УДК 004.89
В. В. Литвин, д-р техн наук,
Д. Г. Досин, канд. техн. наук,
Р. В. Вовнянка

КОМП'ЮТЕРНА СИСТЕМА АВТОМАТИЗОВАНОЇ РОЗБУДОВИ БАЗОВОЇ ОНТОЛОГІЇ CROCUS

Анотація. *Наведено підхід до розроблення комп'ютерної системи автоматизованої розбудови базової онтології. Описано основні модулі системи та їх функціонування. Обґрунтовано вибір програмних засобів для практичної реалізації. Застосування такої системи дає змогу у автоматизованому режимі наповнювати онтологію предметної області.*

Ключові слова: *комп'ютерна система, онтологія, база знань, база даних, текстовий документ, машинне навчання, інтелектуальний агент, корисність, семантика, логіка предикатів*

V. Lytvyn, ScD.,
D. Dosyn, PhD.,
R. Vovnjanka

COMPUTER SYSTEM FOR AUTOMATED ONTOLOGY BUILDING BASIC CROCUS

Abstract. *The article exposes the approach to developing a computer system building automated ontology base. There are basic modules of the system and its operation described. There is described choice of software tools for implementation. The using of this system allows filling the domain ontology in automatic mode.*

Keywords: *computer system, ontology, knowledge base, database, text document, machine learning, intelligent agent, utility, semantic, logic of predicate*

В. В. Лытвын, д-р техн наук,
Д. Г. Досын, канд. техн. наук,
Р. В. Вовнянка

КОМПЬЮТЕРНАЯ СИСТЕМА АВТОМАТИЗИРОВАННОГО РАЗВИТИЯ БАЗОВОЙ ОНТОЛОГИИ CROCUS

Аннотация. *Приведен подход к разработке компьютерной системы автоматизированного развития базовой онтологии. Описаны основные модули системы и их функционирование. Обоснован выбор программных средств для практической реализации. Применение такой системы позволяет в автоматизированном режиме наполнять онтологию предметной области.*

Ключевые слова: *компьютерная система, онтология, база знаний, база данных, текстовый документ, машинное обучение, интеллектуальный агент, полезность, семантика, логика предикатов*

1. Аналіз сучасних досліджень

Аналіз стану досліджень та розробок в галузі інтелектуальних інформаційних систем та Інтернет-послуг [1] дає підстави вважати, що виправданими є наступні програмно-технічні рішення:

реалізація системи синтезу онтології як підсистеми портальної служби Інтернет-пошуку;

застосування OWL як мови подання знань в онтології;

застосування HTN та OWL-S як структури та мови автоматичного планування бази знань;

Java API для Protege-OWL – як програмного середовища та бібліотеки класів опрацювання, зокрема, машинного навчання (навчання з підкріпленням) OWL-онтології та бази знань (БЗ);

Link Grammar Parcer – як засобу граматично-семантичного аналізу англомовних текстових документів в електронному форматі;

Apache-PHP-MySQL – як програмних засобів для побудови інтерфейсу з користувачем за архітектурою веб-порталу;

Wget – як веб-служби для автоматизованого доступу до пошукових серверів за запитом, сформованим з ключових слів;

SWRL – як мову правил логічного виводу нових знань дедуктивним та індуктивним методами;

WordNet – як базовий тлумачний словник англійської мови.

Онтологія на мові OWL містить понятійний апарат верхнього рівня та предметної

області [2]. Онтологія верхнього рівня забезпечує:

логічний вивід нових знань, доповнення отриманих повідомлень контекстом;

верифікацію істинності отриманих тверджень;

оцінку вірогідності джерел повідомлень;

забезпечення логічної цілісності БЗ.

Машинне навчання реалізується засобами Java API Protege-OWL. Ці засоби містять бібліотеки класів, в яких реалізовано методи роботи з OWL-структурами: їх читання, доповнення. Таким чином, засоби машинного навчання (ЗМН) функціонують у взаємодії з OWL-онтологією, беручи з неї шаблони граматично-семантичних структур для розпізнавання тверджень (предикатів логіки 1-го порядку) у досліджуваних і/або навчальних текстах та додаючи до неї нові елементи в результаті такого розпізнавання. Для цього застосовується Link Grammar Parcer [3], який розбиває стверджувальне речення, написане граматично правильною англійською мовою, на семантично пов'язані між собою пари слів. LGP містить у своєму складі таблицю відповідності між граматичними конструкціями англійської мови та типами синтаксично-семантичних зв'язків між словами (поняттями). API LGP дозволяє пов'язати цю таблицю з OWL-онтологією, завдяки чому таблиця може динамічно адаптуватися в процесі навчання до заданої ПО.

Засоби машинного навчання на базі Java-бібліотек Protege-OWL містить узагальнений опис семантичного зв'язку, який служить шаблоном для генерування в процесі навчання нових типів семантичних зв'язків та формування для їх ідентифікації в тексті відповідних векторів ознак цих зв'язків. При цьому до ОВР додаються відповідні класи зв'язків та їх властивості. Екземпляри цих класів служать для опису існуючих та нових класів онтології шляхом їх використання як предикатів логіки 1-го порядку.

Як було обґрунтовано у попередньому розділі, ЗМН онтології мають сенс лише у складі деякої інтелектуальної системи. Оптимальним на нашу думку є рішення, у якому такою інтелектуальною системою є система інформаційного пошуку, для якої адаптивна онтологія з одного боку є інструментом для інформаційного пошуку, аналізу і класифікації, а з другого – сама використовує засоби пошуку для постачання нових даних для свого наповнення, синтезу нових предикатів і правил, навчання нових понять та семантичних зв'язків між ними. Таким рішенням стала інтелектуальна система інформаційного пошуку на базі адаптивної онтології, бази знань у галузі матеріалознавства та бази даних наукових публікацій у цій галузі.

Розроблена архітектура системи синтезу онтології була реалізована з використанням вибраних і описаних раніше засобів і програмно-технічних рішень як модуль програмного забезпечення CROCUS (Cognition Relations Or Concepts Using Semantics – «розпізнавання зв'язків і/або понять за їх семантикою») [4 – 6].

2. Основні модулі системи CROCUS

Загальна концептуальна схема системи CROCUS представлена на рис. 1. Підсистема навчання онтології використовує навчальні тексти анотацій наукових публікацій бази даних статей. Для наповнення бази даних система формує множину ключових слів, за якою вибирає з зовнішнього джерела публікацій в мережі Інтернет (ScienceDirect, CiteSeer, Wiley Online Library, Springer) основні метадані про публікації в заданій ПО, зокрема їх анотації, які стають основою для аналізу та навчання онтології.

Суть методу видобування знань з природомовного текстового документа полягає у побудові плану (стратегії) діяльності інтелектуального агента – інформаційної моделі суб'єкта розпізнавання, або уточнення такого плану на підставі даних, виділених у текстовому документі, що розпізнається. У цій роботі план вважається конкретною реалізацією оптимальної стратегії вирішення деякої задачі, що стоїть перед інтелектуальним агентом в рамках заданої проблемної області.

План будується тою формальною мовою подання знань, якою було розроблено інформаційну модель – базу знань інтелектуального агента. Враховуючи, що така база знань вже становить собою певний загальний план функціонування інтелектуального агента, план, збудований на основі розпізнавання змісту природомовного тексту, є субпланом,

тобто, уточненням (виправленням) і/або деталізацією цього загального плану і базується на ньому. Цінність інформації, отриманої внаслідок розпізнавання змісту текстового документа, визначається за приростом очікуваної корисності від реалізації уточненого таким чином плану функціонування інтелектуального агента [7 – 9].

За результатами аналізу анотацій наукові публікації ранжуються за їх релевантністю до інформаційних потреб користувача, тобто за відповідністю до онтології, яка ці потреби відображає. З цією метою здійснюється аналіз кожної анотації як природомовного тексту, будується її образ в термінах онтології у вигляді множини предикатів та правил, ці предикати та правила додаються до бази знань системи і повторно обчислюється очікувана корисність оптимального плану інтелектуального агента. При такому ранжуванні ближче до початку списку система розташовує ті публікації, внесення даних яких призводить до більшої зміни корисності.

Система здатна адаптуватися до потреб користувача, зберігаючи у базі даних систему його преференції. Кожний користувач може виконування навчання власної онтології, система зберігає дані про цей процес, веде статистику сеансів, забезпечує можливість виправляти помилки навчання, а також повертатися до попередніх версій онтології.

Модулі системи CROCUS наведені на рис. 2. Через користувацький інтерфейс клієнт має можливість керувати пріоритетами у ранжуванні документів, тобто коригувати порядок їх розташування у переліку найбільш важливих (релевантних до інформаційних потреб клієнта) документів і/або класифікувати їх. При цьому найважливіші документи використовуються для навчання онтології та побудови ефективних наборів ключових слів, а нові, отримані з Інтернету статті (їх метадані, включно з анотацією) заносяться до бази даних публікацій у взаємозв’язку з преференціями користувача та іншими передумовами отримання документа, передусім – джерела його отримання.

Опрацювання анотації відбувається після її попередньої обробки, перетворенням на множину предикатів за результатами граматично-синтаксичного розбору модулем Link Grammar Parser та доповнення сформованої таким чином моделі анотації семантично близькими предикатами з онтології – контексту цієї анотації. Доповнені моделі порівнюються між собою для обчислення семантичної відстані між їх центрами семантичної ваги і таким чином обираються найбільш близькі за змістом документи з їх подальшим ранжуванням та класифікацією.

Система CROCUS має дві основні функції:

1) інтерактивна автоматизована побудова онтології заданої проблемної області;

2) пошук, збереження і класифікація (ранжування) наукових публікацій як у інтерактивному напівавтоматичному, так і в автоматичному режимі [10].

Кожна з цих функцій реалізована своїм базовим набором функціональних модулів, але частина з цих програмних модулів має подвійне призначення. Система CROCUS реалізована мовою програмування Java за об’єктно–орієнтованою парадигмою як ієрархія класів програмного коду, екземпляри яких викликають одне одного з визначеними на момент виклику параметрами, і/або взаємодіють через події та їх обробники. Більшість модулів системи мають графічний інтерфейс Swing та AWT бібліотек. Всі підключені бібліотеки мають статус відкритих і безкоштовно розповсюджуються. Завдяки їх застосуванню проект є повнофункціональним і має всі необхідні засоби для послідовного розвитку.

Функціональне призначення основних модулів системи CROCUS наведено у табл.1.

Детальний аналіз подібних систем нами здійснено у роботі [11].

3. Функціональне призначення модулів системи CROCUS

Функціональне призначення основних модулів системи наведено у табл. 1.

Базовим елементом управління системою CROCUS є модуль ControlGUI (графічний користувацький інтерфейс управління).

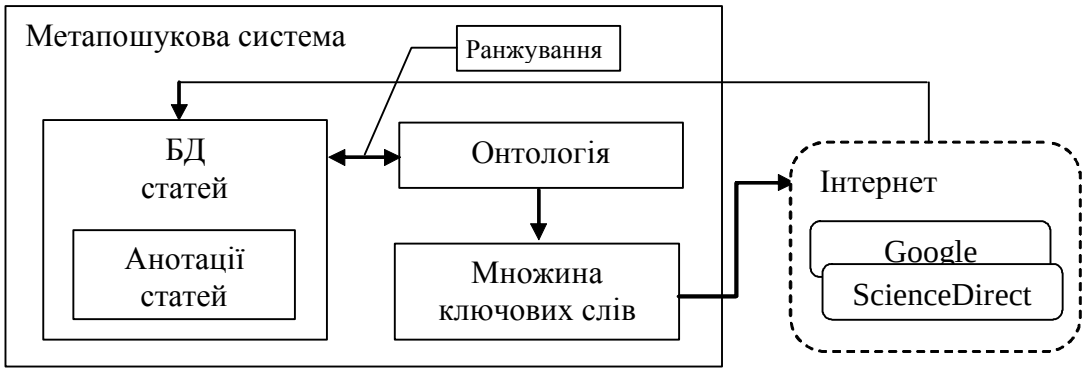

Рис. 1. Концептуальна схема системи CROCUS

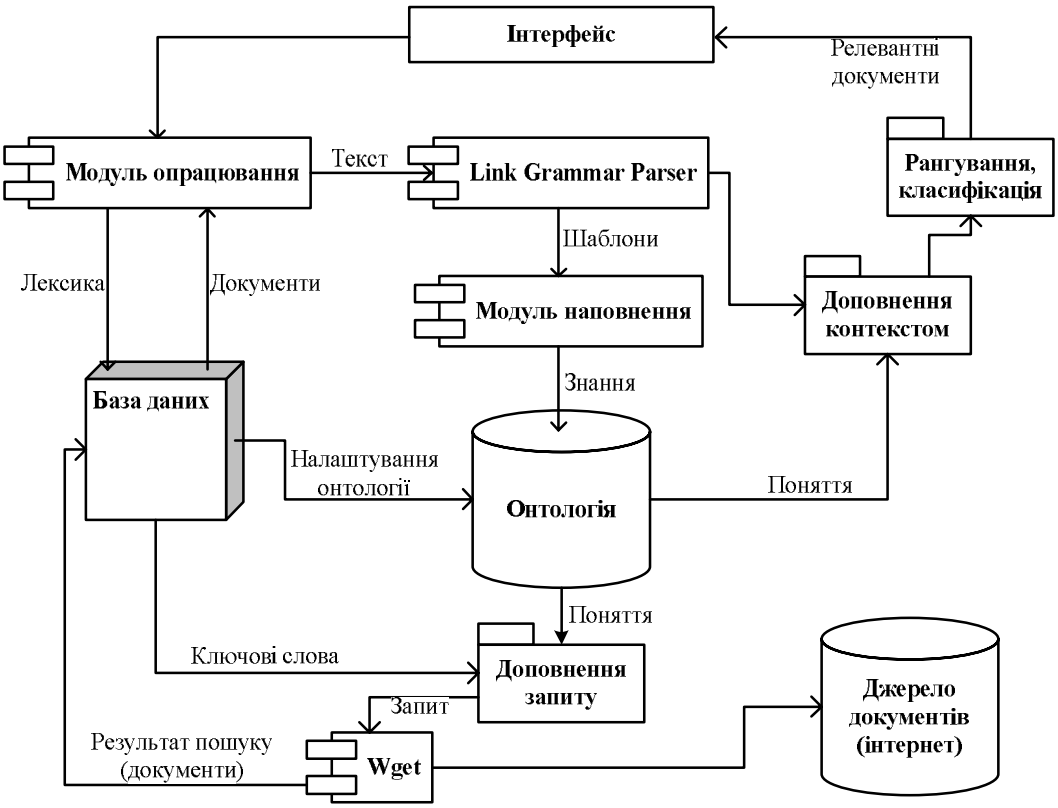

Рис. 2. Модулі системи CROCUS

1. Функціональне призначення основних модулів системи CROCUS

1.	Attribute.java	Підпрограма виявлення атрибутивних (прикметникових) зв'язків в реченнях. Точніше, дієслівних зв'язків. Кожний прикметниковий зв'язок насправді є формою дієслівного: «Помідор зелений» ≡ скорочення від «помідор **має колір** 'зелений'». При цьому онтологія має містити бінарний предикат <**має колір**>(<домен, тт. область визначення = «помідор»>, <діапазон допустимих значень = «зелений колір»>)
2.	BinaryLink.java	Визначення ваги горизонтального бінарного зв'язку

Продовження табл. 1

3.	CGrammarObject.java	Розбирає речення на слова-токени, які після крапки супроводжує буква, що означає тип слова як частина мови (n-noun, v-verb etc.), таким чином, що за допомогою методів **getName()** чи **getType()** можна отримати слово та його відповідний частині мови тип, визначений LinkGrammarParser'ом
4.	CGrammarObjectArr.java	Конструктор екземплярів цього класу формує масив граматичних об'єктів з текстового рядка 'sentence'
5.	CGrammarObjectType.java	Клас типів елементів масиву CGrammarObjectArr
6.	CLinkType.java	Клас методів розбирання стрічки типу зв'язку між парами слів у реченні на основний вид зв'язку **getType()** та його підвиди **getSpecification()**
7.	ControlGUI.java	Головне графічне вікно управління програмою
8.	CSOLink.java	Розбирає результати роботи Link Parser'а - визначає відповідність вказаних парсером типів зв'язків словам, розміченим дужками у попередньому рядку.
9.	CSOLinksArr.java	Складає екземпляри типу CSOLink у динамічний масив
10.	Descriptor.java	Формує множину паттернів в один масив
11.	DParsedData.java	Формуються текстові рядки з результатами роботи LGP
12.	FunctionGUI.java	Шаблон графічного інтерфейсу
13.	GSL.java	Формує паттерн виду семантичного зв'язку для розпізнавання семантичні зв'язки за їх паттернами
14.	Is_a.java	Процедура додавання підкласів до існуючих класів
15.	MainProc.java	Основна процедура[1] у «неграфічному варіанті» - Більше не підтримується 02.05.2011 12:38. 1) Створюються константні параметри для запуску зовнішньої для цього пакету програми Link Grammar Parses 2) Відкриваються три потоки - для входу Parser'а, виводу його даних та його помилок. 3) вказується URL-адреса онтології, розташованої десь on-line; 4) з рядка запуску усієї програми зчитується ім'я файла тексту, з якого буде доповнюватися онтологія (1-й параметр в рядку); 5) зчитується онтологія з заданої раніше адреси; 6) виводиться її вміст у стандартний канал виводу; 7) за допомогою процедури DumpOWLModel виводиться вміст онтології у файл resulting.owl; 8) виконується пошук Is-a зв'язків; 9) виконується пошук Consist-of зв'язків; 10) виконується процедура Attribute.
16.	MetaObject.java	Створює структуру для опису семантичного об'єкта - суб'єкта дії, об'єкта дії чи дії як виду зв'язку між суб'єктом і об'єктом
17.	OntologyMapClass.jav	Додає класи на певний рівень онтології (**addClassesToLevel**)
18.	OWLEvaluation.java	Найчастіше застосовувана процедура цього класу (його об'єктів) • Розрахунок ваги понять та зв'язків онтології • DumpOWLModel(String file_name) виконує вивід онотології у <файл.owl> • ShowAll - виконує процедуру візуалізації онтології в стандарному каналі виводу

Продовження табл. 1

19.	Parser.java	Запуск Link Grammar Parser на виконання, налаштування його режимів, подання на вхід текстового файлу з реченнями, збереження у заданому файлі результатів роботи парсера
20.	Part_of.java	Розпізнає семантичні зв'язки типу Part-of у англомовних реченнях (після їх обробки Link Grammar Parser).
21.	Pattern.java	Конструктор класу Pattern, у якому створюється екземпляр паттерну семантичного зв'язку **для його розпізнавання в реченнях статистичним методом.** Для цього збираються DIM найважливіших елемента триплета «суб'єкт - мета-зв'язок -> об'єкт», а саме «суб'єкт -> мета зв'язок» та «мета зв'язок -> об'єкт», для яких вказано, що вони виникли під час навчання цьому семантичному зв'язку, та відіграють в ньому найважливіше значення (бо найчастіше зустрічаються). В онтології статистика зберігається у вигляді екземплярів даного класу семантичних зв'язків у форматі: default_<ім'я-сем.зв'язку>_<metalink:{D,S,O,...}>_<Subject|Object>_<ім'я-суб'єкта/об'єкта>.
22.	SemLinkDescriptArr.java	З досліджуваного речення вибирає послідовність дескрипторів типу SemLinkDescriptor і створює з них динамічний масив
23.	SemLinkDescriptor.java	Дескриптор семантичного зв'язку у вигляді триплета: *metaSubject (string)-> link-type (string)-> metaObject (string)* (З таких триплетів має складатися вектор ознак семантичного зв'язку. Кожен триплет для кожного типу семантичного зв'язку має свій ваговий коефіцієнт. Розмірність такого вектора вибрана рівною 10.

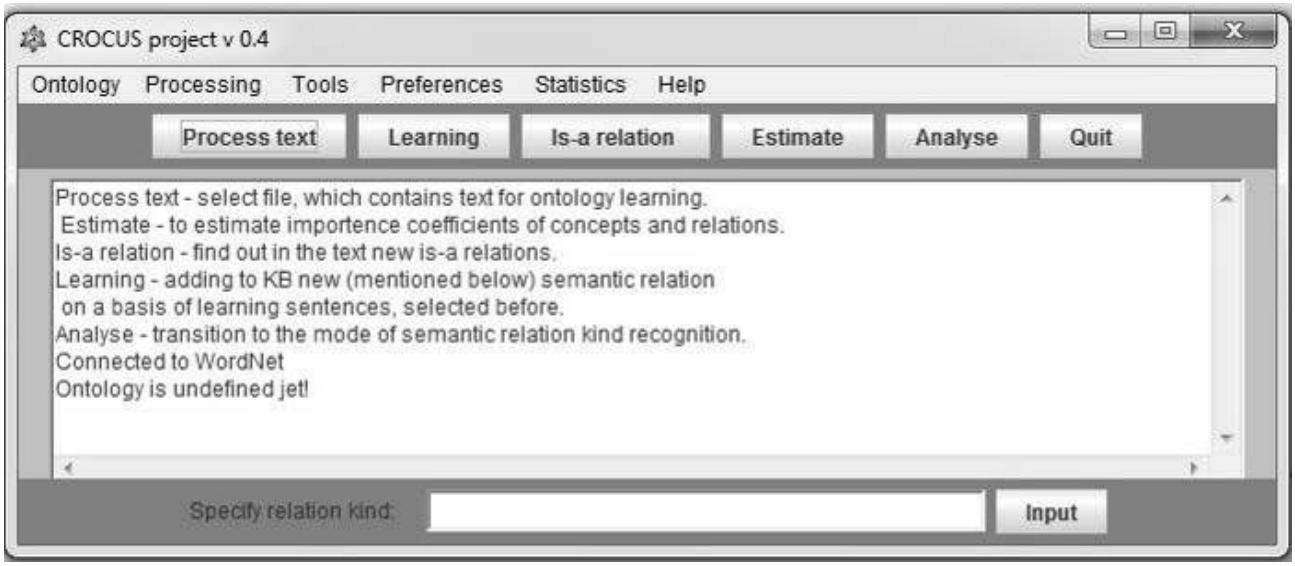

Рис. 3. Головне вікно користувацького інтерфейсу програми CROCUS

Цей модуль має графічний інтерфейс, через який користувач може виконувати процедури, передбачені функціональністю системи. Для цього в модулі передбачене основне меню, а найважливіші його функції виведені на панель інструментів. Контрольний вивід здійснюється на відповідну текстову панель, а для вказання типу семантичного зв'язку в процесі навчання онтології на навчальних реченнях в нижній частині головного вікна передбачене відповідне текстове поле вводу (рис. 3).

Зважаючи на важливість ефективного діалогу система–користувач, як діалогу виду людина–комп'ютер, значну увагу при розробці було приділено графічному дизайну інтерфейсу, його інтуїтивності і лаконічності у поєднанні з функціональною повнотою реалізації завдань проекту, а також його придатності до масштабованості та можливості розвитку функціональності. Зважаючи на гостру конкуренцію між подібними проектами, важливе місце було відведене розробленню у пізнаваного логотипу, який би відповідав змісту слова, утвореного абревіатурою CROCUS. Досвід подібних зарубіжних проектів підтверджує успішність такого підходу (Наприклад, проект Protege з опрацювання OWL-онтологій, проект GATE та інші). Саме тому всі вікна діалогу з користувачем оздоблені логотипом проекту CROCUS – зображенням шестипелюсткової квітки шафрана (крокуса). Зображення та дизайн основних вікон інтерфейсу були розроблені в рамках проекту професійними дизайнерами.

У системі передбачена інтернаціоналізація усіх текстових діалогів. Користувач може вибрати зручну йому мову інтерфейсу з чотирьох можливих на даний момент. Доповнення переліку мов іншою мовою не становить проблеми і полягає у перекладі на цю мову файла діалогів MessagesBundle_xx_XX.properties, де XX – код мови (RU – для російської, uk-UA – для української і т.п.).

Для вибору мови діалогу необхідно в пункті головного меню 'Preferences' вибрати підпункт 'language'.

4. Обгрунтування вибору програмних засобів

Використання стандартних бібліотек програмних засобів дозволяє уникнути невиправданих перевитрат часових, фінансових та людських ресурсів на їх повторне розроблення. Тому в роботі було досліджено широкий перелік діючих аналогічних до розробленого проектів, переважна більшість яких спираються на концепцію відкритого програмного коду та розповсюдження програмних продуктів на умовах безоплатного ліцензування. Провідні групи розробників забезпечують свої проекти засобами API (Application Programming Interface), завдяки яким функціональність цих проектів може бути ефективно використана простим застосуванням каталогізованих і добре документованих процедур і функцій з відповідними параметрами.

Співтовариство розробників програмних засобів напрацювало засади використання пакетів прикладного програмного забезпечення під різними ліцензійними умовами, а також участі у підтримці і розвитку діючих проектів, завдяки чому кожен розробник має можливість отримати, встановити для особистого використання та розвивати на власний розсуд і в міру можливостей ці проекти, або бібліотеки програмних засобів, які входять до їх складу. Такі Інтернет-портали як SourceForge.net містять в собі всю необхідну палітру інструментальних засобів для розміщення там, документування і супроводу проектів довільного ступеня складності, стадії розроблення, рівня доступу і популярності серед користувачів. Розробники активно використовують також спеціалізовані сервери контролю версій розробки, які забезпечують колективне (хай і тисячі учасників) розроблення програмних засобів. Найпопулярнішим серед таких є сервер Git. Він може бути встановлений окремо як індивідуальний чи корпоративний сервер, а може використовуватись загальнодоступний глобальний Git-сервер GitHub.

Серед мов програмування, як показали дослідження, левова частка розробок в галузі опрацювання природомовних текстових документів, майже всі розробки в галузі побудови та навчання онтологій припадають на

мову Java. Крім того, Java зберігає домінування серед мов проектів, розміщених на ресурсі SourceForge.

Вирішальним аргументом на користь вибору Java як мови програмування проекту стала наявність і доступність Java API у проекту Стенфордського університету (США) Protege-OWL, адже саме Стенфордський Центр досліджень з біомедичної інформатики (Stanford Center for Biomedical Informatics Research), став флагманом практичних розробок у галузі засобів розроблення, редагування та навчання баз знань та онтологій мовою подання знань OWL.

Також мовою Java розроблено проекти:

Gate [http://gate.ac.uk/] – множина засобів опрацювання текстових документів з метою виявлення нових знань;

owlapi.sourceforge.net – ще один Java-проект, який являє собою бібліотеку Java-класів з широкою функціональністю з опрацювання OWL-документів;

Pellet [http://clarkparsia.com/pellet/] – програмний засіб – машина логічного виводу на Java для реалізації міркувань (виведення нових знань) з бази знань на мові OWL 2.0.

Висновки

Таким чином, у роботі розглянуто підхід до розроблення комп'ютерної системи автоматизованої розбудови базової онтології. Розроблено архітектуру системи синтезу онтології як модуль програмного забезпечення CROCUS (Cognition Relations Or Concepts Using Semantics – «розпізнавання зв'язків і/або понять за їх семантикою»). Описано основні модулі системи та їх призначення. Обґрунтовано вибір програмних засобів для практичної реалізації системи. Застосування такої системи дає змогу у автоматизованому режимі наповнювати онтологію предметної області.

Список використаної літератури

1. Ourania Hatzi, Dimitris Vrakas, Nick Bassiliades, Dimosthenis Anagnostopoulos, Ioannis Vlahavas The PORSCE II Framework: Using AI Planning for Automated Semantic Web Service Composition. The Knowledge Engineering Review, Cambridge *University Press*, (2010), – Vol. 02:3, pp. 1 – 24.

2. Досин Д. Г. Інтелектуальні системи, базовані на онтологіях // Д. Г. Досин, В. В. Литвин, Ю. В. Нікольський, В. В. Пасічник. – Львів: «Цивілізація», 2009. – 414 с.

3. Link Grammar – Carnegie Mellon University [Електронний ресурс]. – Режим доступу: http://bobo.link.cs.cmu.edu/link (11.02.2014).

4. Литвин В. В. Метод автоматичної розбудови адаптивної онтології [Текст] / В. В. Литвин, Д. І. Угрин // Вісник Національного технічного університету «ХПІ». – Харків. – 2011. – № 10.– С. 75 – 82.

5. Литвин В. В. Автоматизація процесу розвитку базової онтології на основі аналізу текстових ресурсів [Текст] / В. В. Литвин // Вісн. Нац. ун-ту «Львівська політехніка». Серія : Інформаційні системи та мережі. – 2010. – № 673. – С. 319 – 325.

6. Lytvyn V. Design of Intelligent Decision Support Systems Using Ontological Approach / V. Lytvyn // An International Quarterly Journal on Economics in Technology, new Technologies and Modelling Processes. – Krakiv-Lviv. – 2013. – Vol. II. – No. 1. – Pp. 31 – 38.

7. Литвин В. В. Бази знань інтелектуальних систем підтримки прийняття рішень: монографія / В. В. Литвин. – Львів : Видавництво Львівської політехніки, 2011. – 240 с.

8. Lytvyn V. An Ontology Based Intelligent Diagnostic Systems of Steel Corrosion Protection / V. Lytvyn, D. Dosyn, A. Smolarz // Elektronika. – Lodzj. – No. 8. – 2013. – Pp. 22-24.

9. Lytvyn V. V. The Similarity Metric of Scientific Papers Summaries on the Basis of Adaptive Ontologies / V. V. Lytvyn // Proceedings of VIIth International Conference on Perspective Technologies and Methods in MEMS Design, Polyana, Ukraine, 11–14 May 2011. – Lviv : IEEE; LPNU, 2011. – 162 p.

10. Lytvyn V. Searching the Relevant Precedents Based on Adaptive Ontology / V. Lytvyn, N. Shakhovska, V. Pasichnyk, D. Dosyn // Proceedings of XII International Workshop Computational Problems of Electrical Engineering, Kostryna, Ukraine, 5-7 September 2011. – Lviv : IEEE; LPNU, 2011. – 43 p .

11. Ковалевич В. М. Метод оцінювання новизни знань під час навчання онтологій [Текст] / В. М. Ковалевич, В. В. Литвин // Відбір і обробка інформації. – Львів : – 2013. – Вип. 39(115). – С. 82 – 90.

Отримано 03.02.2014

References

1. Ourania Hatzi, Dimitris Vrakas, Nick Bassiliades, Dimosthenis Anagnostopoulos, and Ioannis Vlahavas. The PORSCE II Framework: Using AI Planning for Automated Semantic Web Service Composition. The Knowledge Engineering Review, (2010), *Cambridge University Press,* Vol. 02:3, pp. 1 – 24 (In English).

2. Dosyn D., Lytvyn V., Nikolsky Y., and V. Pasichnyk. Intelektualjni systemy, bazovani na ontolijah [Intelligent System Based on Ontology], (2009), *Cyvilizacija,* Lviv, Ukraine, 414 p. (In Ukrainian).

3. Link Grammar – Carnegie Mellon University, available at: http://bobo.link.cs.cmu.edu/link (accessed 11.02.2014)

4. Lytvyn V., Ugryn D. Metod avtomatychnoji rozbudovy adaptyvnoji ontologiji [Method of Automatic Ontology Building Adaptive], (2011), *Visnyk Nacionaljnogo tehnichnogo universytetu HPI Publ.*, Kharkiv, Ukraine, No. 10, pp. 75 – 82 (In Ukrainian).

5. Lytvyn V. Avtomatyzacija procesu rozvytku bazovoji ontologiji na osnovi analizu tekstovyh resursiv [Automating the Process of Basic Ontology Based on Analysis of Textual Resources] (2010), *Visnyk Nacionaljnogo universytetu "Lvivska politehnika" Publ.*, Lviv, Ukraine, No. 673, pp. 319 – 325 (In Ukrainian).

6. Lytvyn V. Design of Intelligent Decision Support Systems using Ontological Approach, (2013), *An International Quarterly Journal on Economics in Technology, new Technologies and Modelling Processes*, Krakiv-Lviv, Vol. II, No. 1, pp. 31 – 38 (In English).

7. Lytvyn V. Bazy znanj intelektualjnyh system pidtrymky pryjnjattja rishenj [Knowledge Base Intelligent Decision Support Systems], (2011), Lviv, Ukraine: *Vydavnyctvo Nacionaljnogo Universytetu "Lvivska Politehnika"*, 240 p. (In Ukrainian).

8. Lytvyn V., Dosyn D., and Smolarz A. An Ontology Based Intelligent Diagnostic Systems of Steel Corrosion Protection, (2013), *Elektronika*, Lodzj, No. 8, 2–13, pp. 22 – 24 (In English).

9. Lytvyn V.V. The Similarity Metric of Scientific Papers Summaries on the Basis of Adaptive Ontologies, (2011), *Proceedings of VIIth International Conference on Perspective Technologies and Methods in MEMS Design*, Polyana, Ukraine, 11–14 May 2011, Lviv, IEEE; LPNU, 162 p. (In English).

10. Lytvyn V., Shakhovska N., Pasichnyk V., and Dosyn D. Searching the Relevant Precedents Based on Adaptive Ontology, (2011), *Proceedings of XII International Workshop Computational Problems of Electrical Engineering*, Kostryna, Ukraine, 5–7 September 2011, Lviv, IEEE; LPNU, 43 p. (In English).

11. Kovalevych V.M., and Lytvyn V.V. Method of evaluation knowledge novelty during on-learning ontologies [Metod Ocinuvannja Novyzny Znanj pid chas Navchannja Ontologij], (2013), *Vidbir i Obrobka Informaciji*, Lviv, Ukraina, No. 39(115), pp. 82 – 90 (In Ukrainian).

Досин Дмитро Григорович, к.т.н., ст.наук. співробітник, завідувач лабораторії системного аналізу науково-технічної інформації Фізико-механічного ін-ту ім. Г.В. Карпенка НАНУ,
79030, м. Львів,
вул. Наукова, 5
E-mail:
dmytro.dosyn@gmail.com

Литвин Василь Володимирович,
д.т.н., доцент, професор каф. інформаційних систем та мереж ін-ту комп'ютерних наук та інформаційних технологій Нац. ун-ту «Львівська політехніка»,
79012, м.Львів,
вул. С.Бандери, 12

Вовнянка Роман Володимирович,
аспірант каф. інформаційних систем та мереж ін-ту комп'ютерних наук та інформаційних технологій Нац. ун-ту «Львівська політехніка»,
79012, м. Львів,
вул. С.Бандери, 12