МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет радіоелектроніки
Факультет Комп'ютерних наук
(або Центр післядипломної освіти,
або Навчально-науковий центр заочної форми навчання)
Кафедра Програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

Дослідження методів менеджменту і інтеграції різнорідної інформації для моделей знань

Виконав:
студент 2 курсу групи ІПЗздм-21-2
Кириченко Р.П.
(прізвище, ініціали)

Спеціальність	121 – Інженерія програмного забезпечення
Тип програми	Освітньо-наукова
Керівник	проф. Шостак І.В.

(посада, прізвище, ініціали)

Допускається до захисту
Зав. Кафедри _____ З.В. Дудар

2023 р.
Харківський національний університет радіоелектроніки

Факультет Центр післядипломної освіти

Кафедра Програмної інженерії

Рівень вищої освіти другий (магістерський)

Спеціальність 121– Інженерія програмного забезпечення
(код і повна назва)

Тип програми освітньо-наукова програма

Освітньо-наукова програма Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

«___»_____ 202_ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студента Кириченко Романа Павловича
(прізвище, ім’я, по батькові)

1. Тема роботи: Дослідження методів менеджменту і інтеграції різнорідної інформації для моделей знань

затверджена наказом університету від «03» квітня 2023 р. № 83 Стз

2. Термін подання студентом роботи до екзаменаційної комісії «20» травня 2023 р.

3. Вихідні дані до роботи: онтологія, цифровий документообіг, об’єктно-орієнтовне програмування, пояснювальна записка

4. Перелік питань, що потрібно опрацювати в роботі: мета роботи, аналіз предметної галузі і постановка задачі, дослідження менеджменту і інтеграції різнорідної інформації, вивчення можливості використання в науково-освітній сфері.

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної галузі та постановка задачі	03.04.2023	виконано
2	Аналіз проблем	10.04.2023	виконано
3	Аналіз методів та алгоритмів	11.04.2023	виконано
4	Розробка алгоритмів, проектування та розробка ПЗ	20.04.2023	виконано
5	Підготовка пояснювальної записки	30.04.2023	виконано
6	Підготовка презентації та доповіді	05.05.2023	виконано
7	Попередній захист	09.05.2023	виконано
8	Нормоконтроль, рецензування	18.05.2023	виконано
9	Занесення роботи в електронний архів	19.05.2023	виконано
10	Допуск до захисту в зав. кафедри	19.05.2023	виконано

Дата видачі завдання 03.04.2023 р.

Керівник проф._____(Шостак І.В.)

Завдання прийняв до виконання_____(Кириченко Р.П.)

Кваліфікаційна робота магістра містить: 97 с., 15 рис., 1 табл., 30 джерел.

МЕТОДИ МЕНЕДЖМЕНТУ, ІНТЕГРАЦІЯ, РІЗНОРІДНА ІНФОРМАЦІЇ, МОДЕЛІ ЗНАНЬ, ДОСЛІДЖЕННЯ, АНАЛІЗ МОДЕЛЕЙ.

Об’єктом дослідження є методи менеджменту і інтеграції різнорідної інформації для моделей знань.

Метою роботи є підвищення якості та зручності документообігу в науково-освітній сфері.

Методи рішення базуються на таких технологіях як Kotlin, Spring, Hibernate, Liquibase, SQL Server, Swagger-UI, Figma.

В результаті роботи було досліджено методи менеджменту і інтеграції різнорідної інформації для моделей знань, наявні методи аналізу текстової інформації, спроектовано систему документообігу основану на онтології.

Master's qualification work contains: 97 p., 15 pictures, 1 table, 30 sources.

METHODS OF MANAGEMENT, INTEGRATION, DIVERSITY OF INFORMATION, MODELS OF KNOWLEDGE, RESEARCH, ANALYSIS OF MODELS.

The object of research is methods of management and integration of disparate information for knowledge models.

The purpose of the work is to improve the quality and convenience of document circulation in the scientific and educational sphere.

Solution methods are based on such technologies as Kotlin, Spring, Hibernate, Liquibase, SQL Server, Swagger-UI, Figma.

As a result of the work, methods of management and integration of disparate information for knowledge models, available methods of textual information analysis, and a document circulation system based on ontology were designed.

Умови публікації пояснювальної записки

Я,
Кириченко Роман Павлович
(прізвище, ім'я, по батькові)

студент(ка) групи ІПЗздм-21-2 здобувач вищої освіти на другому (магістерському) рівні

кафедра програмної інженерії ,
(повна назва кафедри)

заявляю: моя кваліфікаційна робота на тему
Дослідження методів менеджменту і інтеграції різнорідної інформації для моделей знань ,
(назва роботи)

що буде представлена до ЕК для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений (а) з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ВСТУП

Застосування автоматичних систем аналізу документообігу й систем перетворення документів в сфері освіти, дозволять автоматизувати процес оформлення освітньо-наукової документації, і тим самим спростити процедуру перетворення документів створених за різними стандартами.

Застосування існуючих методів для обробки цього типу документації суттєво ускладнене тим, що документи написані природньою мовою, а застосовуване в них форматування, як правило, не повністю формалізує те семантичне наповнення, яке існує у вихідному документі. Враховуючи точність документації й природно-мовний характер документів, застосування використовуваних зараз статистичних методів не дозволяє вирішити завдання автоматичної обробки освітньо-наукової документації.

Для рішення цього завдання необхідно застосовувати методи, що дозволяють описувати й обробляти семантику документів в «машинно-зрозумілому» вигляді. Необхідно однозначне «машинно-зрозуміле» подання текстової інформації, набір стандартів для подання семантичної інформації, набір універсального інструментарію для опису предметної області, які дозволили б проводити аналіз різних документів і перетворювати їх до певного загального виду.

Застосування нових методів опису семантики документів в «машинно-зрозумілому» виді й розробка механізмів її порівняння й оцінки на основі онтологічно-базуємих доповнень формують новий клас систем, які дозволять автоматизувати процес обробки й формування документів в сфері освіти і науки. Також з'являться можливості для перекладу документів, створених за українськими стандартами до затверджених європейських стандартів.

У даній роботі запропонована модель організації інформаційної системи нового покоління, що враховує семантичну складову текстових документів. Запропоновані методи інтеграції інформації на основі стандартизованих моделей

знань, методів інтеграції онтологій, методи обробки інформації й одержання нових знань.

Метою дослідження є підвищення якості менеджменту документів і спрощення процесу інтеграції різнорідної інформації, на основі використання стандартизованих моделей знань.

Поставлена мета роботи обґрунтувала наступні завдання дослідження:

– розробити методи побудови систем інтеграції й менеджменту електронних документів на основі різних концептуальних моделей (онтологій);

– розробити методи трансформації онтологій для забезпечення автоматизованого розширення, модифікації й інтеграції онтологій;

– розробити методи одержання нової інформації завдяки застосуванню правил RuleML до масиву онтологічних і анотаційних даних;

– розробити методи побудови запитів користувачів до масиву електронних документів і візуалізації результатів.

Об'єктом дослідження є автоматизований менеджмент інформаційних ресурсів в Інтернеті, зокрема, з використанням технологій Semantic Web.

Предметом дослідження є методи менеджменту, інтеграції й повторного використання інформації на основі технології Semantic Web на прикладах електронних документів, що відносяться до освітнього процесу.

Основними методами дослідження є методи моделювання текстових інформаційних систем, методи обробки й кодування інформації, рекомендовані в рамках ініціативи консорціуму W3C [1] щодо анотувань інформаційних джерел.

Наукова новизна роботи. У процесі вирішення поставлених завдань були отримані наступні результати:

1. Розроблений метод побудови інтелектуальних систем менеджменту та інтеграції інформації, яка сформована при використанні стандартів Semantic Web. Завдяки застосування таких систем забезпечується можливість семантичного аналізу різнорідної інформації. Ця можливість надає суттєві переваги перед існуючими системами.

2. Метод трансформації онтологій, який засновані на застосуванні стандартів Semantic Web одержав подальший розвиток. Цей метод надає можливості для інтеграції різнорідної інформації, яка сформована з використанням стандартизованих моделей знань, стандартів Semantic Web і описана завдяки різним концептуальним моделям.

3. Одержав подальший розвиток метод надання «прологоподібних» правил, що описують знання на основі застосування принципів RuleML для написання сценаріїв обробки інформаційного наповнення анотаційної та онтологічної баз. Цей метод дозволяє застосовувати набори «прологоподібних» правил щодо анотаційного масиву даних і отримувати результат логічного виводу інформації.

Розроблені методи побудови систем менеджменту, інтеграції інформації, інтеграції онтологій та методи надання «прологоподібних» правил можуть бути застосовані при створенні інформаційних систем менеджменту та інтеграції інформації нового покоління, які будуть здатні виконувати семантичний аналіз документів.

Розроблені методи найбільше доцільно та ефективно використовувати для вирішення задач проектування в наступних областях:

- проектування інтелектуальних інформаційних систем документообігу;
- проектування інформаційних систем пошуку, менеджменту й обробки інформації в Інтернеті;
- проектування систем автоматичного реферування інформаційних джерел.

На підставі розроблених методів рішення практичних завдань проектування систем автоматичного реферування інформаційних джерел і завдань проектування інформаційних систем пошуку й обробки інформації в Інтернет спроектована підсистема обробки й аналізу текстової інформації для підтримки інформаційного пошуку в глобальних і локальних інформаційних базах.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ ДОСЛІДЖЕННЯ

1.1 Огляд існуючих систем накопичення та аналізу інформації, а також методів обробки інформаційних джерел, використаних в цих системах

Створення інформаційних систем на теперішній момент є одним з найважливіших і ресурсномістких напрямків у діяльності техногенного суспільства.

Інформаційні системи – це будь-які системи, які забезпечують людей даними, знаннями або інформацією про процеси, що протікають у навколишньому світі [2].

Існує багато видів інформаційних систем: системи обробки даних, інформаційні управляючі системи, системи пошуку й накопичення інформації, системи інтеграції гетерогенних інформаційних ресурсів та інші, які використовуються в різних областях людської життєдіяльності.

Новим класом інформаційних систем є системи, що містять компоненти з інтелектуальною обробкою інформації. Ці системи покликані перевести існуючі механізми отримання знань, інтеграції інформації та прийняття рішень на принципово новий рівень.

Ядром таких інтелектуальних інформаційних систем є системи аналізу й обробки інформації, і вдосконалення їх дозволить значно поліпшити якість роботи усіх інформаційних систем загалом.

Згідно з даними в дослідження "How much of the web is archived?" (Яка частина всесвітньої павутини архівована?) [3], проведеного в 2011 році, 30-95% інтернет-сторінок було індексовано тільки однією пошуковою системою (Google, Bing, Yahoo, Diigo, Arhive-It, тощо), 17-49% мають «покриття» у 2-5 пошукових систем, 1-8% сторінок «покриті» на рівні 6-10 систем і тільки 8-63% індексовані більше ніж десятьма пошуковими системами. На практиці це означає, що більшість

веб-сторінок не підлягає індексації та іншим видам обробки, які допомагають користувачам знаходити інформацію в Інтернеті.

Постійне невпинне зростання кількості інформації та збільшення числа користувачів інформаційної мережі Інтернет приводить до стрімкого зростання кількості інформаційних ресурсів (за оцінками сервісу WorldWideWebSize.com приблизна кількість індексованих інтернет сторінок вже перетнула поділку в 4.5 мільярдів), а це, у свою чергу, приводить до підвищення вимог щодо якості, швидкості й точності роботи пошукових систем.

Динамічне зростання вимог неминуче приведе до ситуації інформаційного вибуху, якщо найближчим часом не будуть застосовані принципово нові принципи обробки й надання інформації.

При зверненні до пошукової машини звичайний користувач намагається знайти необхідну інформацію максимально і просто. Критеріями оцінки користувачем якості роботи інформаційно-пошукової системи зазвичай є:

- повнота пошуку;
- точність пошуку інформації;
- актуальність;
- швидкість пошуку;
- наочність надання інформації.

Повнота пошуку інформації – це одна з основних характеристик пошукових систем. Вона показує співвідношення кількості знайдених результатів/документів до загального числа сторінок/індексованих документів в Інтернеті, відповідних запиту користувача. Наприклад, якщо в Інтернеті зберігається інформація стосовно об'єкту на 100 сторінках, а на запит були повернуті посилання на 30 сторінок, то повнота пошуку становить 0.3. Чим більше ця величина, тем більше ймовірність, що користувач знайде потрібний йому документ, звичайно якщо такий документ існує в Інтернеті.

Точність пошуку – це ступінь співвідношення знайдених документів згідно запиту користувача. Наприклад, якщо по запиту користувача знайдено 100

документів і в 50 з них перебуває пошукова інформація, а інші потрапили у відповідь через помилки роботи статистичних механізмів при складанні індексу, то точність пошуку складе 50/100 (0.5). Чим точніше пошук, тем швидше користувач знаходить необхідну для нього інформацію.

Актуальність – це характеристика, яка показує час, з моменту появи документа в мережі до занесення його в індексну базу. Наприклад, якщо документ був занесений автоматичною системою індексації через 2 години після створення, то актуальність такого документу становить 2 години.

Швидкість пошуку – це час, за який пошукова система виконує аналіз запиту й повертає результат пошуку.

Наочність надання інформації – цей параметр характеризує зручність користування пошуковою системою. Досить часті випадки, коли при виконанні пошуку система повертає велику кількість інформації й користувач самостійно робить додатковий пошук відповіді серед запропонованих результатів. У такій ситуації зручний та інтуїтивно зрозумілий інтерфейс відіграє важливу роль у системі.

Через постійне зростанні кількості інформаційних ресурсів необхідною вимогою до пошукових систем також є і постійне поліпшення параметрів пошуку електронних документів.

Методики та їх недоліки для підвищення якості роботи існуючих інформаційних систем накопичення та аналізу інформації

Однієї з найбільш великих і широко використовуваних в Інтернеті пошукових систем є пошукова система Google, яка має ряд характерних особливостей, властивих усім пошуковим комплексам, що працюють у просторі Інтернет.

Основною ідеєю цих методів є пошук по ключових словах: автоматична пошукова система проводять аналіз інформації, що перебуває в Інтернеті та на гіпертекстових сторінках, і створює спеціальний індекс ключових слів. Коли

користувач задає запит до пошукової системи, із запиту виділяються ключові слова й по них виконується пошук сторінок, відповідних до цього запиту.

Зрозуміло, що точність такого пошуку не може бути висока, оскільки не враховується контекстуальна складова текстової інформації. Для підвищення якості роботи системи використовуються різні додаткові методи, які будуть розглянуті далі.

Важливим компонентом, що впливає на значення показника **повноти пошуку**, є робота системи збору й обробки інформації. В сучасних пошукових системах здійснюється вдосконалення пошуку за рахунок збільшення обчислювальної потужності систем попереднього збору й аналізу інформації, що дозволяє обробити одночасно більшу кількість інформаційних джерел.

Ядром системи є спеціальний програмний агент (веб-сканер) [4], що обходить сторінки із заданими URL та індексує їх, заповнюючи спеціальну централізовану інформаційну базу. Використання подібної технології також дозволяє збільшити кількість сторінок/документів, що оброблюються одночасно за рахунок збільшення кількості одночасно працюючих агентів на різних платформах, кожен з яких одночасно запущений на окремому сервері та виконує власні завдання.

Існує кілька технологій паралелізації обчислень:

– відносно різнотипних операцій: коли на одному комп'ютері виконується тільки один тип операцій над даними, і для одержання результату, дані послідовно обробляються на різних обчислювальних платформах та передаються після кожного виконаного етапу від машини до машини по мережі;

– відносно різних обчислювальних потоків: коли вхідна інформація розділяється за будь-якою ознакою, і на кожній програмній платформі обчислення проводяться паралельно.

Пошукові системи можуть застосовувати, як один із цих підходів так і обидва одночасно. В той час, як застосування централізованого сховища дозволяє

об'єднати воєдино різні інформаційні компоненти, забезпечуючи високий ступінь масштабованості системи.

Об'єднання інформації в єдиний індексний файл не є особливо складною операцією: завдяки загальному стандарту відбувається просте злиття й тестування бази, на наявність помилок. Причому інформація з поточної бази передається в нову формовану базу як одна зі складових частин і повністю враховується при формуванні оновленого індексу підчас наступної повторної індексації, що може відбутися за день, тиждень, місяць...

Для поліпшення якості роботи пошукових механізмів і збільшення **точності пошуку** використовуються методи обліку структури Web документів [5]. Оскільки інформація, що перебуває в заголовку документа або коментарі на посилання, мають більшу інформативність (на одиницю пропозиції) щодо змісту документа, аніж просто текст документа, то пошук серед цієї інформації дозволяє одержати кращі показники параметру точності пошуку. Підвищення точності також досягається за рахунок використання додаткових механізмів:

- розпізнавання граматичних омонімів;
- синтаксичного аналізу;
- виділення стійких позначень і обробка їх як окремих лексичних одиниць;
- ранжирування;
- аналізу й обробки складних запитів.

Омоніми – слова, які мають різний сенс та однакове написання. Розрізняють лексичні й граматичні омоніми. Лексичні ставляться до одній частині мови, а граматичні омоніми ставляться до різних частин мови, тому за написанням в них звичайно збігаються тільки окремі форми.

Наявність у тексті омонімів знижує точність пошуку й збільшує індексну базу, а їх облік їх дозволяє якоюсь мірою компенсувати недоліки, пов'язані із присутністю омонімів. На сьогоднішній день застосовують аналізатори/ **здатні розрізняти значення тільки граматичних омонімів**.

Синтаксичний аналіз також дозволяє збільшити точність пошуку за рахунок виявлення деяких власних імен і проводити облік їх при складанні загального

індексу. Таким самим чином вчиняють і зі стійкими означеннями, виконуючи для кожного з них їх обробку у якості окремих лексичних одиниць.

Важливим компонентом, що забезпечує гарні показники точності пошуку при обробці інформації є ранжирування. Це процес, у межах якого кожному посиланню, отриманому в результаті обробки запиту, привласнюється власний індекс, а далі виконується сортування інформації залежно від значень цього індексу.

Для сучасних пошукових систем, що працюють на статистичних методах аналізу інформації, важливими є наступні параметри:

– кількість входжень слів у документ;
– відстань між словами запиту;
– форми слів запиту: найбільший пріоритет віддається словам, використаним у тому ж відмінку, що й у запиті користувача;
– популярність: пошукова машина встановлює індекс популярності сторінок в Інтернеті, облік якого дозволяє поліпшити показники точності пошуку;
– відносна частота входження пошукових слів у документі до відношення загальної числа кількості слів у документі;
– вага посилання документа: обчислюється з урахуванням гіперпосилань, що посилаються на цей документ і містять ключові слова запиту.

Окрім цього, важливими параметрами, здатними поліпшити точність пошуку, є засоби, які надаються користувачеві для вдосконалення самого запиту до пошукової системи:

- уточнення порядку слів
- набір логічних операцій додавання чи виключення
- операція лапок
- облік розділових знаків.

Усе це збільшує точність пошуку, але **зменшує його повноту**. На жаль, **сучасні системи, що працюють на статистичних принципах і без обліку**

контексту, не в змозі забезпечити значення пошуку даних на високому рівні (коли показник точності пошуку буде приблизно рівний 1) .

Для рішення цього завдання необхідне застосування принципове нових методів, що дозволяють виразити інформацію в спеціальній «машинно-зрозумілій» формі.

Підвищення значення параметра актуальності пошуку є технічним завданням, яке вирішується збільшенням обчислювальної потужності системи та застосування оригінальних технічних рішень.

Для поліпшення показника актуальності в пошуковій системі Google застосовуються додаткові, допоміжні, метрики та методи:

1. **Click-Through Rate** (CTR) [6] - співвідношення кількості натискань на результат пошуку до кількості переглядів цього результату. Високий CTR свідчить про те, що результат пошуку відповідає запиту користувача та він знаходить те, що шукав.
2. **Dwell Time** [7] - час, який користувач проводить на веб-сторінці після переходу на неї з результатів пошуку. Чим більше часу користувач проводить на веб-сторінці тим більше це свідчить щодо корисності результату в видачі.
3. **Bounce Rate** [8] - співвідношення кількості користувачів, які відвідали веб-сторінку з результатів пошукової видачі та відразу покинули її без переходу на інші сторінки сайту, до загальної кількості відвідувачів. Високий показник відсотку відвідувачів, що покидають сайт без переходу на інші сторінки, може свідчити, що відвідувачі не знаходять на сайті те, що шукають.
4. **Pogo-Sticking** [9]- процесу переходу користувача з результатів пошуку на одну веб-сторінку, а потім швидке повернення до результатів пошуку та перехід на іншу сторінку. Це може свідчити про те, що відвідувач не знайшов те, що шукав на першій сторінці.
5. **Page Speed –** час на завантаження веб-сторінки. Швидкість завантаження сторінки є важливим фактором для користувачів, тому Google використовує цю метрику для оцінки якості роботи пошукової системи.
6. **Mobile-Friendliness -** наскільки добре веб-сторінка працює на мобільних

пристроях. Ця метрика стає все важливішою для оцінки якості роботи пошукової системи оскільки все більше користувачів використовують мобільні пристрої для пошуку інформації.

7. **Content Quality -** оцінка якості контенту на веб-сторінках, з урахуванням кількості та якості ключових слів, релевантності, авторитетності та актуальності. Відповідність контенту запиту користувача є ключовим фактором для визначення якості роботи пошукової системи.

Важливим компонентом пошукової системи є зручний інтерфейс, що дозволяє проводити додатковий пошук за результатами автоматичного пошуку й ранжирування. Але цей підхід не в змозі компенсувати всі недоліки статистичних методів обробки інформації. Збільшення обсягів інформації частково можна компенсувати збільшенням обчислювальних потужностей і неминучим зниженням точності пошуку.

Ситуація, коли може настати інформаційна криза, досить близька: на даний момент в Інтернеті перебуває більш 15 трильйонів документів [9], і обсяги інформації продовжують зростати з катастрофічною швидкістю.

Аби частково вирішити проблему індексації та аналізу документів в 2001 році Google почала робити індексацію PDF-документів [11] та закликає не тільки партнерів, а й державні органи в різних країнах доєднатися до подібної ініціативи обробки та оцифрування документів за використанням штучного інтелекту [12].

1.2 Сучасні методи надання інформації, що забезпечують можливість її інтелектуальної обробки

У сучасному світі, де існує маса різноманітних засобів і стандартів для надання інформації, пошук, обробка, та одержання нових знань із різнорідних джерел є дуже важкою в реалізації. Ситуація збільшується ще й тим, що більша частина описової інформації, не формалізується та залишається доступної тільки

розробникам. Таким чином, формування при спробі інтелектуальної системи, здатної виконувати різнотипні завдання, ми зустрічаємося з масою ускладнень, розв'язати які досить важко. Ще одна із проблем полягає в самій організації глобального інформаційного простору.

1.2.1 Структура глобального інформаційного простору

Глобальний інформаційний простір (Інтернет) має ряд особливостей: створювався хаотично й, незважаючи на чіткі механізми, що забезпечують нормальне функціонування інформаційної мережі, представляє із себе гетерогенну мережну структуру: складається з великого числа з'єднаних між собою серверних станцій, що забезпечують надання й обробку інформаційних джерел (рисунок 1.1).

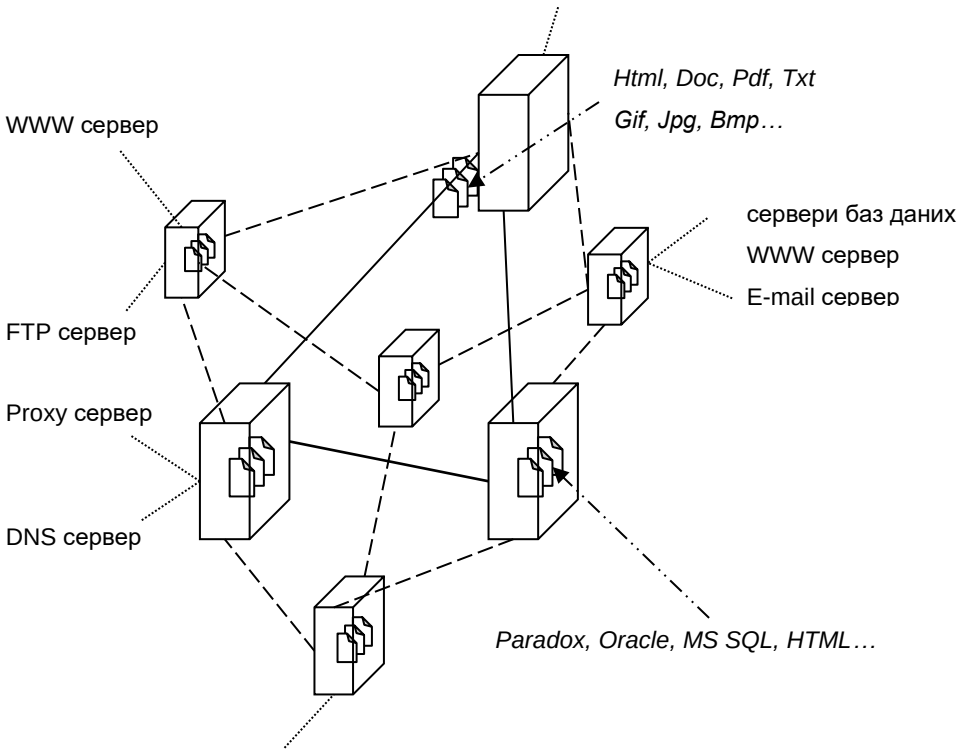

Рисунок 1.1 Схема організації глобальної інформаційної мережі Інтернет (Рисунок виконаний самостійно)

Кожен із серверів, підключених до Інтернету, працює лише в межах свого протоколу передачі інформації, забезпечуючи інформацією тільки сумісні із цим протоколом системи. В той час, як **внутрішні механізми відображення даних на**

серверних станціях залишаються закритими для зовнішнього інформаційного простору, і, як наслідок, при перенесенні інформації з однієї системи в іншу або при проведенні аналізу або обробки, виникають ускладнення, пов'язані з необхідністю проведення перетворення інформації зі стандартів однієї системи в стандарти іншої. Цей факт у значній мірі перешкоджає інтелектуально-автоматичній обробці текстових документів.

Інтернет не має чіткого централізованого управління - ця гетерогенна система буде прекрасно функціонувати навіть у випадку, коли якась частина мережі вийде з ладу. В Інтернеті лише контролюються унікальність IP адреси й HOST комп'ютера. Обмін даними по мережі відбувається з використанням великої кількості різних протоколів передачі даних. Така структура має й свої негативні сторони:

- дані в Інтернеті кодовані з використанням великої кількості різноманітних форматів і текстова інформація в явному вигляді не представлена в «машинно-зрозумілій» формі;
- централізоване керування мережею Інтернет украй важко здійснюване, що також ускладнює процес аналізу інформаційних ресурсів;
- деякі вузли й підмережі підключаються до Інтернету динамічним чином, і після відключення від мережі їх IP адреси та, як наслідок, розташування в мережі Інтернет можуть змінюватися.

Для створення ефективно працюючої інформаційної системи необхідне застосування принципове нових принципів опису інформації. Таким методом є створення унікального ідентифікатора, що забезпечує доступність і унікальність документів, як у локальних, так і в глобальних системах, створення універсального й у теж час формального засобу відображення інформації й розробка механізму опису навколишнього світу (онтології). Розроблені методи в рамках ініціативи W3C консорціуму на основі онтологій одержали загальну назву **Semantic Web** [13].

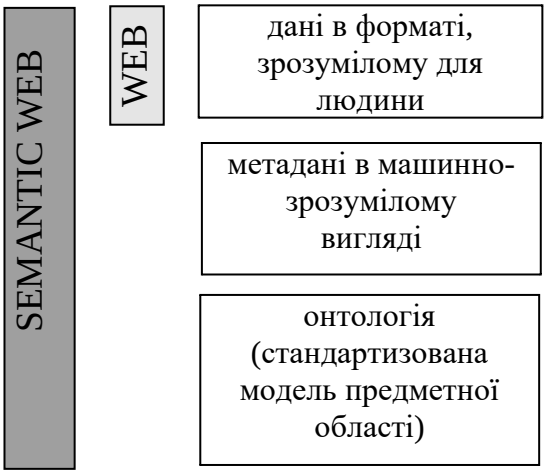

Рисунок 1.2 Логічна структура Semantic Web (Рисунок виконаний самостійно)

Система обробки представленої в Інтернет інформації, що заснована на онтологіях не вимагає перетворення інформації для сумісності, а пропонує додати до цієї інформації невеликий «машинно-зрозумілий» опис (рисунок 1.2).

Технологія Semantic Web – це результат удосконалення технології надання інформації в звичайному Інтернеті.

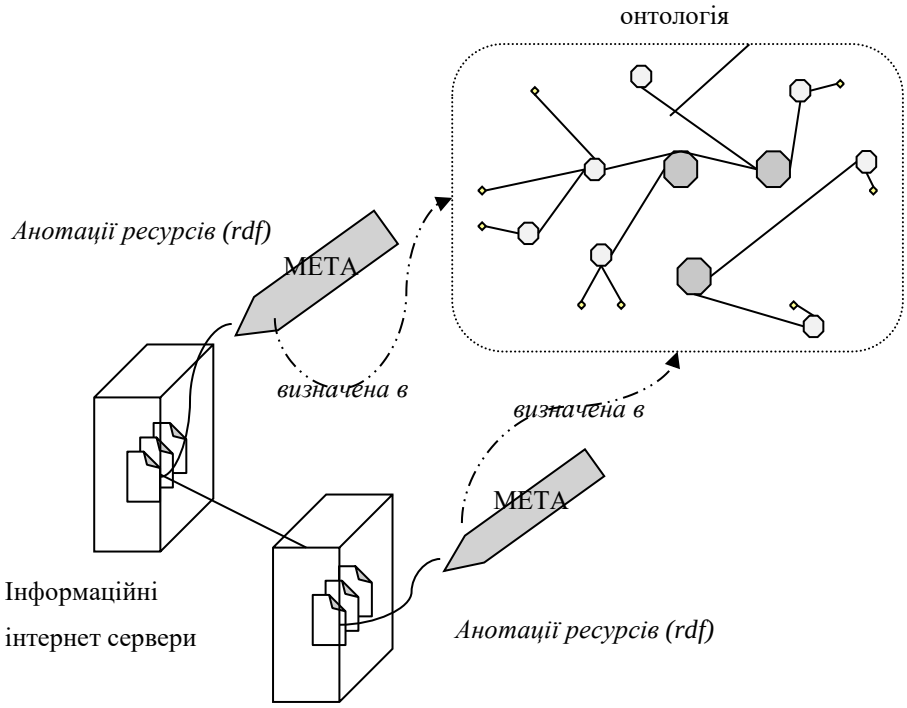

Рисунок 1.3 Топологічна структура Semantic Web (Рисунок виконаний самостійно)

Застосування технології Semantic Web забезпечує сумісність із усіма старими стандартами й механізмами надання інформації. В новому підході інтеграцію інформаційних джерел можливо досягти через зв'язування фрагментів метаінформації з використанням глобальної онтології (рисунок 1.3).

1.2.2 Ініціатива консорціуму World Wide Web Commerce (W3C) стосовно створення механізмів опису інформаційних ресурсів

Відчутним зрушенням в області представлення інформації з'явилися розробки, проведені міжнародним консорціумом W3C. Їхня суть полягала в створенні цілого набору технологій, що дозволяють виразити семантику у формальному вигляді, зістикувати різні семантичні описи за допомогою онтологій, сформувати різноманітні запити до анотаційної та онтологічної баз знань, а також наборів правил для виводу нових знань.

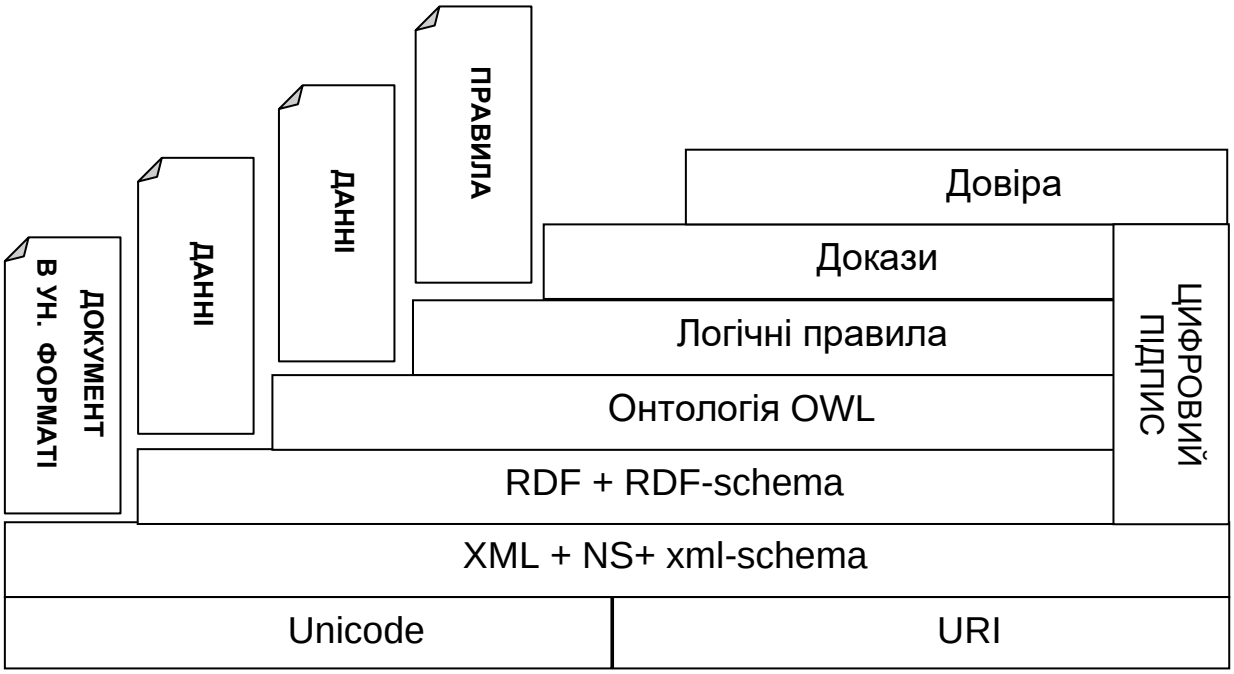

Рисунок 1.4 Схема можливої організації Semantic Web (Рисунок виконаний самостійно)

Ідеї цього підходу вкладаються в схему організації SW (семантичної павутини), запропонованої Тімом Бернерз-Лі [14]. На першому рівні цієї схеми (рисунок 1.4) представлені стандарти, що забезпечують однозначну інформацію щодо місцезнаходження документів в мережі: стандарт **URI** і стандарт універсального шрифту **Unicode**, що містить коди всіх міжнародних мовних символів. Це забезпечує однакове відображення документів на комп'ютерних та платформах з різною локалізацією.

URI (**Uniform Resource Identifiers**) надає простий і гнучкий метод для ідентифікації ресурсів [15]. Не всі ресурси можуть бути доступні через глобальну інформаційну мережу Інтернет, і тому в новому стандарті для вказання місця розташування ресурсу використовується ID локальної системи в тому випадку, якщо в системі немає доступу до глобальної комп'ютерної мережі, або URL системи, якщо доступ в Інтернет є присутнім.

Перший рівень створює необхідну базу для здійснення можливостей створення документів, використовуючи загальні правила синтаксису.

Стандарти наступного рівня – XML стандарти, покликані забезпечити синтаксичну сумісність різнорідної інформації [16].

XML (Extensible Markup Language) – мова розмітки документів, відповідає SGML [17] стандарту й дозволяє структурувати інформацію різного типу, використовуючи довільний набір тегів. Цінністю цього стандарту є те, що він надає можливість для синтаксичної інтеграції різнорідної інформації, вираженої за допомогою будь-яких стандартів, забезпечуючи при цьому надійні механізми обробки, передачі через Інтернет і перетворення інформації стандартними засобами.

XML використовується для відображення інформації в широкому класі завдань, рішення яких зв’язано зі створенням і обробкою структурованої інформації:

– враховуючи більшу гнучкість мови, XML є базовим стандартом для цілого ряду мов, призначених для опису інформаційних ресурсів і навколишнього світу, таких як RDF і OWL;

– завдання розробки складних інформаційних систем, що містять велику кількість розподілених компонентів, що спілкуються між собою за допомогою пересилання пакетів інформації довільної структури;

– завдання опису даних довільного типу й відображення спеціалізованої інформації, такий як математичні, фізичні, хімічні формули тощо;

– документи, створені в рамках XML, можуть використовуватися як проміжний компонент у системах баз даних, заміняючи SQL і надаючи більш широкі можливості для роботи в розподілених системах;

– застосування стильових таблиць дозволяє створювати універсальні документи, що не залежать від конкретного обладнання, на якому буде відбуватися відображення, інформаційні сторінки;

– у завданнях розподілу інформації між декількома функціонуючими системами XML-похідний документ може передаватися, трансформуватися й збережуться на сервері вроздріб, що забезпечує зручні механізми для побудови розподілених комплексів;

– XML-похідні файли також можуть бути використані звичайними додатками для нагромадження й зберігання різних даних.

XML-похідний файл має простий текстовий формат, і це забезпечує простоту й легкість редагування документів. Незважаючи на текстовий вид документів, ці файли легко оброблюються комп'ютером і мають чітку структуру, яку забезпечують спеціальні маркери. Набір маркерів вільно міняється, забезпечуючи цим можливості для застосування мови в множині різнотипних систем.

Для обробки XML-подібних файлів необхідно створювати власні оброблювачі, засновані на спеціальних бібліотеках, які й будуть аналізувати й модифікувати інформаційні файли системи. Подібна організація накладає на внутрішні процедури користувача необхідність інтерпретації даних: інтерпретація задається в явній формі людиною при написанні програмного оброблювача даних.

Для перевірки синтаксису XML інформації передбачений спеціальний механізм, що дозволяє спеціальним програмним бібліотекам перевіряти на правильність дані, що вводяться у файл.

Це механізм, заснований на використанні файлів DTD [18], в результаті перетворених в XML-схему. У цих файлах явно описується синтаксис розробленого XML-подібної мови. Важливим компонентом мови XML є Namespace, що забезпечує унікальну ідентифікацію кожного елемента. Namespace – один з найважливіших компонентів мови, бо він визначає унікальну групу імен для кожного тегу XML-подібного документа, а це дозволяє уникнути помилок при інтеграції декількох документів, створених різними системами або користувачами.

Ще один з важливих компонентів XML – це наявність об'єктної моделі документа (DOM) [19]. DOM – це представлення структури документів у вигляді сукупності певних об'єктів. При обробці документа бібліотеками-парсерами (конвертерами), об'єкти поєднуються в спеціальну деревоподібну структуру й проводиться формування в пам'яті машини дерева об'єктів. Парсери надають програмам усі необхідні методи для завантаження, редагування й збереження XML-подібних документів. XML стандарт також надає зручні можливості для роботи з файлами в мережі: створюючи XML документ користувач одержує можливість віддалено редагувати, завантажувати або зберігати документ, використовуючи тільки можливості XML парсера.

Для здійснення можливості універсальної відображення інформації була розроблена мова XSL (Extensible **Stylesheets Language**) [20]. Ця технологія дозволяє абстрагуватися від конкретної прив'язки до стилю відображення інформації. Застосування XSL дозволяє, з одного боку, відокремити правила оформлення сторінки окремо від її змісту, до того ж структура інформаційного джерела буде заздалегідь описана й ми отримаємо можливість в результаті використовувати інформаційне джерело для рішення завдань машинної обробки.

Недоліком такого підходу є відсутність можливості інтерпретації машиною вмісту документа. Оскільки машина здатна лише відрізнити теги друг від друга, але не в змозі зрадити цим тегам яку-небудь інтерпретацію, за винятком явного опису в XSL, як ці теги відображати.

Хоча реалізація XML мови і стала визначним етапом у розробці універсального засобу вираження семантики через синтаксис мови, але це був лише

перший крок у створенні інтелектуального Web. Незважаючи на переваги XML, він забезпечує тільки перший рівень стандартизації вистави інформації. XML дозволять використовувати загальні синтаксичні правила при формуванні XML-подібного файлу, але вміст цього файлу є повністю незрозумілим для машинної автоматичної обробки.

Для забезпечення загальнозрозумілого відображення даних на інформаційних ресурсах був запропонований і специфікований стандарт **RDF** [21].

RDF (Resource Description Framework) – універсальний механізм опису інформаційних ресурсів.

Розробка цього механізму дозволила покласти початок нової епохи розвитку Інтернету - етапу впорядкування Інтернет простору й переходу до машинно-зрозумілого відображення інформації. RDF створює здатність до взаємодії між інформаційними системами, не накладаючи особливих обмежень на семантику описуваної інформації. Застосування RDF дозволило сформувати новий клас систем, здатних робити внутрішню інтерпретацію вмісту довільних інформаційних джерел.

RDF (Resource Description Framework) – це мова для опису інформаційних ресурсів в Інтернет. Його важливою властивістю є можливість представлення в явному виді метаданих про ресурси глобальної інформаційної мережі, таких як: назва, автор документу, дата останньої модифікації ресурсу Інтернет мережі.

RDF може описувати ресурси, не тільки представлені в Інтернеті. Ця властивість досягається неймовірними виразними можливостями мови й можливістю його розширення. RDF використовується у випадках, коли інформація призначає не тільки для людини, але для машинної обробки. RDF надає спеціальну структуру, що дозволяє виразити інформацію в «машинно-зрозумілому» вигляді.

Основними технологіями, які лягли в основу RDF є URI і технологія опису ресурсів в термінах простих властивостей. Подібна технологія дозволяє надавати прості описи ресурсів таких, як графи, вузли і дуги, та подібні цим ресурсам. При описі подібних ресурсів вказується перелік властивостей і значень цих властивостей для конкретних описуваних об'єктів. Основою для RDF став XML-

подібний синтаксис, що відповідно привнесло в RDF усі гідності XML підходу. Стандарт RDF 1.1 було затверджено у 2014 р.

Основними досягненнями цієї технології стали:

– надання інструментарію для семантичного опису ресурсів;

– забезпечення можливостей створення засобів, що забезпечують взаємодію між застосунками, які обмінюються інформацією.

Для опису ресурсу в стандарті RDF передбачена наступна технологія: усі ресурси описуються набором триплетів «об'єкт-атрибут-значення», де кожний із вхідних у триплет компонентів має своє посилання на бібліотеку концепцій. Це в свою чергу, дозволяє виконати перевірку правильності синтаксичного вживання цього поняття й посилання на інформаційну базу, що містить значеннєвий опис кожного з понять. Якщо всі посилання доступні, то з'являється можливість автоматичної інтерпретації вмісту документа. До інформаційного ресурсу опис прикріплюється завдяки застосуванню технології URI (рисунок 1.5.)

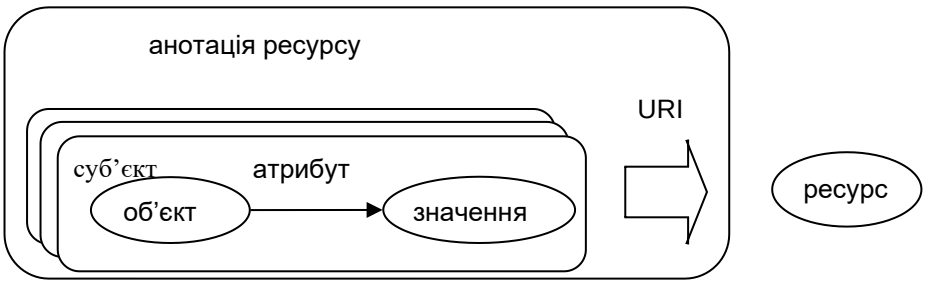

Рисунок 1.5 Структура опису інформаційного ресурсу в рамках технології RDF (Рисунок виконаний самостійно)

Проміжною технологією, що забезпечує значеннєвий опис RDF компонентів, стала технологія **RDFS** [22] (Resource Description Framework Schema). Вона дозволяла описувати RDF компоненти, що входять в опис якого-небудь ресурсу, використовуючи стандартний набір основних класів (рисунок 1.6). Завдяки об'єктам, що входять в RDFS, з'явилася можливість явно вказувати призначення й зміст описів, що входять в RDF- документ. Стандарт RDFS 1.1 було затверджено 2014 році, та він надає наступні можливості:

– опис словників для RDF;
– організація цих словників у типізовані ієрархії;
– надання можливості очевидного оголошення семантичних зв'язків між термінами словників.

Проте використання RDFS не було здатне повною мірою забезпечити можливість опису об'єктів RDF-документів з кількох причин: відсутність зв'язки між декількома RDFS документами й обмежені описові можливості цього стандарту.

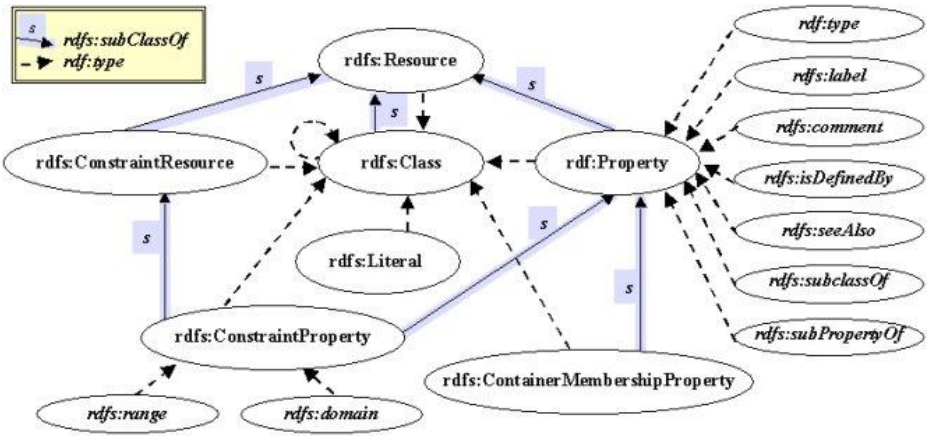

Рисунок 1.6 Ієрархічна структура класів RDF Schema
(https://www.w3.org/2001/sw/RDFCore/Schema/20010913/)

t = (rdfs:subclassof) вказує на те, що об'єкт є підкласом;
s = (rdf:type) вказує на визначення типу об'єкта;
rdfs:Class – вказує на те що об'єкт є класом;
rdfs:Resource – основний клас ресурсів;
rdfs:Property – основний клас властивостей;
rdfs:type – визначає тип об'єкта;
rdfs:range – визначає тип властивості;
rdfs:domain – визначає предметну область об'єкта;
rdfs:Literal – основний клас літерала;
rdfs:Constraintproperty – визначає обмеження для властивостей об'єкта;
rdfs:Subclassof – визначає підклас об'єкта;
rdfs:Subpropertyof – визначає підвластивість об'єкта;
rdfs:Label – основний клас міток;
rdfs:Comment – основний клас коментарів;
rdfs:isdefinedby – вказує на наявність визначення;
rdfs:seealso – клас посилання;
rdfs:Containermembershipproperty – обмеження для групи властивостей.

Як рішення проблеми розширення й модифікації RDF Schema з'явилася теорія онтологій – технологія здатна усунути недоліки RDFS і надати універсальні гнучкі засоби опису.

Мовою для представлення онтологій в Інтернет просторі є OWL (Ontology Web Language) [23]. OWL-документ призначений для машинної обробки й надає набір виразних засобів для опису класів і об'єктів, що використовуються в RDF-документах та поділяється на кілька мов підкласів: OWL Lite, OWL DL, OWL Full [23]. Метою цього поділу є забезпечення максимальної продуктивності при роботі інтелектуальних систем аналізу інформації, зважаючи на те, що навіть фрагменти онтології дуже громіздкі й вимагають значної процесорної потужності для виконання операцій аналізу інформації.

OWL Lite надає мінімальний набір засобів для аналізу інформації, але забезпечує відносно легку машинну оброблюваність, у першу чергу призначений для рішення завдань класифікації.

OWL DL надає набір засобів, які гарантують одержання результату обчислень за кінцевий час.

OWL Full надає всі механізми, описані в стандарті OWL, повністю містить у собі стандарт RDF для опису деяких фрагментів інформації, але є дуже ресурсномістким при обчисленні.

Із 27 жовтня 2009 року OWL2 стала рекомендацією W3C і надає наступні можливості:

- опис онтологій;
- дозволяє зв'язати RDF описи, створені з використанням різних термінів, у єдину «машинно-зрозумілу» систему.

Спільне використання технології OWL і RDF забезпечує основи для автоматичної інтелектуальної обробки інформації [24]. Мова RDF дозволяє виразити інформацію в «машинно-зрозумілому» вигляді, не накладаючи обмеження на інформацію, описану в RDF-документі.

Створюються тільки посилання на сервер де перебуває інформація для опису кожного терміну цього документа. OWL дозволяє описати навколишній світ у тих же самих термінах, у яких RDF описує web сторінки: певний XML елемент, колекції сторінок або об'єкти, безпосередньо не доступний через Інтернет.

Таким чином, для інтелектуальної обробки необхідно зробити аналіз RDF-документа, провести вибір з нього всіх термінів, що перебувають у ньому, знайти онтологію або групу онтологій, здатних описати й зв'язати в єдиний інформаційний простір ці терміни. Тільки після цього з'явиться можливість інтелектуальної обробки вмісту документа.

Розробники напрямку, у рамках W3C консорціуму, DAML+OIL [24] запропонували незавершену концепцію мови RuleML [25], що може бути використана при побудові наборів правил для управління механізмами виводу нової інформації на основі логічних правил. Класифікація підмножин мови RuleML наведена на рисунку 1.7.

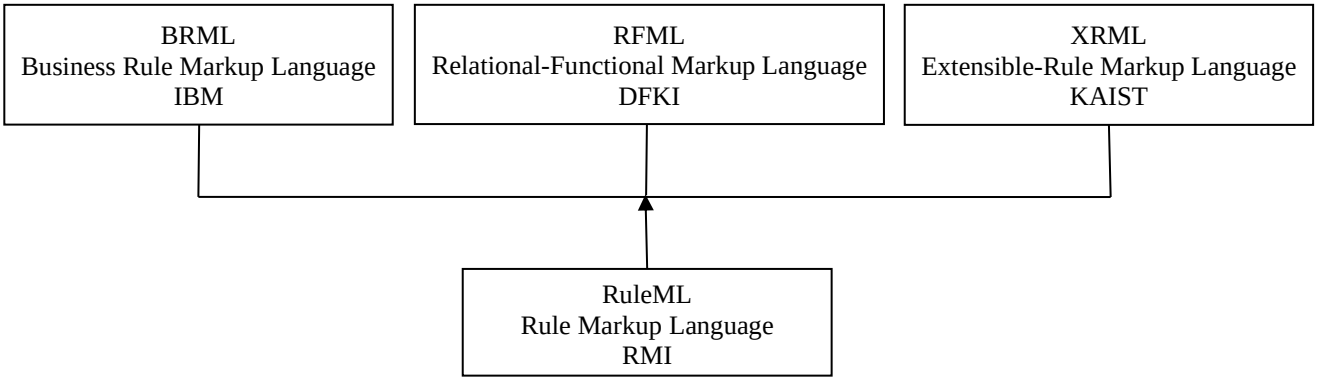

Рисунок 1.7 Класифікація підмножин мови RuleML (Рисунок виконаний самостійно)

Мова RuleML охоплює повний простір правил, починаючи від правил виводу, і закінчуючи реакційними правилами й правилами трансформації.

Використання запитів мовою RuleML дає можливість інтелектуального виводу в Web-онтологіях і формуванні динамічно-адаптивної поведінки сервісів і агентів.

Розроблювальна технологія, безумовно, вимагає складних технічних рішень, великої продуктивності обчислювальних центрів, але вона також здатна перевести обробку текстової інформації на якісно новий рівень, недосяжний для статистичних методів, які використовувалися раніше.

1.3 Аналіз можливостей практичного застосування інтелектуальної технології обробки текстової інформації

Розроблювальна технологія відображення текстової інформації має на даний момент ряд недоробок, проте принципи, що закладені в неї, здатні кардинально змінити якість обробки текстових документів.

Застосовувані зараз технології обробки текстової документації статистичними методами принципово не в змозі забезпечити високої (близькі до одиниці) точності пошуку інформації через відсутність контекстного розуміння змісту документа. В той час, як методи консорціуму W3C у стані врахувати настільки важливу складову, як контекст при обробці документів.

На даний момент розроблені й затверджені стандарти Unicode, URI, XML+XML Schema, RDF +RDF Schema і OWL (був затверджений як W3C рекомендація). Але перераховані стандарти описують тільки спосіб відображення інформації, що ж стосується програмних модулів для забезпечення доступу до компонентів стандартизованих файлів, то вони розроблені тільки для Unicode, URI, XML+XML Schema, RDF +RDF Schema.

Для OWL існує кілька тестових бібліотек для забезпечення доступу, але вони ще не повною мірою підтримують набір необхідних функцій. Для мови більш високого рівня RuleML програмні засоби ще не розроблялися. Але навіть розроблені компоненти для RDF, RDFS, OWL вже дозволяють почати роботу над створенням принципово відмінних інформаційних систем, що значно перевершують по ступеню інтелектуалізації.

Запропонований набір технологій вже дозволяє приступити до рішення складних завдань, пов'язаних з повторним використанням інформації, інтеграцією інформації в рамках єдиного інформаційного простору й проведенням інтелектуального виводу на основі метаданих, виражених у термінах мови RuleML.

Практичне застосування такі системи мали б у широкому класі науково виробничих комплексів:

- у системах менеджменту знань підприємств і консорціумів;
- у системах підтримки порівняльного аналізу українських і європейських стандартів вищої освіти;
- для підтримки високоінтелектуальних мультимедійних систем, що використовують засоби штучного інтелекту;
- для підтримки систем інтелектуалізації комп'ютерних інтерфейсів;
- у системах інтеграції баз знань.

1.4 Постановка задачі

Проведений у попередніх розділах аналіз показав, що найбільш адекватним вирішенням проблем інтелектуальної інтеграції інформації на основі стандартизованих моделей знань, повторного використання інформаційних джерел і інтелектуального логічного висновку на основі використання мови опису правил є використання технології Semantic Web (SW).

Використання цієї концепції вводить набір необхідних засобів для відображення семантики. Одним зі стандартів SW є онтологія. Онтології – це інструмент, а не активний учасник процесу. Очевидно, що система мета-контекстного обміну даними повинна бути розширена шляхом уведення в її структуру додаткової підсистеми онтологічних перетворень.

Особлива увагу необхідно приділити методам моделювання онтологій і процесу їх інтеграції, методам семантичного анотування інформації й методам інтеграції інформації на основі єдиної онтології.

У відповідності зі сказаним вище, у роботі вирішуються наступні завдання:

– розробити архітектуру системи інтеграції електронних документів, що відносяться до науково-освітнього процесу й сформованих на основі різних концептуальних моделей (онтологій);

– розробити методи підтримки семантичного анотування електронних документів у рамках концептуальної моделі, що має відношення до науково-наукового процесу в контексті стандартів освіти;

– розробити систему підтримки створення онтологій у галузі освіти;

– розробити механізм автоматизованого розширення, модифікації й інтеграції онтологій;

– розробити механізм виводу нових знань на основі онтологій, семантичних анотацій і RuleML;

– розробити механізм семантичних запитів користувачів до масиву анотованих електронних документів і візуалізації результатів.

2 АНАЛІЗ МЕТОДІВ МЕНЕДЖМЕНТУ І ІНТЕГРАЦІЇ РІЗНОРІДНОЇ ІНФОРМАЦІЇ

Розвиток інформаційної технології забезпечив ріст кількості інформаційних ресурсів. Проблема пошуку необхідної інформації на даний момент стоїть дуже гостро, сучасні пошукові машини просто не справляються із всезростаючими обсягами інформаційного простору.

Для рішення цього завдання необхідне застосування кардинальне нових методів, оскільки широко використовувані статистичні методи вже повністю вичерпали свій ресурс точності пошуку інформації та покладаються в своїй роботі на додаткові засоби та інструменти.

Для рішення поставленої задачі розробляється технологія семантичного анотування документів (коли до кожного документа додається додатковий файл, що містить опис цього документа в машинно-зрозумілій, структурно-стандартизованій формі) [26]. Оскільки анотації документів створюються різними людьми з використанням різноманітних синонімів і словоформ, ми повинні застосовувати механізм стандартизації семантичних описів.

Метою цього механізму є об'єднання описів у єдиний інформаційний простір. Обробка інформаційного наповнення цього простору здійснюється завдяки розробленим нами методам одержання нових знань, а взаємодія з користувачем – завдяки механізмам формування документів.

Кожний із цих компонентів буде докладно описаний далі.

2.1 Розробка оригінальної моделі представлення інформаційних ресурсів, що забезпечує можливість інтелектуальної обробки інформації

Через незавершеності технологій, що входять у модель представлену на рисунку 1.4, виконана переробка цієї моделі з метою приведення її до завершеного виду й адаптації для застосування в рамках системи менеджменту й інтеграції різнорідної інформації на основі стандартизованих моделей знань, що представлена на рисунку 2.1.

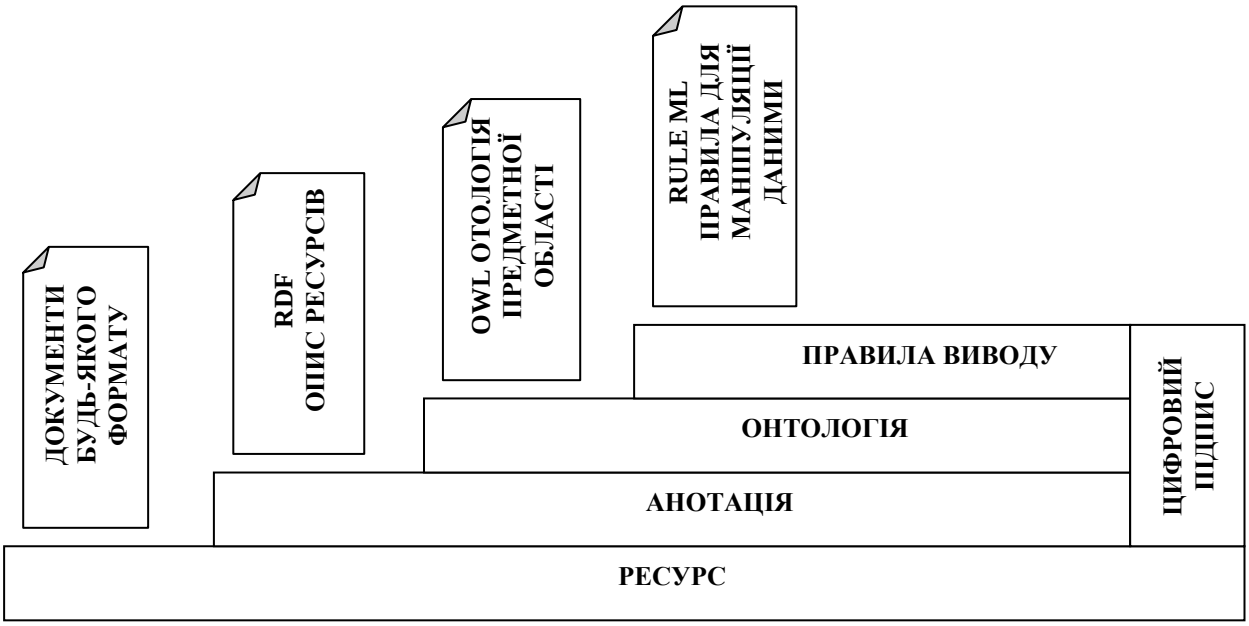

Рисунок 2.1 Ієрархічна схема рівнів представлення знань для системи інтелектуальної обробки інформації (Рисунок виконаний самостійно)

Схема трохи спрощена, враховуючи специфіку й призначення локальної системи інтеграції інформації в рамках концептуальної моделі освітнього процесу. На першому основному рівні перебуває **інформаційний ресурс**, виражений у вигляді електронного документа. Цей документ у рамках системи може бути ідентифікований за допомогою унікального URI. Це забезпечує можливість порівняння змісту вітчизняних освітніх курсів з освітніми курсами, побудованими згідно зразків, відповідних до стандартів країн Болонської конвенції. У якості

документа може виступати будь-який інформаційний ресурс, текст будь-якого формату або графічний файл.

До кожного інформаційного ресурсу прикріплюється своя **анотація**, виражена в «машинно-зрозумілій» формі у форматі RDF. Ця анотація описує вміст документа, причому від ступеня повноти залежить якість обробки документів. Процес перенесення змісту з документа в анотацію на даний момент не може бути повністю автоматизований, і тому введений такий параметр як електронний підпис.

Електронний підпис формується тільки в тих випадках, коли процес наповнення інформаційного змісту тією чи іншою мірою залежить від людини й саме людей повинен стежити за логічною правильністю введених даних. Хоча вводиться кілька критеріїв, що характеризують правильність внесення інформації, і в рамках роботи запропоновані механізми контролю й корекції інформаційних баз, тільки при правильному заповненні (коректному перенесенні змісту) гарантується 100% правильність результату при аналізі й обробці інформації.

Технологія, завдяки якій з'являється можливість формування анотації ресурсів, полягає в наступному:

– документ розділяється на окремі твердження, які описують ту або іншу властивість, ситуацію або дію;

– кожне твердження перекодується у формі триплету «об'єкт-атрибут-значення» у якості об'єкта може виступати ресурс: у якості атрибута – властивість або ресурс; у якості значення – значення властивості ресурсу (рисунок 2.2) [27];

– з онтології вибираються поняття, за допомогою яких і проводиться опис інформаційного ресурсу;

– твердження перетворяться в триплети мови RDF з використанням визначень, описаних в онтології.

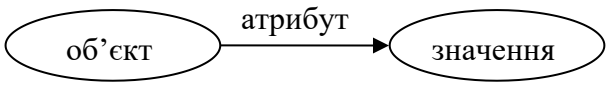

Рисунок 2.2 - Графічне відображення триплету (Рисунок виконаний самостійно)

Для кожної анотації формується ряд характеристик, які визначають повноту, «машинно-оброблюваність» та відсутність несуперечність анотування інформаційного джерела [28, 29].

Коефіцієнт повноти анотування інформаційного джерела показує, наскільки повно виконано перенесення інформації з файлу ресурсу в його семантичну анотацію. Параметр обчислюється автоматизованими системами по формулі:

$$V_{\text{п}} = \frac{N_{\text{а}}}{N_{\text{р}}}, \qquad (2.1)$$

де

$V_{\text{п}}$– коефіцієнт повноти анотування інформаційного джерела;

N_a– кількість «неподільних компонентів інформації» в анотації інформаційного джерела;

$N_{\text{р}}$– кількість «неподільних компонентів інформації» в інформаційному ресурсі.

Під терміном «неподільний компонент інформації» розуміється неподільна інформаційна одиниця, виражена у формі фрагменту або цілого речення. Значення цього коефіцієнта, близьке до 1, вказує на повне дублювання інформації інформаційного ресурсу в анотації.

Коефіцієнт «машинно-оброблюваності» анотації інформаційного джерела показує, наскільки описові компоненти, що входять в анотацію, є розпізнаваними для інформаційної системи. Параметр може бути обчислений автоматизованими системами по формулі:

$$V_{\text{м}} = \frac{C_{\text{а}}}{C_{\text{о}}}, \qquad (2.2)$$

де

$V_{\text{м}}$– коефіцієнт «машинно-оброблюваності» анотації інформаційного джерела;

C_a– кількість термінів і класів понять, що були використані в анотації інформаційного джерела;

$C_{\text{о}}$– кількість термінів і класів понять, що були використані в анотації й описаних в онтологічній базі.

Параметр дозволяє виявити необхідність трансформації онтологічної бази для повної інтерпретації інформаційного ресурсу. Значення параметра, що дорівнює 1, показує, що анотація може бути повністю інтерпретована за допомогою інтелектуальної системи без зміни вмісту онтологічної бази.

Коефіцієнт несуперечності анотаційного змісту вказує на той факт, що інформація, що міститься в анотації, не вступає в протиріччя із вже наявними фактами в рамках загальної анотаційної інформаційної бази. Параметр можна обчислити автоматизованими системами по формулі:

$$V_{\text{нп}} = \frac{F_{\text{нп}}}{F_{\text{о}}}; \qquad (2.3)$$

де

$V_{\text{нп}}$– коефіцієнт несуперечності анотації інформаційного джерела;

$F_{\text{нп}}$– кількість фактів, описаних в анотації, що не мають собі спростування в анотаційній базі;

$F_{\text{о}}$– загальна кількість фактів в анотації.

Параметр дозволяє визначити появу суперечливості у випадку внесення анотації до анотаційної бази. Значення параметра, рівне 1, вказує, що анотація може бути повністю внесена до анотаційної інформаційної бази без появи суперечливої ситуації. Значення, менше 1, вказує на необхідність додаткової переробки інформаційного вмісту анотації (уточнення) або перегляду набору анотацій, що вже входять до анотаційної інформаційної бази.

Онтологія також на першому етапі формується вручну, далі з'являється можливість завдяки запропонованим у роботі механізмам проводити інтелектуальну інтеграцію до загальної онтології з різних несуперечливих фрагментів або онтологій завдань. Для представлення онтології застосовується мова опису онтологій OWL. Застосування онтології відкриває можливості для автоматичної інтерпретації інформаційного змісту й дозволяє проводити інтелектуальну інтеграцію інформації. На рисунку 2.3 надана схема інтеграції

інформаційних ресурсів з використанням анотацій в «машинно-зрозумілому» вигляді та онтологій.

Саме механізм використання онтології дозволяє встановлювати однозначну відповідність між компонентами ресурсних анотацій. Одним з найважливіших питань є питання щодо інтеграції самих онтологій оскільки перед виконанням інтеграції інформації необхідно отримати єдину інформаційну базу.

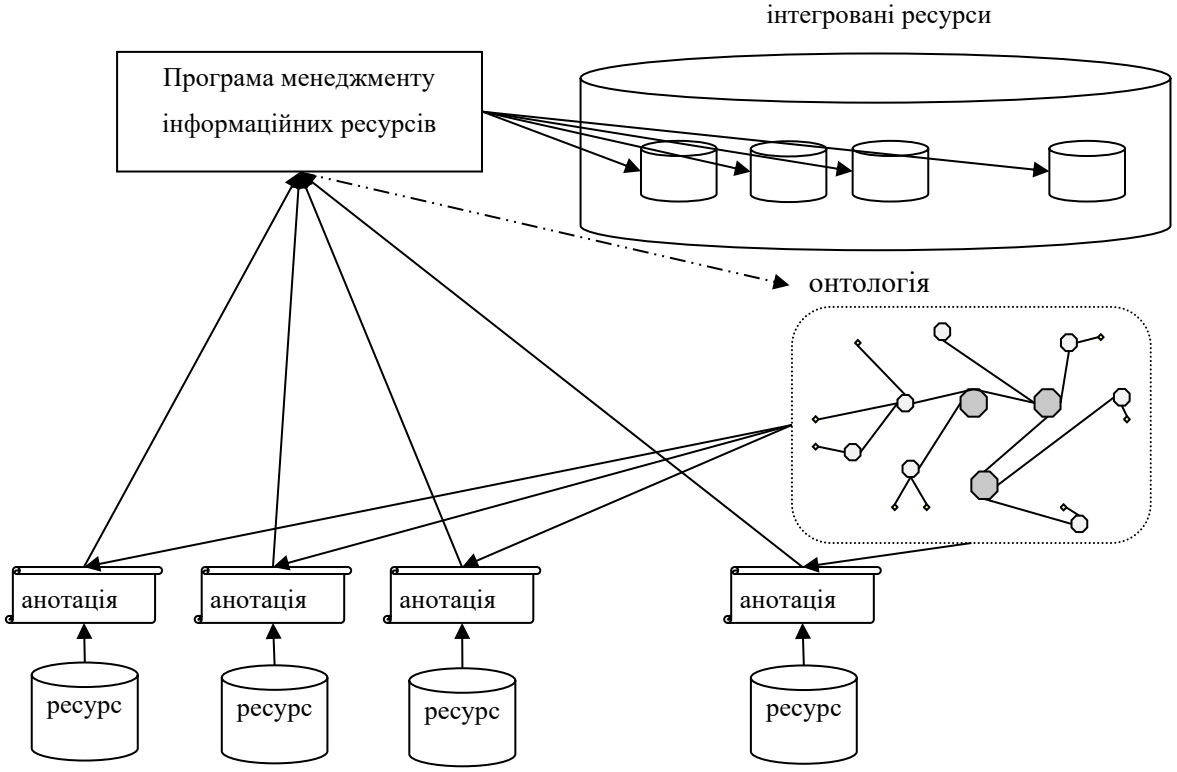

Рисунок 2.3 Схема інтеграції інформації на основі застосування анотацій, що описують ресурс і онтології як компонента формування єдиного інформаційного простору (Рисунок виконаний самостійно)

На вершині ієрархічної схеми перебуває рівень представлення правил виводу. Документи цього рівня також формуються вручну й відтворюють правила виводу, написані мовою RuleML. Вони також вимагають правильності складання для коректної роботи системи.

Застосування цих правил щодо анотацій, що інтерпретуються за допомогою онтологій, дозволяє отримати нові знання й виразити їх у формі нових автоматично згенерованих RDF документів. До того ж фрагменти онтологій та правила виводу

можна представити у вигляді інформаційних ресурсів для цієї схеми представлення інформації, в той час, як анотацією цих ресурсів буде метаінформація, що описує ці ресурси.

При виконанні правил мови RuleML надається можливість формувати й ресурси, які будуть доступні надалі для обробки, та анотації, які описують ці ресурси. Таким чином, архітектура дозволяє організовувати зворотний зв'язок між правилами виводу й перетворення інформації, це у свою чергу, створює можливість формування правил «самонавчання» для системи та з'являється можливість автоматичного внутрішнього коректування онтологічної бази.

2.2 Розробка механізму інтеграції онтологій

У сучасних інтелектуальних системах обробки й накопичення знань важливе місце займає онтологічний підхід. Використання цього підходу дозволяє виявляти зв'язки між класами й усуває неоднозначність представлення даних.

Що таке онтологія? У різних роботах розглядається різне розуміння цього терміну: від глибоко філософського до технічного. В такому випадку ми будемо розуміти під словом онтологія *специфікацію концептуалізації* однієї або декількох предметних областей. Онтологія – це спрямований граф, на вершинах якого перебувають концепти, об'єднані різними зв'язками, що бувають 2-х типів: основні (вказують на наявність узагальнюючого відношення) і додаткові (пов'язані з описом конкретної предметної області) [1]. У формулах теорії множин ми можемо представити онтологію як

$$O =< X, S, R >, \qquad (2.4)$$

де

X – кінцева множина концептів,

S – кінцева множина відносин узагальнення на множині концептів,

R – додаткова кінцева множина відносин, пов'язаних з описом предметної області.

Найпростіший вид онтології називається *таксономією,* він не містить додаткових зв'язків і репрезентує деревоподібну структуру. Для усунення неоднозначності представлення даних разом з онтологією використовується словник синонімічних термінів, що містить назву концепту, представленого в онтології, і синонімічні йому поняття.

Онтології бувають 3-х рівнів:

- глобальна онтологія;
- онтологія предметної області;
- онтологія завдання.

Глобальна онтологія включає базові концепти онтологій предметних областей, в той час, як онтологія предметної області – базові концепти онтологій завдань, що входять у цю предметну область.

Глобальна онтологія є онтологією верхнього рівня й відрізняється від інших тим, що містить найзагальніший та найрідкісніший концепти, як показано на рис 2.4.

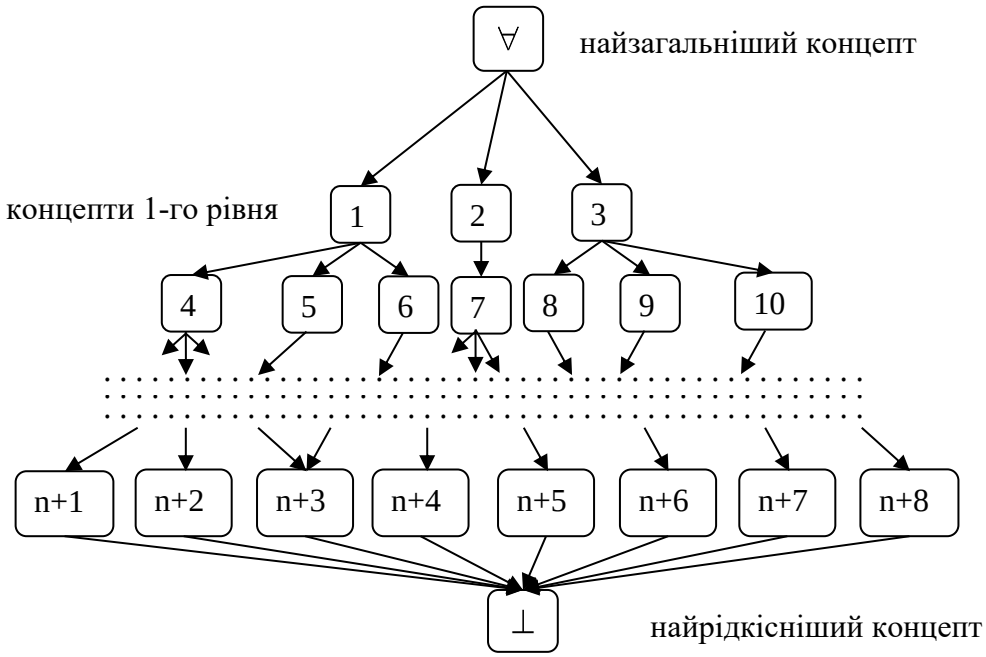

Рисунок 2.4 Загальна модель глобальної (Рисунок створений самостійно)

У роботі онтологія розглядається як структура зі наслідуванням додаткових зв'язків, тобто, якщо вузол 1 має додаткові зв'язки A і B, а вузол 2 є більш

приватним відображенням вузла 1, то зв'язки A і B переносяться на вузол 2 автоматично без явної вказівки цих зв'язків (рис 2.5).

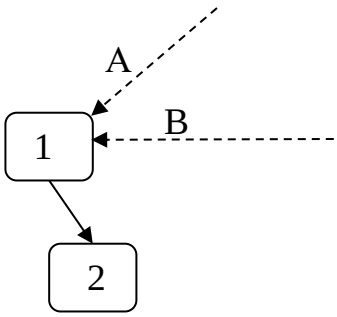

Рисунок 2.5 Організація зв’язків між концептами (Рисунок створений самостійно)

Найважливішим компонентом онтології є вбудовані механізми, що забезпечують можливість виконувати інтелектуальний аналіз інформації. Онтологія складається із класів, а кожний клас містить потенційні можливості для його правильної інтерпретації, одним з таких механізмів є механізм обмежень. Як видно з рисунку 2.4, у глобальній онтології присутні найзагальніший і найрідкісніший концепти, а механізм обмежень дозволяє для кожного класу описати відмінність цього класу від самого загального концепту “thing”.

Обмеження можуть бути різними й накладатися на описи будь-яких властивостей об'єкта: форма, колір, тощо. Якщо об'єкт відповідає цим вимогам, то він є об'єктом цього класу. Такі виразні властивості забезпечують гарну оброблюваність онтологій та надають можливості для автоматичної інтеграції онтологічних баз, навіть у тому випадку, коли однакові класи в різних онтологіях названі зовсім по-різному. Примітивний випадок інтеграції двох онтологій показаний на рисунку 2.6.

Такий підхід дозволяє однозначно ідентифікувати концепти й розвантажити онтологію від зайвих зв'язків. Для виявлення всіх додаткових зв'язків вузла необхідно піднятися по всіх узагальнюючих зв'язках до самого верхнього концепту онтології. Такий підхід ми називаємо *об'єктним підходом* у представленні онтологій.

При проведенні процесу інтеграції онтологій для підтримки вищеописаної структури необхідно після кожного циклу, що інтегрує додаткову онтологічну гілку, проводити перегляд і корекцію результуючої онтології з метою виявлення повторюваних властивостей і зв'язків. Також можлива ситуація, у якій ті самі значення властивостей у різних випадках в онтології будуть називатися абсолютно по-різному. Для вирішення такої ситуації доцільно застосовувати механізм, що дозволяє порівнювати різні зв'язки між собою, використовуючи для ідентифікації зв'язків інтелектуальні механізми зіставлення, засновані на застосуванні додаткової онтології властивостей і зв'язків, у якій явно вказується ідентичність зв'язків.

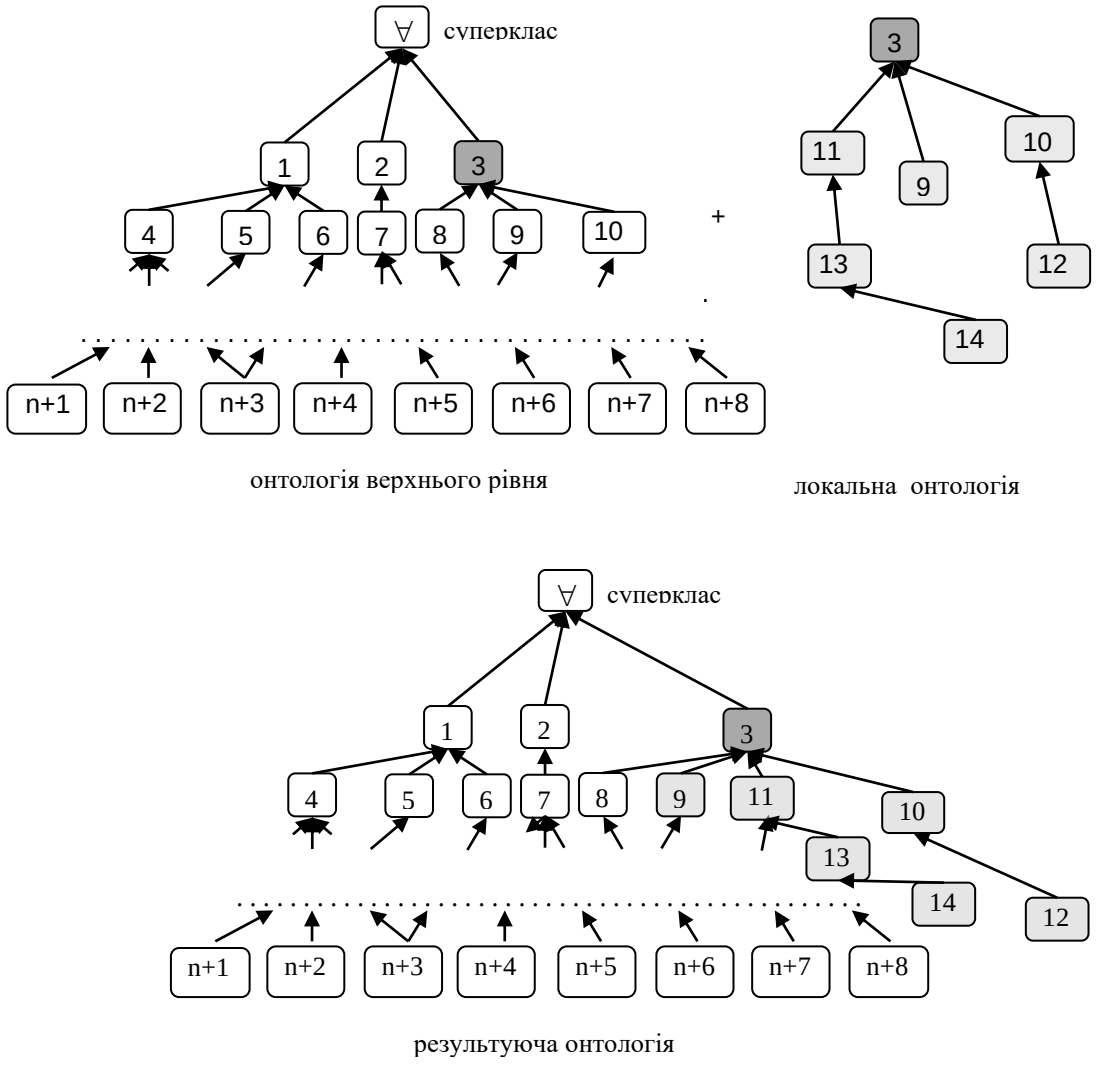

Рисунок 2.6 Інтеграція двох онтологій у єдину онтологічну базу (Рисунок створений самостійно)

Для роботи з онтологіями розроблені математичні операції, основні з яких:

1. ***(+)*** *об'єднання* – бінарна операція, що поєднує дві онтології O_1 і O_2 в O_3. При цьому множина концептів X_1 і X_2 об'єднаються в множина X_3 за правилами теорії множин $X_3 = X_1 \cup X_2$, множина зв'язків S_3 і R_3 формується з S_1, S_2 і R_1, R_2 за спеціальними правилами, описаним нижче. Процес об'єднання виконується в кілька етапів. На першому етапі проводиться пошук ідентичних вузлів (класів) в обох онтологіях (виконується перегляд онтологій, і у випадку знаходження однакових вузлів вони маркіруються внутрішніми спеціальними мітками). На другому етапі виконується об'єднання однакових класів і перенесення зв'язків і вузлів, приєднаних до ідентичних класів і якщо спільних класів знайдено не буде, нова онтологія приєднується як відділений незв'язаний підграф в рамках існуючого загального графу (тобто ми одержуємо 2 незв'язних фрагмента графу). На третьому проводиться аналіз отриманої онтології на наявність помилок і суперечливих ситуацій. Ці ситуації дозволяються або завдяки посиланням на інші онтології, які проводять уточнення інформації, або шляхом видалення компонентів, що викликають конфлікт, з онтології.

Подібний алгоритм є найпростішим і заснований на явних ознаках, що вказують на ідентичність двох вузлів у різних онтологіях. Основним критерієм ідентичності виступає співпадаюча назва.

У якості більш складних методів визначення ідентичності вузлів можливе застосування спеціальної технології, яка могла б дозволити виявляти ідентичні класи у випадку, якщо ці класи не мають однакової назви, але описують ідентичні об'єкти. У такому випадку відсутня явна вказівка на однотипність описуваних об'єктів, але присутні ознаки, що вказують непрямим образом на ідентичність об'єктів. Такий аналіз побудований на застосуванні механізмів порівняння наборів зв'язків. Головним компонентом цієї технології є твердження про те, що: ***якщо значення всіх властивостей декількох класів повністю збігаються, то це той самий об'єкт.***

Використовуючи це твердження можна сформувати наступний математичний апарат інтеграції онтологій:

У відповідності зі стандартами W3C консорціуму й специфікацією OWL, онтологія (O) у роботі розглядається як сукупність 5-ти різних множин: класів (C), властивостей (P), виразів (E), множина екземплярів об'єктів (I) і множини зв'язків між попередніми чотирма множинами (L).

Кожний елемент множини L – це кортеж: $L = \{l_1, l_2, l_3, \ldots\ldots l_n\}$, де $l = \{Type\_Link, Object_1, Object_2\}$, а $Type_{Link} = Tp||Tc||Te$ значення $Object_1, Object_2$ залежать від значення $Type\_Link$ (таблиця 2.1):

Таблиця 2.1 Значення елементів кортежу множини *L*

$Type\_Link$	$Object_1$	$Object_2$	Значення
Tp	c_n	p_m	вказує на приналежність властивості p_m до класу c_n
Tc	c_n	c_m	вказує на те, що c_nє суперкласом c_m
Te	c_n	e_m	вказує на приналежність виразу e_m класу c_n

Для інтеграції онтологій уведені такі операції:

(+) об'єднання – бінарна операція, яка поєднує дві онтології O_1 й O_2 в O_3 ($O_1 + O_2 = O_3$). При цьому множина класів $C(O_1)$ і $C(O_2)$ поєднуються в множина $C(O_3)$ за спеціальними правилами:

$$C(O_3) = C_{\leftrightarrow} \cup\ C_{\neq}(O_1) \cup\ C_{\neq}(O_2), \qquad (2.5)$$

де $C_{\leftrightarrow}(O_1, O_2)$ – множина класів, які мають аналоги в O_1 і O_2;

$C_{\neq}(O_1)$ – множина класів з онтології O_1, які не мають аналогів в онтології O_2;

$C_{\neq}(O_2)$ – множина класів з онтологіїO_2, які не мають аналогів в онтології O_1;

$$C_{\leftrightarrow}(O_1, O_2) = \{c \in C(O_1) \big| \exists j(K(c, c_j(O_2)) = 0)\} \qquad (2.6)$$

$$C_{\neq}(O_1) = \{c \in C(O_1) \big| \forall j(K(c, c_j(O_2)) \neq 0)\} \qquad (2.7)$$

$$C_{\neq}(O_2) = \{c \in C(O_2) \big| \forall j(K(c, c_j(O_1)) \neq 0)\}, \quad (2.8)$$

де $K(c_{*1}, c_{*2})$– це критерій збігу класів, коли він рівняється 0, тоді класи c_{*1}, c_{*2}по всіх параметрах збігаються;

$$K(c_{*1}, c_{*2}) = \text{quant}(P_{\neq}^{c_{*1}, c_{*2}}) + \text{quant}(E_{\neq}^{c_{*1}, c_{*2}}), \quad (2.9)$$

де quant (...) – функція, яка обчислює кількість елементів множини;

$P_{\neq}^{c_{*1}, c_{*2}}$– множина властивостей, зв'язаних тільки з одним із двох класів (c_{*1}, c_{*2}) інтегрувальних онтологій, а $E_{\neq}^{c_{*1}, c_{*2}}$ – множина виразів, які підходять тільки однієї з інтегрувальних онтології;

$$P_{\neq}^{c_*, c_{**}} = \{(p \in P(C_*) || p \in P(C_{**}) \big| \forall i \forall j((p_j(C_*).Name \neq p_i(C_{**}).Name) ||$$
$$||(p_j(C_*).Type \neq p_i(C_{**}).Type)\}, \quad (2.10)$$

де

$p_j(C_*).Name$ – ім'я властивості j класу C_*;

$p_j(C_*).Type$– тип властивості j класу C_*;

$$E_{\neq}^{c_*, c_{**}} = \{(e \in E(C_*) || e \in E(C_{**}) \mid (e.Expression(O_*, C_*) ||$$
$$||e.Expression(O_{**}, C_{**})) \neq false)\}, \quad (2.11)$$

де $e.Expression\ (O_*, C_*)$ – функція, яка перевіряє відповідності виразу e щодо класу C_* й онтології O_*. Множина властивостей формується в такий спосіб:

$$P(O_3) = P(O_1) \cup P(O_2). \quad (2.12)$$

Критерієм однотипності властивостей є наступна ознака:

$$\exists i \exists j((p_j(C_*).Name = p_i(C_{**}).Name) \& (p_j(C_*).Type = p_i(C_{**}).Type)), \quad (2.13)$$

де

$p_j(C_*).Name$– ім'я властивості j класу C_*;

$p_j(C_*).Type$– тип властивості j класу C_*.

Для формування множини виразів, які належать загальній онтології, застосовується наступна формула:

$$E(O_3) = E(O_1) \cup E(O_2), \quad (2.14)$$

де $E(O_1)$ – множина функціональних виразів онтології O_1;

$E(O_2)$– множина функціональних виразів онтології O_2;

$E(O_3)$ – множина функціональних виразів онтології O_3, яка отримана в результаті інтеграції O_1 й O_3;

Множина екземплярів об'єктів також одержують по формулі

$$I(O_3) = I(O_1) \cup I(O_2). \quad (2.15)$$

Критерій збігу екземплярів об'єктів:

$$\exists m \exists n((p_m(I_*).Name = p_n(I_{**}).Name)\&(p_m(I_*).Type = p_n(I_{**}).Type)\&(p_m(I_*).Value = p_n(I_{**}).Value)\&I_*.Type = I_{**}.Type), \quad (2.16)$$

де

$p_m(I_*).Name$ – ім'я властивості m екземпляра об'єкта,

$p_m(I_*).Type$– тип властивості m екземпляра об'єкта,

$p_m(I_*).Value$– значення властивості m екземпляра об'єкта,

$I_*.Type$– тип екземпляра об'єкта (клас-прототип для даного екземпляра).

Множина зв'язків $L(O_3)$ також формується відповідно формулам:

$$L(O_3) = L_c \cup L_k, \quad (2.17)$$

$$L_c = \{l \in L(O_1) || l \in L(O_2) \big| \exists i(l.Object_1 = c^{\neq}{}_i)\&\exists j(l.Object_2 = c^{\neq}{}_j)\}$$
$$L_t = (L(O_1) \cup L(O_2) \cap L_c) \quad (2.18)$$
$$L_k = \{replace(L_t)\}.$$

Функція $replace(\dots)$ виконує заміну компонентів O_1 і O_2 кожного кортежу l множини L_t в такий спосіб: якщо в ролі O_1 або O_2 виступає об'єкт $c(\mathrm{O}_2)$ із множини $\mathrm{C}(\mathrm{O}_2)$ й існує клас c_x, для якого $K(c_x, c(O_2)) = 0$, то в кортежі l виконується заміна $c(\mathrm{O}_2)$ на c_x.

Недоліком такої технології є потенційна можливість помилки, зв'язана з випадками часткового опису властивостей класу. Якщо не всі властивості класу були зазначені при його описі, то виникає потенційна можливість помилки при наявному зіставленні двох концептів онтологій.

2. ***(-)*** *різниця* – бінарна операція, що виділяє різниця між 2-ма онтологіями O_1 і O_2. Результатом цієї операції є онтологія O_3, що містить множину концептів, обумовлених формулою 2.19

$$C(O_3) = C_{\neq}(O_1) \cup C_{\neq}(O_2)\ , \qquad (2.19)$$

де

$C_{\neq}(O_1)$ – множина класів з онтології O_1, які не мають аналогів в онтології O_2;

$C_{\neq}(O_2)$ – множина класів з онтології O_2, які не мають аналогів в онтології O_1;

При цьому множина властивостей формується згідно з формулою: $P(O_3) = P_{\neq}^{c_*, c_{**}}$, а множина виразів згідно формули: $E(O_3) = E_{\neq}^{c_*, c_{**}}$, де $P_{\neq}^{c_{*1}, c_{*2}}$– множина властивостей, зв'язаних тільки з одним із двох класів (c_{*1}, c_{*2}) інтегрувальних онтологій, а $E_{\neq}^{c_{*1}, c_{*2}}$ – множина виразів, які підходять тільки однієї з інтегрувальних онтології.

Виконання операції виявлення різниці засновані на використанні алгоритму виділення з онтології O_1 усіх концептів, що входять в онтологію O_2, і видалення з отриманого підграфу всіх зв'язків, що відносяться до вилучених концептів. Операція призначена для зменшення обсягів онтології й необхідна при побудові окремих онтологій (онтологій завдань) з онтологій, що містять більшу кількість інформації (онтологій предметних областей).

На поточний момент програмні засоби, що застосовуються для обробки онтологій, досить слабкі й мають ряд недоліків, пов'язаних з необхідністю використання більших ресурсних витрат, навіть для обробки невеликих онтологій, а швидкість обробки онтології дуже сильно залежить від кількості концептів (класів), що входять у цю онтологію, тому й видалення зайвих об'єктів із загальної онтології здатне в значній мірі прискорити процес аналізу й обробки онтології.

3. ***()*** *виділення* – операція, що виконує виділення частини онтології O_1. Виділення виконується від концепту, вказаного в операнді операції, зі збереженням усіх зв'язків, властивих концептам більш низьких рівнів у межах частини онтології, що відокремлюється (відсікання зверху). Ця операція дозволяє чітко відокремити

онтології різних рівнів і сприяє процедурі виділення онтології завдання з більших онтологій.

4. ***(∨)*** *стискання* – операція скорочення кількості концептів зі збереженням зв'язків між концептами. У якості операндів передаються: концепт, від якого виконується звуження; рівень, після якого проводиться відсікання та рівень стискання (відсікання знизу зі стисканням). Операція виконує трансформацію онтології до вигляду, в якому зберігаються всі основні концепти опису й видаляються другорядні. При виконанні процесу стискання онтології виконується аналіз, заснований на кількостях зв'язків конкретного концепту. Якщо кількість властивостей або зв'язків, що специфікуються цим об'єктом, менша за певну межу, то буде проводитися перенесення усіх властивостей на об'єкти у зв'язку з успадкування та видаленням концепту з онтології. Такий підхід дозволяє скорочувати обсяг онтології, підвищуючи можливість машинної обробки й знижуючи вимоги до обчислювальної потужності систем, що проводять обробку.

Використовуючи перераховані вище операції, можна змінювати розмір, ступінь деталізації, поєднувати й розділяти онтології.

Операції з онтологіями є послідовністю дій щодо конкретних концептів цих онтологій.

Операція об'єднання виконує трансформацію кінцевої множини S_1 і S_2 в S_3 за допомогою послідовного виконання операції трансформації концептів та їх зв'язків. Для проведення процедури інтеграції онтологій запропонований наступний набір операцій над концептами, що входять в інтегрувальні онтології:

(A) *Add* – об'єднання двох концептів за допомогою додавання відсутніх зв'язків з однієї онтологічної гілки в іншу. *Онтологічною гілкою* називають послідовність концептів, об'єднаних зв'язком узагальнення.

(E) *Expansion* – розширення зв'язку між концептами. Якщо одна з онтологій містить більш великий опис ділянки предметної області, ніж інша, то результуюча онтології буде містити більш докладну ділянку.

(R) *Restore* – встановлює зв'язки узагальнення між концептами в онтологіях, за умови відсутності явної вказівки цих зв'язків.

Операції виконуються доти, поки всі концепти O_1 і O_2 не будуть включені в результуючу онтологію O_3.

Задля правильного об'єднання множин R_1 і R_2 необхідно застосувати до всіх концептів онтології операцію:

(U) *Up* – перенесення всіх додаткових зв'язків на більш загальні концепти, якщо це можливо.

Операція різниці заснована на порівнянні концептів різних онтологій і виявленні концептів, що не входять в одну з онтологій. Множина S_3 відновлюється залежно від концептів, що залишилися.

Для трансформації множин R_1 і R_2 у множині R_3 виконуються операції:

(D) *Down* – перенесення зв'язків від найзагальнішої до найрідкіснішої вершини.

Операції відсікання й звуження онтології виконуються за допомогою використання наступних операцій над концептами:

(Div) *Division* – виконує відсікання найзагальніших концептів (обмеження зверху).

(Dim) *Diminution* – виконує відсікання найрідкісніших концептів (обмеження знизу).

(T) *Taper* – дає зменшення кількості концептів у ланцюжку за рахунок видалення малоінформативних об'єктів.

В результаті застосування описаних операцій ми можемо отримати можливість трансформувати, накопичувати й розділяти знання, що на даний момент є дуже важливим завданням у багатьох інтелектуальних програмах.

Виконання інтеграції онтологій дозволяє на кожному етапі створювати нову онтологічну базу, що забезпечує автоматичну обробку інформаційних джерел. Оскільки онтології виражені у формальному виді, то забезпечується їхня повна автоматична інтеграція.

Запропоновані механізми дозволяють проводити інтеграцію онтологій, використовуючи тільки явні вказівки на відповідність їх різних частин. Але при

проведенні інтеграції можливе застосування більш універсальних механізмів для ідентифікації різних об'єктів, заснованих на порівнянні інтерпретацій об'єктів.

2.3 Механізм побудови запитів до анотаційної та онтологічної інформаційним базам

Рівень логічних правил надає можливість для створення наборів правил, здатних виконувати прості й складні трансформації ресурсної та анотаційної баз. Оскільки набір правил і фрагменти онтологій – це окремі випадки ресурсів, то створюється можливість для непрямої модифікації онтологічної бази й бази правил. Це надає можливості формувати метаправила, після виконання яких змінюється база правил й програма під час функціонування може змінити характер своєї роботи.

Мова побудови логічних правил відіграє важливу роль у структурі, відображеної на рисунку 2.1, а враховуючи можливість застосування механізму трансформації внутрішнього представлення документів у звичайний вигляд, стає очевидно, що за допомогою мови логічного рівня, стає можливим: (a) виконувати інтелектуальну обробку документів, (b) на основі шаблонів формувати нові документи, що містять дані з анотаційної бази, (c) виконувати навчання й самонавчання системи.

У якості мови для вибору інформації з анотаційної та онтологічної баз використовується універсальна мова SQL із структурою бази даних на основі графу [30].

Подібна структура дозволяє формувати спеціальні запити, які після виконання повертають набори даних. Загальна структура подібних запитів відображена на рисунку 2.7.

SELECT
список змінних та значень, що мають бути повернені

FROM
елемент графу чи крайнє значення графу

WHERE
граф-шаблон, визначений списком предикатів триплетів

AND
логічний вираз щодо URI та літералами
(арифметичне порівняння, диз'юнкція, кон'юнкція, заперечення, тощо)

USING
список namespace - префіксів.

Рисунок 2.7 Загальна структура SQL запиту (Рисунок виконаний самостійно)

Після виконання такого запиту можливо отримати інформація, яку згодом можна обробити, модифікувати й результат знову зберегти в інформаційних базах. Інформація, що може міститься в подібних графах може бути досить громіздкою, в той час як зазначена структура бази дозволяє легко отримати окрему частину із цього графа. Це не інтелектуальне завдання, але від ефективності її рішення залежить ефективність роботи всієї системи.

2.4 Механізм виводу нової інформації й нових знань на основі мови рівня логічних правил RuleML

Головним компонентом логічного рівня є технологія, що дозволяє проводити інтелектуальний вивід інформації – мова RuleML. На рисунку 2.8 зображені всі компоненти логічного рівня. Найкориснішими з них є правила, що дозволяють проводити вивід інформації (підгрупа правил виводу). Ця підгрупа правил у свою чергу ще ділиться на SQL-подібні й Prolog-подібні правила.

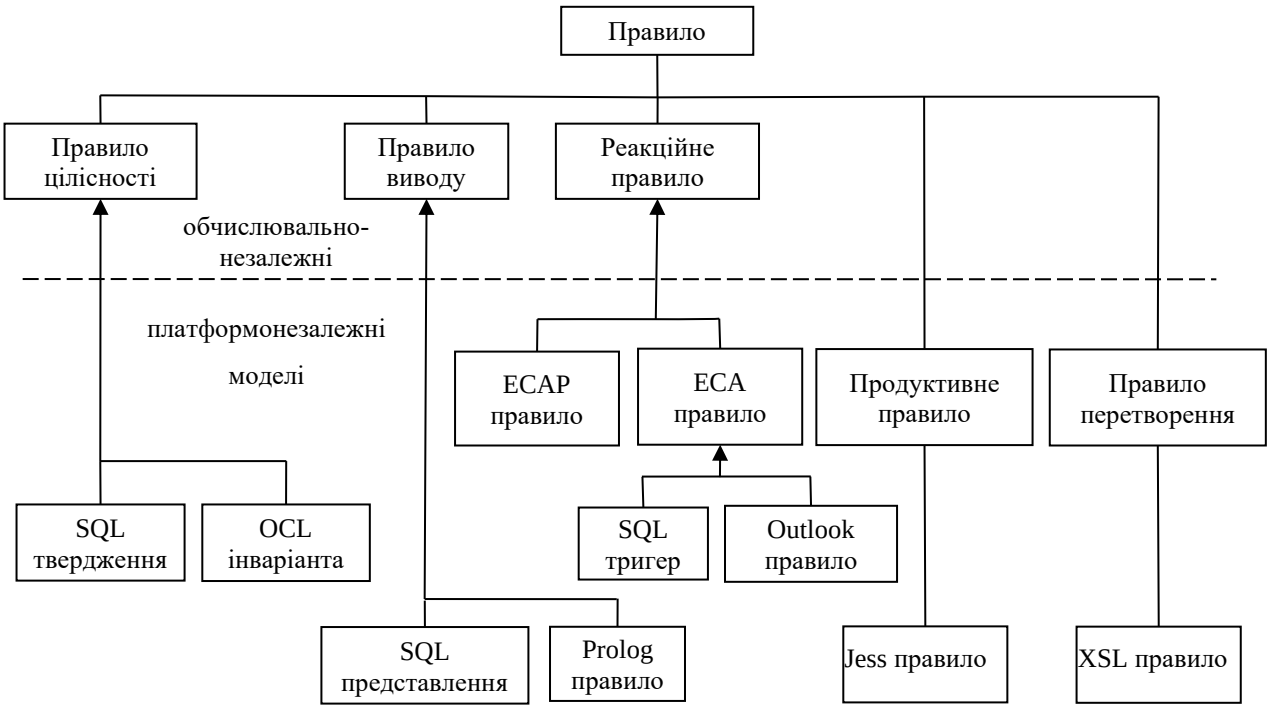

Рисунок 2.8 Схема класифікації технологій для рівня логічних правил RuleML (Рисунок виконаний самостійно)

Розроблена мова входить до підмножини «прологоподібних» мов, основне призначення якої – забезпечення можливості написання правил для обробки інформаційного вмісту анотаційної та онтологічної баз. У створенні універсальної мови для побудови логічних правил RuleML були використані наробітки консорціуму W3C. Основу розробленої мови становить XML синтаксис, логічна

«прологоподібна» модель, представлена далі, та основи, закладені при формуванні мов RDF, RDF Schema, OWL.

Сформована мова складається з аксіоми або набору аксіом, які споконвічно вважаються вірними, і складаються з наборів «прологоподібних» правил (формула 2.20).

$$axiom ::= rule \tag{2.20}$$

Кожне з аксіоматичне правило у свою чергу складається зі зв'язки **антецедент-консеквент**. Антецедент виражає набори логічних умов, а консеквент – результуюче знання, отримане, якщо всі логічні умови вірні. Кожний антецедент і консеквент складаються з наборів **атомів** (формула 2.21).

$$\begin{aligned} rule &::= 'Implies('\ antecedent\ consequent\ ')' \\ antecedent &::= 'Antecedent('\ \{\ atom\ \}\ ')' \\ consequent &::= 'Consequent('\ \{\ atom\ \}\ ')' \end{aligned} \tag{2.21}$$

Атом є базовою структурою при побудові виразу. Він може бути представлений більшою кількістю різних компонентів (формула 2.22).

$$\begin{aligned} atom ::= &\ description\ '('\ i-object\ ')' \\ &| individualvaluedPropertyID\ '('\ i-object\ i-object\ ')' \\ &| datavaluedPropertyID\ '('\ i-object\ d-object\ ')' \\ &| sameAs\ '('\ i-object\ i-object\ ')' \\ &| differentFrom\ '('\ i-object\ i-object\ ')' \end{aligned} \tag{2.22}$$

Атом може складатися із ID(x), P(x,y), sameas(x,y) або differentfrom(x,y), де ID – це OWL клас, а P – це OWL властивість, і x, y – це інші вільні змінні, OWL екземпляри об'єктів або OWL значення змінних. У якості об'єктів можуть виступати змінні або посилання на об'єкти двох типів: конкретні екземпляри об'єктів класів (*individual*) і інформаційні константи (*literal*) (формула 2.23).

$$i-object ::= i-variable | individualID$$
$$d-object ::= d-variable | dataLiteral \quad (2.23)$$

Кожна зі змінних має своє унікальне ім'я, виражене за допомогою URI (формула 2.24).

$$i-variable ::= 'I-variable('URIreference')'$$
$$d-variable ::= 'D-variable('URIreference')' \quad (2.24)$$

Наведена модель побудови правил повністю забезпечує всі необхідні механізми для представлення знань а метаданих у рамках моделі, описаної в розділі 1.2.2, і, як наслідок, сумісна з моделлю, розробленої в розділі 2.1.

Для забезпечення можливості машинної обробки наборів правил відповідно до наведеної моделі (формули 2.20-2.24) у рамках XML розроблено представлення компонентів інформаційної моделі, що в свою чергу забезпечує спрощену обробку наборів правил і стандартизацію наданих даних.

Для забезпечення стикування з моделлю, представленої в розділі 2.1, створено додаткові XML-теги для мови RuleML:

Елемент Онтологія - дозволяє описати фрагмент онтології й включити інформацію в онтологічну базу явно та може містити в собі усі компоненти, властиві мові OWL. Фрагмент онтології виражається наступною XML-конструкцією:

$$< OntologyName = xsd: anyURI > content </OntologyName >,$$

де значення атрибута *name* задає унікальний ідентифікатор для однозначної ідентифікації фрагмента онтології, а вміст онтології складається з наборів анотацій, класів, підкласів, списків властивостей, значень властивостей, правил тощо.

$$content ::= (Imports | Annotation | Class[axiom] | EnumeratedClass(D,F) | SubClassOf(D,F)$$
$$| EquivalentClasses | DisjointClasses(D,F) | DatatypeProperty | ObjectProperty$$
$$| SubPropertyOf | EquivalentProperties | Individual[axiom] | SameIndividual$$
$$| DifferentIndividuals | Rule[axiom] | Variable[axiom])$$

Елемент Variable [аксіома]: дозволяє описати змінні, що містять аксіоми, які використовуються при побудові правил

$$< Variable\ \ name\ =\ xsd{:}\ anyURI > content\ </Variable >,$$

де значення атрибута *name* задає унікальний ідентифікатор для змінної, а *content* містить анотацію змінної. Батьківським елементом для змінної є онтологія.

Елемент Rule [аксіома]: дозволяє сформувати правила логічного рівня для модифікації онтологічної та анотаційної баз.

$$< Rule > Content{:}\ (\ antecedent, consequent\ ) </Rule >$$

У якості вмісту правил використовуються зв'язки антецедент і консеквент. Батьківським елементом для правила є онтологія.

Антецедент виражає набори тверджень, при виконанні яких основна частина висловлення, включена в **консеквент**. Кожний з компонентів представлений з використанням RuleML у такий спосіб:

$$< antecedent >\ \ Content{:}\ (\ atom*\ ) </antecedent >$$

$$< consequent >\ \ Content{:}\ (\ atom*\ ) </consequent >$$

Компонентами антецедента й консеквента є **атоми**, які у свою чергу можуть складатися з унарних предикатів (класів), бінарних предикатів (властивостей), рівностей або нерівностей.

$$atom{:}{:}=\ (classAtom \mid individualPropertyAtom \mid datavaluedPropertyAtom \mid sameIndividualAtom \mid differentIndividualsAtom)$$

Батьківськими компонентами для атомів є консеквент і антецедент.

Атомарний клас може містити описи різних об'єктів.

$$< classAtom >\ \ Content{:}\ (\ description, iObject\ ) </classAtom >$$

Батьківським компонентом для атомарного класу виступає атом. Атоми можуть містити елементи **individualPropertyAtom** і **datavaluedPropertyAtom**.

За допомогою цих компонентів можна виразити назву екземпляра класу, назва змінних, значення змінних.

Для **individualPropertyAtom**

$$< individualPropertyAtom\ property\ =\ xsd{:}anyURI\ >$$
$$Content{:}\ (\ iObject, iObject\ )$$
$$</individualPropertyAtom>$$

Для **datavaluedPropertyAtom**

$$< datavaluedPropertyAtom\ property\ =\ xsd{:}anyURI\ >$$
$$Content{:}\ (\ iObject, dObject\ )$$
$$</datavaluedPropertyAtom>$$

де $property$ – посилання на назву властивості

Елемент **sameIndividualAtom** вказує на ідентичність об'єктів.

$$< sameIndividualAtom > Content{:}\ (\ iObject*\ ) </sameIndividualAtom >$$

Елемент **differentIndividualsAtom** вказує на відмінність між об'єктами

$$< differentIndividualsAtom >$$
$$Content{:}\ (\ iObject*\ )$$
$$</differentIndividualsAtom >$$

Основна група об'єктів містить об'єкти 2-х типів: **i-object** і **d-object,** що відрізняються тим, що i-object містить посилання на конкретний об'єкт, реалізацію його або класу, а об'єкт d-object містить конкретні інформаційні значення.

$$i-Object ::=\ \text{(Individual[ID] | Variable[ID])}$$
$$d-Object ::=\ \text{(DataValue | Variable[ID])}$$

Елемент **Variable[ID]** дозволяє задати посилання на змінну.

$$< Variable\ name\ =\ xsd{:}anyURI\ >\ Content{:}\ (\#\#empty) </Variable >$$

Наведений вище набір тегів дозволяє закодувати правила в рамках інформаційної моделі, представленої формулами (2.20) – (2.25). Такий підхід надає

універсальні засоби для вираження й опису правил, знань та метаправил для наступної інтелектуальної обробки інформації.

2.5 Механізм представлення семантичних запитів користувачів до масиву анотованих електронних документів і візуалізації результатів

Інформація, представлена в анотаційній інформаційній базі має «машинно-зрозумілий» вигляд, проте повністю не зрозуміла для користувача. Задля перетворення цієї інформації у вигляд, призначений для перегляду людиною, розроблена спеціальна технологія, основою якої є використання шаблонів вихідних даних. Ці шаблони містять інформацію щодо форматування вихідного документа, наборів правил для виділення інформації з онтологічної й анотаційної баз і застосування мови запитів SQL для формування запитів до анотаційної і онтологічної базам.

Для представлення інформації в шаблону обраний універсальний формат rtf, бо він дозволяє представити статичну інформацію і її повне форматування. В той час, як для відображення динамічного компонента шаблонів необхідно зробити трансформацію rtf. Для впровадження динамічного компонента і його опису використовується стандарт XML. Для виконання цього завдання у роботі була проведена розробка нових XML-тегів:

$$< Pattern\ name\ =\ xsd{:}\ anyURI\ >\ content\ </Pattern>$$

де $name$ – це ім'я й унікальний ідентифікатор шаблону, а $content$– виражає весь вміст документа шаблону.

Формований шаблон розбивається на три частині.

У першій частині проводиться опис змінних, які будуть використовуватися в документі. Перша секція виділяється наступним XML описом:

$$< Variables >$$
$$< Variable\ name = "xsd{:}\ anyURI"/>$$
$$</Variables >$$

де секція опису змінних виділяється тегом *Variables*, а змінні, які описуються в межах цієї секції виділяються тегом *Variable* з атрибутом $name$ (ім'я й унікальний ідентифікатор змінної).

Змінні не мають типу й у цій секції проводиться тільки вказівка на наявність змінної з відповідним ім’ям.

У другій секції проводиться ініціалізація змінних значеннями з анотаційної та онтологічної баз, на основі спеціальних запитів, побудованих з використанням мови SQL.

$$
\begin{aligned}
&< SQL\_section > \\
&\ < Variable\_Int\ name = "xsd: anyURI" > \\
&\quad SQLQuery \\
&\ </Variable\_Int > \\
&</SQL\_section >
\end{aligned}
$$

де тег *SQL_section* виділяє секцію ініціалізації змінних, тег *Variable_Int* з атрибутом name визначає змінну, що ініціалізується та фрагмент *SQLQuery* визначає семантичний запит для кожної змінної моделі, що ініціалізується.

Третя секція являє собою документ, що містить статичні елементи й елементи форматування, написані в стандарті rtf. Для зв'язку з інформаційними базами застосовується спеціальна вставка, яка дозволяє додавати значення змінної, що ініціалізується в будь-якому місці форматованого тексту.

$$
\begin{aligned}
&< Document > \\
&\ < rtf\_sec\,t\,ion > \\
&\quad < value\_of\_variablename = "xsd: anyURL"/> \\
&\ </rtf\_sec\,t\,ion > \\
&</Document >
\end{aligned}
$$

де тег *Document* виділяє третю секцію шаблону документа, тег *rtf_section* виділяє фрагмент тексту, розміченого з використанням технології rtf [RTF_Document], а тег *value_of_variable* з атрибутом name вказує на вставку значення змінної в статичний текстовий документ.

Використовуючи вищеописану мову, отримана можливість формування шаблонів документів для динамічного заповнення їх інформацією з онтологічної й анотаційних баз. Шаблони, що використовуються в системі, представляються як окремий випадок ресурсів і можуть бути модифіковані на основі логічних правил, описаних відповідно до синтаксису, розробленого в розділі 2.4.

Розроблені методи представлення інформації в «машинно-зрозумілому» виді, інтеграції онтологій, представлення логічних правил для вираження знань і метаданих, теги для формування запитів до анотаційної і онтологічної інформаційним базам і розробка механізму побудови семантичних запитів користувачів до масиву анотованих електронних документів і візуалізації результатів, виконані в цій роботі, і розробки консорціуму W3C дозволили зробити комплексну розробку інтелектуальної системи менеджменту й інтеграції різнорідної інформації на основі стандартизованих моделей знань.

Висновки розділу

Другий розділ посвячено розробці набору методів для розширення можливостей технології Semantic Web, доповнюючи її необхідними інструментами для створення інтелектуальної системи менеджменту й інтеграції інформації.

Розроблені методи оцінки якості представлення інформації у відповідності зі стандартом Semantic Web для такої системи як параметри оцінки такої інформації. Детально розглянутий онтологічний підхід до інформаційних джерел, механізму інтеграції онтологій, заснованому на використанні формальних описів класів об'єктів, специфікованою мовою OWL для забезпечення ефективної інтеграції й менеджменту інформації.

Подальший розвиток одержав метод трансформації онтологій, які засновані на використанні стандартів Semantic Web. Завдяки використанню цього методу стає можливим об'єднання різнорідних концептуальних моделей.

Для забезпечення можливості створення програм/сценаріїв, за допомогою яких було б можливо аналізувати й модифікувати анотаційну та онтологічні бази, були розроблено два методи:

1) метод заснований на принципах мови SQL

2) заснований на принципах «прологоподібних» мов.

Перший метод призначений для формування запита для одержання інформації з анотаційної та онтологічної баз. Другий метод дозволяє формувати «прологоподібні» правила виводу. Цей метод базується на використанні спеціальної моделі, розробленої на основі RuleML.

Таким чином, у другій частині розроблені теоретичні основи для створення інтелектуальної системи інтеграції й менеджменту інформації на основі стандартизованих моделей знань.

3 АРХІТЕКТУРА ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ МЕНЕДЖМЕНТУ Й ІНТЕГРАЦІЇ РІЗНОРІДНОЇ ІНФОРМАЦІЇ НА ОСНОВІ СТАНДАРТИЗОВАНИХ МОДЕЛЕЙ ЗНАНЬ

Враховуючи напрямок розвитку вищої освіти України, та інтеграція з європейською системою освіти, розробка інтелектуальної системи інтеграції освітніх стандартів є не тільки актуальною, але й вкрай важливою для всієї системи вищої освіти України. Перехід до автоматичної обробки документів стандарту вищої освіти України дозволить забезпечити легку модифікованість та переносимість інформації описаної в цих стандартах. Така система дозволить робити порівняння й інтеграцію інформації зі стандартів України в загальноєвропейські стандарти вищої освіти.

Це необхідно при порівнянні спеціальностей, навчальних планів і освітніх проектів з метою визначення якісного рівня й можливостей перетворення компонентів українського стандарту в аналогічні йому європейські документи.

Враховуючи специфіку стандартів вищої освіти України, така система повинна мати високий ступінь інтелектуальності й мати можливість розбирати й аналізувати природньо-мовну інформацію. Це досить складно й зв’язано з використанням контекстно-залежної обробки. Оскільки природньо-мовна інформація складно формалізована, розглядається підхід, в межах якого здійснюється перехід інформації від природньо-мовної форми до формалізованої. Цей перехід виконується за допомогою напівавтоматичних систем з обов'язковою участю людини.

3.1 Розробка основних принципів системи інтеграції інформації на основі стандартизованих моделей знань

Вирішуючи завдання побудови інтелектуальної системи інтеграції інформації на основі стандартизованих моделей знань, доводиться вирішувати цілу низьку питань, пов'язаних з організацією обробки й аналізу інформації:

а) Який вибрати формат для представлення різнотипних семантичних даних (передбачається, що до існуючого інформаційного джерела додається додатковий файл, що містить опис його семантичного наповнення)?

б) Як забезпечити взаємодію описаних інформаційних ресурсів?

в) Яку мову програмування вибрати для опису предметної області або групи областей?

г) Як розробити механізми порівняння погоджених семантичних описів, та механізми пошуку документів на їхній основі?

д) Як забезпечити можливість представлення інформації користувачам у зручному для наступної обробки вигляді?

е) Яким чином забезпечити заповнення інформаційних баз?

Першим етапом вирішення поставлених задач є правильний **вибір механізмів представлення інформаційних ресурсів**. Враховуючи специфіку інформації й складність вимог, яким повинна відповідати система, а саме: можливість контекстної обробки інформації, можливість інтелектуального аналізу інформаційного змісту й можливість повторного використання інформації; доцільно застосувати модель представлення інформації, запропоновану в пункті 2.1, трансформувавши її для рішення завдань інтеграції інформації в сфері освіти. Під трансформацією розуміється чітка специфікація документів кожного окремого типу.

Першому рівню представлення інформації відповідають інформаційні **ресурси**: стандарти вищої освіти України, документи, що відносяться до освітнього

процесу, виражені в будь-якому форматі даних (текстова, інтерактивна, аудіо та відео інформація). Обмеження щодо формату ресурсів, що оброблюються у системі, не накладається. Таким чином, усі раніше використані ресурси для організації документообігу повністю можуть бути оброблені й в інтелектуальній системі інтеграції інформації.

Документи другого рівня – це **анотації,** створені із застосуванням технології RDF, що й описують усі інформаційні ресурси, що включаються в систему. Анотації служать сполучною ланкою між ресурсами, вираженими в будь-якому форматі, і механізмами, що проводять інтелектуальну обробку інформаційного змісту. RDF анотації формуються так, щоб терміни, якими виконуються описи ресурсів, обов'язковим чином включалися в інформаційну базу більш високого рівня – до **онтології** напряму. Онтологія в межах пропонованої концепції відрізняється тільки своєю спеціалізацією й призначена для опису об'єктів науково-освітнього процесу. **Документи рівня логічних правил** являють собою спеціальні інформаційні ресурси, побудовані на основі правил, описаних у розділі 2.4. Сформована модель дозволяє організувати автоматизовану обробку інформаційного змісту.

Така модель організації представлення інформації, що доповнена механізмами інтеграції онтологій, описаними в розділі 2.2, дозволяє розв'язати **питання здатності до взаємодії** описів інформаційних ресурсів зазначеним нижче чином. Усі ресурси описані за допомогою спеціальних анотацій, створених з використанням універсальної мови опису інформаційних ресурсів RDF. При побудові кожного RDF використовуються тільки поняття, що входять в онтологію завдання або предметної області, яка використовується в рамках концептуальної моделі, у якій створюється ця анотація.

Для перевірки можливості машинної обробки анотацій використовуються критерії, описані в розділі 2.1. Кожна з анотацій, а, отже, і наповнення кожного з ресурсів описані «машинно-зрозуміло» однією і тою самою системою. Проте в межах кожної інформаційної системи використовується своя онтологія, й ресурси, описані в одній системі, без модифікації онтології не будуть оброблятися в іншій.

Тому для забезпечення здатності до взаємодії використовуються методи модифікації або інтеграції онтологій, описані в розділі 2.2.

Питання вибору мови опису предметної області автоматично вирішується запропонованою моделлю представлення даних - мови OWL. Вона надає всі необхідні механізми для інтелектуальної обробки інформаційного змісту анотаційних ресурсів і дозволяє виконувати інтеграцію різних онтологій, описану в розділі 2.2, тим самим забезпечуючи можливість розширення й модифікації концептуальної моделі, в межах якої проводиться аналіз і обробка анотаційних описів які були використані в системі.

Питання стосовно аналізу узгоджених семантичних описів вирішується шляхом написання наборів управляючих правил, виражених відповідно до моделей, описаних у пункті 2.4. Для забезпечення порівняння й аналізу узгоджених описів виконується інтеграція інформації з декількох інформаційних джерел: поєднуються ресурси, вносяться нові анотації цих ресурсів у загальні анотаційні бази, формується загальний інформаційний простір шляхом інтеграції онтологій. Далі до загального інформаційного змісту застосовуються набори правил для порівняння й аналізу інформації. Застосування унікального ідентифікатора URI забезпечує можливість точного визначення первісного джерела інформації й завжди існує можливість простежити ланцюжок логічних висновків і перетворень, вироблених у системі інтеграції й обробки інформації.

Оскільки інформація в системі виражається в зручній для машинної обробки формі, виникає **завдання перетворення інформації у вигляд, зрозумілий для людини**. Для рішення цього завдання створюється спеціальний тип ресурсів, що дозволяє виражати шаблони документів і запити користувачів до інформаційної бази, технологія формування запитів користувачів описано в розділі 2.3, а методи складання, опису й представлення шаблонів розглянуто в розділі 2.5.

Важливим питанням при формуванні інформаційної системи є **питання щодо наповнення інформаційних баз**. Для заповнення *ресурсного рівня* досить перетворення інформації щодо документообігу в області освітнього й наукового процесу в електронний вигляд. Враховуючи, що зараз організація документообігу

тією чи іншою мірою заснована на застосуванні комп'ютерів, і вся інформація стосовно документів так чи інакше виражена в електронному вигляді у формі, стандартизованої для комп'ютерних систем, то заповнення інформацією першого рівня не викликає певних ускладнень і може бути виконана в автоматичному режимі. Питання наповнення інформаційної бази даними *анотацій* виглядає складніше. Анотації (на поточному етапі розвитку комп'ютерних інформаційних технології) можуть бути сформовані тільки неавтоматичним чином із залученням експерта в предметній області. Передбачається, що знання й інформація у формалізованому виді щодо документів системи вищої освіти будуть зберігатися у вигляді анотацій, при цьому вони будуть містити весь обсяг інформації, що міститься в оригіналі документів. Формувати ці анотації повинні будуть автори документів або експерти за допомогою спеціальної програми, у яку вони зможуть вводити інформацію, що міститься природньою мовою в оригіналі документа, ґрунтуючись на концепції опису інформації у формі триплету «об'єкт-атрибут-значення». Результатом роботи цієї системи буде файл анотації в машинно-зрозумілій формі.

Тим не менш, для машини стандартизованим і розпізнаваним є тільки структура описаної інформації, а внутрішній зміст описується в термінах складової анотації людиною. Для обробки цього змісту, аналізу або порівняння з іншою аналогічною анотацією потрібен додатковий механізм, що містить інформацію щодо предметної області, завдяки чому з'являється можливість зіставлення двох різноманітних анотацій, написаних різними людьми. Важливою вимогою до такого механізму є наявність у ньому всіх компонентів, що перебувають у різних анотаціях. Таким універсальним механізмом може виступати *онтологія*.

Онтологія містить набір об'єктів, що описують предметну область й два набори зв'язків. Перший тип зв'язків аналогічний триплетній моделі анотацій, що містить зв'язку «об'єкт-атрибут-значення». Другий тип зв'язків нехарактерний для анотації, бо він містить набір зв'язків, що характеризують об'єкти й класи, завдяки існуючому набору обмежень і уточнень, які накладаються на існуючі класи. Цей механізм дозволяє зрівняти кілька різних анотацій, що містять синоніми та можуть

бути написаних на різних мовах, за умови, що онтологія буде містити всі або більшу частину об'єктів, описаних у цих анотаційних джерелах.

На онтологію, що використовується в системі, накладається ряд вимог, що забезпечують повноту й правильність обробки інформації. Найважливішою подібною вимогою є наявність в онтології всіх об'єктів і зв'язків, що містяться в анотаціях. Також є важливим контекст, у якому об'єкт або клас описаний в онтології. Контроль виконання першої умови є досить простим, і перебування всіх об'єктів в онтології свідчить про відсутність помилок цього класу. Помилки другого класу діагностуються складніше, і механізм їх виявлення вимагає застосування інтелектуальної функції. Методи й критерії оцінювання якості опису онтологій і анотацій розглянуто в розділі 2.1.

Спочатку онтології створюються й вносяться в систему неавтоматичним чином - експертами в предметній області за допомогою спеціальної програми, що забезпечує всі необхідні інструменти для розробки онтологій. На наступному етапі онтології розширюються й доповнюються новими класами при складанні анотацій, що забезпечує можливість опису абсолютно будь-яких ресурсів, а третім етапом є повністю автоматична інтеграція онтологій, створених різними людьми й застосовуваних у різних системах.

Для *рівня логічних правил* технологія їх представлення розроблена описана в розділі 2.4. Формування документу, що містить логічні правила, здійснюється за допомогою спеціальної програми й усі правила, що функціонують у системі, виражаються в спеціальному «прологоподібному» виді (правила формуються експертами).

Кожний з вищеописаних документів відповідно до моделі рисунку 2.1 можна представити як інформаційний ресурс, і така організація відкриває широкі можливості для автоматичного навчання й автоматичного самовдосконалення інтелектуальної інформаційної системи, побудованої на вищевикладених принципах.

3.2 Розробка загальної архітектури інтелектуальної системи інтеграції інформації на основі стандартизованих моделей знань

Архітектура системи інтеграції інформації заснована на принципах, викладених у попередньому розділі. З попереднього розділу в інформаційній системі фігурують наступні важливі типи документів: ресурс, анотація, онтологія документи, що управляють чи містять набори правил для логічного виводу, шаблони документів для перетворення «машинно-зрозумілої» інформації до звичайного вигляду. Як видно з визначення ресурсу (розділ 2.1), будь-який електронний документ є ресурсом, а, отже, анотації, онтології, набори правил, шаблони можуть бути представлені як інформаційні ресурси. Для забезпечення ефективної роботи в системі сформовано два типи баз даних: інформаційні бази, що містять усю вхідну інформацію поділену на 3 групи: ресурси, анотації, онтології; та операційні бази - містять усі компонент інформаційної моделі, описаної в розділі 2.1, які можуть динамічно трансформуватися системою. Виділення онтологій і анотацій в окремі інформаційні бази викликане необхідністю спростити заповнення інформаційних баз і спростити механізми взаємодії із зовнішніми комплексами.

Інформаційні бази сховищ ресурсів – бази даних, де будь-яка представлена інформація спочатку копіюється в операційні бази й лише там трансформується. Такий підхід дозволяє створити копії інформаційних ресурсів і в будь-який момент провести аналіз і перезапуск інформаційної системи. У систему також внесений ще один вид ресурсів, призначений надати можливості для гнучкого розвитку системи аналізу, обробки інформації й одержання нових знань. Це plug-in модулі, що дозволяють здійснити динамічне розширення бази елементарних функцій і атомів, які використовуються при побудові «прологоподібних» правил. Plug-ins – додаткові модулі, що виконуються при підключенні та керуються системою.

Інформаційні бази складаються із трьох компонентів: база анотацій, де анотації зберігаються в оригінальному незміненому вигляді, база онтологій, де

зберігаються фрагменти інтегрувальних онтологій, і база інформаційних ресурсів, де зберігаються всі інші ресурси.

Операційні бази містять чотири компоненти: база нормованих анотацій, основна онтологія системи, база управляючих ресурсів і база шаблонів. Архітектура системи представлена на рисунку 3.1.

Анотаційна операційна база містить усі анотації в спеціальному нормалізованому виді, процес нормалізації анотацій буде описаний надалі. У цю базу після обробки переносяться всі анотації з інформаційної бази. Кожній анотації в інформаційній базі точно відповідає аналогічна їй нормалізована анотація. Процес нормалізації призначений для зменшення ступеня перевантаження анотаційної бази різними концептуальними описами, що виражають ідентичні концепти.

Онтологічна операційна база містить тільки одну динамічно розширювану онтологію. Ця онтологія заповнюється первісною інформацією, що описує необхідні класи для проведення інтелектуального аналізу інформації. Надалі компоненти або цілі онтології доповнюють інформаційну базу. Вони створюються разом з описами ресурсів або переносяться з інших систем. Результатом їх роботи є цілісна загальна онтологія. При формуванні загальної онтології можливі випадки виникнення помилкових ситуацій. Для рішення їх застосовується кілька механізмів: перевірка й уточнення інформації в загальній онтологічній інформаційній довіреній базі й встановлення пріоритетів довіри до різних баз і інформування експерта щодо необхідності внесення корекції в онтологічну базу. Принципи, що застосовуються підчас інтеграції онтологій, описано в пункті 2.2.

База управляючих ресурсів містить інформацію стосовно усіх plug-in, внесених у систему та усі набори правил, призначених для одержання нових знань. Універсальні модулі plug-in і набори правил виділяються з ресурсної бази завдяки спеціальній інформації в анотаціях до цих ресурсів.

Після їхнього виділення вони копіюються в базу керуючих ресурсів і починають брати активну участь в обчислювальних процесах інформації, що виконуються в інформаційній системі інтеграції.

База шаблонів документів призначена тільки для зберігання інформації стосовно вихідних шаблонів документів. Кожний шаблон має свою ресурсну анотацію, у якій зазначені параметри вибору умови застосування кожного конкретного шаблону.

Також до складу системи інтеграції інформації входить **система менеджменту інформаційних і ресурсних баз**. У систему менеджменту входять три компоненти (рисунок 3.1): загальна система менеджменту, підсистема одержання нових знань і модуль ядра системи.

Ядро системи складається з декількох модулів, першим з яких є модуль базових функцій - містить усі необхідні функції для роботи системи, у тому числі і функції, що забезпечують первісне завантаження анотаційних баз. Функції цього модуля забезпечують надійну автономну роботу інформаційної системи, перевірку вмісту інформаційних ресурсів і оцінювання несуперечності даних, що надходять. Цей модуль є повністю статичним і з високим ступенем надійності та забезпечує нормальну працездатність системи в будь-якій ситуації.

Другий компонентом ядра системи - набір plug-in модулів, які призначені для динамічного вдосконалювання системи за рахунок включення в неї нових функцій і нових компонентів. Архітектура системи є відкритою, що дозволяє без перебудови основної програми робити модифікації та вдосконалення внутрішніх функцій і механізмів програми. Тим більше, що запропонована модель написання «прологоподібних» правил, описана в розділі 2.4, є тільки частиною технології побудови правил логічного рівня (рис. 2.8). Набори plug-in модулів дозволяють розширювати й модифікувати систему інтеграції інформації й одержання нових знань на основі логічних правил.

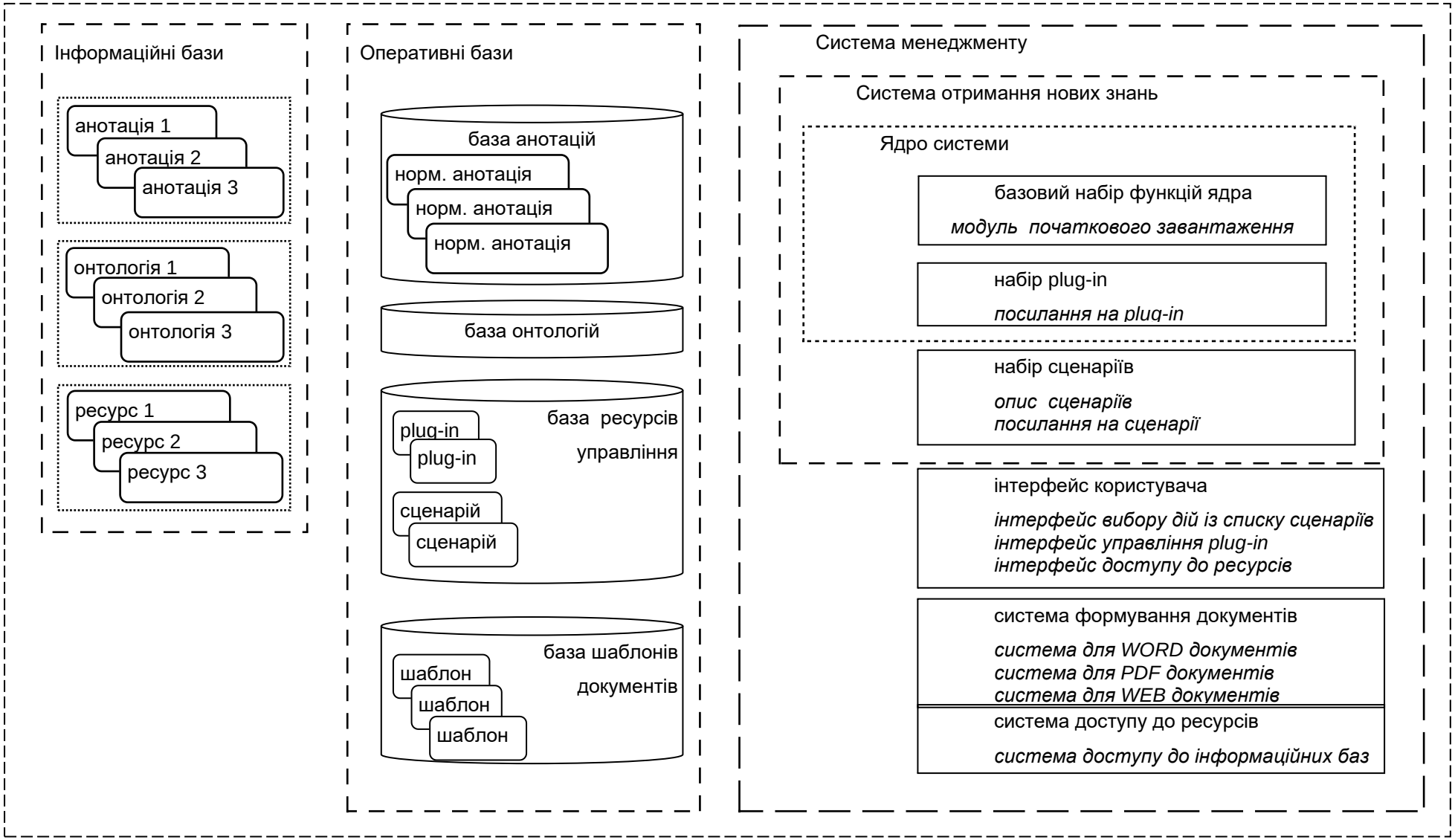

Рисунок 3.1 Архітектура системи інтеграції інформації на основі стандартизованих моделей знань

(Рисунок виконаний самостійно)

Система одержання нових знань повністю містить у собі ядро системи й додатковий модуль логічних правил, в складі останнього набори сценаріїв і їх описання, отримані шляхом обробки інформаційних анотацій. Сценарії отримуються шляхом виділення інформації з управляючих ресурсів, що містить набори управляючих правил. Запуск правил для формування нової інформації здійснюється тільки при виборі людиною відповідного скрипту, що запускає той або інший набір правил.

Для взаємодії з користувачем передбачені компоненти системи менеджменту інформаційних ресурсів: інтерфейс взаємодії з користувачем та система формування документів та система доступу до ресурсів, що входить до модулю менеджменту. Модуль одержання нових знань також повністю входить до системи менеджменту інформаційних ресурсів (рисунок 3.1).

Модуль інтерфейсу взаємодії з користувачем призначений для управління програмним комплексом, ресурсами та механізмами підключення додаткових модулів. Модуль інтерфейсу взаємодії складається із трьох підсистем: інтерфейс вибору дій зі списку сценаріїв, призначений для надання користувачеві списку доступних дій для модифікації й аналізу інформації, що перебуває в анотаційних базах, та генерації нових даних та знань; інтерфейс керування plug-in модулями, призначений для вибору додаткових модулів, що функціонують у системі; і інтерфейс доступу до ресурсів, що забезпечує надійну авторизацію користувачів і забезпечення ймовірність внесеної й аналізованої інформації.

Розроблена архітектура відрізняється високим ступенем надійності, та в той самий час, достатньою гнучкістю для автоматичної модифікації й вдосконалення. Система має відкриту архітектуру й дозволяє робити підключення додаткових модулів, що реалізують розширення можливостей й технології логічного виводу даних та знань.

3.3 Розробка механізмів наповнення та індексації інформаційних операційних баз

Одним з важливих етапів роботи інформаційної системи є наповнення та індексація операційних баз. На цьому етапі проводиться перетворення вмісту інформаційних баз для поліпшення показників роботи системи. Виконується формування загальної концептуальної моделі для всього вмісту інформаційної системи. Керування процесом індексації виконується функціями ядра системи, спеціальним модулем наповнення операційних баз у наступній послідовності:

1. Виконується аналіз ресурсної інформаційної бази, а потім виділення шаблонів і керуючих ресурсів. Визначення типів ресурсів здійснюється згідно семантичних анотацій, які використовуються в системі. Ця описова інформація із семантичних анотацій вноситься в базу ресурсів (рисунок 3.2 зв'язком під номером (1));

2. Виконується копіювання ресурсів управління та шаблонів у спеціальні операційні бази з маркуванням цих ресурсів для запобігання наступної обробки (рисунок 3.2 зв'язок (2)). Одночасно з цим виконується поділ ресурсів управління на бінарні модулі plug-in та набори скриптів, що містять правила для одержання нових знань;

3. Наступною дією виконується нормалізація анотацій і завантаження їх в операційні бази нормованих анотацій, кожній анотації в інформаційній базі відповідає одна нормована анотація, за винятком тих випадків, коли анотація описує ресурс управління або шаблон документа (рисунок 3.2 зв'язок (3));

4. Для формування єдиного інформаційного простору загальної концептуальної моделі застосовується технологія інтеграції інформаційних онтологічних баз у єдину операційну онтологію. На першому етапі завантажується загальна онтологія, призначена для підтримки роботи основних інформаційних механізмів. Підчас інтеграції інформаційного змісту декількох систем виникає необхідність створити концептуальну модель, в межах якої можна буде проводити

аналіз інформаційних ресурсів. У таких випадках у спеціальну інформаційну базу заносяться додаткові фрагменти онтологій, що описують предметну область для формування анотацій. Процес відображений на рисунку 3.2 зв'язком (4).

Усі ресурси, визначені як документи, зберігаються тільки в ресурсній базі та призначені для випадків, коли не потрібна інтелектуальна машинна обробка. Основна робота виконується з анотаційними базами, бо саме інформація, що входить до анотаційних баз і призначена для подальшої машинної обробки.

Найважливішим компонентом при формуванні системи менеджменту й інтеграції інформації, безумовно, є **розширення й доповнення онтологічної бази** (основи механізмів якої описано в розділі 2.4). Безумовно, розглянуті в роботі технології дозволяють виконувати тільки просту інтеграцію онтологій, в той час, як застосування більш складних технологій неминуче спричинить появу складно діагностованих помилок.

Важливим процесом є одержання нових знань на основі застосування спеціального набору логічних правил.

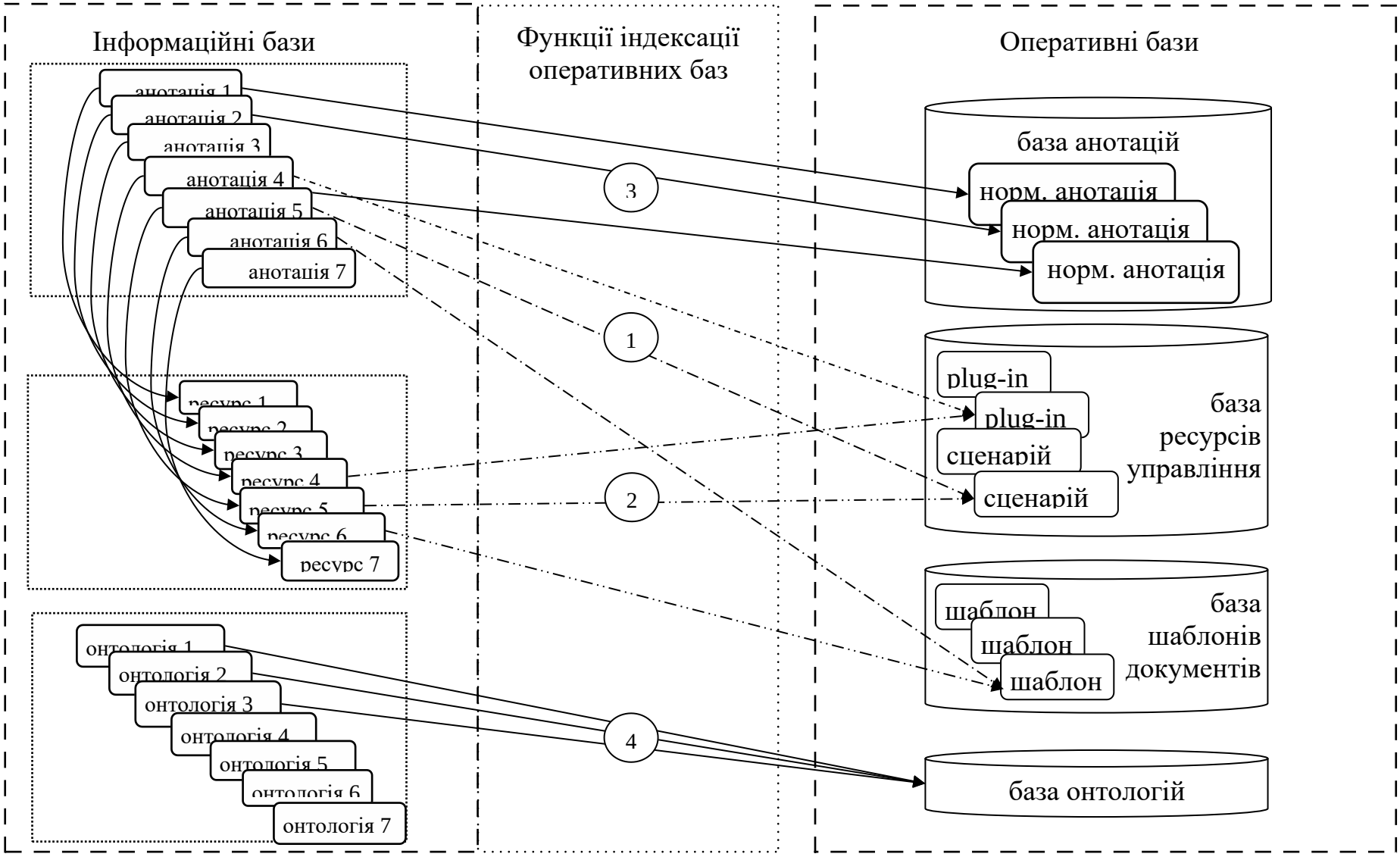

Рисунок 3.2 Схема наповнення та індексації операційних баз *(1 – виділення анотацій управління і шаблонів; 2 – виділення ресурсів управління і шаблонів; 3 – нормалізація анотацій; 4 – інтеграція онтологій)*

3.4 Розробка механізму використання логічних правил та генерації нових знань

Після індексації операційних баз система починає своє звичайне функціонування. Надаються можливості поповнення інформаційних баз, виконання наборів логічних правил, одержання нових знань і виконання процедур формування документів. Одним зі найскладніших системних процесів є одержання нових знань, схема виконання процедури якого показана на рисунку 3.3, та відбувається у кілька етапів:

1. Виконується аналіз вмісту операційної бази ресурсів управління і виділення з неї plug-in модулів й підключення їх до механізмів ядра системи. Plug-in призначені для доповнення базових функцій, що розширюють можливості ядра системи. На рисунку 3.3 ці дії показані зв'язком 1;

2. Також з операційних баз виділяються скрипти управління, які реєструються в системі спеціальним чином, розширюючи набори дій, які здатна виконувати система для користувача. Функції, і методи їх опису повинні підтримуватися набором plug-in, завантажених у систему. При завантаженні модуля набору скриптів проводиться аналіз семантичних описів, завантажених до операційної бази. Для кожного скрипту в його описі вказується набір модулів і plug-in, необхідних для нормальної роботи модулів виводу інформації. На рисунку 3.3 ці дії показані зв'язком 2;

3. Модуль сценаріїв формує необхідну інформацію для автоматичної генерації інтерфейсу компонентів управління. Ці компоненти надають користувачеві можливість вибрати ті або інші дії для одержання нових даних і знань на основі інформаційного наповнення анотаційних баз (рисунок 3.3 - зв'язок 3).

4. Модуль сценаріїв формує необхідну інформацію для автоматичної генерації інтерфейсу компонентів управління. Ці компоненти надають користувачеві можливість вибрати ті або інші дії для одержання нових даних і

знань на основі інформаційного наповнення анотаційних баз (рисунок 3.3 - зв'язком 4).

5. Після вибору відповідної дії виконується аналіз скрипту й запуск необхідних функцій plug-in, що можуть бути запущені послідовно в міру необхідності їх виклику. Приміром, скрипт розбивається на блоки й для кожного блоку викликається функція plug-in, що підтримує «прологоподібні» правила, відповідальні за обробку цього блоку, яка у свою чергу, викликає функцію антецедента з параметрами, вираженими у формі атомів, відповідно розробленій у пункті 2.4 моделі, і результат передає функції консеквента з атомом, що містять результуюче твердження. На останньому етапі здійснюється читання або запис інформації з анотаційної і онтологічної баз. У правилах атоми виражаються в тих же термінах, що анотація, згідно з концептуальною моделлю, у рамках якої написані нормовані анотації. Для здійснення можливості повноцінної автоматичної обробки необхідно щоб усі атоми, застосовувані в «прологоподібних» правилах, були в обов'язковому порядку описані в онтології (рисунок 3.3 – зв’язок 5).

6. За наявності потреби читання даних з анотаційної бази виконується пошук необхідного класу в отології, та після отримання назви в нормалізованих анотаційних базах відбувається вибірка даних та анотацій. Ці процеси на рисунок 3.3 показано під номерами 7 – для читання з онтології й 6 – для читання й запису інформації в нормалізовану анотаційну базу. Результатом виконання операції є новий інформаційний файл, створений у форматі семантичної анотації. Саме в ньому описуються отримані дані зі вказівкою джерела їх формування.

За допомогою такої організації створюється можливість чіткого контролю правильності логічного виводу й можливості для відкату у випадку виникнення помилок. Важливим компонентом у всій системі інтелектуальної інтеграції інформації є онтологічна база. Саме на основі онтології й з'являється можливість обробки інформації створеної різними людьми з використанням різних описових конструкцій.

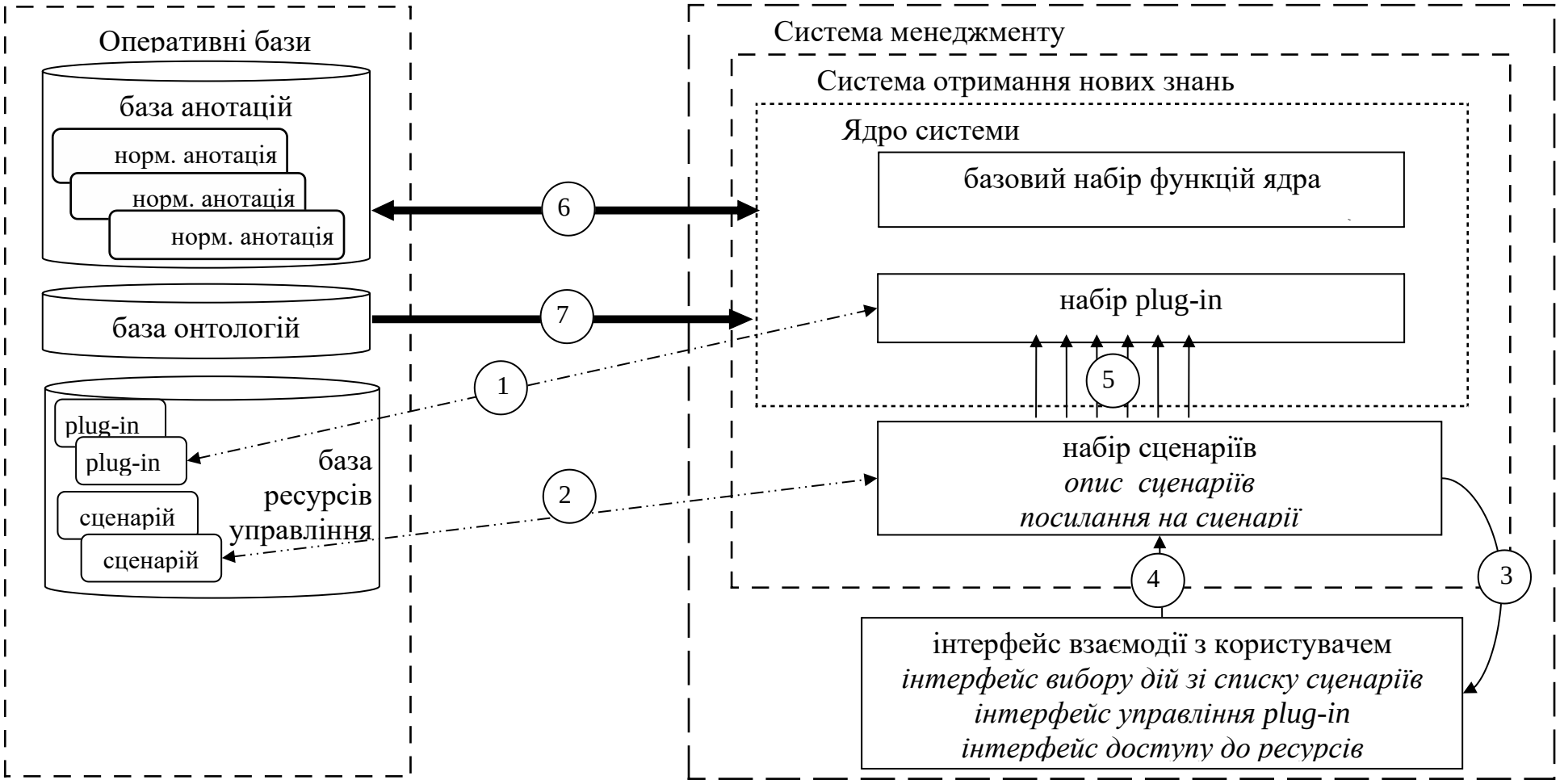

Рисунок 3.3. Обробка інформаційного змісту й генерація нових знань (Рисунок виконаний самостійно)

(1 – формування списку доступних plug-in; 2 – формування списку доступних сценаріїв; 3 – динамічне формування інтерфейсу користувача; 4 – вибір дії користувачем; 5 – виконання сценарію й багаторазовий запуск plug-in; 6 – взаємодія з базою нормалізованих анотацій; 7 – взаємодія з онтологічною базою).

3.5 Розробка механізму виводу інформації в зрозумілому для людині вигляді

У попередніх розділах були розглянуті питання, пов'язані з машинною обробкою вмісту інформаційних баз, але для представлення інформації в зрозумілому людині вигляді був розроблений спеціальний модуль представлення інформації на основі описаної в пункті 2.3, 2.5 технології шаблонів. Використовуючи метод, розроблений у пункті 2.5, з'явилася можливість описувати та структурувати інформаційні ресурси, призначені для формування результуючих документів. Інформація вноситься до інформаційні бази у вигляді ресурсів з додатковим семантичним описом, в якому вказується тип і призначення ресурсу.

За допомогою динамічного інтерфейсу можливо виконувати генерацію нових документів, що містять фрагменти семантичної інформації з анотаційної бази. При внесенні інформації до анотаційної бази здійснюється перетворення інформації у формальний «машинно-зрозумілий» вигляд. Далі існує можливість за допомогою механізмів виводу інформації в зрозумілому для людини вигляді зробити зворотне перетворення, проте, зважаючи на те, що інформація перебуває в загальній базі та виражена в межах єдиної концептуальної моделі, існує можливість при формуванні документів вибирати несуперечливі фрагменти з різних анотацій і використовувати їх в межах одного нового документу.

Таким чином, вирішується питання щодо повторного використання інформаційних ресурсів в автоматичному режимі. Також існує можливість одержання нових знань і даних, а потім генерація результуючих документів, що містять ці дані. Для формування нових документів згідно деталізованій в розділах 2.3, 2.5 технології шаблонів використовується спеціальний режим роботи системи (рисунок 3.4). Відмінною рисою такого механізму є семантичний опис інформації, яка повинна бути додана до шаблону в процесі аналізу інформаційних баз. Така організація дозволяє створювати шаблони різним людям, використовуючи свої терміни для опису інформації, яка повинна бути представлена в документах.

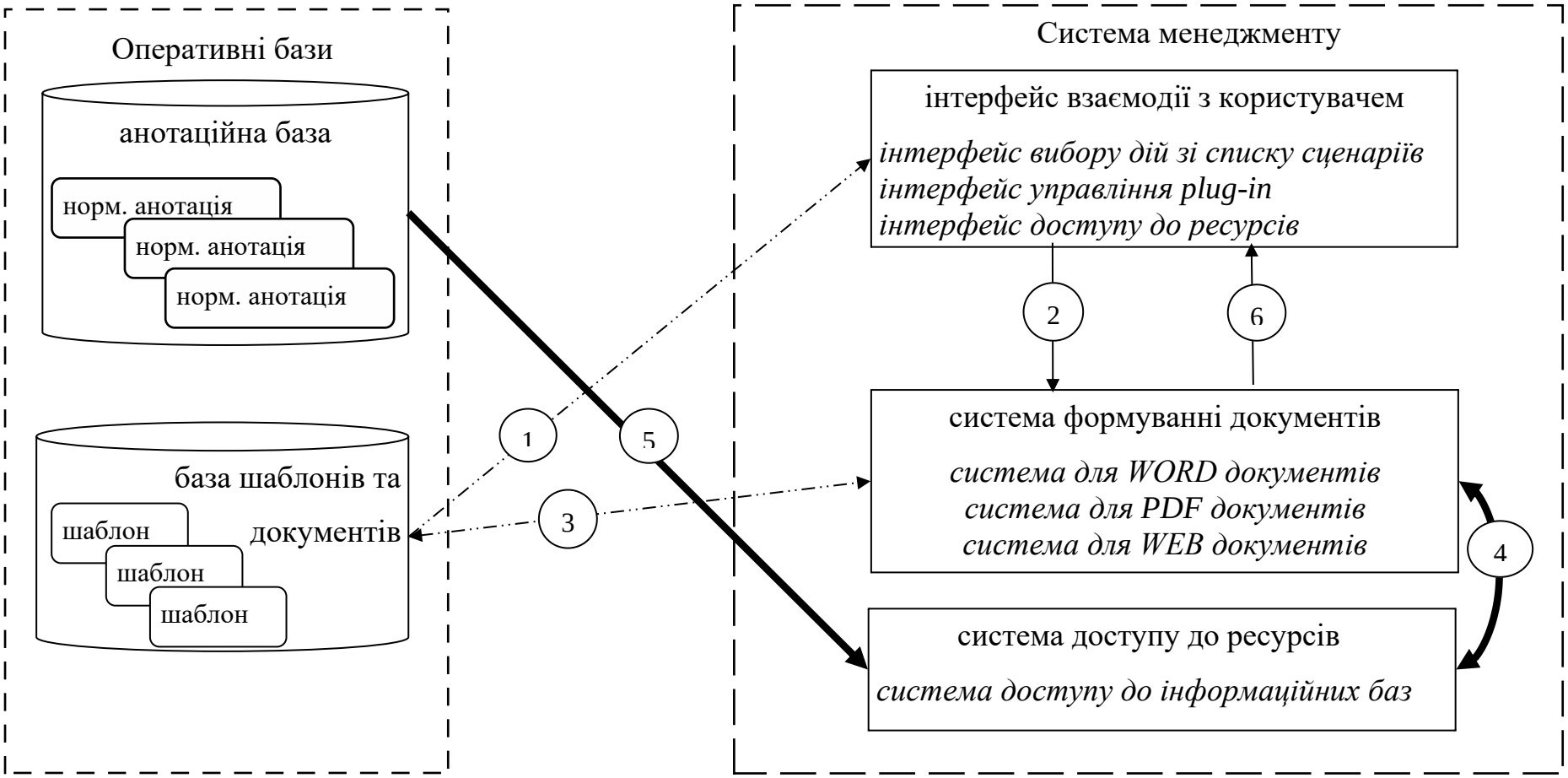

Рисунок 3.4. Формування результуючих документів у форматі зрозумілому для людини (Рисунок виконано самостійно)

(1 – динамічне формування інтерфейсу користувача; 2 – вибір типу формованого документа користувачем; 3 – читання шаблону; 4 – взаємодія із системою контроля доступу до ресурсів; 5 – читання інформації з анотаційної бази.).

Процес формування вихідних документів проводиться в кілька етапів, відображених на рисунку 3.4:

1. Після остаточного індексування операційних баз виконується аналіз вмісту бази шаблонів і динамічне створення компонентів інтерфейсу користувача. До кожного шаблону, внесеного в інформаційну базу, за допомогою URI прикріплюється спеціальна анотація. У цій анотації перебуває інформація щодо типу, призначення шаблону, коментарів до шаблону, інформації для відображення в інтерфейсі користувача тощо. Ця інформація в процесі заповнення операційних баз переноситися в спеціальну базу шаблонів разом із самими шаблонами. При формуванні динамічного інтерфейсу для кожного шаблону використовується інформація з анотацій та формується свій компонент користувацького інтерфейсу, що дозволяє виконувати встановлення спеціалізованих параметрів для документів, сформованих цим шаблоном (рисунок 3.4 - зв'язком 1).

2. Після вибору користувача запускається спеціалізований компонент програми, призначений для формування документів рисунок 3.4 (зв’язок 2). До цього компоненту входить система формування документів у форматі Word, PDF та Інтернет документів мовою HTML. Застосування останньої дозволяє досить легким способом модифікувати систему для можливості її використання в комплексі з Web системами, що представляють інформацію в Інтернеті. Системи виконують розбір шаблонів і перенесення статичної інформації у вихідні документи.

3. Для заповнення динамічних інформаційних полів, описаних на початку шаблону, згідно з технологією, описаною в пункті 2.5, виконується спеціальний запит до підсистеми доступу до ресурсів. У якості параметрів передаються сформовані запити мовою SQL в пункті 2.3 (рисунок 3.4 – зв’язок 4).

4. Сформований документ через компоненти динамічного інтерфейсу повертається користувачеві у форматі файлу текстового процесора Word, PDF або у форматі Інтернет сторінки (рисунок 3.4 – зв’язок 6).

Після виконання всіх робіт наступним етапом створюється можливість одержання повністю завершених документів у форматі Word, PDF та HTML.

Підхід, застосовуваний при створенні архітектури, дозволяє робити автоматичну інтелектуальну трансформацію й перенесення інформації з одних документів, де інформація виражена в будь-якому, навіть контекстно-залежному вигляді, в інші документи без залучення людини. Такий результат досягається за рахунок перетворення інформації у вигляд, що відповідає моделі, розробленої в розділі 2.1, і застосування принципів для інтеграції онтологій, описаних у розділі 2.2. До того ж, шаблони створення документів також є ресурсами, і створюється можливість для їхньої автоматичної генерації засобами інтелектуальних механізмів системи інтеграції інформації.

Висновки розділу

У третьому розділі спроектований метод побудови інтелектуальних систем менеджменту й інтеграції інформації, заснованої на використанні технології Semantic Web. При використанні цього методу й рекомендацій зі створення систем менеджменту документів спроектовано систему менеджменту й інтеграції інформації, відповідної до стандартів Semantic Web.

4 ОПИС ПРОГРАМНОЇ СИСТЕМИ МЕНЕДЖМЕНТУ, ІНТЕГРАЦІЇ Й ПОШУКУ ІНФОРМАЦІЇ, ПРЕДСТАВЛЕНОЇ НА ОСНОВІ СТАНДАРТИЗОВАНИХ МОДЕЛЕЙ ЗНАНЬ

Основною відмінністю систем основаних на онтологіях є наявність можливості семантичного опису інформації. Якщо раніше при інтеграції інформації людині необхідно було виконати аналіз інформації в різних інформаційних базах та встановити поля цих баз у відповідність, то системи основані на онтологіях можуть виконувати цю операцію в автоматичному режимі. Вони мають інтелектуальні механізми, які дозволяють виконувати автоматичну інтерпретацію інформації.

Це їхня властивість відкриває масу різних можливостей щодо застосування систем основаних на онтологіях. В межах роботи були розроблені механізми організації подібних систем, методи й принципи підтримки їх функціонування (розділ 2). На основі розроблених моделей та принципів була спроектована архітектура системи інтеграції інформації на основі стандартизованих моделей знань і проведена її спеціалізація для задач освітньо-наукового процесу (розділ 3). У четвертому розділі проводиться опис практичної реалізації компонентів системи.

4.1 Проектування інформаційної системи для інтеграції інформації освітньо-наукової документації

Спроектована архітектура системи інтелектуальної інтеграції інформації на основі стандартизованих моделей знань є відкритою й надає можливості для свого розширення без зміни внутрішньої організації системи. Вона побудована таким чином, що в стані вирішувати широкий спектр завдань, пов'язаних з обробкою, накопичення інформації й одержанням нових знань на основі наборів «прологоподібних» правил.

Одним із найважливіших завдань системи є перетворення освітньо-наукової документації, семантична обробка, порівняння та аналіз якої є непідсильним завданням для традиційних статистичних методів, через складністю її перетворення «машинно-зрозумілий» вигляд.

Однак система, відповідна до архітектури розробленої в третьому розділі, в стані впоратися із завданнями такої складності. Однією з її переваг є можливість інтерпретувати й обробляти семантичну інформацію.

Механізми обробки інформації, що реалізують вказані функції в межах системи інтеграції інформації на основі стандартизованих моделей знань, показані на рисунку 4.1.

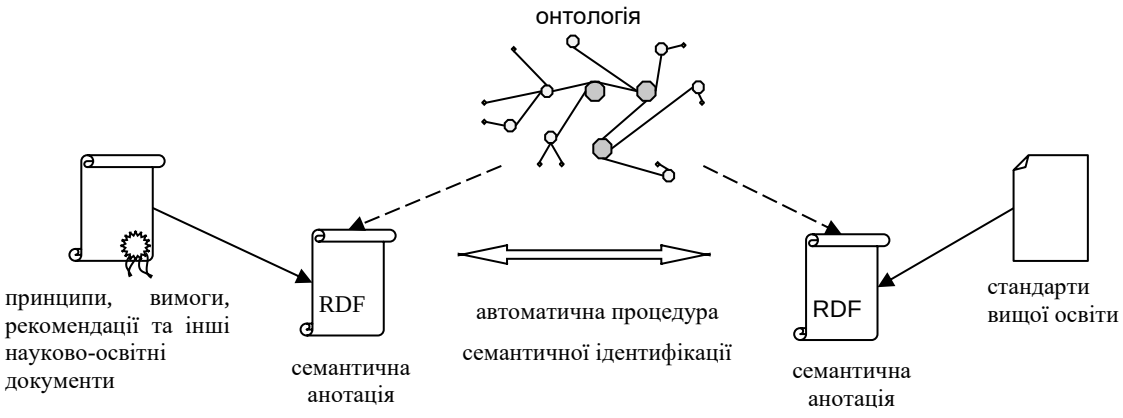

Рисунок 4.1. Механізм обробки інформації в межах системи інтеграції інформації на основі стандартизованих моделей знань (Рисунок виконано самостійно)

В спроектованій процедурі однозначного визначення інформації (рисунок 4.1) використовується механізм інтерпретації, закладений у концепціях SW. Завдяки цьому, основний код має виконувати порівняння інформації згідно значень семантичних змінних, на основі логічних правил, описаних у розділі 3.4. Основним компонентом такої системи є онтологія, що виконує роль універсальної бази знань з предметної області.

4.2 Вибір середовища реалізації програми

Основною задачею кваліфікаційної роботи є проектування та розробка системи створення, автоматизованого розширення, модифікації та інтеграції онтологій для аналізу документів сфери освіти з можливість отримання нових знань та взаємодією з запитами користувача до масиву анотованих електронних документів.

Для реалізації подібної задачі було обрано мову програмування Kotlin з використанням декількох додаткових фреймворків, як то Spring, Hibernate, Liquibase які в змозі забезпечити обмін великими масивами даних з інформаційними та онтологічними базами та створювати структуру бази даних в момент розгортання системи. В якості баз даних обрано декілька типів: MS SQL Server з графоподібною структурою бази для роботи з онтологією, Redis у якості оперативної бази даних, через її високу швидкість роботи та можливість звітування та зміну статусів роботи в якості «подій» та IBM Db2 для роботи з RDF документами в форматі XML.

Використання мови програмування Kotlin мотивоване ти, що вона дозволяє створити інтерфейси взаємодії з користувачем різного типу, зручно працює як з очікуваними так і несподіваними винятковими ситуаціями, має зручний механізм роботи гіпертекстовими посиланнями, файлами, та обробкою XML-документів.

В розробці програми використовуються об’єктно-орієнтовний підхід – програма складається з сукупності об’єктів, які взаємодіють між собою на різних рівнях абстракції.

До додаткових переваг мови програмування Kotlin можна віднести:

- мультиплатформенність – дозволяє запускати програму на різних операційних системах та пристроях, отримуючи при цьому ідентичний результат роботи;
- готова для серверного розгортання – при розділення логіки на клієнт-серверну дозволяє розміщувати частини коду на різних серверах, з можливістю обміну даними;
- легко інтегрується з JavaScript задля створення web чи мобільної версії;
- має високопродуктивний компілятор коду в байт-код;
- легко розширюється Data Science бібліотеками;

Не в останню чергу в виборі мови програмування відіграє її JVM складова, що дозволяє використовувати open-source бібліотеки та компоненти з інших JVM-мов програмування, як то Java, Groovy, Scala, тощо, без проблем із сумісністю та взаємодією.

Завдяки компіляції коду в байт-код, пришвидшується швидкість розгортання та запуску змін, як то в серверній інфраструктурі, так і у випадку зборки коду в програму для запуску на звичайному комп'ютері. Через постійне покращення швидкості роботи компілятора мова програмування Kotlin вже майже наблизилася до швидкості Scala, що займає лідуючу позицію з швидкості компіляції коду серед JVM-мов програмування.

4.3 Основний функціонал програми автоматизації документообігу

Програма автоматизації документообігу призначена для обробки науково-освітніх документів з використання онтологій й має реалізувати функціонал згідно визначених в розділі 3 інтерфейсів взаємодії та програмних модулів.

Інтерфейс взаємодії з користувачем має бути забезпечений наступними можливостями:

- інформувати щодо списку можливих сценаріїв дії;

- надавати інструменти управління plug-in;

- виконувати додавання нового документу в систему з його наступним аналізом;

- виводити на екран, в зручному для користувача вигляді, обраний документ та загальний список існуючих документів в системі;

- виконувати перетворення із наступним зберіганням в файлову систему обраний користувачем документ з системи у WORD, PDF, HTML;

- надавати інтерфейс доступу до інформаційних баз даних.

Враховуючи усе вищесказане, з використанням векторного онлайн-сервісу розробки інтерфейсів та прототипування Figma, спроектовано дизайн можливого вигляду програми автоматизації документообігу (рисунок 4.2).

Блок Ієрархія відображає усі наявні в системі класи, які формують загальну онтологію. Базовим абстрактним суперкласом для всіх класів є клас :THING. Відношення між класами описується за допомогою додавання до одного з них спеціального об'єкта типу :CHILD-NODE, що містить посилання на інший. Таким чином, включаючи в кожний із класів посилання на об'єкт або групу об'єктів, ми отримуємо повну схему взаємодії між класами системи.

Кожний елемент списку може містити інформацію щодо тривалості (у годинах), експертну оцінку важливості, посилання на експерта або експертів (викладачів), які виконали оцінювання важливості, та посилання на набір дисциплін, що входять до цього елементу.

Важливою особливістю системи є можливість відображення даних у форматі RDF та можливість використання динамічно сформованої онтології, що описує обраний елемент (клас).

При виборі певного елементу онтології користувач також має можливість додати в нього новий (дитячий) елемент деревовидної структури, чи видалити існуючий елемент.

Блок **Властивості –** відображає всі деталі щодо обраного елемента списку: дату створення чи модифікації, взаємодію з іншими елементами по ієрархії, дані анотацій та усіх полів, що наповняють елемент (клас) даними.

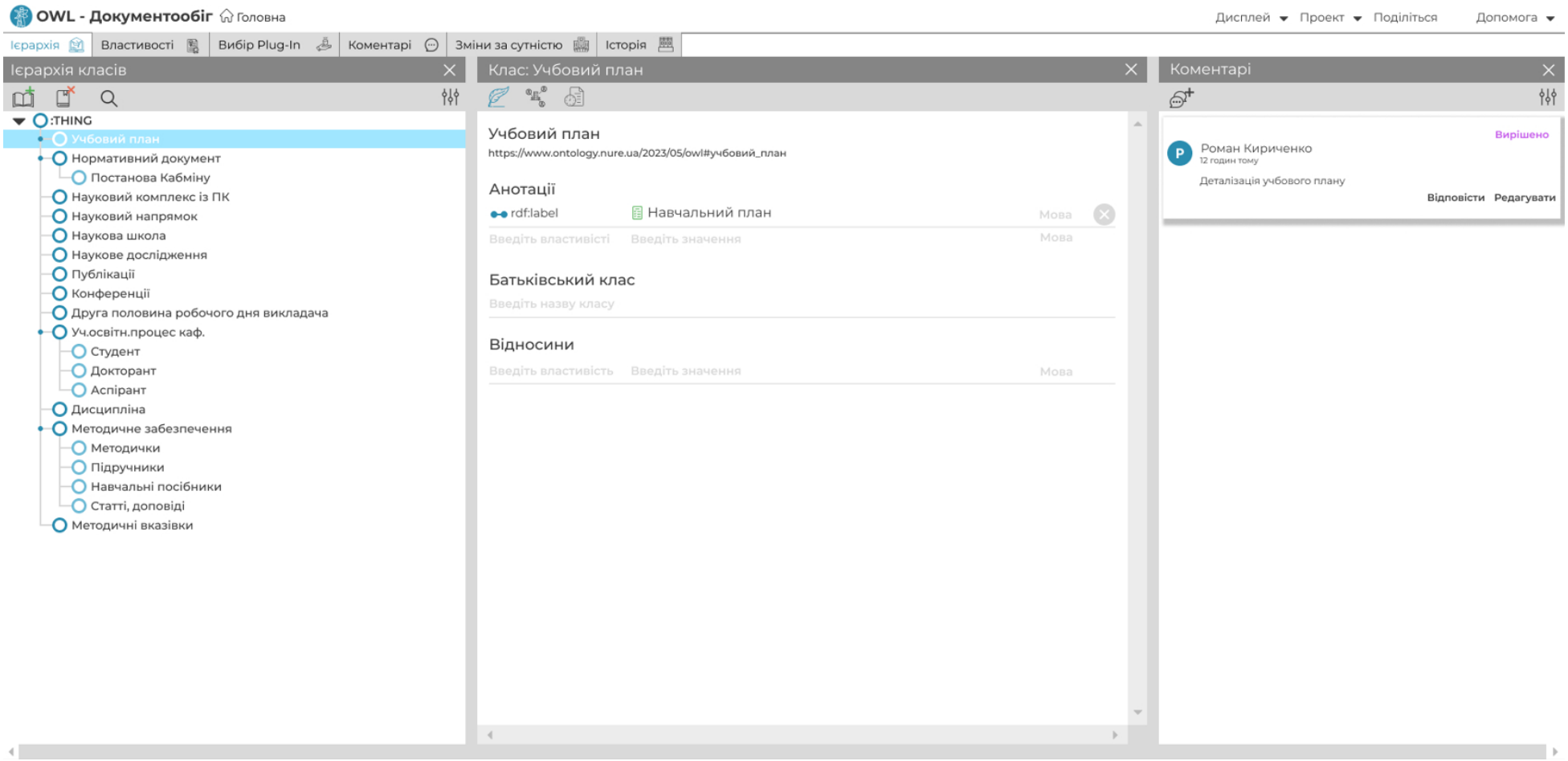

Рисунок 4.2 - Дизайн вигляду програми автоматизації документообігу (Рисунок виконано самостійно)

До додаткового функціоналу цього блоку можливо віднести можливість перетворення RDF документу до одно з підтримуємих типів документів (WORD, PDF, HTML) зі збереженням отриманого результату у файлову систему користувача.

Блок **Plug-In –** надає користувачеві список усіх підключених в систему плагінів, відображаючи стосовно кожного дані стосовно: текстового опису функціоналу обраного plug-in, поточну версію, та параметри налаштування.

Блок **Коментарі –** списком виводить усі коментарі в системі створені усіма користувачами, згідно дати створення (найновіші згори) та надає можливість відповісти на коментар, закрити зі статусом «вирішено», відредагувати коментар (у випадку відкриття його автором), відсортувати за статусом та датою створення.

Блок **Зміни за сутністю –** містить інформацію стосовно авторства елементу, історії змін та типу.

В блоці **Історії** вказано список загальних подій, що сталися в системі в відсортованому вигляді, де найвищий елемент – останній елемент за хронологією.

4.4 Програмна реалізація основних елементів застосунку

Перша задача, яка має бути вирішена підчас створення застосунку - створення структури баз даних. Найкращій спосіб виконання цього програмним способом – використання бібліотеки Liquibase, що надає можливість версіонування баз даних одночасно для усіх запущених клієнтів з можливістю планового розгортання змін з урахуванням авторства та синтаксису обраної бази даних.

Задля створення подібної структури було відображення цього процесу в файловому форматі yaml через його зручність його використання та простоту сприйняття інформації.

Бібліотека Liquibase розгортає зміни структури бази через так названі **changeset**, кожний з яких має на меті виконати певне оновлення бази даних. Список змін, що можуть бути впроваджені в межах подібної зміни доволі великий і виконує

основні операції, як то додавання, змінення, видалення: таблиць, колонок, основних чи додаткових ключів, типів значень, обмежень, та ще багато іншого.

Для створення базової структури онтологічної бази даних, із основними колонками та обмеженнями створено перший changeset, частину з якого приведено нижче. Усі подальші зміни структури бази чи даних, може бути виконано подібними до цього змінами, які легко відслідковувати та перевіряти до моменту їх впровадження в основний застосунок.

```
- changeSet:
    id: 1
    comment: Basic table structure creation
    author: Roman_Kyrychenko
    changes:
      - createTable:
          tableName: thing
          columns:
            - column:
                name: id
                type: bigint
                autoIncrement: true
                constraints:
                  primaryKey: true
                  nullable: false
            - column:
                name: uri
                type: varchar
                defaultValue: http://www.w3.org/2002/07/owl#Thing

      - createTable:
          tableName: node
          columns:
            - column:
                name: id
                type: bigint
                autoIncrement: true
                constraints:
                  primaryKey: true
                  nullable: false
            - column:
                name: uri
                type: varchar(2000)
            - column:
                name: thing_id
                type: bigint
                constraints:
                  nullable: true
            - column:
                name: parent_id
                type: bigint
                constraints:
                  nullable: true
```

```
  - createTable:
      tableName: annotation
      columns:
        - column:
            name: id
            type: bigint
            autoIncrement: true
            constraints:
               primaryKey: true
               nullable: false
```

Аби взаємодіяти з новоствореною структурою бази даних необхідно зробити об’єкти-репрезентатори таблиці (entity) в кодовому представленні. При створенні подібного відображення використано фреймворк Hibernate, що дозволяє взаємодіяти з таблицями з коду та виконувати усі необхідні модифікації в базі даних.

Основним елементом структури онтологічної бази даних є клас Node, що містить в собі інформацію стосовно URI, текстових анотацій, використаних для опису певного документу, батьківського та дитячих елементів в онтологічному відображенні. Нижче наведено кодове відображення таблиці Node.

```
@Entity
class Node(
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(nullable = false)
    val id: Long = 0,
    val uri: URI?,

    @Transient
    private val parentId: Long,

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "thing_id")
    val thing: Thing?,

    @ManyToMany
    @JoinTable(
        name = "node_to_annotation",
        joinColumns = [JoinColumn(name = "node_id",
referencedColumnName = "id")],
        inverseJoinColumns = [JoinColumn(name = "annotation_id",
referencedColumnName = "id")]
    )
    val annotations: MutableList<Annotation> = mutableListOf(),
```

```
    @ManyToOne(fetch = FetchType.LAZY, optional = true)
    @JoinColumn(name = "parent_id")
    private val parent: Node?,

    @OneToMany(mappedBy = "parent", fetch = FetchType.LAZY, cascade =
[CascadeType.ALL], orphanRemoval = true)
    val children: MutableList<Node> = mutableListOf()
)
```

Висновки розділу

У четвертому розділі виконано проектування системи інтеграції інформації на основі стандартизованих моделей знань.

Спроектовано методи для підтримки процесу створення онтологій. Використання онтології в системі інтеграції й менеджменту інформації дозволяє використовувати систему для обробки документів.

Наведено основний функціонал системи інтеграції, пошуку й обробки інформації на основі стандартизованих моделей знань і її компонентів.

Створено основний дизайн, структуру бази даних та об’єкти-відображень, частину з яких надано в якості кодового представлення.

ВИСНОВКИ

У кваліфікаційній роботі наведено теоретичне узагальнення й нове рішення наукового завдання, розробки методів створення систем менеджменту й інтеграції інформації, заснованої на використанні принципів Semantic Web.

В ході проведених досліджень отримано нові наукові результати:

– У результаті аналізу сучасного стану проблеми семантичної обробки інформації визначений ряд недоліків існуючих систем, зв'язаних зі складністю аналізу інформації, представленої в природно-мовному вигляді. Шляхом вирішення цих проблем є створення інтелектуальних інформаційних систем, заснованих на технології Semantic Web, яка надає стандарти представлення інформації й забезпечує можливість для семантичної машинної обробки інформації, але не надає алгоритмів і методів такої обробки. В зв’язку з чим доцільним є виконання розробки методів підтримки семантичної обробки інформації, заснованих на технології Semantic Web, та розробку на базі цих методів інтелектуальної систему менеджменту й інтеграції різнорідної інформації. Це визначило вибір напрямку дослідження, формулювання мети й завдань кваліфікаційної роботи.

– Спроектований метод побудови інтелектуальних систем менеджменту й інтеграції різнорідної інформації, яка сформована на базі технології Semantic Web. Використання цього методу надасть можливості для створення систем семантичного аналізу різнорідної інформації, які мають істотні переваги над існуючими інформаційними системами.

– Подальший розвиток одержав метод трансформації онтологій, які засновані на застосуванні стандартів Semantic Web. Завдяки застосуванню цього методу стає можливо інтегрувати різнорідну інформацію, сформовану з відповідності зі стандартами Semantic Web.

– Розроблений метод формування запитів до анотаційної бази знань на основі технології Semantic Web. Цей метод забезпечує можливість одержання різної інформації з анотаційної та онтологічної баз.

– Також подальший розвиток отримав метод представлення «прологоподібних» правил, які описують знання, на основі використання принципів RuleML, для опису знань, що дозволяє одержувати нову інформацію завдяки застосуванню цих правил до інформаційного масиву.

– Спроектовано дизайн та основний функціонал системи менеджменту й інтеграції різнорідної інформації на основі стандартизованих концептуальних моделей, методу трансформації онтологій, методу представлення у виконання «прологоподібних» правил у підсистемі обробки й аналізу текстової інформації.

– Через масштабність спроектованої системи виконано лише частину її можливого функціоналу, що стосується структури онтологічної бази, представлення, редагування онтології та її збереження до бази даних.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Semantic Web Development // World Wide Web Consortium. URL: http://www.w3.org/2000/01/sw/ (Дата звернення: 21.01.2023)

2. Симонович С., Євсєєв Г., Алексєєв А. Загальна інформатика. // АСТ-Прес, 2000 - 592с.

3. Scott Ainsworth, Ahmed AlSum, Hany SalahEldeen, Michele C. Weigle, and Michael L. Nelson, “How Much of the Web Is Archived?” // In Proceedings of the ACM/IEEE Joint Conference on Digital Libraries (JCDL). Ottawa, Canada, June 2011 - 133-136 p.

4. Як працює Google пошук. // Пошук Google – Дізнайтеся, як працює Пошук Google. URL: https://www.google.com/search/howsearchworks/how-search-works/organizing-information/ (Дата звернення: 25.01.2023)

5. Google Search Engine Optimization (SEO) Starter Guide. // SEO Starter Guide: The Basica | Google Search Central | Documentation | Google Developers URL: https://developers.google.com/search/docs/fundamentals/seo-starter-guide (Дата звернення 25.01.2023)

6. Clickthrough rate (CTR): Definition. // Google Ads Help URL: https://support.google.com/google-ads/answer/2615875 (Дата звернення: 27.01.2023)

7. What Is Dwell Time? // Dwell Time and SEO: The Complete guide. URL: https://backlinko.com/hub/seo/dwell-time (Дата звернення: 30.01.2023)

8. About bounce rate. // Bounce Rate – Analytics Help. URL: https://support.google.com/analytics/answer/1009409 (Дата звернення: 03.02.2023)

9. Pogo-Sticking in SEO: What It Is & What to Do About It. // A-hrefs blog. URL: https://ahrefs.com/blog/pogo-sticking/ (03.02.2023)

10. How many files are there in the world? // CloudFiles. URL: https://cloudfiles.ghost.io/how-many-files-are-there-in-the-world/ (Дата звернення: 07.02.2023)

11. PDFs in Google search results. // Google Search Central. URL: https://developers.google.com/search/blog/2011/09/pdfs-in-google-search-results (Дата

звернення: 12.02.2023)

12. Automate document processing with Document AI for the government. // Google Cloud Blog. URL: https://cloud.google.com/blog/topics/public-sector/automate-document-processing-document-ai-government (Дата звернення: 18.02.2023)

13. Semantic Web // World Wide Web Consortium. URL: http://www.w3.org/2001/sw/ (Дата звернення: 21.01.2023)

14. T. Berners-Lee, J. Hendler, O. Lassila. The Semantic Web // Scientific American 284(5), - May 2001. – p. 34-43 URL: https://lassila.org/publications/2001/SciAm.pdf

15. URI: The Uniform Resource Identifier Explained. // Digital Guide IONOS. URL: https://www.ionos.com/digitalguide/websites/web-development/uniform-resource-identifier-uri/ (Дата звернення: 24.02.2023)

16. XML introduction – XML: Extensible Markup Language // MDN. URL: https://developer.mozilla.org/en-US/docs/Web/XML/XML_introduction (Дата звернення: 24.02.2023)

17. Standard Generalized Markup Language. // Wikipedia. ULR: https://uk.wikipedia.org/wiki/SGML (Дата звернення: 25.02.2023)

18. XML DTD. // Java T Point. URL: https://www.javatpoint.com/xml-dtd (Дата звернення: 01.03.2023)

19. DOM: Living standard. // DOM. URL: https://dom.spec.whatwg.org (Дата звернення: 07.03.2023)

20. XSLT: Extensible Stylesheet Language Transformations. // MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/XSLT (Дата звернення: 09.03.2023)

21. Resource Description Framework (RDF). // World Wide Web Consortium. URL: https://www.w3.org/RDF/ (Дата звернення: 10.03.2023)

22. RDF Schema. // Wikipedia. URL: https://en.wikipedia.org/wiki/RDF_Schema (Дата звернення: 10.03.2023)

23. Web Ontology Language. // Wikipedia. URL: https://uk.wikipedia.org/wiki/Web_Ontology_Language (Дата звернення:

12.03.2023)

24. OWL 2 Web Ontology Language Document Overview. // World Wide Web Consortium. URL: https://www.w3.org/TR/owl2-overview/ (Дата звернення: 12.03.2023)

25. DAML+OIL Reference Description // World Wide Web Consortium. URL: https://www.w3.org/TR/daml+oil-reference/ (Дата звернення 15.03.2023)

26. Sharonova, N., Kyrychenko, I., Gruzdo, I., Tereshchenko, G. Generalized Semantic Analysis Algorithm of Natural Language Texts for Various Functional Style Types // CEUR Workshop Proceedings, 2022, 3171, pp. 16–26

27. Rule Markup Language (RuleML). // Cover Pages: Rule Markup Language (RuleML) URL: http://xml.coverpages.org/ruleML.html (Дата звернення: 16.03.2023)

28. S. Sharoff, L. Sokolova. Semanticization of language-independent concepts in a multilingual system. // Proc. IWCS-4, Utrecht, Holland, January, 2001. p. 433-446.

29. S.V. Bulgakov, E.A. Sidorova, Yu. A. Zagorulko. Ontology-Oriented Multi-Agent Approach to Development of Knowledge Internet Portal // Proceedings of the 6th International Workshop on Computer Science and Information Technologies. CSIT'2004. Budapest, Hungary, 2004, v1, pp. 182-187.

30. SQL Graph Architecture // SQL Graph Architecture – SQL Server | Microsoft Learn. URL: https://learn.microsoft.com/en-us/sql/relational-databases/graphs/sql-graph-architecture (Дата звернення: 12.04.2023