

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра Автоматизації та комп'ютерних технологій систем управління

«До захисту в ЕК»

Директор інституту(декан факультету)

Форсюк А.В.

(підпис)

(прізвище та ініціали)

« » лютого 2021р.

«До захисту допущено»

Завідувач кафедри

Ельперін І.В.

(підпис)

(прізвище та ініціали)

«10» лютого 2021р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

зі спеціальності 151 "Автоматизація та комп'ютерно-інтегровані технології"
(код та назва спеціальності)

освітньо-професійної програми Інтелектуальні комп'ютерні системи керування

на тему: Інтелектуальне керування технологічними процесами приготування тіста на основі онтологічного підходу

Виконав: здобувач ІІ курсу, групи ЗАК-2-1М

Рябушко Дмитро Миколайович

(прізвище, ім'я, по батькові повністю)

(підпис)

Керівник Кишенько Василь Дмитрович

(прізвище, ім'я та по батькові повністю)

(підпис)

Консультанти

(прізвище та ініціали)

(підпис)

(прізвище та ініціали)

(підпис)

(прізвище та ініціали)

(підпис)

Рецензент

Мошенський А.О.

(прізвище та ініціали)

(підпис)

Засвідчую, що в цій кваліфікаційній
роботі немає запозичень із праць
інших авторів без відповідних
посилань.

Здобувач

(підпис)

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем

Кафедра Автоматизації та комп'ютерних технологій систем управління

Освітній ступінь _____

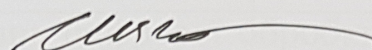
Спеціальність 151 «Автоматизація та комп'ютерно-інтегровані технології»
(код і назва)

Освітньо-професійна програма Інтелектуальні комп'ютерні системи керування
(назва)

ЗАТВЕРДЖУЮ

Завідувач

кафедри АКТСУ Енергетик 1.В.


"09" листопада 2020 року

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

Рябушка Дмитра Миколайовича

(прізвище, ім'я, по батькові)

1. Тема роботи Інтелектуальне керування технологічними процесами приготування тіста на основі онтологічного підходу

керівник роботи Кишенько Василь Дмитрович, ктн, доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від "09" листопада 2020 року
№ 933-кС

2. Строк подання здобувачем роботи 09 лютого 2021 року

3. Вихідні дані до роботи характеристика об'єкта автоматизації, технологічний регламент, мережа інтернет, та ін.

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
1. Мета і завдання виконання кваліфікаційної роботи. 2. Дослідження і адаптація концептуальних і формальних засобів системно-об'єктного підходу для вирішення завдань онтологічного інжинірингу. 3. Розробка засобів побудови онтології на основі системно-об'єктного підходу. Розробка і апробація алгоритмічного та програмного забезпечення для побудови онтологій на основі УФО-моделі та здійснення логічного виведення на УФО-моделі. 4. Висновки.

5. Перелік графічного матеріалу

7. Дата видачі завдання 09 листопада 2020 року

КАЛЕНДАРНИЙ ПЛАН

[illegible]

Здобувач

(підпис)

Керівник роботи

(підпис)

Рябушко Д.М.

(прізвище та ініціали)

Кишенько В.Д.

(прізвище та ініціали)

АНОТАЦІЯ

Сучасна потреба в новітніх рішеннях, при розробці алгоритмів обробки даних і архітектури інтелектуальних систем керування, вивела на перший план нові актуальні завдання. Створення і використання пристроїв цифрової обробки даних, які здатні працювати в складі інтелектуальних систем керування технологічними процесами, повинні вміти отримувати в реальному часі достовірну картину про стан контрольованих фізичних величин і їх параметрів за рахунок закладених в них ефективних алгоритмів обробки даних.

Одним з найбільш актуальних завдань цифрової обробки сигналів – є завдання очищення сигналу від шуму, виділення окремої «локальної» складової. На практиці сигнал завжди містить не тільки корисну інформацію, а й сліди деяких зовнішніх впливів (перешкоди, шум або аномалії), що додає додаткову складність в процес обробки даних.

Для підвищення ефективності хлібопекарського виробництва особливу роль відведено автоматизації технологічних процесів, основними завданнями якої є поліпшення якості хлібопекарської продукції, застосування ресурсоощадних технологічних режимів, збільшення продуктивності виробництва.

Одним із перспективних напрямків у підвищенні якості хлібопекарської продукції є використання онтологій для вирішення завдань інтеграції даних. Процес автоматизації будь-якої діяльності, як правило, вимагає формалізованої постановки задачі і її аналізу з метою вироблення методу рішення. Методи інтеграції даних на основі онтологій ефективні, однак побудова онтології вимагає експертних знань у досліджуваній предметній галузі та займає суттєвий обсяг часу, тому актуальним завданням є автоматизація процесу побудови онтології.

Ключові слова: обробка даних, хлібопекарське виробництво, продуктивність, якість, онтології.

ANNOTATION

The current need for the latest solutions, in the development of data processing algorithms and the architecture of intelligent control systems, has brought to the fore new pressing challenges. The creation and use of digital data processing devices that are able to work as part of intelligent process control systems must be able to obtain a real-time reliable picture of the state of controlled physical quantities and their parameters due to their effective data processing algorithms. When measuring information is distorted by interference, the intelligent control system may make the wrong decision, leading to an accident or irreversible consequences. One of the most urgent tasks of digital signal processing is the task of cleaning the signal from noise, allocating a separate "local" component. In practice, the signal always contains not only useful information, but also traces of some external influences (interference, noise or anomalies), which adds additional complexity to the data processing process.

To increase the efficiency of bakery production, a special role is given to the automation of technological processes, the main objectives of which are to improve the quality of bakery products, the use of resource-saving technological modes, increase productivity. One of the promising areas in improving the quality of bakery products is the use of ontologies to solve data integration problems. The process of automation of any activity, as a rule, requires a formalized statement of the problem and its analysis in order to develop a method of solution. Methods of data integration based on ontologies are effective, but the construction of the ontology requires expert knowledge in the research subject area and takes a significant amount of time, so the urgent task is to automate the process of building the ontology.

Key words: data processing, bakery production, productivity, quality, ontologies.

ЗМІСТ

ВСТУП.....

РОЗДІЛ 1. МЕТА І ЗАВДАННЯ ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

1.1. Огляд завдань управління знаннями організацій і підприємств.....

1.2. Аналіз напрямків розвитку і основних проблем онтологічного інжинірингу.....

1.3. Огляд існуючих методів і підходів до побудови онтологій.....

1.4. Огляд і аналіз системно-об'єктного УФО-підходу та результатів УФО-моделювання.....

1.5. Постановка завдання побудови онтологій предметних областей на основі результатів візуального комп'ютерного моделювання.....

Висновки 1-го розділу

РОЗДІЛ 2. ДОСЛІДЖЕННЯ І АДАПТАЦІЯ КОНЦЕПТУАЛЬНИХ І ФОРМАЛЬНИХ ЗАСОБІВ СИСТЕМНО-ОБ'ЄКТОВОГО ПІДХОДУ ДЛЯ ВИРІШЕННЯ ЗАВДАНЬ ОНТОЛОГІЧНОГО ІНЖИНІРИНГУ

2.1. Дослідження і систематизація відомостей про предметну область, що містяться в візуальних графоаналитических моделях, представляються за допомогою УФО-підходу.....

2.2. Дослідження взаємозв'язку системно-об'єктних УФО-моделей і формалізованих засобів запису онтологій.....

2.3. Дослідження і адаптація засобів алгебраїчного опису УФО-моделей для їх використання при перетворенні УФО-моделей в конструкції мов формального уявлення онтологій.....

2.4. Розробка методу опису УФО-моделі на формалізованому мовою записи онтології.....

Висновки 2-го розділу

РОЗДІЛ 3. РОЗРОБКА ЗАСОБІВ ПОБУДОВИ ОНТОЛОГІЇ НА ОСНОВІ СИСТЕМНО-ОБ'ЄКТОВОГО ПІДХОДУ. РОЗРОБКА І АПРОБАЦІЯ АЛГОРИТМІЧНЕ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПОБУДОВИ ОНТОЛОГІЇ НА ОСНОВІ УФО-МОДЕЛІ ТА ЗДІЙСНЕННЯ ЛОГІЧНОГО ВИВЕДЕННЯ НА УФО-МОДЕЛЕЙ.....

3.1. Розробка методу побудови онтології з використанням візуальної графоаналитического системно-об'єктного УФО-моделі.....

3.2. Особливості валідації онтології, побудованої на основі графоаналитического УФО-моделі

3.3. Розробка методу логічного висновку на візуальних графоаналитических УФО-моделях за рахунок їх інтеграції з формальними засобами записи онтології.

3.4. Дослідження застосовності існуючих засобів і інструментів для здійснення логічного висновку на візуальних графоаналитических УФО-моделях.

3.5. Розробка алгоритму опису УФО-моделі на формалізованому мовою записи онтології

3.6. Рішення задач логічного висновку на візуальних графоаналитических УФО-моделях за рахунок їх інтеграції з формальними засобами записи онтології.

3.7. Розробка дослідницького прототипу програмного пакету побудови онтологій хлібопекарського виробництва.....

3.8. Виклад основних результатів дослідження.....

Висновки 3-го розділу.....

ВИСНОВКИ.....

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ.....

ВСТУП

Для підвищення ефективності хлібопекарського виробництва особливу роль відведено автоматизації технологічних процесів, основними завданнями якої є поліпшення якості хлібопекарської продукції, застосування ресурсоощадних технологічних режимів, збільшення продуктивності виробництва. Враховуючи високу лабільність характеристик сировини, актуальною проблемою автоматизації на хлібопекарських підприємствах залишається оперативна комплексна оцінка в реальному часі якості сировини, напівфабрикатів та готової продукції методами динамічної кваліметрії. Розв'язання таких завдань можливо лише на основі сучасних інтелектуальних технологій і передових досягнень у теорії та практиці автоматизованого управління.

Одним із перспективних напрямків у підвищенні якості хлібопекарської продукції є використання онтологій для вирішення завдань інтеграції даних. Процес автоматизації будь-якої діяльності, як правило, вимагає формалізованої постановки задачі і її аналізу з метою вироблення методу рішення. Методи інтеграції даних на основі онтологій ефективні, однак побудова онтології вимагає експертних знань у досліджуваній предметній галузі та займає суттєвий обсяг часу, тому актуальним завданням є автоматизація процесу побудови онтології.

Онтологія подає інформацію так, що вона також може застосовуватися не тільки для відображення, але й для автоматизації, інтеграції та повторного використання тієї ж інформації в різних додатках, які включають у себе штучний інтелект, інформаційно-пошукові системи тощо. Онтологія визначається як сукупність великої кількості понять, їх визначень і представлених в ієрархічному порядку відносин. Існують різні інструменти, доступні для розробки онтологій, наприклад, Hozo, AltovaSemantic Works та ін. Для побудови онтології було використано *Protégé* – одним із найбільш широко використовуваних онтологій-редакторів, який визначає розвиток

концепції онтології (класи), властивості, таксономії, різні обмеження та екземпляри класів.

Підсистема інтелектуальної підтримки та прийняття рішень на основі динамічної експертної системи з використанням онтологій забезпечить реалізацію оптимального управління технологічним комплексом хлібопекарського виробництва.

Тому доцільно проведення дослідження з метою створення методу побудови онтологій, який дозволив би забезпечити максимальну об'єктивність і повноту створюваної онтології по відношенню до розв'язуваної прикладної задачі, знизити частку безпосереднього участі експертів в процесах збору концептів і визначення зв'язків між ними, а також надати можливості автоматизації процесів онтологічного інжинірингу. Так, альтернативою «ручному» способу побудови онтології, при якому безпосередній відбір понять ведуть експерти, може стати використання на початковому етапі для відбору понять і створення прототипу онтології вже існуючих або розроблюваних комп'ютерних візуальних моделей предметної області. Останнє дозволяє розглядати в якості перспективного для рішення задачі даного дисертаційного дослідження метод візуального графоаналитического моделювання, заснований на системно-об'єктному підході «Вузол-Функція-Об'єкт» (УФО-підхід).

В основі даного підходу і базується на ньому УФО-аналізу лежить уявлення тієї чи іншої системи у вигляді трьохелементної конструкції «Вузол-Функція-Об'єкт», що і зумовило назву підходу. Вузол в контексті УФО-підходу – це уявлення системи як структурного елементу, «перехрестя» всіх її зв'язків. Під функцією розуміється функціональний елемент, який грає певну, задану роль для підтримки надсистеми шляхом балансування даного вузла. Об'єкт представляє систему як субстанціальний елемент, конкретну сутність, що реалізовує функцію. Комбінації з таких УФО-елементів є системно об'єктні моделі (УФО-моделі). В даний час область, що стосується можливості інтеграції УФО-підходу із засобами онтологічного інжинірингу та використання в контексті роботи з онтологіями, не досліджена і в літературі практично не описана. Разом з тим необхідно відзначити, що застосування коштів УФО-підходу в цілях побудови онтологій дозволяє вирішити ряд

існуючих в цій сфері проблем. Так, однією з найважливіших задач є зниження частки безпосереднього («ручного») участі експертів в процесі створення онтології і автоматизація найбільш трудомістких і тривалих операцій. Для вирішення даного завдання потрібні методи, мають досить суворе формальне обґрунтування. УФО-підхід володіє широкими можливостями формалізації, що дозволяє використовувати його кошти в даному випадку. Крім того, з метою ефективності вирішення прикладних завдань розробляється онтологія повинна, з одного боку, максимально охоплювати предметну область, а, з іншого боку, не бути надмірною, тобто не містити зайвих для вирішення певного кола завдань концептів і відомостей про них. Джерела знань про предметну область, на основі яких будується онтологія, повинні бути актуальними і максимально об'єктивні (містити інформацію, представлену з різних точок зору). Цим критеріям в повною мірою задовольняють УФО-моделі.

Таким чином, розробка методу побудови онтологій на основі системно-об'єктного підходу і УФО-моделей предметної області дозволяє вирішити ряд важливих проблем в області онтологічного інжинірингу, що обумовлює актуальність даного дослідження.

1. МЕТА І ЗАВДАННЯ ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Метою роботи є дослідження складного технологічного комплексу хлібозаводу за допомогою методів інженерії знань шляхом розробки онтологій, що дають необхідні можливості практичного застосування результатів для розробки ефективних ресурсощадних стратегій управління технологічним комплексом хлібопекарського виробництва.

Поставлена мета передбачає розв'язання таких завдань:

1. Скласти глосарій понять предметної галузі (хлібопекарське виробництво).
2. Визначити класи онтологій та побудувати ієрархію онтологій хлібопекарського виробництва.
3. Визначити атрибути класів та властивості екземплярів онтологій хлібопекарського виробництва.

Відповідно до мети дослідження необхідно вирішити такі завдання:

1. Дослідити і систематизувати відомості про предметну область, що містяться в візуальних графоаналитических моделях, представляються за допомогою системно-об'єктного підходу «Вузол-функція- Об'єкт »(УФО-підходу).
2. Проаналізувати існуючі методи побудови онтологій і засоби їх формального опису. Виявити і обґрунтувати можливість побудови онтологій на основі візуальних графоаналитических моделей в термінах «Вузол-Функція-Об'єкт» (УФО-моделей).
3. Проаналізувати існуючі засоби алгебраїчного опису УФО-моделей і адаптувати їх для використання при перетворенні УФО-моделей в конструкції мов формального уявлення онтологій.
4. Розробити метод опису УФО-моделі на формалізованому мовою записи онтології.
5. Розробити метод побудови онтології на основі візуальної графоаналитического УФО-моделі.
6. Розробити метод логічного висновку на УФО-моделях з урахуванням інтеграції засобів УФО-підходу та онтологічного інжинірингу.

7. Спроекувати і реалізувати алгоритмічне та програмне забезпечення для побудови онтології на основі УФО-моделі і здійснення логічного висновку на УФО-моделях з урахуванням розроблених методів.

1.1. Огляд завдань управління знаннями організацій і підприємств

Організаційне знання як частина інтелектуального капіталу підприємств в останні десятиліття викликає підвищений інтерес, як з боку вчених, так і з боку самих організацій. В сучасних дослідженнях в галузі управління знаннями відзначається, що носієм як явних, так і неявних знань може виступати не тільки індивідум, але і організація [75]. При цьому організаційне знання, фактично будучи акумуляцією накопиченого досвіду, знань і навичок співробітників, являє собою одну з найбільших цінностей в умовах нового якісного стану світової економіки, що базується на знаннях. Так, згідно з [81], організаційне знання являє собою стратегічний актив, а грамотно вибудований внутрішній процес управління знаннями дозволяє підприємствам залишатися успішними і конкурентоспроможними в сучасному ринковому середовищі. Все зростаюча цінність інтелектуального капіталу підприємств обумовлює цілі і основні завдання управління організаційними знаннями. Так, Віїг [72] сформулював дві основні мети управління знаннями: максимально розумне здійснення організаційні дій і інше застосувань кращих цінностей знань. Дейвенпорт, Де Лонг і Бірс [10] зазначає, що метою управління організаційними знаннями є підвищення здатності організації більш ефективно функціонувати. При цьому в якості основних практичних завдань в роботі [10] названо створення сховища знань, полегшення доступу до знань, розвиток «Середовища знань» («knowledge environment» - практик і механізмів, спрямованих на сприяння спільному створенню знань, прийняття рішень і висновків, спрямованих на досягнення певної мети), управління знаннями в якості активу. В роботі [38] в якості основних завдань управління знаннями в організаціях також згадується обмін інформацією про витягнутих уроках і створення передумов для появи інновацій. Таким чином, більшість авторів сходяться на думці, що управління організаційним знанням вирішує ряд важливих завдань, які в сукупності сприяють поліпшенню показників ефективності діяльності організації. Так, створення передумов для інновацій дозволяє оптимізувати процес

виробництва товарів або послуг, скоротити витрати. Організація загального доступу до накопичених знань, систематизованим і організованим у вигляді сховища, сприяє підвищення кваліфікації співробітників, зниження ризиків прийняття невірних управлінських рішень. Поширення знань про витягнутих з наявного досвіду уроках знижує ймовірність повторення негативних сценаріїв в майбутньому. Створення нових знань на основі вже наявних, в свою чергу, збільшує інтелектуальний капітал організації і може служити основою нових конкурентних переваг. З вищевикладеного випливає, що організації з метою підвищення ефективності власної діяльності і збільшення конкурентоспроможності змушені приділяти значну увагу процесам виявлення, накопичення, зберігання, використання та обміну знаннями всередині компанії. У процесі побудови системи управління організаційним знанням суб'єкти можуть зіткнутися з низкою проблем. Їх рішення, а також вибір спеціальних методів, технік і засобів для найбільш ефективної організації менеджменту знань залежить від обраного підходу управління знаннями.

1.2. Аналіз напрямків розвитку і основних проблем онтологічного інжинірингу.

Щодо молодим, але при цьому досить перспективним і динамічно розвивається підходом до управління знань є онтологічний підхід, в основі якого, згідно з назвою, лежить розробка і використання онтологій. Термін «онтологія» запозичений з філософії і має безліч тлумачень. Під онтологією в класичному уявленні розуміється експліцитно (явна) специфікація концептуалізації [25]. Іншими словами, онтологія являє собою детальну специфікацію структури певної проблемної області. Згідно [98] онтологія - це система, що складається з набору понять і набору тверджень про ці поняття, на основі яких можна описувати класи, відносини, функції та індивіди. Онтологію предметної області також можна розглядати як словник термінів, специфічних для даної предметної області, разом з сукупністю аксіом, які забезпечують інтерпретацію і правильне використання цих термінів [99]. Ще одне тлумачення, представлене в [102], являє онтологію як «... мережу, в якій вершинами є поняття, виражені лексичними засобами, а відносини відображають зв'язку теоретико-множинного або неформально-семантичного характеру». Незважаючи на відсутність єдиної думки про значення терміна

«Онтологія», розробка та використання подібних структур здатне допомогти у вирішенні ряду складних завдань. Так, в роботі [86] наводяться наступні можливі варіанти використання онтологій: комунікація (Розробка та спільне використання формально визнаного експертами предметної області мови); інтеграція розподілених і різнорідних ресурсів інформації; аналіз даних і логічний висновок по термінології. Аналіз знань та логічний висновок, який здійснюється за допомогою онтології, у край затребуваний для вирішення практичних завдань. Розвиток цієї галузі досліджень дозволяє проектувати і створювати експертні системи і системи підтримки прийняття рішень в слабоструктурованих складних предметних областях. Прикладом зазначеного використання онтологій є експертні системи в клінічній медицині (проект HEARTFAID Project і розроблена в рамках нього онтологія Heart Failure Ontology [40]; Virtual Medical Doctor [20]; PHASES - Personal Health Assistance Expert System). Крім медицини, онтології успішно застосовуються і в технологічних експертних системах. Приклад – Onto WEDSS System - система управління процесом очищення води на водоочисних станціях [63] з модулем міркувань з використанням онтології WaWO (Waste Water Ontology). російськими вченими розроблена COMT - система оперативного моніторингу технологічної інфраструктури, що застосовується для зниження енергетичних витрат нафтогазовидобувних підприємств і підвищення їх екологічної безпеки [83]. У дослідженнях, проведених Інститутом проблем управління складними системами РАН, описується підхід до розробки інтелектуальних систем, в тому числі систем підтримки прийняття рішень (СППР), засобами gB-системи онтологічного моделювання, містить засоби для розробки та візуалізації онтологій, здійснення логічного висновку, зберігання знань і даних. Такий підхід реалізований в СППР при управлінні групою енергосервісних компаній і при управлінні поданням консолідованих послуг в сфері охорони здоров'я і соціального розвитку Самарської області [103]. В сфері економіки та управління онтології також знаходять застосування, наприклад, у вигляді СППР з управління компетенціями організацій на основі онтології компетенцій. Вищевикладені приклади підтверджують, що одним з найбільш значущих застосувань онтологій предметних областей є використання їх в різних інтелектуальних інформаційних системах. Перевагою таких систем є те, що онтологія дозволяє оперувати загальновизнаною колом експертів для

даної предметної області термінологією, будуючи запити і отримуючи на виході якісь логічно несуперечливі затвердження щодо понять, що входять в структуру онтології. Однак для створення зазначених інтелектуальних інформаційних систем, особливо призначених для підтримки прийняття рішень в складних і слабоструктурованих областях, потрібно грамотно побудована, максимально повна і об'єктивна онтологія предметної області. Рішенням завдань ефективного побудови нових і модернізації існуючих і розвитку існуючих онтологій займається відповідне науковий напрям – онтологічний інжиніринг. Відзначимо, що готові онтології предметних областей найчастіше об'єднуються в цілі сховища (OntologWiki, KWTR, Protégé Ontologies, Schemapedia і інші [46, 49-52, 56, 58]). Крім сховищ, на сьогоднішній день існують пошукові системи, орієнтовані на онтології [15, 66]. Однак найчастіше оцінити ступінь достовірності, охоплення предметної області, актуальності та відповідності необхідної тематики містяться в онтології відомостей може бути важко. також існує проблема надмірності готових онтологій - коли для вирішення конкретних задач необхідна тільки частина онтології предметної області, що містить вузьке коло концептів, але більш докладно відображає всі їх взаємозв'язку, а наявна онтологія охоплює предметну область більш широко. Основна частина існуючих онтологій предметних областей розроблена на англійській мові і непридатна для використання в російськомовних додатках, так як переклад онтологій утруднений через істотні відмінностей в синтаксисі, семантикою і прагматиці мов. Оптимальним рішенням для подальшого використання в інтелектуальних інформаційних системах є розробка власної онтології предметної області. Однак в даній області досліджень також є невирішені проблеми. Зокрема, в роботі [27] відзначається, що побудова онтологій часто є складний, тривалий і витратний процес, що вимагають істотних зусиль з боку фахівців-учасників. Перш за все, це розробники, що професійно володіють технологіями інженерії знань

- від методів вилучення знань до структурування і формалізації [73]. Також для максимального повного і адекватного відображення знань про предметної області в розробці онтологій повинні також брати участь експерти. У більшості випадків, коли предметна область не пов'язана безпосередньо з інженерією знань, експерти не мають необхідними для розробника онтології

навичками. Відповідно, для створення онтології в даному випадку буде потрібно участь значного числа фахівців, до яких пред'являються істотні професійні вимоги. В процесі розробки онтології при використанні практично будь-який з існуючих методик велика частка «ручного» праці експертів предметної області і розробників. Більшість операцій по виявленню концептів, їх зв'язків, побудови таксономії, доопрацювання прототипу онтології виконуються без допомоги будь-яких автоматизованих коштів. Це ускладнює розробку онтології і збільшує ймовірність того, що підсумкова онтологія буде недостатньо адекватна вихідним вимогам. З огляду на зростаючу роль онтологій в розвитку семантичного вебу і інтелектуальних інформаційних систем, актуальним стає питання про формалізацію і подальшої автоматизації процесу побудови онтологій предметних областей. На сьогоднішній день відсутні розвинені методи і засоби, що дозволяють з мінімальними зусиллями отримувати прототипи онтологій предметних областей і використовувати їх в інформаційних системах. У сукупності перераховані проблеми побудови онтологій можуть істотно затримати і ускладнити розробку заснованих на них інформаційних систем. Як наслідок, в процесі розробки відомості про предметної області, відображені в онтології, втрачають свою актуальність і повноту, зростає вартість розробки системи. Це, в свою чергу, знижує ймовірність успіху всього проекту в цілому. перераховані аспекти розвитку онтологічного інжинірингу обумовлюють актуальність даної роботи.

1.3. Огляд існуючих методів і підходів до побудови онтологій

В ході становлення онтологічного інжинірингу як самостійного наукового напрямку сформувався ряд підходів до питання побудови онтологій. Вони розглянуті в декількох базових дослідженнях, присвячених теоретичним аспектам створення онтологій - виділенню найважливіших етапів побудови онтології, аналізу їх особливостей і розробці алгоритму їх проходження, а також практичним питань реалізації цих етапів. У літературі зустрічаються приклади порівняння існуючих методів і підходів до побудови онтологій по ряду критеріїв. Так, в роботі [7] розглядається характеристика методологій створення онтологій з точки зору їх «зрілості» і відповідності методологій розробки програмного забезпечення та інженерії знань. Автори робіт [16, 17] аналізують відомі методології на предмет відповідності стандарту IEEE Standart for Developing Software Life Cycle Processes 1074-1995, а також

проводять порівняння методологій побудови онтологій за кількома ознаками. За аналогічним ознаками в роботі [21] наводиться характеристика більш широкого кола методологій. Однак в зазначених дослідженнях не розглядається будь-яка класифікація; опису кожної з методологій наводяться у вигляді коментарів по кожному з визначених ознак. Нами пропонується класифікація, заснована на найбільш істотних, на наш погляд, ознаках.

Відповідно до цієї класифікації всі існуючі методології розробки онтологій можна умовно розділити на наступні групи.

1. За типом розробки онтології:

- методології централізованої розробки;
- методології децентралізованої розробки.

2. За ступенем залежності від програми:

- залежні;
- напівзалежні;
- незалежні.

3. За типом використовуваної моделі життєвого циклу (ЖЦ) онтології [62]:

- без вказівки моделі ЖЦ онтології;
- з ітеративною моделлю ЖЦ онтології;
- з моделлю ЖЦ онтології на основі еволюційного прототипування.
- з збігається з моделлю ЖЦ програми.

4. По можливості формалізації:

- передбачають способи формалізації;
- які не передбачають формалізації.

5. За типом вихідних даних, що запускають процес створення онтології:

- на основі питань компетенцій;
- на основі опису сценаріїв (варіантів використання);
- на основі моделі предметної області;
- на основі вихідної онтології.

6. За стратегією виділення концептів предметної області:

- знизу-вгору (bottom-up);
- зверху-вниз (top-down);
- від середини (middle-out);
- поєднують різні стратегії.

Розглянемо докладніше існуючі методології та підходи до створення онтологій. Одними з перших в 1995 році Ушолд і Кінг запропонували методологію побудови онтологій, зараз іменується «Скелетної» [69]. Дана методологія має низку недоліків. В Зокрема, вона не передбачає розвиток онтології відповідно до яких- або життєвим циклом. Це робить скрутним всі подальші процеси онтологічного інжинірингу, що протікають після етапу безпосереднього створення онтології - наприклад, розширення, модифікації, порівняння і злиття з іншими онтологіями. Автори розглянутої методології не призводять точного опису необхідних для побудови онтології технік і дій. В роботі [69] дано лише деякі приклади можливих дій. Наприклад, для виділення концептів предметної області з метою включення в онтологію може бути використаний метод мозкового штурму. Однак автори відзначають, що дана частина їх роботи, що описує деякі подробиці процесу побудови онтології, не є формальним керівництвом, а наведені приклади не можуть вважатися кращими в порівнянні з іншими підходами. Також Ушолдом і Кінгом не пропонується жодного можливого способу формалізації знань, отриманих в ході розробки онтології. Однією з перших робіт в області побудови онтологій є також методологія, запропонована Грюнінгером і Фоксом [26]. Особливістю даної методології є її Напівзалежна від додатки характер - розробляється за допомогою такої методології онтологія буде обмежена в застосуванні сценаріями, визначеними додатком. Також автори не наводять вказівок про конкретні техніках і діях, що застосовуються в рамках методології - наприклад, залишається неясним, як саме повинні формуватися питання, на основі яких розроблятиметься онтологія. Таким чином, розглянуті дослідження не пропонують конкретних методик для окремих етапів побудови онтології – наприклад, для виділення концептів предметної області та визначення зв'язків між ними, що відзначається, зокрема, в [104]. частково ця проблема вирішується в ряді інших досліджень. Так, досить широку популярність здобула методологія METHONTOLOGY, розроблена в Наприкінці 90-х років іспанськими дослідниками [18, 19, 22, 24]. Ця методологія створювалася на основі стандарту IEEE 1074-1995 [33], досить детально описує всі етапи створення онтологій, є незалежною від конкретного додатка і представляє життєвий цикл онтології у вигляді розвитку прототипів. На сьогоднішній день саме METHONTOLOGY є найбільш зрілим,

сформованим підходом, на думку авторів статті [4]. Однак, незважаючи на гнучкість і відносну повноту, METHONTOLOGY не уточнює окремі подробиці підпроцесів створення онтології: автори не дають уточнень або рекомендацій в області технік контролю якості розробок. також, якщо аналізувати відмінності між METHONTOLOGY і лежачим в її основі стандарту IEEE 1074-1995, можна зробити висновок про те, що в даній методології відсутній опис стадії, що передує безпосередньої розробки. Ще одним дослідженням, які мають безпосереднє відношення до проблеми розробки методології побудови онтологій, є проєкт EU IST-1999-10132 On-To-Knowledge. В його рамках розроблена однойменна методологія, скорочена назва якої - ОТКМ. Основною спрямованістю і призначенням даної методології є побудова систем управління знаннями, заснованих на онтологіях («an ontology based KM applications»). Зокрема, ОТКМ націлена на так звані метапроцеси знань - процеси, що управляють розробкою і початковою налаштуванням (установкою) онтологій [64]. Основною особливістю, і, на наш погляд, недоліком методології On-To-Knowledge є її спеціалізація для розробки онтологій для централізованих систем. З урахуванням зростаючої потреби в розподілених, найчастіше децентралізованих додатках, в тому числі і заснованих на знаннях, застосування даної методології може бути вкрай обмеженим. також, оскільки авторами мається на увазі циклічність етапів доопрацювання (Удосконалення), оцінки і застосування і розвитку, виникає проблема прийняття рішення про необхідність кожного наступного циклу. Авторами не конкретизує будь-які методики або способи такого аналізу, отже, не визначені будь-які «точки зупину» подібного алгоритму розробки онтології. Процес онтологічного інжинірингу стає нескінченним, а оцінка ступеня повноти, адекватності та можливості використання онтології для вирішення проблем на будь-якому етапі ускладнюється. Серед безлічі розроблених методологій побудови онтологій виділяється методологія, заснована на підході SENSUS. Відмінною особливістю даної методології є те, що нова онтологія будується шляхом додавання концептів і подальшого скорочення вже існуючої, заздалегідь заданої онтології. В якості такої онтології використовується SENSUS - онтологія, розроблена в інституті ISI (Information Science Institute) і призначена для використання в операціях на природній мові. Спочатку вона призначалася для застосування в розробках в

області машинного перекладу [43, 44] і на початковому етапі розробки представляла собою результат злиття двох лінгвістичних онтологій верхнього рівня - PENMAN Upper Model [1] і ONTOS. Послідовність кроків, виконуваних для отримання нової онтології на основі онтології SENSUS, наводиться в роботі [65].

Негативними рисами методології, заснованої на підході SENSUS, є:

1. Досить високі вимоги до творців онтології (вони повинні мати уявлення про предметну область, достатню для прийняття адекватного рішення про включення тих чи інших концептів в онтологію).

2. Відсутність згадок про конкретні методики виконання кроків побудови онтології. Так, наприклад, автори не уточнюють, з якого принципом або відповідно до якими критеріями необхідно відбирати вихідні концепти-«насіння».

3. Значна ступінь суб'єктивного впливу учасників розробки на результат процесу створення онтології. Зокрема, процес прийняття рішень про необхідність включення в онтологію вкладеного дерева концептів того чи іншого вузла не формалізовані і не регламентовані жодними методиками. Відповідно, зростає ступінь суб'єктивності рішень експертів або творців онтології. Групою дослідників на чолі з Амайя Бернерес в рамках проекту The ESPIRIT KACTUS був розроблений підхід, що передбачає тісну взаємозв'язок між розробкою онтологій і додатків. Точніше, онтології пропонується будувати на основі («витягувати») з баз знань конкретних додатків за допомогою методу «знизу-вгору», тобто шляхом абстракції концептів. Чим більше додатків було задіяно при створенні і розвитку онтології, тим більше загальної та всеосяжної вона стає. Автори підходу KACTUS в роботах [4, 59] не призводять будь-яких конкретних способів формалізації одержуваних і оброблюваних знань. У поєднанні з фактом відсутності чітких і докладних алгоритмів виконання окремих етапів побудови онтології, це ускладнює застосування методології KACTUS на практиці. Фактично, авторами наводиться лише один приклад використання розробленого ними підходу - побудова онтології в області електричних мереж [4]. В середині 2000-х років був розроблений підхід до побудови онтологій, названий DILIGENT (Distributed, Loosely-controlled and evolving Engineering of Ontologies) [54]. Дана методологія передбачає п'ять основних етапів (кроків) створення

онтології: безпосередньо побудова, локальна адаптація, аналіз, ревізія і локальне оновлення [18]. Відмінною особливістю Diligent є відсутність вимог до повноти первісної онтології. Вона може бути досить невеликий, що сприяє простоті процесу подальшої розробки. Далі користувачі працюють з створеною вихідною онтологією локально, вносячи необхідні, на їхню думку, зміни. Незважаючи на широке поширення, яке отримала дана методологія, вона володіє, на наш погляд, істотним недоліком. Для її реалізації необхідно участь великої кількості фахівців у різних сферах – безпосередніх творців онтології, рядових користувачів, експертів в предметній області. При цьому практично всі етапи виконуються учасниками «Вручну», евристичним шляхом, без застосування автоматизованих коштів або формальних підходів. Ще одним мінусом методології Diligent може вважатися неповнота первісної онтології, з якої починається робота. Фактично, первісна онтологія не завжди може бути використана за призначенням, так як недостатньо охоплює предметну область.

При цьому спрогнозувати, коли учасники розробки доповнять вихідну онтологію усіма необхідними для вирішення поставлених завдань термінами, їх визначеннями і взаємозв'язками, вкрай важко. Окрема група досліджень, присвячених підходам до створення онтологій - це методи побудови онтологій за допомогою машинного навчання. Більшість досліджень в цьому напрямку присвячені використанню методів лінгвістичного аналізу текстів для подальшого автоматичного побудови онтологій [5, 42, 48, 101]. У ряді робіт розглядаються методи створення онтологій на основі колекцій документів [32, 41, 96]. Однак у підходів, заснованих на побудові онтологій за результатами лінгвістичного аналізу текстів, є ряд істотних недоліків. Так, за твердженням Є.А. Єрмакова [79], однією з основних причин відсутності на сьогоднішній день «... прикладних програм, що використовують «онтології в сильному сенсі» є недосконалість алгоритмів аналізу (вилучення знань з текстів) і спірне якість джерел інформації (текстів). У зв'язку з цим виникає необхідність значного втручання експерта в процес онтологічного інжинірингу, зокрема, верифікації та редагування прототипу онтології, отриманого після роботи тих чи інших засобів автоматизації даного процесу. Джерела інформації, тобто аналізовані за допомогою алгоритмів тексти, є, по суті, виразом суб'єктивної думки автора про предметну область. При цьому неможливо достовірно

визначити, чим керувався автор при написанні тексту з тієї чи іншої проблеми, висловлює він загальноприйнятну або альтернативну точку зору по проблемі.

Відповідно, при використанні для побудови онтологій предметної області методів лінгвістичного аналізу велика ймовірність отримати недостовірний прототип онтології, що містить помилкові визначення або зв'язку між термінами. подальше редагування онтології експертом може лише посилити проблему. Ще однією слабкою стороною створення онтологій за допомогою методів лінгвістичного аналізу текстів є дотримання принципу розширюваності, сформульованого в [25]. Під цим принципом мається на увазі, що в онтології повинен бути передбачений «механізм розширення (обмеження) спільно використовуваних словників понять без порушення цілісності системи» [100]. А в більшості програмних систем видалення знань з текстів, згідно з зауваженням авторів [93], відсутня модель предметної області, що забезпечує використання знань експерта і розширення моделі в процесі роботи. У зв'язку з цим виникає гіпотеза про використання моделі предметної області (моделі даних) як якоїсь основи онтології, служить «опорою» для побудови і подальшого розширення онтології, її базисом. Порівняння понять і сутності онтології і моделі даних розглядається, наприклад, в роботах [37, 61]. Автори сходяться на думці, що поняття онтології можна вважати еквівалентним поняттю моделі даних, оскільки є суттєва відмінність між цими поняттями. Одним з призначень онтології є забезпечення інтеграції інформаційних ресурсів, обмін даними між різними системами. Тоді як ні концептуалізація, ні словник моделі даних априорі не можуть бути використані в подібних цілях, оскільки модель даних специфічна для кожного конкретного випадку, в рамках якого розробляється. Деякі автори, наприклад, [85], вважають онтологію предметної області формальним описом результатів моделювання. Так, модель даних і онтологія мають ряд спільних рис. Як правило, модель даних представляє в деякому формалізованому вигляді структуру предметної області. Довільну підсистему онтології, що представляє собою набір термінів предметної області та зв'язків між ними без урахування аксіом, забезпечують інтерпретацію і правильне використання цих термінів, можна також вважати описом структури предметної області. Оскільки один з початкових етапів безпосереднього створення онтології передбачає вибір концептів, а також визначення існуючих

між ними зв'язків, можна припустити, що подібний вибір може здійснюватися на підставі результатів моделювання предметної області. Одним з перших підходів, які передбачають використання для побудови онтологій результатів аналізу предметної області у вигляді деякої моделі, став Фасетное підхід Рубена Пріето-Діаз. Даний підхід передбачає використання поєднання висхідній і низхідній методик проектування онтології, а також включає в себе застосування методів аналізу предметної області. Фасетний підхід, за твердженням авторів, може істотно спростити завдання кодування онтології і подальшого документування. Однак одержувані за допомогою такого підходу структури можна вважати лише базисом для подальшого побудови онтологій, оскільки результат не включає формальні визначення понять і аксіом. Тому необхідні подальші дослідження питання перетворення моделей предметної області в повні формалізовані онтології, що відзначається і самими авторами [55]. В роботі [12] представлений підхід до побудови онтології, використовує на одній зі стадій абстрактну модель предметної області, а саме use-case-діаграми на мові UML. Це - методологія UPON (Unified Process for ONtology Building). Вона призначена для підтримки діяльності творців онтології - інженерів по знаннях і експертів предметної області [12, 13]. Зазначена методологія побудови онтологій розроблена відповідно до так званим уніфікованим процесом розробки програмного забезпечення (Software Development Unified Process, UP) [36]. Автори [12] підкреслюють, що процес побудови онтології відповідно до підходу UPON узгоджується з UP, завдяки чому можна уявити створення онтології, по аналогії з розробкою програмного забезпечення, у вигляді деяких об'єднаних кроків (стадій). Зокрема, методологія UPON увазі ієрархічне представлення процесу створення онтології - у вигляді циклів, які складаються з чотирьох фаз, кожна з яких поділяється на ітерації, а ті, в свою чергу - на роботи. При цьому процес розробки онтології приймає ітеративний і інкрементальний характер, що, незважаючи на переваги (підвищення якості, гнучкості і масштабованості результативною онтології), істотно збільшує навантаження на всіх учасників процесу - експертів, предметної області, розробників, інженерів по знаннях. Для підтримки розгляду окремих положень проекту розробки онтології в методології UPON пропонується використовувати мову UML. Слід зазначити, що використання UML в процесі створення онтологій має передісторію. Так,

в роботах [3, 8, 9, 45] була розглянута ідея використання UML як мови представлення онтологій. Однак авторами статті [28] була висловлена думка, що UML не є відповідним мовою для подання онтологій, але може бути використаний в цілях проектування процесу створення онтології. Застосування абстрактного моделювання з використанням мови UML, незважаючи на наявність позитивних моментів, на нашу думку, володіє істотними негативними сторонами. Так, у випадку з методологією UPON здійснюється побудова use case-діаграм з метою досягнення угоди між користувачами, експертами і в рамках розробки вимог. Незважаючи на те, що UML-діаграми в даному випадку здатні забезпечити простоту і наочність представлення цілей створення онтології, процес розробки онтології спочатку виявляється спрямованим на вирішення низки певних завдань, фактично сформульованих в розроблених use case - діаграмах. При цьому по своїй суттю онтологія передбачає багаторазове використання різними додатками.

Отже, при побудові онтології з використанням use-case діаграм виникає проблема застосовності онтології для вирішення нових, заздалегідь невизначених задач. Резюме характеристик основних існуючих методологій побудови онтологій відповідно до розробленої нами класифікацією можна представити в наступному вигляді (таблиця 1.1).

Таким чином, кожен з розглянутих підходів до побудови онтологій володіє тими чи іншими недоліками.

Основними невирішеними проблемами залишаються:

- забезпечення максимальної об'єктивності і повноти створюваної онтології;
- зниження ролі людського фактора при розробці онтології;
- розробка формалізованих методів побудови онтологій і їх подальша автоматизація;
- забезпечення застосовності онтології для використання різними додатками, в тому числі для заздалегідь не визначених завдань.

Завдання забезпечення можливості використання онтології для вирішення різних завдань і використання її декількома різними додатками доповнюється логічно впливають із неї вимогами. Так, наприклад, в [39] відзначається, що опис методик розширення (Доповнення) існуючих онтологій, а також принципи вибору підходящої для вирішення конкретного завдання онтології з бібліотеки повинні бути частиною методології побудови

онтологій. В існуючих методологіях рішення подібних завдань або не наводиться, або описується поверхнево, що обумовлює необхідність більш ретельного дослідження даного питання. Виходячи з наведених фактів, завдання розробки ефективних методів і засобів побудови онтологій предметних областей є по- Як і раніше актуальною. Це обумовлює робочу гіпотезу, прийняту в рамках даної роботи - використовувати для побудови онтологій УФО- моделі предметних областей, одержувані за допомогою застосування УФО- підходу.

1.4. Огляд і аналіз системно-об'єктного УФО-підходу та результатів УФО-моделювання

Альтернативою «ручному» способу побудови онтології, при якому безпосередній відбір понять ведуть експерти, може стати використання на початковому етапі для відбору понять і створення прототипу онтології вже існуючих або розроблюваних моделей предметної області. Аналіз і моделювання різних предметних областей на сьогоднішній день є більш розвиненою областю досліджень по порівняно з онтологічним інжинірингом. У тому чи іншому вигляді моделі даних присутні практично в будь-якому бізнес-процесі, в будь-якій предметній області. Існуючі методи і засоби дозволяють здійснювати моделювання без попереднього отримання спеціальних навичок. Це означає, що в найпростішому вигляді побудувати модель даних предметної області може не тільки фахівець-аналітик, а й експерт предметної області, що не володіє знаннями в області аналізу і моделювання бізнес-процесів. Одним з ефективних рішень задачі аналізу і моделювання предметної області є використання коштів так званого УФО-підходу. Системно-об'єктний підхід «Вузол-Функція-Об'єкт» (УФО-підхід) запропонований і розроблений С.І. Маторіну. Основні положення підходу описані в роботах [88, 89], і в подальшому отримали розвиток в [91, 92]. В основі УФО-підходу і заснованого на ньому так званого УФО- аналізу лежить уявлення тієї чи іншої системи у вигляді трьохелементної конструкції «Вузол-Функція-Об'єкт». Вузол - це уявлення системи як структурного елементу, «перехрестя» всіх її зв'язків. під функцією розуміють функціональний елемент, який грає певну, задану роль для підтримки надсистеми шляхом балансування даного вузла. Об'єкт представляє систему як субстанціальний

елемент, конкретну сутність, що реалізовує функцію [91]. Комбінації з таких УФО-елементів є системно-об'єктні моделі (УФО-моделі). Подання деякої предметної області у вигляді УФО-моделі забезпечує найбільш цілісне, системне уявлення його призначення, суті, положення в структурі надсистеми і зв'язку з іншими елементами в цілому, адже УФО-модель містить в собі інформацію як про структурні, так і про функціональні характеристики модельованої системи. УФО-модель - результат застосування УФО-аналізу - є адекватною і досить повною моделлю предметної області; в подібній моделі легко виділити основні концепти (терміни, сутності), їх характеристики (Атрибути, властивості) і роль в системі, а також взаємозв'язку таких концептів. В УФО-моделі відображаються тільки необхідні для опису предметної області терміни, і саме в тому значенні, в якому використовуються безпосередньо в предметній області. Побудова УФО-моделі предметної області не вимагає значних трудових і тимчасових витрат. Метод УФО-аналізу дозволяє включати в результативну модель інформацію про предметну область, представлену з різних точок зору, що забезпечує максимально можливу об'єктивність відображення знань про предметну область. Підсумкова УФО-модель має властивість масштабованості і легко модифікується при змінах в предметній області. На відміну від інших поширених методологій системного і об'єктного моделювання, УФО-підхід має широкі можливостями формалізації. На сьогоднішній день єдиний стандартний метод формалізації УФО-моделей не затверджений, але в ряді досліджень розглядаються різні способи формалізації результатів УФО-аналізу. Так, в роботах [94-95] наводиться опис методу формалізації УФО-моделей за допомогою пі-обчислення. Авторами робіт [78, 80] для створення математичного апарату УФО-підходу застосовується теорія патернів Гренандера [77], що дозволяє формалізувати опис вузлових (Структурних) характеристик систем як УФО-елементів. Ще один спосіб формалізації УФО-підходу запропонований в роботі [84]. Кожен з перерахованих методів дозволяє розробити найбільш підходящий для вирішення конкретного завдання формальний математичний апарат УФО-моделей, що забезпечує гнучкість і, в той же час, сувору визначеність методу УФО-аналізу. Всі перераховані особливості обумовлюють вибір УФО-аналізу і одержуваних в його результаті УФО-моделей в якості базису для розробки

прототипу онтологій предметних областей. Зокрема, передбачається використовувати УФО-моделі предметної області як джерела концептів (Термінів) і пов'язаних з ними фактів, які потім можна записувати на тій чи іншій мові уявлення онтологій. Дійсно, УФО-підхід являє кожен сутність предметної області у вигляді УФО-елемента. Ім'я його вузла є окремим терміном, який відбивається в онтології, а опис може служити значенням цього терміна. Як зв'язків між концептами в онтології можуть виступати вхідні та вихідні зв'язку, які притаманні УФО-елементу. У більшості випадків мови представлення онтологій засновані на записи всієї інформації у вигляді фактів. У випадку з УФО-моделлю фактами можуть служити наявність у самого УФО-елемента конкретної функції, у його вузла - певних зв'язків, а у об'єкта - ряду властивостей. Відповідно, з формалізованого представлення УФО-моделі можливо отримати запис фактів на тій чи іншій мові уявлення онтологій. Сукупність таких фактів, представлених за допомогою формальної мови, дозволяє отримати прототип онтології предметної області. Цей прототип може бути доповнений і/або скоректований як вручну експертами предметної області при необхідності, так і автоматично при зміні базової УФО-моделі. Важливою особливістю є відповідність описаного підходу положенням про життєвий цикл онтології, сформульованим в комюніке онтологічного саміту - 2013 [97]. Незважаючи на те що висновки, що містяться в зазначених положеннях, можуть вважатися лише попередніми, про що згадується в резюме документа [97], ідеї про життєвому циклі онтології можуть істотно спростити, систематизувати і уніфікувати процеси розробки онтологій для всіх їх учасників. Зокрема, модель життєвого циклу онтології, в Відповідно до [97], може бути схематично представлена у вигляді малюнка 1.1. Відповідно до такого життєвим циклом, при розробці онтології за допомогою засобів УФО-підходу етап формулювання вимог і етап онтологічного аналізу виконуються ще в рамках безпосереднього УФО-аналізу предметної області. Етап проектування фактично пропускається, оскільки кошти уявлення онтології визначені заздалегідь, а структура онтології залежить від вихідної УФО-моделі предметної області.

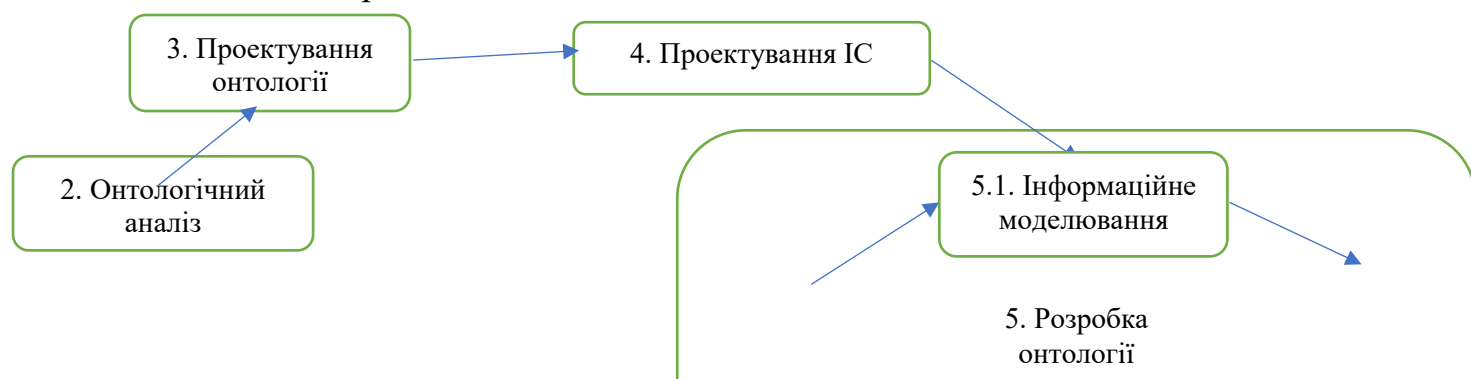


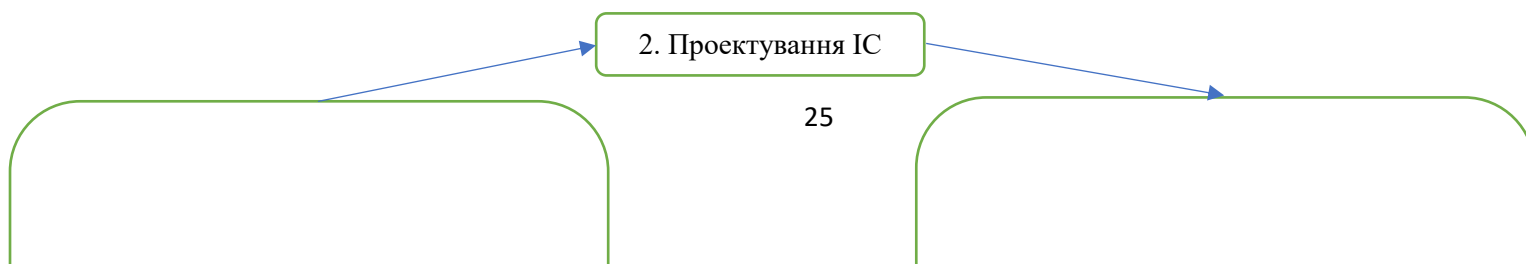


Рисунок 1.1 - Модель життєвого циклу онтології згідно Комюніке Онтологічного саміту-2013

Етап безпосередньої розробки онтології, відповідно до документом [97], підрозділяється на чотири дії. У рамках інформаційного моделювання може здійснюватися доопрацювання вихідної УФО-моделі. Це особливо актуально для тих випадків, коли для побудови онтології вибирається вже існуюча, а не спеціально розроблена УФО-модель предметної області. Формального моделювання відповідає процес формальної записи фактів, витягнутих з УФО-моделі. Подальша дія операційної адаптації при необхідності може виконуватися експертами вручну або повертати онтологію до дії інформаційного моделювання і повторювати наступні дії. Відповідно, життєвий цикл онтології, розробленої з використанням засобів УФО-підходу, можна представити таким чином (рисунок 1.2).

Таким чином, перевагами описаного підходу є:

1. Автоматизація значної частини трудомістких операцій по розробці онтології.
2. Зниження ступеня суб'єктивності розробляється онтології.
3. Забезпечення гнучкості і масштабованості розробленої онтології.
4. Дотримання принципу можливості використання розробленої онтології різними додатками для вирішення різних завдань.



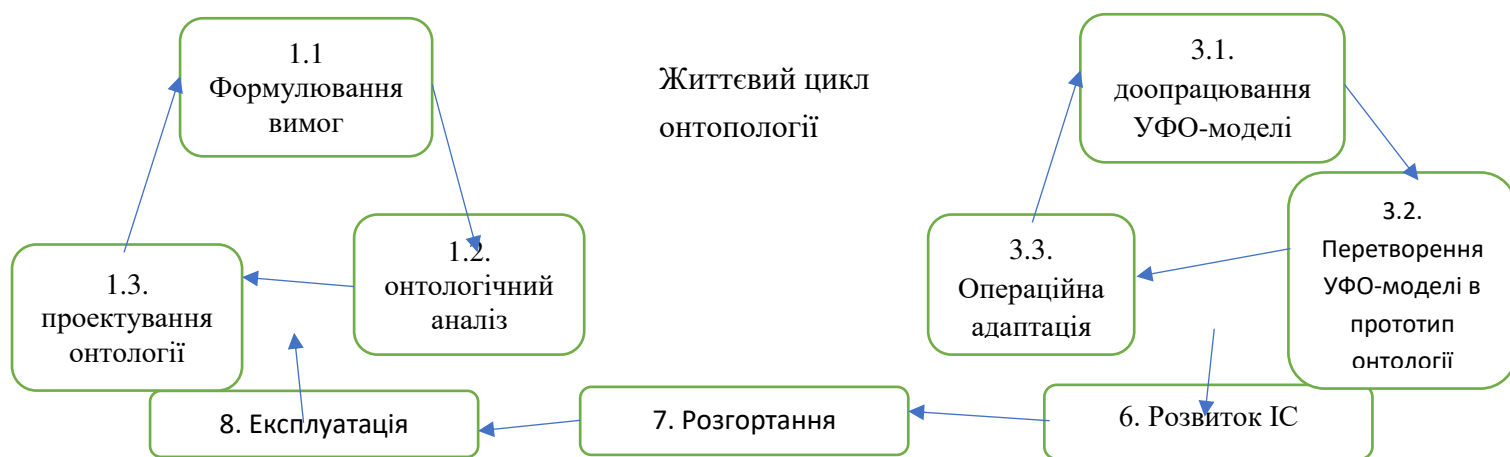


Рисунок 1.2 - Модель життєвого циклу онтології, побудованої з використанням коштів УФО-підходу.

Формалізація і суворе визначення алгоритму побудови онтології на основі УФО-моделі предметної області забезпечує можливості для автоматизації основних процесів, таких як відбір концептів для включення в онтологію і визначення їх взаємозв'язків. Зазначені операції можуть здійснюватися за допомогою автоматизованих засобів перекладу знань, відображених в УФО-моделі, на мову уявлення онтології, і не вимагати ручної праці експертів. При цьому знімається проблема вибору джерел фактів, відома для ряду існуючих методик [5, 42, 48, 101] - в якості такого джерела виступає єдина УФО-модель, яка з легкістю адаптується під зміни в предметній області.

Таким чином, наступні зміни, коригування та доповнення можуть бути внесені в онтологію також за допомогою автоматизованих засобів за рахунок тісного взаємозв'язку з УФО-моделлю предметної області, що зумовлює гнучкість і здатність до масштабування одержуваної онтології. В рамках проведення УФО-аналізу передбачається відображення фактів предметної області не з однією з в принципі можливих точок зору, як, наприклад, при використанні методології IDEF0 [34], а з єдиної, представляє собою консенсус думок експертів і аналітиків, точки зору. Це в повній мірі відповідає визначенню онтології і сприяє зниженню ступеня суб'єктивності виявляються фактів. УФО-модель предметної області, яка береться за основу для розробки онтології, спочатку орієнтована не на конкретний сценарій застосування, а на максимально повне уявлення структурних і функціональних характеристик предметної області. Таким чином, факти, витягнуті з УФО-моделі і представлені в формалізованому вигляді, можуть бути використані будь-якими додатками і не вимагають додаткової обробки і перетворень.

1.5. Постановка завдання побудови онтологій предметних областей на основі результатів візуального комп'ютерного моделювання

Таким чином, метою дисертаційного дослідження є вдосконалення засобів створення онтологій шляхом розробки нового методу автоматизованої побудови онтологій предметних областей на базі візуальних графоаналитических системно-об'єктних моделей для вирішення завдань подання знань організацій і підприємств та управління ними. Відповідно, для досягнення поставленої мети необхідно вирішити такі завдання.

1. Дослідити і систематизувати відомості про предметну області, що містяться в візуальних графоаналитических моделях, представляються за допомогою системно-об'єктного підходу «Вузол-функція-Об'єкт» (УФО-підходу).

2. Проаналізувати існуючі методи побудови онтологій і засоби їх формального опису. Виявити і обґрунтувати можливість побудови онтологій на основі візуальних графоаналитических моделей в термінах «Вузол-Функція-Об'єкт» (УФО-моделей).

3. Проаналізувати існуючі засоби алгебраїчного опису УФО-моделей і адаптувати їх для використання при перетворенні УФО-моделей в конструкції мов формального уявлення онтологій.

4. Розробити метод опису УФО-моделі на формалізованому мовою записи онтології.

5. Розробити метод побудови онтології на основі візуальної графоаналитического УФО-моделі.

6. Розробити метод логічного висновку на УФО-моделях з урахуванням інтеграції засобів УФО-підходу та онтологічного інжинірингу.

7. Спроектувати і реалізувати алгоритмічне та програмне забезпечення для побудови онтології на основі УФО-моделі і здійснення логічного висновку на УФО-моделях з урахуванням розроблених методів.

1.6. Висновки до розділу

В даному розділі обґрунтовується актуальність завдання розробки ефективних методів побудови онтологій предметних областей; наводиться розроблена класифікація, огляд і аналіз існуючих методів і підходів до побудови онтологій, формулюються їх недоліки. Для вирішення проблеми розробки ефективних методів побудови онтологій пропонується використовувати кошти системно-об'єктного УФО- підходу. Наводиться короткий опис названого підходу, розглядаються переваги його використання з метою створення онтологій. У заключній частині сформульовані основна мета і завдання дисертаційного дослідження.

РОЗДІЛ 2. ДОСЛІДЖЕННЯ І АДАПТАЦІЯ КОНЦЕПТУАЛЬНИХ І ФОРМАЛЬНИХ ЗАСОБІВ СИСТЕМНО-ОБ'ЄКТОВОГО ПІДХОДУ ДЛЯ ВИРІШЕННЯ ЗАВДАНЬ ОНТОЛОГІЧНОГО ІНЖИНІРИНГУ

2.1. Дослідження і систематизація відомостей про предметну область, що містяться в візуальних графоаналитических моделях, представляються за допомогою УФО-підходу.

УФО-модель містить в собі набір знань експертів про модель предметної області. Всі подібні знання, так чи інакше, можна витягти з УФО-моделі у вигляді окремих фактів - аксіом. У загальному випадку безліч подібних фактів, що містяться в УФО-моделі, підлягає класифікації, зокрема, поділу на кілька груп залежно від рівня (точки зору), на якому ці факти розглядаються. Розробка подібної класифікації фактів дозволяє формалізувати і строго визначити склад концептів і аксіом, знання про які повинні бути вилучені з вихідної УФО-моделі і надалі включені в онтологію. З урахуванням призначення розробляється класифікації (використання в процесі побудови онтології на базі УФО-моделі предметної області) пропонується застосувати в якості критерію класифікації рівень розгляду предметної області та її окремих складових з точки зору УФО-підходу. Так, окремі характеристики предметної області і її складових можна сформулювати при розгляді на рівні окремих УФО-елементів як вузлів, як функцій і як об'єктів, а також при аналізі зв'язків між УФО-елементами. Додаткова інформація, яка може бути включена в підсумкову онтологію предметної області, міститься в даних про декомпозиції окремих УФО-елементів моделі. Наведемо опис кожного факту, що включається в класифікацію, на природною мовою. Надалі з метою побудови онтології витягнуті з УФО-моделі факти про предметну область пропонується формалізовувати у вигляді конструкцій одного з загальноприйнятих мов уявлення онтологій. Зокрема, пропонується використовувати мову RDF (Resource Description Framework), що тягне за собою необхідність враховувати при розробці класифікації видобутих фактів ряд особливостей їх формулювання і уявлення. Так, слід

взяти до увагу, що відношення між об'єктом і суб'єктом в RDF-триплети є спрямованим. Отже, один і той же факт може бути записаний двома різними RDF-триплетами в залежності від того, яка сутність буде об'єктом, а яка - суб'єктом. Для зручності класифікації і подальшої формалізації приймемо, що в даному випадку мова йде про двох різних, але пов'язаних фактах про предметну область. такі пари фактів будемо позначати N і $N (inv)$ і відносити їх до груп з позначеннями G і $G (inv)$ відповідно.

Виділимо кілька груп і сформулюємо що входять до них факти з урахуванням наведених положень:

I. На рівні УФО-елемента (рисунок 2.1).



Рисунок 2.1 - I група фактів, які з УФО-моделі.

1. Вузол балансується функцією.
2. Функція реалізується за допомогою об'єкту.
3. Вузол зайнятий об'єктом.

I (inv). На рівні УФО-елемента (рисунок 2.2).

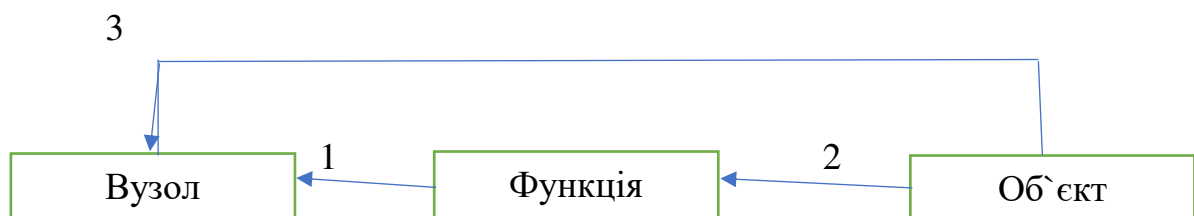


Рисунок 2.2 - I (inv) група фактів, які з УФО-моделі.

1. Функція балансує вузол. 40

2. Об'єкт реалізує функцію.

3. Об'єкт займає вузол.

II. На рівні Вузла (рисунок 2.3).

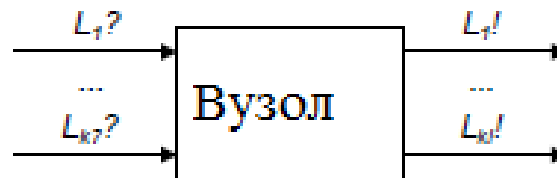


Рисунок 2.3 - II група фактів, які з УФО-моделі.

Позначимо кількість вхідних зв'язків вузла $k?$, кількість вихідних зв'язків - $k!$, кількість різних типів вхідних зв'язків - $t?$, кількість різних типів вихідних зв'язків - $t!$. тоді:

1. (Для кожного $m = 1..t?$) Вузол має вхідний зв'язок (порт) класу β_m

2. (Для кожного $m = 1..t!$) Вузол має вихідний зв'язок (порт) класу β_m .

3. (Для кожного $m = 1..k?$) Вузол має вхідний зв'язок $L_m?$

4. (Для кожного $m = 1..k!$) Вузол має вихідний зв'язок $L_m!$

5. (Для кожного $m = 1..k?$) Зв'язок $L_m?$ належить класу β_j , причому $j \in (1..t?)$

6. (Для кожного $m = 1..k!$) Зв'язок $L_m!$ належить класу β_j , причому $j \in (1..t!)$

III. На рівні Функції. 1. (Для кожного $m = 1..k?$) Функція перетворює вхід

$L_m?$ 2. (Для кожного $m = 1..k!$) Функція видає вихід $L_m!$

IV. На рівні Об'єкту. Позначимо кількість ознак (атрибутів) об'єкта q . тоді: 1.

(Для кожного $m = 1..q$) Об'єкт має ознаку α_m . V. На рівні зв'язків між двома

УФО-елементами (рисунок 2.4).

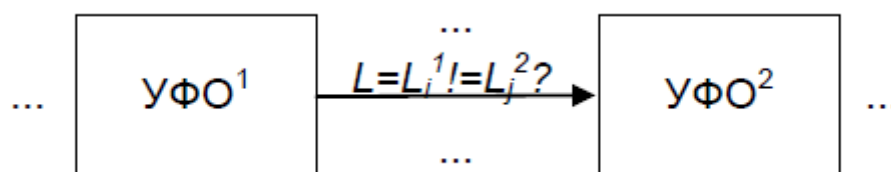


Рисунок 2.4 - V група фактів, які з УФО-моделі.

Позначимо кількість зв'язків між УФО-елементами УФО1 і УФО2 символом h . тоді:

1. (Для кожного $m = 1..h$) УФО1 передає L УФО2. V (inv). На рівні зв'язків між двома УФО-елементами (рисунок 2.4).

1. (Для кожного $m = 1..h$) УФО2 отримує L від УФО1

VI. На рівні декомпозиції УФО-елемента.

Якщо УФО-елемент УФО1 може бути декомпозований, тобто представлений у вигляді комбінації

УФО-елементів УФО2 ... УФО n , то:

1. УФО1 має в якості частини УФО n

VII. На рівні агрегації УФО-елементів.

З урахуванням вищенаведених положень:

1. УФО n є частиною УФО1.

Дана класифікація дозволяє визначити склад основних фактів про предметної області, які в загальному випадку можуть бути вилучені з будь-якої УФО-моделі і в подальшому використані в якості основи для побудови онтології. З метою стандартизації та автоматизації процесу вилучення зазначених фактів з УФО-моделі необхідно розглянути питання формального подання таких конструкцій. оскільки в рамках завдання побудови онтології на основі візуальної графоаналитического УФО-моделі передбачається запис витягнутих фактів за допомогою засобів формалізованого представлення онтологій, необхідно розглянути взаємозв'язок таких з математичним апаратом УФО-підходу.

2.2. Дослідження взаємозв'язку системно-об'єктних УФО-моделей і формалізованих засобів запису онтологій.

Крім неформального, вербального подання онтологій, вченими розроблено безліч засобів формального опису онтологій з використанням спеціалізованих мов: RDF, DAML + OIL, OWL, OCML, LOOM і ряду інших. Найбільш широке поширення отримав 42 OWL (Web Ontology Language) -

мова для визначення та подання веб- онтологій [53]. OWL дозволяє групувати інформацію в онтології і представляти її в якості документів. Надалі за допомогою спеціалізованого інструментарію може бути проведена обробка таких документів, що дозволяє отримати інформацію, закладену в онтології.

Однак вихідним синтаксисом обміну, базисом для OWL є RDF. RDF (Resource Description Framework) - це універсальна мова для представлення знань в Мережі. Він дозволяє представляти знання в самому Загалом вигляді, при цьому будучи «універсальним засобом для обміну даними між різними програмами» [29, 60]. В основі RDF лежить представлення фактів як триплетів виду «Суб'єкт - Ставлення (Предикат) - Об'єкт». RDF підтримує кілька нотацій, серед яких найбільш поширеними є:

- Turtle - компактний, зручний для сприйняття людиною формат.
- N-Triples - простий, в тому числі для машинного читання, «Лінійний» формат запису; видається менш компактним по порівняно з Turtle.
- N-Quads - являє собою розширення N-Triples для серіалізації множинних графів.
- JSON-LD - формат серіалізації даних RDF, заснований на форматі JSON.
- Notation3, або скорочено N3 - формат, схожий з Turtle, але володіє деякими додатковими можливостями, такими як можливість опису правил виведення.
- RDF / XML - перший формат для серіалізації RDF, заснований на синтаксисі XML.

Оскільки RDF є базисом для деяких інших, більш спеціалізованих для онтологій мов типу OWL, в контексті завдання розробки нового методу побудови онтологій, заснованого на принципах 43 і засобах УФО-аналізу, доцільно розглянути зв'язок УФО-підходу саме з RDF. На основі результатів дослідження робіт в області розробки системно-об'єктного підходу, можна виділити його основні принципи. Так, в основі УФО-підходу і базується на

ньому УФО- аналізу лежить уявлення тієї чи іншої системи у вигляді трьохелементної конструкції «Вузол-Функція-Об'єкт». Під вузлом розуміють систему у вигляді структурного елемента, «перехрестя» всіх її зв'язків. функція являє систему як функціональний елемент, який грає задану роль для підтримки надсистеми за допомогою балансування даного вузла. об'єкт представляє систему у вигляді конкретного освіти, субстанціонального елемента, що реалізує функцію [91]. Комбінації з подібних УФО- елементів є системно-об'єктні моделі (УФО-моделі). Таким чином, використання системно-об'єктного УФО-аналізу забезпечує поєднання функціонального і об'єктного моделювання в одній моделі. Це дозволяє підвищити об'єктивність і адекватність моделі предметної області, а також надає можливість частково автоматизувати аналітичну діяльність.

Дослідниками розроблені різні способи формалізації УФО-підходу, кожен з яких має свої особливості і використовується для вирішення певних завдань. Один з таких способів описаний в роботах [78, 80]. Його суть полягає в застосуванні для створення математичного апарату УФО-підходу теорії патернів Гренандера. Це дозволяє формалізувати опис вузлових (структурних) характеристик систем як УФО-елементів.

В основі розглянутого підходу лежить зіставлення утворює в теорії патернів з УФО-елементом в УФО-підході. Під утворює в теорії патернів [77] розуміється об'єкт, що володіє деякими ознаками α , а також вхідними та вихідними зв'язками (в свою чергу характеризуються деякими показниками β) [78]. відносно УФО- елемента, в даному випадку поняття «функція» і «об'єкт» можна розглядати як ознаки утворює, а поняття «вузол» використовувати для уявлення зв'язків. При цьому типи зв'язків вузла будуть показниками відповідних зв'язків утворює. подібне уявлення утворює і УФО- елемента графічно можна зобразити наступним чином (рисунок 2.5).



Рисунок 2.5 - Графічні формалізми - утворює і УФО-елемент.

В теорії патернів конфігурації являють собою структурні об'єднання утворюють. Розглянемо найпростішу конфігурацію, що складається з двох утворюють і одного з'єднання між ними. Уявімо графічно також два УФО-елемента, з'єднані деякої зв'язком, за аналогією з описаною конфігурацією. порівняємо отриману конструкцію з RDF-кодом (рисунок 2.6).

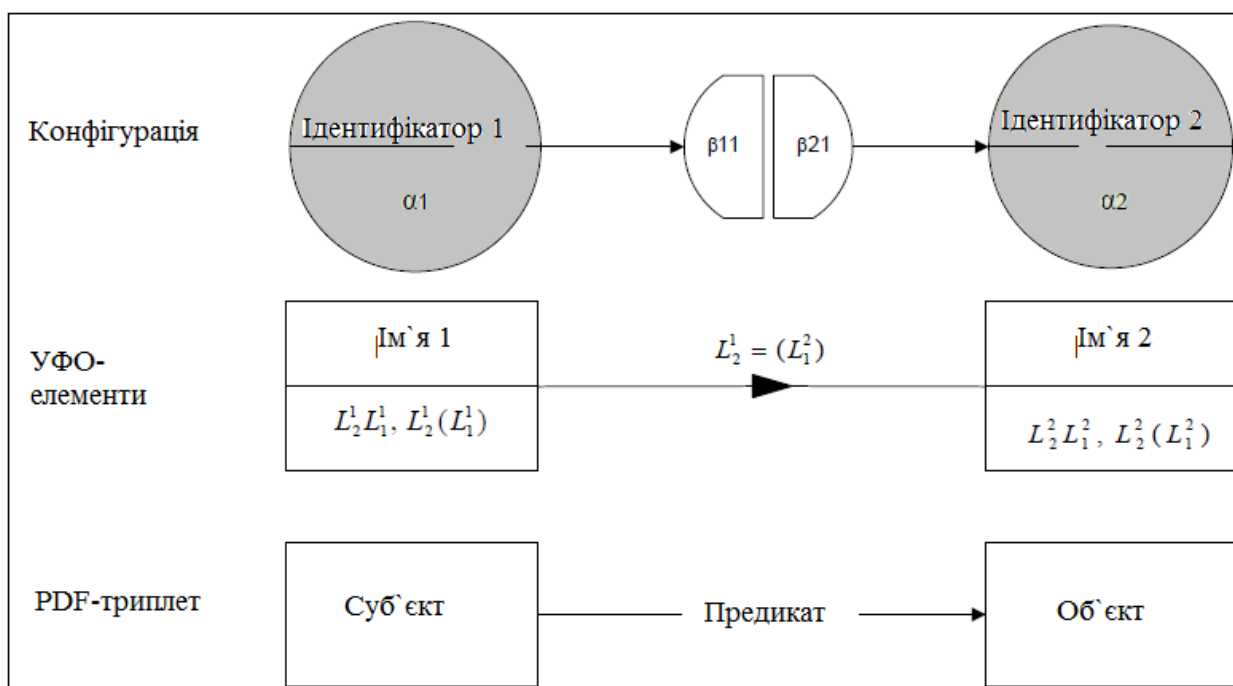


Рисунок 2.6 - Відповідність конфігурації, УФО-елементів та RDF-триплетів

Дані конструкції мають деякі спільні риси, що дозволяє зіставляти їх один одному. Так, в RDF-триплети суб'єкт і об'єкт представляються у вигляді імен конкретних або абстрактних сутностей. такі імена використовуються лише для забезпечення можливості посилатися на позначаються ними сутності і самі по собі не несуть будь-якого сенсу. Аналогічним ознакою є ідентифікатор у утворює. він також служить виключно для визначення, відмінності один від одного декількох утворюють. Кожен УФО-елемент має ім'я, яке дозволяє однозначно визначати його в межах моделі. При поданні УФО- елемента з використанням термінів теорії патернів ім'я елемента може служити ідентифікатором відповідної утворює. Звідси випливає, що імена

УФО-елементів можуть бути використані в якості суб'єктів і об'єктів в RDF-триплетах.

Важливою складовою RDF-триплетів є предикат - відношення між двома сутностями. При цьому особливо важливий порядок розташування суб'єкта і об'єкта. Тобто таке ставлення є спрямованим. це дозволяє зіставити предикату в RDF-триплети з'єднання, що входить в структуру конфігурації в термінах теорії патернів. При цьому, як видно з малюнка 2.6, аналогічним елементом буде і зв'язок між двома УФО-елементами. Фактично, вона являє собою поєднання двох зв'язків - вихідного зв'язку УФО-елемента ім'я1 і вхідного зв'язку УФО- елемента ім'я2. Але, оскільки згідно УФО-підходу в термінах теорії патернів, ці зв'язки повинні мати однакові показники (тобто мати однаковий тип з класифікації зв'язків), то подібне з'єднання можна позначити єдиним показником, аналогом якого і є предикат RDF-триплетів.

Ще одним важливим аналізованих аспектом взаємозв'язку між формальним апаратом УФО-підходу і мовою RDF як засобом опису онтологій є аналіз наявності у вузла УФО-елемента визначених зв'язків. У термінах теорії патернів в даному випадку слід враховувати, що зв'язок у утворює має деякий показник зв'язку. Аналогічно зв'язок між УФО-елементами, яка може розглядатися 46 як вхідний зв'язок вузла одного елемента, так і як виходить вузла іншого елемента, має тип з класифікації зв'язків. Такий показник (тип зв'язку) можна також розглядати як ім'я якоїсь абстрактної сутності. Отже, факт наявності зв'язку у вузла УФО-елемента теж можна представити у вигляді RDF-триплетів (рисунок 2.7).

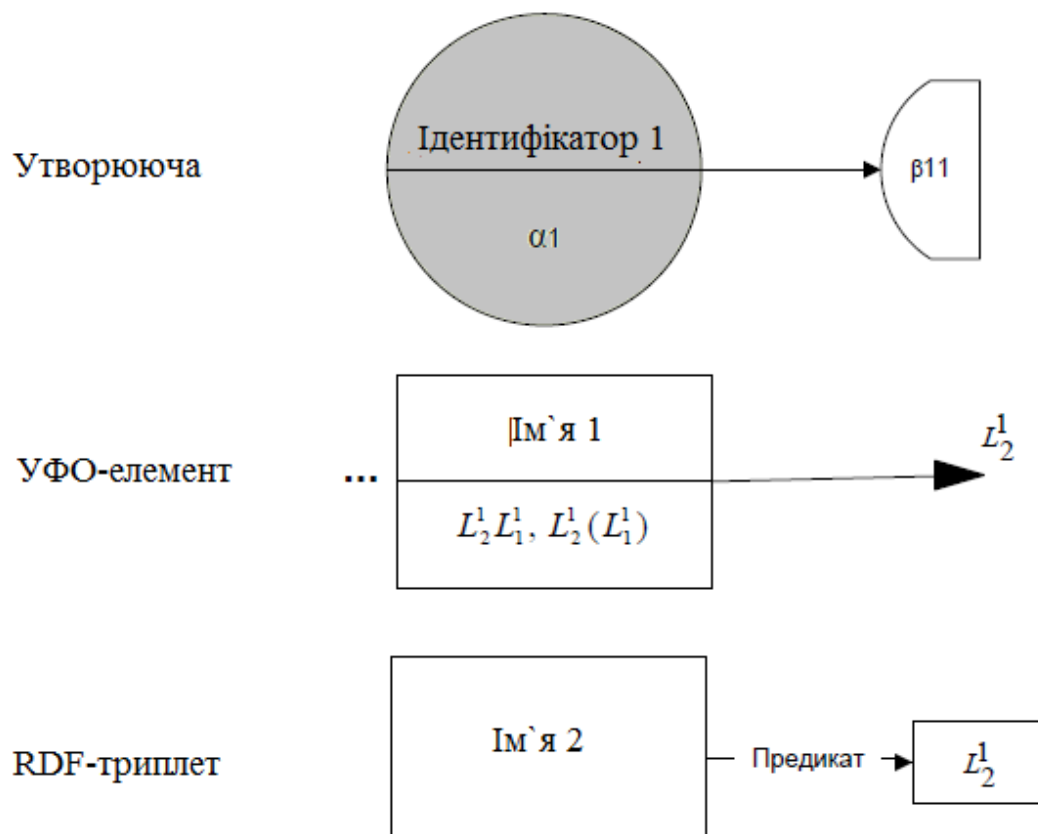


Рисунок 2.7 - Відповідність утворює, фрагмента УФО-елемента і RDF-триплета

Як предиката тут потрібен певний абстрактне ім'я. Причому для подібних триплетів, що формалізують факт наявності у вузла зв'язку, можуть бути прийняті два предиката у вигляді дієслів, що позначають характер конкретної зв'язку - вихідні та вхідні, наприклад «МожетПередавать» і «МожетПринимать». При перекладі УФО-моделі в RDF-факти слід враховувати наявність у УФО-елемента конкретного об'єкта. Переклад самого факту «УФО-елемент є представлений об'єктом n» в RDF недоцільний, оскільки RDF-триплети описують факти щодо реальних або абстрактних сутностей предметної області. При записи УФО-моделей на RDF як ім'я суті використовується ім'я (ідентифікатор) об'єкта УФО-елемента, а не нотація самого УФО-елемента.

Однак об'єкт УФО-елемента, в свою чергу, має ряд властивостей α . Дані факти придатні для запису на RDF. При цьому суб'єкт буде представлений ім'ям $n \in N$ з безлічі імен об'єктів; предикат відповідно може бути позначений виразом «має властивість»; об'єктом буде конкретне властивість з безлічі α : Невід'ємною частиною УФО-елемента є функція. За аналогією з описаними раніше випадками, факт наявності у УФО-елемента функції також може бути перетворений в RDF-триплет. В якості суб'єкта в цьому випадку може виступати ім'я об'єкта УФО-елемента як відображення якоїсь сутності предметної області, в якості об'єкта - ім'я (нотація) самої функції. Предикат може бути позначений виразом «виконує функцію». Формально процес перетворення факту наявності функції у УФО- елемента в RDF-триплет представляється функцією виду:

$$F(e) = \langle n : \text{Executes} : S \rangle. \quad (2.1)$$

Систематизуючи вищесказане, можна зробити висновок про те, що принцип представлення даних в УФО-підході (засобах його формалізації) і мови RDF багато в чому аналогічні. Це дає підстави стверджувати, що УФО-модель предметної області може бути представлена за допомогою мови RDF.

2.3. Дослідження і адаптація засобів алгебраїчного опису УФО-моделей для їх використання при перетворенні УФО-моделей в конструкції мов формального уявлення онтологій

Щоб уявити витягнуті з УФО-моделі предметної області факти за допомогою формальних мов, зокрема, розглянутих раніше RDF або OWL, потрібно мати початковий формалізоване уявлення УФО-моделі, на основі якого і буде здійснюватися вилучення та запис фактів про предметну область. У літературі зустрічаються підходи до формалізації УФО-підходу, використовують поєднання декількох алгебраїчних засобів [81], таких, як 48 теорія патернів Гренандера, обчислення процесів Мілнера, обчислення об'єктів. Кожен з таких математичних апаратів успішно використовується для формалізації окремого аспекту УФО-підходу, причому вибір

здійснюється в залежності від поставлених завдань. У випадку з завданням вилучення і формальної записи фактів про предметну область з УФО- моделі потрібно мати єдиний математичний апарат, що дозволяє формально описати все видобувні факти. Тому підходи, засновані на використанні різних засобів формалізації для окремих аспектів УФО-підходу, в даному випадку неефективні.

Отже, можливі два варіанти розробки способу формалізації УФО- моделі для вилучення фактів про предметну область: 1. Вибір і адаптація для розв'язуваної задачі одного з існуючих математичних апаратів. 2. Розробка нового способу формалізації на основі поєднання засобів, в тій чи іншій мірі застосовуються для формалізації УФО-підходу. Переваги та недоліки зазначених підходів можна уявити в вигляді таблиці 2.1.

Таблиця 2.1 - Переваги і недоліки різних підходів до вибору способу формалізації

Підхід	Переваги	Недоліки
Використання існуючого способу формалізації, адаптованого для вирішення поставленої задачі	• Відсутність необхідності розробки «з нуля» і обґрунтування термінології, позначень, основних положень і теорем. • Уніфіковане (Збірне, одноманітне) опис всіх розглянутих аспектів.	• Наявність обмежень і особливих умов застосування в використовуваному способі формалізації. • Односторонній погляд на формалізуються теорію.
Розробка нового способу формалізації	• Облік всіх необхідних аспектів формалізується області.	• Необхідність ретельної розробки термінології, позначень, основних положень і теорем. • Складність поєднання різних засобів і точок зору в одному

		розробляється методі формалізації.
--	--	------------------------------------

Також слід зазначити, що в разі вибору та адаптації будь-якого існуючого методу для вирішення задачі формалізації УФО-моделей в 3 метою подальшого вилучення фактів процес зіставлення елементів математичного апарату і понять формалізуемой області буде йти в напрямку від обраного методу до формалізуемой області. іншими словами, з усього комплексу засобів, що надаються обраних способом формалізації, будуть відбиратися тільки ті, які можуть бути застосовані для формалізації УФО-моделей з урахуванням подальшого вилучення фактів. У тому випадку, коли розробляється новий спосіб формалізації, наприклад, шляхом поєднання і доповнення засобів, представлених в декількох відомих теоріях, даний процес буде мати зворотну спрямованість. Тобто для конкретних понять і аксіом формалізуемой області будуть розроблятися відповідні кошти формалізації.

Як видно з таблиці, обидва варіанти мають як достоїнствами, так і недоліками. З точки зору необхідності отримання максимально повного і всеосяжного формального опису більш краща розробка нового спеціалізованого способу формалізації УФО-моделей для потреб вилучення фактів про предметну область. Однак на початковому етапі для дослідження можливостей вилучення з УФО-моделі фактів про предметну область і їх формальної записи, наприклад, з допомогою RDF, досить ефективним може бути вибір, адаптація і застосування один з існуючих методів формалізації. Надалі, на основі результатів аналізу недоліків такого підходу, можна розробити новий спосіб формалізації, що включає як найбільш ефективні засоби спочатку обраного відомого способу, так і доповнення у вигляді нових засобів. Розглянемо подальші кроки по вилученню фактів про предметну області з УФО-моделей з урахуванням того, що для формалізації

буде використовуватися один з уже відомих підходів, адаптований для вирішення поставленого завдання.

Кожна з описаних раніше груп видобутих фактів являє собою певну точку зору на УФО-модель або її частина. Відповідно, спираючись на проведені раніше дослідження можна припустити, який спосіб формального уявлення УФО-моделей дозволить максимально повно врахувати і легко виділити з формальної записи ту чи іншу групу фактів (таблиця 2.2).

Група фактів	Можливі засоби формального опису
На рівні УФО-елемента	теорія патернів
I (inv). На рівні УФО-елемента	теорія патернів
На рівні Вузла	• Теорія патернів; • Обчислення об'єктів
На рівні Функції	Обчислення процесів (CCS)
На рівні Об'єкту	• Теорія патернів; • Обчислення об'єктів
На рівні зв'язків між двома УФО-елементами	теорія патернів
V (inv). На рівні зв'язків між двома УФО-елементами	теорія патернів
На рівні декомпозиції УФО-елемента	• Теорія патернів; • обчислення процесів (CCS)
На рівні переходу до контекстного рівню УФО-моделі	• Теорія патернів; • обчислення процесів (CCS)

Таблиця 2.2 - Вибір способу формалізації УФО-моделей для відображення груп фактів про предметну область

Очевидно, що для відображення фактів всіх груп, за винятком групи III запропонованої класифікації (на рівні Функції), може використовуватися з тими чи іншими припущеннями теорія патернів Гренандера [77]. Для того щоб врахувати і цю групу, можна скористатися способом формалізації, запропонованим в роботі [84]. Погляньмо на цей спосіб формалізації в контексті завдання вилучення фактів про предметну область з УФО-моделей. Маторин С.І. і Зимовець О.А. [84] представляють систему (УФО-елемент) у вигляді кортежу:

$$e = \langle U, F, O \rangle, \quad (2.2)$$

де U - Вузол, F - Функція, O - Об'єкт відповідно. таке уявлення дозволяє в зручній формі зберігати і витягувати з формального уявлення УФО-моделі факти з групи I. З урахуванням положень, розроблених в роботах [78, 80, 84], кожен елемент наведеного раніше кортежу може бути деталізований. В Зокрема, Вузол можна представити у вигляді виразу:

$$U = (L?, L!), \quad (2.3)$$

де $L?$ - безліч вхідних зв'язків; $L!$ - безліч вихідних зв'язків. Подібна форма запису дозволяє враховувати факти 3 і 4 з групи II. Дійсно, в даному випадку мається на увазі, що $L?$ і $L!$ - це безлічі (сукупності) вхідних і вихідних зв'язків відповідно. Іншими словами, якщо у вузла є $k?$ вхідних і $k!$ вихідних зв'язків, то:

$$\forall m = 1..k?: L_m? \subset L?, \quad (2,4)$$

$$\forall m = 1..k!: L_m! \subset L!,$$

Значить, що міститься в формальному поданні УФО-моделі інформацію про елементи множин $L?$ і $L!$ можна в подальшому перетворити в факти виду «Вузол має вхідну зв'язок $L_m?$ » і «Вузол має вхідні зв'язок $L_m!$ ».

Для обліку фактів груп I і I (inv) також потрібне використання ідентифікатора (імені) вузла. У більшості приватних випадків імена Вузла і Об'єкта збігаються, проте в загальному випадку будемо враховувати ім'я Вузла окремо, позначаючи його символом u і включає в формальне подання вузла:

$$U = (u, L?, L!), \quad (2,5)$$

Другий елемент кортежу, що представляє УФО-елемент, - Функцію – автори роботи також представляють в деталізованому вигляді, використовуючи поняття, аналогічні термінам CCS. Однак в рамках поставленого завдання особливий інтерес представляє можливість вилучення з формальної записи фактів групи III, куди входять факти виду «Функція перетворює вхід $Lm?$ » і «Функція видає вихід $Lm!$ ». Відповідно, для цього не потрібно деталізоване формальне подання Функції як складової УФО-елемента; досить її імені (змісту), яку будемо представляти в формальній записи літерою F .

Для опису об'єкта як складової УФО-елемента пропонується використовувати поняття теорії патернів. При цьому вводиться наступне уявлення Об'єкта:

$$O = (n, \alpha, \beta?, \beta!), \quad (2.6)$$

де n - ім'я об'єкта; α - безліч ознак Об'єкту; $\beta?$ – безліч показників безлічі вхідних зв'язків $L?$; $\beta!$ - безліч показників безлічі вихідних зв'язків $L!$.

В даному контексті приймемо в якості показників зв'язків їх клас – речові зв'язку, енергетичні зв'язки, зв'язки з управління і зв'язку за даними [91]. Відповідно, для кожної зв'язку з множин $L?$ і $L!$ в формальній записи УФО-моделі присутні записи «Вузол має зв'язок класу $\beta?$ і »і« Вузол має зв'язок класу $\beta!$ j »відповідно, що відповідає фактам 1 і 2 з групи II. З урахуванням наведених положень загальну запис УФО-елемента можна представити таким чином:

$$O = \langle (u, L?, L!), F, (n, \alpha, \beta?, \beta!) \rangle, \quad (2.7)$$

При цьому подібного формального уявлення УФО-елемента досить для вилучення фактів груп I, I (inv), II, III і IV запропонованої раніше класифікації. Відповідно, необхідно розглянути формальне обґрунтування вилучення фактів груп V і V (inv), що стосуються наявності в УФО-моделі зв'язків між конкретними УФО-елементами. наявність зв'язків між УФО-елементами являє собою досить великий джерело інформації про предметну область, що тягне за собою необхідність детального вивчення формальної сторони процесу вилучення відповідних фактів. Для наочності уявлення факту наявності зв'язку між двома УФО-елементами моделі пропонується використовувати складені по певним правилам матриці. Представляються допустимими кілька варіантів запису таких матриць. Наприклад, в загальному випадку матрицю зв'язків для і-го УФО-елемента з k? вхідних зв'язків і k! вихідних зв'язків можна представити таким чином:

$$\begin{array}{c|c}
 L_1^i? & 0 \\
 L_2^i? & L_m^j! \\
 \dots & \dots \\
 L_{k?}^i? & 0 \\
 L_1^i! & 0 \\
 L_2^i! & 0 \\
 \dots & \dots \\
 L_{k!}^i! & L_m^j?
 \end{array}$$

(2.8)

де k? - кількість вхідних зв'язків розглянутого УФО-елемента; k! - кількість вихідних зв'язків розглянутого УФО-елемента; L! J m і L? J m - це деякі вхідні і вихідна зв'язку відповідно j-го УФО- елемента, з яким пов'язаний розглянутий і-й УФО-елемент. У зазначеній матриці в рядках перераховуються всі вхідні і вихідні зв'язку УФО-елемента, а навпроти них проставляється позначення рівнозначною зв'язку пов'язаного УФО-елемента.

Тобто зв'язку, які мають значення, відмінне від нуля, у другому стовпці матриці, є зв'язком (Потоком) між двома УФО-елементами. Відповідно, на підставі подібних матриць з формальної записи УФО-моделі можна витягти факти груп V і $V(inv)$, що характеризують наявність зв'язків між УФО-елементами. Більш універсальним способом є створення матриці розміром $N \times M$ елементів, що відображає всі існуючі на даному рівні УФО-моделі взаємозв'язку між елементами. При цьому:

$$N = \sum_{i=1}^n k^i ?, \quad (2.9)$$

де n - число УФО-елементів на даному рівні моделі (рівні декомпозиції вихідного елемента, що відображає систему в цілому); $k^i ?$ - число входять зв'язків кожного з таких елементів аналогічно:

$$M = \sum_{i=1}^n k^i !, \quad (2.10)$$

де n - число УФО-елементів на даному рівні моделі (рівні декомпозиції вихідного елемента, що відображає систему в цілому); $k^i !$ - число вихідних зв'язків кожного з таких елементів.

Відповідно, в якості рядків такої матриці виступають входять зв'язку всіх УФО-елементів розглянутого рівня моделі, а в якості стовпців - всі вихідні зв'язку. На перетині рядка і стовпця проставляється значення «1» в разі, коли вхідний зв'язок з'єднується (Є однойменної) з відповідною вихідної зв'язком, і «0» в інакше.

Розглянемо найпростіший приклад - якийсь рівень УФО-моделі, на якому розташовані три взаємопов'язаних УФО-елемента (рисунок 2.8).

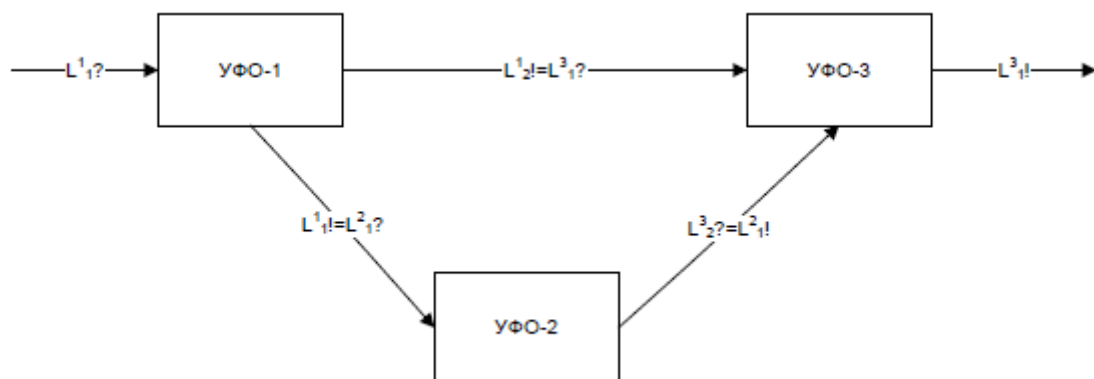


Рисунок 2.8 - Приклад фрагмента УФО-моделі

Для подібної моделі матриця зв'язків буде виглядати наступним чином:

	$L_1^1!$	$L_2^1!$	$L_1^2!$	$L_1^3!$
$L_1^1?$	0	0	0	0
$L_1^2?$	1	0	0	0
$L_1^3?$	0	1	0	0
$L_2^3!$	0	0	1	0

(2.11)

Згідно з положеннями УФО-підходу, УФО-елемент не може мати циклічних (замкнутих на себе) зв'язків. Відповідно, в матриці не має бути значень «1» на перетині рядків і стовпців, що означають відповідно вхідні та вихідні зв'язку одного і того ж УФО- елемента. У наведеному вище прикладі це елементи матриці $L^? 1$

$1 L^! 11, L^? 1 1 L^! 1 2, L^? 2 1 L^! 2 1, L^? 3 1 L^! 3 1$ і $L^? 3 2 L^! 3$

1. На основі даного положення може здійснюватися перевірка коректності УФО-моделі, а саме відсутність циклічних зв'язків у УФО-елементів. Для кожного рядка і стовпчика може бути розрахований контрольний розряд - сума значень всіх осередків рядка або стовпця відповідно. Значення, рівне 1, свідчить про те, що даний зв'язок є внутрішньої, тобто з'єднує між собою два УФО-елемента на даному рівні моделі. Значення, рівне 0, говорить про те, що даний зв'язок є інтерфейсної. Таким чином, формальне подання наявності зв'язків між УФО-елементами у вигляді єдиної матриці для кожного рівня декомпозиції дозволяє не тільки зберігати знання про ці факти в зручній для подальшої обробки формі, а й здійснювати додаткову перевірку коректності аналізованої УФО-моделі. Однак при використанні такого підходу представляється скрутним встановлення відомостей про кількість і характер зв'язків між конкретними двома УФО-

елементами в моделі. Тому в окремих випадках може використовуватися інший підхід для формального запису зв'язків між УФО-елементами - складання кількох матриць, характеризують зв'язку між двома окремо взятими УФО-елементами одного рівня. Розмірність кожної матриці може бути визначена як $N \times M$, причому

$$\begin{aligned} N &= k^i? + k^i!, \\ M &= k^j? + k^j!, \end{aligned} \quad (2.12)$$

де $k^i?$ і $k^j?$ - число входять зв'язків i -го і j -го УФО-елементів, $k^i!$ і $k^j!$ - число вихідних зв'язків i -го і j -го УФО-елементів відповідно. Тобто в Як рядків підставляються все (спочатку входять, потім вихідні) зв'язку i -го УФО-елемента, а в якості стовпців - все зв'язку j -го УФО- елемента в аналогічному порядку. Наприклад, для елементів УФО-1 і УФО-2, зображених на малюнку 2.8, подібна матриця буде виглядати наступним чином:

	$L_1^2?$	$L_1^2!$
$L_1^1?$	0	0
$L_1^1!$	1	0
$L_2^1!$	0	0

(2.13)

До переваг даного підходу можна віднести те, що з подібних матриць легко встановити кількість зв'язків між двома конкретними УФО-елементами. Недоліком, очевидно, є необхідність складання кількох матриць. Зокрема, в разі, якщо на одному рівні моделі розташовано n взаємопов'язаних УФО-елементів, знадобиться складання C матриць, де:

$$C = \frac{n!}{2 \cdot (n-2)!}. \quad (2.14)$$

Виходячи з припущення про те, що середня кількість елементів на одному рівні, при якому модель є зручною для сприйняття, так само 5, необхідна побудова 10 матриць. Якщо ж на одному рівні в моделі розташовані 6 елементів, число матриць зростає до 15. Тому в загальному випадку

пропонується користуватися першим, більш універсальним і компактним варіантом подання матриці зв'язків: здійснювати побудова єдиної матриці розміром $N \times M$ (вираження (2.9) і (2.10), в якій відображати всі наявні в моделі зв'язку між UFO-елементами. Для опису груп фактів VI і VII потрібно більш складний і детальний спосіб формалізації. Оскільки UFO-модель предметної області будується по низхідному принципу, тобто від контекстного рівня до потрібного ступеня декомпозиції, пропонується спочатку розглядати процес декомпозиції UFO-елементів. Питання формалізації декомпозиції при побудові UFO-моделей частково зачіпаються в роботі [90]. Однак авторами розглянуто тільки окремий випадок - інтерфейсна декомпозиція. З метою вилучення фактів про предметної області необхідно розглядати декомпозицію UFO- елемента в більш загальному сенсі, враховуючи можливі варіанти внутрішніх взаємозв'язків UFO-елементів на рівні декомпозиції. Якщо в розглянутій UFO-моделі той чи інший UFO-елемент має декомпозицію, значить, він може бути представлений сукупністю взаємопов'язаних UFO-елементів нижчого рівня. Для запису відповідного факту при вилученні з UFO-моделі пропонується ввести особливий предикат (тип відносини) - «є частиною». все UFO- елементи нижчого рівня фактично «є частиною» декомпозируемого UFO-елемента. Схематично це можна представити наступним чином (рисунок 2.9).

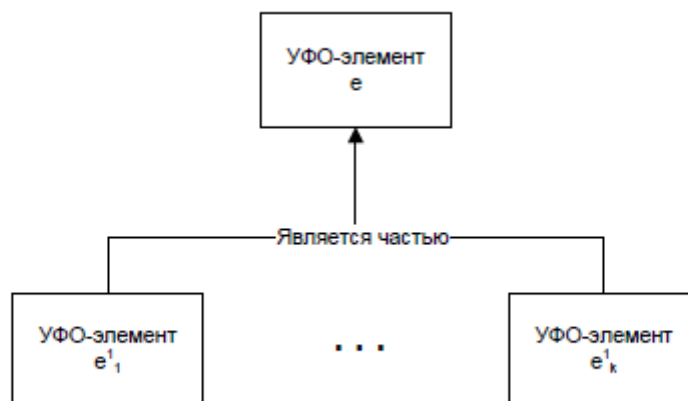


Рисунок 2.9 - Схема уявлення факту наявності декомпозиції UFO-елемента

З метою виявлення фактів про «дочірніх» УФО-елементах розглянемо докладніше «Функцію» декомпозируемого УФО-елемента, оскільки декомпозиція є деяким поданням його внутрішньої функціональної структури у вигляді сукупності взаємопов'язаних підсистем (УФО-елементів) зі своїми функціями. Для цього скористаємося формальним поданням функції УФО-елемента, наведеними в роботі [90]:

$$N = (S, S^0, L_\tau), \quad (2.15)$$

де S - безліч підпроцесів процесу, відповідного «Функції», які реалізуються елементами першого рівня декомпозиції; S^0 - безліч інтерфейсних підпроцесів, L_τ - безліч зв'язків в S , здійснюють обіг об'єктів глибинного ярусу пов'язаних підпроцесів. Оскільки під час запису фактів ми оперуємо термінами предметної області, введемо в вираз «Функції» УФО-елемента ще одну складову - безпосередньо ім'я (позначення) функції, яке будемо також позначати через F . Безліч S в загальному вигляді можна виразити як:

$$S = \bigcup_{n=1}^k S_n^{-1}, \quad (2.16)$$

де k - число підпроцесів, на які декомпозирується функція (процес) УФО-елемента, а $1 \leq n \leq k$ - власне такий підпроцес.

У свою чергу, S^0 можна представити виразом виду:

$$S^0 = S^? \cup S^! = \left(\bigcup_{m=1}^h S_m^{-1} ? \right) \cup \left(\bigcup_{t=1}^l S_t^{-1} ! \right), \quad (2.17)$$

де h - число вхідних інтерфейсних підпроцесів, l - число вихідних інтерфейсних підпроцесів, а $S^? 1 \leq m \leq h$ і $S^! 1 \leq t \leq l$ - відповідно самі такі підпроцеси.

В контексті записи фактів про предметну область в якості підпроцесів, що позначаються символами S , будемо використовувати «Функції» відповідних УФО-елементів нижнього рівня (рівня декомпозиції). В формулою також для наочності замінимо символи S , що позначають

підпроцеси, на F - імена їх функцій. Для формального представлення безлічі $\square$ L внутрішніх зв'язків необхідно розглянути матрицю зв'язків першого рівня декомпозиції, оскільки $L \sqsubseteq L_1 \sqsubseteq \dots \sqsubseteq L_n$. внутрішніми зв'язками можна вважати безліч зв'язків:

$$L_{\tau} = L^{-1} \setminus (L^{-1} ? \cup L^{-1} !), \quad (2.18)$$

де $L ? L ! \sqsubseteq 1 \sqsubseteq 1 \sqsubseteq \dots \sqsubseteq 1$ - безліч інтерфейсних зв'язків.

Таким чином, «функцію» декомпозируемого УФО-елемента в загальному вигляді можна представити за допомогою формули:

$$F = (F, \bigcup_{n=1}^k F_n^{-1}, \bigcup_{m=1}^h (F_m^{-1} ?) \cup (\bigcup_{t=1}^l F_t^{-1} !), L^{-1} \setminus (L^{-1} ? \cup L^{-1} !)). \quad (2.19)$$

Права частина даного рівності і являє собою характеристику суті декомпозиції, тобто відносини «ціле-частина». Відмітимо, що остання складова даного вираження буде врахована в момент характеристики зв'язків між УФО-елементами на рівні декомпозиції. Тому при записі того факту, що УФО-елемент може бути декомпонований, немає необхідності дублювати такі факти. Таким чином, відомості про наявність в УФО-моделі декомпозиції УФО-елемента на кілька інших взаємопов'язаних УФО-елементів можуть бути представлені у вигляді безлічі фактів, що характеризують відношення «Ціле-частина» між декомпозируемым УФО-елементом і кожним з УФО- елементів на рівні декомпозиції. Вираз (2.19) дозволяє формалізувати уявлення фактів груп VI і VII класифікації, характеризують декомпозицію і агрегацію УФО-елементів моделі.

2.4. Розробка методу опису УФО-моделі на формалізованому мовою записи онтології

Оскільки в якості засобу формалізації записи фактів про класифікації зв'язків використовуються нотації мови RDF, всі факти будуть представлені у вигляді триплетів. При формуванні таких RDF-триплетів часто в якості

предикатів виступають типові дії і позначення відносин, які багаторазово зустрічаються в різних предметних областях.

З метою уніфікації записи даних, представлених за допомогою RDF, а також спрощення їх сприйняття, подібні предикати включають до складу так званих словників RDF. Прикладами найбільш розвинених подібних словників, що містять найбільш загальні, міждисциплінарні терміни (Предикати), можуть вважатися RDFS (RDF schema) [57], Dublin Core [11], schema.org [30]. Для окремих сфер діяльності та професійних областей також розробляються спеціалізовані словники метаданих та специфікації. Коротка характеристика основних RDF-словників приведена в таблиці 2.3. Перераховані проекти об'єднує ідея формування загальноприйнятих конструкцій для використання в метаданих.

Таблиця 2.3 - Найбільш відомі і часто використовувані RDF-словники

Коротке Найменування	Повний найменування	Спеціалізація, предметна область	Адреси в мережі Інтернет
RDFS	RDF schema	Базовий набір класів і властивостей для моделі представлення знань RDF	http://www.w3.org/TR/rdfschema/
Dublin Core	Dublin Core Metadata Element Set	Словник основних понять	http://dublincore.org/documents/dces/

		англійської мови	
schema.org	schema.org	Стандарт семантичної розмітки і структурування даних в мережі Інтернет	https://schema.org/
FOAF	Friend of a Friend	Соціальний веб	http://xmlns.com/foaf/spec/
SIOC	Semantically-Interlinked Online Communities	Соціальний веб	http://www.sioc-project.org/ http://rdfs.org/sioc/spec/#secsioc-rdf
OMF	Open Source Metadata Framework	Програмне забезпечення з відкритим кодом	http://www.ibiblio.org/osrt/omf/index.html
The Music Ontology	-	Музика	http://musicontology.com/specification/

Найбільш затребуваним застосуванням подібних RDF-словників є область машинного пошуку і побудови web-сторінок з метаданими, що забезпечують високу якість пошуку інформації в мережі Інтернет.

Як правило, предикати, що відповідають основним можливим видам відносин між об'єктом і суб'єктом, необхідні для запису витягнутих з УФО-моделі фактів, вже включені в існуючі базові словники RDF. Однак, оскільки факти, що відображають відносини між складовими УФО-елементів та самими УФО-елементами, досить специфічні, багато з відповідних предикатів відсутні в найбільш розвинених і загальноприйнятих словниках метаданих та специфікаціях. Тому виникає необхідність створення власного спеціалізованого словника, до якого увійдуть конструкції (предикати), призначені для опису відносин в рамках предметної області з точки зору УФО-аналізу. Як найменувань таких предикатів будемо використовувати вираження, записані латинськими літерами і відображають зміст відносини між суб'єктом і об'єктом. Використання символів латинського алфавіту обумовлено тим, що більшість спеціалізованих для роботи з мовами на зразок RDF програмних коштів (редакторів коду, трансляторів між нотаціями, перетворювачів коду в графічні файли) не підтримують кирилицю.

Розглянемо представлення фактів з розробленою класифікацією з допомогою конструкцій мови RDF. Для зручності будемо використовувати одну з найбільш простих для сприйняття нотацій RDF - N-Triples. Групи I і I (inv) класифікації містять перелік фактів, характеризують УФО-елемент, взаємозв'язок і взаємодія окремих його «складових» - вузла, функції і об'єкта. Для запису даних фактів буде потрібно шість спеціальних предикатів, яких немає в поширених RDF-словниках. Додамо їх у власний словник. Перелік предикатів (відносин), необхідних для запису фактів груп I і I (inv), наведено в таблиці 2.4.

Таблиця 2.4 - RDF-предикати для фактів груп I і I (inv) класифікації

Група фактів	Найменування відносини	Предикат	Повне уявлення предиката в нотації N-Triples мови RDF
I	балансується	isBalancedBy	<http://www.ufo-toolkit.ru/#isBalancedBy>
	реалізується за допомогою	isRealizedBy	<http://www.ufo-toolkit.ru/#isRealizedBy>
	зайнятий	isRepresentedBy	<http://www.ufotoolkit.ru/#isRepresentedBy>
I(inv)	балансується	balances	<http://www.ufo-toolkit.ru/#balances>
	реалізується	realizes	<http://www.ufo-toolkit.ru/#realizes>
	займає	represents	<http://www.ufo-toolkit.ru/#represents>

У загальному випадку факти груп I і I (inv) з використанням зазначених предикатів в нотації N-Triples будуть виглядати наступним чином:

Група I:

1. <u isBalancedBy F>;
2. <F isRealizedBy n>;
3. <u isRepresentedBy n>.

Група I(inv):

1. <F balances u>;

2. $\langle n \text{ realizes } F \rangle$;

3. $\langle n \text{ represents } u \rangle$,

де u - нотація вузла, F - ім'я функції, n - ім'я об'єкта.

Як зазначалося раніше, факти груп I і I (inv) відображають одні і ті ж відносини між двома сутностями, але розглядають їх в різних напрямках в залежності від того, яка сутність виступає об'єктом, а яка - суб'єктом. Отже, предикати відповідних триплетів позначають протилежно спрямовані відносини (дії, потоки), що саме по собі є фактом і має бути відображено в формальній записи. Для позначення такого виду відносин в RDF може використовуватися спеціальний предикат `owl: inverseOf`. Таким чином, до фактами груп I і I (inv) додадуться ще три службових триплета, характеризують відносини між використовуваними предикатами. Такі триплети мають бути присутні в кожній RDF-записи фактів, витягнутих з УФО-моделі. Їх уявлення буде незмінним і незалежним від аналізованої предметної області. Приклад їх подання в нотації NTriples наведено нижче:

$\langle \text{http://www.ufo-toolkit.ru/\#balances} \rangle$

$\langle \text{http://www.w3.org/2002/07/owl\#inverseOf} \rangle$

$\langle \text{http://www.ufo-toolkit.ru/\#isBalancedBy} \rangle$.

$\langle \text{http://www.ufo-toolkit.ru/\#realizes} \rangle$

$\langle \text{http://www.w3.org/2002/07/owl\#inverseOf} \rangle$

$\langle \text{http://www.ufo-toolkit.ru/\#isRealizedBy} \rangle$.

$\langle \text{http://www.ufo-toolkit.ru/\#represents} \rangle$

$\langle \text{http://www.w3.org/2002/07/owl\#inverseOf} \rangle$

$\langle \text{http://www.ufo-toolkit.ru/\#isRepresentedBy} \rangle$.

Перші два факти групи II класифікації можна логічно узагальнити в одне твердження про здатність Вузла мати порт, який може бути вхідними або вихідними і ставитися до певного класу.

Відповідно, для запису фактів необхідні такі предикати, як «мати входить порт класу ... »(`hasInputPortClass`) і « мати вихідний порт класу ... »(`hasOutputPortClass`), які є окремими випадками відносин «мати входить

порт» (hasInputPort) і «мати вихідний порт» (HasOutputPort). Ці поняття, в свою чергу, можуть розглядатися як окремий випадок загального поняття «мати порт» (hasPort). Подання взаємозв'язків перерахованих понять у вигляді графа наведено на рисунку 2.10.

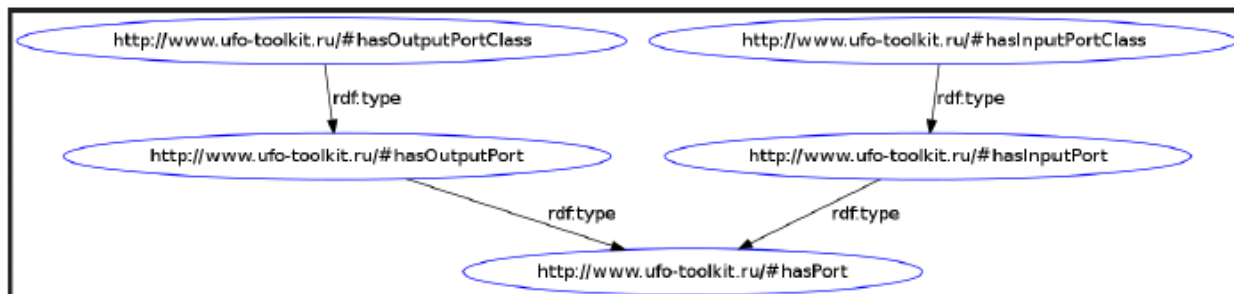


Рисунок 2.10 - Взаємозв'язок предикатів для запису фактів на рівні Вузла в вигляді графа

Аналогічним чином можна уявити взаємозв'язку предикатів, необхідних для запису фактів, що відображають наявність у Вузла зв'язків: «мати вхідний зв'язок »(hasInputRelation), «мати вихідний зв'язок » 64 (HasOutputRelation), «мати зв'язок» (hasRelation). співвідношення даних понять проілюстровано на малюнку 2.11.

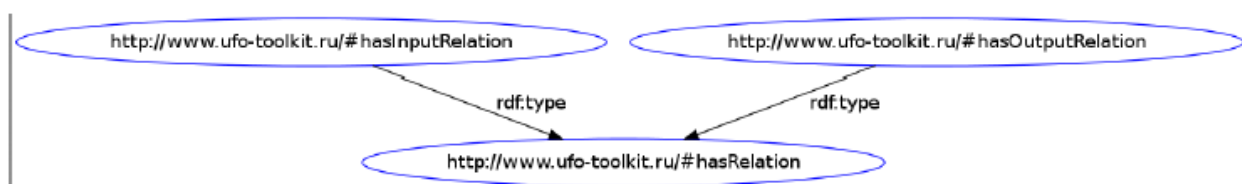


Рисунок 2.11 - Взаємозв'язок предикатів, що характеризують наявність зв'язків у УФО-елемента

Таким чином, для представлення перших чотирьох фактів групи II класифікації необхідно використання наступних RDF-предикатів (Таблиця 2.5).

Таблиця 2.5 - RDF-предикати для фактів групи II класифікації

Найменування відносини	Предикат	Повне уявлення предиката в нотації N-Triples мови RDF
Мати порт	hasPort	<i><http://www.ufo-toolkit.ru/#hasPort></i>
Мати вхідний порт	hasInputPort	<i><http://www.ufo-toolkit.ru/#hasInputPort></i>
Мати вихідний порт	hasOutputPort	<i><http://www.ufo-toolkit.ru/#hasOutputPort></i>
Мати вихідний порт класу ...	hasInputPortClass	<i><http://www.ufotoolkit.ru/#hasInputPortClass></i>
Мати вихідний порт класу ...	hasOutputPortClass	<i><http://www.ufotoolkit.ru/#hasOutputPortClass></i>
Мати зв'язок	hasRelation	<i><http://www.ufo-toolkit.ru/#hasRelation></i>
Мати вихідний зв'язок	hasInputRelation	<i><http://www.ufotoolkit.ru/#hasInputRelation></i>
Мати вхідний зв'язок	hasOutputRelation	<i><http://www.ufotoolkit.ru/#hasOutputRelation></i>

З урахуванням введених предикатів розглядаються факти групи II класифікації в загальному вигляді можна представити таким чином:

1. $\square m \square 1..t ? : <u \text{ hasInputPortClass } \beta m >$;
2. $\square m \square 1..t ! : <u \text{ hasOutputPortClass } \beta m >$;
3. $\square m \square 1..k ? : <u \text{ hasInputRelation } Lm ? >$;
4. $\square m \square 1..k ! : <u \text{ hasOutputRelation } Lm ! >$.

Факти, що стосуються приналежності кожної конкретної зв'язку УФО-елемента того чи іншого класу, будуть враховані в поданні класифікації зв'язку, використовуваної в аналізованій УФО-моделі. Розглянемо групу III класифікації фактів, які з УФО-моделі. Це - відомості, що стосуються Функцій УФО-елементів. Для представлення даних відомостей у вигляді RDF-триплетів потрібно два спеціальних предиката, відсутніх в поширених RDF-словниках (Таблиця 2.6).

Таблиця 2.6 - RDF-предикати для фактів групи III класифікації

Найменування відносини	Предикат	Повне уявлення предиката в нотації N-Triples мови RDF
Перетворює вхід	translateInput	http://www.ufo-toolkit.ru/#translateInput
Видає вихід	giveOutput	http://www.ufo-toolkit.ru/#giveOutput

З використанням запропонованих предикатів загальний вигляд запису фактів групи II виглядає наступним чином:

1. $\square m \square 1..k ? : <F \text{ translateInput } Lm ? >$
2. $\square m \square 1..k ! : <F \text{ giveOutput } Lm ! >$

Крім розглянутих раніше, в класифікації виділяється група фактів IV, що характеризує наявність у об'єктів конкретних ознак. Предикат, необхідний для запису таких фактів на мові RDF, представлений в таблиці 2.7.

Таблиця 2.7 - RDF-предикат для фактів групи IV класифікації

Найменування відносини	Предикат	Повне уявлення предиката в нотації N-Triples мови RDF
Має ознаку	hasAttribute	<i><http://www.ufo-toolkit.ru/#hasAttribute></i>

У загальному вигляді факти групи IV можна представити таким чином:

$\square m \square 1..q : \langle n \text{ hasAttribute } am \rangle$, де q - кількість ознак (атрибутів) об'єкта, n - ім'я об'єкта.

Приклад подання факту групи IV за допомогою нотації N-Triples мови RDF наведено нижче:

<http://www.ufo-toolkit.ru/#o.Відділ продажів> <http://www.ufotoolkit.ru/#hasAttribute> <http://www.ufo-toolkit.ru/#Description>

Одними з найважливіших груп фактів є групи V і V (inv) :, характеризують взаємозв'язку UFO-елементів одного рівня моделі. Відомості про наявність таких зв'язків в формальному поданні UFO-моделі зберігаються в спеціальних матрицях зв'язків. Однак крім матриці зв'язків, представленої в тому чи іншому вигляді, при добуванні фактів про предметну області з UFO-моделі необхідно враховувати і приналежність кожної з таких зв'язків до певного класу. Побудова UFO-моделі завжди передбачає використання існуючої або створення нової, спеціалізованої для даної предметної області класифікації всіх можливих зв'язків між UFO-елементами, заснованої на базовій

класифікації зв'язків [91]. Однак в конкретній УФО-моделі можуть використовуватися не всі перераховані в класифікації зв'язку, а лише частину з них. Тому пропонується витягувати і зберігати набір фактів (триплетів), характеризують використовувану при побудові УФО-моделі класифікацію зв'язків. Подібний набір фактів також може бути представлений за допомогою RDF. Зокрема, в класифікації зв'язків використовується тільки один вид відносин - родовидові. відповідно, факти, що стосуються безпосередньо класифікації зв'язків (їх родовидових відносин), можуть бути представлені у вигляді триплетів з одним і тим же предикатом. Зокрема, для позначення відношення «є різновидом» («загальне-приватне») в класифікації зв'язків між УФО- елементами можна використовувати властивість `rdfs: subClassOf` зі словника RDFS. Воно позначає відношення «є підкласом», що в повній мірі відповідає відносинам між розташованими на різних рівнях ієрархії класами зв'язків. Приклад триплета, що характеризує приналежність підкласу зв'язків «батьківського» класу, показаний нижче:

```
<http://www.ufo-toolkit.ru/#M> <http://www.w3.org/2000/01/rdfschema#
subClassOf> <http://www.ufo-toolkit.ru/#L> .
```

Формуючи подібним чином триплети для кожного класу, згаданого в класифікації зв'язків, отримаємо загальне формальне подання такої класифікації на мові RDF. Так, наприклад, базову класифікацію зв'язків, яка використовується в УФО-підході, з урахуванням тільки родовидових відносин між класами можна уявити за допомогою нотації N-triples моделі RDF наступним чином:

```
<http://www.ufo-toolkit.ru/#M> <http://www.w3.org/2000/01/rdfschema#
subClassOf> <http://www.ufo-toolkit.ru/#L> .
```

```
<http://www.ufo-toolkit.ru/#I> <http://www.w3.org/2000/01/rdf-
schema#subClassOf>
```

```
<http://www.ufo-toolkit.ru/#L> .
```

```
<http://www.ufo-toolkit.ru/#S> <http://www.w3.org/2000/01/rdf-
schema#subClassOf>
```

<http://www.ufo-toolkit.ru/#M> .

<http://www.ufo-toolkit.ru/#E>
schema#subClassOf>

<http://www.ufo-toolkit.ru/#M> .

<http://www.ufo-toolkit.ru/#C>
schema#subClassOf>

<http://www.ufo-toolkit.ru/#I> .

<http://www.ufo-toolkit.ru/#D>
schema#subClassOf>

<http://www.ufo-toolkit.ru/#I> .

<http://www.w3.org/2000/01/rdf-

<http://www.w3.org/2000/01/rdf-

<http://www.w3.org/2000/01/rdf-

Сучасні засоби обробки формального уявлення фактів в моделі RDF дозволяють також отримувати графічні зображення (графи) концептів, які беруть участь в RDF-триплетах в формальній записи. В Зокрема, наведений вище код можна представити у вигляді графа, показаного на рисунку 2.12.

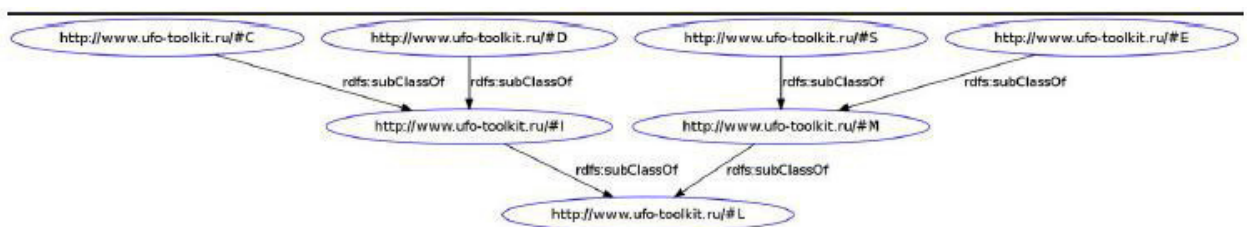


Рисунок 2.12 - Граф, який представляє базову класифікацію зв'язків в UFO-підході

Проте класифікація зв'язків не обмежується родовидові відносинами між класами. Класифікація зв'язків, як правило, містить певний опис кожного класу у вигляді безлічі атрибутів. Для отримання їх переліку звернемося до формального поданням UFO- моделі, створеної в спеціалізованому інструментальному засобі. Для здійснення моделювання предметної області з використанням UFO- підходу в більшості випадків застосовується інструментарій «UFO-Toolkit» [87]. UFO-модель, створена в даному

програмному засобі, зберігається в вигляді XML-файла. Оскільки класифікація зв'язків є обов'язковою частиною будь-якої УФО-моделі, в таких XML-файлах присутня фрагмент, який містить відомості про конкретну варіації базової класифікації зв'язків, використовуваної в даній моделі. Фрагмент XML-коду УФО-моделі, створеної в програмному засобі «UFO-Toolkit», що характеризує один з класів зв'язків, представлений нижче.

```
<PortHierarchy ID="6">
```

```
<Item Name="Связь" Notation="L" ID="7" PortType="Link">
```

```
</PortHierarchy>
```

Аналіз XML-коду, що містить відомості про класифікацію зв'язків, показує, що крім інформації про родовідових відносинах зв'язків, в ньому містяться дані про найменування, нотації, унікальному ідентифікатор кожної зв'язку і типі порту, до якого відноситься дана зв'язок. Таким чином, в запропонованому раніше способі подання класифікації зв'язків з допомогою нотацій моделі RDF не були враховані найменування зв'язків, їх ідентифікатори і типи портів. ці відомості можна уявити за допомогою триплетів виду:

```
<«Нотація зв'язку» еквівалентна «Найменування зв'язку»>
```

```
<«Нотація зв'язку» має ідентифікатор «Ідентифікатор зв'язку»>
```

```
<«Нотація зв'язку» має тип порту «Тип порту»>
```

Для подання відповідних фактів на мові RDF введемо в розробляється словник предикати «equivalent», «hasID» і «hasPortType» відповідно.

У подібному вигляді опис класифікації зв'язків з допомогою RDF представляється практично повним. Однак з метою спрощення процедури подальшої формалізації з використанням RDF фактів, які з УФО-моделі предметної області, пропонується доповнити формальне уявлення класифікації зв'язків додатковими видами триплетів. Так, при формалізації записи триплетів, що відображає факт наявності конкретної зв'язку, потрібне використання предиката, що відображає характер зв'язку з цим. Однак найменування зв'язку може не відображати в повною мірою того відносини,

яке встановлюється між УФО- елементами (об'єктом і суб'єктом в RDF- предикате). Тому пропонується для кожного класу зв'язків ввести відповідні найменування прямого і зворотного дій (відносин), які встановлюються між УФО- елементами при наявності між ними зв'язку даного класу. відповідно, це повинно знаходити відображення в формальній записи класифікації зв'язків, використовуваної для УФО-моделі. Для цього можуть бути використані спеціальні триплети виду:

<Клас зв'язку> предполагаєПрямоеОтношение <найменування прямого відношення>

<Клас зв'язку> предполагаєОбратноеОтношение <Найменування зворотного відносини>

Сукупність описаних триплетів становить формальне RDF- уявлення класифікації зв'язків і може служити джерелом фактів про зв'язках, що входять в конкретну УФО-модель. Для запису таких триплетів потрібні предикати, які означають відносини «передбачає пряме ставлення» і «передбачає зворотне ставлення». Пропонується в даному випадку використовувати предикати «hasDirectRelation» і «hasReverseRelation», які будуть включені в спеціалізований для формалізації фактів з УФО-моделі RDF-словник.

Таким чином, для подання класифікації зв'язків, використовуваної в УФО-моделі, на мові RDF потрібні предикати, представлені в таблиці 2.8.

Таблиця 2.8 - Предикати для подання класифікації зв'язків мовою RDF.

Найменування відносини	предикат	Повне уявлення предиката в нотації N-Triples мови RDF
є різновидом (підкласом)	subClassOf	<http://www.w3.org/2000/01/rdfschema#subClassOf>

еквівалентна	equivalent	<http://www.ufo-toolkit.ru/#equivalent>
має ідентифікатор	hasID	<http://www.ufo-toolkit.ru/#hasID>
має тип порту	hasPortType	<http://www.ufo-toolkit.ru/#hasPortType>
передбачає пряме відношення	hasDirectRelation	<http://www.ufotoolkit.ru/#hasDirectRelation>
передбачає зворотне відношення	hasReverseRelation	<http://www.ufotoolkit.ru/#hasReverseRelation>

Розглянемо безпосередньо уявлення фактів груп V і V (inv) класифікації на мові RDF. З урахуванням запропонованого раніше способу формального уявлення зв'язків між УФО-елементами у вигляді матриці зв'язків, а також сформульованих раніше особливостей подання класифікації зв'язків, використовуваної в УФО-моделі, можна запропонувати наступний алгоритм формального запису відповідних фактів з допомогою тієї чи іншої нотації RDF.

1. Знайти елемент матриці зв'язків, рівний 1.
2. Для «прямих» фактів (група V):
 - Як об'єкт виступає вузол, до якого відноситься виходить зв'язок. Нотація відповідного вузла міститься в верхньому індексі заголовка стовпчика.
 - Як суб'єкт виступає вузол, до якого відноситься вхідний зв'язок. Нотація відповідного вузла міститься в верхньому індексі заголовка рядка.
 - Як предиката виступає пряме відношення, що відповідає класу поточної зв'язку. Відповідно, для отримання предиката необхідно:

- Визначити клас розглядається зв'язку. Нотація відповідного класу є заголовком стовпця і рядка.

- У формальній записи класифікації зв'язків, використовуваної в даній УФО моделі, знайти триплет, що задає відповідність знайденому класу зв'язків предиката, що позначає прямий стосунок.

3. Для зворотних фактів (група V (inv)):

- Як об'єкт виступає вузол, до якого відноситься вхідний зв'язок. Нотація відповідного вузла міститься в верхньому індексі заголовка рядка.

- Як суб'єкт виступає вузол, до якого відноситься виходить зв'язок. Нотація відповідного вузла міститься в верхньому індексі заголовка стовпчика.

- Як предиката виступає зворотне відношення, що відповідає класу поточної зв'язку. Відповідно, для отримання предиката необхідно:

- Визначити клас розглядається зв'язку. Нотація відповідного класу є заголовком стовпця і рядка.

- У формальній записи класифікації зв'язків, використовуваної в даній УФО-моделі, знайти триплет, що задає відповідність знайденому класу зв'язків предиката, що позначає прямий стосунок.

Розглянемо подання на мові RDF фактів групи VI класифікації. Дана група містить відомості про декомпозиції УФО-елементів моделі. Безліч входять до складу декомпозиції УФО-елементів можна представити у вигляді декількох фактів виду:

(Для кожного $q=1..Q$) УФО k m має в якості частиник m_q F „

де Q - кількість УФО-елементів, на які декомпоziрован розглянутий УФО-елемент верхнього рівня, k - рівень, на якому розташований декомпоziруемый УФО-елемент, m - порядковий номер декомпоziруемого УФО-елемента, УФО k m - декомпоziруемый УФО-елемент, k m_q F - УФО-елемент рівня декомпозиції. Ставлення «мати в якості частини» в RDF можна виразити за допомогою предиката `hasPart`, що входить до складу словника `schema.org`. В нотації N-Triples факти групи VI будуть виглядати наступним чином:

`<http://www.ufo-toolkit.ru/#УФО $k$   $m$ > <http://schema.org/hasPart>`

$\langle \text{http://www.ufo-toolkit.ru/\#Fk } m_q \rangle$.

Сьома група фактів, сформульована в класифікації, стосується агрегації декількох UFO-елементів одного рівня моделі. До цієї групи входять факти виду:

(Для кожного $q = 1..Q$) $k \ m_q \ F$ є частиною $\text{UFO}_k \ m$, де Q - кількість UFO-елементів, на які декомпозирован розглянутий UFO-елемент верхнього рівня, k - рівень, на якому розташований декомпозіруемый UFO-елемент, m - порядковий номер декомпозіруемого UFO-елемента, $k \ m_q \ F$ - входить в агрегацію UFO-елемент, $\text{UFO}_k \ m$ - підсумковий UFO-елемент.

Предикат, що відображає ставлення «бути частиною», також входить в загальнодоступний словник `schema.org`, отже, створення окремого елемента для даного відносини в спеціалізованому Словник не виправдано. Тому для запису наведеного раніше безлічі фактів будемо використовувати посилання на сутність `isPartOf`, включену в словник `schema.org`. З використанням синтаксису записи N-Triples отримаємо Q фактів (По числу UFO-елементів на рівні декомпозиції) виду:

$\langle \text{http://www.ufo-toolkit.ru/\#Fk } m_q \rangle \ \langle \text{http://schema.org/isPartOf} \rangle$

$\langle \text{http://www.ufo-toolkit.ru/\#UFO}_k \ m \rangle$,

де q - порядковий номер UFO-елемента рівня декомпозиції, k - рівень, на якому розташований декомпозіруемый UFO-елемент, m – порядковий номер декомпозіруемого UFO-елемента.

Перетворивши цю конструкцію в граф, отримаємо зображення, наведене на рисунку 2.13.

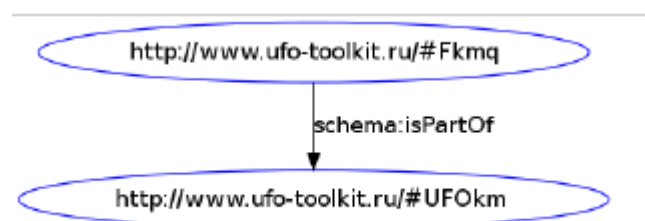


Рисунок 2.13 - Вистава у вигляді графа триплета про наявність у UFO-

елемента декомпозиції

Відзначимо, що предикати «hasPart» і «isPartOf», використовувані для записи фактів груп VI і VII, є протилежними за значенням. Відповідно, ставлення між ними можна представити за допомогою окремого триплета, в якому в якості предиката виступить згаданий раніше owl: inverseOf. Такий триплет буде постійним, так не буде залежний від аналізованої предметної області. Вид такого триплета в нотації NTriples наведено нижче:

```
<http://schema.org/hasPart> <http://www.w3.org/2002/07/owl#inverseOf>  
<http://schema.org/isPartOf> .
```

Таким чином, для перетворення УФО-моделей в конструкції формальної мови уявлення онтологій RDF потрібно безліч предикатів, що становить відповідний RDF-словник, представлений в додатку А.

Як випливає з таблиці А.1, для подання УФО-моделі з допомогою мови RDF вводиться ряд спеціальних предикатів. Для того щоб упорядкувати їх, введемо відповідні класи, по відношенню до яких зазначені предикати будуть властивостями. Це можливо за допомогою застосування засобів RDFS (RDF Schema, або «схема RDF») – розширення мови RDF, що дозволяє використовувати набір класів і властивостей для опису онтологій. Оскільки більшість розглянутих сутностей входять в спеціально розроблений RDF-словник для опису УФО-моделей, їх імена (URI) будуть починатися однаково. Для скорочення записи використовуємо так зване простір імен, яке дозволяє використовувати замість довгого URI (в даному випадку <http://www.ufotoolkit.ru/> ...) скорочене (ufo: ...). Для кожного введенного властивості також вводиться пара триплетів (наведені в нотації N Triples мови RDF), які дозволяють задати правила використання даного властивості як предиката в інших триплетах, а саме класи суб'єкта та об'єкта таких триплетів. Клас суб'єкта в цьому випадку визначає триплет з предикатом rdfs: domain, а клас об'єкта - триплет з предикатом rdfs: range.

1. УФО-елемент (ufo: UFO) Даний клас характеризує

УФО-елемент в цілому. його екземпляри (Сутності) можуть бути використані для опису УФО-елементів аналізованої моделі, а також їх декомпозиції і агрегації.

- *ufo:hasPart*

<ufo:hasPart> <rdfs:domain> <ufo:UFO> .

<ufo:hasPart> <rdfs:range> <ufo:UFO> .

- *ufo:isPartOf*

<ufo:isPartOf> <rdfs:domain> <ufo:UFO> .

<ufo:isPartOf> <rdfs:range> <ufo:UFO> .

2. Вузол (ufo: UFONode)

Цей клас відповідає за опис вузлів як складових УФО- елементів системно-об'єктної моделі. Відповідно, його властивості – це предикати, що описують певні характеристики вузла:

- *ufo:isBalancedBy*

<ufo:isBalancedBy> <rdfs:domain> <ufo:UFONode> .

<ufo:isBalancedBy> <rdfs:range> <ufo:UFOFunction> .

- *ufo:isRepresentedBy* *<ufo:isRepresentedBy> <rdfs:domain> <ufo:UFONode> .*

<ufo:isRepresentedBy> <rdfs:range> <ufo:UFOObject> .

- *ufo:hasPort*

<ufo:hasPort> <rdfs:domain> <ufo:UFONode> .

<ufo:hasPort> <rdfs:range> <ufo:Port> .

- *ufo:hasInputPort*

<ufo:hasInputPort> <rdfs:domain> <ufo:UFONode> .

<ufo:hasInputPort> <rdfs:range> <ufo:Port> .

- *ufo:hasOutputPort*

<ufo:hasOutputPort> <rdfs:domain> <ufo:UFONode> .

<ufo:hasOutputPort> <rdfs:range> <ufo:Port> .

- *ufo:hasInputPortClass*

<ufo:hasInputPortClass> <rdfs:domain> <ufo:UFONode> .

<ufo:hasInputPortClass> <rdfs:range> <ufo:Link> .

- *ufo:hasOutputPortClass*

<ufo:hasOutputPortClass> <rdfs:domain> <ufo:UFONode>

<ufo:hasOutputPortClass> <rdfs:range> <ufo:Link> .

3. Функція (ufo: UFOFunction)

Зазначений клас відповідає поняттю «функція UFO-елемента». Його властивості використовуються як предикати в триплетах, що характеризують відомості про функції конкретних UFO-елементів описуваної за допомогою мови RDF UFO-моделі.

- *ufo:isRealizedBy*

<ufo:isRealizedBy> <rdfs:domain> <ufo:UFOFunction> .

<ufo:isRealizedBy> <rdfs:range> <ufo:UFOObject> .

- *ufo:balances*

<ufo:balances> <rdfs:domain> <ufo:UFOFunction> .

<ufo:balances> <rdfs:range> <ufo:UFONode> .

- *ufo:translateInput*

<ufo:translateInput> <rdfs:domain> <ufo:UFOFunction>

<ufo:translateInput> <rdfs:range> <ufo:Link> .

- *ufo:giveOutput*

<ufo:giveOutput> <rdfs:domain> <ufo:UFOFunction> .

<ufo:giveOutput> <rdfs:range> <ufo:Link> .

4. Об'єкт (ufo: UFOObject)

Аналогічно двом попереднім класам, ufo: UFOObject представляє є віддзеркаленням в мові RDF однією зі складових UFO-елемента - об'єкта. Як його властивостей задаються предикати, які використовуються в триплетах для опису об'єктних характеристик UFO-елементів системно об'єктної моделі предметної області.

- *ufo:realizes*

<ufo:realizes> <rdfs:domain> <ufo:UFOObject> .

<ufo:realizes> <rdfs:range> <ufo:UFOFunction> .

- *ufo:represents*

<ufo:represents> <rdfs:domain> <ufo:UFOObject> .

<ufo:represents> <rdfs:range> <ufo:UFONode> .

- *ufo:has Attribute*

<ufo:represents> <rdfs:domain> <ufo:UFOObject> .

<ufo:represents> <rdfs:range> <rdfs:Literal> .

5. Клас зв'язку (ufo: LinkClass)

Даний клас вводиться для представлення даних про класифікацію зв'язків, яка використовується в будь-якій УФО-моделі. характеристики класу зв'язків УФО-моделі, відповідно, можуть бути подані мовою RDF за допомогою триплетів, в яких в якості предикатів беруть участь такі властивості класу ufo: LinkClass:

- *ufo:hasID*

<ufo:hasID> <rdfs:domain> <ufo:LinkClass> .

<ufo:hasID> <rdfs:range> <rdfs:Literal> .

- *ufo:hasPortType*

<ufo:hasPortType> <rdfs:domain> <ufo:LinkClass> .

<ufo:hasPortType> <rdfs:range> <rdfs:Literal> .

- *ufo:equivalent*

<ufo:equivalent> <rdfs:domain> <rdfs:Literal> .

<ufo:equivalent> <rdfs:range> <rdfs:Literal> .

- *ufo:hasDirectRelation*

<ufo:hasDirectRelation> <rdfs:domain> <ufo:LinkClass> .

<ufo:hasDirectRelation> <rdfs:range> <rdfs:Literal> .

- *ufo:hasReverseRelation*

<ufo:hasReverseRelation> <rdfs:domain> <ufo:LinkClass> .

<ufo:hasReverseRelation> <rdfs:range> <rdfs:Literal> .

6. Зв'язок (ufo: Link) - цей клас є поданням в мові RDF такої складової УФО-моделі, як зв'язок між УФО-елементами.

7. Порт (ufo: Port) - використовується для опису на мові RDF відомостей про портах вузла - складової УФО-елемента. Наведені конструкції дозволяють детально описати УФО-модель предметної області на мові RDF з урахуванням всіх особливостей УФО- підходу. Це свідчить про можливість вилучення знань про предметної області з графоаналитического системно-об'єктної моделі та побудови на їх основі онтології.

2.5. Висновки до розділу

У другому розділі проаналізовано математичний апарат УФО- підходу і засоби формального уявлення онтологій, виявлена їх взаємозв'язок. Окреслено основні положення, які є основою інтеграції системно-об'єктного підходу та засобів онтологічного інжинірингу.

З метою систематизації та формалізації набору знань про предметну області, що містяться в УФО-моделі, сформульована класифікація відповідних фактів - триплетів, що характеризують одиницю відомостей про предметної області. Для кожної групи фактів, що входить в розроблену класифікацію, наведено докладний формальне обґрунтування, що дозволяє формалізувати витяг відповідних фактів з УФО-моделей і здійснити їх подальше перетворення в цілях подання на мові RDF.

В рамках даної глави сформований спеціалізований RDF- словник, необхідний для запису витягнутих з УФО-моделі фактів на формалізованому мові представлення онтологій RDF. Введені спеціальні предикати, відсутні в загальноприйнятих RDF-словниках, представлені як властивості нових класів, що вводяться в RDF-опис онтології засобами розширення мови RDF - RDFS. дані класи дозволяють описувати в онтології основні поняття і сутності УФО-підходу, такі як УФО-елемент, Вузол, Функція, Об'єкт, зв'язок, класифікація зв'язків. Результати даної частини дослідження дозволяють сформулювати основу для розробки положень методу побудови онтологій на основі візуальних графоаналитических УФО-моделей, а також забезпечити можливість алгоритмізації і автоматизації окремих складових даного методу.

РОЗДІЛ 3. РОЗРОБКА ЗАСОБІВ ПОБУДОВИ ОНТОЛОГІЇ НА ОСНОВІ СИСТЕМНО-ОБ'ЄКТОВОГО ПІДХОДУ. РОЗРОБКА І АПРОБАЦІЯ АЛГОРИТМІЧНЕ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПОБУДОВИ ОНТОЛОГІЇ НА ОСНОВІ УФО-МОДЕЛІ ТА ЗДІЙСНЕННЯ ЛОГІЧНОГО ВИВЕДЕННЯ НА УФО-МОДЕЛЕЙ

3.1. Розробка методу побудови онтології з використанням візуальної графоаналитического системно-об'єктного УФО-моделі.

Обґрунтована в розділі 2 можливість вилучення відомостей про предметної області з графоаналитического УФО-моделі, а також можливість подання УФО-моделі за допомогою мови RDF дозволяють сформулювати метод побудови онтології на основі результатів УФО-аналіза предметної області. Схема взаємозв'язку основних складових даного методу представлена на рисунку 3.1.



Рисунок 3.1 - Схема методу побудови онтології на основі результатів УФО-аналізу.

У загальному вигляді послідовність дій, складових запропонований метод побудови онтологій, виглядає наступним чином:

1. Моделювання виробництва хліба з використанням засобів УФО-підходу (побудова УФО-моделі).
2. Витяг з УФО-моделі фактів про виробництва хліба.
3. Представлення отриманих фактів на мовах зразок RDF, OWL.
4. Затвердження (оцінка) отриманого формального уявлення онтології.
5. Використання онтології виробництва хліба.
6. Модифікація УФО-моделі, коригування онтології.

На початку процесу побудови онтології, згідно із запропонованим методом, здійснюється аналіз виробництва з використанням УФО-підходу.

Даний етап фактично замінює процедури відбору концептів для включення в онтологію, а також виявлення зв'язків між ними. В традиційних підходах, як правило, зазначені процедури виконуються вручну експертами предметної області. У розробленому методі побудови онтологій джерелом концептів і даних про їх взаємозв'язках служить УФО-модель предметної області, яка акумулює в собі сукупність знань експерта або групи експертів про моделюється предметної області. Одними з переваг використання саме візуальних системнооб'єктних моделей в даному випадку можна вважати їх простоту і універсальність - вони не вимагають спеціальних навичок для побудови і евристичного аналізу, а також не мають прив'язки до будь-якої певній предметній області.

Побудована Графоаналітична УФОмодель предметної області може застосовуватися не тільки для цілей онтологічного інжинірингу, але і для моделювання і реінжинірингу бізнес-процесів, аналізу структури систем, дослідження взаємозв'язків сутностей і рішення інших аналітичних завдань. створена одного разу УФО-модель предметної області може бути в подальшому актуалізована і доповнена, що дозволяє забезпечувати актуальність джерела даних для побудови онтології. Оскільки інформація, видобута на першому етапі УФО-модель є джерелом даних про концептах і їх взаємозв'язках, а також про основні аксіомах, які будуть включені в онтологію, на наступному етапі здійснюється витяг зазначених даних з моделі. Для цього використовується запропонована раніше класифікація фактів про предметну область, що містяться в УФО-моделях. Формальне представлення УФОмоделі аналізованої предметної області аналізується на предмет наявності всіх видів фактів згідно групам, наведеними в класифікації. Виявлені факти використовуються в якості аксіом, що складають основу розроблюваної онтології.

Процес безпосереднього перетворення вихідного формального уявлення УФО-моделі в конструкції мов опису онтологій, таких як RDF і OWL, займає центральне місце в запропонованому методі побудови онтологій. Саме на даному етапі відбувається перехід від візуальної системно-об'єктної моделі до прототипу онтології. В цілях автоматизації процесу подібного перетворення в якості вихідного формалізованого представлення графоаналітического УФО-моделі пропонується використовувати XML-файл, який генерується при моделюванні предметної області в спеціалізованому редакторі UFOToolkit.

Перевагою такого представлення вихідних даних є те, що більшість коштів формальної записи онтологій, такі як RDF і OWL, підтримують нотації, засновані на XML, що спрощує процес перетворення. На етапі перетворення витягнуті з УФО-моделі факти про предметної області трансформуються з вихідного уявлення в конструкції спеціалізованих мов, призначених для формального опису онтологій (RDF, OWL). оскільки класифікація видобутих фактів строго визначає вихідні дані, а перетворення в конструкції мов уявлення онтологій здійснюється з урахуванням синтаксису вибраної мови

(наприклад, RDF), процес трансформації може бути представлений у вигляді алгоритму і згодом автоматизований.

Створювані в ході перетворення конструкції мови уявлення онтологій записуються в результуючий файл, який служить формальним поданням онтології, створеної на базі УФО-моделі предметної області. Подібне формальне подання онтології у вигляді коду перед безпосереднім застосуванням для вирішення прикладних завдань потребує перевірки на коректність.

Способи і засоби такої перевірки вивчаються в рамках розвитку такого поняття і напрямки дослідження, як оцінка онтологій. Відзначимо, що на даному етапі необхідно переконатися, що, по-перше, формальна запис онтології коректна з точки зору синтаксису і семантики мови, на якому вона представлена, а, по-друге, в тому, що дані, що містяться в онтології, несуперечливі з урахуванням принципів УФО-підходу. Після отримання формального уявлення онтології, сформованого на основі результатів УФО-аналізу предметної області, здійснюється перехід до найбільш оптимальному в контексті розв'язуваної завдання способу подання онтологій: у вигляді графів, у вигляді коду в одній з нотацій мови RDF, OWL чи іншого синтаксису.

Наступними складовими запропонованого методу є безпосереднє використання отриманої онтології, паралельно з яким здійснюється її актуалізація і розвиток. підпроцес підтримання актуальності онтології має спільні риси з описаними раніше етапами побудови онтології. Так, при зміні предметної області і / або уточнення, розширення знань експертів про неї зміни вносяться в джерело даних про концептах і аксіомах - УФО-модель, побудовану на початковому етапі. З заданою періодичністю або безпосередньо при внесенні змін у вихідну модель виконується повторне перетворення такої моделі в формальне подання онтології на мові RDF. Отриманий файл порівнюється з наявним. Якщо в них знайдені розбіжності, результуючий файл (файл онтології) замінюється на актуальний. При цьому, в залежності від обсягів даних і розв'язуваних задач, можлива як повна заміна результуючого файлу, так і оновлення тільки окремих, що потребують внесення змін фрагментів. Отримане в результаті нове або оновлене формальне подання онтології піддається валідації з метою забезпечення відсутності логічних помилок і помилок синтаксису. Якщо таких не знайдено, оновлена онтологія може вважатися готовою до використання, в Інакше знайдені помилки необхідно усунути.

Таким чином, всі зміни, внесені в УФО-модель аналізованої предметної області, транслуються в розроблену на основі УФО-моделі онтологію. Отже, онтологія підтримується в актуальному стані. Описаний метод дозволяє не тільки розробити онтологію, описує предметну область, змодельовану засобами УФОпідходу, а й забезпечити її постійну актуалізацію з урахуванням всіх внесених у вихідну візуальну графоаналитического УФО-модель змін. Таким чином, у порівнянні з відомими підходами і методами побудови онтологій запропонований метод має низку переваг. В відміну від

методології Ушолда і Кінга, розроблений метод в повній мірою відповідає загальноприйнятій на поточний момент життєвого циклу онтології і має на увазі досить формалізовану техніку відбору концептів, що включаються в онтологію, оскільки перелік таких концептів витягується за заданими правилами з УФО-моделі предметної області.

Запропонований метод є незалежним від програми, так як в основі онтології лежить універсальна системно-об'єктна модель, всебічно характеризує предметну область. Дана властивість є перевагою в порівнянні з методологією побудови онтологій Грюнінгера і Фокса. Метод побудови онтологій на основі візуальних графоаналитических системно-об'єктних моделей в частині циклічності процесу розробки, а саме, ітеративності етапу актуалізації онтології, схожий з принципами проекту On-To-Knowledge. Однак в запропонованому в даній роботі метод відсутня проблема вибору критерію зупину, оскільки процес внесення змін до онтологію запускається тоді, коли в цьому є необхідність, тобто при внесенні змін у вихідну УФО-модель. Ще однією перевагою розробленого методу в порівнянні з існуючими підходами до побудови онтологій (наприклад, Diligent) є прогнозованість часу, необхідного на створення онтології.

Це можливо, зокрема, завдяки автоматизації процесу перетворення УФО-моделі в опис онтології на одному з формальних мов, що також веде до скорочення безпосередньої участі експертів в процесі підготовки основи онтології.

Це пов'язано з тим, що, як зазначалося раніше, рішення про включенні концептів і відносин між ними в онтологію приймаються за заданими правилами відповідно до розроблених методами, а не на основі суб'єктивних рішень кожного учасника процесу, як, наприклад, в методології SENSUS. На відміну від підходів до побудови онтологій, які передбачають витягів знань з текстів, метод створення онтологій на основі системно-об'єктного підходу дозволяє повною мірою дотриматися принцип розширюваності за рахунок можливості актуалізації онтології при доповненні вихідної УФО-моделі. Незважаючи на схожість принципу використання моделі предметної області, що лежить в основі запропонованого методу розробки онтологій і методології UPON, новий метод має низку переваг. Зокрема, при використанні в Як основу для побудови онтології системно-об'єктної моделі знижується ризик виникнення проблеми використання результату для рішення заздалегідь невизначених завдань (оскільки УФО-модель, на відміну від use-case діаграм, що застосовуються в UPON, всебічно характеризує предметну область як систему).

Відносно невелика кількість ітерацій процесу створення онтології в запропонованому методі по порівняно з методологією UPON дозволяє знизити навантаження на учасників процесу розробки онтологій. Отже, розроблений метод побудови онтологій на основі візуальних графоаналитических системно-об'єктних моделей дозволяє вирішити ряд відомих проблем

онтологічного інжинірингу, що обумовлює його переваги в порівнянні з існуючими підходами і методологіями.

3.2. Особливості валідації онтології, побудованої на основі графоаналитического УФО-моделі

Відзначимо, що і в процесі первинного побудови онтології з допомогою запропонованого методу, і згодом при підтримці її повноти та актуальності шляхом внесення змін необхідним кроком є оцінка та валідація онтології.

Даний процес вимагає більш детального розгляду. Оцінка онтології (evaluation) являє собою комплексний термін, який в різних дослідженнях інтерпретується по-різному. Так, наприклад, Бранк і Гробельник описують оцінку онтологій як процес прийняття рішення про якість онтології за певним критерієм і виборі відповідно до таким рішенням найбільш придатною з наявних онтологій. Інші автори поділяють оцінку онтологій на верифікацію (verification) і валідацію (validation), причому під верифікацією розуміють аналіз коректності самого процесу створення онтології, а під валідацією - перевірку відповідності отриманої онтології реального світу, тобто предметної області, яку вона описує.

Систематизуючи основні дослідницькі праці в даному напрямку, можна виділити основні, найбільш поширені розуміння оцінки онтології: — аналіз відповідності онтології вимогам додатка, в якому вона буде використовуватися; — перевірка на наявність дубльованих і / або некоректно згрупованих концептів, наприклад, при об'єднанні кількох онтологій або побудові онтології на основі декількох різних джерел даних; — оцінка повноти і точності словника предметної області, що міститься в онтології; — вибір кращої онтології з наявних. Процес оцінки онтології є невід'ємною частиною її життєвого циклу. Даний процес може бути як окремим етапом, так і свого роду надпроцесом, впровадженим в усі етапи життєвого циклу онтології: від прототипування до безпосереднього використання готової онтології. В контексті розробляється методу побудови онтології на основі графоаналитических УФО-моделей доцільно розглядати оцінку (валідацію) онтології в якості окремої складової такого методу. Дійсно, етап підготовки даних, розробки структури і прототипування онтології виконується в рамках УФО-аналізу предметної області, а отримання формального уявлення онтології засноване на використанні його результатів і може бути в значною мірою автоматизовано. Відповідно, оцінка отриманого уявлення онтології повинна виконуватися на наступному кроці, безпосередньо перед переходом до використання такої онтології. Як правило, в якості джерела відомостей про предметну область в цілях побудови онтологій буде використовуватися одна загальна акумулює знання експертів УФО-модель, а не кілька різних. Отже, найбільшу важливість на етапі оцінки онтології буде мати процес її валідації.

Під валідацією при цьому розуміється оцінка несуперечності онтології і відповідність її формального уявлення регламентованим правилам відповідної мови (Формату).

Залежно від понять, які вкладаються дослідниками в оцінку онтологій, виділяють різні підходи і методи такої оцінки. Варіант їх систематизації наводиться, наприклад, в роботі [76]. автори дослідження [68] виділяють три основні підходи до оцінки та валідації онтологій, засновані на оцінці зміни (розвитку) онтологій зі часом, на правилах і на метриках відповідно.

Розглянемо детальніше підхід, пов'язаний з використанням правил. З метою забезпечення валідації онтології, побудованої на базі візуальної графоаналитического системно-об'єктної моделі, необхідно застосування правил, передбачених стандартом формату представлення онтології, і деякого спеціального набору додаткових правил.

Стандартні правила служать для забезпечення відповідності уявлення онтології нормам синтаксису мови, зокрема, RDF. подібна перевірка може бути виконана вручну фахівцем в області онтологічного інжинірингу, а також в автоматизованому режимі за допомогою будь-якого з існуючих валідаторів, як вбудованих в редактори онтологій і вирішувачі (Pellet Integrity Constraints; Fluent Editor, an Ontology Editor; Protégé), так і реалізованих у вигляді окремих програмних засобів (RDF Validator; rdfabout.com's Validator; Total Validator і інші). Перевірка RDF і OWL-коду на відповідність синтаксису мови і відсутність логічних помилок, пов'язаних зі стандартними предикатами, досить добре вивчена, як правило, виконується автоматично і не вимагає окремого опису.

У свою чергу, особливий інтерес представляють додаткові правила перевірки, які необхідні для перевірки несуперечності онтології з урахуванням специфіки понять і принципів УФО-підходу.

Розглянемо дані правила на природній мові, а також їх подання на мові правил семантичного вебу (Semantic Web Rule Language, SWRL). Дана мова ґрунтується на принципі «openworld» - відкритого світу. Цей принцип передбачає наступне: деякий твердження може бути істинним, якщо ми знаємо, що воно істинне, або не знаємо, істинно воно або помилково. Подібний підхід є протилежним по відношенню до принципу «closedworld» - закритого світу, який, в свою чергу, передбачає, що деяке твердження істинно тільки тоді, коли нам відомо, що воно правдиве в іншому випадку твердження вважається хибним.

Правила, написані на SWRL, складаються з консеквента і антецендент, тобто заголовка і тіла відповідно. консеквент є істинним тоді, коли антецендент порожній або всі його складові істинні.

Для забезпечення простоти сприйняття людиною використовуємо так званий «человекочитаемый» синтаксис мови SWRL (Human Readable Syntax), який дозволяє представляти SWRL-правила у вигляді: антецендент (Тіло, умова) $\Rightarrow$ консеквент (заголовок, наслідок) [67]. необхідно враховувати, що SWRL не підтримує заперечення, тому правила валідації в більшості випадків

будуть будуватися «від зворотного», тобто бути перевіркою на наявність в онтології некорретних фактів.

1. Для кожної зв'язку повинен бути зазначений клас:

$$hasClass(?L, null) \Rightarrow hasNoLinkClass(?L), (3.1)$$

де? L - змінна, в якості значення якої може виступати будь-яка зв'язок між вузлами в УФО-моделі, тобто сутність «Зв'язок» в онтології; ? LC - змінна, в якості значення якої може виступати один з примірників класу «Клас зв'язку».

2. Для кожної зв'язку повинен бути зазначений тільки один клас: $hasClass(?L, ?LC1) \wedge hasClass(?L, ?LC2) \Rightarrow hasManyLinkClasses(?L), (3.2)$ де? L - змінна, в якості значення якої може виступати будь-яка зв'язок між вузлами в УФО-моделі, тобто сутність «Зв'язок» в онтології; ? LC1 і? LC2 - змінні, як значення яких можуть виступати екземпляри класу «Клас зв'язку».

3. В онтології не повинно бути декількох сутностей з однаковим значенням ідентифікатора (ID):

$$rdfs : hasID(?x, ?id) \wedge rdfs : hasID(?y, ?id) \Rightarrow hasSameID(?x, ?y), (3.3)$$

де? x і? y - деякі змінні, як значення яких можуть виступати будь-які сутності онтології; ? Id - строкове або целочисленное значення ідентифікатора. Якщо консеквент цього виразу приймає істинне значення, значить, онтологія містить помилку: дві сутності мають один і той же значення властивості ID. Це може свідчити або про помилку в вихідних даних, тобто в формальному поданні УФО-моделі, або про помилку, що виникла в процесі трансформації УФО-моделі в онтологію.

4. Одна і та ж функція не повинна балансувати більше одного вузла:

$$balances(?f, ?u1) \wedge balances(?f, ?u2) \Rightarrow balancesManyNodes(?f), (3.4)$$

де? f - змінна, в якості значення якої може виступати функція (Як складова УФО-елемента); ? U1,? U2 - змінні, як значень яких можуть виступати вузли (як складові УФО-елемента).

5. Один і той же об'єкт не повинен займати більше одного вузла:

$$represents(?o, ?u1) \wedge represents(?o, ?u2) \Rightarrow representsManyNodes(?o), (3.5)$$

де? o - змінна, в якості значення якої може виступати об'єкт (Як складова УФО-елемента); ? U1,? U2 - змінні, як значень яких можуть виступати вузли (як складові УФО-елемента).

6. В онтології не повинно бути декількох об'єктів з однаковим ім'ям (значенням властивості *Bread*):

$$Bread(?o1, ?N) \wedge Bread(?o2, ?N) \Rightarrow same\ Bread\ Objects(?o1, ?o2), (3.6)$$

де? o1,? o2 - змінні, як значення яких можуть виступати об'єкти (як складові УФО-елемента); ? N - змінна, в якості значення якої може виступати деякий літерал, що позначає ім'я об'єкта.

Істинність консеквента говорить про те, що в онтології присутні дані про два об'єкти (маються на увазі об'єкти як складові УФО-елементів вихідної УФО-моделі) з однаковими іменами. дане правило не є критичним в тому сенсі, що присутність об'єктів з однаковими іменами в УФО-моделі допустимо, однак може свідчити про неточність в описі предметної області і сприяти виникненню логічних помилок при використанні онтології.

7. В онтології не повинно бути декількох функцій з однаковим ім'ям (значенням властивості Name).

$$\text{Bread} (?f1, ?N) \wedge \text{Bread} (?f2, ?N) \Rightarrow \text{same Bread Functions} (?f1, ?f2), (3.7)$$

де? f1,? f2 - змінні, як значення яких можуть виступати функції (як складові УФО-елемента); ? N - змінна, в якості значення якої може виступати деякий літерал, що позначає ім'я функції. Семантична інтерпретація даного правила аналогічна попередньому: наявність двох функцій (як складових УФО-елементів) з однаковими іменами говорить про можливі неточності в описі предметної області, представленому у вигляді УФО-моделі і, отже, в онтології, отриманої після виконання відповідного перетворення.

8. У онтології не повинно бути декількох вузлів з однаковою нотацією (значенням властивості Notation).

$$\text{Notation} (?u1, ?N) \wedge \text{Notation} (?u2, ?N) \Rightarrow \text{sameNotationNodes} (?u1, ?u2), (3.8)$$

де? u1,? u2 - змінні, як значення яких можуть виступати вузли (як складові УФО-елемента); ? N - змінна, в якості значення якої може виступати деякий літерал, що позначає нотацію вузла.

Зміст даного правила аналогічним двом вищевикладеним, тобто наявність в онтології відомостей про двох вузлах з однаковим значенням властивості «Нотація» може свідчити про помилки в поданні відомостей про предметну область, допущених як на етапі моделювання з застосуванням засобів УФО-підходу, так і на етапі перетворення отриманої графоаналитического моделі в онтологію. Сформульовані правила можуть бути представлені за допомогою інших нотацій мови SWRL, наприклад, з використанням синтаксису RDF / XML. В такому випадку правила подаються у вигляді самостійних аксіом і додаються в код онтології, отриманої на етапі перетворення з УФО-моделі предметної області. Таким чином, включення до складу онтології SWRL-правил валідації дозволяє підвищити рівень достовірності і знизити число логічних помилок, що містяться в онтології.

3.3. Розробка методу логічного висновку на візуальних графоаналитических УФО-моделях за рахунок їх інтеграції з формальними засобами записи онтології.

У сучасних умовах розвитку інформаційних технологій і засобів управління знаннями видається важливим забезпечення можливості машинної обробки візуальних моделей предметних областей, включаючи витяг містяться в них знань і їх обробку. Вітчизняним дослідником Жихарева А.Г. в рамках розробки системно-об'єктного методу подання знань (СОМПЗ) [81] був запропонований формалізований метод і алгоритм виведення на графоаналитических УФО-моделях організаційних знань, подаються засобами СОМПЗ.

Однак дані дослідження присвячені проблемі формалізованого подання та обробки саме організаційних знань, важливим цей аспект є в світлі проблем побудови онтологій на базі УФО-моделей, оскільки створювані онтології можуть охоплювати предметні області різного масштабу і різної

специфіки: від конкретного технологічного процесу до складних систем і цілих галузей знань експертів. Прикладами в останньому випадку можуть служити онтологія музики (The Music Ontology), проект «Відкриті біомедичні онтології », спеціалізована медична онтологія в області кардіології Heart Failure Ontology.

Відповідно, актуальним завданням є пошук універсального методу логічного висновку на графоаналитических УФО-моделях, що дозволяє здійснювати обробку акумульованих в них знань поза Залежно від предметної області і специфіки моделі. Одним із способів вирішення такого завдання є адаптація і використання поширених у світовій практиці, апробованих в різних умовах і динамічно розвиваються інструментів, прикладом яких можуть служити механізми логічного висновку на формальних уявленнях онтологій. Зокрема, уявлення УФО-моделей предметних областей з допомогою мови RDF, описане раніше, відкриває широкі можливості в даній галузі.

Сформоване в даному випадку формальне подання візуальної графоаналитического моделі у вигляді коду на мові RDF, а також що виникли на його основі модифікації, дозволяє зробити вихідну УФОмодель машиночитаємою. Наявність в мові RDFнесколько нотацій, конвертація між якими можлива за допомогою досить розповсюджених спеціальних програмних засобів, дозволяє використовувати найбільш зручну в тому чи іншому випадку. Так наприклад, нотація N-Triples, використовувана нами раніше, є більш простий для сприйняття людиною, в той час як нотація RDF / XML наближена до формату XML, що дозволяє використовувати її в інформаційних системах і тоді як сфера застосування УФО-підходу та коштів УФО-аналізу предметних областей значно ширше. Особливо спеціалізованих програмних засобах, обробляючи, в тому числі, з допомогою існуючих XML-парсерів.

Як наслідок такого методу обробки знань, що містяться в УФОмоделі предметної області, стає можливим здійснення процесу логічного висновку на онтологіях, основою яких є зазначені моделі. На основі механізмів логічного висновку і отримання знань з онтологій, створених на базі УФО-моделей, можуть бути побудовані інтелектуальні інформаційні системи, що включають в себе елементи експертних систем і систем підтримки прийняття рішень.

Визначимо коло основних завдань логічного висновку на побудованих на основі УФО-моделей онтологіях, які можуть бути корисними при рішенні проблем в прикладних областях. Такі завдання можна умовно згрупувати за типом даних, на основі яких вони будуються. В Залежно від предметної області і мети одержання знань формулювання задач може варіюватися.

Далі наведено класифікацію основних завдань логічного висновку на УФО-моделях, які можуть бути вирішені за допомогою засобів логічного висновку на онтологіях.

1. Завдання на основі даних про зв'язки.

- 1.1. Побудова класифікації зв'язків.

- 1.2. Висновок переліку підкласів заданого класу зв'язків.

- 1.3. Отримання повної інформації про заданому класі зв'язків.
2. Завдання на основі даних про вузли.
 - 2.1. Отримання докладної інформації про заданому вузлі.
 - 2.2. Визначення функції, що балансує заданий вузол.
 - 2.3. Отримання інформації про об'єкт, що займає даний вузол.
 - 2.4. Висновок переліку портів (всіх, вхідних, вихідних) заданого вузла.
 - 2.5. Визначення вузлів, які мають подібні порти з заданим.
 - 2.6. Формування переліку порожніх / зайнятих портів заданого вузла.
3. Завдання на основі даних про функції.
 - 3.1. Висновок докладної інформації про заданої функції.
 - 3.2. Побудова списку вузлів, що балансується заданою функцією.
 - 3.3. Отримання списку об'єктів, що реалізують задану функцію.
4. Завдання на основі даних про об'єкти.
 - 4.1. Відображення докладної інформації про заданому об'єкті.
 - 4.2. Виведення списку вузлів, займаних заданим об'єктом.
 - 4.3. Отримання функцій, що реалізуються конкретним об'єктом.
5. Завдання на основі даних про УФО-елементах в цілому.
 - 5.1. Формування переліку входів і виходів заданої функції (на основі даних про вузол, його портах і балансує його функції).
 - 5.2. Отримання інформації про зв'язки між двома заданими УФОелементами (на основі інформації про вузли, їх портах і існуючих в моделі зв'язках).
 - 5.3. Побудова ланцюжків взаємодії між заданими УФОелементами (на основі даних про вузли, їх портах, існуючих в моделі зв'язках).

Розглянемо докладніше метод логічного висновку на візуальних графоаналитических УФО-моделях за рахунок їх інтеграції з формальними засобами записи онтології.

В основі цього методу лежить ідея вилучення знань, що містяться в будь-який УФО-моделі, перетворенні їх до виду, для якого стає можливим застосування механізмів і програмних засобів логічного висновку на онтологіях.

Складовими зазначеного методу є перераховані нижче кроки (етапи):

- Вилучення знань про предметну область з комп'ютерної графоаналитического системно-об'єктної моделі.
- Подання витягнутих знань на формальній мові записи онтологій, тобто у вигляді, придатному для машинної обробки.
- Застосування інструментів логічного висновку (мови запитів або решателя) до отриманого формальному поданням онтології, побудованої на основі УФО-моделі.

Перераховане можна представити у вигляді алгоритму, блок-схема якого показана на рисунку 3.2

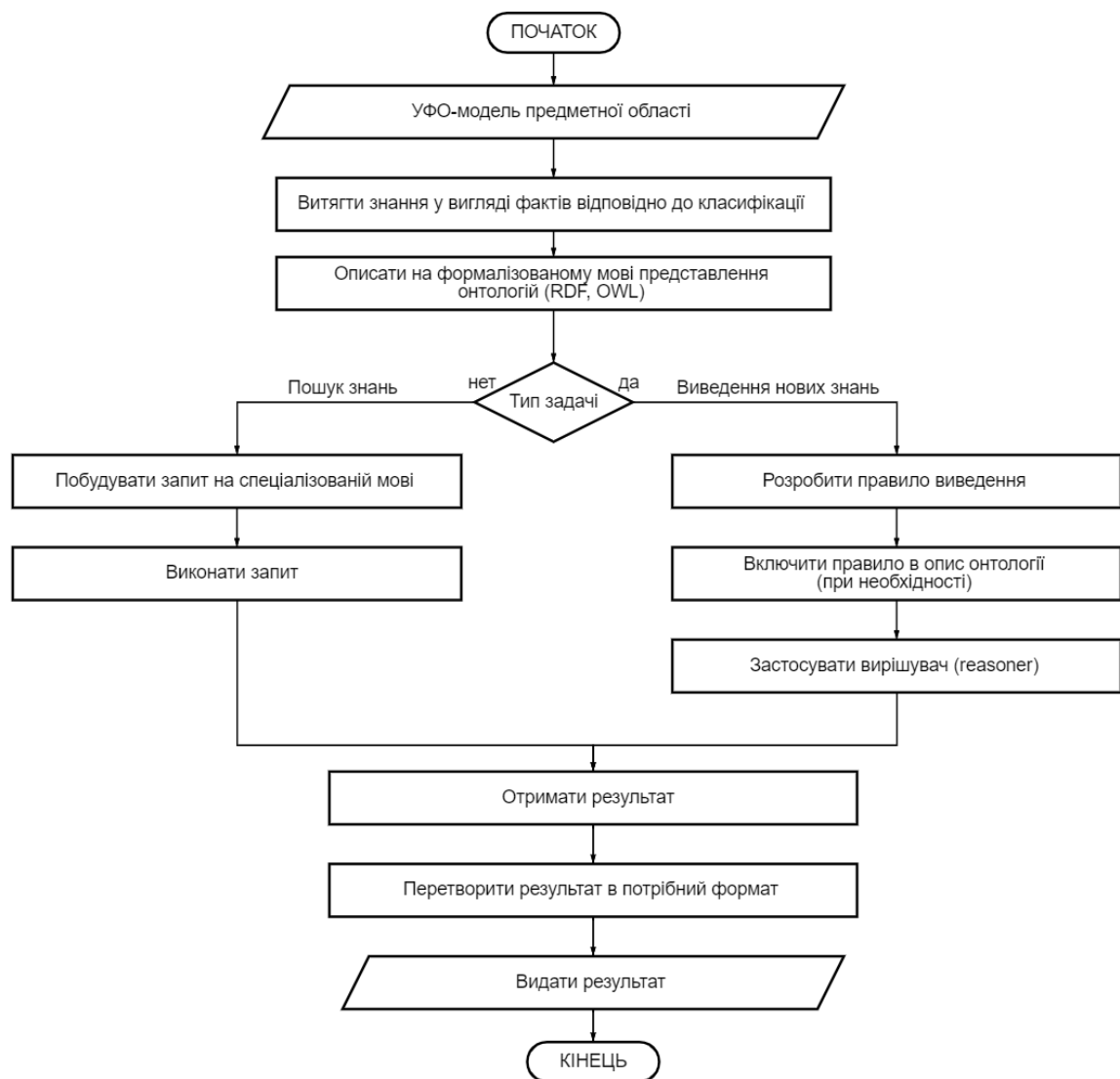


Рисунок 3.2 - Блок-схема алгоритму логічного висновку на UFO-моделях з допомогою засобів логічного висновку на онтологіях.

Дійсно, як було показано в главах 2 і 3, UFO-модель містить в собі сукупність знань про предметну область, які можуть бути представлені у вигляді окремих фактів. подібні факти систематизуються за допомогою класифікації, запропонованої в розділі 2.1 глави 2. Можливість отримання подібного набору фактів про предметну області обґрунтована за допомогою відповідної адаптації засобів формалізації UFO-підходу і описана в розділі 2.2 глави 2.

Таким чином, комп'ютерну графоаналитического UFO-модель можна вважати джерелом знань про предметну область. Крім знань, що містяться в такій моделі в явному вигляді і витягають із неї в процесі побудови онтології, з UFO-моделі можуть бути виведені нові знання про предметну область. Для їх отримання (Виведення) необхідно застосування відповідних механізмів і алгоритмів.

Як правило, подібні механізми піддаються автоматизації, і на їх основі розробляються спеціальні програмні засоби. Однак застосування подібних

алгоритмічних і програмних засобів можливо тільки до формалізованого поданням джерела знань, тобто УФОмоделі. Оскільки для вирішення завдань логічного висновку на УФО-моделях передбачається використовувати напрацювання, які існують для вирішення аналогічних завдань логічного висновку на онтологіях, в якості бази знань пропонується використовувати онтологію, побудовану на основі аналізованої УФО-моделі предметної області за описаним раніше методу. Іншими словами, пропонований метод логічного висновку знань на графоаналитических УФО-моделях заснований саме на переході від аналізованої УФО-моделі предметної області до онтології і застосуванні відповідних механізмів до отриманого формального опису побудованої онтології. Досвід світового дослідницького співтовариства дозволяє скористатися в цьому випадку кращими практиками застосування онтологій для вирішення прикладних завдань.

3.4. Дослідження застосовності існуючих засобів і інструментів для здійснення логічного висновку на візуальних графоаналитических УФО-моделях.

Використання онтологій в прикладних інформаційних системах на сьогоднішній день фактично є окремим дослідним напрямком, в рамках якого розроблено принципи, підходи, алгоритми, а також засновані на них бібліотеки і програмні засоби, що дозволяють здійснювати логічний висновок на онтологіях. В якості одного з найбільш поширених засобів обробки знань, що містяться в онтології, можна виділити мови запитів, такі як DQL, N3QL, RDFQ, RDQ, RDQL, SPARQL. Серед них найбільш вивченим і широко застосовуваним є SPARQL (SPARQL Protocol and RDF Query Language) - мова запитів до даних, представленим по моделі RDF, а також протокол для передачі цих запитів і відповідей на них. даний мова є рекомендацією консорціуму W3C. Запити, що формуються на мовою SPARQL, мають конструкцію, схожу з запитами до баз даних на мові SQL. Синтаксис таких запитів докладно описаний в рекомендаціях консорціуму W3C. Результати SPARQL-запитів до RDF-графам можуть бути представлені в різних форматах, таких як XML, Json, а також CSV і TSV.

Мови запитів до формального поданням онтологій на RDF успішно застосовуються для вирішення завдань отримання фрагментів явних знань, що містяться в онтології. Так, подібним чином можуть бути вирішені задачі 1.2, 2.2-2.6, 3.2, 3.3, 4.2, 4.3 наведеної раніше класифікації завдань логічного висновку на УФО-моделях.

Більш складні інструменти, призначені, зокрема, для виведення нових знань з онтологій, в літературі [2, 6, 14, 47] також називаються вирішувач (reasoner). Ряд з які у класифікації задач може бути вирішено стандартними засобами, наданими найбільш поширеними інструментами логічного висновку на онтологіях.

Вирішувачі для здійснення логічного висновку на онтологіях можна умовно розділити на наступні типи:

- На основі правил (O-DEVICE).
- На основі резолюцій (пошуку протиріч): KAON2.
- На основі семантичного табло (semantic tableau, таблоалгоритма): FaCT ++, HermiT, Pellet.

Серед решателів, що базуються на правилах, в свою чергу, виділяють підвиди залежно від використовуваного алгоритму: forwardchaining («прямі ланцюжка») і / або backward-chaining («зворотні ланцюжка»). Вирішувачі першого підвиду ґрунтуються на вихідних (вже відомих) фактах, виводячи з них нові.

Подібні інструменти використовуються для виконання конкретних запитів до наявних даних або отримання особливого, відомого типу знань.

Вирішувачі на основі «зворотних ланцюжків» призначаються для перевірки заданого факту на відповідність вихідним даними. Більшість з існуючих інструментальних засобів - решателів призначене для роботи з онтологіями, представленими на мові OWL. Раніше зазначалося, що метод побудови онтологій на основі УФО-моделей дозволяє отримати код в нотації RDF / XML мови RDF, який може бути перетворений в OWL-код. Це, в свою чергу, забезпечує застосування широкого кола інструментальних засобів, зокрема, призначених для роботи з OWL решателів, для логічного виведення на онтологіях, побудованих на базі УФО-моделей.

Розглянемо деякі особливості інструментальних засобів, призначених для здійснення виведення знань на даних, представлених за допомогою форматів OWL. Оскільки мова OWL заснований на дескрипційних логіках, в літературі [31] наводиться наступне відповідність понять (таблиця 3.1).

Таблиця 3.1 - Відповідність понять в дескрипційних логіках і в мові OWL

Поняття в дескрипційних логіках	поняття в OWL
Концепт	Клас
Роль	Властивість
Індивід	Об'єкт
База знань	Онтологія

Як впливає з таблиці, у разі подання на мові OWL і здійснення операцій логічного висновку онтологію можна приймати за базу знань. У такій базі знань, згідно із загальними положеннями дескрипційних логік, виділяють набір термінологічних аксіом (TBox) і набір тверджень про індивідів (ABox). TBox містить дані (аксіоми) наступних видів:

- Аксіома вкладеності концептів - вираз виду, де і - довільні концепти.
- Аксіома еквівалентності концептів - вираз виду, де і - довільні концепти.
- Аналогічно, аксіомою вкладеності ролей називається вираз виду, де і - довільні ролі.

— аксіомою еквівалентності ролей - вираз виду, де i - довільні ролі. Тут символом позначається вкладеність (subsumption). Важливим аспектом є підтримка існуючими вирішувачі логічного висновку в області ABox, тобто виведення знань про об'єктах (індивідів). Відповідно до огляду, наведеним в [14], такий функціонал реалізований в інструментах FaCT ++, HermiT, RP, TrOWL (REL). Відповідно, при виборі існуючого решателя для прикладних задач логічного висновку на онтологіях необхідно враховувати ряд чинників:

- підтримувані алгоритми;
- наявність підтримки вирішення завдань логічного висновку в області ABox;
- підтримувані дескрипційні логіки;
- вартість рішення (безкоштовне або комерційне програмне забезпечення);
- використовувані і допустимі формати даних;
- мову програмування, на якому реалізований інструмент;
- можливості інтеграції в власне програмне забезпечення.

Крім цього, важливим критерієм вибору інструментів логічного висновку є можливість їх модифікації. Зокрема, що лежать в основі УФО-моделей, на базі яких будуються онтології в запропонованому методі, принципи і положення УФО-підходу, а також можливість застосування зазначеного методу для побудови онтологій в найрізноманітніших предметних областях, зумовлюють необхідність додаткової настройки таких інструментів. така настройка полягає в завданні додаткових правил (алгоритмів) логічного виведення, що враховують особливості предметної області і принципи УФОпідходу.

Тому в прикладних інформаційних системах потрібно використання або відомих існуючих, але модифікованих і доповнених інструментів логічного висновку, або власних, розроблених з урахуванням зазначених особливостей решателів. З використанням існуючих механізмів та інструментарію логічний висновок на онтологіях, побудованих шляхом спеціальних перетворень УФО-моделей предметних областей, дозволяє вирішити ряд істотних проблем, властивих евристичної обробці моделей предметних областей. Зокрема, такий механізм обробки і видалення знань з УФО-моделей має наступні переваги: 1.

1. Відсутність необхідності ручного формування вихідної бази знань. УФО-моделі предметної області завдяки простоті УФО-підходу та зручності програмних засобів, що підтримують його принципи при моделюванні, можуть бути побудовані як окремим (зовнішнім по відношенню до модельованої середовищі) фахівцем-аналітиком, так і безпосереднім експертом в даній області. В будь-якому випадку, одержувана в результаті УФО-аналізу модель предметної області вже містить в собі знання і уявлення експерта про дану предметну область. Оскільки логічний висновок виробляється на онтології, побудованої шляхом строго визначених перетворень формального подання вихідної УФО-моделі, така модель і є вихідною базою знань. В такому випадку зникає необхідність в окремих процесах формування первісної бази знань, що дозволяє заощадити час і зусилля експертів.

2. Гнучкість і масштабованість бази знань Актуалізація наявних даних про предметну область може бути виконана у вигляді внесення змін у вихідну УФО-модель. При цьому внесені зміни транслюються в створювану на основі УФО-моделі онтологію, по якій і здійснюється логічний висновок необхідних знань. Відповідно, подібна конструкція має простотою модифікації і широкими можливостями доповнення вихідних даних.

3. Застосування для вирішення складних завдань аналізу моделей предметних областей У разі аналізу складних предметних областей підсумкові УФОмоделі можуть бути досить громіздкими і важкими для сприйняття людиною. Найчастіше візуальний аналіз складної графоаналитического моделі не дозволяє швидко і коректно отримати необхідні відомості, містяться в такій моделі або виводяться на її основі. побудова на базі УФО-моделі онтології і здійснення логічного висновку на її основі дозволяють автоматизувати і значно спростити аналіз складних об'ємних моделей предметних областей.

3.5. Розробка алгоритму опису УФО-моделі на формалізованому мовою записи онтології .

З метою побудови опису онтології на мові RDF пропонується використовувати вихідне уявлення УФО-моделі у вигляді XML-файла. Вибір даного формату вихідного коду обумовлений двома причинами. Поперше, однією з найбільш поширених нотацій мови RDF є RDF / XML, за своїм синтаксису максимально наближена до мови розмітки XML. Для зручності подальшого використання створюваної онтології доцільно здійснювати перетворення записи вихідних даних саме в цю нотацію. Відповідно, спростити процес самої трансформації може використання вихідних даних, представлених в форматі XML або близьких до нього. По-друге, як правило, на практиці для здійснення моделювання предметної області з використанням коштів УФО-підходу застосовується спеціалізоване програмне засіб - UFO-toolkit. Даний програмний продукт зберігає створені з його допомогою УФО-моделі саме в файлах формату XML. Таким чином, в якості вихідного файлу приймається XML-файл УФО-моделі, отриманий за допомогою програмного засобу UFO-toolkit. Для здійснення його перетворення в конструкції мови RDF, описують онтологію предметної області, в загальному випадку необхідно виконати наступні кроки.

1. Виділити в вихідному файлі концепти предметної області в Відповідно з наступними типами:

- УФО-елементи (вузли);
- функції УФО-елементів;
- класи зв'язків;
- зв'язку, присутні в моделі;
- порти вузлів.

2. Визначити всі значущі атрибути (характеристики) виявлених концептів (ідентифікатори, найменування, опису, специфічні властивості).

3. Встановити взаємозв'язку між виявленими концептами згідно класифікації витягають із УФО-моделі фактів.

4. Сформувати результуючий файл в наступній послідовності:

— Додати загальні заголовки для коректної інтерпретації мови RDF.

— Сформувати блок інформації про класифікацію зв'язків.

— Включити в файл фрагмент, що описує ієрархію загальних предикатів (понять), що стосуються характеристик портів вузла.

— Додати конструкції, які характеризують кожен УФО-вузол, його атрибути, факти групи Ікласифікації, де вузол виступає в якості об'єкта триплета, а також факти групи Ікласифікації, що стосуються наявності у вузла портів.

— Додати конструкції, що характеризують кожен УФО-функцію, її атрибути і факти групи І (inv) класифікації, де функція виступає в Як об'єкт триплета.

— Додати конструкції, які характеризують кожен УФО-об'єкт, його атрибути (тобто факти групи IV класифікації) і факти групи І (inv) класифікації, де УФО-об'єкт виступає в якості об'єкта триплета.

— Сформувати блок інформації про існуючі в моделі зв'язках між УФО-вузлами (факти групи V і V (inv) класифікації).

Розглядаючи перелічені дії з точки зору потенційно автоматизуючого процесу виробництва хліба, алгоритм перетворення XML-представлення УФО-моделі в формальний опис онтології на мові RDF з використанням нотації RDF / XML можна представити у вигляді такої блок-схеми (рисунок 4.1).

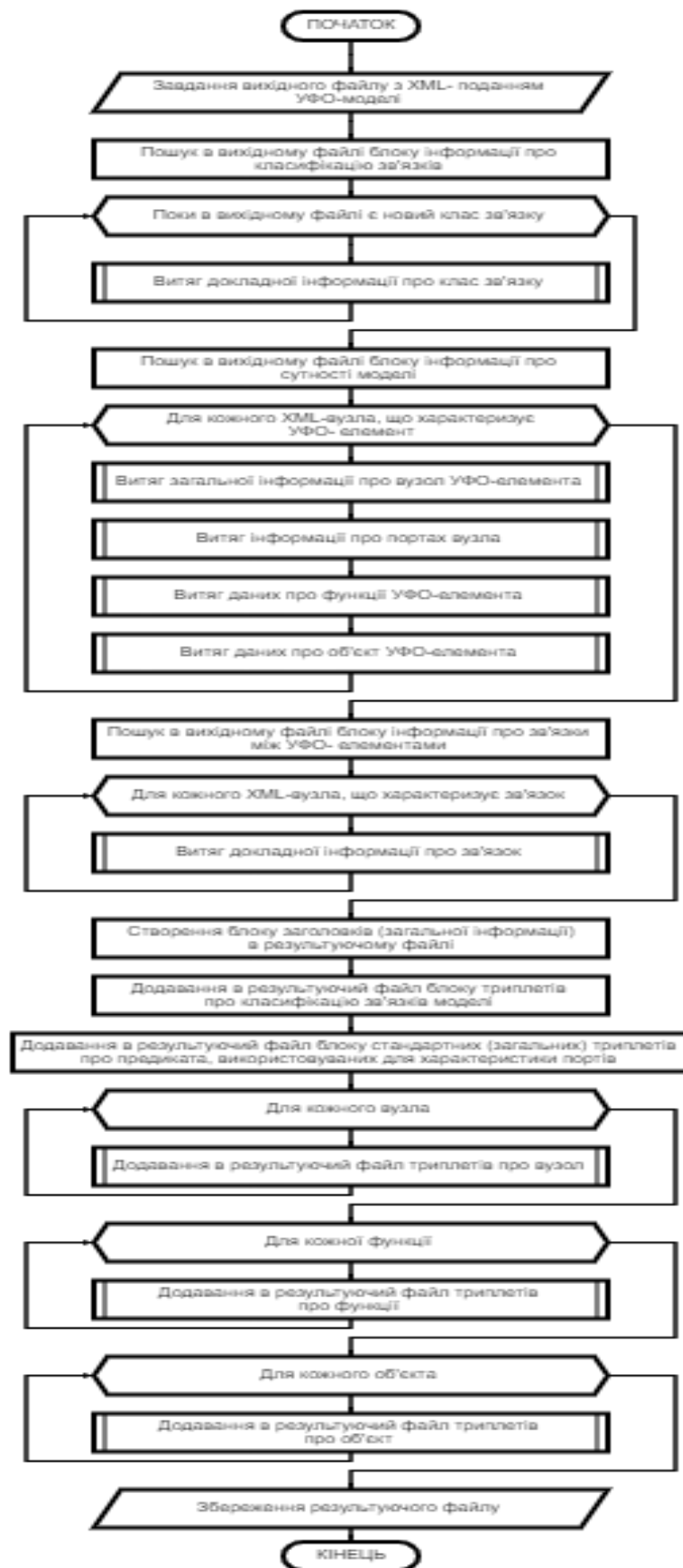


Рисунок 3.3 - Блок-схема алгоритму перетворення XML-представлення UFO-моделі в конструкції мови RDF

Розглянемо підпроцеси перетворення XML-вистави УФОмоделі в код на мові RDF детальніше. Як видно з малюнка 4.1, одним з перше підпроцесов, що зустрічаються в запропонованому раніше алгоритмі перетворення, є витяг з вихідного файлу докладних відомостей про класифікацію зв'язків, використовуваної в аналізованій графоаналитического моделі.

Як правило, відомості про класифікацію зв'язків в XML-представлення УФО-моделі, сформованому за допомогою програмного засобу UFOToolkit, зберігаються в XML-вузлі з ім'ям PortHierarchy. першим і єдиним дочірнім (вкладеним) XML-вузлом є вузол, характеризує зв'язок як таку. Іменем даного класу є «Зв'язок» або «Link», нотацією - «L», а типом порту - «Link». В даний вузол в вихідному файлі вложені XML-вузли, які відповідають усім дочірнім класам зв'язків. Це як базові (стандартні, присутні в кожній моделі за замовчуванням) класи зв'язків, так і спеціальні, прикладні, введені в класифікацію зв'язків спеціально для даної моделі.

Розглянемо структуру типічного XML-вузла, що характеризує клас зв'язку:

```
<ItemName = "Матеріальна" Notation = "M" ID = "8" PortType = "Material">
```

Подібні вузли мають ім'я «Item». Як атрибути виступають ім'я класу зв'язку (Name), його нотація (Notation), унікальний в межах моделі ідентифікатор класу зв'язку (ID) і тип порту (PortType).

Таким чином, для вилучення відомостей про класи зв'язків з подібних конструкцій вихідного XML-файла потрібно наступний алгоритм (рисунок 3.4).

Наступним підпроцесом, яке трапляється в загальному алгоритмі перетворення, є витяг з вихідного файлу інформації про вузлі як складової УФО-елемента, а також його портах. потім в алгоритмі слід підпроцес вилучення відомостей про функції, балансує вузол і об'єкти, що займає його.

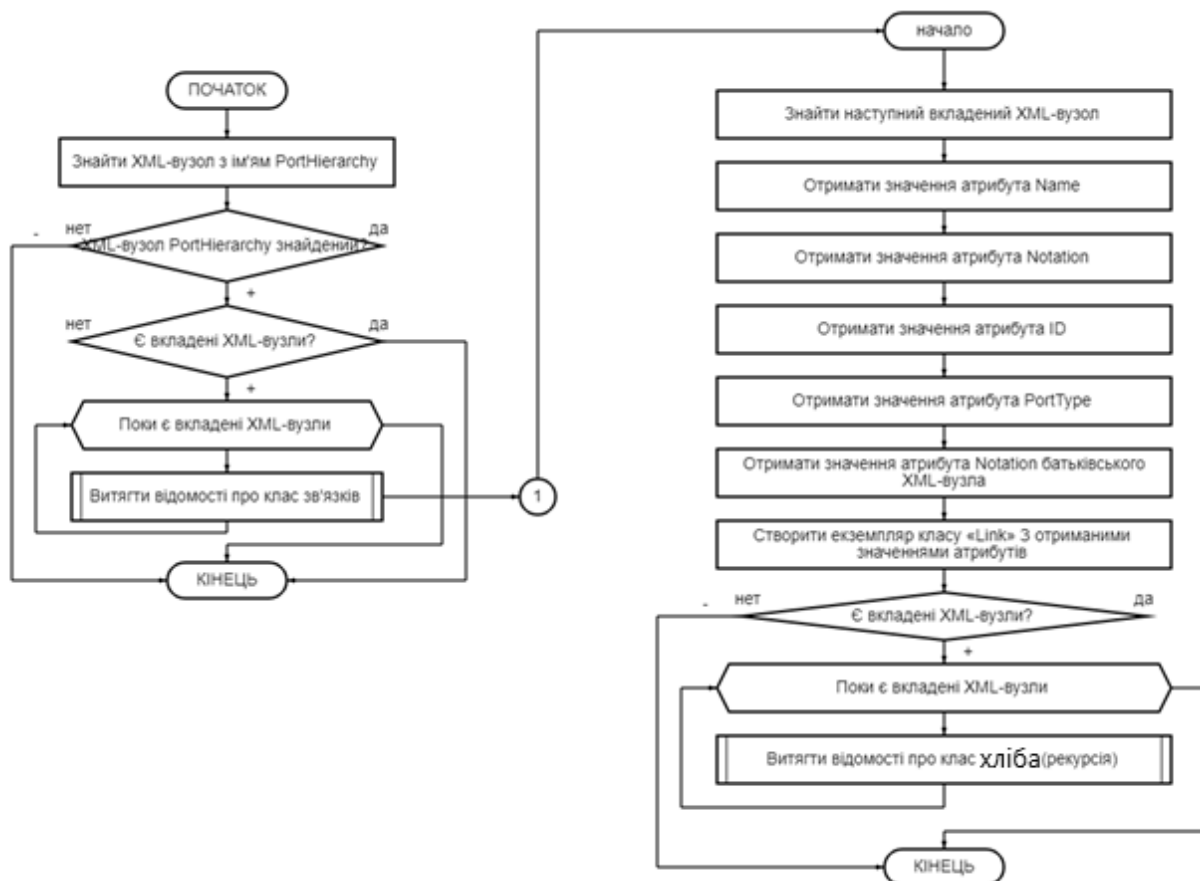


Рисунок 3.4. Блок-схема алгоритму вилучення відомостей про класифікацію хліба з вихідного XML-файла УФО-моделі.

Загальні відомості про УФО-елементах зберігаються в XML-представлення УФО-моделей в XML-вузлах, дочірніх по відношенню до вузла Diagram, який, в свою чергу, є вкладеним по відношенню до XML-вузлу DiagramRepository. XML-вузли, що характеризують безпосередньо УФО-елементи, мають ім'я Node, в той час як XML-вузли, що відповідають зв'язкам між УФО-елементами, мають ім'я Link. відповідно, витяг відомостей про елементи аналізованої моделі в більш формалізованому вигляді можна представити таким чином (рисунок 3.5).

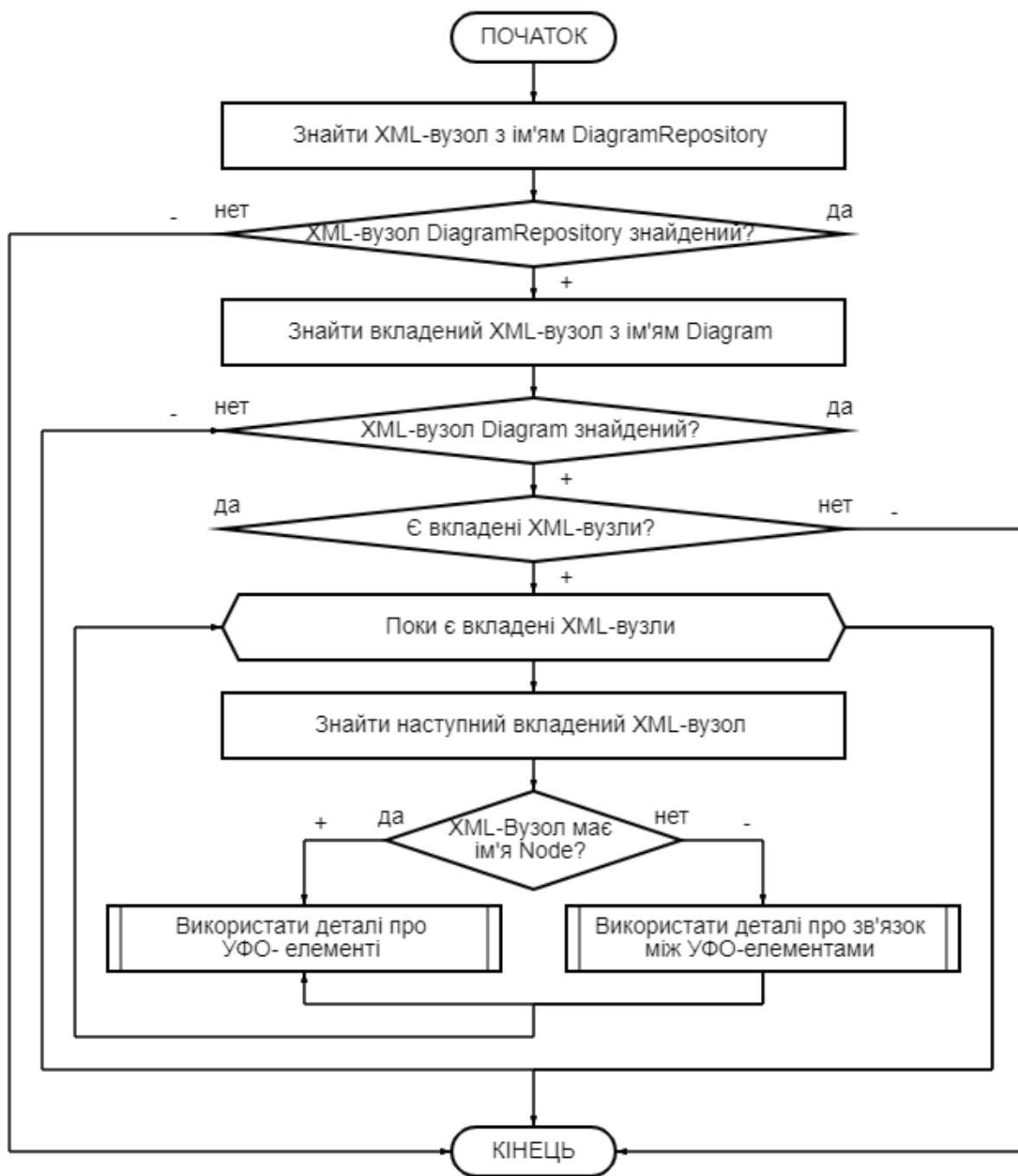


Рисунок 3.5 - Блок-схема алгоритму вилучення відомостей про UFO-елементах і зв'язках між ними з XML-вистави UFO-моделі.

Витяг інформації про UFO-елементі - досить складна процедура, яку можна виділити як окремий підпроцесса. Так, на даному етапі аналізується фрагмент вихідного файлу UFO-моделі, а саме XML-вузол з ім'ям Node, а також його дочірні XML-вузли. Розглянемо фрагмент коду, що відображає типову структуру зазначених XML-вузлів:

<NodeID = "22" StyleID = "3" Left = "550" Top = "120" Width = "200" Height = "100">

<Node Notation = "Dev" Description = "Developers department" ID = "23" CurrentFunction = "0">

<InputPorts ID = "24">

<Port ID = "26" StyleID = "3" PortID = "18" Position = "3.5" />

</ InputPorts>

<OutputPorts ID = "25">

<Port ID = "27" StyleID = "3" PortID = "19" Position = "2.64"

CustomPosition = "1" />

</ OutputPorts>

<Function Name = "Development" ID = "53" DiagramWidth = "1040" DiagramHeight = "790" ShowGrid = "0" SnapToGrid = "1" GridStyle = "Dots" HorzGridStep = "10" VertGridStep = "10" Zoom = "1"

NodeContents = "CurrentFunctions" CurrentObject = "0">

<Object Name = "Department of development" ID = "54" />

</ Function> </ Node> </ Node>

Дана конструкція об'єднує в рамках одного атрибута верхнього рівня (Node) всі відомості про УФО-елементі. В атрибутах елемента Node другого рівня зберігаються основні відомості, такі як нотація (Notation), ідентифікатор (ID), опис (Description), відомості про наявність в діаграмі функції, що балансує поточний вузол. Останній параметр зберігається в вигляді атрибуту CurrentFunction і сам по собі не містить інформації про функції. Він лише вказує на наявність (значення 0) або відсутність (Значення -1) у вузла функції. В рамках елементів InputPorts і OutputPorts зберігаються відомості про портах даного вузла. Кожен порт є вкладеним елементів і має ім'я Port. В атрибутах даного елемента зберігаються відомості про унікальний в межах моделі ідентифікатор порту (атрибут ID), його класі (атрибут PortID), стилі (атрибут StyleID). У разі якщо на аналізованій діаграмі для УФО-елемента явно задана функція, відомості про неї зберігаються в елементі Function, вкладеному в Node. В атрибутах даного елемента можна знайти ім'я функції (Name), її ідентифікатор (ID), а також відомості про наявність чи відсутність даних про об'єкт, що реалізує поточну функцію. За відомості про наявність / відсутність на діаграмі об'єкта для даної функції відповідає атрибут CurrentObject елемента Function. Якщо він має значення 0, об'єкт заданий, якщо -1, об'єкт, який реалізує функцію, не вказано. Докладні відомості про об'єкт зберігаються

у вкладеному в елемент Function елементі Object. Як атрибути для об'єкта можуть бути задані ім'я (Name) і ідентифікатор (ID).

Таким чином, для опису УФО-елемента, знайденого в вихідному файлі (тобто екземпляра класу ufo: UFO в RDF-поданні створюваної онтології), необхідно визначити його основні властивості: ідентифікатор (ID), нотацію, опис, функцію (при наявності). Функція буде в даному випадку бути самостійною сутністю (екземпляром класу ufo: UFOFunction в RDF-коді створюваної онтології). Для її подання в онтології потрібно визначення: ідентифікатора (ID), найменування, об'єкта (при наявності). Об'єкт тут - екземпляр класу ufo: UFOObject в RDF-коді розробляється онтології, для опису якого потрібно вказівка: ідентифікатора (ID), найменування, опису. Таким чином, процес вилучення відомостей про УФО-елементі можна представити у вигляді блок-схеми, показаної на малюнку 3.6.

На окрему увагу заслуговує процес вилучення з вихідного XML-файла даних про зв'язки між УФО-елементами. У XMLпредставленні УФО-моделі, сформованому редактором UFOToolkit, дані про кожну зв'язку зберігаються у вигляді XML-вузла наступного виду:

```
<Link ID = "46" StyleID = "3" FromID = "36" ToID = "26" />
```

З подібної конструкції можна витягти унікальний ідентифікатор зв'язку в межах моделі (атрибут IDXML-вузла), а також ідентифікатори вихідного (атрибут FromID) і входить (атрибут ToID) портів присутніх в моделі вузлів, від і до якого відповідно направлена зв'язок. За цим ідентифікаторів в подальшому при необхідності можна отримати дані про пов'язаних досліджуваної зв'язком УФО-елементах і їх складових (вузлах, функціях, об'єктах).

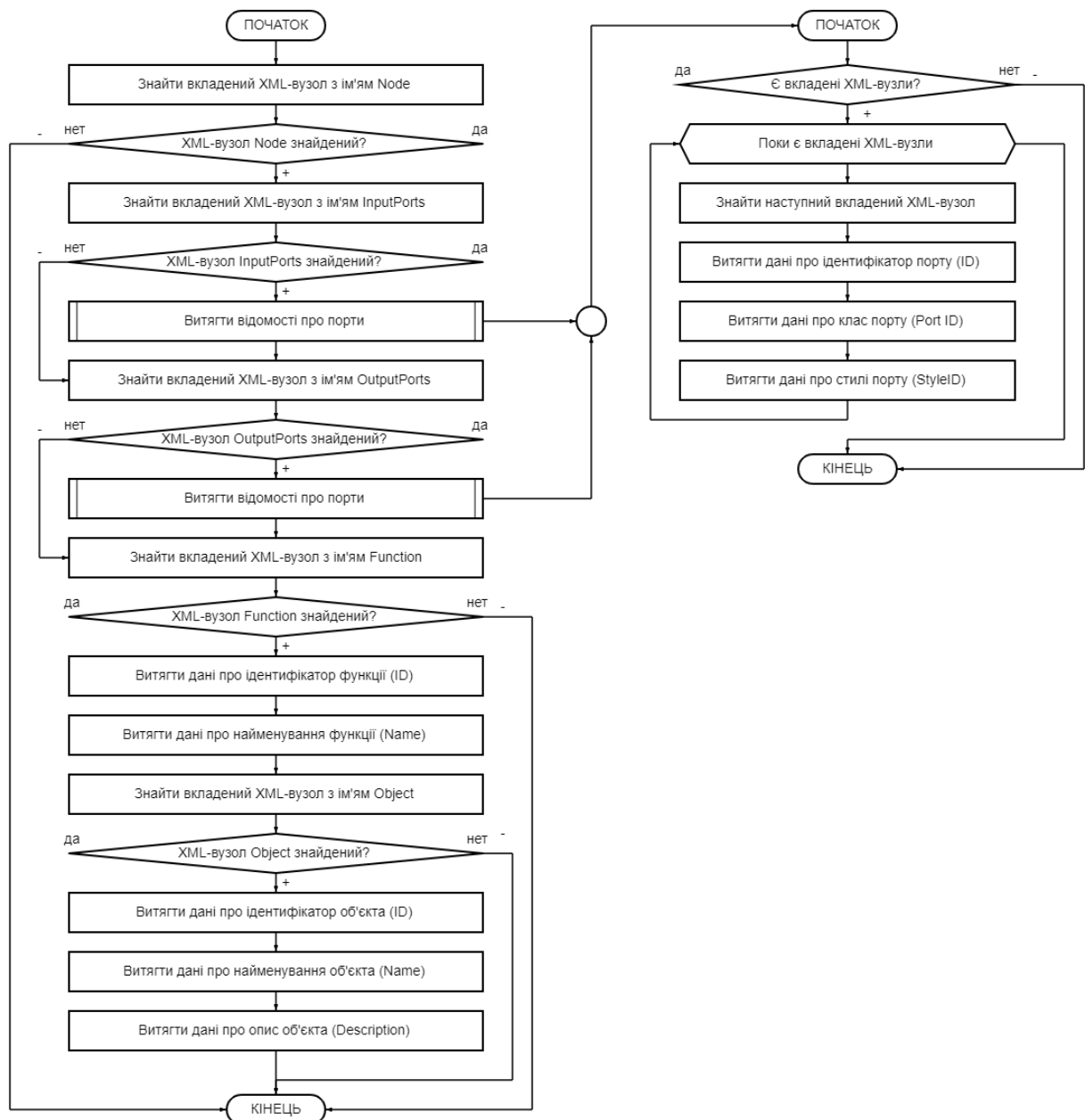


Рисунок 3.6 - Блок-схема вилучення з XML-вистави УФО-моделі докладних відомостей про окремий УФО-елементі.

Після виконання описаних вище операцій етап вилучення даних з вихідного файлу аналізованої УФО-моделі можна вважати завершеним. Наступним етапом перетворення є побудова RDF-триплетів і запис їх в результуючий файл. спочатку в результуючий файл додається загальна інформація (заголовки), які забезпечують коректне розпізнавання мови RDF і його нотації RDF / XML, за допомогою яких представлені дані. Зокрема, в цих заголовках визначаються використовувані простору імен, такі як `rdf`, `rdfs`, `owl` і введене нами `ufo`.

Далі в файл записуються RDF-триплети, що характеризують використовувану в даній УФО-моделі класифікацію зв'язків. В загальному випадку кроки даного процесу замкнуті в цикл, який виконується для кожного класу зв'язку, знайденого в даних, витягнутих раніше з вихідного XML-файла.

Формування RDF-триплетів для кожного класу зв'язку складається з наступних кроків (приклади наведені в використовуваній на Протягом усього процесу перетворення нотації RDF / XML):

1. Додати в результуючий файл XML-вузол із загальною інформацією про клас зв'язку (конструкція виду

`<ufo: LinkClass rdf: about = "http://www.ufo-toolkit.ru/#Left">`)

2. Додати до створеного на першому кроці XML-вузлу дочірній (Вкладений) XML-вузол, що відображає факт еквівалентності найменування і нотації класу зв'язку (конструкція виду

`<ufo: equivalent rdf: resource = "http://www.ufo-toolkit.ru/#Left" />`)

3. Додати до створеного на першому кроці XML-вузлу дочірній (Вкладений) XML-вузол, що відображає факт наявності ID у даного класу зв'язку (конструкція виду `<ufo: hasID> 11 </ ufo: hasID>`).

4. Додати до створеного на першому кроці XML-вузлу дочірній (Вкладений) XML-вузол, що відображає відомості про тип порту даного класу зв'язку (конструкція виду `<ufo: hasPortType rdf: resource = "http://www.ufotoolkit.ru/#User" />`)

5. Додати до створеного на першому кроці XML-вузлу дочірній (Вкладений) XML-вузол, що відображає факт того, що даний клас зв'язку є підкласом іншого класу (при наявності у вихідному файлі відомостей про батьківському класі зв'язку). Дана конструкція має вигляд:

`<Rdfs: subClassOf rdf: resource = "http://www.ufo-toolkit.ru/#S" />`.

На наступному кроці процесу перетворення в результуючий файл додається інформація про ставлення предикатів, використовуваних для вираження фактів наявності портів у вузлів УФО-елементів. даний блок є стандартним, тобто не змінюється в залежності від аналізованої УФО-моделі. Код такого блоку в нотації RDF / XML наведено нижче:

`<Rdf: Description rdf: about = "http://www.ufo-toolkit.ru/#hasInputPort"> <Rdfs: subPropertyOf rdf: resource = "http://www.ufo-toolkit.ru/#hasPort" /> </ Rdf: Description>`

`<Rdf: Description rdf: about = "http://www.ufo-toolkit.ru/#hasOutputPort"> <Rdfs: subPropertyOf rdf: resource = "http://www.ufo-toolkit.ru/#hasPort" /> </ Rdf: Description>`

`<Rdf: Description rdf: about = "http://www.ufo-toolkit.ru/#hasInputPortClass"> <Rdfs: subPropertyOf rdf: resource = "http://www.ufo-toolkit.ru/#hasInputPort" /> </ Rdf: Description>`

`<Rdf: Description rdf: about = "http://www.ufo-toolkit.ru/#hasOutputPortClass"> <Rdfs: subPropertyOf rdf: resource = "http://www.ufo-toolkit.ru/#hasOutputPort" />`

`</ Rdf: Description>`

Далі йде етап побудови RDF-коду, що характеризує вузли УФО-елементів. Підпроцес побудови RDF-триплетів про кожному вузлі має на увазі наступні кроки:

1. Додати в результуючий файл XML-вузол, що характеризує поточний УФО-елемент в цілому.

2. Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Вузол ... балансується функцією ...».

3. Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Вузол ... зайнятий об'єктом ... ».

4. Якщо у поточного вузла є хоча б один порт:

— Для кожного з портів поточного вузла додати в результуючий файл дочірній по відношенню до доданому в п. 1 XML-вузол, характеризує факт виду «Вузол ... має вхідний / вихідний порт класу ... ».

5. Для кожної зв'язку вузла (мається на увазі присутня в моделі зв'язок між двома УФО-елементами) виконати обробку: Якщо зв'язок є вхідною:

— Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Вузол має вхідний зв'язок класу ...».

— Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Вузол <Зворотний предикат зв'язку> <Нотація вузла, від якої спрямована зв'язок> ». Якщо зв'язок є вихідної:

— Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Вузол має вихідний зв'язок класу...»

— Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Вузол <прямий предикат зв'язку> <Нотація вузла, до якого спрямована зв'язок> ». На наступному кроці алгоритму перетворення УФО-моделі в формальне подання на мові RDF передбачається формування блоку коду, що відповідає за характеристику всіх заданих в моделі функцій УФО-елементів. Даний етап складається з наступних дій:

1. Додати в результуючий файл XML-вузол, що характеризує функцію поточного елемента в цілому.

2. Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Функція ... реалізується за допомогою об'єкту ... ».

3. Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Функція ... балансує вузол ... ».

4. Для кожної зв'язку вузла, балансируемого поточної функцією: — Якщо зв'язок є вхідною, додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Функція перетворює вхід ...».

— Якщо зв'язок є вихідної, додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Функція видає вихід ...».

— Перераховані дії виконуються циклічно для кожної з функцій УФО-елементів, знайдених на етапі отримання даних з вихідного файлу. Після блоку RDF-коду, що характеризує функції, в результуючий файл додаються дані про

всі об'єкти УФО-елементів, присутніх в моделі. Для кожного знайденого об'єкта (Складової УФО-елемента) виконуються наступні дії:

1. Додати в результуючий файл XML-вузол, що характеризує об'єкт поточної функції в цілому.

2. Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Об'єкт ... займає вузол ... ».

3. Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Об'єкт ... реалізує функцію ... ».

4. Для кожного з властивостей об'єкта:
— Додати в результуючий файл дочірній по відношенню до доданому в п.1 XML-вузол, що характеризує факт виду «Об'єкт ... має властивість (має ознаку) ... ».

На закінчення після виконання зазначених раніше дій в результуючий файл додаються закривають XML-теги, необхідні для коректної обробки файлу. Отриманий XML-файл буде аналогом вихідної УФО-моделі, представленої за допомогою нотації RDF / XML мови XML.

3.6. Рішення задач логічного висновку на візуальних графоаналитических УФО-моделях за рахунок їх інтеграції з формальними засобами записи онтології.

Одне з основних застосувань онтологій - використання їх в Як машиночитаемого способу представлення знань. відповідно, важливим з практичної точки зору є рішення задач логічного висновку на знаннях, що містяться в онтології. Розглянемо способи вирішення основних видів завдань логічного висновку на онтологіях, побудованих на базі візуальних графоаналитических УФО-моделях предметних областей за допомогою запропонованого раніше методу.

Як зазначалося в розділі 3, завдання логічного висновку на онтологіях, побудованих на основі УФО-моделей, можна умовно згрупувати в Відповідно до класифікації. Ряд із запропонованих груп завдань пропонується вирішувати за допомогою спеціалізованої мови запитів SPARQL (SPARQL Protocol and RDF Query Language) до формального поданням онтології.

Так, наприклад, типовим випадком, що вимагає застосування мови SPARQL, є пошук інформації про об'єкт, що займає заданий вузол. Таке завдання можна представити таким чином: в коді онтології на мові RDF потрібно знайти триплети виду: «Об'єкт займає вузол », де в якості об'єкті виступає конкретний вузол вихідної УФОмоделі. Мова SPARQL для пошуку необхідних триплетів дозволяє сформулювати запит, синтаксис якого близький до синтаксису мови SQL (Structured query language) - мови управління даними в реляційних базах даних. Стосовно до описаної задачі пошуку об'єкта, що займає конкретний вузол вихідної УФО-моделі, будується SPARQL-запит виду:

PREFIX ufo: < <http://www.ufo-toolkit.ru/>>

SELECT ?o

WHERE

{ ?o ufo:represents ?u. }

В результаті виконання такого запиту формується список суб'єктів предикатів, що задовольняють заданій в тілі умови запиту «Шаблоном». Отримані концепти і є шуканими об'єктами, займають заданий вузол УФО-елемента.

Більш складні завдання логічного висновку вимагають застосування гнучких інструментів, що дозволяють використовувати складні правила (алгоритми), в тому числі спеціалізовані, що задаються користувачем або включені в онтологію. Розглянемо ряд практичних завдань, які можуть бути вирішені за допомогою мови правил виведення SWRL, який застосовується до формальному поданням онтології на мові OWL або синтаксично сумісних нотацій мови RDF.

— Включення в онтологію фактів виду «Функція перетворює Вхід...».

В даному випадку для отримання факту потрібно його логічний висновок на основі наявних даних. Зокрема, використовуються відомості про балансування функцією певного вузла, а також про наявність у вузла конкретної вхідного зв'язку. Оскільки перший факт може бути включений в онтологію в одному з двох видів («Функція балансує вузол» або «Вузол балансується функцією »), необхідно використовувати два SWRL-правила, представлених нижче: $balances(?f, ?u) \wedge hasInputRelation(?u, ?Li) \Rightarrow translateInput(?f, ?Li)$, $isBalancedBy(?u, ?f) \wedge hasInputRelation(?u, ?Li) \Rightarrow translateInput(?f, ?Li)$;

— Включення в онтологію фактів виду «Функція видає вихід ...».

Дане завдання аналогічна описаній раніше, з тією відмінністю, що під другій частині правила використовуються факти наявності у вузла вихідного зв'язку. SWRL-правила для виведення факту «Функція видає вихід ...» показані нижче.

$balances(?f, ?u) \wedge hasOutputRelation(?u, ?Lj) \Rightarrow giveOutput(?f, ?Lj)$,
 $isBalancedBy(?u, ?f) \wedge hasOutputRelation(?u, ?Lj) \Rightarrow giveOutput(?f, ?Lj)$;

— Включення в онтологію фактів виду «Об'єкт займає вузол». У XML-представлення УФО-моделей, формованому найбільш поширеним програмним продуктом для моделювання предметних областей з використанням засобів УФО-підходу «UFO Toolkit», наявність об'єкта як складової УФО-елемента описується в вигляді «дочірньої» по відношенню до Функції характеристикою (вкладеним XML-вузлом). Тому при автоматизації перетворення УФО-моделі в конструкції мови опису онтологій факт виду «Об'єкт займає Вузол» може виводитися з інших, раніше сформованих триплетів онтології, що характеризують Вузол і Функцію. Так, для виведення факту про ставлення Об'єкту і Вузла потрібно мати в онтології факт балансування Функцією цього Вузла і факт реалізації Об'єктом даної Функції. З урахуванням можливих варіацій записи зазначених фактів отримаємо чотири SWRL-правила для виведення описаного факту:

$realizes (? o, ? f) \wedge balances (? f, ? u) \Rightarrow represents (? o, ? u), realizes (? o, ? f) \wedge isBalancedBy (? u, ? f) \Rightarrow represents (? o, ? u), isRealizedBy (? f, ? o) \wedge balances (? f, ? u) \Rightarrow represents (? o, ? u), isRealizedBy (? f, ? o) \wedge isBalancedBy (? u, ? f) \Rightarrow represents (? o, ? u);$

— Облік декількох рівнів декомпозиції (УФО-елемент є частиною іншого елемента на кілька рівнів вище).

У разі, коли відомості про те, що УФО-елемент є частиною декомпозиції іншого УФО-елемента, що знаходиться більш ніж на одному 119 рівні вище, необхідно включити безпосередньо до складу онтології, застосовується SWRL-правило виду: $isPartOf(? u_{i\ n-2}, ? U_{j\ n-1}) \wedge isPartOf(? U_{j\ n-1}, ? U_n) \Rightarrow isPartOf(? U_{i\ n-2}, ? U_n);$

— Визначення вузлів, які мають подібні порти з заданим. Таке завдання вирішується за допомогою циклічної обробки кожного факту наявності вхідного або вихідного порту у першого порівнюваного вузла. В ході такої перевірки використовуються правила (для вхідних і вихідних портів відповідно):

$hasInputPort (? u1, ? Li) \wedge hasInputPort (? u2, ? Li) \Rightarrow hasSamePort (? u1, ? u2),$
 $hasOutputPort (? u1, ? Li) \wedge hasOutputPort (? u2, ? Li) \Rightarrow hasSamePort (? u1, ? u2),$

Для пошуку вузлів, один з яких має вхідний, а інший - вихідний порт одного типу, застосовується одна з таких SWRL-конструкцій: $hasInputPort (? u1, ? Li) \wedge hasOutputPort (? u2, ? Li) \Rightarrow hasSamePort (? u1, ? u2), hasOutputPort (? u1, ? Li) \wedge hasInputPort (? u2, ? Li) \Rightarrow hasSamePort (? u1, ? u2).$

SWRL-правила забезпечують можливість гнучкого налаштування логічного висновку нових відомостей про предметну область на основі вже наявних фактів (триплетів). Наведений перелік практичних завдань не є вичерпним і може бути розширений для кожного конкретного випадку. Отже, експерти в процесі розвитку онтології предметної області можуть доповнювати її новими фактами, включаючи їх безпосередньо в саме опис онтології на формальній мові, що забезпечує легкість масштабування і можливість оперативного доповнення онтологій необхідними «стандартними» знаннями.

3.7. Розробка дослідницького прототипу програмного пакету побудови онтологій хлібопекарського виробництва.

Для ефективного розв'язання проблем представлення знань та реалізації механізму логічного виведення в інтелектуальних системах промислової автоматизації набули широкого поширення онтології [112, 113].

Задано:

а) категоріальний апарат онтології, у вигляді кортежу глобальних об'єктів:

$$KA = \langle t, a, s, e, p, q \rangle, \text{ де} \quad (3.1)$$

t – знак «поняття»;

a – знак «дія»;

s – знак «стан»;

e – знак «подія»;

p – знак «властивість»;

q – знак «величини»;

б) конструкція знаку «поняття»:

$T = \langle D, P, A, C, S, T, M \rangle$, де (3.2)

D – множина дефініцій поняття t ;

P – множина дій поняття t ;

C – множина термінів t ;

S – множина стану поняття t ;

T – множина поняття, що має відношення з терміном t ;

M – множина метазнаків поняття t ;

в) текст термінологічного словника TD , релевантної моделі предметної області, проілюстрованої у вигляді двійки [114, 115]:

$Msa = \langle Tk, Kk \rangle$, де (3.3)

Tk – множина понять ;

Kk – кортеж, що описує множину конструктів та їх взаємозв'язок.

Завданням досліджень є розробка методу, призначеного для розпізнавання у хлібопекарському виробництві глобальних відносин та об'єктів, а також елементів конструкцій їх знаків за наступних обмежень:

1) ступінь покриття конструктів у породжених правилах вихідних конструктів повинна прагнути до одиниці;

2) результати розв'язування задач побудови онтологій із застосуванням створених систем хлібопекарської продукції і результати роботи експертів повинні бути близькі, а в ідеальному випадку – однакові [116].

Щоб розв'язати поставлену задачу й успішно використати експертні знання, потрібно сформулювати модель предметної галузі хлібопекарського виробництва. Застосовуючи методології еволюційних обчислень, на основі знань про модель предметної галузі генеруються системи продукційних правил як декларативна ілюстрація методу. Так, якщо експерт міркує природною мовою, то продукція матиме природно-мовне уявлення і її потрібно перетворити в формальне подання [117].

Передбачається, що розуміння онтології може бути використано для управління трьома ключовими процесами [118], а саме: а) комунікація; 2) інтеграція; 3) логічний висновок. Ці процеси ґрунтуються один на одному й відображають збільшені рівні формальності в сенсі використаних онтологій залежно від зростання складності необхідних знань. У зв'язку з цим збільшуються вимоги до рівня формальності онтологій, починаючи з простих словників до формальних логічних структур, які дають змогу виконувати складні логічні висновки.

Структура онтології має відображати структуру предметної галузі хлібопекарського виробництва. Технологія виробництва хліба – галузь знань, що включає широкий спектр питань: технологічні процеси, обладнання, техніко-економічні аспекти виробництва. Хлібопекарське виробництво здійснюється в технологічному середовищі, вивчення якого можливе лише за

умови дотримання принципів системного підходу. У його системі виділяють підсистеми: функціональну, елементну та організаційну, які представляють собою сторони єдиного цілого. Функціональна сторона виробничої системи (заводу) і його підсистем (ділянок, ліній) визначається їх технологічним призначенням. Елементна сторона визначається складом цих ділянок та їх обладнанням для забезпечення технологічного призначення. Організаційна сторона встановлює структуру системи, зрозумілу мету для кожної її складової частини й реалізує виконання мети відповідно до функціонального призначення.

Онтологія – це логічний підхід до опису структури понять предметної галузі, який дає змогу детально, осмислено, в науковому й зрозумілому для дослідника та експерта вигляді, сформувати ієрархію ознак, що впливають на якість. Можна створити ієрархічне дерево ознак, яке може бути імпортовано в розроблювану систему підтримки прийняття рішень у галузі оцінки якості, що репрезентує собою web-додаток для проведення експертизи під назвою «Кваліметрична експертиза». Онтологія як модель представлення прецеденту описує його як набір екземплярів класів і стосунків між ними [117]. Як база знань у предметній галузі вона описує знання як правила опису логіки (ОЛ) з використанням процедур виведення. Як БД прецедентів онтологія зберігає їх як частину знань предметної галузі (рис. 3.7).

Практично створення онтології включає:

- 1) визначення класів в онтології;
- 2) розташування класів у таксономічну ієрархію (підклас – надклас);
- 3) заповнення значень слотів екземплярів.

Під час реалізації онтології формуються дві незалежні одна від одної підсистеми: безпосередньо сама база знань (сутності й відношення між ними) і комплекс програм, призначений для обробки знань (функції інтерпретації, задані на сутності та відношеннях онтології), утворюючи класичну інтелектуальну систему [110, 113].

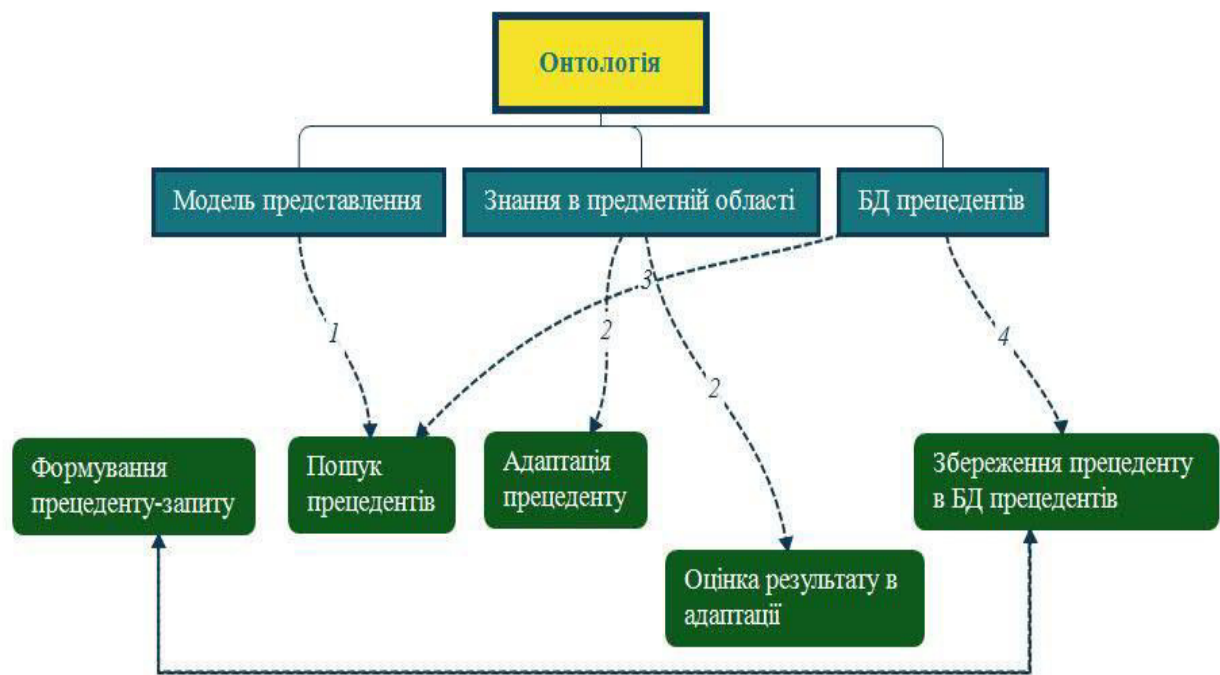


Рис. 3.7. Схема інтеграції процесу міркування із прецедентів і онтології: 1 – опис моделі надання прецедентів; 2 – формалізовані правила в предметній галузі; 3 – прецедент із БД прецедентів; 4 – прецедент нового вирішеного завдання

База знань створюється з урахуванням її майбутнього використання. Саме тут міститься «ядро» всієї системи, яке побудоване на основі моделей знань із використанням онтологій предметної галузі. Підсистема обробки знань повинна будуватися на принципах формальної логіки. В основі побудови онтології повинні лежати теоретичні знання, що ілюструють систему пов'язаних між собою понять і висловлювань в аналізованій предметній галузі, які зв'язуються відповідно до концептів. Форми, в яких фіксуються знання про об'єкт управління в результаті інтелектуальної діяльності, зокрема такі, як «поняття», «висловлювання», «міркування», «логічне виведення», з одного боку, вже містяться в деякому формалізованому вигляді в базі знань, з іншого – можуть бути отримані алгоритмічно на основі цієї бази [120].

Онтологія для технологічного комплексу хлібопекарського виробництва створювалася в програмному засобі для розробки онтологій – Protégé, який включає редактор онтологій. На основі сформованих онтологій у Protege згенеровані форми здобуття знань для введення екземплярів класів і підкласів. Protege заснований на фреймовій моделі представлення знання й забезпечений рядом інструментів, що дає змогу адаптувати його для редагування моделей у різних форматах (стандартний текстовий, бази даних, UML, мов XML, XOL, SHOE, RDF і RDFS, Dam1+oil, OWL) [121].

На рис. 3.8 зображено вигляд головного вікна програмного засобу, в якому формувалася онтологія, наведено ієрархію класів (Class Hierarchy), слоти вибраного класу (Template Slots) та обмеження, накладені на слоти (Other Facets).

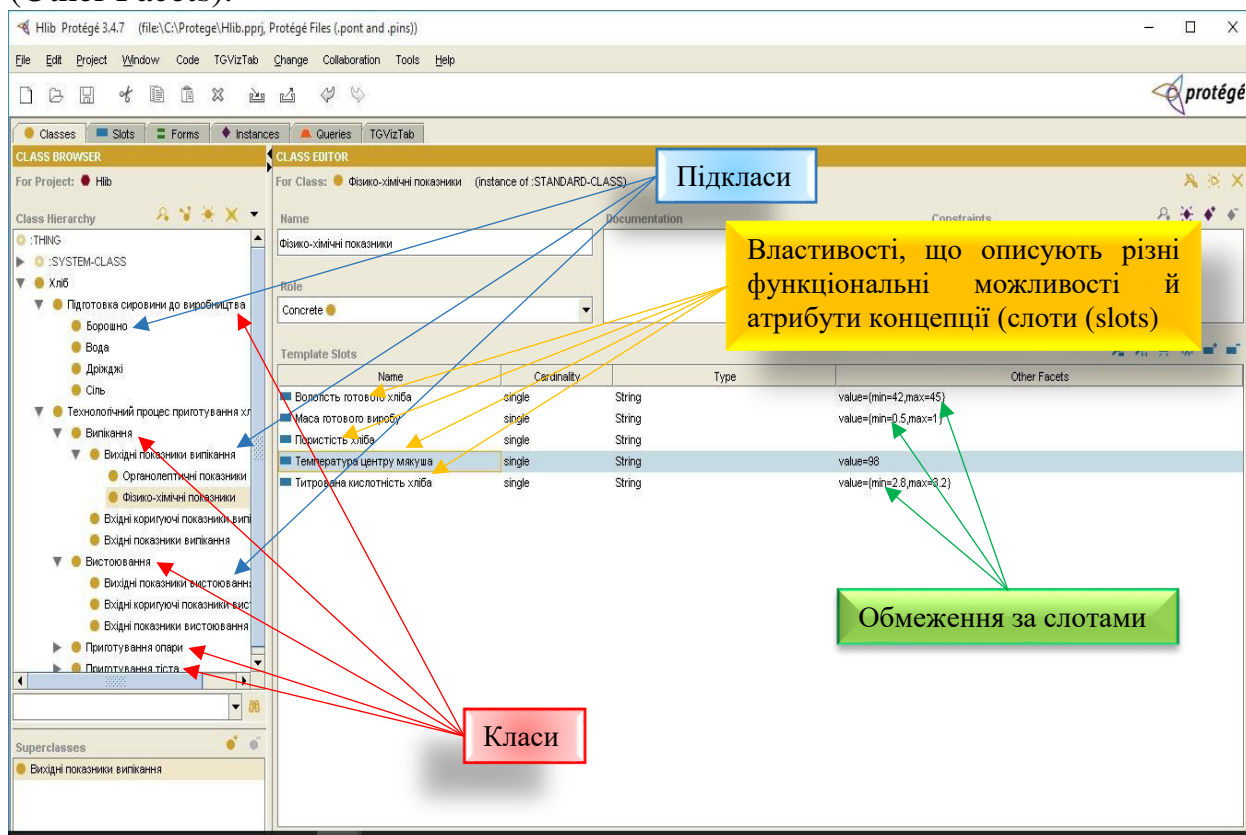


Рис. 3.8. Головне вікно програмного засобу Protégé

За допомогою інтеграції програмного засобу OntoGraf в Protégé та використовуючи вкладку OntoGraf можливо графічно зобразити створену онтологію (рис. 3.9).

Створена онтологія складається з головного класу «Хліб», який включає етапи, які представлені у вигляді класів:

- підготовка сировини до виробництва, а також підкласи «борошно», «сіль», «дріжджі», «вода»;
- технологічний процес приготування хліба, а також підкласи «приготування опари», «приготування тіста», «вистоювання, випікання» [120-122].

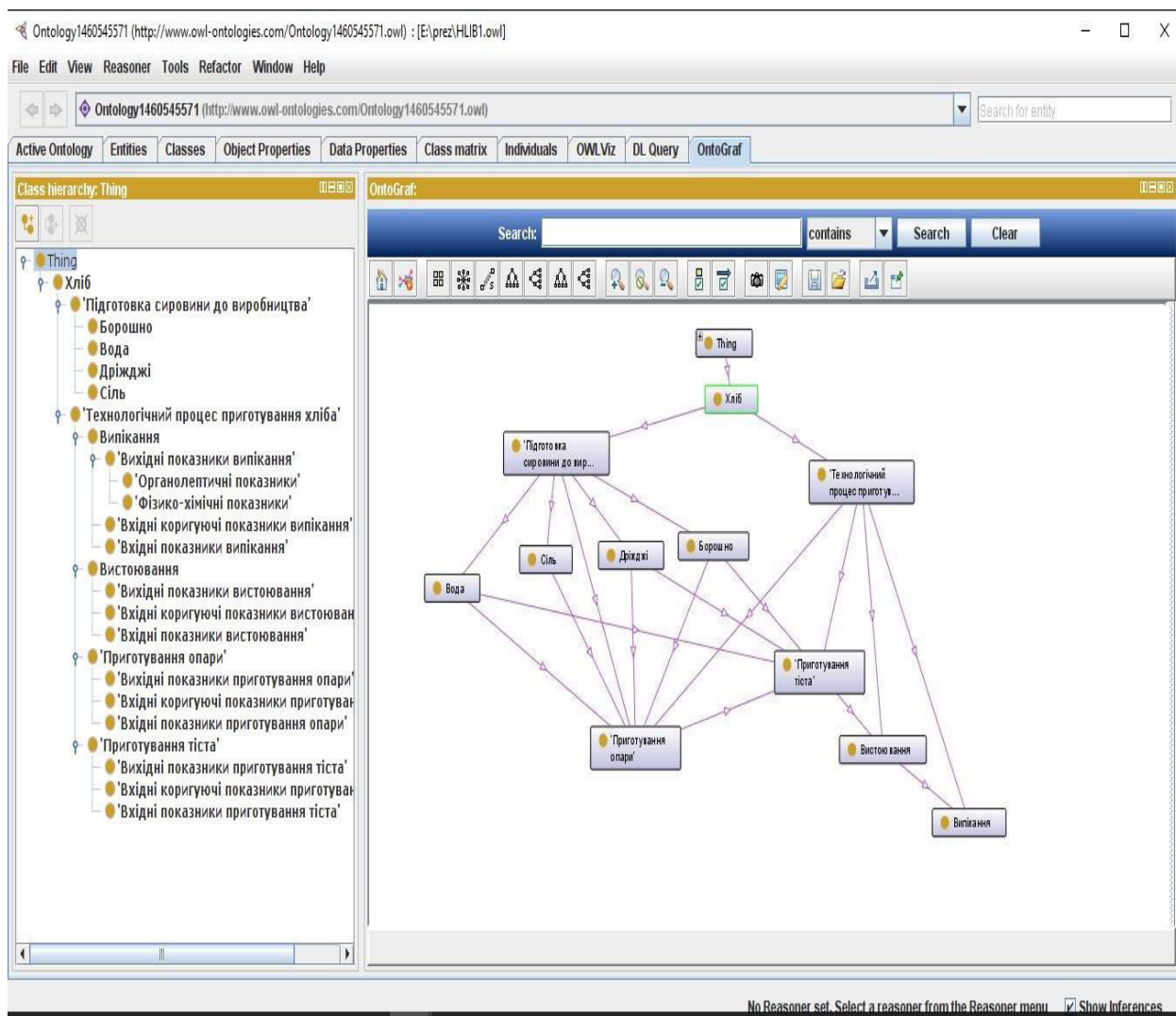


Рис. 3.9. Графічне представлення створеної онтології хлібопекарського виробництва

Для стадії «випікання» було створено підклас «вихідні показники випікання», який включає в себе підклас «фізико-хімічні показники». Для фізико-хімічних показників характерні наступні слоти, на які накладаються свої обмеження (допустимі діапазони технологічних змінних) залежно від технологічного процесу [120-122]:

- 1) вологість готового хліба;
- 2) маса готового виробу;
- 3) пористість хліба;
- 4) температура центра м'якуша;
- 5) титрована кислотність хліба.

Онтології на базі концептуалізації передбачають виділення її трьох взаємопов'язаних компонентів: таксономії термінів, описів сенсу термінів, а також правил їх використання й обробки у вигляді онтологічної моделі як знакової системи.

$M = (O, A, R)$, де $O = \{o_1, o_2, \dots\}$ – множина онтологій; A – множина аксіом $\{a_1, a_2, \dots\}$;

R – функція, яка ставить у відповідність кожному елементу множини O деяку підмножину елементів із множини A [4].

У ході проведених досліджень, у результаті аналізу літературних джерел і опитування експертів було виявлено такі залежності параметрів, які використовуються для оцінки якості виготовлення хліба на стадії приготування опари:

$$\begin{aligned} W_{on} &= f_{21}(F_{\bar{o}}, ВПЗ_{\bar{o}}, G_{\bar{o}}^o, G_{\bar{e}}^o, t_{\bar{e}}); \\ PC_{on} &= f_{22}(\Gamma_{\bar{o}}, AA_{\bar{o}}, PC_{\bar{o}}, K_{on}, G_{\bar{o}}^o, G_{\bar{e}}^o, G_{\bar{o}} \tau_{3o}, t_{on}, \tau_{\bar{o}o}); \\ K_{on} &= f_{23}(AA_{\bar{o}}, K_{\bar{o}}, K_{\bar{o}}, G_{\bar{o}}^o, \tau_{3o}, t_{\bar{e}}, \tau_{\bar{o}o}), \end{aligned}$$

де W_{on} – вологість опари;

$F_{\bar{o}}$ – сила борошна;

$ВПЗ_{\bar{o}}$ – водопоглинальна здатність борошна;

$G_{\bar{o}}^o$ – дозування частини борошна під час приготування опари;

$G_{\bar{e}}^o$ – дозування частини води під час приготування опари;

$t_{\bar{e}}$ – температура води;

PC_{on} – підйомна сила опари;

$\Gamma_{\bar{o}}$ – газоутворювальна здатність борошна;

$AA_{\bar{o}}$ – автолітична активність борошна;

$PC_{\bar{o}}$ – підйомна сила дріжджової суспензії;

K_{on} – титрована кислотність опари;

$G_{\bar{o}}$ – дозування дріжджової суспензії;

τ_{3o} – зміна тривалості замісу опари;

t_{on} – температура опари;

$\tau_{\bar{o}o}$ – тривалість бродіння опари;

$K_{\bar{o}}$ – титрована кислотність борошна;

$K_{\bar{o}}$ – титрована кислотність дріжджової суспензії.

Для стадії приготування тіста встановленні наступні залежності між параметрами:

$$\begin{aligned} W_m &= f_{31}(ВПЗ_{\bar{o}}, AA_{\bar{o}}, G_{\bar{e}}^m); \\ PC_m &= f_{32}(\Gamma_{\bar{o}}, AA_{\bar{o}}, PC_{on}, K_m, G_{\bar{o}}^m, G_{\bar{e}}^m, G_{cp}, \tau_{3.on}, t_m, \tau_{\bar{o}m}); \\ K_m &= f_{33}(AA_{\bar{o}}, K_{\bar{o}}, K_{on}, G_{\bar{o}}^m, \tau_{3.m}, t_{\bar{e}}, \tau_{\bar{o}m}), \end{aligned}$$

де W_m – вологість тіста;

PC_m – підйомна сила тіста;

K_m – титрована кислотність тіста;

t_m – температура тіста;

$\tau_{\bar{o}m}$ – тривалість бродіння тіста;

$\tau_{3.m}$ – тривалість замісу тіста;

Для стадії вистоювання :

$$\Phi_{3m} = f_{41}(F_{\bar{o}}, H_{idk}, W_m, t_{ш}, \tau_{вuc});$$

$$W_{m3} = f_{42}(W_m, \varphi_{ш}, t_{ш}),$$

де Φ_{m3} – формоутримувальна здатність тістової заготовки;

W_m – вологість тіста;

$t_{ш}$ – температура в шафі вистоювання;

$\tau_{вuc}$ – тривалість знаходження тістової заготовки в шафі вистоювання;

$W_{mз}$ – вологість тістової заготовки;

$\varphi_{ш}$ – вологість у шафі вистоювання;

$t_{ш}$ – температура в шафі вистоювання.

У процесі випікання, залежно від вхідних показників, змінюють:

– температуру ($t_{вun}$) й вологість ($\varphi_{вun}$) випікання;

– тривалість випікання ($\tau_{вun}$).

3.8. Виклад основних результатів дослідження.

Онтологія для технологічного комплексу хлібопекарського виробництва створювалася в програмному засобі для розробки онтологій – Protégé. Protege включає редактор онтології. На основі сформованих онтологій в Protege згенеровані форми здобуття знань для введення екземплярів класів і підкласів. Protege заснований на фреймовій моделі представлення знання і забезпечений рядом інструментів, що дозволяє адаптувати його для редагування моделей в різних форматах(стандартний текстовий, бази даних, UML, мов XML, XOL, SHOE, RDF і RDFS, Daml+oil, OWL).

На рис. 3.10 зображено вигляд головного вікна програмного засобу, в якому створювалася онтологія, наведено ієрархію класів (Class Hierarchy), слоти вибраного класу (Template Slots) та обмеження, накладені на слоти (Other Facets).

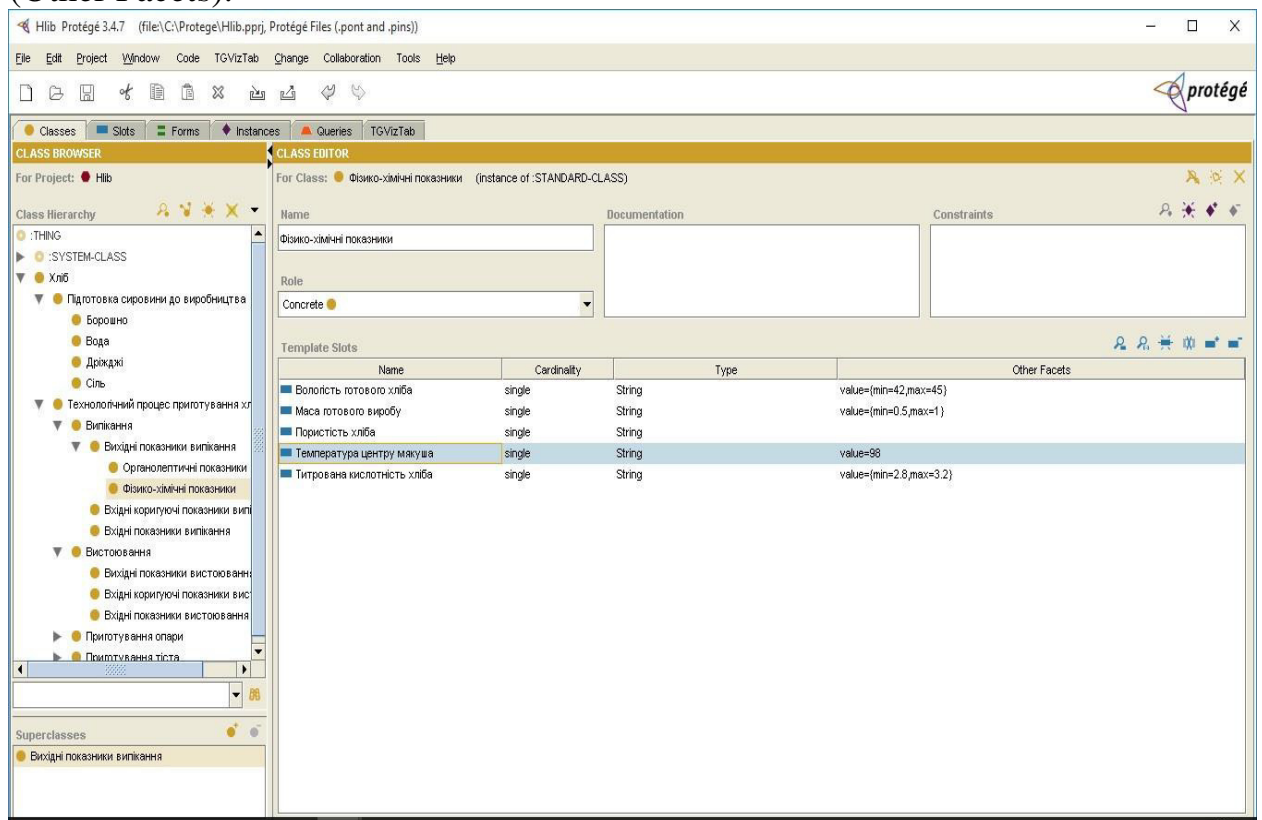


Рис. 3.10. Головне вікно програмного засобу Protégé

За допомогою інтеграції програмного засобу OntoGraf в Protégé та використовуючи вкладку OntoGraf можливо графічно зобразити створену онтологію (рис. 3.11).

Створена онтологія складається з головного класу “Хліб”, який включає наступні етапи, які представлені у вигляді класів:

- Підготовка сировини до виробництва, а також підкласи “борошно”, “сіль”, “дріжджі”, “вода”;
- Технологічний процес приготування хліба, а також підкласи “приготування опари”, “приготування тіста”, “вистоювання, випікання”.

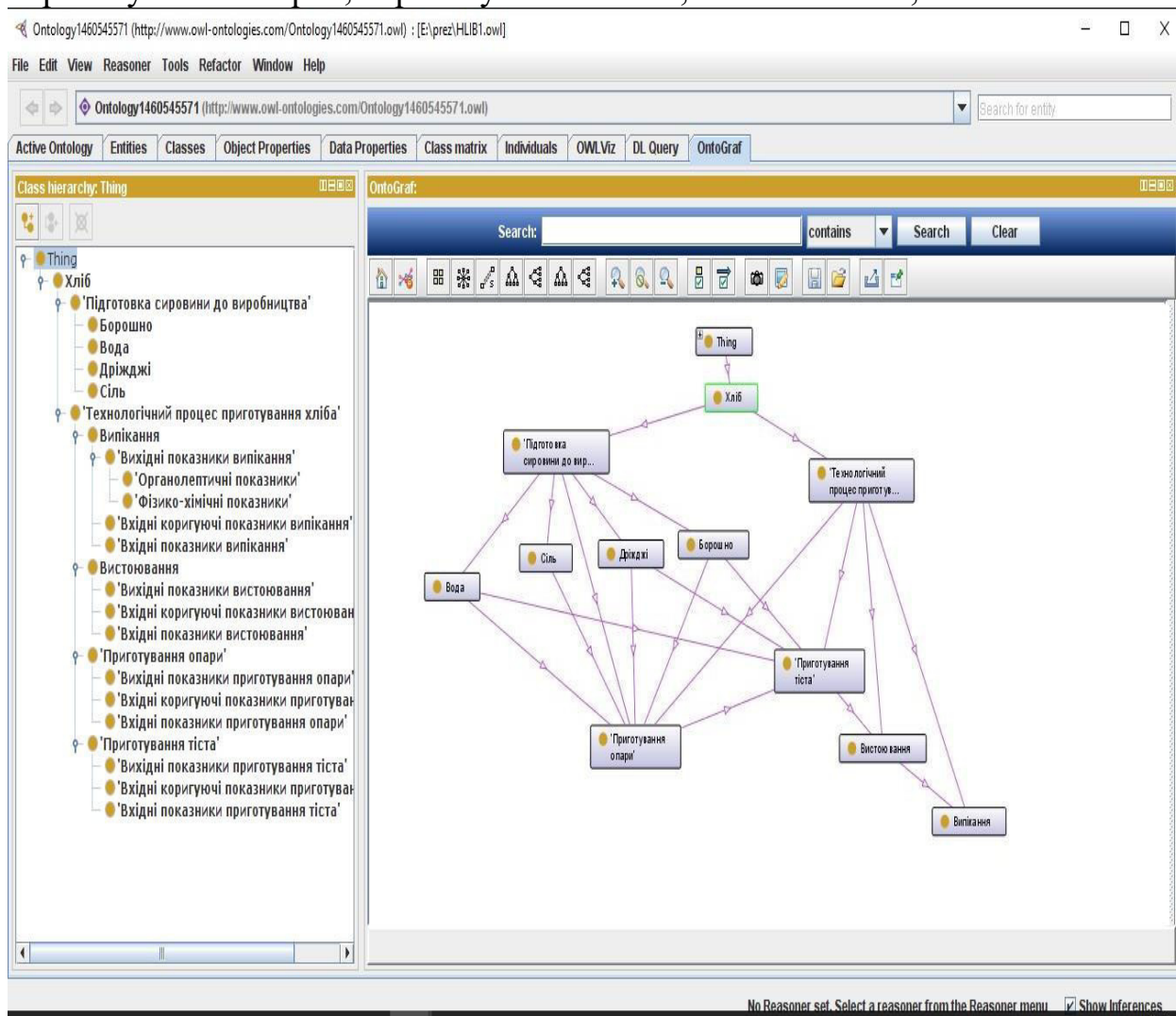


Рис. 3.11. Графічне представлення створеної онтології

Для стадії «випікання» було створено підклас «вихідні показники випікання» і який включає в себе підклас «фізико-хімічні показники». Для фізико-хімічних показників було створено наступні слоти і на ці слоти накладаються свої обмеження (допустимі діапазони технологічних змінних) в залежності від технологічного процесу:

- 6) Вологість готового хліба;
- 7) Маса готового виробу;
- 8) Пористість хліба;

- 9) Температура центра м'якуша;
- 10) Титрована кислотність хліба.

Була розроблена онтологія підтримки прийняття рішень при приготуванні хліба (рис 3.12).

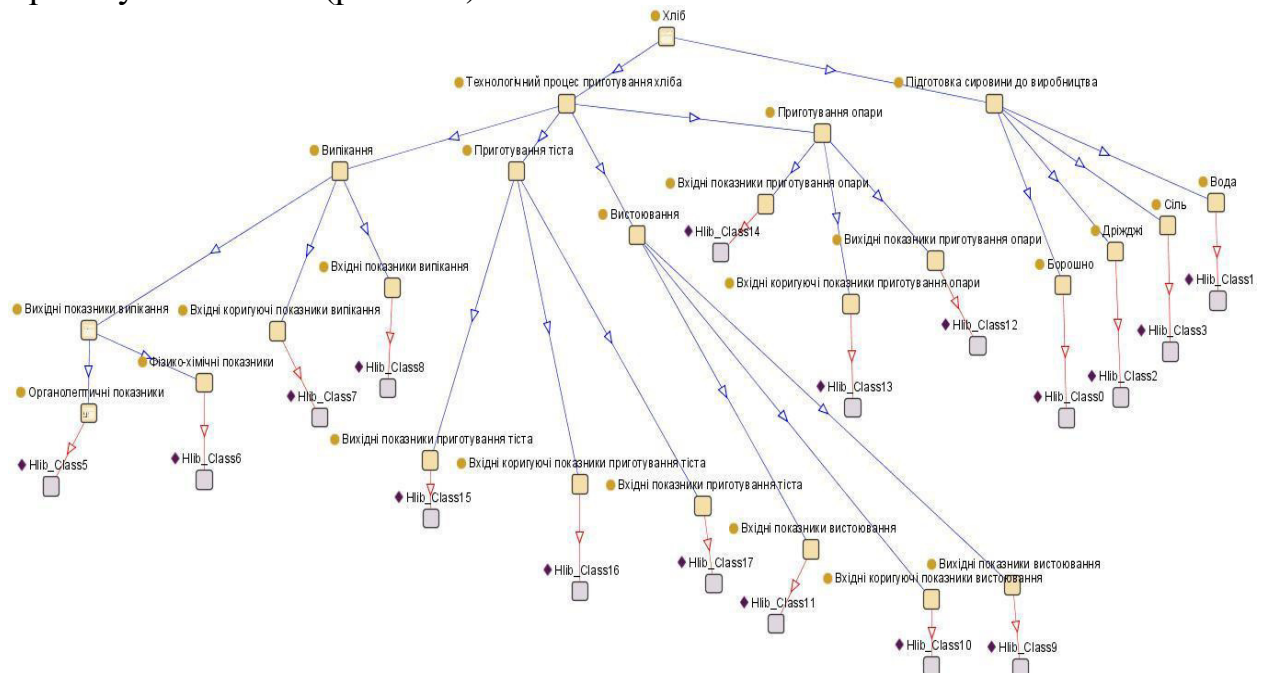


Рис 3.12. Онтологія підтримки прийняття рішення при приготування хліба.

Розроблена онтологія підтримки прийняття рішень в хлібопекарському виробництві може бути представлена як набір елементів [1]:

$$O = \langle C, R, Pr, V, I, A, D \rangle, \quad (1)$$

де, C – множина класів $\{C_1, C_2, \dots, C_n\}$ та їх інтерпретацій;

R – множина відносин $\{R_1, R_2, \dots, R_n\}$;

Pr – властивості класів;

V – значення властивості;

I – множина екземплярів класу $\{I_1, I_2, \dots, I_n\}$, визначається за допомогою аксіом і визначення конкретних властивостей класів;

A – множина аксіом $\{A_1, A_2, \dots, A_n\}$;

D – множина алгоритмів логічного виведення на онтології $\{D_1, D_2, \dots, D_n\}$;

Формалізація правил здійснена на основі дескриптивної логіки, а прецеденти є екземплярами класів онтології і являють собою сукупність об'єктів. Онтології на базі концептуалізації передбачають виділення її трьох взаємопов'язаних компонентів: таксономії термінів, описів сенсу термінів, а також правил їх використання і обробки у вигляді онтологічної моделі як знакової системи.

$$M = (O, A, R),$$

де $O = \{o_1, o_2, \dots\}$ – множина онтологій;

A – множина аксіом $\{a_1, a_2, \dots\}$;

R – функція, яка ставить у відповідність кожному елементу множини O деяку підмножину елементів з множини A [1, 4].

В ході проведених досліджень, в результаті аналізу літературних джерел і опитування експертів було виявлено такі залежності параметрів, які використовуються для оцінки якості виготовлення хліба на стадії приготування опари:

$$\begin{aligned} W_{on} &= f_{21}(F_{\bar{o}}, ВПЗ_{\bar{o}}, G_{\bar{o}}^o, G_e^o, t_e); \\ PC_{on} &= f_{22}(\Gamma_{\bar{o}}, AA_{\bar{o}}, PC_{\bar{o}}, K_{on}, G_{\bar{o}}^o, G_e^o, G_{\bar{o}} \tau_{zo}, t_{on}, \tau_{\bar{o}o}); \\ K_{on} &= f_{23}(AA_{\bar{o}}, K_{\bar{o}}, K_{\bar{o}}, G_{\bar{o}}^o, \tau_{zo}, t_e, \tau_{\bar{o}o}), \end{aligned}$$

де W_{on} – вологість опари;

$F_{\bar{o}}$ – сила борошна;

$ВПЗ_{\bar{o}}$ – водопоглинальна здатність борошна;

$G_{\bar{o}}^o$ – дозування частини борошна при приготуванні опари;

G_e^o – дозування частини води при приготуванні опари;

t_e – температура води;

PC_{on} – підйомна сила опари;

$\Gamma_{\bar{o}}$ – газоутворююча здатність борошна;

$AA_{\bar{o}}$ – автолітична активність борошна;

$PC_{\bar{o}}$ – підйомна сила дріжджової суспензії;

K_{on} – титрована кислотність опари;

$G_{\bar{o}}$ – дозування дріжджової суспензії;

τ_{zo} – зміна тривалості замісу опари;

t_{on} – температура опари;

$\tau_{\bar{o}o}$ – тривалість бродіння опари;

$K_{\bar{o}}$ – титрована кислотність борошна;

$K_{\bar{o}}$ – титрована кислотність дріжджової суспензії.

Для стадії приготування тіста встановленні наступні залежності між параметрами:

$$\begin{aligned} W_m &= f_{31}(ВПЗ_{\bar{o}}, AA_{\bar{o}}, G_e^m); \\ PC_m &= f_{32}(\Gamma_{\bar{o}}, AA_{\bar{o}}, PC_{on}, K_m, G_{\bar{o}}^m, G_e^m, G_{cp}, \tau_{z.on}, t_m, \tau_{\bar{o}m}); \\ K_m &= f_{33}(AA_{\bar{o}}, K_{\bar{o}}, K_{on}, G_{\bar{o}}^m, \tau_{z.m}, t_e, \tau_{\bar{o}m}), \end{aligned}$$

де W_m – вологість тіста;

PC_m – підйомна сила тіста;

K_m – титрована кислотність тіста;

t_m – температура тіста;

$\tau_{\bar{o}m}$ – тривалість бродіння тіста;

$\tau_{z.m}$ – тривалість замісу тіста;

Для стадії вистоювання :

$$\Phi_{3m} = f_{41}(F_{\bar{o}}, H_{idk}, W_m, t_{us}, \tau_{вус});$$

$$W_{mз} = f_{42}(W_m, \varphi_{us}, t_{us}),$$

де $\Phi_{mз}$ – формоутримуюча здатність тістової заготовки;

W_m – вологість тіста;

$t_{ш}$ – температура в шафі вистоювання;

$\tau_{вис}$ – тривалість знаходження тістової заготовки в шафі вистоювання;

$W_{мз}$ – вологість тістової заготовки;

$\varphi_{ш}$ – вологість в шафі вистоювання;

$t_{ш}$ – температура в шафі вистоювання.

В процесі випікання, в залежності від вхідних показників, змінюють:

– температуру ($t_{вип}$) і вологість ($\varphi_{вип}$) випікання;

– тривалість випікання ($\tau_{вип}$).

Розглянуто методологічний підхід до розробки онтологічних моделей, визначена область досліджень, виявлені теоретичні аспекти розробки онтології предметної області. Зроблена структуризація і формалізація даних предметної області «Приготування хліба». Визначені характеристики елементів онтології і описані їх значення. Розроблена структура класів, підкласів, виявлені характеристики, що описують ці поняття, і розроблена понятійна структура онтології. На початку дослідження доцільно виконати перенесення знань з різних джерел у базу знань комп'ютерної програми, що називається експертною системою (ЕС). ЕС дозволяють обґрунтувати отримані результати, посиляють користувачеві необхідну інформацію і сприяють упевненому ухваленню правильного рішення.

На основі створеної онтології розробляються продукційні правила, на мові програмування експертних систем CLIPS, які будуть складати базу знань предметної області технологічного процесу для системи підтримки та прийняття рішень на основі експертної системи.

3.9. Висновки до розділу

Розділ присвячений розробці методу побудови онтологій на основі результатів моделювання предметної області з використанням засобів УФО-підходу. Розглянуто основні складові зазначеного методу. Виявлено значення етапу валідації отриманої за допомогою розробленого методу онтології.

Розроблено основні правила валідації, наведено їх опис на мові SWRL. Систематизовано основні завдання логічного висновку на УФО-моделях, запропоновано їх класифікацію. Розроблено метод логічного виведення на УФО-моделях шляхом їх перетворення в онтології і застосування засобів логічного висновку на онтологічних структурах, розглянуті його переваги.

Дослідження існуючих засобів і інструментів логічного висновку показало, що вони можуть бути використані для вирішення задач логічного висновку знань на УФО-моделях, однак вимагають модифікації і адаптації для вирішення конкретних прикладних завдань; при цьому з окремих випадках оптимальним рішенням може бути розробка нових інструментів, які враховують специфіку УФО-підходу.

Також у цьому розділі розглядаються результати розробки алгоритмічного та програмного забезпечення розроблених методів побудови онтологій на основі УФО-моделей і логічного висновку на них. Запропоновано алгоритм перетворення формального уявлення УФО-моделі в форматі XML в

конструкції мови опису онтологій RDF в нотації RDF / XML. Описано його основні кроки і особливості окремих, найбільш складних етапів, таких як: витяг з вихідного файлу відомостей про класифікацію зв'язків, використовуваної в УФО-моделі, 132 формування на основі вихідного файлу триплетів, що описують УФОелемент в цілому і його характеристики з точки зору Вузла, Функції, Об'єкта, а також зв'язку між УФО-елементами. Розглянуто можливості вирішення прикладних завдань логічного виведення на УФО-моделях за допомогою засобів, що застосовуються до онтології (Мова запитів SPARQL, мова правил SWRL). Спроектовано та реалізовано прототип програмного продукту, реалізує запропоновані алгоритми. Прототип адаптований і апробований на прикладі вирішення завдань управління знаннями в компаніїразработчіке програмного забезпечення. впровадження показало позитивний ефект запропонованих методів і засобів, зокрема, дозволило знизити навантаження на співробітників підприємства, скоротити час пошуку і формування нових знань, систематизувати наявні знання і організувати загальний доступ до них.

Висновки

1. На основі аналізу технологічного комплексу хлібозаводу складено глосарій понять предметної галузі.
2. Визначені класи онтологій та побудована ієрархія онтологій хлібопекарського виробництва, що створило необхідні передумови для побудови бази знань динамічної експертної системи управління технологічним комплексом хлібозаводу.
3. Визначені атрибути класів, зв'язки між класами та властивості екземплярів онтологій хлібопекарського виробництва, що забезпечило конкретизацію рішень по управлінню технологічними процесами хлібопекарського виробництва у відповідності із ситуацією, що склалась на об'єкті управління.
4. В результаті систематизації відомостей про предметну область, містяться в поданих за допомогою УФО-підходу візуальних графоаналитических моделях, розроблена класифікація відповідних фактів.
5. На підставі результатів аналізу існуючих методів побудови онтологій обґрунтована можливість створення онтологій на основі візуальних графоаналитических моделей в термінах «Вузол-ФункціяОб'єкт» (УФО-моделей).
6. Проаналізовано та адаптовані для використання при перетворенні УФО-моделей в конструкції мов формального уявлення онтологій існуючі засоби алгебраїчного опису УФО-моделей.
7. Розроблено метод і алгоритм опису УФО-моделі на формалізованому мові представлення онтологій RDF (Resource Description Framework), що дозволило спростити і автоматизувати первинний етап побудови онтології, що включає в себе відбір концептів і встановлення взаємозв'язків між ними.
8. Розроблено метод побудови онтології на основі візуальної графоаналитического УФО-моделі, що дозволяє використовувати в якості джерела знань результати моделювання предметної області засобами УФО-аналізу.
9. Запропоновано метод вирішення основних завдань логічного висновку на УФО-моделях з урахуванням інтеграції засобів УФО-підходу та онтологічного інжинірингу.

10. Розроблена програмна реалізація запропонованих алгоритмів перетворення опису УФО-моделі в формалізоване уявлення 134 онтології на мові RDF (Resource Description Framework) і логічного виведення на УФО-моделях, працездатність якої апробована на прикладі управління організаційними знаннями підприємства в сфері розробки програмного забезпечення. Рекомендації.

Результати дослідження й надалі можуть бути використані для вирішення завдань організації баз знань, створення прикладних інтелектуальних інформаційних систем, призначених, в тому числі, для підтримки прийняття рішень в складних слабоструктурованих областях, створення семантичних web-сервісів. Одним з перспективних напрямків подальшої розробки теми є розширення переліку прикладних задач логічного висновку на УФО-моделях, а також розробка способів їх вирішення. результати виконаного дослідження також дозволяють закласти основи для подальшої розробки нового методу вирішення завдань порівняння і інтеграції (злиття) декількох онтологій засобами системнооб'єктного підходу, зокрема, за допомогою методу «зворотного» перетворення опису онтології на спеціалізованій мові в формальний опис УФО-моделі.

Список використаної літератури

1. Луцька Н.М. Сучасні технології проектування інтелектуальних систем керування [Електронний ресурс] конспект лекцій для здобувачів освітнього ступеня «Магістр» спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньо-професійної програми «Інтелектуальні комп'ютерні системи керування» денної та заочної форм навчання / Н.М. Луцька, А.П. Ладанюк К.: НУХТ, 2019. – 117 с.
2. Ицкович Э.Л. Методы комплексной автоматизации производства предприятий технологических отраслей / Э.Л. Ицкович. – М.: КРАСАНД. 2013. – 232 с.
3. Методичні рекомендації до виконання випускної кваліфікаційної роботи на здобуття освітнього ступеня «магістр» спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньої програми «Інтелектуальні комп'ютерні системи керування» денної та заочної форм навчання / Уклад.: А.П. Ладанюк, Н.М. Луцька, Я.В. Смітюх, В.Д. Кишенько.[Електронний ресурс]. – К. : НУХТ, 2020. – 70 с.
4. Проектування систем автоматизації галузі [Електронний ресурс]: Метод. рекомендації до викон. курс. проекту для студ. освітнього ступеня «магістр» спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» спеціалізації «Інтелектуальні комп'ютерні системи керування» ден. форм навч. / уклад.: Трегуб В.Г., Луцька Н.М., А.П. Ладанюк. – К.: НУХТ, 2017. – 48 с.
5. Нестеров А. Л. Проектирование АСУТП. Книга 1 / А.Л. Нестеров. – СПб.: ДЕАН, 2006. –552 с.
6. Нестеров А. Л. Проектирование АСУТП. Книга 2 / А.Л. Нестеров. – СПб.: ДЕАН, 2009. – 944 с.
7. Про KPI та OEE. Загальні розрахунки згідно ISO 22400-2. URL: <http://www.slideshare.net/pupenasan/kpi-oee>.

8. Промислові мережі та інтеграційні технології в автоматизованих системах: Навч. посібник / О.М. Пупена, І.В. Ельперін, Н.М. Луцька, А.П. Ладанюк. – К.: Ліра-К, 2011. – 552 с.
9. Пупена О.М. [Електронний ресурс]: Автоматизовані системи управління виробництвом (MES-рівень): курс лекцій для студ. освіт. ст. "магістр" спец. 151 "Автоматизація та комп'ютерно-інтегровані технології" спеціалізації "Інтегровані автоматизовані системи управління " денної та заочної форм навчання / О.М. Пупена, Р.М. Міркевич. – К.: НУХТ, 2016. – 135 с.
10. Трегуб В.Г. Автоматизація об'єктів періодичної дії: [підручник] / В.Г. Трегуб. – К.: Ліра-К, 2016. – 136 с.
11. Фёдоров Ю.Н. Справочник инженера по АСУТП: Проектирование и разработка. Учебно-практическое пособие / Ю.Н. Фёдоров. – М.: Инфра-Инженерия, 2008. – 928 с.
12. A Practical Guide to SysML. The Systems Modeling Language. 2-d ed/ /Sanford Friedenthal, Alan Moore, Rick Steiner, Elsevier Inc. 2012.
13. Буч Г., Рамбо Д., Якобсон И. Б90 Язык UML. Руководство пользователя. 2-е изд.: Пер. с англ. Мухин Н. – М.: ДМК Пресс, 2006. – 496 с.
14. Трегуб В.Г. Проектування систем автоматизації: навч. посібник / В.Г. Трегуб. – К.: Ліра-К, 2014.
15. Нестеров А.Л. Проектирование АСУТП. Книга 1 / А.Л. Нестеров // СПб.: Издательство ДЕАН. – 2006. – 844 с.
16. Нестеров А.Л. Проектирование АСУТП. Книга 2 / А.Л. Нестеров // СПб.: Издательство ДЕАН. – 2009. – 944 с.
17. Ельперін І.В. Промислові контролери: Навчальний посібник / І.В. Ельперін // К.: НУХТ. – 2003. – 320 с.
18. Ладанюк А.П. Автоматизація технологічних процесів та виробництв харчової промисловості: Підручник / Ладанюк А.П, Трегуб В.Г., Ельперін І.В., Цюцюра В.Д. // К.: Аграрна освіта. – 2001. – 224 с.