УДК 681.3

О.М. Овдій, Г.Ю. Проскудіна

Онтології у контексті інтеграції інформації: представлення, методи та інструменти побудови

Резюме

У даному огляді розглядається використання онтологій для підтримки задач інтеграції в семантично гетерогенних інформаційних системах. Представлені основні поняття та визначення онтологій, причини та приклади побудови. Розглядаються моделі і мови для представлення онтологій та використання відображень між ними. Досліджуються методи й інструментальні засоби для інженерії онтологій і підтримки інтеграції інформації. Також наводяться приклади візуалізації, побудови та злиття двох онтологій за допомогою сучасних інструментів.

Вступ

Всесвітня павутина Інтернет стрімко входить буквально в усі сфери життя. Інтернет стає усе більш могутнім і важливим джерелом інформації. Засобам обробки даних у мережі усе складніше і складніше справлятися з лавиною інформації, що вже існує і, що додається в мережу щодня. Крім того, дані в Інтернет організовані вкрай стихійно і не систематично. Для ефективного спільного використання інформації повинно бути вирішено багато технічних проблем. Проблема з'єднання гетерогенних і розподілених комп'ютерних систем відома як проблема інтероперабельності (interoperability problem). Інтероперабельноіть потрібно забезпечувати і на технічному і на інформаційному рівні. Проблеми, що могли б виникати внаслідок гетерогенності даних, уже відомі в межах співтовариства розподілених систем баз даних: *структурна гетерогенність* (схематична гетерогенність) і *семантична гетерогенність* (гетерогенність даних) [KS 96]. 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. Семантичні конфлікти відбуваються всякий раз, коли два контексти не використовують однакову інтерпретацію інформації. Можна виділити наступні причини семантичної гетерогенності:

- *Конфлікти змішання* відбуваються, коли інформаційні елементи, здається, мають однакове значення, але відрізняються в дійсності, наприклад унаслідок різних часових контекстів.
- *Конфлікти масштабування* відбуваються, коли для виміру значення використовуються різні довідкові системи. Приклади - різні валюти.
- *Конфлікти іменування* відбуваються, коли значно відрізняються схеми позначення інформації. Часте явище - присутність омонімів і синонімів.

Використання *онтологій* для пояснення неявного і схованого знання - можливий підхід перебороти проблему семантичної гетерогенності.

Отже, мета даної роботи була:

1. Зрозуміти, що означає термін “онтологія”.
2. Дослідити призначення отнологій і таким чином зрозуміти, як використовувати їх для проблем, що пов’язані з інтеграцією інформації.
3. Ознайомитися з сучасним станом технології, зокрема:
   - основними представленнями або моделями сучасних онтологій;
   - методами та інструментальними засобами для підтримки процесу побудови та використання онтологій.
4. Мати уявлення про потенційне використання онтологій.

1. Що таке онтологія?

Походження, історія розвитку та основні визначення. Термін онтологія прийшов з філософії (походить зі спроби Аристотеля класифікувати предмети у світі), де він використовується для позначення системи знань, що відносяться до навколишнього світу (у противагу системі знань про внутрішній світ людини). Іншими словами, онтологія - це наука про буття, наука про природу речей і взаємозв'язки між ними. У контексті інформаційних технологій представлення знань, терміном онтологія можна визначити деякий механізм, спосіб, що використовується для опису деякої області знань (предметної області), зокрема базових понять цієї області, їхніх властивостей та зв'язків між ними.

Існує безліч визначень онтології, деякі з яких суперечать один одному, але найбільш широко використовуємим є визначення [GR 93]: “Онтологія – це явна специфікація концептуалізації”. Тут

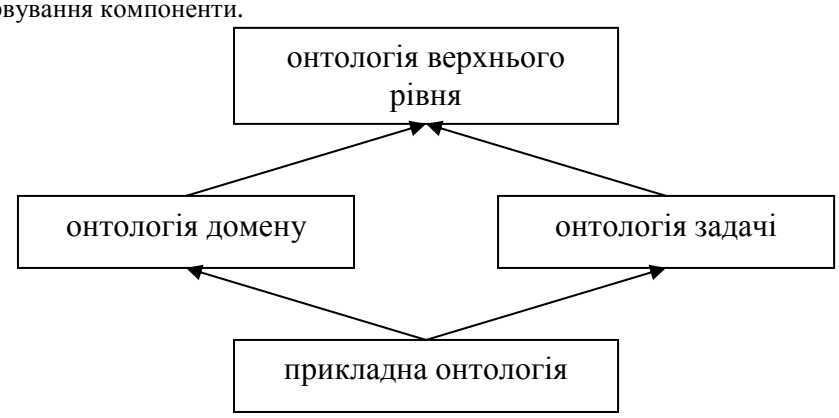
концептуалізація означає абстрактне представлення предметної області. Поширене також визначення [UG 96]: “Онтологія – спільне розуміння деякої області інтересу”.

Слово “онтологія” часто сприймається майже як синонім інженерії знань у штучному інтелекту, концептуальному моделюванню у базах данных і моделюванню предметної області (домену) в об’єктно-орієнтованому проектуванні.

На сьогодні під онтологією можна розуміти [MS 02]:

- надійний семантичний базіс у визначенні змісту;
- загальну логічну теорію, що складається зі словника та набору тверджень на деякій мові логіки;
- основу для комунікації між людьми та комп’ютерними агентами.

Застосовувані в контексті розробки інтелектуальних систем, онтології були класифіковані на *прикладні онтології*, *онтології домену*, *онтології задачі* і *онтології верхнього рівня*, що охоплюють різні аспекти, доречні в моделюванні інтелектуальних систем [G 98] (Мал. 1). Онтології верхнього рівня описують дуже загальні концепти, наприклад, місце, час, речовина, об’єкт, подія, дія і т.п, що є незалежними від конкретної проблеми або домену: тоді, здається , принаймні теоретично, розумно мати уніфіковані онології верхнього рівня для веких спільнот користувачів. Онтології домену та онтології задач віповідно описують словник, що відносяться до типової області (наприклад, медицина, автомобілізм) або типової задачі або діяльності (наприкдад, діагностика, продаж) за допомогою спеціалізації термінів, представлених в онтології верхнього рівня. Прикладні антології описують концепти, що залежать і від онтології задач і від онтології домену. Ці концепти часто відповідають ролям, що грають сутності домену під час виконання якоїсь діяльності, наприклад, заміни частини або резервування компоненти.



Мал. 1. Види онтологій, згідно з їх рівнем злежності від конкретної задачі або точки зору

Наприкінці дев'яностих років досить обмежена роль і вплив онтологій, разюче змінилося розумінням, що концептуальна, усе ще здійсненна модель прикладного домену забезпечує істотне додаткове значення для усіх видів прикладних сценаріїв подібно керуванню знаннями або електронній комерції. Очевидно, головний поштовх онтологіям, дало передбачення SemanticWeb [BLHL 01]. 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 "перекинути місток" між новими поняттями, з якими система ще не зустрічалася, і описами уже відомих класів, відносин, властивостей і об'єктів реального світу.

Мотивація. Приведемо деякі причини розробки онтологій [NM 01]:

- спільне використання загального розуміння структури інформації серед людей або програмних агентів;
- можливість повторного використання знань домену;
- можливість робити явними припущення домену;
- відокремлення знань домену від операційного знання;
- аналіз знань домену.

Спільне використання загального розуміння структури інформації серед людей або програмних агентів - одна з найбільш загальних цілей у розробці онтологій [UG 96]. Наприклад, припустимо, що декілька різних Інтернет-сторінок містять медичну інформацію або надають медичні 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 було однією з рушійних сил недавньої хвилі 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.

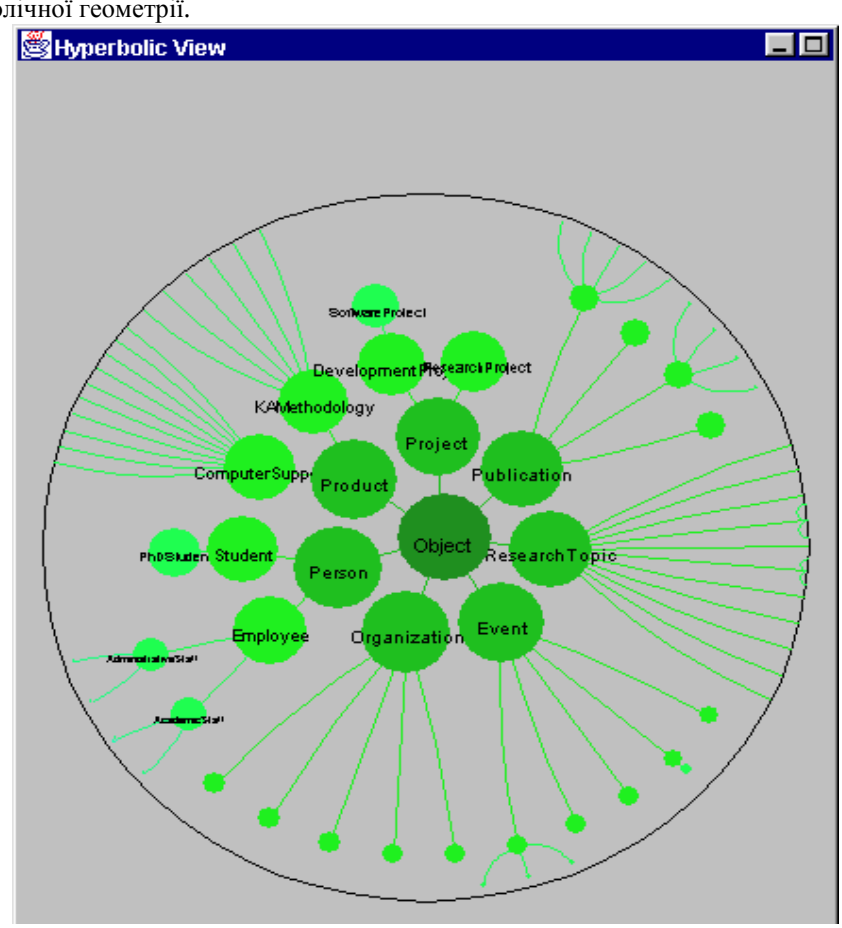
Зробити явними припущення домену, які лежать в основі реалізації, робить можливим легко змінити ці припущення, якщо наше знання про домен змінюється. Тверде кодування припущень про світ у коді мови програмування робить ці припущення не тільки важкими для знаходження і розуміння але також і важкими для зміни, особливо для когось без досвіду програмування. Крім того, явні специфікації знання домену корисні для нових користувачів, які повинні вивчити який терміни що означає.

Відділення знання домену від операційного знання - інше звичайне використання онтологій. Ми можемо 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 Формальний аналіз термінів надзвичайно цінний коли робиться спроба повторно використовувати існуючі онтології і їх розширення.

Застосування та приклади. Онтології довели, що можуть бути істотними елементами багатьох прикладних програм. Вони використовуються в агентних системах, системах керування знаннями і електронній комерції. Вони можуть також поєднувати інтелектуальну інформацію, забезпечувати заснований на семантиці доступ в Інтернет, і витягати інформацію з текстів на додаток до того, що вони використовуються в багатьох інших прикладних програмах для явного визначення знань, укладених у них. Однак, онтології корисні не тільки для прикладних програм, у яких знання відіграє ключову роль, але вони можуть також викликати значну зміну в поточному змісті Web у відповідності до концепції SemanticWeb. Прикладами використання онтологій в Інтернет можуть бути сайти електронної комерції, пошукові машини, Web-сервіси. Деякі приклади існуючих онтологій наведено в Додатку А.

Для більш повного уявлення того, що є онтологія, наведемо на наш погляд цікавий приклад підходу до візуалізації онтологій [LR 96]. У процесі роботи з онтологіями користувачу часто необхідний стислий огляд усієї ієрархії, щоб був можливий швидкий і легкий перехід від одного класу ієрархії до іншого. Ці вимоги задовольняються схемою представлення, заснованою на гіперболічній геометрії Ця методика візуалізації дозволяє швидкий перехід до класу, який знаходиться далеко від центру, також як і більш детальне дослідження класів та їх оточення. На Мал. 2 представлено зображення наукової онтології KA^2 [BFDG 99] за допомогою гіперболічної геометрії.



Мал.. 2. Приклад візуалізації онтології за допомогою гіперболічної геометрії.

2. Представлення онтологій

Компоненти, з яких складаються онтології, залежать від парадигми представлення. Але практично всі моделі онтологій у тім чи іншому ступені містять *концепти* (інші варіанти: поняття, класи, сутності), *властивості* концептів (інші варіанти: слоти, атрибути, ролі), *відношення* між концептами (інші варіанти: зв'язки, залежності, функції) та додаткові *обмеження* (що визначаються аксіомами, у деяких парадигмах фасетами).

Концепти [SAL 99] використовуються в широкому змісті. Вони можуть бути абстрактними чи конкретними, елементарними чи складеними, реальними чи фіктивними. Іншими словами, концептом може бути що-небудь, до чого щось стверджується і, тому, могло б також бути описом задачі, функції, дії, стратегії, процесу міркування і т.д.

Концепти в онтології зазвичай організовані в таксономії. Іноді таксономії розглядаються як повні онтології, хоча не можна обмежувати онтології до цього. Таксономії широко використовуються для організації онтологічного знання предметної області, використовуючи відношення узагальнення/спеціалізації, через які могло б застосовуватися одиничне/множинне спадкування.

Відношення представляють тип взаємодії між концептами предметної області. Прикладами бінарних відношень можуть служити "part-of " і "connected-to".

Аксіоми використовуються для моделювання тверджень, що завжди є істинними. Вони можуть бути включені в онтологію для декількох цілей, таких як обмеження інформації, що міститься в онтології, перевірка правильності або вивід нової інформації.

Термін екземпляр використовується для представлення елементів у предметній області, тобто елемента даного концепту. Онтологія разом з множиною окремих *екземплярів* складає *базу знань*.

Слід відмітити, що на сьогодні відкритим залишається питання, щодо ясного розмежування онтології і бази знань. Оскільки екземпляри самі можуть бути концептами і часто при моделюванні треба вирішувати, що відносити до концепту, а що до екземпляру, а що одночасно до концепту і екземпляру. Тому деякими авторами, наприклад, [GR 93], екземпляр включено до елементів формалізації.

Як було сказано вище, онтології складаються з ієрархічних описів важливих концептів у домені разом з описами властивостей кожного концепту. Ступінь формалізму, що застосовується в зафіксованих описах може бути мінлива, змінюючись від природної мови до логічних формалізмів, але очевидно, що зростання формалізму та регулярності полегшує машинну інтерпретацію інформації.

Моделі онтологій можуть бути класифіковано наступним чином:

- прості моделі (наприклад, що мають лише концепти),
- на основі фреймів (мають лише концепти і властивості),
- на основі логік (наприклад, Ontolingua, DAML+OIL).

Неформальні лінгвістичні моделі часто використовуються для специфікації онтологій. У таких моделях онтологічні концепти визначені у відповідності з вербальними визначеннями, подібно толковому словнику. Між концептами можуть бути встановлені деякі види найважливіших зв'язків. Глосарій термінів у деякій предметній області, тезаурус зі своїми поняттями (концептами) і зв'язки, що визначають терміни природної мови можуть розглядатися як онтології. Для встановлення зв'язку між вербально визначеними концептами, а також пошуку концептів релевантних запиту використовуються методи добування інформації.

На відміну від вербальних моделей, онтології, що основані на логіках формально визначені і мають здатність до формального судження. Однією з перших серед запропонованих моделей, була Ontolingua [GR 92, FFR 96].

Ontolingua - мова, яка заснована на KIF [FIK 92] і на FrameOntology [GR 93]. 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. Однак, FrameOntology, що сформована поверх KIF, дозволяє визначати онтологію відповідно з парадигмою фреймів, включаючи такі терміни як клас, екземпляр, підклас і т.д. Але вона в той же час є менш виразною ніж KIF. Ontolingua дозволяє включати у визначення вирази KIF, засновані на FrameOntology. Ontolingua була призначена як проміжна мова для взаємодії гетерогенних онтологій.

Онтологічна модель OKBC (Open Knowledge Base Connectivity) [CAL 98], 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 OKBC служить як проміжна мова онтології, що підтримує зв'язок, використовуючи OKBC.

Модель XOL (XML-based Ontology Exchange Language) [KCT 99] 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.

FLogic (Frame Logic) [KLW 95] - мова, що інтегрує мови засновані на фреймах і числення предикатів першого порядку. Вона враховує такі структурні аспекти об'єктно-орієнтованих і заснованих на фреймах мов як тотожність об'єкта, комплексні об'єкти, спадкування, поліморфні типи, методи запиту, інкапсуляцію та ін. У певному розумінні, FLogic знаходиться в такому ж відношенні до об'єктно-орієнтованої парадигми як класичне числення предикатів до реляційного програмування.

Інший клас формальних онтологічних моделей ґрунтується на різних видах дескриптивних логік (DL) [DLH03]. 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 принаймні одна інтерпретація відповідно до якої концепт відповідає непорожній множині. Категорізація між концептами означає, що для будь-якої інтерпретації множина, що інтерпретує один концепт, є підмножиною множини, що інтерпретує інший концепт. Доказ категорізації зводиться до такої здійсненності.

Першою серед систем DL була KL-ONE [BS 85], де заявлялось про перехід від семантичної мережі до більш обґрунтованим термінологічним (дискриптивним) логікам. Вплив був дуже суттєвим, тому KL-ONE розглядається як родоначальник цілого сімейства мов. У KL-ONE представлені найбільш ключові поняття, досліджувані в наступних роботах по DL, наприклад, поняття концептів і ролей та їх взаємодія, важливі ідеї по "обмеженню значень" та "числовому обмеженню", що змінили використання ролей у визначеннях концептів та найбільш важливе виведення категорізації та класифікації.

Ідея створення CLASSIC [BBMR 89] базувалась 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 [DLH03]. Але на сьогодні CLASSIC широко використовується різними системами, наприклад, OBSERVER [MIKS 96].

LOOM [MG 91] 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 домагається 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 00] - система, що забезпечує підтримку міркування для проектування онтологій, інтеграції і верифікації. FaCT – DL класифікатор, що може також використовуватися 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 зоснована на CORBA клієнт-серверна архітектура. Оптимізації FaCT, що націлені на покращення продуктивності системи при класифікації реалістичних онтологій, зробили її найбільш швидкодіючою порівняно з попередніми DL-системами.

Найбільш значимий розвиток онтологічних моделей на основі логік відбувається в рамках розробки Semantic Web. Звернемо увагу на такі Web стандарти, як XML і RDF.

XML (eXtensible Markup Language) [BPS 98] 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) [LS 99] 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) [MM 03], 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.

Далі розглянемо мови, засновані на цих стандартах.

OIL (Ontology Interchange Language) [FHH 00], 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.

DAML+OIL (DARPA Agent Markup Language+OIL) [H 02] була розроблена спільним комітетом з США і Європейського Об'єднання (IST) у контексті DAML, проекту DARPA для надання семантичної інтероперабельності в XML. Вона являє собою результат злиття мов DAML і OIL і сформована на ранніх стандартах W3C таких як RDF і RDF Schema, розширюючи їх більш багатими примітивами моделювання. Ця модель була запланована як основа для Web Ontology Language (OWL).

OWL розширює RDF і RDF Schema, забезпечуючи поряд з формальною семантикою додатковий словник, що закладено в основу DAML + OIL, а також убудовану 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) [LH 00], 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.

3. Методології проектування онтологій

Попередні розділи забезпечили інформацію про використання та мови онтологій. Але важливо є й підтримка процесу розробки онтологій. У цьому розділі, ми опишемо, які методології забезпечують підтримку процесу інженерії онтологій.

Процес побудови або розробки онтологій, що використовуються в інформаційних системах, залишається скоріше мистецтвом ніж наукою, не існує одного правильного способу або методології для розробки онтологій. Ця ситуація може бути змінена тільки через розуміння того, як приступити до побудови онтологій, тобто необхідна гарна методологія для розробки онтологій. До теперішнього моменту з'явилося декілька пропозицій для такої методології, де люди відбивають свій досвід у побудові онтологій. Приведемо найбільш багатообіцяючі з них.

METHONTOLOGY [GFV 96, FGJ 97] була розроблена в Лабораторії штучного інтелекту Мадридського Політехнічного Університету. METHONTOLOGY починається з ідентифікації наступних дій, які включені в розробку онтологій: специфікації, збору знання, концептуалізації, інтегрування, реалізації, оцінки, документування. Цикл життя онтології, заснований на удосконаленні прототипу, дозволяє побудову онтологій на рівні знань і включає: ідентифікацію процесу розробки онтології, життєвий цикл, заснований на розвитку прототипів і приватних методів виконання кожної дії. METHONTOLOGY є однією з найбільш повних; однак, потрібні рекомендації для процесів попереднього розвитку, і більш докладно повинні бути визначені деякі дії і методи.

Методологія TOVE [GF 95] заснована на досвіді проектування TOVE (Toronto Virtual Enterprise). Вона головним чином, включає побудову логічної моделі знання, що повинна визначатися через онтологію. Ця модель не створюється безпосередньо. Спочатку, виконується неформальний опис зі специфікацій, яким повинні задовольняти онтології, а потім цей опис формалізується. У цій методології немає вказівок на життєвий цикл. Крім того, хоча були розроблені онтології по цій методології, і є прикладні програми, що використовують ці онтології, область їхнього використання обмежується тільки бізнесом. Підхід TOVE найбільш цікавий для

виділення оцінки онтології, особливо, оскільки засоби виконання цієї оцінки забезпечуються у формі теорем повноти. Ці теореми корисні в багатьох задачах підтримки онтологій.

Методологія на основі SENSUS. Ця Онтологія використовується в обробці природної мови і розроблена в ISI (Інститут Інформатики) групою природної мови для забезпечення всеосяжної концептуальної структури розробки машинних трансляторів [KCP 95]. Дійсний зміст був отриманий, витягаючи і з'єднуючи інформацію з різних електронних джерел. Для створення основи онтології процес почався з ручного з'єднання моделі верхнього рівня PENMAN і ONTOS (дві, дуже високорівневі лінгвістичні онтології) і семантичних категорій зі словника. Потім з онтологічною базою була з'єднана (знову вручну) онтологія WordNet. Засіб з'єднання потім використовувався для злиття WordNet з Англійським словником. Після цього, для підтримки машинного перекладу, результат цього злиття, був поповнений Іспанськими і Японськими лексичними входами з Іспано-Англійського словника Коллинза і Японо-англійського словника Kenkyusha. SENSUS має більше 50,000 концептів, що організовані в ієрархії у відповідності з рівнем абстракції. Включає терміни з високим і середнім рівнем абстракції, але не охоплює терміни зі спеціальних доменів.

CommonKADs[SAL 94] є широко використовуваною методологією для розробки систем баз знань, у яких онтології відіграють важливу роль. Проект KACTUS [SWJ 95] був наступним проектом, що зосередився на проблемі проектування онтологій. Дана методологія підкреслює проектування, перепроектування і багаторазове використання модулів. Онтологія створюється з бібліотеки онтологій маленького масштабу, що вимагає відображення між різним онтологіями, включеними в розробку нової онтології. Вибір релевантних онтологій з бібліотеки підтримується схемою індексації. Такий підхід до розробки онтологій обумовлений розробкою прикладних програм. Так, щораз, коли будується прикладна програма, будується й онтологія, що представляє знання, необхідні для створення прикладної програми. Ця онтологія може бути розроблена з застосуванням повторного використання інших онтологій і може також інтегруватися в онтології наступних прикладних програм.

Методологія [UK 95] заснована в значній мірі на досвіді розробки Enterprise ontology. Даний підхід розрізняє неформальні та формальні стадії побудови онтологій. Неформальна стадія стосується визначення ключових концептів і дає текстові визначення концептів і відношень. Оскільки рекомендується використання існуючих методів збору знання для цієї стадії, не даються поради про те, як ідентифікувати онтологічний концепт.

Проект Plinius [MAL 94] є спробою напівавтоматичного добування знання з текстів природною мовою. Онтологія Plinius була розроблена для підтримки трансляції речень природної мови у вирази мови представлення знань. Проектні вирішення, прийняті при розробці онтології й удавані незалежними від предметної області і були запропоновані як загальні принципи проектування онтологій

Мотивом для методології ONIONS (ONtologic Integration Of Naive Sources) [GSG 96] послужила проблема інтегрування знань, тобто як інтегрувати гетерогенні джерела інформації в придбанні знань. Ця проблема звернена до створення формальної онтології предметної області інтегруванням існуючих архівів знань. Один з найбільш відмінних аспектів підходу ONIONS - метод придбання онтології. Він робить неформальну онтологію, схематичну оцінку концептуалізації предметної області.

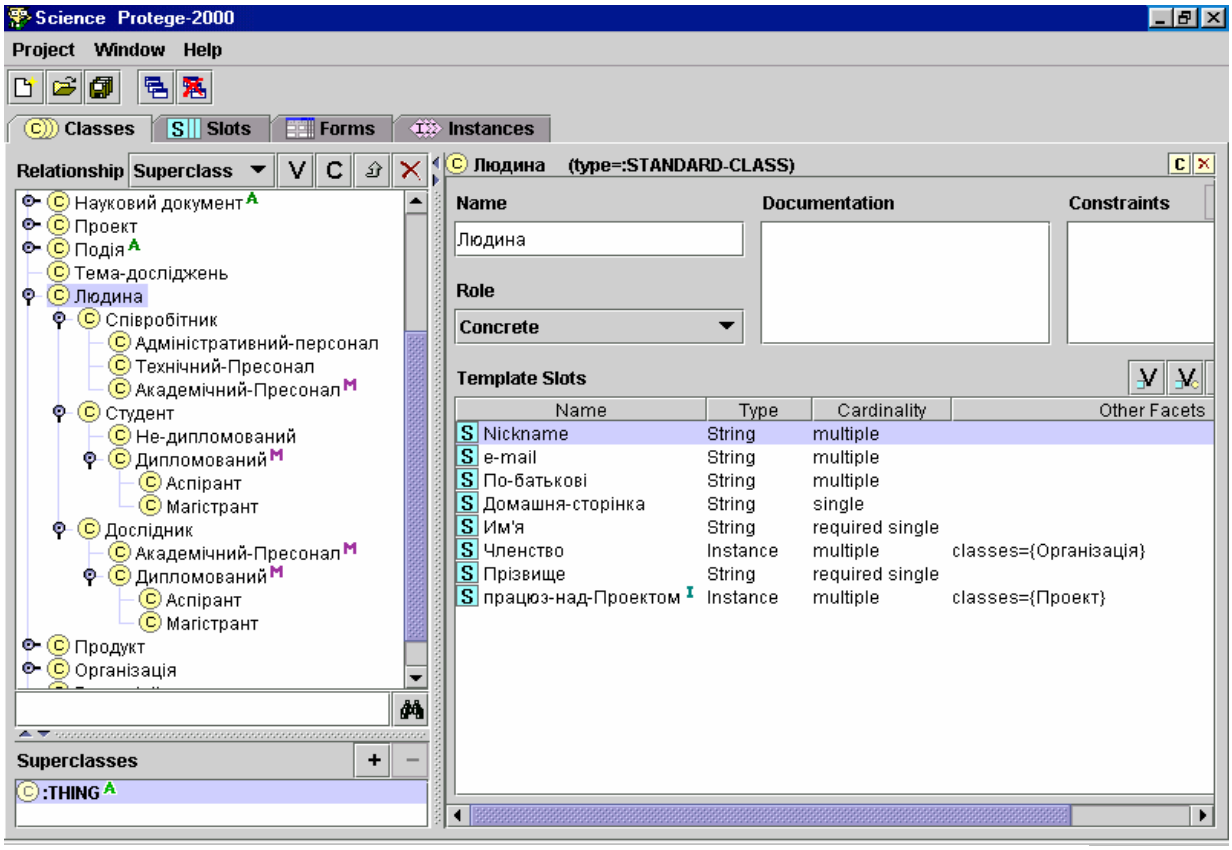
4. Інструменти для інженерії онтологій

Для допомоги користувачам в інженерії і підтримці онтологій розроблено ряд інструментальних засобів. Розглянемо деякі з них [DSWKB 99, WVV 01].

WebOnto [DOM 98] був розроблений Knowledge Media Institute of the Open University. WebOnto призначений для підтримки спільного перегляду, створення і редагування онтологій. Він був розроблений для доповнення Tadzebao - інструмента дослідження онтологій. WebOnto головним чином є графічно-орієнтованим інструментом для побудови онтологій. Для моделювання онтологій у ньому використовується мова OCML (Operational Conceptual Modeling Language) розроблена у контексті проекту VITAL [MOT 97]. Інструмент має ряд корисних особливостей: збереження структурних діаграм, окремий перегляд відношень, класів, правил і т.д. Інші можливості включають, наприклад, спільну роботу над онтологією, використання діаграм та функцій передачі і прийому.

Protege [MUS 98] – вільно розповсюджуєма локальна Java програма на базі Windows, що розроблена групою медичної інформатики Стенфордського університету. Програма призначається для побудови (створення, редагування та перегляду) онтологій моделей прикладної області. Її ціль - допомогти розроблювачам програмного забезпечення у створенні і підтримці явних моделей предметної області і у включенні цих моделей безпосередньо в код програми. Protege складається з трьох головних частин, що повинні використовуватися по черзі. Спочатку йде редактор онтологій, що дозволяє проектувати онтології розвертаючи ієрархічну структуру і включаючи абстрактні або конкретні класи і слоти. Ґрунтуючись на сформованій онтології, Protege здатний генерувати інструмент придбання знань для введення екземплярів онтології. Остання частина програми - інтерпретатор схем, який дозволяє робити екземпляри для класів і підкласів. Інструмент має повний графічний інтерфейс, що є дуже зручним для використання недосвідченими користувачами.

Наведемо приклад застосування Protege для роботи з онтологією Science (http://protege.stanford.edu/ontologies/ontologyOfScience/Science.zip), що є модифікацією онтології $КА^2$ [BFDG99], яка моделює спільноту придбання знань, її дослідників, теми, продукти та інш. (Мал. 3).



Мал. 3. Приклад застосування Protege для роботи з онтологією.

OntoSaurus [MG 91] є web-браузером для баз знань LOOM, він забезпечує графічний інтерфейс до них. OntoSaurus також забезпечує обмежені засоби редагування, його головна функція - перегляд онтологій.

ODE (Ontological Design Environment) [FGP 99] - інструмент конструювання онтологій, що взаємодіє з користувачами на концептуальному рівні, на відміну від інструментальних засобів, подібно OntoSaurus, що спілкуються на символьному рівні. Мотивом для ODE служить те, що людям набагато краще формулювати онтології на концептуальному рівні. ODE забезпечує користувачів набором таблиць для заповнення (концептів, атрибутів, відношень і т.д.) і автоматично генерує для них код, у даний час у Ontolingua [FFR 96] і F-Logic [KLW 95]. ODE складає частину методології для повного життєвого циклу побудови онтології відповідно до Methontology [GFV 96, FGJ 97].

KADS22 [SAL 94] - інструмент для підтримки проектування моделей знання згідно CommonKADS методології. Онтології утворюють частину з таких моделей знань (інша частина є моделлями виводу). CommonKADS моделі визначені в CML (Conceptual Modeling Language). Він забезпечує інтерактивний графічний інтерфейс для CML із наступними функціональними можливостями: синтаксичним аналізом файлів CML, друком, переглядом гіпертексту, пошуком, генеруванням глосарія і генеруванням HTML.

OntoEdit [SM 00]: Цей інструмент дає можливість інспектувати, переглядати, кодувати і модифікувати онтології і використовувати ці можливості для підтримки задач розробки і підтримки онтологій. На сьогодні OntoEdit підтримує такі мови представлення, як FLogic, включаючи механізм виводу, OIL, розширення Karlsruhe RDF Scheme і внутрішню, засновану на XML серіалізацію моделі онтології з використанням OXML.

SHOE's Knowledge Annotator [HH 00]: За допомогою цього інструмента, користувач може описувати зміст Web-сторінки. Knowledge Annotator має інтерфейс, що відображає екземпляри, онтології і твердження (зібрані документи). Інструмент також забезпечує перевірку цілісності. Анотуємі Web-сторінки можуть бути проаналізовані іншим інструментом по імені Expos'e, а зміст буде збережено в репозитарії. Це SHOE-знання потім зберігається в базі знань Parka [STO 97].

Подальший розвиток у межах проекту DWQ веде до інструмента, названого i·com [FN 00]. i·com - інструмент підтримки для концептуальної стадії дизайну. Цей інструмент використовує розширену (EER) модель даних, і збагачує її обмеженнями агрегації і міжсхем. i·com не забезпечує методологію, і при цьому це не інструмент анотації, він служить головним чином для інтелектуального концептуального моделювання.

5. Інструменти для об'єднання і відображення онтологій

Сьогодні онтології доступні в багатьох різних формах. Але, що користувач повинен робить, коли він знаходить кілька онтологій, які він хотів би використовувати, 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. У цьому розділі дається короткий огляд деяких з існуючих підходів.

Інструментальні засоби, що мають справу з виявленням відповідностей між онтологіями можна класифікувати наступним чином [NM 03]:

- 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 релевантних частин початкових онтологій (наприклад ONION)

Інший спосіб категорізувати інструментальні засоби по типу вхідних даних, на який покладається інструмент у своєму аналізі:

- імена класів і визначення природною мовою (наприклад, інструментальні засоби, розроблені в ISI/UCS для напівавтоматичного вирівнювання (alignment) онтологій домену у велику центральну онтологію [HOV 98])
- ієрархію класу (Chimaera)
- ієрархію класу, слоти, і фасети (iPrompt, AnchorPrompt, ONION)
- екземпляри класів (GLUE і FCA-Merge)
- опису класів (інструментальні засоби, засновані на дескриптивній логіці, такі як OBSERVER)

Розглянемо тепер вищезгадані засоби більш докладно.

Chimaera [MFRW 00] - інтерактивний інструмент для об'єднання, заснований на редакторі онтологій Ontolingua [FFR 96]. 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 [DMQ 02] 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. Потім механізм виведення зробить вивід, заснований на твердженнях у початкових онтологіях і аксіомах сполучення, у такий спосіб створюючи нові дані в об'єднаній онтології. OntoMerge надає інструменти для трансляції даних екземплярів в об'єднану онтологію.

Ontomorph [CHA 00] визначає набір операторів перетворення, що можуть застосовуватися до онтології. Потім людина-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. Однак, експерт не одержує ніякого керівництва за винятком початкового списку співпадаючих пар.

GLUE [DMDH 02] - 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 вірогіднісна модель. Учні покладаються на онтології, що мають екземпляри, і вони працюють набагато краще, коли більшість значень слотів містять текст, ніж посиланнь на інші екземпляри.

Система OBSERVER [MIKS 00] 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.

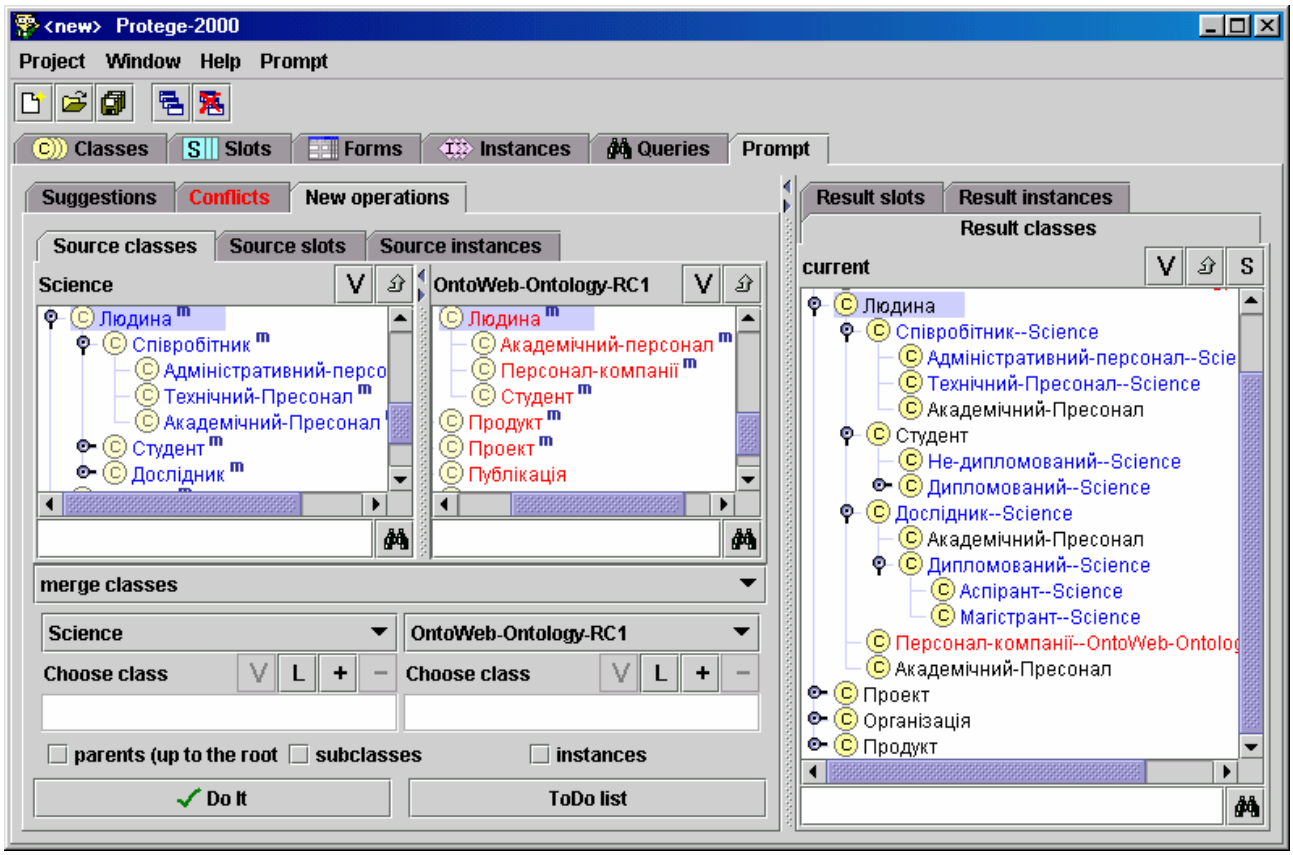
FCA-Merge [SM 01] - метод для порівняння онтологій, що мають набір спільних екземплярів або набір спільних документів, що анотуються за допомогою концептів з початкових онтологій. Ґрунтуючись на цій інформації FCA-Merge використовує математичні методи з Formal Concept Analysis для того щоб зробити решітку 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 [MWD 01] 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 [HOV 98] описує набір евристик, 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 слова останову, 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 подоби дає потенційні збіги, які користувач повинен дослідити і схвалити.

PROMPT [NM00, NM99] - 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.



Мал.4. Приклад злиття двох онтологій за допомогою Prompt

На Мал. 4. показано процес злиття двох онтологій, а саме концептів “Людина”, вищезгаданої онтології Science та OntoWeb Ontology, яка створена для проекту OntoWeb (http://ontoweb.aifb.uni-karlsruhe.de), що присвячений заснованому на онтологіях інформаційному обмінові для управління знаннями і електронної комерції.

Розмаїтість у типах інструментальних засобів робить складним їхнє безпосереднє порівняння. 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 виборати 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

Будучи на даний час важливою дослідницькою і прикладною темою в багатьох предметних областях, онтології складають поле дії, що стрімко розвивається і у дослідженні й у промисловості.

У даній роботі була зроблена спроба зробити огляд сучасного стану розвитку онтологій у контексті інтеграції інформації в семантично гетерогенних інформаційних системах. Представлені основні поняття та визначення онтологій, причини та приклади побудови. Розглядаються моделі і мови для представлення онтологій та використання відображень між ними. Досліджуються методи й інструментальні засоби для інженерії онтологій і підтримки інтеграції інформації. Також наводяться приклади візуалізації, побудови та злиття двох онтологій за допомогою сучасних інструментів.

Список посилань

[BLHL 01] *Berners-Lee T., Hendler J., Lassila O.* The Semantic Web. Scientific American, May 2001.

[BFDG 99] *Benjamins V. Fensel D, Decker S., Gemez-Perez A.* (KA)2: Building Ontologies for the Internet // a Mid Term Report. ─ 1999.

[BBMR 89] *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.

[BS 85] Brachman R., Schmolze J. An Overview of the KL-ONE Knowledge Representation System // Cognitive Science, ─ Vol. 9, ─ No.2, ─ 1985.─ P.171-216.

[BPS 98] *Bray T., Paoli J., Sperberg C.* Extensible Markup Language (XML) 1.0. W3C Recommendation. ─ Feb 1998. ─ http://www.w3.org/TR/RECxml.

[MM 03] *Manola F., Miller E.* RDF Primer. Oct.2003. ─ http://www.w3.org/TR/rdf-primer/.

[CAL 98] 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.

[CHA 00] *Chalupsky H,* OntoMorph: A translation system for symbolic knowledge // In: Cohn A. G., Giunchiglia F., Selman B. (Eds.), Principles of Knowledge Representation and Reasoning // Proc. of the Seventh Intern. Conf. (KR2000). Morgan Kaufmann Publishers, San Francisco, CA.

[DLH 03] The Description Logic Handbook. Theory, Implementation and Applications. Edited by F. Baader, D. Calvanese, D. McGuinness, D. Nardi, Peter Patel-Schneider, Cambridge ─ 2003 ─ 574 pages.

[DMDH 02] *Doan A., Madhavan J., Domingos P., Halevy A.* Learning to map between ontologies on the Semantic Web // In: The Eleventh Intern. WWW Conference. Hawaii, US. ─ 2002. ─ doan02learning.pdf

[DMQ 02] *Dou D., McDermott D., Qi P.,* Ontology translation by ontology merging and automated reasoning // EKAW'02 workshop on Ontologies for Multi-Agent Systems. SigËuenza, Spain. ─ 2002.

[DOM 98] *Domingue J.* Tadzebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web // Proc. of the Eleventh Workshop on Knowledge Acquisition, Modeling and Management, KAW'98, Banff, Canada ─ 1998.

[DSWKB 99] Wondertools? A comparative study of ontological engineering tools. A. Duineveld, R. Stoter, R. Weiden, B. Kenepa, V. Benjamins // Proc.of the 12th Workshop on Knowledge Acquisition, Modeling and Management, Ban, Canada – 1999. – P. 4.6.1-4.6.20. – wondertools.pdf http://sern.ucalgary.ca/KSI/KAW/KAW99/papers/Duineveld1/wondertools.pdf

[FFR 96] *Farquhar A., Fikes R., Rice J.* The Ontolingua Server: A Tool for Collaborative Ontology Construction // In Proc. of KAW96. Banff, Canada. ─ 1996.

[FHH 00] 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

[FGJ 97] *Fernández M., Gomez-Perez A., Juristo N.* METHONTOLOGY: From Ontological Art Towards Ontological Engineering // AAAI-97 Spring Symposium on Ontological Engineering, Stanford University, ─ 1997. ─ sss97.ps

[FGP 99] *Fernández M, Gómez-Pérez A., Pazos J.* A Building a Chemical Ontology Using Methondology and the Ontology Design Environment // IEEE Intelligent Systems, ─ January/February 1999. – P. 37-46. ─ IS1999-x1037.pdf.

[FIK 92] *Fikes M.* Knowledge Interchange Format // Technical Report. Computer Science Department. Stanford University. Logic-92-1. ─ 1992.

[FN 00] *Franconi E., Ng G.* The i.com tool for intelligent conceptual modelling // 7th Intl. Workshop on Knowledge Representation meets Databases, KRDB'00, Berlin, Germany, ─ August 2000. ─ franconi00icom.pdf

[GF 95] *Gruninger M., Fox M.* Methodology for the Design and Evaluation of Ontologies // Proc. Workshop on Basic Ontological Issues in Knowledge Sharing, IJCAI-95, Montreal, ─ August 1995. ─ grninger95methodology

[GFV 96] *Gomez-Perez A., Fernandez M., Vicente A.* Towards a Method to Conceptualize Domain Ontologies // ECAI-96 Workshop on Ontological Engineering, Budapest. ─ 1996. ─ ECAI96.pdf.

[GR 92] *Gruber T.* Ontolingua: a Mechanism to Support Portable Ontologies. Knowledge Systems Laboratory of Stanford University, ─ Jun 1992. ─ gruber92ontolingua.pdf.

[GR 93] *Gruber T.* A translation Approach to Portable Ontology Specifications // Knowledge Acquisition Journal, vol. 5 – 1993. – P. 199-220. – KSL-92-71.ps

[GSG 96] *Gangemi A., Steve G., Giacomelli F.* ONIONS: An Ontological Methodology for Taxonomic Knowledge Integration // ECAI-96 Workshop on Ontological Engineering, Budapest, ─ August 1996. ─ gangemi96onions.pdf

[G 98] *Guarino N.* Formal Ontology and Information Systems // National Research Council, Proceedings of FOIS'98, Trento, Italy, ─ June 1998. ─ Amsterdam, IOS Press, ─ P. 3-15. ─ Guarino98-formal.pdf

[HH 00] *Heflin J. Hendler J.* Dynamic ontologies on the web // In Proc. of American Association for Artificial Intelligence Conference, Menlo Park, CA, AAAI Press.─ 2000.

[H 02] *Horrocks I.* DAML+OIL: a Description Logic for the Semantic Web // Bulletin of the IEEE Comp. Society Technical Committee on Data Engineering. ─ 2002. ─ Vol.25, No.1, March. ─ P.4-9.

[HOV 98] *Hovy E.* Combining and standardizing largescale, practical ontologies for machine translation and other uses. // First Intern. Conf. on Language Resources and Evaluation, LREC. Granada, Spain, ─ 1998. ─ P. 535–542.

[KCT 99] *Karp R., Chaudhri V., Thomere J.* XOL: An XML-Based Ontology Exchange Language // July 1999.

[KS 96] *Kashyap V., Sheth A.* Semantic and schematic similarities between database objects: a context-based approach // The VLDB Journal. ─ 1996. ─ No.5. ─ P.276–304. ─ kashyap96semantic.pdf

[KLW 95] *Kifer M., Lausen G., Wu J.* Logical Foundations of Object-Oriented and Frame-Based Languages. // Journal of the ACM, ─ 1995. ─ kifer90logical.pdf.

[KCP 95] Filling Knowledge Gaps in a Broad-Coverage MT System. Knight K.; Chancer I.; Haines M.; Hatzivassiloglou V.; Hovy E.; Iida M.; Luk S.; Whitney R.A.; Yamada, K. // In Proc. of the 14th IJCAI Conference. Montreal. Canada. ─ 1995.

[LR 96] *Lamping J., Rao R. Pirolli P.* A focus + context technique for visualizing large hierarchies // Journal of Visual Languages and Computing. ─ 1996. ─ Vol.7, No.1. ─ P.33–55. ─ lamping95focuscontext.pdf

[LH 00] *Luke S., Heflin J.* SHOE 1.01. Proposed Specification. SHOE Project. ─ February 2000. ─ http://www.cs.umd.edu/projects/plus/SHOE/spec1.01.htm

[LS 99] *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.

[MAL 94] Semi-automatic Knowledge Acquisition in Plinius: An Engineering Approach. *Mars N., Ter Stal W., De Jong H., Van der Vet P., Speel P.* // Proc. 8th Banff Knowledge Acquisition for Knowledge-based Systems Workshop, Banff, January -Febraury 1994. – P. 4.1-4.15.

[MFRW 00] *McGuinness D., Fikes R., Rice J., Wilder S.* An environment for merging and testing large ontologies // In Proc. of the Seventh Int. Conf., KR2000, Morgan Kaufmann Publishers, San Francisco, CA. ─ 2000.

[MG 91] *MacGregor R.* Inside the LOOM classifier // SIGART bulletin, ─ 1991. ─ Vol.3, No.2, ─ P.70-76.

[MS 02] *Maedche A., Staab S.* Tutorial on Ontologies: Representation, Engineering, Learning and Application // ISWC'2002. ─ iswc-tutorial-maedche-staab.pdf

[MIKS 96] *Mena E., Illarramendi A., Kashyap V., Sheth A.* OBSERVER: An approach for query processing in global information systems based on interoperation across preexisting ontologies // Distributed and Parallel Databases—An International Journal. ─ 2000. ─ Vol.8, No.2. ─ mena96observer.pdf.

[MOT 97] *Motta E.* Reusable Components for Knowledge Modelling // Ph.D. Thesis. The Open University. ─ 1997.

[MUS 98] *Musen, M.* Domain Ontologies in Software Engineering: Use of Protégé with the EON Architecture // Methods of Information in Medicine. – 1998. – P.540-550.

[MWD 01] *Mitra P., Wiederhold G., Decker S.* A scalable framework for interoperation of information sources // The 1st Intern. Semantic Web Working Symposium, SWWS'01, Stanford University, Stanford, CA. 2001. ─ mitra01-scalable.pdf

[NM 00] *Noy N., Musen M.* PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment // Proc. of Seventeenth National Conference on Artificial Intelligence, Austin, TX, USA, ─ 2000. ─ P. 450-455. ─ SMI-2000-0831.pdf.

[NM 01] *Noy N., McGuinness D.* Ontology Development 101: A Guide to Creating Your First Ontology // Stanford University. ─ March 2001. ─ ontology-tutorial-noy-mcguinness.pdf.

[NM 03] *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.

[NM 99] *Noy, N., Musen, M.,* SMART: Automated Support for Ontology Merging and Alignment // In Proc. of the 12th Workshop on Knowledge Acquisition, Modeling and Management, Ban, Canada, ─ 1999.

[SAL 94] CommonKADS: A Comprehensive Methodology for KBS Development. A. Schreiber, B. Wielinga, R. Hoog, J. Akkermans, W. Van de Velde // IEEE Expert ─ 1994. ─ Vol. 9, No. 6. – P.28-37.

[SAL 99] The CommonKADS Methodology. *Schreiber G., Akkermans H., Anjewierden A., Hoog R., Shadbolt N., Van de Velde W., Wielinga B.* // Knowledge engineering and management. MIT press, Massachussets. ─ 1999.

[SM 00] *Staab S., Madche A.* Ontology engineering beyond the modeling of concepts and relations // ECAI'2000 Workshop on on Applications of Ontologies and Problem-Solving Methods, Berlin, ─ 2000. ─ staab00-ontology-engineering.pdf.

[SM 01] *Stumme G., Medche A.* FCA-Merge: Bottom-up merging of ontologies // 7th Intl. Conf. on Artificial Intelligence, IJCAI '01. Seattle, WA ─ 2001. ─P. 225–230.

[STO 97] *Stoffel K., Taylor M., Hendler J.* Efficient management of very large ontologies // American Association for Artificial Intelligence Conf., AAAI97, Menlo Park, CA. ─ 1997. ─ P. 442–447. ─ stoffel97efficient.

[SWJ 95] *Schreiber A., Wielinga B., Jansweijer W.* The KACTUS View on the 'O'Word // IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, ─ August 1995.

[UG 96] *Uschold M. , Gr¨uninger M.* Ontologies: Principles, methods and applications // Knowledge Engineering Review. ─ 1996. .─ Vol. 11, No. 2. ─ P.93–155. ─ uschold96ontologie.pdf

[UK 95] *Uschold M., King M.* Towards A Methodology for Building Ontologies // IJCAI-95 Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, Canada, ─ 1995. ─ uschold96building.pdf

[WVV 01] Ontology-Based Integration of Information - A Survey of Existing Approaches. *H. Wache T. Vögele U. Visser H. Stuckenschmidt G. Schuster H. Neumann S. Hübner* // Proc. of the IJCAI-01 Workshop: Ontologies and Information Sharing. ─ 2001. ─ wache01ontologybased.pdf

ПРИКЛАДИ ОНТОЛОГІЙ Додаток А

- Онтології загального призначення
  - WordNet / EuroWordNet, http://www.cogsci.princeton.edu/~wn
  - Thr Upper Cyc Ontology, http://www.cyc.com/cyc-2-1/index.html
  - IEEE Standart Upper Ontology, http://suo.ieee.org/
- Онтології предметних областей та прикладної специфіки
  - RDF Site Summary RSS, http://groups.yahoo.com/group/rss-dev/files/schema.rdf
  - UMLS, http://www.nlm.nih.gov/research/umls/
  - KA^2 Science Ontology, http://ontobroker.semanticweb.org/ontos/ka2.html
  - RETSINA Calendaring Agent, http://ilrt.org/discovery/2001/06/schemas/icalfull/hybrid.rdf
  - AIFB Web Page Ontology, http://ontobroker.semanticweb.org/ontos/aifb.html
  - Wev KB Ontology, http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo11/www/wwkb
  - Dublin Core, http://dublincore.org
- Мета-Онтології
  - Semantic Translation, http://www.ecimf.org/contrib/onto/ST/index.html
  - RDFT, http://www.cs.vu.nl/~borys/RDFT/0.27/RDFT.rdfs
  - Evolution Ontology, http://kaon.semanticweb.org/exemples/Evolution.rdfs
- Онтології у широкому змісті
  - Agrovoc, http://fao.org/agrovoc/
  - Art and Architecture, http://gretty.edu/research/tools/vocabulary/aat/
  - UNSPSC, http://eccma.org/unspsc/
  - DTD standardizations, e.g. HR-XML, http://www.hr-xml.org/