УДК 004.9
О.К.Жигаревич
Луцький інститут розвитку людини Університету “Україна”

ЗАСТОСУВАННЯ МОВИ ОНТОЛОГІЇ ДЛЯ ВИРІШЕННЯ ПРОБЛЕМИ НЕОДНОРІДНОСТІ МОВ ПРЕДСТАВЛЕННЯ ЗНАНЬ

У статі описуються мови онтології, а також розглядається їх об’єднання за допомогою спеціальних методів: iPrompt, Chimaera, OntoMerge, OntoMorph, GLUE, OBSERVER, FCA-Merge, ONION. Досліджується призначення онтології на основі моделі XOL (XML-based Ontology Exchange Language), яку створено для забезпечення потреб у мові з семантикою об’єктно - орієнтованих систем представлення знань. Характеризуються концепти дескриптивної логіки DL. Стаття висвітлює сучасний стан комп’ютерних технологій, методи та засоби для підтримки процесу побудови та використання онтології.

Ключові слова: *онтологія, семантика, дескриптивна логіка.*

Постановка проблеми

У XXI ст. із швидким розвитком комп’ютерних технологій, людина для пошуку потрібної інформації користується всесвітньою павутиною Інтернет. Інформація в мережі організована не систематично навіть хаотично, це уповільнює пошук потрібного матеріалу, можна декілька разів знаходити одні і тіж дані. Засобам обробки даних у мережі все складніше і складніше справлятися з масою інформації, що вже існує і яка додається щодня. Для більш ефективного також взаємного використання інформації повинно бути вирішено багато технічних проблем. А саме проблема з'єднання гетерогенних і розподілених комп'ютерних систем (interoperability problem). Інтероперабельність потрібно забезпечувати і на технічному і на інформаційному рівні. Проблеми, що могли б виникати внаслідок гетерогенності даних, вже відомі в межах співтовариства розподілених систем баз даних: структурна гетерогенність (схематична гетерогенність) і семантична гетерогенність (гетерогенність даних). Структурна гетерогенність означає, що різні інформаційні системи зберігають свої дані в різних структурах. Семантична гетерогенність розглядає зміст інформаційного елемента. Щоб досягати семантичної інтероперабельності в гетерогенній інформаційній системі, зміст інформації, якою обмінюються, повинний бути зрозумілим у всіх системах.

Стародавній філософ Арістотель при спробі класифікувати предмети у світі запропонував термін онтологія. Іншими словами онтологія – це наука про буття, наука про природу речей і взаємозв’язки між ними. Терміном онтологія можна визначити деякий механізм інформаційних технологій представлення знань, а саме спосіб для опису предметної області, базові поняття цієї області, їхні властивості та зв’язки між ними. На даний час під терміном онтологія можна розуміти:

- надійний семантичний базис у визначенні змісту;
- логічну теорію, що складається із словника та набору тверджень;
- основу для взаємин між людьми та комп’ютерними агентами.

Використовується онтологія для пояснення неявного і схованого знання - можливий підхід перебороти проблему семантичної гетерогенності. Семантична гетерогенність розглядає зміст інформаційного елемента. Щоб досягати семантичної інтероперабельності в гетерогенній інформаційній системі, зміст інформації, якою обмінюються, повинний бути зрозумілим у всіх системах. Семантичні конфлікти відбуваються всякий раз, коли два контексти не використовують однакову інтерпретацію інформації.

Головним поштовхом у розвитку онтологій, було передбачення SemanticWeb [1]. У SemanticWeb онтології забезпечують концептуальне підкріплення для створення семантики метаданих.

Онтології дозволяють представити нові поняття так, що вони стають придатними для машинної обробки. За допомогою онтології можна забезпечити обмін між новими поняттями, з якими система ще не зустрічалася, із описами уже відомих класів, відносин, властивостей і об'єктами реального часу.

Метою дослідження виступає наступне:

- спільне використання загального розуміння структури інформації серед користувачів;
- можливість повторного використання знань домену;
- можливість робити явними припущення домену;
- відокремлення знань домену від операційного знання;

Спільне використання загального розуміння структури інформації серед людей - одна з найбільш загальних цілей у розробці онтологій. Наприклад, припустимо, що декілька різних Інтернет-сторінок містять юридичну інформацію або надають юридичні послуги електронної комерції. Якщо ці Інтернет-сторінки спільно використовують і видають однакову основну онтологію термінів, які вони використовують, то комп'ютерні агенти можуть витягати і з'єднувати інформацію з цих різних Інтернет-сторінок. Агенти можуть використовувати цю з'єднану інформацію, щоб відповісти на запити користувача або як вхідні дані для інших прикладних програм.

Надання повторного використання знання домену є однією з рушійних сил в дослідженні онтології. Наприклад, моделі для багатьох різних доменів повинні представити поняття часу. Це представлення включає поняття інтервалів часу, точки в часі, відносні міри часу і т.д. Якщо одна група дослідників розробила таку докладну онтологію, інші можуть просто повторно використовувати її для своїх доменів. Додатково, якщо необхідно будувати велику онтологію, можна інтегрувати кілька існуючих онтологій, що описують частини великого домену.

Відділення знання домену від операційного знання - інше звичайне використання онтологій. Можна описувати задачу конфігурації продукту з його компонентів відповідно до необхідної специфікації і реалізувати програму, що робить цю конфігурацію, незалежно від продукту і компонентів безпосередньо. Як тільки стає доступною декларативна специфікація термінів, можливий аналіз знання домену.

Ontolingua - мова, яка заснована на KIF [7] і на FrameOntology [8]. KIF (Knowledge Interchange Format) був розроблений для вирішення проблеми неоднорідності мов для представлення знань. Він забезпечує визначення об'єктів, функцій і відношень. KIF має декларативну семантику, що заснована на численні предикатів першого порядку з префіксною нотацією. Власне кажучи, оскільки KIF є форматом обміну, його використання занадто трудомістке для специфікації онтологій. Ontolingua дозволяє включати у визначення вирази KIF, засновані на FrameOntology.

Онтологічна модель OKBC (Open Knowledge Base Connectivity) [4], раніше відома як Generic Frame Protocol, використовує у своїй основі фреймову модель. Вона визначає протокол для доступу до баз знань, що зберігаються у фреймових системах представлення знань і розглядається як доповнення до мовних специфікацій, які розроблені для підтримки спільного використання знань. Модель знань GFP є неявним формалізмом представлення, що лежить в основі OKBC і забезпечує набір конструкцій таких як: константи, фрейми, слоти, фасети, класи і бази знань.

Модель XOL (XML-based Ontology Exchange Language) [10] розроблена з метою задовольнити потребу в мові з семантикою об'єктно-орієнтованих систем представлення знань, але з синтаксисом XML. Визначення онтології, яку призначений кодувати XOL, включають як схемну інформацію (метадані), такі як визначення класу, так і несхемну інформацію (основні факти) такі як визначення об'єкта в об'єктних базах даних. XOL є форматом для обміну онтологічними визначеннями і тому, не призначений для розробки онтологій Він служить в якості проміжної мови для передачі онтологій серед різних систем баз даних, інструментів розробки онтологій або прикладних програм. XOL подібний до інших мов обміну онтологіями, зокрема, використовує модель OKBC.

Інший клас формальних онтологічних моделей ґрунтується на різних видах дескриптивних логік (DL) [5]. Сімейство формалізмів представлення знань, заснованих на логіках, добре підходить для представлення і міркування про термінологічне знання й онтологій. Вони головним чином характеризуються набором конструкторів, що дозволяють формувати складні концепти і ролі з елементарних концептів.

Ці моделі трактуються в задачах перевірки категоризації між концептами і прийнятності, щодо визначень концептів подібно множинам взаємозалежних концептів. KL-ONE, LOOM, CLASSIC, FaCT - приклади систем, здатних до міркування, які часто використовуються для моделювання онтологій. У них забезпечуються типові сервіси міркувань - класифікація, контроль сумісності та екземплярів.

Для розгляду концептів дескриптивної логіки у вигляді множини об'єктів вводиться поняття інтерпретації. Перевірка здійсненності складається з доказу, що стверджує, що існує принаймі одна інтерпретація відповідно до якої концепт відповідає непорожній множині. Категоризація між концептами означає, що для будь-якої інтерпретації множина, що інтерпретує один концепт, є підмножиною множини, що інтерпретує інший концепт. Доказ категоризації зводиться до такої здійсненності.

Першою серед систем DL була KL-ONE [2], де заявлялось про перехід від семантичної мережі до більш обґрунтованих термінологічних (дискриптивних) логік. Вплив був дуже суттєвим, тому KL-ONE розглядається як родоначальник цілого сімейства мов.

Ідея створення CLASSIC [1] базується на забезпеченні виразної мови та ефективного міркування. CLASSIC, підтримує опис об’єктів в термінах їх відношень до інших об’єктів, а також в термінах рівня змістовної структури. Суттєвий недолік систем такого типу - незавершені алгоритми міркування [5].

LOOM [13] відноситься до того ж типу моделей, що і CLASSIC. Це мова програмування високого рівня і середовище, призначене для використання в побудові експертних систем та інших інтелектуальних прикладних програм. LOOM вимагає щільного інтегрування між парадигмами заснованими на правилах і фреймах. LOOM підтримує дескриптивну мову для моделювання об’єктів і відношень, і мову тверджень для визначення обмежень на концептах і відношеннях.

FaCT (Fast Classification of Terminologies) [FHH 6] - система, що забезпечує підтримку міркування для проектування онтологій, інтеграції і верифікації. FaCT – DL класифікатор, що може також використовуватись для перевірки несуперечності у модальних та інших подібних логіках. Найбільш цікавою характеристикою FaCT є її виразна логіка (зокрема механізм міркування SHIQ), її оптимізована таблична реалізація (що стала стандартом для DL-систем) та її заснована на CORBA клієнт-серверна архітектура.

Розвиток онтологічних моделей на основі логік відбувається в рамках розробки Semantic Web, а саме стандарти, XML і RDF.

XML (eXtensible Markup Language) [3] метамова яка походить від SGML (Standard General Markup Language). Вона була розроблена W3C, для простоти реалізації і інтероперабельності з SGML та HTML. Як мови для Web, головними перевагами XML є наступне: її просто аналізувати, синтаксис добре визначений і вона підходить для людського сприйняття. Оскільки XML широко використовується є багато програмних інструментальних засобів для синтаксичного аналізу і керування нею. XML дозволяє користувачам визначати свої власні теги й атрибути, визначати структури даних (вкладаючи їх), витягати дані з документів. Безпосередньо XML не має ніяких спеціальних можливостей для специфікації онтологій, оскільки він пропонує усього лише простий, але могутній спосіб визначити синтаксис для мови специфікацій онтологій. Тому, XML може використовуватися для таких цілей, як забезпечення синтаксису набору мов, таких як XOL або OIL, і для покриття потреб обміну онтологіями.

RDF (Resource Description Framework) [12] розроблений W3C для створення метаданих, що описують ресурси Web. Це дає можливість специфікації семантики даних, заснованих на XML, стандартизованим, інтероперабельним способом. Модель даних RDF складається з трьох типів об'єктів: ресурсів - об'єкти, що посилаються на адресу в Web; властивостей, що визначають певні аспекти, характеристики, атрибути чи відношення використовувані для опису ресурсу; і інструкції, що призначають значення для властивості у певному ресурсі.

Модель даних RDF не забезпечена механізмами для визначення відношень між властивостями (атрибутами) і ресурсами. Це - роль RDF Schema (RDF Schema Specification language) [14], декларативної мови, що використовується для визначення RDF схем. Вона заснована на деяких ідеях представлення знання (семантичних мережах, фреймах і логіці предикатів), але набагато більш проста в реалізації (але й менш виразна) ніж повні мови числень предикатів такі як CycL і KIF. Основні класи - клас, ресурс і властивість, крім того можуть бути визначені ієрархії й обмеження типів. Деякі базові обмеження також визначені. Але все ж для онтологічних визначень у RDF Schema не вистачає функцій і аксіом, але можуть бути легко визначені концепти, відношення й екземпляри.

OIL (Ontology Interchange Language) [6], є пропозицією спільного стандарту для опису й обміну онтологіями. Вона є першою мовою представлення онтологій у W3C. Це - розвиток існуючих пропозицій типу OKBC, XOL, RDF і перша заснована на Web мова, яка призначена для формування онтологій з формальною семантикою і сервісами міркування, що забезпечуються дескриптивною логікою. У OIL, онтологія це структура, що складається з декількох компонентів,

організованих у три рівні: об'єктний рівень (який має справу з екземплярами), перший мета-рівень (який містить визначення онтологій) і другий мета-рівень або онтологічний контейнер (який, містить інформацію щодо особливостей онтології, таку як авторство). Концепти, відношення, функції й аксіоми можуть бути визначені, використовуючи онтологічні визначення OIL.

OWL розширює RDF і RDF Schema, забезпечуючи поряд з формальною семантикою додатковий словник, що закладено в основу DAML + OIL, а також вбудовану підтримку відображення онтології. OWL має три діалекти: OWL Lite, OWL DL і OWL Full. Вони відрізняються складністю і можуть бути використані в різних прикладних програмах у залежності від необхідності або простоти виводу, або формальності описів.

SHOE (Simple HTML Ontology Extension) [11], розроблена в Університеті Штату Меріленд, була першим розширенням HTML, з метою включення зрозумілого для машин семантичного знання в HTML або інші документи Web. Вона нещодавно була адаптована для підтримки XML. Намір цієї мови полягає в тому, щоб зробити можливим для агентів зібрати значиму інформацію відносно Web сторінок і документів, поліпшуючи механізми пошуку і збір знань. Цей процес складається з двох фаз: визначення онтології, анотування HTML сторінки онтологічною інформацією для опису себе й інших сторінок.

OML (Ontology Markup Language), розроблена в Університеті Вашингтона, частково заснована на SHOE. Фактично, спочатку розглядалося XML перетворення SHOE. Отже, OML і SHOE мають багато однакових особливостей. Існує чотири різних рівні OML: Ядро OML, що зв'язане з логічними аспектами мови, його включають всі інші рівні; Простий OML, що безпосередньо відображається у RDF Scheme; Скорочений OML , який включає концептуальні графічні можливості і Стандартний OML - найбільш виразна версія OML.

На даний час онтології доступні в багатьох різних формах. Перед користувачем постає питання, коли він знаходить кілька онтологій, які б хотів використати, але вони не відповідають одна одній? Дослідники в різних областях інформатики працюють над автоматичним або підтримуваним інструментально об'єднанням онтологій (або ієрархії класів, або об'єктно-орієнтованих схем, або схем баз даних — визначена термінологія змінюються в залежності від області). Однак, і автоматичне об'єднання онтологій і створення інструментальних засобів, що керували б користувачем у цьому процесі, знаходяться на ранніх стадіях розвитку.

Інструментальні засоби, що мають справу з виявленням відповідностей між онтологіями можна класифікувати наступним чином:

- для об'єднання двох онтологій з метою створення однієї нової (наприклад, iPrompt, Chimaera, OntoMerge)
- для визначення функції перетворення, що перетворює одну онтологію в іншу (наприклад, OntoMorph)
- для визначення відображення між концептами в двох онтологіях, знаходячи пари зв'язаних концептів (наприклад GLUE, OBSERVER, FCA-Merge)
- для визначення правил відображення для зв'язку тільки початкових онтологій (наприклад ONION)

Chimaera - інтерактивний інструмент для об'єднання, заснований на редакторі онтологій Ontolingua. Chimaera дозволяє користувачу з'єднувати онтології, розроблені в різних формалізмах. Користувач може запросити аналіз або керівництво від Chimaera у будь-якій точці протягом процесу об'єднання. Потім інструмент направить його на місця в онтології, де потрібно його втручання. У своїх пропозиціях, Chimaera головним чином покладається на те, з якої онтології прибули концепти, засновуючись на їхніх іменах. Наприклад, Chimaera направить користувача на клас в об'єднаній онтології, що має два слоти, отримані від онтологій з двох різних джерел, або який має два підкласи, що прийшли з різних онтологій. Chimaera цілком залишає вирішення того, що робити користувачу і не робить ніяких пропозицій самостійно. Єдине таксономічне відношення, що розглядає Chimaera – відношення підклас-суперклас.

У OntoMerge об'єднана онтологія є об'єднанням двох початкових онтологій і набору аксіом сполучення. Перший крок у процесі об'єднання OntoMerge, полягає в трансляції обох онтологій до загального синтаксичного представлення мовою, розробленою авторами. Потім інженер онтології визначає аксіоми сполучення, що містять терміни з обох онтологій. Процес трансляції екземплярів виглядає в такий спосіб: всі екземпляри у початкових онтологіях, розглядаються, як ті що знаходяться в об'єднаній онтології.

Ontomorph визначає набір операторів перетворення, що можуть застосовуватися до онтології. Потім експерт використовує початковий список пар і початкові онтологій для

визначення набору операторів, що повинні застосуватися до початкових онтологій для усунення розходжень між ними, і Ontomorph застосовує ці оператори. Таким чином, сукупні операції можуть бути виконані за один крок.

GLUE - система використовує методи машинного навчання для знаходження відображення. GLUE використовує декількох учнів, що використовують інформацію в екземплярах концепту і таксономічній структурі онтологій. Для об'єднання результатів різних учнів використовується ймовірнісна модель.

Система OBSERVER використовує дескриптивну логіку для відповіді на запити, що використовують декілька онтологій та інформацію щодо відображень між ними. Спочатку, користувачі визначають набір міжонтологічних відношень. Система допомагає користувачам впоратися з цією задачею, за допомогою пошуку синонімів у початкових онтологіях. При визначенні відображення, користувачі можуть формулювати запити в термінах дескриптивної логіки, через свою власну онтологію. Потім OBSERVER використовує інформацію відображення для формулювання запитів до початкових онтологій. OBSERVER у значній мірі покладається на той факт, що описи в онтологіях і запитах є змістовними.

FCA-Merge - метод для порівняння онтологій, що мають набір спільних екземплярів або набір спільних документів, що анотуються за допомогою концептів з початкових онтологій. Ґрунтуючись на цій інформації FCA-Merge використовує математичні методи з Formal Concept Analysis для того щоб зробити решітку концептів, що зв'язує концепти з початкових онтологій. Алгоритм пропонує відношення еквівалентності і підклас-суперклас. Потім інженер онтології може аналізувати результат і використовувати його як порадник для створення об'єднаної онтології. Однак, припущення, що дві поєднувані онтології використовують спільний набір екземплярів або мають набір документів, у якому кожен документ анотується термінами обох джерел занадто сильне і на практиці така ситуація відбувається рідко. Як альтернативу, автори пропонують використання методів обробки природної мови для анотації набору документів концептами з цих двох онтологій.

Система ONION заснована на алгебрі онтології. Тому, вона надає інструментальні засоби для визначення правил артикуляції (сполучення) між онтологіями. Правила артикуляції звичайно враховують тільки маленькі релевантні частини початкових онтологій. ONION використовує і лексичні методи, і методи на основі графів для пропозиції артикуляції. Метод знаходить лексичну подібність між іменами концептів використовуючи словники і методи семантичної індексації, засновані на спільному місцезнаходженні групи слів у текстовій сукупності.

Hovy описує набір інструкцій, що використовували дослідники в ISI/USC для напівавтоматичного вирівнювання онтологій домену у велику центральну онтологію. Їхні методи засновані головним чином на лінгвістичному аналізі імен концептів і визначеннях концептів на природній мові, є також дуже обмежене використання таксономічних відношень. Спочатку, виявник збігів використовує методи обробки природної мови для розбивки імен, що складаються з декількох слів (звичайне місцезнаходження в іменах концептів). Потім, для того, щоб знайти подібні концепти, він порівнює підрядки різної довжини. Потім розглядаються слова, що використовуються у визначеннях концептів на природній мові. Виявник збігів зупиняється на словах у визначеннях, видаляє у словах основу, але зберігає дублікати. Потім він порівнює число і коефіцієнт спільних слів у визначеннях для знаходження подібних визначень. Експериментально визначена формула для об'єднання цих показників дає потенційні збіги, які користувач повинен дослідити і схвалити.

PROMPT - доповнення до системи Protege, є алгоритмом для об'єднання і групування онтологій. При об'єднанні двох отологій PROMPT створює список запропонованих операцій. Операція може заключатися, наприклад, в об'єднанні двох термінів або копіюванні термінів у нову онтологію. Користувач може виконати операцію вибором однієї з запропонованих або безпосереднім визначенням операції. PROMPT виконує обрану операцію і додаткові зміни, що визиває ця операція. Потім список запропонованих операцій модифікується і створюється список конфліктів і можливих вирішень цих конфліктів. Це повторюється поки не буде готова нова онтологія.

Фактично, коли розроблювач повинен вибрати інструмент для використання в керуванні множинними онтологіями, який інструмент є найбільш придатним буде залежати від конкретної задачі. Наприклад, якщо поєднувані онтології спільно використовують набір екземплярів, то найкраще може працювати FCA-Merge. Якщо онтології мають екземпляри, але спільно їх не використовують, і багато значень слотів містять текст, кращим вибором може бути GLUE. Якщо

тільки частини онтологій повинні відображатися, можна було б вибирати інструмент ONION. Якщо онтології мають дуже обмежену структуру, а концепти мають докладні визначення на однаковій природній мові, інструментальні засоби ISI/USC можуть забезпечувати кращі відповіді. Якщо екземпляри взагалі не доступні, і онтології містять багато відношень між концептами, Prompt може працювати найкраще.

Висновки

На даний час важливою дослідницькою темою в багатьох предметних областях є онтології, яка стрімко розвивається з кожним днем, місяцем, роком.

1. Borgida A., Brachman R., McGuiness D., Resnick L. Classic: A structural data model for objects // ACM SIGMOID Int. Conf. on Management of Data, Portland, Oregon, USA, ─ 1989.
2. Brachman R., Schmolze J. An Overview of the KL-ONE Knowledge Representation System // Cognitive Science, ─ Vol. 9, ─ No.2, ─ 1985.─ P.171-216.
3. Bray T., Paoli J., Sperberg C. Extensible Markup Language (XML) 1.0. W3C Recommendation. ─ Feb 1998. ─ http://www.w3.org/TR/RECxml.
4. OKBC: A Programmatic Foundation for Knowledge Base Interoperability. V. Chaudhri, A. Farquhar, R. Fikes P. Karp J. Rice // Fifteenth National Conf. on Artificial Intelligence. AAAI98, AAAIPres/The MIT Press, Madison. ─ 1998─ P.600-607.
5. The Description Logic Handbook. Theory, Implementation and Applications. Edited by F. Baader, D. Calvanese, D. McGuinness, D. Nardi, Peter Patel-Schneider, Cambridge ─ 2007 ─ 574 pages.
6. OIL in a Nutshell. Fensel D., Horrocks I., Harmelen F., Decker S., Erdmann M, Klein M. // 12th Intern. Conf. on Knowledge Engineering and Knowledge Management EKAW2000, Juanles-Pins, France. ─ 2000. ─ fensel00OIL.pdf
7. Fikes M. Knowledge Interchange Format // Technical Report. Computer Science Department. Stanford University. Logic-92-1. ─ 1992.
8. Gruber T. A translation Approach to Portable Ontology Specifications // Knowledge Acquisition Journal, vol. 5 – 1993. – P. 199-220. – KSL-92-71.ps
9. Heflin J. Hendler J. Dynamic ontologies on the web // In Proc. of American Association for Artificial Intelligence Conference, Menlo Park, CA, AAAI Press.─ 2000.
10. Karp R., Chaudhri V., Thomere J. XOL: An XML-Based Ontology Exchange Language // July 2004.
11. Luke S., Heflin J. SHOE 1.01. Proposed Specification. SHOE Project. ─ February 2000. ─ http://www.cs.umd.edu/projects/plus/SHOE/spec1.01.htm
12. Lassila O., Swick R. Resource Description Framework (RDF) Model and Syntax Specification. W3C Proposed Recommendation, – January 1999 – http://www.w3.org/TR/PR-rdf-syntax.
13. MacGregor R. Inside the LOOM classifier // SIGART bulletin, ─ 1991. ─ Vol.3, No.2, ─ P.70-76.
14. Noy N., Musen M. The PROMPT Suite: Interactive Tools For Ontology Merging And Mapping Stanford Medical Informatics, Stanford University. ─ August 2003. ─ SMI-2003-0973.pdf.