PRACE NAUKOWE AKADEMII EKONOMICZNEJ WE WROCŁAWIU
Nr 1064 2005
Pozyskiwanie wiedzy i zarządzanie wiedzą

Anna Zygmunt, Jarosław Koźlak, Marek A. Valenta
Akademia Górniczo-Hutnicza w Krakowie

WNIOSKOWANIE NA PODSTAWIE WIEDZY WYRAŻONEJ ONTOLOGICZNIE – KONCEPCJA ŚRODOWISKA*

1. Wstęp

Termin „ontologia" zaczerpnięty jest z filozofii. Mianem tym określa się naukę zajmującą się teorią bytu, badaniem charakteru i struktury rzeczywistości. W informatyce ontologie są obiektem badań w różnorodnych środowiskach, takich jak inżynieria wiedzy, przetwarzanie języka naturalnego, współpracujące systemy informatyczne (np. wymieniające wiedzę systemy agentowe), inteligentna integracja wiedzy, pozyskiwanie wiedzy, handel elektroniczny oraz zarządzanie wiedzą. Idea ontologii i ich pierwsze użycia wywodzą się ze sztucznej inteligencji, w której zostały zastosowane w celu ułatwienia współdzielenia i ponownego użycia zgromadzonej wiedzy.

Zainteresowanie ontologiami gwałtownie wzrosło wraz z zaproponowaniem przez T. Berners-Lee koncepcji sieci semantycznych (*semantic web*).

Ontologie zaczęto wykorzystywać do opisu danych znajdujących się w sieci. Wprowadzono je, by usystematyzować i ustandaryzować informacje znajdujące się w sieci, gdyż aby takie dane można było automatycznie analizować, muszą zostać przystosowane do efektywnego przetwarzania, przeszukiwania i znajdowania związków. Trzeba więc dostarczyć dodatkowych informacji, które pozwolą na ich zidentyfikowanie, zaklasyfikowanie oraz na znalezienie powiązań z innymi danymi.

Obecnie standardem wymiany informacji staje się XML. Niestety, o ile XML może być zrozumiały dla człowieka, o tyle dla maszyny już nie. Ze znacznikami XML nie są bowiem związane żadne informacje objaśniające ich znaczenie. Dlate-

* Praca finansowana z grantu KBN nr 4T11C023 23

go potrzebny jest standard, który pozwoliłby maszynom rozumieć znaczenie poszczególnych znaczników, a także powiązań między nimi. XML jest zbyt niskopoziomowy, aby stać się takim standardem. Wprawdzie istnieją rozwiązania oparte na XML, będące pewnymi standardami w pewnych dziedzinach wymiany informacji, ale nie są one rozszerzalne na wszelkie dziedziny.

Tak więc XML dobrze nadaje się do opisu pojedynczych bytów, natomiast trudno za jego pomocą opisać ich znaczenie oraz powiązania między nimi. Do tego celu służą specjalne języki, pozwalające w sposób zrozumiały dla maszyn nie tylko opisać pojedyncze byty, ale także ich znaczenie i powiązania z innymi bytami.

Obecnie najbardziej popularnymi językami opisu ontologii są Resource Description Framework (RDF) [10; 17] i Web Ontology Language (OWL) [11]. Z kolei do przygotowania ontologii można wykorzystać darmowe środowisko do edycji baz wiedzy Protege 2000 [20]. Biblioteka Jena [7] umożliwia odwoływanie się w programach do wiedzy opisanej w postaci ontologicznej. Stosowane ontologie składają się z [13]: pojęć/klas opisujących elementy/klasy dziedziny modelu (*classes*, *concepts*), własności opisujących cechy pojęcia (*slots*, *roles*, *properties*), ograniczeń nałożonych na wartości, jakie mogą przyjmować cechy pojęcia (*facets*, *role description*). Baza wiedza jest tworzona z ontologii wraz ze zbiorami instancji klas.

Zaletami ontologii, nie podnoszącymi efektywność przygotowania i użycia wiedzy wyrażonej za ich pomocą, jest łatwość współdzielenia tak wyrażonej wiedzy przez różne aplikacje oraz reużywalność stworzonych komponentów wiedzy w innych systemach. Wykorzystywana wiedza może zostać wyrażona w języku do specyfikacji ontologii (jak RDF czy OWL), zapisana w pliku XML i udostępniona pod podanym adresem URL. Za reużywalne można przyjąć takie elementy wiedzy, które mogą zostać potem wykorzystywane w innych aplikacjach. Można tu zaliczyć wiedzę stworzona z użyciem Protege 2000 oraz wyrażoną w OWL lub RDF/RDFS.

2. Wybrane metody łączenia ontologii

Ontologie umożliwiają współdzielenie wiedzy. Oznacza to, że stworzone ontologie mogą być wykorzystywane przez szerszą grupę zastosowań. Zaproponowano wiele różnych technik łączenia ontologii umożliwiających wykorzystanie pojęć z relacjami zdefiniowanymi w różnych ontologiach składowych. Jak więc można zauważyć, ontologie, które chcemy łączyć, mogą reprezentować wiedzę z różnych dziedzin; mogą również być wyrażone za pomocą różnych formatów czy metod reprezentacji wiedzy.

W literaturze [5; 19] wyróżnia się trzy główne podejścia do łączenia ontologii. W pierwszym podejściu ontologia staje się dzieloną ontologią globalną. Może się ona składać z mniejszych jednostek ontologicznych.

Z kolei w drugim podejściu każde źródło informacji ma swoją własną, lokalną ontologię, dzięki czemu może używać odrębnego słownika. Ontologie mogą być rozwijane niezależnie. Scalenie ich w jedną wirtualną ontologię wymaga odwzorowania.

Ostatnie podejście jest próbą połączenia zalet dwóch poprzednich, z jednoczesnym wyeliminowaniem ich wad. Każde źródło informacji ma własną ontologię lokalną, wywodząca się z ontologii globalnej, w celu zapewnienia łatwiejszego ich uzgadniania.

Proces łączenia ontologii obejmuje takie metody jak [19]: uzgadnianie (*aligning*), łączenie (*merge*), integracja (*integrating*), kombinacja (*combining*), odwzorowanie (*mapping*) i kontrola wersji (*versioning*).

Uzgadnianie stosuje się w przypadku, gdy wiedza opisana przez różne ontologie ma być wykorzystana w celu stworzenia nowej ontologii, a integrowane ontologie nakładają się i uzupełniają wzajemnie. Metoda ta polega na stworzeniu połączeń między ontologiami, które powinny w pełni reprezentować nakładające się obszary opisywanych dziedzin, identyfikować związki między nimi, zachowując przy tym zgodność i spójność całości.

Metoda **łączenia** polega na połączeniu dwóch lub więcej ontologii źródłowych w jedną spójną ontologię wynikową, która może zastąpić każdą z ontologii źródłowych lub może być użyta jako ontologia pośrednicząca w komunikacji między systemami korzystającymi z ontologii źródłowych [18].

Integracja [15] z kolei prowadzi do powstania jednej ontologii wynikowej (podobnie jak w przypadku łączenia ontologii), przy czym dziedzina ontologii źródłowych jest z reguły inna niż dziedzina ontologii wynikowej, ale istnieje między nimi jakiś związek. Ontologie źródłowe można integrować różnymi sposobami: bez zmian (ontologia spełnia wymagania bez zmian), przez adaptację (ontologia może wymagać dopasowania), specjalizację (zbyt ogólna ontologia może zostać wyspecjalizowana – dziedzina ontologii nie ulega zmianie) czy rozszerzenie (ontologia może zostać rozszerzona o nowe pojęcia).

Metoda **odwzorowania** polega na znajdowaniu powiązań między pojęciami i relacjami pochodzącymi z różnych źródeł, uważanych za podobne (zgodnie z pewną funkcją podobieństwa). Aby odwzorować dwie ontologie, należy dla każdego pojęcia z jednej ontologii znaleźć równorzędne pojęcia o podobnej semantyce w drugiej ontologii (i odwrotnie). Należy zatem zdefiniować semantyczne relacje, które mogą istnieć między dwoma powiązanymi pojęciami, oraz zaimplementować algorytm wyszukiwania pojęć o podobnym znaczeniu semantycznym.

Głównym celem **kontroli wersji** jest umożliwienie efektywnego wykorzystania istniejących ontologii w nowych dla nich zastosowaniach. Metoda ta polega na zachowaniu relacji między nowo utworzonymi ontologiami, ich starszymi wersjami oraz zgodnymi z nimi danymi.

Łączenie ontologii jest procesem złożonym i jest z nim związanych wiele problemów [19]. Niezależnie tworzone ontologie mogą być opisane w zupełnie róż-

nych językach, o różnym stopniu ekspresji i sposobie opisu (niezgodności językowe). Ontologie mogą różnić się w opisie rzeczywistości, sposobu jej pojmowania, stosowanego zestawu pojęć (niezgodności na poziomie interpretacji dziedziny, w podejściu do modelowania, niezgodności w terminologii). Łączenie i uzgadnianie ontologii jest procesem bardzo skomplikowanym, wymagającym dużego nakładu pracy od strony projektanta. Ostatnio pojawiły się narzędzia wspomagające proces reużywalności ontologii. Podstawowym problemem jest jednak znalezienie pojęć, które mogą być uzgodnione. Uzgodnienie pojęć może mieć pewne ukryte konsekwencje, które są trudne od przewidzenia w czasie projektowania.

Na obecnym etapie rozwoju technologii łączenie ontologii jest procesem półautomatycznym. Wynika to między innymi z tego, iż znalezienie pojęć, które można powiązać, jest skomplikowane w automatyzacji, a konsekwencje danego mapowania są trudne do przewidzenia. Istniejące narzędzia wspomagające, które prowadzą użytkownika przez proces integracji, zwracają jego uwagę na istotne fakty. Niektóre narzędzia pozwalają na wielokrotne użycie pewnych pasujących elementów ontologii. Integracja ontologii może być przeprowadzona za pomocą takich narzędzi, jak biblioteka Jena czy wtyczka (*plugin*) Prompt do edytora ontologii Protege 2000.

Prompt

Prompt [14] jest dodatkiem (wtyczką) do Protege 2000, pozwalającą porównywać, kopiować (całość lub fragmenty) oraz integrować ontologie. Prompt jest narzędziem wspomagającym, a jego ideą nie jest automatyzacja, lecz raczej przeprowadzenie użytkownika krok po kroku przez proces integracji, z pominięciem przy tym szczegółów implementacyjnych.

Prompt wykorzystuje trzy operacje: łączenie klas/slotów, kopiowanie klas/slotów, które występują w jednej z ontologii źródłowych, oraz usuwanie cyklicznych powiązań poprzez usuwanie przodków. Najistotniejsze jest to, że Prompt pobiera informacje z ontologii źródłowych, rozbudowując nową ontologię wyjściową. Oznacza to, że kolejność wykonywanych operacji ma fundamentalne znaczenie. Prompt daje możliwość pełnej kontroli nad procesem integracji: począwszy od zmiany kolejności działań poprzez ustalanie zakresu tychże działań (czy ma dotyczyć tylko aktualnej klasy, czy też podklas lub nadklas), aż do możliwości zmiany nazw integrowanych klas.

Prompt został stworzony z myślą o scalaniu. Często jednak będziemy mieli do czynienia z mapowaniem (te same klasy, ale inaczej nazwane) lub integracją. Najczęściej zaś z hybrydą wymienionych. Prompt korzysta z logiki zapisanej w ontologii, jednakże nie jest w stanie rozpoznać, że dwie klasy – zupełnie różnie nazwane, z inaczej ponazywanymi slotami – stanowią w rzeczywistości tę samą klasę. Dlatego też często sugestie przy integracji będą błędne.

Prompt ma dopracowany i intuicyjny interfejs. W każdej chwili pozwala na podejrzenie ontologii źródłowych (z podświetleniem aktualnie badanych

klas/slotów/instancji), a także ontologii wyjściowej; podaje również wyjaśnienia swoich sugestii. Można dzięki niemu zobaczyć także aktualnie wykryte kolizje. Przede wszystkim zaś istnieje możliwość stworzenia własnych operacji integracji.

Pomimo tych ułatwień, przeprowadzenie poprawnej integracji nie jest sprawą prostą. Wynika to ze złożoności samego zagadnienia – często nawet proste ontologie można połączyć na kilka sposobów. Sprawa komplikuje się jeszcze bardziej, gdy w grę wchodzi integracja instancji, które nie zawsze muszą być poprawne. Tym samym, aby otrzymać pożądany efekt, poza wprawą należy przygotować plan integracji.

Tworzenie takiego planu należy zacząć od nakreślenia sobie schematu diagramu, który pozwoli nam ustalić kolejność działań. Analiza takiego schematu ułatwi także identyfikację potencjalnych problemów – jest to istotne, gdyż często błąd popełniony w czasie integracji powoduje konieczność ponownego przeprowadzenia procesu.

Prompt wspiera integrację ontologii, przy czym najlepiej się nadaje do integracji schematów. Integracja na poziomie danych (w tym wypadku instancji) jest już nieco bardziej kłopotliwa.

Biblioteka Jena

Jena [7] jest platformą do tworzenia semantycznych aplikacji sieciowych, dostarczającą rozmaite mechanizmy do definiowania ontologii i zarządzania wiedzą w nich zawartą.

Mechanizmy Jeny pozwalają na przeglądanie i modyfikowanie modeli oraz na identyfikację tych samych zasobów na podstawie identyfikatora URI (także w różnych modelach). Umożliwiają również zbudowanie modelu z poziomu aplikacji.

Jena składa się z następujących komponentów:

- RDF API - API w języku Java służącego do manipulacji bazą wiedzy zdefiniowaną w języku RDF;
- ARP – RDF/XML – parsera Jeny zgodnego z ostatnią specyfikacją RDF;
- Podsystemu utrwalania (Persistence Subsystem) – mechanizmu umożliwiającego składowanie wiedzy z modelu w zdenormalizowanej postaci stwierdzeń w zewnętrznej bazie danych. Obecnie Jena współpracuje z systemami baz danych MySQL, PostgreSQL i Oracle w środowisku Linux i Windows XP;
- Podsystemu wnioskowania (Reasoning Subsystem) – regułowego motora wnioskowania wraz ze zdefiniowanym zestawem reguł dla formalizmów RDFS i OWL/Lite (części OWL Full). Umożliwia on m.in. definiowanie modelu wnioskowania w celu prezentacji stwierdzeń RDF wraz z wiedzą z nich wywnioskowaną. Motor wnioskowania jest skonstruowany tak, aby umożliwić również korzystanie z zewnętrznych zbiorów reguł;
- Podsystemu ontologicznego (Ontology Subsystem) – API w Javie do zarządzania ontologiami zdefiniowanymi w języku bazującym na RDF (OWL,

DAML+OIL, RDFS) zbudowanym nad rdzennym RDF API. Udostępnia programiście zbiór wygodnych w obsłudze klas dziedziczących po ogólnych klasach Resource i Property, umożliwiając tym samym bardziej ścisłe i efektywne modelowanie elementów wiedzy ontologicznej i relacji między nimi (m.in. dziedziczenie zasobów i ich właściwości, ekwiwalentność, rozłączność, ograniczenie zakresu wartości etc.). API współpracuje ściśle z podsystemem wnioskowania, pozwalając do modelowania wykorzystać wiedzę wydedukowaną. Jego część stanowią również klasy umożliwiające hierarchiczną organizację ontologii w moduły wielokrotnego wykorzystania oraz klasy do zarządzania importem takich modułów;

- RDQL – języka zapytań do bazy wiedzy RDF utrzymywanej w relacyjnej bazie danych. Umożliwia dynamiczne generowanie zoptymalizowanych zapytań SQL.

Najważniejsza klasa Jeny to OntModel, która jest obiektową reprezentacją ontologii zawierającą trójki (*subject*, *predicat*, *object*), co jest jak najbardziej zgodne z podstawami teoretycznymi ontologii w informatyce. Dodatkowo w platformie Jena trójki te są przechowywane w postaci zasobów (klasa Resource).

Środowisko **Jena** jest wyposażone w podstawowe operacje na modelach ontologii, takie jak m.in. suma i część wspólna. Dzięki temu realizacja tych operacji sprowadza się do stworzenia odpowiednich modeli ontologii – obiektów klasy *OntModel* oraz wywołania metody *intersection* lub *union*.

Jena zawiera mechanizm mapowania *localquery* i *query*. Mapowanie *query* polega na tym, że piszemy zapytanie na wzór języka SQL o dane przechowywane w postaci danej ontologii, następnie to zapytanie jest mapowane na zapytanie drugiej ontologii. Wówczas następuje wykonanie zapytania oraz przetworzenie otrzymanych wyników. Zaimplementowany algorytm jest taki sam, z tym jednak wyjątkiem, że *localquery* został wprowadzony na potrzeby testowania (dokonuje mapowania na tej samej ontologii bez udziału mediatora).

3. Wnioskowanie

Proces wnioskowania powinien wykorzystywać rozproszoną wiedzę zawartą w poszczególnych źródłach wiedzy wyrażonej ontologicznie. Obecnie istnieje kilka rozwiązań umożliwiających prowadzenie wnioskowania na podstawie wiedzy reprezentowanej w postaci ontologii (w formacie OWL lub zgodnym ze środowiskiem Protege 2000 [16]): Racer [21], F-OWL z Flora2 [4], Bossam [2], Jess [8], Algernon [1]. Najbardziej interesującym i obiecującym podejściem jest wnioskowanie na podstawie Description Logic5 [3; 12].

Description Logics

Logika opisowa (*description logics*, – DL) operuje na reprezentacji pojęć oraz związków między nimi. Jej idea oparta jest na takich metodach reprezenta-

cji wiedzy, jak ramy oraz sieci semantyczne, tzn. takich, w których występują struktury sieciowe, reprezentujące właśnie jednostki oraz związki pomiędzy nimi [12].

Wiedzę reprezentowaną w logice opisów można także przedstawić jako graf, w którym węzły charakteryzują pojęcia (zbiory/klasy jednostek), natomiast powiązania – relacje pomiędzy nimi.

Wiedza w logice opisów jest zgromadzona w postaci TBoxes (opisujących podstawowe cechy, pojęcia, relacje i własności) oraz ABoxes (opisujących poszczególne instancje).

Wnioskowanie w logice opisowej może być prowadzone na podstawie wyrażeń zawierających pojęcia, instancji pojęć oraz aksjomatów wyrażających własności pojęć. Typowymi rodzajami wnioskowania, jakie można prowadzić z użyciem logiki opisowej, są [3]: sprawdzenie istnienia w danym modelu instancji o zadanych właściwościach, zawieranie się zbiorów opisywanych przez dane pojęcie (*subsumption*) będące podstawą operacji klasyfikacyjnych, sprawdzanie spójności bazy wiedzy (*consistency*) oraz weryfikacja instancji (*instance checking*) – umożliwiające sprawdzenie, czy dany element jest instancją danego pojęcia.

Racer

Racer jest systemem umożliwiającym prowadzenie procesu wnioskowania z użyciem logiki opisowej, z uwzględnieniem opisanych powyżej TBoxes oraz ABoxes.

System implementuje logikę opisową ALCQHIR+ (nazywaną też SHIQ), która ma następujące dodatkowe cechy: ograniczenia ilościowe, hierarchię ról (powiązań), powiązania odwrotne oraz przechodniość powiązań. Racer umożliwia także prowadzenie wnioskowania algebraicznego, m.in. z wykorzystaniemu ograniczeń wartości liczb całkowitych (minimalnych lub maksymalnych) czy z operowaniem na wybranych rodzajach równań lub nierówności liniowych i nieliniowych. Racer do komunikacji z innymi systemami DL wykorzystuje standard DIG z użyciem protokołu opartego na XML.

4. Architektura systemu

Celem Autorów był taki wybór architektury, który spełniał by następujące kryteria:

- zachowywał rozproszony charakter wiedzy, która mogła być rozproszona na różnych lokacjach w sieci internet i być dostępna za pośrednictwem URL, a także mieć charakter otwarty (możliwe było dodanie nowych ontologii bez konieczności przebudowy systemu jako całości) oraz umożliwiać wielokrotne użytkowanie wiedzy (nowa wiedza może odwoływać się do pojęć już wcześniej zdefiniowanych);

- umożliwiał dokonywanie na wiedzy operacji, wymagających dostępu do różnych rozproszonych ontologii. Operacjami tymi mogą być przeszukiwanie, wykorzystywanie przy konstruowaniu odpowiedzi na złożone zapytania lub też w procesie wnioskowania;
- zapewniał rozproszenie modułów operujących na wiedzy by odpowiadały rozproszeniu wiedzy, co zapewniało uniknięcie wąskich gardeł oraz ułatwienie rozbudowy systemu wykorzystującego wiedzę w sposób odpowiadający otwartości wiedzy.

Z wyżej wymienionych powodów zdecydowano się wprowadzić architekturę, w której można wyróżnić następujące warstwy logiczne:

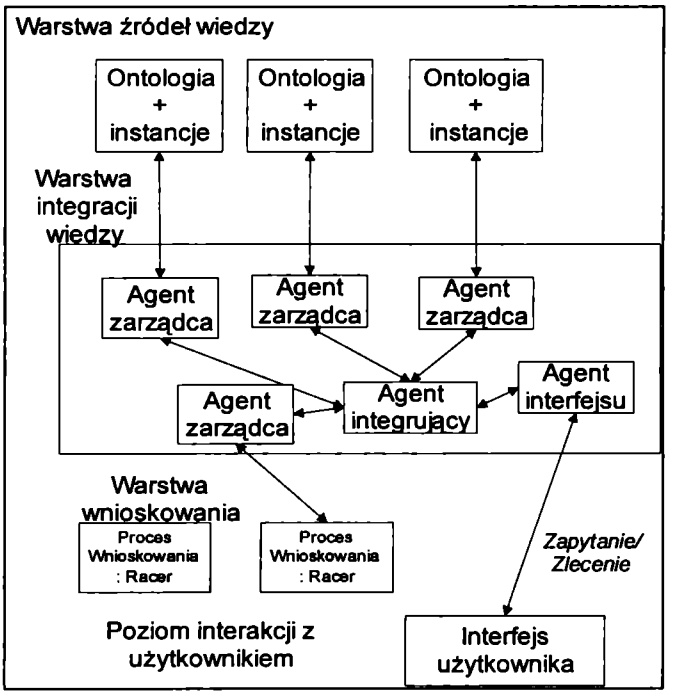
- **warstwę wiedzy** mającą dostęp do ontologii wyrażonych w językach RDF/RDFS lub OWL, ontologie są plikami XML dostępnymi za pośrednictwem internetu pod odpowiednimi URL. Dostęp do ontologii oraz operacje na nich są wykonywane przez agentów za pośrednictwem funkcji biblioteki JENA;
- **warstwę integracji wiedzy** zawierającą infrastrukturę – platformę agentową – umożliwiającą komunikację pomiędzy agentami oraz prowadzenie procesu integracji wiedzy. Ta warstwa jest oparta na platformie agentowej JADE, zgodnej ze standardem agentowym FIPA. Na platformie uruchomieni są agenci odpowiedzialni za komunikację z użytkownikiem, prowadzenie procesów zapytań i wnioskowania oraz integrowanie wiedzy;
- **warstwę wnioskowania** opartą na. systemie Racer, umożliwiającą prowadzenie wnioskowania na podstawie zintegrowanej wiedzy.

Realizację infrastruktury systemu rozproszonego, w którym działają inteligentni agenci zarządzający wiedzą, może zapewnić platforma agentowa JADE [6], zgodna ze standardem FIPA. Platforma agentowa może być rozproszona pomiędzy maszynami pracującymi pod różnymi systemami operacyjnymi, a do jej kontroli można wykorzystać zdalny interfejs graficzny. Konfigurację można zmieniać nawet w czasie działania systemu, przenosząc agentów pomiędzy maszynami.

W środowisku JADE uruchomiono następujące rodzaje agentów [9, s. 265-272]:

- **agentów interfejsu** prowadzący interakcję z użytkownikiem, pobierający zapytania, które następnie są tłumaczone na język zapytań RDQL i przesyłane do agentów udostępniających wiedzę oraz do agentów integrujących wiedzę;
- **agentów zarządcy wiedzy** udostępniający instancje wiedzy oraz wiedzę zawartą w skojarzonych z nimi ontologiach;
- **agentów integrujący wiedzę** wybierający wiedzę (pojęcia, atrybuty) zapisaną w ontologiach źródłowych i wstawiający ją do instancji opartych na docelowej ontologii.

Ponadto, system może komunikować się z zewnętrznymi modułami wnioskowania. Zdecydowano się w tej roli zastosować system Racer.



Rys. 1. Architektura systemu do zarządzania wiedzą zapisaną w postaci ontologii

Źródło: opracowanie własne.

Przykładowa interakcja użytkownika z systemem przebiega według następującego schematu:

1) użytkownik wysyła zapytanie do systemu;

2) agent interfejsu odbiera zapytanie oraz przesyła je do innych agentów – zarządców wiedzy odpowiedzialnych za interakcje z odpowiednimi źródłami wiedzy;

3) agenci-zarządcy realizują zapytanie na podstawie związanych ze sobą źródeł wiedzy;

4) uzyskana z różnych źródeł wiedza jest dostarczana do modułu wnioskującego, opartego na systemie Racer;

5) moduł wnioskujący przeprowadza wnioskowanie oraz zwraca jego wyniki do związanego z nim agenta zarządcy, który następnie przekazuje wyniki do agenta – interfejsu;

6) wyniki zapytania oraz przeprowadzonego wnioskowania są dostarczane użytkownikowi.

5. Podsumowanie

Koncepcje przedstawione w artykule zespół badawczy Katedry Informatyki AGH próbował zrealizować, opierając się na medycynie. Z racji złożoności pro-

blemu oraz niekwestionowanej jego ważności jest to ciekawy obszar zastosowań. Przy współpracy z lekarzami z Katedry Mikrobiologii CM UJ oraz Polskiego Towarzystwa Zakażeń Szpitalnych opracowano ontologie. W pracach [9, s. 265-272; 22, s. 373-384] zaprezentowano możliwości wybranych metod łączenia ontologii z obszaru zakażeń szpitalnych oraz zaprezentowano zalety takiego podejścia. Łączenie ontologii zwiększa efektywność wykorzystywania posiadanej wiedzy, możliwa jest bowiem konstrukcja opisu nowych elementów wiedzy na bazie tych poprzednio istniejących. Zmniejsza się w ten sposób liczbę błędów w wyborze własności i w ich nazwach, umożliwia wykorzystanie wiedzy już wyspecyfikowanej (instancji), czyli wyrażonej za pomocą ontologii źródłowych, oraz ułatwia wyszukiwanie wiedzy z wykorzystaniem aspektu semantycznego stosowanych ontologii. W dalszych pracach podjęto próby rozszerzenia zakresu zastosowań na inne obszary medyczne (endokrynologia).

Zintegrowane z rozproszonych heterogenicznych źródeł ontologie mogą zostać wykorzystane przez inteligentnych agentów w celu przeprowadzenia wnioskowania i uzyskania nowej wiedzy. Autorom udało się opracować prototyp systemu o architekturze przedstawionej na rys. 1. Na podstawie reguł zapisanych w ontologiach system Racer potrafi przeprowadzić rozumowanie wnioskujące o rodzaju zakażenia szpitalnego, wykorzystując do tego charakterystykę pacjenta.

Literatura

[1] Algernon - Rule-Based Programming, http://algernon-j.sourceforge.net/.

[2] Bossam - A Java-based Rule Processor for the Semantic Web, http://mknows.etri.re.kr/bossam.

[3] Donini F.M., Lenzerini M., Nardi D., Schaerf A., *Reasoning in Description Logic, A Great Collection*, red. U. Gnowho, CLSI Publications, 1997.

[4] F-OWL: An OWL Inference Engine in Flora-2, http://fowl.sourceforge.net/.

[5] Fridman N., Musen M., *SMART: Automated Support for Ontology Merging and Alignment, Twelth Workshop on Knowledge Acquisition*, Modeling, and Management, Banff, Canada 1999.

[6] Java Agent DEvelopment Framework, http://jade.cselt.it/.

[7] JENA – A Semantic Web Framework for Java, http://jena.sourceforge.net/index.html.

[8] JESS ™, the Rule Engine for the Java™ Platform, http://herzberg.ca.sandia.gov/jess/.

[9] Koźlak J., Zygmunt A., *Zaawansowane metody reprezentacji wiedzy z wybranego obszaru medycyny*, [w:] *Współczesne problemy sieci komputerowych. Zastosowanie i bezpieczeństwo*, red. A. Kwiecień, A. Grzywak, Wydawnictwo Naukowo-Techniczne, Warszawa, Gliwice 2004.

[10] Manola F., Miller E., RDF Primer, W3C Recommendation, http://www.w3.org/TR/2004/REC-rfd-primer-20040210.

[11] McGuinness D.L., Harmelen F., OWL. Web Ontology Language. Overview, W3C Recommendation, http://www.w3.org/TR/2004/REC-owl-features-20040210, 2004.

[12] Nardi D., Brachman R.J., *An Introduction to Description Logic*, [w:] red. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider *Description Logic Handbook. theory, Implementation and Applications*, Cambridge University Press, 2003.

[13] Noy N.F., McGuinness D.L., *Ontology Development 101: A Guide to Creating Your First Ontology*, SMI Tech. Report, SMI-2001-0880.
[14] Noy N.F., Musen M.A., *The PROMPT Suite: Interactive Tools For Ontology Merging And Mapping*, „International Journal of Human-Computer Studies", 2003.
[15] Pinto H.S., Gomez-Perez A., Martins J.P., *Some Issues on Ontology Integration, Proceedings of IJCAI99 Workshop on Ontologies and Problem Solving Methods: Lessons Learned and Future Trends*, Stokholm, Sweden, 1999.
[16] Protégé OWL Plugin, http://protege.stanford.edu/plugins/owl/.
[17] Resource Description Framework (RDF), http://www.w3.org/RDF.
[18] Sowa J.F., *Ontology, Metadata, and Semiotics*, [w:] *Conceptual Structures: Logical, Linguistic, and Computational Issues, Lecture Notes in AI #1867*, red. B. Ganter, G.W. Mineau, Springer-Verlag, Berlin 2000.
[19] Tamma V., *An Ontology Model supporting Multiple Ontologies for Knowledge sharing*, Thesis of University of Liverpool, 2001.
[20] Welcome to the Protege Project, http://www.w3.org/RDF/.
[21] Wessel M., Haarslev V., Möller R., *RACER User's Guide and Reference Manual Version 1.7.19.*
[22] Zygmunt A., Koźlak J., *Metody efektywnego zarządzania wiedzą reprezentowaną w postaci ontologii*, [w:] *Strategie informatyzacji i zarządzanie wiedzą*, red. Z. Szyjewski, J.S. Nowak, J.K. Grabara, Wydawnictwo Naukowo-Techniczne, Warszawa 2004.

CONCEPT OF ENVIRONMENT FOR REASONING ON BASIS OF KNOWLEDGE EXPRESSED USING ONTOLOGIES

Summary

Classical approach to reality modelling in databases causes difficulties in development of model of represented data as well as integration of data stored in different, heterogeneous and distributed systems. It lacks of common representation methods of data and their semantics as well as methods of data exchanging among sources, and in consequence – knowledge obtained on their basis.

The need of the introduction of standards to knowledge descriptions came into being. One of the intensely developing approaches are ontologies. Ontologies make it possible to model a reality using concepts, their properties and relationships among them. Knowledge based on ontologies may also contain the set of instances of concepts defined in such a way.

Several ontology description languages were proposed (for example Ontolingua, OIL, DAML). Currently, commonly used are RDF, RDFS and OWL (the standards of description of these languages were prepared and accepted by W3C).

Ontologies provide concepts which may be understood in the same way by all the systems operating on the knowledge represented in this manner. They make it possible to share knowledge and to realize cooperation between different knowledge sources. It is possible to develop an ontology structure performing the integration process, assuming that ontologies contain only partial descriptions of some concepts and may be complementary to one another. As a result one can obtain a model of reality with higher expression. Ontology integration may be performed using tools like library Jena or Prompt – a plug-in to ontologies editor Protégé 2000.

Such a knowledge organization provides environment for performing reasoning process. Reasoning may use distributed knowledge from particular sources containing knowledge expressed in ontologies. There exist several solutions making it possible to perform a reasoning process using ontological representation (in OWL or other compatible to Protégé 2000 format): Racer, F-OWL with

Flora2, Bossam, Jess, and Algernon. The most interesting and promising approach is Description Logic reasoning.

In the paper, a concept of a system which makes it possible to reason on the basis of distributed sources of medical knowledge, expressed in RDF or OWL, is presented. One can distinguish the following logical layers: knowledge sources using ontological representation, a knowledge integration layer (based on agent approach, supported by JADE platform), a reasoning layer using Racer system and a user interaction layer (responsible for queries about knowledge, among the others the ones which need a reasoning process).