областей;

– підсистема порівняння та об’єнання онтологій, виконує інтерграцію онтологій з різних предметних областей;

– підсистема для вирішення можливих конфліктів при порівнянні понять онтології;

– підсистема, що служить для формування моделі кожного входу інформаційного ресурс, визначени;

– допоміжні компоненти (адаптери, конфігуратори).

Модуль генерування вмісту Mashup виконує:

– аналіз доступних семантичних метаданих вхіднох інформації (імпорт та деталізація отриманих семантичних метаописів ресурсів, виділення та експорт даних у репозитарій);

– визначення відповідності запиту з семантичними метаописами вебресурсів (імпорт метаописів та виділення пошукового зображення запиту, аналіз релевантності запиту користувача та метаописів, надання релевантних метаописів та експорт даних у репозитарій);

– керування результататами (отримання з репозитарію даних, що відображає метаописи, релевантні запиту користувача, ранжування отриманих даних за критерієм подібності подібності, відображення отриманих даних).

Вона реалізована наступним чином:

– підсистема обробки запитів користувача відповідає за перевірку введення даних для запиту і контролює імпорт метаописів інформаційних ресурсів із

репозитарію метаданих, отримування пошукового зображення запиту, аналіз релевантності запиту користувача та метаописів, експорт даних у сховище;

– підсистема редагування даних, яка враховує зміни, внесені до запиту користувача, на етапі коригування, завдяки інтерактивній комунікацій з модулем обробки запиту;

– підсистема відображення даних, що відповідає за ранжування даних за схожістю та новизною їх публікації на ресурсі;

– допоміжні компоненти (адаптери, конфігуратори).

Робота програмної системи, полягає у формуванні уніфікованого набору динамічних компонент, отриманих з різних джерел відповідно до запиту користувача. Автоматизована система знаходить та відображає інформацію, яка зберігається в слабоструктурованих неоднорідни інформаційних вебресурсах, інтегрованих у онтологічну модель, яка потім застосовується для пошуку інформаційних ресурсів відповідно до зформованого запиту користувача.

Вхідна інформація - це запит користувача, дані про джерела інформаційних ресурсів та про самі вебресурси.

Вихідна інформація - це множина метаописів вебресурсів з посиланнями на джерела ресурсу.

При розробці систему для композитної збірки малоструктурованих даних у вебзастосунок, необхідно враховувати трирівневу організацію обробки даних, яка використовується при еталонній побудові систем Mashup (рис. 4.3).

Рисунок 4.3 - Організація опрацювання даних на основі технології Mashup

При роботі системи передбачені такі категорії користувачів:

– адміністратор;

– модератор;

– аналітик (експерт, який приймає аналітичні рішень у системі);

– користувач (отримує необхідну йому інформацію, як взаємопов'язаний набір даних на основі зформованого пошукового запиту).

Для досягнення коректної злагодженої роботи всіх модулів системи використовується принцип інтерпретації метамоделі формування результату роботи системи. Системна метамодель — це інформаційна модель з вищим рівнем абстракції, ніж поточна модель домену. Метамодель охоплює та описує функціональні можливості не окремих дій, а широкого спектру дій, виділяючи загальні абстракції, правила опрацювання даних і керування бізнес-процесами в цих діях. Метамодель адаптується до специфіки, необхідної вже в момент виконання, перетворюючи метадані в динамічний код і гарантуючи його динамічний зв’язок із середовищем запуску. За допомогою метамоделі система розпізнає метадані інформаційних активів і динамічно створює модель предметної області на основі доменої моделі, яка вже інтерпретує вхідні дані. Метадані дозволяють метамоделі побудувати динамічну модель проблеми, яку потрібно вирішити, тобто метадані інтегрують опис даних і здатність метамоделі інтерпретувати отриману інформацію.

Метамодель вбудована в кожен компонент інформаційної системи та містить абстрактні функції для широкого класу діяльності. При запуску компонентів, метамодель розгортається на прикладній віртуальній машині, але необхідно отримати дані та метадані, щоб інтерпретувати їх динамічно, тобто перетворити їх у модель домену або проблеми, яку потрібно вирішити. Інтерпретація динамічної метамоделі — це процес створення моделі домену з метамоделі, метаданих і даних. Динамічна інтерпретація відбувається як на стороні клієнта, так і на стороні сервера під час виконання. При динамічній інтерпретації модель предметної області не повністю вбудована в пам’ять, а інтерпретується по частинах у міру обробки вхідних даних і може бути кешована у віртуальній машині додатка, щоб уникнути витрат на ресурси для кількох інтерпретацій.

Блок опрацювання запиту від користувача відповідає за виконання наступних завдань:

– аналіз коректності подання запиту;

– виділення для кожного ресурсу певної кількості метаописів джерел, яка далі буде використовуватися для аналізу на схожість із пошуковим образом запиту.

– отримання пошукового образу запиту (виділення термінів із пошукового запиту та семантично подібних терміни за допомогою використання загальної онтології предметної області) та формування асоціативних зв’язків;

– аналіз релевантності запиту користувача та метаопис (знаходження коефіцієнтів подібності запиту і метаопису);

– додання подібних метаописів до знайдених за допомогою коефіцієнта подібності метаописів;

– зберігання отриманої інформації у репозитарій, якщо у системі передбачено дану процедуру.

Дана інформація має записуватися до бази даних системи у структурі спеціально створених взаємопов’язаних таблиць окремо для кожного вебресурсу (рис. 4.4).

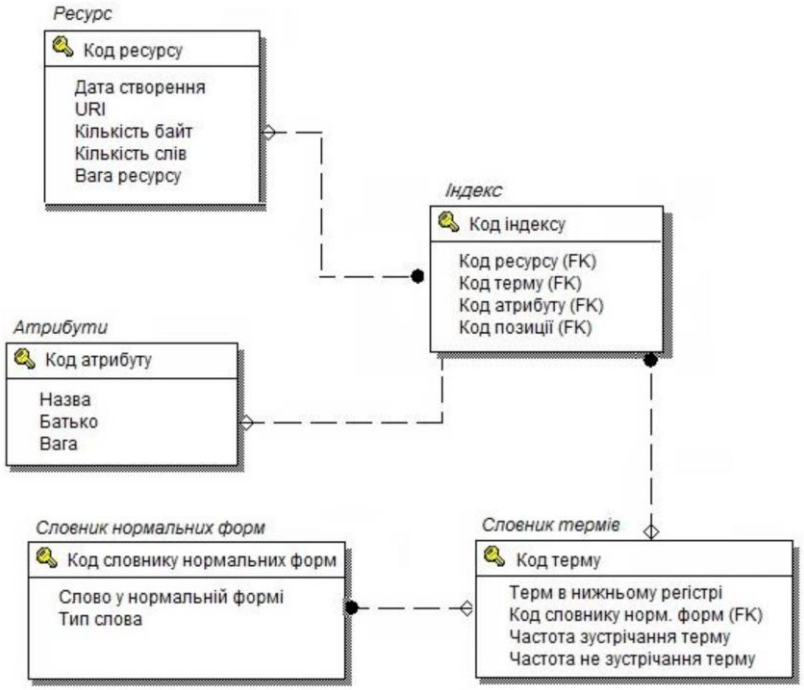


Рисунок 4.4 – Таблиці для зберігання індексів метаописів вебресурсів

Метамодель описує специфікацію метаданих, необхідних для їх динамічної інтеграції. Якщо це перший сеанс компонента, і в постійній пам'яті ще немає доступних метаданих, їх можна отримати за допомогою ідентифікатора URI (адреси) або запиту, введеного користувачем або переданого із системи іншого компонента (але ще не підключені в компоненті). Цим процесом керує компонент сеансу віртуальної машини програми. Пізніше метадані можна кешувати, щоб мінімізувати кількість запитів і оновлюватися лише при змынні версія чи контрольна сума в джерелі (репозитарій зберігання даних чи віддаленому компоненті). Отримання модифікованих метаданих призводить до реструктуризації моделі домену в RAM.

Отримані таким чином метадані в системних компонентах інтегрують метамодель з її декларативними параметрами та імперативними сценаріями, завдяки чому стає можливим динамічно будувати класи домену, наповнювати їх отриманими даними та обробляти їх за допомогою динамічно побудованої бізнес-логіки. У результаті як функціональний компонент, так і інтерфейси для взаємодії з іншими компонентами можуть змінюватися у великому класі активності без перекомпіляції системи, і багато змін можливі просто шляхом встановлення нових параметрів метаданих (навіть без модифікації сценаріїв).

Динамічно створювані структури даних програми та сценарії програми (події, методи обробки даних) базуються на викликах API віртуальної машини програми, на якій працює динамічна модель. Цей API, у свою чергу, «мешкає» в API середовища запуску. Динамічна побудова моделі відбувається як у серверному, так і в клієнтському (користувацькому) компонентах інформаційної системи, тобто інтерфейс формується динамічно, а використовувана бібліотека відображення (компоненти GUI) виконує роль API.

Користувач ініціює події в GUI, які призводять до локальної обробки даних чи мережевих викликів. При динамічній інтеграції ідентифікатори віддалених процедур, їх параметри та структура даних повертаються, не пов'язані з викликаним компонентом, а оримуються із метаданих під час процесу динамічної інтерпретації метамоделі. Віддалена ресурс також не має фіксованого інтерфейсу,

але заздалегідь відомі лише протокол обміну (типи параметрів, які підтримуються для обробки, формати передачі даних) і механізм самоаналізу, який він створює.

Роль проміжної машини в динамічній інтерпретації полягає в підтримці сеансу з необхідним набором даних як на клієнті, так і на сервері між мережевими сесіями. Це необхідно для забезпечення послідовної або інтерактивної людино-машинної обробки даних, коли дії користувача приводять модель у певний стан і кожен наступний сеанс залежить від прецеденту в рамках транзакції. Крім того, кінцевий автомат може значно знизити вартість трафіку та обчислювальних ресурсів за рахунок кешування метаданих.

Таким чином, багаторівнева структура компонентів інформаційної системи контролюється знизу в репозиторії, обмеженому рівнем абстракції метамоделі, і контролюється зверху користувачем, який може змінювати не тільки дані, але й метадані, що перенаправляє інформаційної системи до безперервних змін у потребах розв'язуваної проблеми, полягає в гнучкості підходу та спрощенні модифікації системи, знижуючи рівень кваліфікації користувача, необхідний при внесенні змін до структури і функцій. Але зрештою цей підхід окупає себе за рахунок збільшення повторного використання коду з високою абстракцією, автоматизації багатьох завдань зв’язування компонентів, зменшення людського впливу модифікації та інтеграції системи та спрощення інтеграції між компонентами системи програмного забезпечення.

Було застосовона такі вимоги при використанні стандартів при розробці системи:

– система має використовувати стандартні, рекомендовані консорціумом W3C семантичні вебтехнології;

– онтологічна модель має бути описана з використанням мови OWL у форматі RDF;

– доступ до інформації повинен здійснюватися за допомогою спеціалізованої мови запитів SPARQL;

– система завдяки механізму простору імен, має підтримувати використання існуючих онтологій верхнього рівня;

– система має реалізувати запропоновані рішення базуючись на стандарти відображення реляційних баз даних;

– використання вебсервісів SOAP та адаптерів бази даних JDBC, для звернення до інтегрованих систем.

4.2 Розроблення системи композитної збірки

Система має архітектуру, що складається з ядра, рівня основних базових операцій та рівня функціонального розширення. Вона розгорнута на платформі Eclipse (рис.4.5). Eclipse побудований у вигляді набору підсистем, що розширюються, а не як єдиний монолітний додаток. В Eclipse входять три підпроекти, що розробляються більш-менш незалежно один від одного – Platform, PDE (Plug-in development environment) та JDT (Java development tools) [23]. Platform надає базові сервіси та служби, JDT дозволяє створювати програмне забезпечення на Java, а PDE – нові компоненти Eclipse. Надалі, визначившись з метою, ви можете завантажити та використовувати будь-яку збірку, відповідну для ваших завдань, вже обладнану необхідними модулями та розширеннями.

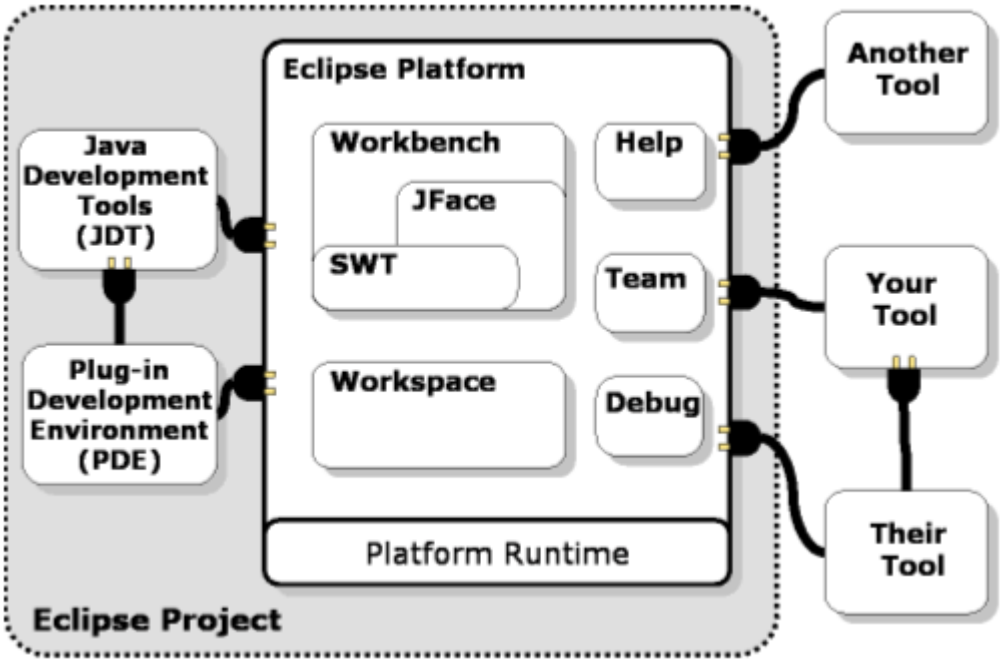


Рисунок 4.5 – Платформа Eclipse

Platform` є центральною частиною Eclipse. Сама по собі ця підсистема не містить особливо корисних для користувачів функцій, але без неї неможлива робота інших сірвісів Eclipse. Сервіси, які надає платформа, дозволяють розробникам визначати видимі для користувача артефакти, створювати інтерфейси користувача, працювати з системами контролю версій, середовищами налагодження та довідковою системою. Відповідними компонентами платформи, які реалізують ці служби, є Workspace (керування вмістом), Team, Workbench (базовий інтерфейс користувача Eclipse), Debug і Help. Основними поняттями, визначеними Workspace, є проект (project), робоча область (workspace), папка (folder) і файл (file). Всі ці об'єкти називаються ресурсами (resources) у термінології Eclipse.

```
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.0"?>
<plugin
  id="com.mycompany.propbuilder"
  name="Property Builder Plug-in"
  version="1.0.0"
  provider-name="MYCOMPANY"
  class="com.mycompany.propbuilder.PropertyBuilderPlugin">

  <runtime>
    <library name="propbuilder.jar">
      <export name="*"/>
    </library>
  </runtime>

  <requires>
    <import plugin="org.eclipse.core.runtime"/>
    <import plugin="org.eclipse.core.resources"/>
  </requires>
</plugin>
```

Рисунок 4.6 – Плагін для імпорту Eclipse

Ресурси можуть бути локальними або розташованими в репозитарії керування джерелами. В обох випадках доступ до вмісту ресурсу та керування ним здійснюється через стандартний інтерфейс користувача, спільний для локальних і віддалених об’єктів.

Більшість коректно розроблених програм Eclipse ніколи не звертаються до файлів безпосередньо, використовуючи для цього стандартну Java або будь-які інші засоби. Натомість за допомогою інтерфейсів робочої області здійснюється доступ до файлової системи. Однак використання стандартних інструментів Workspace є необов’язковим — якщо це з будь-якої причини неприйнятно. Цілком

прийнятно використовувати альтернативні механізми. Маркер — це об’єкт, прив’язаний до певної позиції в конкретному ресурсі, який має кілька заздалегідь визначених атрибутів і певну кількість атрибутів, визначених розробником. Стандартні атрибути, такі як текст повідомлення маркера або його позиція у файлі, визначають, як маркер буде відображатиметься платформою Eclipse. Додаткові атрибути мають можливість відображати нові функції програми. Наприклад, в атрибутах повідомлення про помилку може міститися інформація для автоматичного виправлення помилок. Те, як платформа Eclipse інтерпретуватиме конкретний проект, повністю залежить від його характеристик. Найважливішими властивостями, які підтримує платформа, є характер проекту та конструктори.

Конструктори, у найзагальнішому розумінні, — це компоненти, щообробляють ресурси, включені в проект, наприклад, компілятор Java, який перетворює вихідні тексти у двійкові файли .class. Після первинної розробки проекту конструктори повинні реагувати на зміни, які відбуваються в його ресурсах. Ідея інкрементної обробки тут дуже важлива: щоб підвищити продуктивність, компонувальники повинні, якщо це можливо, обробляти лише ті ресурси, які змінилися з часу останнього запуску конструктора.

Ядро системи включає сервер додатків Tomkat 10.0 (рис. 4.7).

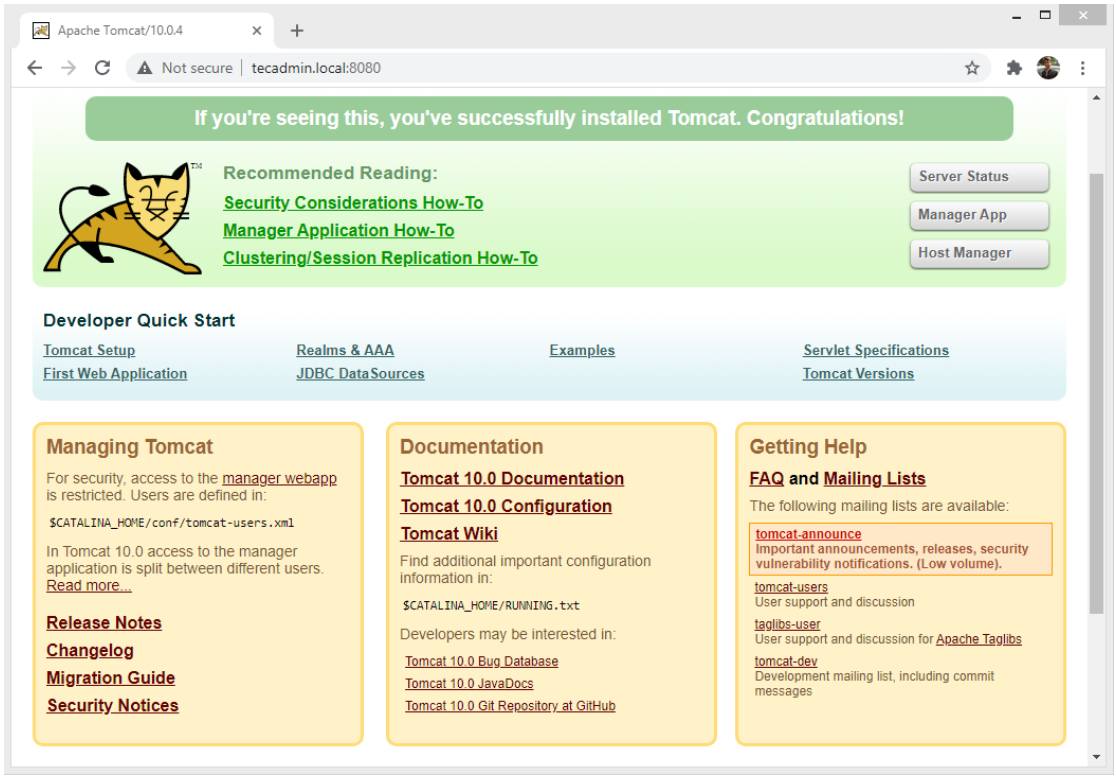


Рисунок 4.7 - Сервер додатків Tomkat 10.0.

Apache Tomcat є однією з небагатьох програм із відкритим кодом за своєю природою [13]. Представляє контейнер сервлета. Програмний продукт відповідає за реалізацію специфікації сервлету. Існує також підтримка двох інших важливих специфікацій від JavaServer. Додаток працює на Java. Ця програма містить багато інших менш значущих утиліт. Використовуючи програмне забезпечення, ви можете працювати з будь-яким вебдодатком. Але багато хто вирішує завантажити Apache Tomcat з однієї причини: щоб надати програмному продукту роль справжнього вебсервера. Програма може взяти на себе роль сервера контенту. Працюйте неповний робочий день із власним HTTP-сервером Apache. Або, як згадувалося раніше, діяти як контейнер сервлетів. Цей контейнер уже буде безпосередньо підключений до інших серверів додатків: GlassFish / JBoss.

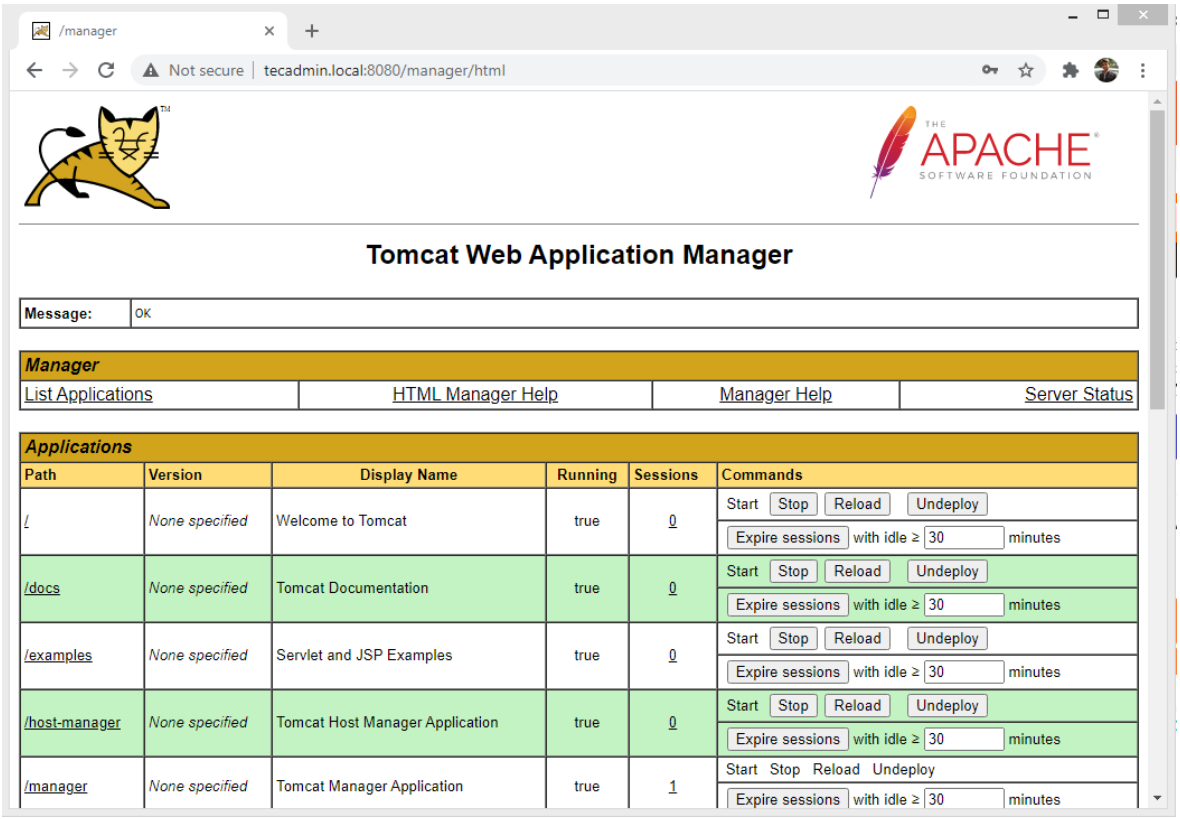


Рисунок 4.8 – Функціонал Tomkat 10.0

Всього програмний продукт складається з п'яти компонентів, кожен з яких по-своєму важливий і відповідає за особливі завдання. Java Apache Tomcat виділяється тим, що підтримується низкою сторонніх розробників-ентузіастів,

завдяки яким кінець епохи додатків настане нескоро. Розробник має доступ не тільки до вихідного коду, а й до бінарних файлів. Наступною частиною програми є Servlet API. Без цього компонента інші технології Java не можуть бути реалізовані. Ми говоримо про технології, які безпосередньо пов'язані з Інтернетом.Завдяки цим функціям веб-контент генерується динамічно, незважаючи ні на що. Для цього використовуються спеціальні бібліотеки. Це окремий механізм у сервері Apache Tomcat (JSP). Програма працює на другій версії Jasper. Механізм відповідає за розбір спеціальних файлів. Потім вони компілюються в код. Важливу роль у програмному забезпеченні відіграє компонент Coyote. Це стек HTTP. Ця підтримка потрібна як з боку контейнера програми, так і безпосередньо з веб-серверів. Цей компонент відповідає за вхідні підключення, точніше за їх перехоплення [17].

Spring Framework, або просто Spring, є одним із найпопулярніших фреймворків для створення вебзастосунків Java. Простіше кажучи, використовуючи бібліотеку, ви просто створюєте об’єкти класів, які вона містить, викликаєте потрібні вам методи, а потім отримуєте потрібний результат [12]. Найчастіше ваші класи реалізують деякі інтерфейси з фреймворку або успадковують деякі класи з нього, таким чином отримуючи частину функціональності, яка вже написана для вас. У фреймворку намагаються максимально відійти від такого жорсткого зв’язку (коли класи безпосередньо залежать від деяких класів/інтерфейсів цього фреймворку) і використовують для цього анотації. Але важливо розуміти, що spring - це просто набір якихось класів та інтерфейсів. Spring можна використовувати не тільки для вебдодатків, але і для самої звичайної консольної програми.

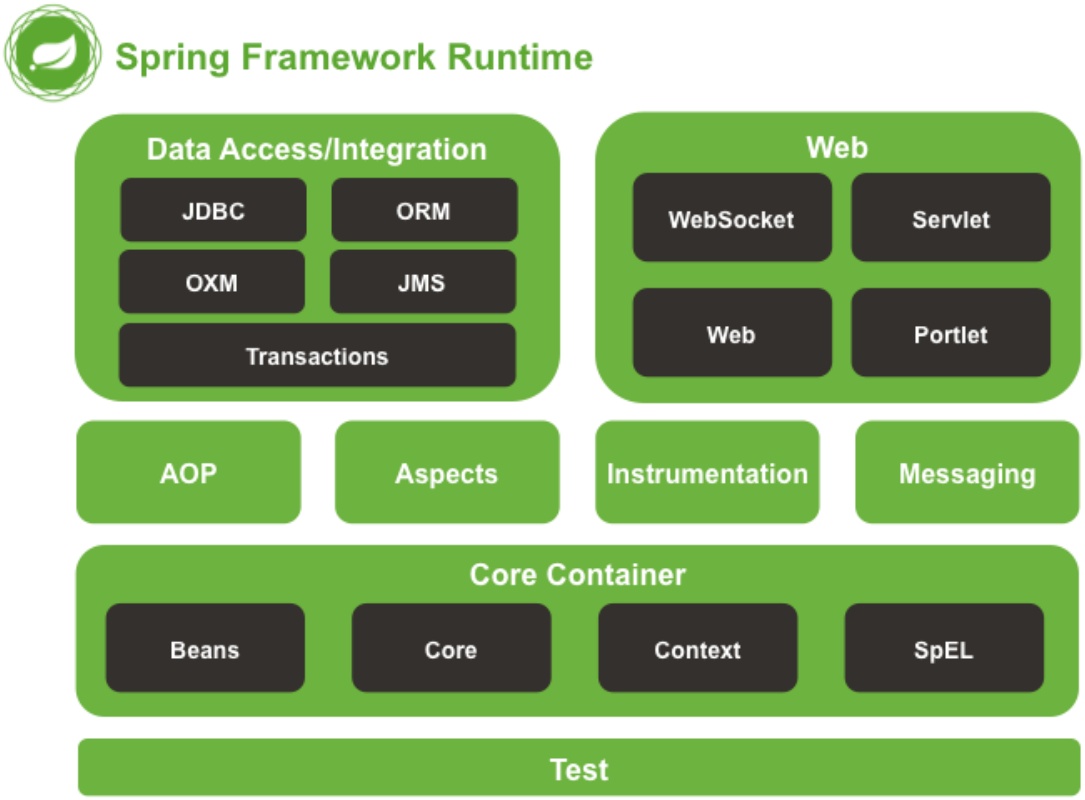


Рисунок 4.9 – Функціонал Spring Framework

2. Apache Jena Fuseki — це сервер SPARQL, який відкриває дані RDF, що зберігаються в triplestore, через HTTP та API у стилі REST. На цій сторінці наведено інструкції щодо встановлення та налаштування самокерованого сервера Apache Jena Fuseki лише для цілей розробки та тестування (тобто невиробничого) [14].

Щоб налаштувати сервер Apache Jena Fuseki для збереження даних RDF triplestore на диску, а отже, тривожного зберігання під час перезапусків сервера, ми можемо використовувати –loc параметр під час запуску сервера Fuseki таким чином.

```
# Перейдіть до каталогу встановлення Fuseki
cd /opt/apache-jena-fuseki/apache-jena-fuseki-4.3.1

# Запустіть сервер Fuseki за допомогою дискового сховища
./fuseki-server --loc=/opt/apache-jena-fuseki/apache-jena-fuseki-4.3.2/data --update /ontopop
```

Рисунок 4.10 – Встановлення Apache Jena Fuseki

3. Apache Jena Fuseki — це сервер SPARQL. Він може працювати як служба операційної системи, як веб-додаток Java (файл WAR) і як окремий сервер [22].

```
java -cp jena-fuseki-server-$VER.jar:...OtherJars... \
    org.apache.jena.fuseki.main.cmds.FusekiMainCmd ARGS
```

Рисунок 4.11 – Точка входу до серверу

Fuseki доступний у двох формах: єдиний системний у поєднанні з інтерфейсом користувача для адміністратора та запитів, а також як «головний» сервер, придатний для роботи як частини більшого розгортання, включно з Docker або запущеним вбудованим. Обидві форми використовують той самий механізм ядра протоколу та однаковий формат файлу конфігурації.

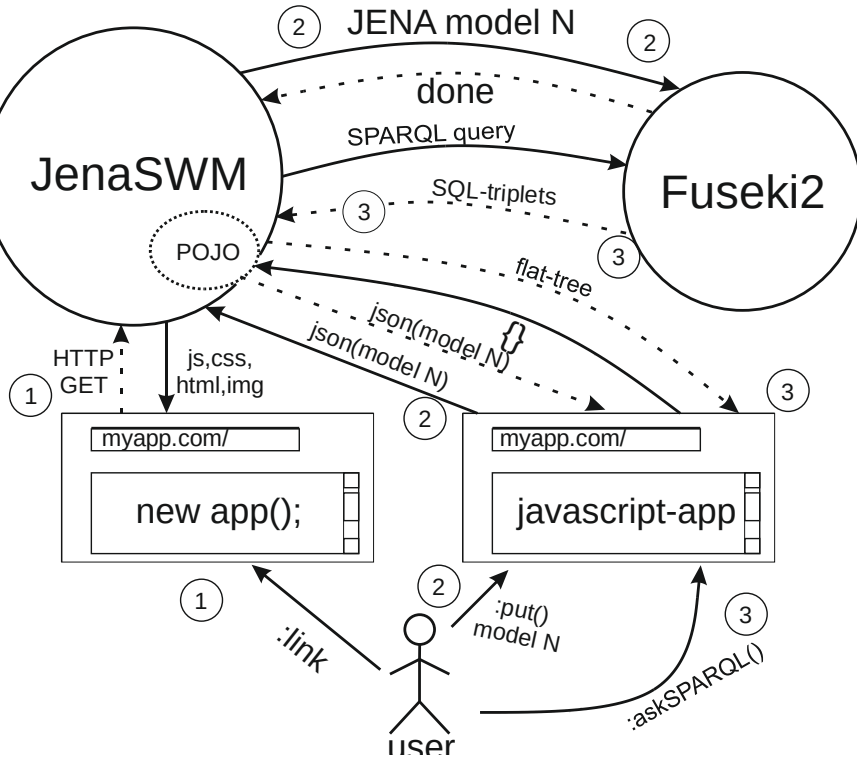


Рисунок 4.12 – Архітектура використаних технологій для розробки застосунку

Представлене в роботі рішення дозволяє сформувати платформу, на якій буде зформовано єдиний інформаційний простір зі специфікаціями та системою складання проєктування профілів, заснованих на семантичному описі структури. Усі вузли онтологічного зформованого графу визначають конкретні документи та пов’язані з ними блоки.

На стороні програми Java розроблено модель, яка може відображати будь-яку структуру, побудовану відповідно до принципів роботи специфікацій JSON і OWL. На цьому рівні присутній рівень API, який додає нові структури без перекомпіляції системи, а на платформі NodeJS реалізований блок керування моделлю Model Manager, що дозволяє працювати з мовою javascript. Шар функціонального розширення дозволяє модифікувати структуру моделей на рівні структурної області та забезпечити підтримку механізмів на рівні базової роботи. Модуль керування моделлю Model Manager може служити платформою для роботи зі службами Jena, оскільки ці моделі можна завантажувати та змінювати на льоту (режим виконання) без необхідності компілювати чи перезавантажувати.

Використовуючи запропоновану технологію розроблено прототип програмного засобу, для автоматизованого процесів збору даних, обробки даних та їх представлення як сервісу в системі, що реалізує метод композитної збірки вебдодатків, використовуються Discounts Mashup, систему пошуку знижок на купівлю товарів, подорожей, різноманітних сервісів, тощо. Представлено програмні реалізації розроблених систем з підсистемами, що працюють на базі запропропонованох технології.

Процес обробки даних у композитних вебзастосунках відбувається за допомогою методу, розробленого для опису структури та змісту вхідних даних, та алгоритмів формування набору агрегованих динамічних даних, які мають загальну структуру і єдиний зміст

Вхідними даними систем у вебзастосунку Discounts Mashup є джерела інформаційних ресурсів, до яких система має доступ, і запит користувача. Користувач формує запит і отримує відповідь у вигляді набору інформаційних ресурсів доступних джерел, що відповідають даному запиту.

Приклад роботи з системою можна побачити на рисунку. При введені в запиті «знижки на подорожі» ми отримаємо такий композитний вебзастосунок (рис. 4.13).

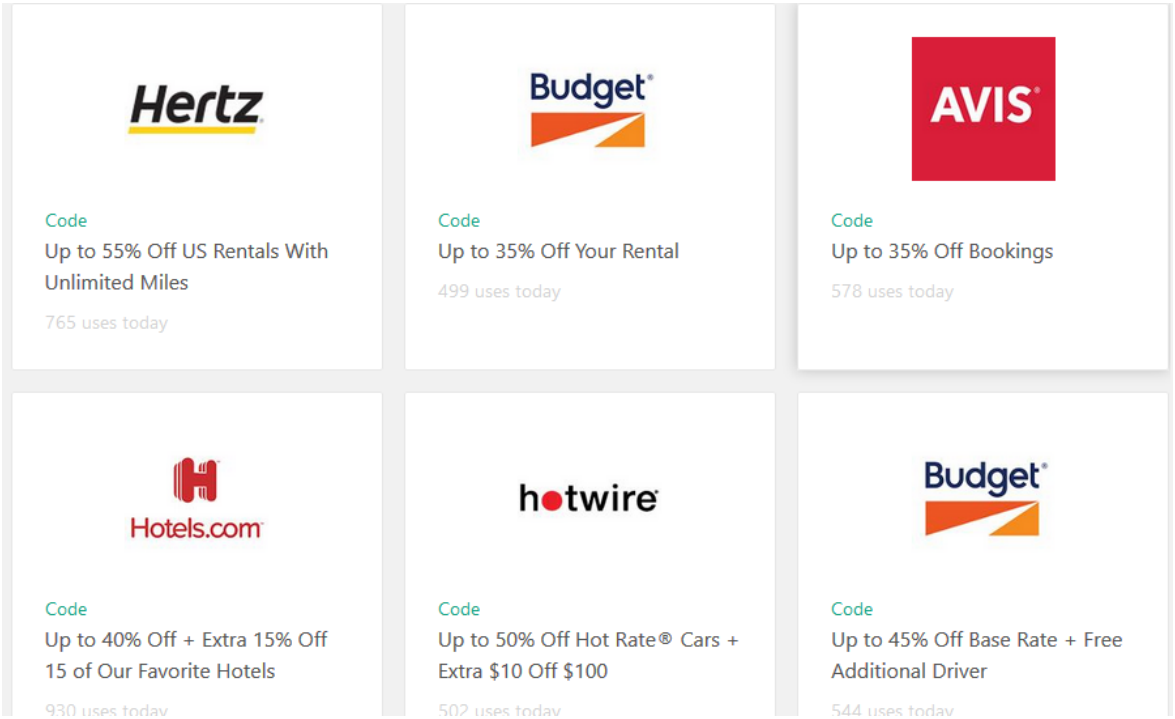


Рисунок 4.13 - Результат роботи системи Travels and Discounts Mashup

Надзвичайно зрозуміла і проста схема роботи даного композитного вебзастосунку дає змогу користувачу із мінімальними навиками володіння комп’ютерними технологіями здійснити пошук купону на знижку для подорожі чи проживання. Далі, знайшовши потрібну знижку, користувач має можливість перейти до вебресурсу, який надає купон на знижку і ознайомитися із умовами акції більш детальніше.

Після проведення низки експериментів щодо формування композитного вебзастосунку, що релевантність інформаційних ресурсів знаходиться в межах 92-93%, а кількість пертинентних посилань серед перших десяти, одержаних унаслідок пошуку в середньому дорівнює вісім.

4.3 Висновок

У четвертому розділі запропонован реалізація тестової автоматизованої системи композитної збірки.

На основі методу, запропонованого в розділі 3, була побудована трирівнева модель загальної архітектури, щоа включає ядро, рівень базових операцій та має рівень функціонального розширення, і також платформу, що забезпечує інтероперабельність компонент для формування стандартизованих профілів.

Запропонована прикладна модель архітектури системи, яка грунтується на трирівневій моделі архітектури, виконана:

– на рівні ядра у вигляді MVC-системи в фреймворку Java-Spring, що включає рівень Jena і сервера Tomcat 10 та Fuseki.

– на рівні базових операцій використовується сукупність контролерів, моделей та представлень, які забезпечують безпосередньо розгортання розподіленої системи і взаємодіюя з RDF сервером для виконання операцій з онтологіями;

– на рівні функціонального розширення використовується блок керування моделями Model manager, який реалізований на платформі NodeJS з використанням рушія Google.

ВИСНОВКИ

Зміст кваліфікаційної роботи охоплює завдання та рішення побудови композитних вебдодатків шляхом вдосконалення інформаційної технології адаптивної інтеграції слабоструктурованої інформації у вебсистемах.

Результати проведеного дослідження узагальнені в наступних пунктах:

1. Проведено аналітичний огляд вже існуючих готових рішень для вебсистем, проаналізувано Semantic Web концепцію при побудові вебзастосунків, її переваги та недоліки.

2. Було формалізовано семантичну модель стандартизованого профілю системи, а також формалізовану модель онтологічної бази знань на її основі. Запропонована модель включає 3 рівні: рівень специфікації, рівень домену, рівень інформаційної системи. Модель визначає загальну структуру, в межах якої можуть бути побудовані прикладні семантичні моделі, які застосовуються безпосередньо на практиці.

3. Запропоновано підхід до підвищення якості результату процесу визначення вхідних даних системи збору інформації шляхом застосування процедури опису структури та змісту вхідних інформаційних ресурсів, що дає змогу покращити якість процесу інтеграції даних з різних джерел для подальшого їх використання в системі збірки вебзастосунку. Для цього була розроблена модель процесу композитної збірки вебдодатків у вигляді семантичної мережі з використанням продукційних правил.

4. Розроблено алгоритм побудови онтологічної моделі композитного вебзастосунку, що дозволило вдосконалити процес автоматичного створення загальної динамічної структури вхідних інформаційних ресурсів з можливістю урахуванням їх змісту.

5. Вдосконалено метод композитної збірки вебдодатків, який включає три етапи: вибір процесу; підбір компонентів; формування складу композитного вебзастосунку. Для організації механізмів збору інформації використовувалася схема агентів керування запитами до бази знань, на основі якого будується автоматизована система композитної збірки вебдодатків. Для цього також

запропоновано алгоритм перетворення інформації про процеси та компоненти вебзастосунку із загальної моделі, у модель бази знань онтологічного типу.

6. В роботі розроблено інформаційну технологію та програмні застосунки, які дають можливість зменшення інформаційного шуму до 7-8% та покращення узгодженості результатів у системі композитного формування вебзастосунку за технологією Mashup.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Бахрушин В. Є. Математичні основи моделювання вебсистем: Навчальний посібник для студентів. / Бахрушин В.Є. О.І. Михальов – Запоріжжя: Класичний приватний університет, 2015. – 224 с.

2. Джулій В.М. Методи і алгоритми розробки вебдодатків / В.М. Джулій, Ю.О. Гунченко, Д.В. Чешун // Збірник наукових праць Військового інституту Київського національного університету імені Тараса Шевченка. – К.: ВІКНУ, 2017. – Вип. № 56. – с.107-115.

3. Жежнич П.І. Основні правила побудови семантично відкритих інформаційних вебсистем /П.І. Жежнич, Р.Б. Кравець, А.М. Пелещишин // Інформаційні системи та мережі: Вісник Нац. ун-ту “Львівська політехніка - 24 2019. - №383. - С. 84-95.

4. Клапчук Р. Г. Монолітні вебсервіси: порівняння та вибір / Р. Г. Клапчук, В. С. Харченко // Радіоелектронні і комп'ютерні системи. - 2017. - № 1. - С. 51-56. - Режим доступу: http://nbuv.gov.ua/UJRN/recs 2017.

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

6. Муляр І.В., Войнарович С.Б Аналіз підходів до структурної збірки вебзастосунків // Тези доповідей XII Міжнародної науково-практичної конференції; Військова освіта і наука: сьогодення та майбутнє . 28квітня 2017.– К. : ВІКНУ, 2017, с 96

7. Муляр І.В. Метод опрацювання слабоструктурованих даних у вебсистемах / І.В. Муляр, В.Р. Жила, Л.О. Ряба // Тези доповідей Всеукраїнської науковоїконференції молодих вчених, ад’юнктів, слухачів, курсантів і студентів «Молодіжна військова наука у Київському національному університеті імені Тараса Шевченка» 25квітня 2019 року. / за заг. редакцією І.В. Толока. –К. : ВІКНУ, 2019. – с. 132.

8. Мілер В.П. Методи проектування захищенних вебдодатків / К.В. Молодецька, В.І. Чорненький, А.А. Берназ В.П. Мілер // Тези доповідей XVI Міжнародної практичної конференції "Військова освіта і наука: сьогодення та майбутнє" Том 2 [Текст] / за заг. редакцією Ігоря Толока. – К. : ВІКНУ, 2020. – С. 49-50

9. Пічура В.Ю. Аналіз підходів до проєктування децентралізованих систем з використанням технології блокчейн / І.В. Муляр, О.В. Мірошніченко, А.Р. Віхтюк, В.Ю. Пічура, Л.В. Солдєва // Тези доповідей XVIII Міжнародної науково-практичної конференції " Військова освіта і наука: сьогодення та майбутнє" – К. : ВІКНУ, 2022. – 57 с.

10. Петрик М.Р. Моделювання програмних додатків : науково-методичний посібник / М.Р. Петрик, О.Ю. Петрик - Тернопіль : Вид-во ТНТУ імені Івана Пулюя, 2015. - 200 с.

11. Пелещишин А.М. Використання апарату абстрактних автоматів для моделювання вебсистем / А.М. Пелещишин // Інформаційні системи і мережі: Вісник Нац. ун-ту “Львівська політехніка”. - 1998. - №330. - С. 188-201.

12. Проскудіна Г.Ю. Онтології у інформаційних системах та модулях / Г.Ю. Проскудіна., О.М. Овдій //Теоретичні і прикладні аспекти побудови програмних систем: спец. випуск Вісника Київськ. нац. ун-ту ім.Т.Г. Шевченка, 2004. - С.164-169.

13. Щербакова М. Є. Функціональні особливості Java-додатків / М. Є. Щербакова, Є. В. Щербаков // Вісник Східноукраїнського національного університету імені Володимира Даля. - 2014. - № 10. - С. 142-146.

14. Barlow K. Like Technology from a Advanced Alien Culture: Google Maps for Education at ASU / K. Barlow, J. Lane // In Proceedings of Special Interest Group on University Computing Centers, Indiana, 2017. - P. 8-10.

15. Chong E. An efficient SQL based RDF querying scheme / E. Chong, S. Das, G. Eadon, J. Srinivasan // VLDB, 2015.

16. Computer science [Електронний ресурс] / Д. Барашев // Big Data’13. Режим доступу: URL: http://compscicenter.ru/sites/default/files/materials/2019_04_18_BigData_lecture_09.pdf.

17. Duda C. Ajaxsearch: crawling and searching Web applications / C. Duda, G. Frey, D. Kossmann, C. Zhou // *PVLDB,* 2018. - P. 1440-1443.

18. Eberrichi, M. Malki // In Proceedings of World of Science, Knowledge and Technology, USA, 2016. - P. 11-18.

19. Ehrig M. Research and applications / M. Ehrig, Y. Sure // Proc. 1st European Semantic Web Symposium. LNCS. Berlin: Springer, 2004. - P. 3053.

20. Fisher T. An overview of approaches Web generation / T. Fischer, F. Bakalov, A. Nauerz // Proceedings of the International Workshop on Knowledge Services and Mashups, 2019, P. 157-158.

21. Heflin J. Searching the Web with SHOI / J. Heflin // Defense Technical Information Center, 2020.

22. Hendrik. Towards Semantic Web Tools / Hendrik, A. Anjomshoaa, A. Tjoa // Information and Communication Technology, Volume 8407, 2014. - P. 129-138.

23. Herron D. Node.js Web Development: Server-side development with Node / David Herron., 2018. - 492 c. - (4th Revised edition).

24. Ontology Language [Електронний ресурс]. Режим доступу: URL: http://www.w3.org/TR/owl-features/.

25. Recchia G. A Comparison of String Measures for Data Analysis // G. Recchia, M. Louwerse // Proceedings of The First ACM SIGSPATIAL International Workshop on Computational Models of Place. - 2019. - P. 54-62.

26. Simpson K. You Don't Know JS [Електронний ресурс] / Kyle Simpson. - 2018. - Режим доступу до ресурсу: http s:// github .com/ getify/Y ou-Dont-Know-JS.

27. Shoham Y., Leyton-BrownK. Multiagent Systems: Algorithmic, Theoretic, and Logical Foundations. Cambridge University Press, 2018.

28. Wagner G. The Agent-Object-Relationship Metamodel: Towards a Unified View of Behavior Information Systems G. Wagner, 2017.

29. William, L. Follow the Rules Regarding Concurrency Management. BuildSecurityIn / L. William // In Net Kernel Developer’s Manual. - 2015.-pp. 45-48.

ДОДАТОК А

(Обов'язковий)

Копії публікацій

к.т.н., доц. Муляр І.В. *(ХмНУ),*
Віхтюк А.Р. *(ХмНУ),*
Пічура В.Ю. *(ХмНУ)*

АНАЛІЗ ПІДХОДІВ ДО ПРОЄКТУВАННЯ ДЕЦЕНТРАЛІЗОВАНИХ СИСТЕМ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЇ БЛОКЧЕЙН

Здобутки блокчейн базуються на руйнуванні організаційної парадигми систем централізованого зберігання і обробки даних.

Організація і функціонування централізованих програмних систем відрізняється простотою, їх розробка є формалізованою і базується на відомих методах та алгоритмах, тому вони є популярними. Прикладами таких систем є більшість веб-додатків. Під централізованою їх організацією мається на увазі, що вебдодаток має один або декілька серверів, які є центрами і серцем системи. На серверах запущено вебсервер додатку та виконується його програмний код. Всі взаємодії із системою є централізованими і орієнтованими на використання цих серверів. Стабільність такої системи безпосередньо залежить від провайдера серверів.

Розміщення всіх даних на локальних серверах і використання ресурсів одного провайдера зумовлюють ненадійність централізованих додатків і ризики втрати конфіденційності інформації [1].

Однією із основних характеристик, яка зумовила поширення блокчейн, є надійність. Надійність блокчейн забезпечується ланцюговим хешуванням інформаційних блоків, розподіленим зберіганням даних з багатократним їх резервуванням, використанням алгоритму цифрового підпису з асиметричним шифруванням на основі еліптичної кривої (ECDSA) для перевірки автентичності транзакцій.

Популярності технології блокчейн додає можливість збереження анонімності у виконанні операцій, відсутність потреб у послугах банківських та інших фінансових установ, висока продуктивність систем надання послуг.

Розвиток методів створення децентралізованих додатків призвів до розвитку альтернативних підходів до проектування програмних систем на блокчейн і до появи принципово нових програмних продуктів. Це було зумовлено децентралізованою природою нових програм та технологіями, які використовуються для зберігання даних. Такі функції, як розумний контакт проекту Ethereum, сприяли створенню великих децентратлізованих додатків [2]. Принцип децентралізації як основа технології блокчейн – це актуальна задача, вирішення якої робить можливою створення ефективних і надійних програмних додатків, але використання цієї можливості залежить від способу розробки децентралізованих додатків

ЛІТЕРАТУРА:

1. David López and Bilal Farooq, (2019). A multi-layered blockchain framework for smart mobility data-markets. Transportation Research Part C: Emerging Technologies. 111. 10.1016/j.trc.2020.01.002

2. Safe Decentralized Applications Development Using Blockchain Technologies / Viktor Cheshun, Ihor Muliar, Vasyl Yatskiv, Ruslan Shevchuk, Serhii Kulyna, Taras Tsavolyk // 2020 10th International Conference on Advanced Computer Information Technologies (ACIT), 16-18 Sept. 2020, Deggendorf, Germany. – Publisher: IEEE, 2020. – P. 800-805.

УДК 004.056:621.397.3:004.942

Володимир ДЖУЛІЙ
Хмельницький національний університет
ORCID http://orcid.org/0000-0003-1878-4301
e-mail:dg2303@ukr.net
Ігор МУЛЯР
Хмельницький національний університет
ORCID http://orcid.org/0000-0002-6659-605X
muliariv@khmnu.edu.ua
Орислава ЗАЦЕПІНА
Хмельницький національний університет
e-mail: orysiia@gmail.com
Вадим ПІЧУРА
Хмельницький національний університет
e-mail: vadimpichura007@gmail.com

ІНФОРМАЦІЙНО-ОЗНАКОВА МОДЕЛЬ ДЖЕРЕЛА ШКІДЛИВОЇ ІНФОРМАЦІЇ В СОЦІАЛЬНИХ МЕРЕЖАХ

Розглянута актуальна задача побудови інформаційно-ознакової моделі джерела шкідливої інформації в соціальних мережах. Інформаційно-ознакова модель шкідливої інформації в соціальних мережах, дозволяє сформувати дані для виявлення та протидії поширенню шкідливої інформації в мережі. Комплекс моделей складається з моделі шкідливої інформації, інформаційно-ознакової моделі шкідливої інформації, моделі джерела інформації, моделі соціальної мережі. Кожна з моделей містить унікальні атрибути та відношення між інформаційними об'єктами, також комплекс моделей дозволяє сформувати відповідні вимоги до алгоритмів оцінки та аналізу джерел повідомлень та забезпечує вибір контрзаходів.

Ключові слова: моделі, алгоритми, модель шкідливої інформації, соціальні мережі, контрзаходи, джерела повідомлень.

Volodymyr DZHULIY, Igor MULYAR,
Oryslava ZACEPINA, Vadym PICHURA
Khmelnytsky National University

AN INFORMATION-SIGN MODEL OF THE SOURCE OF HARMFUL INFORMATION IN SOCIAL NETWORKS

Abstract. The problem of detecting and countering the spread of harmful information has an insufficient number of scientific and technical solutions. The available means of combating and detecting harmful information in social networks do not meet the requirements for adequacy, speed, accuracy and completeness of the decisions made. This is due to the following reasons: the systems are divided into two unrelated modules - monitoring, countermeasures, between which the operator is located. It is necessary to process extremely large flows of messages in real time, implement countermeasures in a short period of time, in manual mode the operator is unable to stop the spread of malicious information in the social network.

The task of the research is to develop: models of harmful information, source and social network; algorithms for analyzing the sources of messages spreading harmful information in social networks and ranking countermeasures.

Solving the set tasks will allow: to improve the quality of decisions made in the process of detecting and countering harmful information; sort information objects of influence for the operator by priority; set the input data for the configuration of the system for detecting and countering the spread of malicious information in networks.

The information-sign model of malicious information in social networks allows you to generate data for detecting and countering the spread of malicious information in the network. The complex of models consists of a model of malicious information, an information-sign model of malicious information, a model of the source of information, a model of a social network. Each of the models contains unique attributes and relationships between information objects, as well as a set of models allows for the formation of appropriate requirements for algorithms for the evaluation and analysis of message sources and provides a choice of countermeasures.

Keywords: models, algorithms, malicious information model, social networks, countermeasures, message sources.

Постановка проблеми у загальному вигляді
та її зв'язок із важливими науковими чи практичними завданнями

На сучасному етапі, глибина проникнення у повсякденне життя соціальних мереж є значною. Перевагою соціальних мереж є можливість учасникам комунікації висловлювати оперативно свою думку значній кількості групі людей, публікувати відео-, медіа файли. Соціальні мережі є не лише засобом спілкування групи людей, а й також інструментом поширення інформації, в тому числі шкідливої інформації. Необхідно зазначити, що злочинні та терористичні угруповання беруть на озброєння, дедалі частіше, засоби інформаційного впливу, розробляють та пишуть стратегії, спрямовані на залучення нових адептів та розширення сфери впливу через соціальні мережі. Таким чином, однією зі складових надійного забезпечення інформаційної безпеки держави є виявлення, моніторинг, аналіз та активна протидія розповсюдженню шкідливої інформації в соціальних мережах [1-3].

Проблема виявлення та протидії поширенню шкідливої інформації має не достатню кількість науково-технічних рішень. Доступні засоби протидії та виявлення шкідливої інформації в соціальних мережах не відповідають вимогам до адекватності, швидкості, точності та повноти прийнятих рішень. Це обумовлено наступними причинами: системи розділені на два незв'язаних модулі – моніторинг, протидія, між якими знаходиться оператор. Соціальні мережі мають складну структуру, до складу яких входять різнорідні повідомлення, що недостатньо враховується під час реалізації мети протидії – джерело, тип повідомлення, та інші характеристики. Необхідно обробляти у реальному масштабі часу надвеликі потоки повідомлень, в стислий термін реалізувати контрзаходи, в ручному режимі оператор не в змозі зупинити поширення шкідливої інформації в соцмережі [2-6].

Постановка задачі

Протидія поширенню шкідливої інформації у соцмережах є важливим елементом інформаційної безпеки особистості, суспільства, держави, проте більшість систем, на теперішній час не враховують простір функціональності системи виявлення та протидії шкідливій інформації, необхідна автоматизація процесу протидії. Соціальні мережі мають складну структуру, параметри повідомлень та джерел не в повній мірі враховуються під час виборів мети виявлення та протидії шкідливій інформації. При розробці методу протидії поширенню шкідливої інформації необхідно: в повній мірі враховувати кількість повідомлень на сторінці, характеристики джерела, зворотній зв'язок від джерела та аудиторії; підтримувати дві стадії: експлуатація, налаштування; ранжувати контрзаходи з урахуванням коефіцієнтів складності [4,7,8].

Задача дослідження полягає у розробці: моделей шкідливої інформації, джерела та соціальної мережі; алгоритмів проведення аналізу джерел повідомлень поширення шкідливої інформації у соціальних мережах та проведення ранжування контрзаходів; методу виявлення та протидії поширенню шкідливої інформації у соціальних мережах з урахуванням вимог до обґрунтованості; архітектури компонентів системи протидії поширеню шкідливої інформації в соцмережах [8,9].

Вирішення поставлених задач дозволить: підвищити якість прийнятих рішень у процесі виявлення та протидії шкідливій інформації; сортувати інформаційні об'єкти впливу для оператора по пріоритету; задати вхідні дані налаштування системи виявлення та протидії поширенню шкідливої інформації в мережах.

Основна частина

Моделі даних соціальних мереж характеризуються, незалежно від їх структури, загальними атрибутами - джерела, повідомлення, ознаки зворотного зв'язку на повідомлення суб'єкта. Наявність ознак зворотного зв'язку в моделі соцмережі дозволяє характеризувати джерело повідомлення. Нехай $ACTIVITY$ $\{countLike, countREpost, countComment, countView\}$ множина у повідомленнях від реципієнтів всіх ознак зворотного зв'язку інформації у соцмережі, де $countLike$ – кількість позначок, $countREpost$ – кількість копій з посиланням на джерело («репостів»), $countComment$ - кількість коментарів, $countView$ – кількість переглядів.

Виходячи з поставлених задач, необхідно визначити атрибути множини $ACTIVITY$, а також відношення $R(SOURCE, MESSAGE)$, які в подальшому дозволять проводити аналіз повідомлення та джерела, що містять шкідливу інформацію та вибирати відповідний об'єкт для протидії. Наприклад, якщо сума елементів активності до повідомлення дає можливість обчислити індекс активності повідомлення, таким чином може бути отриманий, в даній ситуації, інтегральний показник індексу активності, який в свою чергу залежить від кількості повідомлень джерела, очевидно одним із атрибутів моделі даних джерела буде *index_active*. Якщо кількість переглядів повідомлення дозволяє обчислити індекс перегляду, таким чином можемо отримати інтегральний показник індексу перегляду для джерела інформації, отримаємо наступний атрибут моделі даних джерела - *index_viewability*. Функція $f : MESSAGE \to SOURCE$ задає область визначення, вхідні та вихідні значення (аргументи). Функція сюр'єктивна - є відображенням множини $MESSAGE$ на множину $SOURCE$, при відображенні кожен елемент множини $SOURCE$ є образом множини $MESSAGE$ (хача б одного елемента). Таким чином, отримаємо:

$$\forall source \in SOURCE \exists message \in MESSAGE : source = f(message) \quad (1)$$

Повідомлення (аргументи) на стіні джерела можуть бути різного типу (відповідь, пост, коментар). Таким чином, для окремих аргументів (повідомлень) може бути заданий числовий коефіцієнт (рейтинг) у дереві повідомлень соціальної мережі (рис. 1).

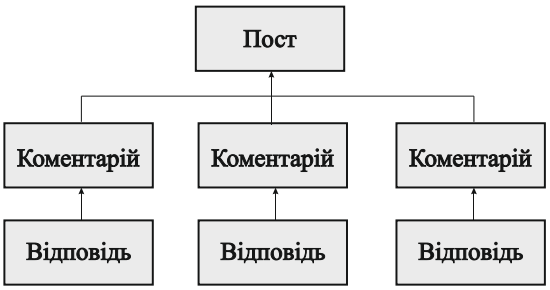


Рис. 1. Дерево повідомлень соцмережі

В залежності від кількості аргументів, джерело повідомлення можливо оцінити за потенціалом (potential): джерело із низьким потенціалом; джерело із середнім потенціалом; джерело із високим потенціалом. Якщо джерело повідомлень має атрибути: index_active, index_viewability, то можна задати індекс впливу - index_impact, який відображає рівень впливу джерела повідомлення на аудиторію.

Виділимо атрибути в кортежі, що характеризують $SOURCE$ через елементи множини $ACTIVITY$ і відношення $R(SOURCE, MESSAGE)$ - $\langle index_{active}, index_{viewabiliy}, potential, index_{impact} \rangle$. Також, атрибутами моделі джерел повідомлень є: *social_network_type* – тип даних структури соцмереж; *followers* – кількість пов'язаних користувачів; *registration_time* - час реєстрації в мережі джерела інформації. Модель даних джерела повідомлень відрізняється наявністю нових атрибутів, класів, відношень.

Розглянемо модель шкідливої інформації в мережі Інтернет. Основою для формування поняття - шкідлива інформація виступають два терміни [10]: *I* – information (Інформація); *IO* – information object (Інформаційний об'єкт) –логічно цільний блок відповідної інформації, представлений у фіксованої формі, використовується та створений в ході інформаційної діяльності. Формально терміни пов'язані між собою, так, що $IO \subseteq I$ (рис. 2. а) - інформаційний об'єкт є елементом множини всієї інформації, над якою проводиться аналіз. Із терміном «інформація» також пов'язаний термін - *IA* – information area («інформаційний простір»), а множини I, IO є підмножинами інформаційного простору. Соціальні мережі представляють собою сукупність взаємозалежних вузлів: спільноти, акаунти, сторінки, вкладення, пости; зв'язки між об'єктами – однорівневі відношення (перебувають у співтоваристві, у друзях); відношення вкладеності (сторінка запису містить посилання на пост, стіна містить пост) [6-8]. Соціальні мережі можуть бути представленні графами: частина об'єктів - інформаційні, вершина графа, а зв'язки між об'єктами - ребра між вершинами. Таким чином, справедливо, що $IO \subseteq I \subseteq IA$, $SN \subseteq I \subseteq IA$, область перетину між *SN* (соцмережа) та *IO* є предметом дослідження у соціальних мережах розробки моделі шкідливої інформації (рис. 2. б).

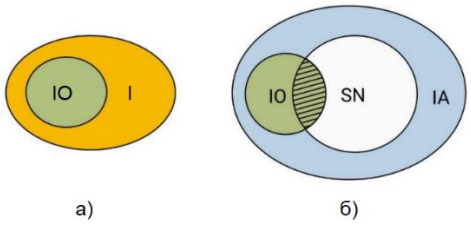


Рис. 2. Графічне представлення відношень множини інформаційного простору

Інформаційний об'єкт - *MIO* (шкідливий інформаційний об'єкт), містить відповідні ознаки, які дозволяють прийняти рішення, що інформація завдає шкоди державі, суспільству, бізнесу особистості. В залежності від умови експерт встановлює ознаки *Token* інформаційної загрози T. Наприклад, батько сам вибирає обмеження для дитини, у випадку використання система батьківського контролю. Якщо представник бізнес - компанії зацікавлений у захисті конфіденційного інформації бізнеса, в даній ситуації він їх сам задає. Таким чином, у соціальній мережі, теоретико-множинна модель шкідливої інформації, включає наступні базові елементи: *IO* - інформаційний об'єкт (Information Object); T - інформаційна загроза (Threat); *MIO* – шкідливий інформаційний об'єкт; *Token* – ознака інформаційної загрози, що знаходиться у шкідливому інформаційному об'єкті; *Feature* – ознака наявності інформаційного об'єкта [0,1]; зв'язок між інформаційними об'єктами.

Теоретико-множинна модель шкідливої інформації (2) формально представлена наступним чином:

$$\begin{aligned}
&IO = \{io\}; MIO = \{io\}; MIO_i = \{io\} \\
&MIO \subset IO; \forall io \in MIO : io \in IO \\
&MIO_i \subseteq MIO; \forall io \in MIO_i : io \in MIO \\
&Token_{mio_i} \subset T; Token_{mio_i} = \{t\} \\
&CheckFeature(io,t) = \{True; False\} \\
&io \in MIO_i \Leftrightarrow \exists Token_{mio_i} : checkFeature(io,t) = True
\end{aligned} , \quad (2)$$

де *IO* – множина інформаційних об'єктів, *io* – інформаційний об'єкт, T - множина ознак інформаційної загрози, t_i –*i -a* ознака інформаційної загрози, *MIO* – множина шкідливих інформаційних об'єктів мережі, MIO_i - *i-й* клас шкідливої інформації, $Token_{mio_i}$ - множина ознак загрози, що характеризують *MIO*.

Таким чином, для виявлення та протидії поширенню шкідливій інформації в мережі необхідно задати набір ознак, характерних для інформаційної загрози в соціальній мережі.

Особливістю моделі шкідливої інформації соціальної мережі є те, що модель допускає наявність дискретних ознак у множині ознак: зв'язок інформаційного об'єкта з іншими інформаційними об'єктами у соцмережі; частота повторення ознаки; дата створення інформаційного об'єкта.

Протидія поширенню шкідливого інформаційного об'єкта в соціальній мережі може здійснюватися лише на рівні джерел чи повідомлень. Таким чином, необхідно виділити такі інформаційні загрози та відповідні інформаційні ознаки повідомлення у соцмережі, що характеризують його як шкідливий об'єкт. Інформаційно-ознакова модель (табл.1) - впорядкована сукупність інформації про зв'язки повідомлень та їх ознак зі змістом повідомлень. Інформаційні ознаки повідомлень - окремі властивості повідомлень, їх зміст. Інформаційна загроза – задається оператором системи. Шкідлива інформація в соціальній мережі – задається оператором шляхом формування набору відповідних ключових слів. Інформаційні ознаки - формують множину усіх можливих інформаційних ознак *t*. На рис.3 наведено співвідношення, взаємозв'язок різних рівнів інформаційно-ознакової моделі шкідливої інформації Показано, що формується повідомлення, розміщується повідомлення в джерелі розповсюдження, на сторінці групи, облікового запису. Повідомлення можуть містити ознаки шкідливої, а також не містити ознаки шкідливої інформації. Інформаційні ознаки (табл. 1) формують рівень інформаційних загроз в соціальній мережі.

Таблиця 1

Інформаційно-ознакова модель шкідливої інформаціїсоціальної мережі

Інформаційні загрози	Шкідлива інформація у соцмережах	Інформаційні ознаки
	Приклад Повідомлення, що містить прохання, пропозицію, наказ зробити самогубство, описує самогубство як спосіб вирішення проблем	t_1
	Приклад Повідомлення, що містить позитивну оцінку схвалення вчинення самогубства, дій, спрямованих на самогубство	t_2

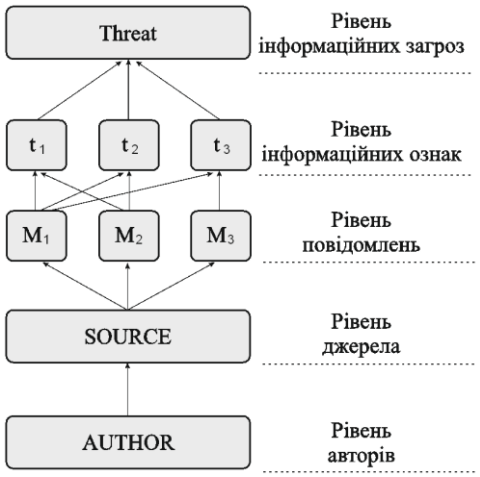


Рисунок 3. Інформаційно-ознакова модель шкідливої інформації

Таким чином, зібравши відповідну інформацію на сторінці джерела можливо визначити, які з цих повідомлень інформаційної мережі належать до шкідливих повідомлень. Результатом виявлених загроз та їх кількості буде прийняте відповідне рішення про протидію джерелу, повідомленню.

Запропонована інформаційно-ознакова модель шкідливої інформації в соціальних мережах, дозволяє сформувати дані для виявлення та протидії поширенню шкідливої інформації в мережі. Комплекс моделей складається з моделі шкідливої інформації, інформаційно-ознакової моделі шкідливої інформації, моделі джерела інформації, моделі соціальної мережі. Кожна з моделей містить унікальні атрибути та відношення між

інформаційними об’єктами, також комплекс моделей дозволяє сформувати відповідні вимоги до алгоритмів оцінки та аналізу джерел повідомлень та забезпечує вибір контрзаходів.

Висновки з даного дослідження і перспективи подальших розвідок у даному напрямі

Запропонована модель соціальної мережі, що включає джерела, повідомлення, зв'язки (відношення) між інформаційними об’єктами, відрізняється наявністю нових рв'язків та структурних елементів. Розроблено модель джерела, в якій враховуються наступні параметри: індекс впливу, індекс активності, індекс перегляду, потенціал. Запропонована теоретико-множинна модель шкідливої інформації в соціальній мережі, складається з ознак шкідливої інформації та взаємопов'язаних об'єктів, що в сукупності формують шкідливо-інформаційні об'єкти в мережі Інтернет. Також розроблена інформаційно-ознакова модель шкідливої інформації в соціальних мережах, дозволяє сформувати дані для виявлення та протидії поширенню шкідливої інформації в соціальних мережах.

Літертура

1. Ленков, С.В. Модель безпеки поширення забороненої інформації в інформаційно-телекомунікаційних мережах / С.В. Ленков, В.М. Джулій, В.С. Орленко, О.В. Сєлюков, А.В. Атаманюк // Збірник наукових праць Військового інституту Київського національного університету імені Тараса Шевченка. – К.: ВІКНУ, 2020. – Вип. №68. – С. 53-64.

2. Соціальні мережі – реальні загрози віртуального світу. [Електронний ресурс]. – Режим доступу : http://ogo.ua/ articles/view/011–02–23/26490.htm.

3. Ленков, С.В. Методы и средства защиты информации. В 2-х томах /С.В. Ленков, Д.А. Перегудов, В.А. Хорошко –К: Арий, 2008.–464с

4. Остапов С. Е. Технології захисту інформації: навчальний посібник / С.Е. Остапов, С.П. Євсеєв, О.Г. Король– Харків : Вид-во ХНЕУ, 2016. – 476 с.

5. Лєнков, С.В. Аналіз існуючих методів та алгоритмів виявлення атак в бездротових мережах передачі даних / С.В. Лєнков, В.М. Джулій, Н.М. Берназ, С.О. Божук // Збірник наукових праць Військового інституту Київського національного університету імені Тараса Шевченка. – К.: ВІКНУ, 2017. – Вип. № 56. – С.124-132

6. Довгий, С.О. Сучасні телекомунікації: мережі, технології, економіка, управління, регулювання / С.О. Довгий, О.Я. Савченко, П.П. Воробієнко – К.: Український Видатничій Центр, 2012. – 520 с.

7. Джулій, В.М. Модель оцінки ймовірнісно-часових характеристик інформаційної взаємодії в мережі інтернет речей / В.М. Джулій, І.В. Муляр, О.В. Сєлюков, Б.М. Кізюн // Збірник наукових праць Військового інституту Київського національного університету імені Тараса Шевченка. – К.: ВІКНУ, 2019. – Вип. № 63. – С.96-106

8. Джулій, В.М., Кльоц Ю.П., Муляр І.В., Жилевич М.Л., Джулій А.В. Контроль додатків інтернет-трафіка комп’ютерних мереж методами машинного навчання. Вісник Хмельницького національного університету. Технічні науки. 2021. № 5. С. 22-26.

9. Джулій, В.М. Метод класифікації додатків трафіка комп'ютерних мереж на основі машинного навчання в умовах невизначеності / В.М. Джулій, О.В. Мірошніченко, Л.В. Солодєєва // Збірник наукових праць Військового інституту Київського національного університету імені Тараса Шевченка. – К.: ВІКНУ, 2022. – Вип. №74. – С. 73-82.

10. Лавров, Є. А. Математичні методи дослідження операцій : підручник / Є. А. Лавров, Л. П. Перхун, В. В. Шендрик – Суми : Сумський державний університет, 2017. – 212 с.

References

1. Lenkov, S.V. (2020), Model bezpeky poshyrennia zaboronenoi informatsii v informatsiino-telekomunikatsiinykh merezhakh / S.V. Lenkov, V.M. Dzhulii, V.S. ORLENKO, O.V. Sieliukov, A.V. Atamaniuk // Zbirnyk naukovykh prats Viiskovoho instytutu Kyivskoho natsionalnoho universytetu imeni Tarasa Shevchenka. – K.: VIKNU. – №68. – pp. 53-64.

2. Cotsialni merezhi – realni zahrozy virtualnoho svitu. [Elektronnyi resurs]. – Rezhym dostupu : http://ogo.ua/ articles/view/011–02–23/26490.htm

3. Lenkov, S.V. (2008), Metodyy sredstva zashchytы ynformatsyy. V 2-kh tomakh / S.V. Lenkov, D.A. Perehudov, V.A. Khoroshko –K: Aryi–464s.

4. Ostapov, S. E. (2016) Tekhnolohii zakhystu informatsii: navchalnyi posibnyk / S.E. Ostapov, S.P. Yevseiev, O.H. Korol– Kharkiv : Vyd-vo KhNEU. – 476 s.

5. Lenkov, S.V. (2017), AnalIz Isnuyuchih metodiv ta algoritmiv viyavlennya atak v bezdrotovih merezhah peredachI danih / S.V. Lenkov, V.M. Dzhuliy, N.M. Bernaz, S.O. Bozhuk // Zbirnik naukovih prats Viyskovogo Institutu Kiyivskogo natsionalnogo universitetu imeni Tarasa Shevchenka. – K.: VIKNU. – Vip. No 56. – p.124-132

6. Dovhyi, S.O. (2012), Suchasni telekomunikatsii: merezhi, tekhnolohii, ekonomika, upravlinnia, rehuliuvannia /S.O. Dovhyi, O.I. Savchenko, P.P. Vorobiienko – K.: Ukrainskyi Vydatnychii Tsentr. – 520p.

7. Dzhulii, V.M. (2019), Model otsinky ymovirnisno-chasovykh kharakterystyk informatsiinoi vzaiemodii v merezhi internet rechei / V.M. Dzhulii, I.V. Muliar, O.V. Sieliukov, B.M. Kiziun // Zbirnyk naukovykh prats Viiskovoho instytutu Kyivskoho natsionalnoho universytetu imeni Tarasa Shevchenka. – K.: VIKNU. – Vyp. № 63. – p.96-106

8. Dzhulii V.M., Klots Yu.P., Muliar I.V., Zhylevych M.L., Dzhulii A.V. (2021), Kontrol dodatkiv internet-trafika kompiuternykh merezh metodamy mashynnoho navchannia. Visnyk Khmelnytskoho natsionalnoho universytetu. Tekhnichni nauky. – Khmelnytskyi. – №5. – pp. 22–26.

9. Dzhulii, V.M. (2022), Metod klasyfikatsii dodatkiv trafika kompiuternykh merezh na osnovi mashynnoho navchannia v umovakh nevyznachenosti / V.M. Dzhulii, O.V. Miroshnichenko, L.V. Solodieieva // Zbirnyk naukovykh prats Viiskovoho instytutu Kyivskoho natsionalnoho universytetu imeni Tarasa Shevchenka. – K.: VIKNU. – Vyp. №74. – pp. 73-82.

10. Lavrov, Ye. A. (2017.), Matematychni metody doslidzhennia operatsii : pidruchnyk / Ye. A. Lavrov, L. P. Perkhun, V. V. Shendryk – Sumy : Sumskyi derzhavnyi universytet, – 212 p.

ДОДАТОК Б

(Обов’язковий)

Презентація

Тема: Метод побудови композитних вебдодатків на основі інтеграції слабоструктурованих даних

Мета магістерської роботи полягає в побудові побудові композитних вебдодатків шляхом вдосконалення інформаційної технології адаптивної інтеграції слабоструктурованої інформації у вебсистемах

Об'єкт дослідження: процес композитної збірки вебзастосунків на основі інтеграції слабоструктурованих даних.

Предмет дослідження: методи агрегації та впорядкування наборів слабоструктурованих даних

Задачі досліджень у роботі формулюються наступним чином:

- проаналізувати Semantic Web концепцію при побудові вебзастосунків, її переваги та недоліки;
- вдосконалити модель стандартизованого профілю системи збору інформації;
- розробити алгоритм побудови онтологічної моделі композитного вебзастосунку
- розробити семантичну модель профілю системи збору інформації;
- вдосконалати метод композитної збірки вебдодатків;
- вдосконалати структурну модель системи композитної збірки, яка використовує технологію Mashup;
- дослідити пошук та вибір сервісів для компонування і реалізації розроблених алгоритмів композитної збірки вебдодатку.

Наукова новизна роботи

- вдосконалено структурну модель стандартизованого профілю системи збору інформації на основі технології Mashup, що відрізняється від відомих доданням функціональних компонентів для визначення та опису семантичних процесів вхідних та вихідних наборів даних, що дає можливість підвищити якість результатів використання таких систем.
- вдосконалено метод композитної збірки вебдодатків, що відрізняється можливістю отримувати набори компонент по їх частковому опису.

Практична цінність. В роботі розроблено інформаційну технологію та програмні застосунки, які дають можливість зменшення інформаційного шуму до 7% та покращення узгодженості результатів у системі композитного формування вебзастосунку за технологією Mashup.

Апробація роботи. Наукові результати і основні положення магістерської роботи доповідались і обговорювались на всеукраїнських та міжнародних науково-технічних конференціях,

Публікації. По темі магістерської роботи опубліковано 1 наукові статті, 1 - теза доповідей на всеукраїнських конференціях.

Класифікація рівнів інтеграції даних

Порівняльна характеристика традиційного та семантичного підходів до інтеграції даних

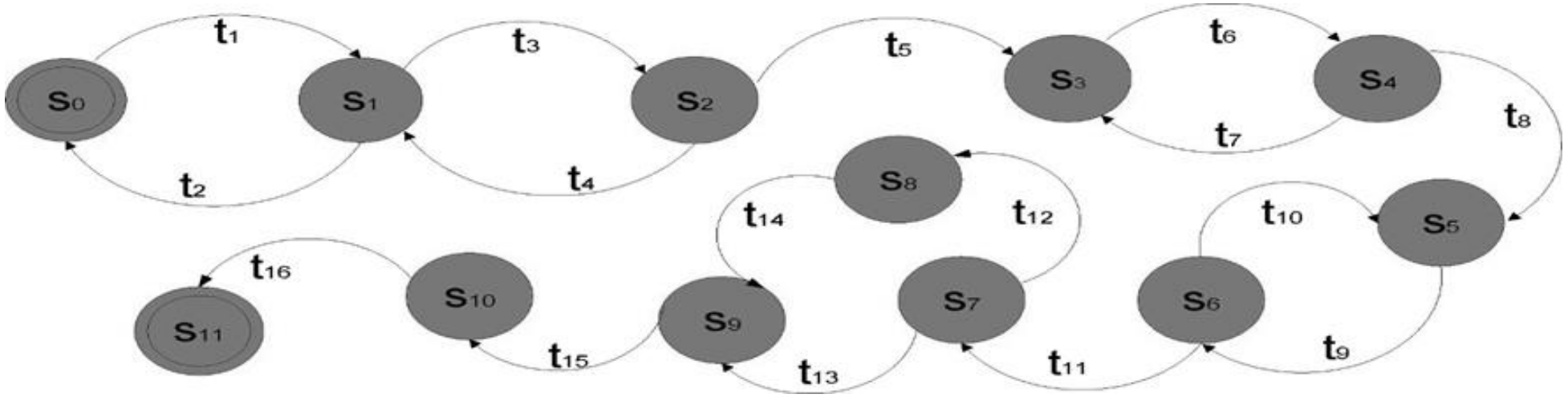
	Традиційна інтеграція	**Семантична інтеграція**
Структура даних	Переважно реляційна, орієнтована на однакові набори даних.	Орієнтована на відношення між одиницями даних, не залежно від їх подібності.
Метод інтеграції	Дані витягуються із джерел, перетворюються, відповідно до заданих вимог і завантажуються 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.

Стандартизований профіль системи збору інформації можна

$$H = \langle DI, Q, DO, C, T, \lambda, \beta \rangle$$

де: DI – набір вхідних даних; Q – множина запитів користувачів; DO – глобальний набір вхідних даних; C – множина умов інтеграції; T – час транзакцій на оновлення даних; λ - оператор визначення вхідних даних; β - оператор формування вихідних даних.

Діаграма переходів станів системи



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);

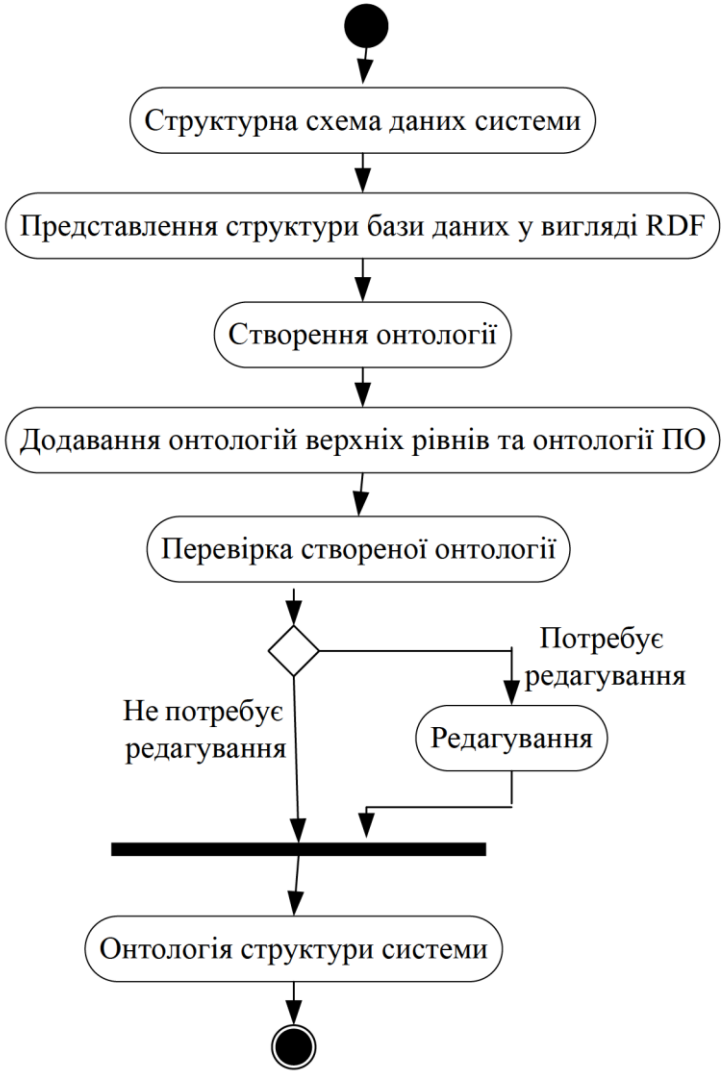
s_8 - стан редагування пошукових критеріїв (Search with editing);

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);

Алгоритм побудови онтологічної моделі систем, що інтегруються



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

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.

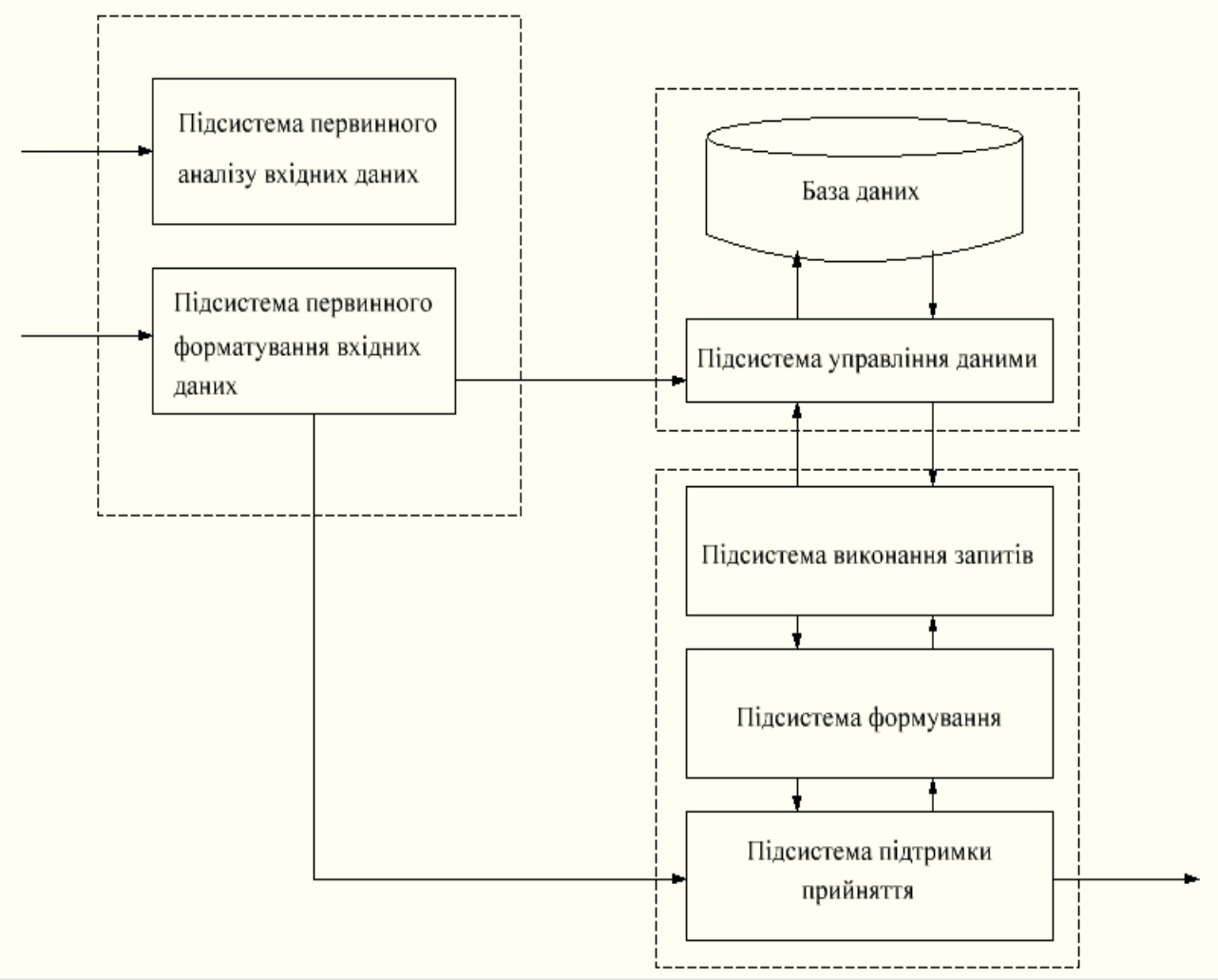
3 Додавання онтологій верхніх рівнів та онтології предметної області. Реалізуємо даний крок за допомогою мови OWL, використовуючи команду owl : *import.* Завдяки правилу транзитивності в RDF, додаткові онтології розширюють предметні області і додають нові концепти і властивості.

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.

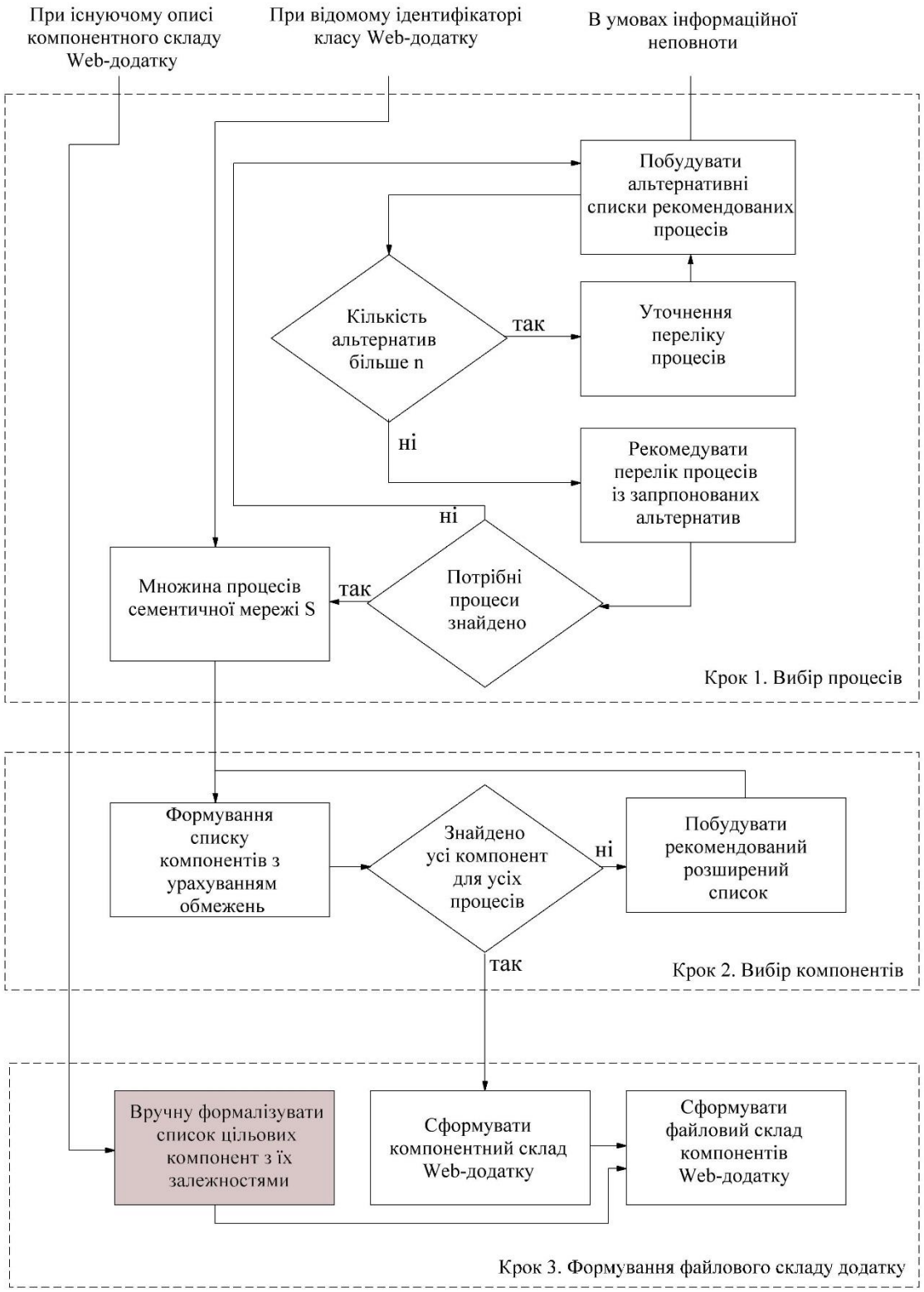
5 Редагування витягнутої онтології і додавання зв'язків між концептами. Далі повертаємося до кроку 4.

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

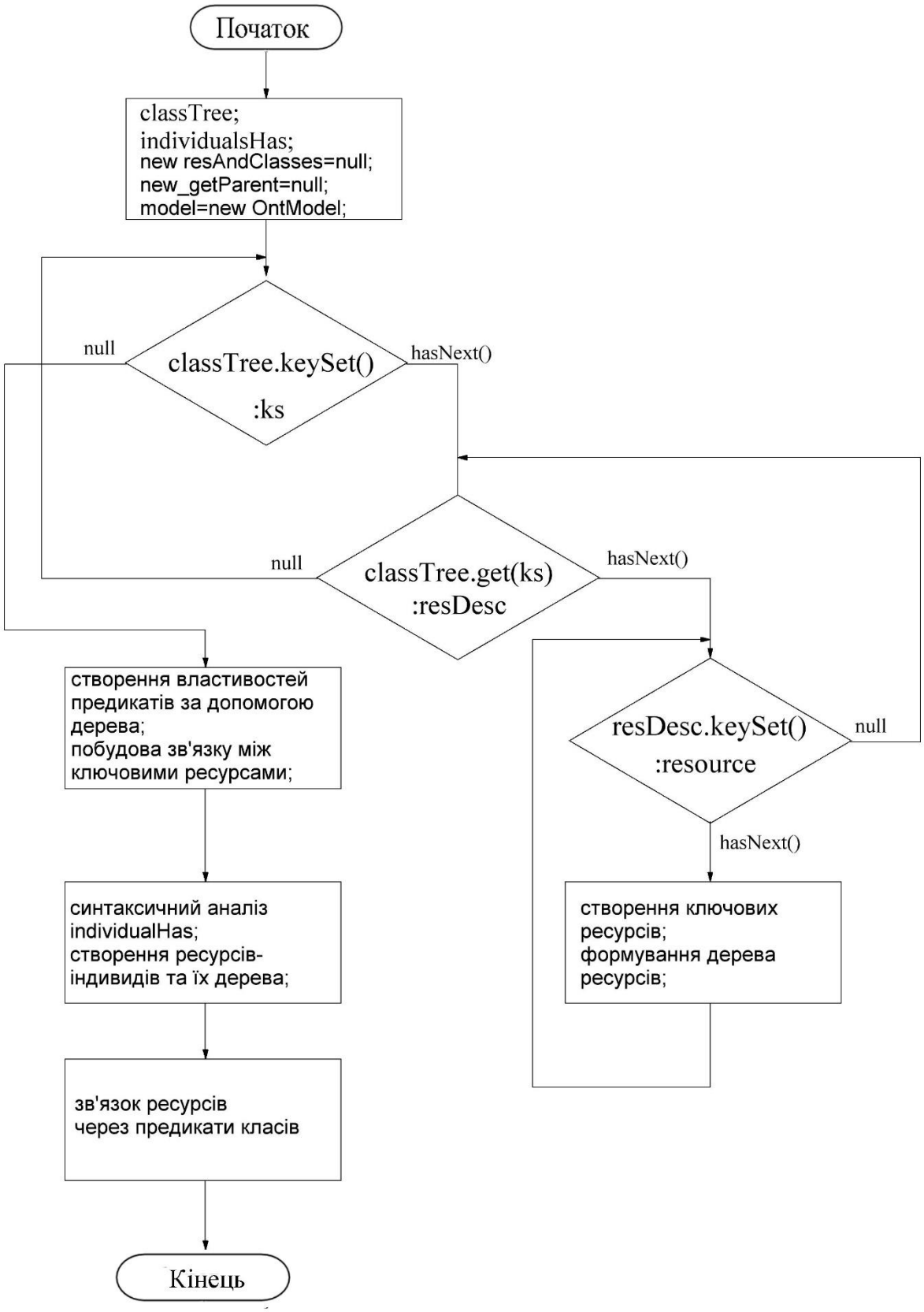
Схема системи композитної збірки вебдодатків



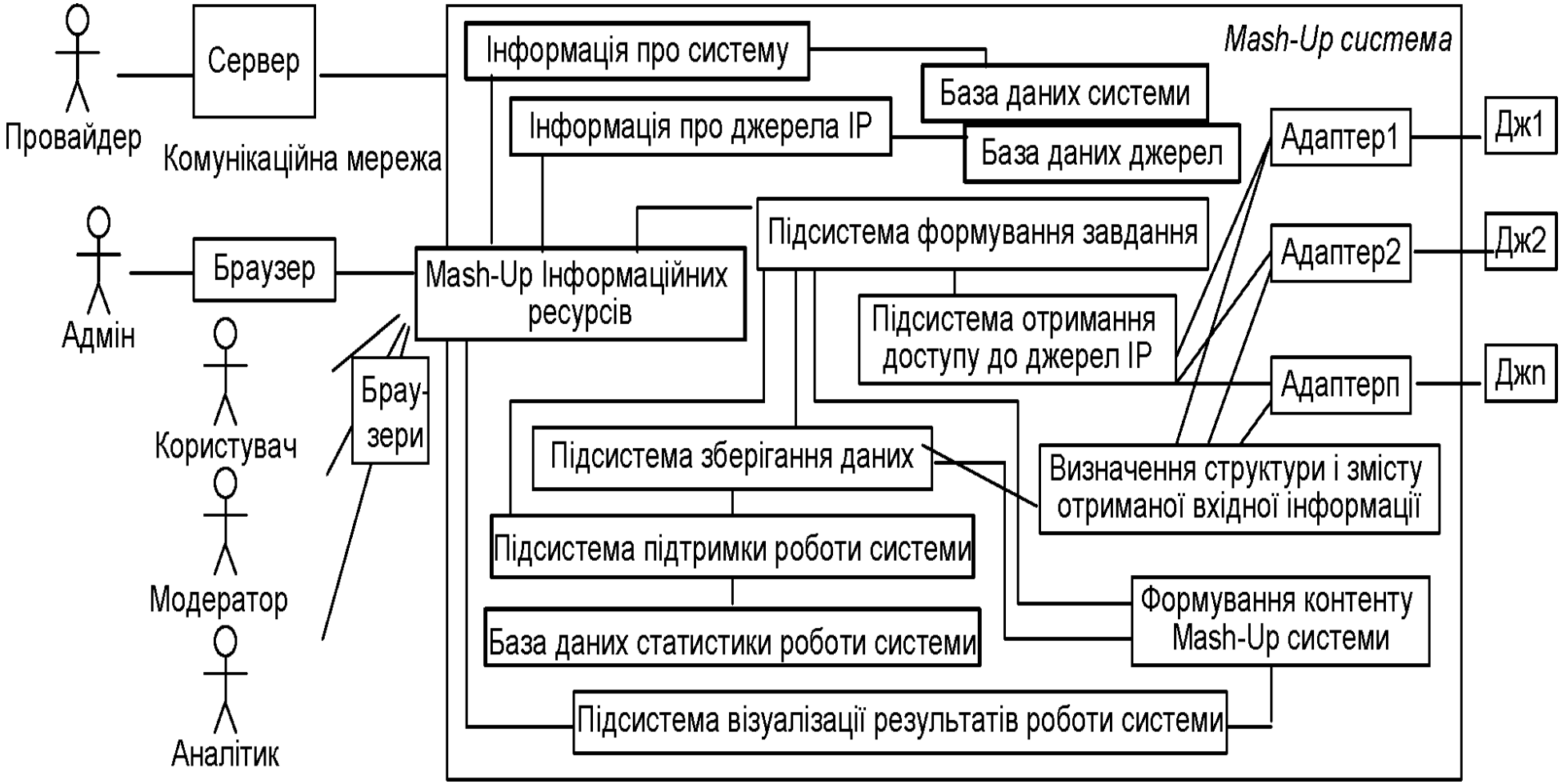
Вдосконалений метод композитної збірки вебдодатків



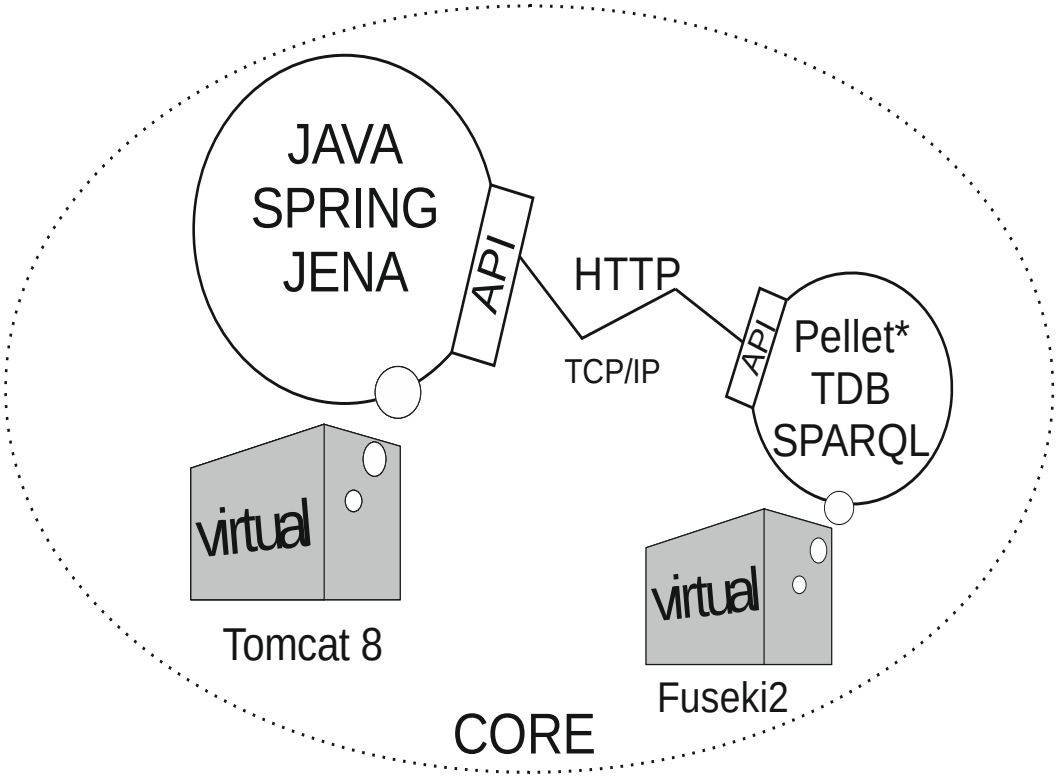
Блок-схема алгоритму перетворення інформації із загальної моделі в онтології



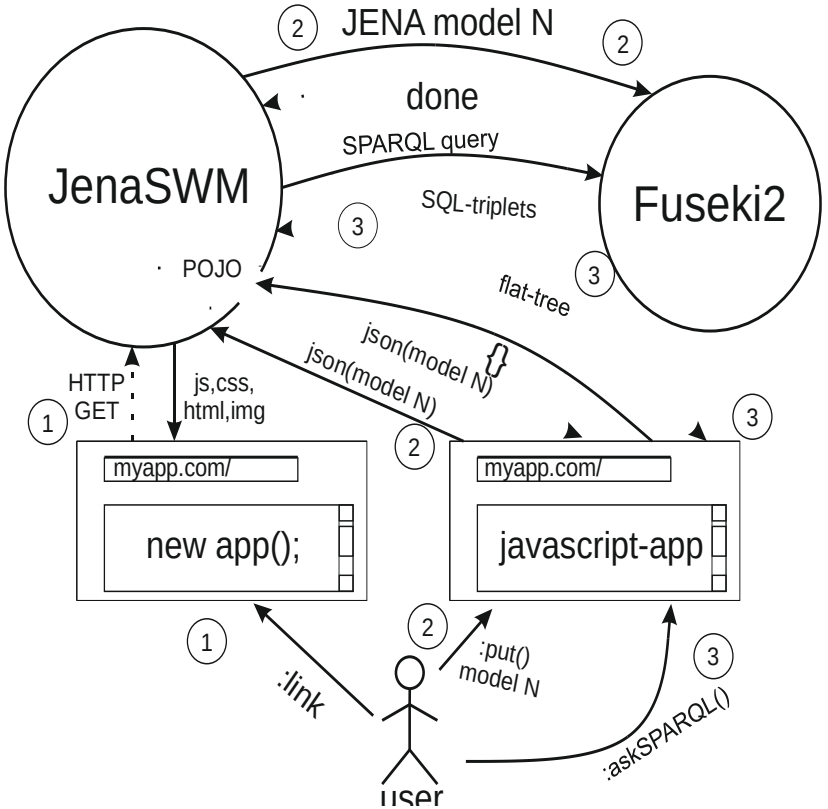
Структурна модель системи динамічної інтеграції слабоструктурованих даних, що працює на основі Mashup



Архітектура системи динамічної інтеграції



Рівень ядра



Рівень базових операцій

Висновки

Результати проведеного дослідження узагальнені в наступних пунктах:

1. Проведено аналітичний огляд вже існуючих рішень для вебсистем, проаналізованно Semantic Web концепцію при побудові вебзастосунків, її переваги та недоліки.

2. Було формалізовано семантичну модель стандартизованого профілю системи, а також формалізовану модель онтологічної бази знань на її основі. Запропонована модель включає 3 рівні: рівень специфікації, рівень домену, рівень інформаційної системи. Модель визначає загальну структуру, в межах якої можуть бути побудовані прикладні семантичні моделі, які застосовуються безпосередньо на практиці.

3. Запропоновано підхід до підвищення якості результату процесу визначення вхідних даних системи збору інформації шляхом застосування процедури опису структури та змісту вхідних інформаційних ресурсів, що дає змогу покращити якість процесу інтеграції даних з різних джерел для подальшого їх використання в системі збірки вебзастосунку. Для цього була розроблена модель процесу композитної збірки вебдодатків у вигляді семантичної мережі з використанням продукційних правил.

4. Розроблено алгоритм побудови онтологічної моделі композитного вебзастосунку, що дозволило вдосконалити процес автоматичного створення загальної динамічної структури вхідних інформаційних ресурсів з можливістю урахуванням їх змісту.

5. Вдосконалено метод композитної збірки вебдодатків, який включає три етапи: вибір процесу; підбір компонентів; формування складу композитного вебзастосунку. Для організації механізмів збору інформації використовувалася схема агентів керування запитами до бази знань, на основі якого будується автоматизована система композитної збірки вебдодатків. Для цього також запропоновано алгоритм перетворення інформації про процеси та компоненти вебзастосунку із загальної моделі, у модель бази знань онтологічного типу.

6. В роботі розроблено інформаційну технологію та програмні застосунки, які дають можливість зменшення інформаційного шуму до 7-8% та покращення узгодженості результатів у системі композитного формування вебзастосунку за технологією Mashup.

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ

КАФЕДРИ КІБЕРБЕЗПЕКИ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Метод побудови композитних вебдодатків на основі інтеграції слабоструктурованих даних

Автор: Пічура Вадим Юрійович
Спеціальність: 123 – Комп'ютерна інженерія
Освітня програма: Програмування та захист комп'ютерних систем і мереж
Науковий керівник: Тітова В.Ю., к.т.н, доц.
Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних). Робота приймається до захисту.	відповідає
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи.	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
5	Інше:	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) запозичення розміщені в розділах аналізу існуючих аналогів та прототипів, які не описують безпосередньо авторське дослідження і не стосуються результатів роботи;
2) усі запозичення фрагментарні, або мають належним чином оформленні посилання;
3) всі зафіксовані системою ознаки модифікації тексту відносяться до комбінування латинських символів зі україномовними скороченнями індексів в формулах, що не є модифікацією тексту.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості, складає 4.57% і адресується до 64 першоджерела, що, з урахуванням наведених обгрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ Віра ТІТОВА

Завідувач кафедри кібербезпеки _____ Юрій КЛЬОЦ

Дата: 06.12.2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

освітнього ступеня «магістр»

Магістр Пічура В.Ю.

Тема: Метод побудови композитних вебдодатків на основі інтеграції слабоструктурованих даних

Галузь знань 12 – Інформаційні технології

Спеціальність 123 –Комп'ютерна інженерія

Обсяг кваліфікаційної роботи освітньо-кваліфікаційного рівня «магістр»:

кількість листів креслень 8; кількість сторінок записки 78

1. В рамках кваліфікаційної роботи створено семантичну модель процесів і модулів вебдодатку і заснованих на ній метод, реалізований у вигляді інформаційно-пошукових алгоритмів компонентної збірки, що дозволяють зменшити часові витрати і скоротити число помилок на етапі реалізації

2. Висновок про відповідність кваліфікаційної роботи завданню Кваліфікаційна магістерська робота у повній мірі відповідає поставленому завданню як в теоретичній, так і в практичній частині.

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У вступі висвітлюється актуальність теми роботи, дається аналіз досліджуваної проблеми і обґрунтовується застосовуваний підхід до її вирішення, формулюються цілі і завдання дослідження, описується наукова новизна і практична значимість отриманих результатів. У першому розділі розглядаються питання забезпечення ефективного функціонування композитних вебдодатків. Наступні розділи присвячені розробці семантичної моделі процесів і модулів вебдодатку, та його структурному синтезу Розглянуто питання застосування розробленого методу.

4. Позитивні сторони роботи полягають в розробці методу формування об'єднаного структурованого динамічного набору вихідних даних, який допоміг значно знизити часові та ресурсові затрати під час виконання збірки вебдодатків

5. Негативні сторони роботи Використання розробленого методу потребує поглиблених знань розробників в мові опису онтологій OWL.

6. Оцінка графічного оформлення та пояснювальної записки роботи Графічне оформлення виконане відповідно до теми кваліфікаційної роботи з дотриманням стандартів. В загальному графічне оформлення виконане якісно. Пояснювальна записка відповідає нормам для її оформлення.

7. Відгук про роботу в цілому В загальному кваліфікаційної роботи заслуговує позитивної оцінки. Весь матеріал дипломної роботи структурований, чіткий та послідовний. Усі розділи роботи послідовні та логічні, що дозволяє чітко розуміти викладений матеріал в рамках тематики кваліфікаційної роботи. Графічний матеріал дозволяє наочно побачити доцільність та ефективність рішень, які були прийняті за основу для досягнення поставленої задачі.

8. Інші зауваження

9. Оцінка кваліфікаційної роботи Розглянувши позитивні та негативні сторони представленої кваліфікаційної роботи, можна зробити висновок, що вона заслуговує оцінку «добре»

РЕЦЕНЗЕНТ (прізвище, ім'я, по батькові, посада, місце роботи)
Мартинюк Валерій Володимирович, зав каф. ЕКІТ д т н професор

« 2 » 12 2022. (підпис)

Завідувачу кафедри кібербезпеки
к.т.н., доц. Кльопу Ю.П.
Пічури Вадима Юрійовича
ПІБ здобувача вищої освіти
студента ФІТ, 2 курсу, групи КІм-21-1

ЗАЯВА

З правилами чинного Положення «Про дотримання академічної доброчесності в Хмельницькому національному університеті» від 26.09.2020 (зі змінами від 26.11.2020), згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений (а). Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщений (а) та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надасться в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

5.12.2022
дата

підпис

UNICHECK

Ім'я користувача:
Кафедра кібербезпеки

ID перевірки:
1013209785

Дата перевірки:
06.12.2022 12:35:06 EET

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
06.12.2022 14:21:15 EET

ID користувача:
100008300

Назва документа: **Магістерська_Пічура**

Кількість сторінок: **77** Кількість слів: **13167** Кількість символів: **102489** Розмір файлу: **2.54 MB** ID файлу: **1012971520**

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

4.57%
Схожість

Найбільша схожість: **2.21%** з Інтернет-джерелом (https://lpnu.ua/sites/default/files/2020/dissertation/1655/diserkushnir.

3.32% Джерела з Інтернету	64	Сторінка 79
2.23% Джерела з Бібліотеки	61	Сторінка 79

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи	50
Підозріле форматування	17 сторінок

Anti-Plagiarism v-15.257

Максимальное совпадение с одним документом 0.0%

Словари проверки: en_US, ru_RU, ua_UA. **Ошибок в документах: 11%**

	Документ		Суммарное совпадение по Базе Данных	
ID: 109021 Название: Метод побудови композитних вебдодатків на основі інтеграції слабоструктурованих даних Добавлено в БД: 2022-12-06 Авторы: Пічура В.Ю. Руководители: Тітова В.Ю. Консультанты: Опоненты:	Символы	Лексемы	Символы	Лексемы
	83366	1259	1020 (1%)	20 (2%)

Источник плагиата

ID	Описание	Наличие плагиата в документе	
		Символы	Лексемы