Postprint of: Boiński T. M., Budnik Ł., Jakowski A., Mroziński J., Mazurkiewicz K., OSC - domain oriented ontology creation system, POLISH JOURNAL OF ENVIRONMENTAL STUDIES, Vol. 18, Iss. 3B (Supplement) (2009), pp. 35 - 38

mgr inż. Tomasz Boiński, mgr inż. Łukasz Budnik, mgr inż. Andrzej Jakowski, Jacek Mroziński, Krzysztof Mazurkiewicz
Department of Computer Architecture
Faculty of Electronics, Telecommunication and Informatics
Gdańsk University of Technology

OCS - System wytwarzania ontologii dziedzinowo zorientowanych

Streszczenie

Ontologie, jako składnik sieci semantycznej, są podstawową cegiełką pozwalającą na nadaniu treści dostępnej w internecie znaczenia zrozumiałego dla komputerów. W niniejszej publikacji zaprezentowano architekturę i funkcjonalność dziedzinowego portalu grupowego tworzenia oraz składowania ontologii realizowanego przez katedrę KASK wydziału Elektroniki, Telekomunikacji i Informatyki Politechniki Gdańskiej. Omówiono najistotniejsze elementy wytwarzanego systemu odróżniające go od innych podobnych rozwiązań. Zaprezentowany został sposób przechowywania informacji w bazie danych oraz dostępu do nich, mechanizm konwersji elementów z postaci obiektowej na zbiór trójek i vice versa a także metodologia grupowej pracy nad wytwarzaniem ontologii przyjęta w opisywanym systemie.

Słowa kluczowe: ontology, versioning, development

Wstęp

Wraz z rozwojem internetu to, co stanowiło o jego sile – prostota języka HTML, staje się obecnie przeszkodą w jeszcze lepszym i efektywniejszym wykorzystaniu jego zasobów. Tim-Berners Lee[1] zaproponował ideę sieci semantycznej[2], w której zasoby są powiązane ze sobą znaczeniowo. Dzięki temu podejściu prezentowana informacja mogłaby być przetwarzana oraz interpretowana zarówno przez człowieka jak i maszyny.

W3Cache Consortium[3] zaproponowało wizję światowej pajęczyny sieci semantycznej. Do jej realizacji niezbędny jest efekt „kuli śnieżnej", czyli powszechnego zastosowania technologii semantycznych w światowych zasobach Internetu. Kluczowym elementem w konstrukcji globalnej semantycznej pajęczyny jest pojęcie ontologii. Ontologia jest pojęciem interdyscyplinarnym i jako takie zyskało wiele znaczeń. W informatyce służy do reprezentowania wiedzy poprzez zdefiniowanie zbioru pojęć i relacji miedzy nimi. Jest swoistym meta opisem dla danych. W niniejszej publikacji ontologia stanowi opis

zagadnienia w określonej dziedzinie. Reprezentowana jest przez graf powiązań pomiędzy koncepcjami związanymi z modelowanym problemem i opisanych w języku OWL[4].

Głównym problemem, z jakim borykają się twórcy zasobów opisanych semantycznie jest pozyskanie lub wytworzenie odpowiedniego zbioru powiązanych z nimi pojęć. Istniejące w sieci gotowe rozwiązania, pomimo iż są precyzyjne dla określonych dziedzin, implementowane są w różnych językach, a także realizują często bardzo odmienne spojrzenie na problem. Kluczowym więc elementem jest tutaj jednorodna płaszczyzna pojęciowa obejmująca wszystkie wykorzystywane zasoby.

Jednym z najbardziej znanych narzędzi do wytwarzania ontologii jest Protégé[5][6][7]. Edytor ten nie był jednak tworzony z myślą o pracy grupowej nad ontologiami. Co prawda równolegle do prezentowanego w tej publikacji systemu powstało rozszerzenie do Protégé zwane Collaborative Protégé[8], jednak nie przewiduje ono funkcjonalności składowania ontologii w ogólnodostępnym repozytorium czy też kontroli nad kształtem ontologii przez jej właściciela – przyjęto model głosowania nad zmianami.

W niniejszej publikacji zaprezentowano narzędzie OCS[9] do konstruowania ontologii w środowisku rozproszonym umożliwiające uzgadnianie wspólnej warstwy konceptualnej. Opisywany system służy do kolaboracyjnego wytwarzania ontologii, a także ich składowania i udostępniania innym systemom.

Charakterystyka systemu

Podstawową funkcjonalnością systemu jest możliwości edycji ontologii, czyli tworzenia, modyfikacji oraz usuwania klas, atrybutów oraz relacji. Ponadto system umożliwia trwałe przechowywanie za jego pomocą tworzonych i modyfikowanych ontologii, a także umożliwić podgląd i modyfikację na dowolnym etapie ich konstrukcji. Operacje te odbywają się dzięki zaimplementowanych mechanizmom wersjonowania podobnych do znanych z np. Subversion[10], jednak operujących na poziomie semantycznym.

Ważnym aspektem systemu jest możliwość pracy kolaboracyjnej. Narzędzie umożliwia pracę grupową nad ontologiami. Modyfikacje do ontologii wprowadzać może dowolny zarejestrowany użytkownik. Każda modyfikacja rejestrowana jest jako sugestie zmian, które z kolei muszą zostać zaakceptowane przez użytkowników odpowiedzialnych za stan danej ontologii. Dopiero po tym fakcie sugerowane zmiany stają się integralną częścią opracowywanego rozwiązania wchodząc w skład nowej wersji ontologii. Dzięki temu, pomimo pracy nad ontologią dowolnie dużej grupy ludzi, ostateczne rozwiązanie zachowuje spójność oraz minimalizowane jest ryzyko wprowadzenia fałszywych informacji. Proces

MOST WIEDZY Downloaded from mostwiedzy.pl

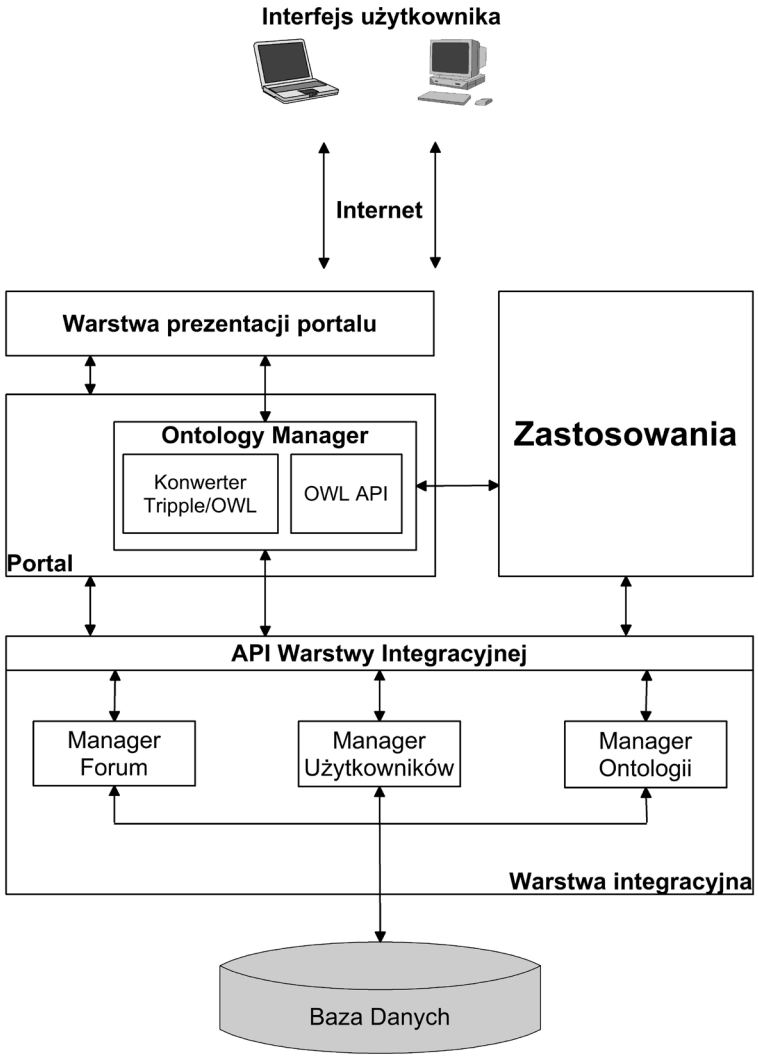
korekty konfliktów pomiędzy propozycjami zmian wykonywany jest ręcznie przez autora sugestii.

W przypadku pracy nad dużymi ontologiami przewidziano możliwość podziału prac między różnych ekspertów, zajmujących się jedynie fragmentami całego rozwiązania. Każdy z ekspertów rozpatruje jedynie swój fragment ontologii z właściwymi mu propozycjami zmian.

Drugim elementem wspomagającym pracę grupową nad ontologią jest forum dołączone do każdej z opracowywanych ontologii. Za jego pomocą użytkownicy systemu będą mogli prowadzić dyskusje nad kształtem poszczególnych elementów ontologii czy kierunku, w jakim zmierzać powinno docelowe rozwiązanie.

Architektura systemu

Ogólna architektura portalu zaprezentowana jest na Rys. 1.



Rys. 1: Architektura systemu OCS

MOST WIEDZY Downloaded from mostwiedzy.pl

System składa się z 4 głównych komponentów. Są to:

- Warstwa integracyjna wraz z bazą danych – odpowiedzialnej za przechowywanie ontologii, oraz zarządzanie wersjami ontologii oraz użytkownikami systemu, steruje dostępem do danych przechowywanych w systemie. Ontologie przechowywane są w bazie danych w postaci zbioru trójek. Ten moduł dostarcza też całej logiki biznesowej związanej z zarządzaniem ontologiami oraz ich wersjami, dostępem do danych, zarządzaniem użytkownikami itp. Dostęp do przechowywanych w bazie danych odbywa się dwojako: bezpośrednio poprzez EJB lub WebServices oraz za pośrednictwem biblioteki OntologyManager. Zarejestrowany użytkownik ma możliwość pobrania dowolnej wersji każdej z ontologii i wykorzystać ją w dowolnym zewnętrznym zastosowaniu. Warstwa integracyjna, poprzez mechanizmy tworzenia ontologii może być wykorzystywana przez systemy automatycznego generowania ontologii na podstawie automatycznie przetwarzanych informacji o zewnętrznym świecie.
- OntologyManager – będący biblioteką przetwarzającą ontologię z postaci trójkowej (przechowywanej w bazie danych) do postaci obiektowej wykorzystywanej w samym edytorze, czyli obiektów OWL API[11][12]. W dużej mierze odpowiada za komunikację z Warstwą integracji opakowując bezpośrednie wywołania poprzez EJB. Dzięki tej bibliotece możliwe jest korzystanie z zasobów repozytorium ontologii bez konieczności samodzielnego konwertowania reprezentacji ontologii z trójkowej na obiektową i odwrotnie. Umożliwia też bezpieczne zdalne połączenie z repozytorium dzięki zastosowaniu szyfrowania SSL do wszystkich przesyłanych komunikatów. Pozwala również na generowanie propozycji zmian, służących do zgłaszania poprawek do ontologii. Poprzez bibliotekę narzędziową dostępne jest całe API udostępniane przez warstwę integracyjną.
- Portal internetowy – graficzny edytor ontologii. Umożliwia prace offline jak i online – połączenie z siecią konieczne jest jedynie w momencie pobierania projektu z serwera oraz zgłaszania propozycji zmian do ontologii. Umożliwia wizualizację wytworzonej ontologii dzięki zastosowaniu biblioteki Prefuse[13]. Z edytorem zintegrowano również bibliotekę wnioskującą Pellet[14], co umożliwia graficzny podgląd pełnej, wywnioskowanej hierarchii rozszerzonej o elementy wyrażone nie wprost za pomocą właściwości obiektów i ich wzajemnych powiązań.

MOST WIEDZY Downloaded from mostwiedzy.pl

- Moduły zastosowań – będące elementami zewnętrznymi wykorzystujące w praktycznych zastosowaniach ontologie wytworzone w systemie. Jako moduły zastosowań traktuje się wszystkie zewnętrzne systemy korzystające z repozytorium ontologii zarówno poprzez bezpośrednie wywołania EJB czy WebServices czy tez za pośrednictwem Ontology Managera.

Baza danych

Wybór platformy bazodanowej i jej optymalizacja była kluczowym elementem systemu.
Projekt musiał uwzględnić następujące istotne czynniki:

- ontologie są zbiorami potencjalnie bardzo dużymi, w rozbiciu na krotki ontologia może składać się z setek tysięcy rekordów,
- system powinien przechowywać wszystkie wersje ontologii umożliwiając pełny wgląd ekspertom na każdym etapie jej rozwoju,
- pojedyncze operacje zgłaszania sugestii, promocji sugestii, publikacji wersji czy kopiowania ontologii wymagają kosztownych operacji na wielu rekordach,
- użytkownicy i eksperci systemu mogą pracować równolegle nad wieloma ontologiami czy też wersjami jednej ontologii.

Do realizacji warstwy bazodanowej wybrana została baza IBM DB2 pureXML[15]. Wykorzystanie przemysłowej bazy danych podyktowane było: ogromną ilością rekordów jakie będzie przechowywał docelowy system, zapewnieniem szybkiego dostępu do bazy, jak również bogatym wachlarzem narzędzi administracyjnych (zdalne zarządzanie, monitoring, backup).

Dzięki użyciu bazy IBM DB2 tabele bazodanowe są kompresowane, co znacząco redukuję liczbę fizycznych operacji odczytów i zapisów (szybkość dysków ms, kompresji danych w pamięci operacyjnej - szybkość rzędu ns). Dzięki temu zmniejszono czas odczytów/zapisów jak i zredukowano ilość potrzebnego miejsca na dysku.

IBM DB2 pureXML oferuje również przechowywanie danych w postaci XML co może być przydatne przy przechowywaniu stabilnych wersji ontologii jako pojedynczych plików XML. Przechowywanie ontologii w postaci jednego dużego pliku nie wymaga skomplikowanych operacji pobierania ontologi z bazy danych, konwertowania jej na model pośredni a z modelu pośredniego na OWL API.

MOST WIEDZY Downloaded from mostwiedzy.pl

Oprócz optymalizacji wydajności poprzez wybór przemysłowego silnika bazodanowego, dokonano także logicznej optymalizacji procesu przetwarzania i przechowywania ontologii. W tym celu opracowano model bazodanowy w którym zmaksymalizowano relacje wiele do wielu.

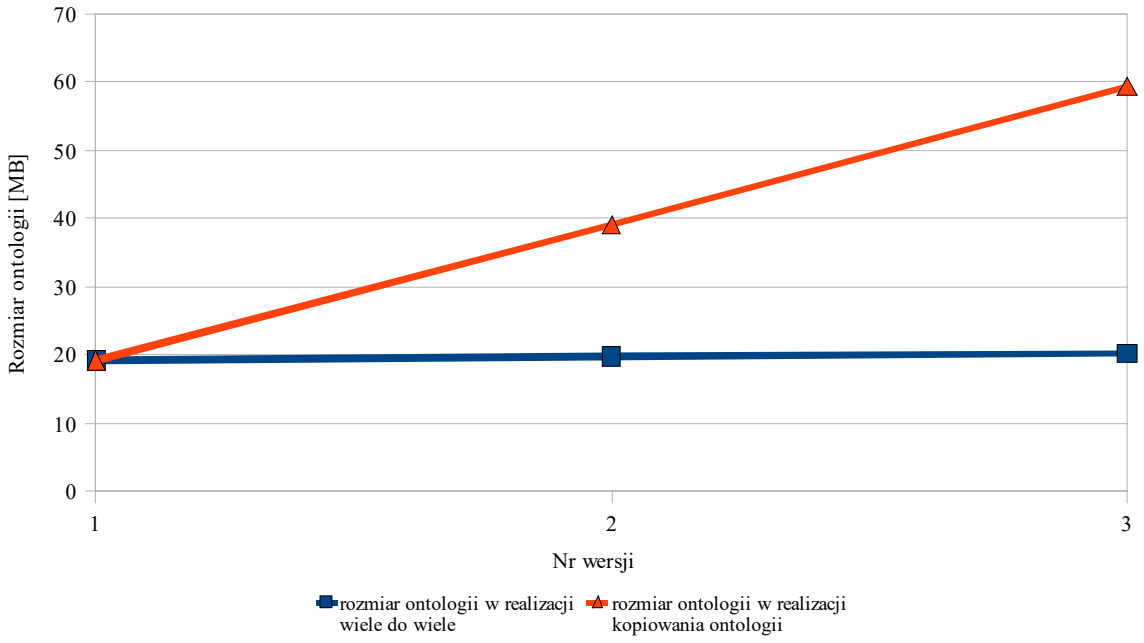
Ontologia może mieć wiele wersji. Każda z wersji składa się z krotek wersji (konsekwencja realizacji relacji wiele do wielu), które to łączą wersje z krotkami. Dzięki temu krotka może należeć do wielu wersji jednocześnie. W przypadku promowania krotki do kolejnej wersji w tablicy krotek wersji wstawiane są rekordy łączące krotki z nową wersją.

W przypadku dodania nowej krotki, zapisywane są dwa nowe rekordy: rekord krotki oraz rekord krotki wersji wiążący nowo dodaną krotkę z obecną wersją ontologii.

W przypadku usunięcia krotki z nowej wersji w tablicy krotek wersji nie jest wpisywany żaden rekord.

Na Rys. 2 przedstawiono porównanie implementacji przechowywania wersji i krotek poprzez 1) kopiowanie wszystkich krotek podczas zatwierdzania nowej wersji z 2) wybraną do realizacji w systemie OCS, opisaną powyżej, implementacją.

Ostatnim etapem optymalizacji wydajnościowej warstwy bazodanowej było jej wdrożenie na serwerze aplikacji z automatycznym menadżerem transakcji oraz dostatecznie dużą pamięcią podręczną pozwalającą przechowywać setki tysięcy obiektów. W tym celu zastosowano serwer Apache Geronimo w wersji 2.1[16].



Rys. 2: Zależność rozmiaru ontologii w bazie danych od nr wersji

MOST WIEDZY Downloaded from mostwiedzy.pl

Konwersja reprezentacji ontologii

Edytor Ontologii systemu OCS korzysta z biblioteki OWL API. Podczas pracy nad ontologią wszelkie operacje wykonywane są na jej obiektowej reprezentacji, natomiast przechowywanie ontologii możliwe jest dopiero po jej eksporcie. Jako że formaty XML, wspierane przez OWL API, słabo nadają się do przechowywania w bazach danych podjęto decyzję o wykorzystaniu formatu trójkowego. W formacie tym ontologia jest listą trójek opisujących relację pomiędzy jej elementami. Podejście to pozwala na użycie relacyjnej bazy danych jako repozytorium ontologii – każda trójka może być pojedynczym wierszem w tabeli bazy danych. Rozwiązanie to pozwala też na łatwe generowanie i przechowywanie wspomnianych propozycji zmian, bez konieczności przechowywania całej zmienionej ontologii.

W związku z powyższym pojawiła się konieczność konwersji reprezentacji ontologii. Przy jej zapisie, przekształcana jest z postaci obiektowej, używanej w edytorze, do formatu trójkowego, używanego w bazie danych będącej repozytorium ontologii. Przy wczytywaniu ontologii z bazy danych do edytora konieczna jest konwersja odwrotna.

W trakcie odczytu ontologii z bazy pobierana jest lista trójek opisujących ontologię oraz dodatkowe elementy takie jak jej URI. Tworzona jest pusta ontologia w postaci obiektu obsługiwanego przez OWL API. Następnie trójki z listy są pojedynczo konwertowane na odpowiednie aksjomy i dodawane do wcześniej utworzonego obiektu reprezentującego ontologię. Po przetworzeniu wszystkich trójek obiekt ten zawiera pełną reprezentację ontologii i jest zwracany jako wynik parsowania.

Konwersja odwrotna przebiega analogicznie. Z obiektowej reprezentacji ontologii kolejno odczytywane są aksjomy a następnie generowana jest ich postać trójkowa. W wyniku tej operacji ontologia zwracana jest w postaci listy trójek odpowiadającym wszystkim aksjomom, a następnie zapisywana w bazie danych.

Dzięki trójkowemu formatowi reprezentacji ontologii możliwa jest efektywna implementacja tworzenia, przechowywania, pobierania i wprowadzania zmian ontologii. Tworzenie propozycji odbywa się poprzez porównanie trójkowej reprezentacji ontologii przed i po wprowadzeniu zmian. Sprowadza się ono do porównania ich w celu ustalenia, które trójki zostały dodane, a które usunięte. W wyniku tej operacji powstaje lista trójek wzbogacona o informację o konieczności usunięcia bądź dodania danej trójki z reprezentacji ontologii w celu wprowadzenia zmian. Propozycje reprezentowane w tym formacie są łatwe do przechowania w repozytorium, przygotowanym do obsługi trójek.

Poprzez zapis trójkowy jest również możliwe pobieranie zgłoszonych propozycji zmian oraz prezentowanie ich na obecnie wczytanej ontologii. Trójki te dzielone są na grupę

MOST WIEDZY Downloaded from mostwiedzy.pl

trójek, które należy dodać do ontologii oraz takich, które należy usunąć z ontologii. Na ich podstawie generowane są odpowiadające im aksjomy w postaci obiektów OWL API. W kolejnym kroku są one umieszczane na listach AddAxiom i DeleteAxiom. Na tej podstawie, korzystając z metod OWL API, możliwe jest zmodyfikowanie bieżącej ontologii tak by odzwierciedlała wybrane bądź wszystkie zmiany zasugerowane przez użytkowników systemu.

Podsumowanie

Ontologie są jednym ze sposobów na wzbogacenie treści znajdujących się w internecie o znaczenie zrozumiałe dla komputerów. Aby tak się jednak stało konieczne jest powszechne ich wykorzystanie w oparciu o spójną bazę pojęciową. W tym celu niezbędne jest kolaboracyjne wytwarzanie tychże ontologii. Dzięki temu gotowy produkt będzie uwzględniał punkt widzenia wielu środowisk, a nie tylko jednego autora. Prezentowane narzędzie zaprojektowane zostało z myślą o pracy grupowej. Łączy w sobie możliwość wpływu na kształt generowanej ontologii ze strony wielu osób z kontrolą kierunku zmian przez jej właściciela. Dzięki temu autor ontologii jest w stanie uwzględnić znacznie szerszą bazę pojęciową, niż gdyby samodzielnie pracował i rozwijał wybraną ontologię. Z drugiej strony nadzór jednej lub kilku osób nad jej ostatecznym kształtem gwarantuje jej spójność i użyteczność w rzeczywistych zastosowaniach.

Bibliografia:

1. Berners-Lee T., Hendler J., Lassila O.: *The Semantic Web*, Scientific American, May 2001
2. The Semantic Web Community Portal: http://semanticweb.org 2008
3. Semantic Web – W3C: http://www.w3.org/2001/sw/ 2001
4. OWL Web Ontology Language Overview: http://www.w3.org/TR/owl-features/ 2004
5. Stanford Center for Biomedical Informatics Research: http://protege.stanford.edu/ 2009
6. Gennari J. H., Musen M. A., Fergerson R. W., Grosso W. E., Crubézy M., Eriksson H., Noy N. F., Tu S. W.: *The evolution of Protégé: An environment for knowledge-based systems development*, Technical Report SMI-2002-0943, Stanford Medical Institute, 2002

MOST WIEDZY Downloaded from mostwiedzy.pl

7. Noy N. F., Fergerson R. W., Musen M. A.: *The knowledge model of Protégé-2000: Combining interoperability and flexibility*, Lecture Notes in Computer Science, Springer-Verlag, 2000
8. Stanford Center for Biomedical Informatics Research: http://protegewiki.stanford.edu/index.php/Collaborative_Protege 2009
9. Boiński T.: *Ontology portal: architecture of domain oriented ontology portal*, Gdańsk University of Technology, 2008 [in Polish]
10. CollabNet: http://subversion.tigris.org/ 2008
11. Horridge M., Bechhofer S., Noppens O.: *Igniting the OWL 1.1 Touch Paper: The OWL API*, OWLED 2007, 3rd OWL Experienced and Directions Workshop, 2007.
12. Bechhofer S., Lord P., Volz R.: *Cooking the Semantic Web with the OWL API*, 2nd International Semantic Web Conference, 2003
13. Prefuse.: http://prefuse.org/ 2009
14. Sirin E., Parsia B., Grau B. C., Kalyanpur A., Katz Y.: *Pellet: A practical OWL-DL reasoner*, Journal of Web Semantics, 2007
15. IBM: http://www-01.ibm.com/software/data/db2/9/ 2009
16. The Apache Software Foundation: http://geronimo.apache.org/ 2008

MOST WIEDZY Downloaded from mostwiedzy.pl

mgr inż. Tomasz Boiński, mgr inż. Łukasz Budnik, mgr inż. Andrzej Jakowski, Jacek Mroziński, Krzysztof Mazurkiewicz
Department of Computer Architecture
Faculty of Electronics, Telecommunication and Informatics
Gdańsk University of Technology

OCS – Domain oriented ontology creation system

Abstract

Ontologies are, as a part of semantic web, a basic component to enrich content available in the Internet with a meaning understandable for computers. In this publication an architecture and functionality of a domain oriented web portal for collaborative creation and storage of ontologies is presented. The system is being created by Gdańsk University of Technology Electronics, Telecommunications and Informatics faculty's Department of Computer Architecture. Most important elements of presented system that distinguish it form other similar solutions are shown. The method for storing data and database access is described as well. Object to triple and triple to object conversion and methodology for collaborative ontology development used in presented system was described as well.

Keywords: ontology, versioning, development

Introductions

During expansion of the Internet what was considered as its strength – simplicity of a HTML language, became an obstacle preventing us from better and more effective usage of Internet's resources. Tim-Berners Lee[1] proposed an idea of a semantic web[2] where resources are connected with each other throughout a meaning. Thanks to that, information stored in and presented through the Internet could be processed by both human and machines.

W3Cache Consortium[3] proposed a vision of a world wide semantic web. For realization of this vision a "snowball" effect is needed, which is a wide usage of semantic oriented technologies in world's resources available in the Internet. A key element for constructing such a global semantic web are ontologies. Ontology is an interdisciplinary idea and as such gain a lot of meanings. In computer science it is used to represent knowledge by defining a set of concepts and relations between them. It can be considered as a meta-language for data description. In this publication ontology is used as a description of a part of

MOST WIEDZY Downloaded from mostwiedzy.pl

knowledge from a given domain. It is being represented as a graph of relations between concepts connected with modeled problem and described in OWL[4] language.

The main problem that all developers of semantically oriented resources face is a difficulty of obtaining or creation of concepts related to that resource. Throughout the web there are many solutions that usually are very precise for given domain but implemented in a language or presenting a point of view that is not compatible with the one needed for current problem. Thus, a common concept layer for all resources is needed.

One of the most well known tools for ontology creation is Protégé[5][6][7]. However, it was not created with collaboration in mind. There is a plug-in called Collaborative Protégé[8] which is developed for Protégé. However, it doesn't take into consideration a possibility of storing ontologies in a common repository or any more complex mechanisms for managing conflicts and changes in the created ontology or its owner control over development porcess – a simple voting mechanism is implemented.

In this publication OCS[9], a tool for constructing ontologies in distributed environment, is presented. It allows to reach an agreement for common conceptual layer. The system described enables users to develop, store and share ontologies with other systems.

System characteristics

Basic functionality of the presented system is a possibility of editing ontologies, i.e. creating, modifying and removing of classes, attributes and relations between them. The system also allows persistent storage of ontologies that were created and modified with the use of described portal as well access to any version of given ontology at any time. It is possible thanks to versioning mechanisms similar to those known from e.g. Subversion[10], but working on a semantic level instead of syntactic level.

An important aspect of this system is its possibility of a collaborative work on ontologies. Modification to ontologies can be performed by any registered user. Each modification is stored on the server as a proposition which needs to be accepted by any user responsible for the shape of the ontology e.g. its owner. When the suggested changes become an integral part of the edited ontology and new version of the ontology is created. This process ensures that even in situations where many developers are working on single ontology it remains consistent and risk of introducing false information is minimized. When two or more propositions conflict with each other they creators need to adjust them manually.

MOST WIEDZY Downloaded from mostwiedzy.pl

When working on big ontologies the responsibility for accepting propositions can be divided across an ontology owner and one or more experts. Each expert takes responsibility for some part of the ontology and processes only propositions valid for that part.

Another element supporting collaborative work is a forum attached to each ontology. With its help users will be able to discuss the shape of each ontology element or a direction in which the developed ontology should tend.

System architecture

General portal architecture is presented on Fig. 1.

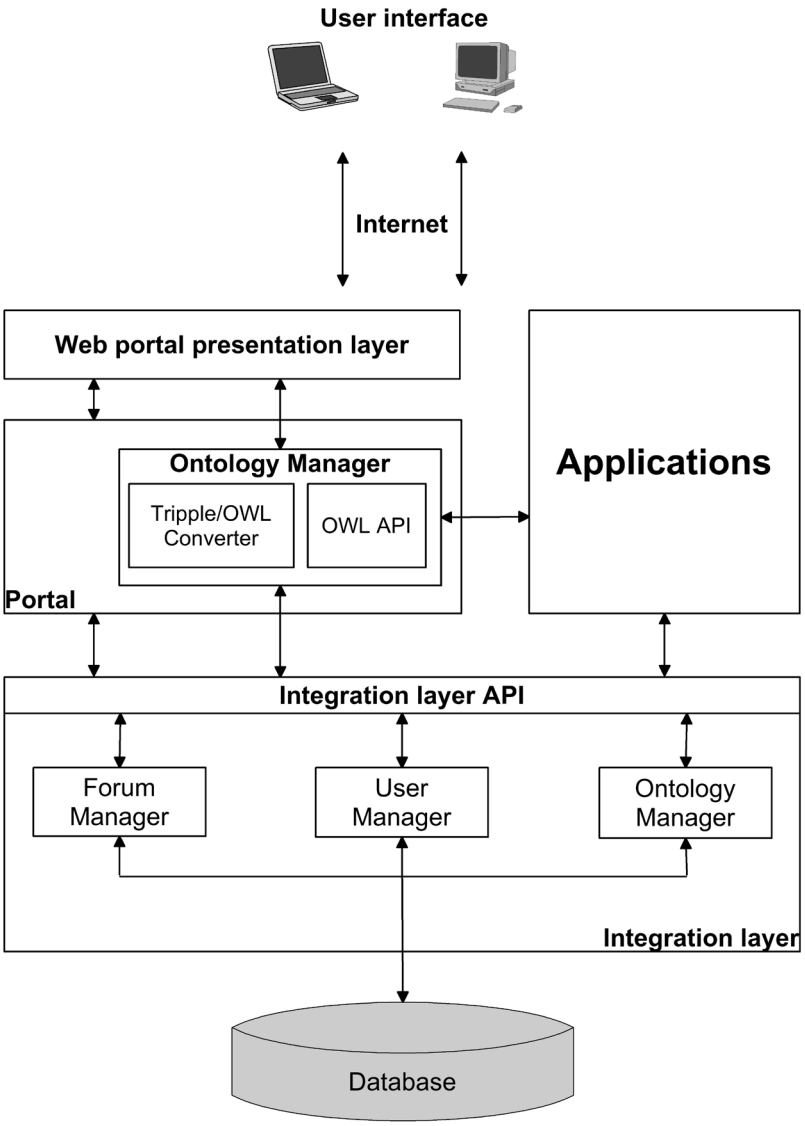


Fig. 1: OCS system architecture

MOST WIEDZY Downloaded from mostwiedzy.pl

It consists of four main components:

- Integration layer with database – responsible for ontology storage, user and ontology versions management, data access. Ontologies are being stored in database as a set of triples. Data stored withing database can be accessed by two means: directly by EJB or WebServices and using utility library (OntologyManager) described later. This module also provides whole business logic connected with management of ontologies and their versions, database access, user management etc. Registered user can download any version of any ontology and use it in any external system. All ontology creation mechanisms are also available through API so any external system for ontology creation, even automated ones.
- OntologyManager – a library for converting ontology representation from triple format (stored in database) to OWL API[11][12] object format used in editor. OntologyManager is also responsible for communication with Integration layer encapsulating EJB calls. All connections with Integration layer are encapsulated in SSL introducing security for password transmission. OntologyManager supports all operations available through Integration layer's API. This way library can be used in any other project to communicate with ontology repository. External systems can than operate on ontologies stored within Integration layer without need to implement classes responsible for communication and conversion between ontology representations.
- Web portal – graphical ontology editor. It allows both off-line and on-line work – network connection is needed only during project checkout and committing propositions of changes. Web portal uses Prefuse[13] library to visualize ontologies as well as reasoned hierarchy created by Pellet[14] library. Pellet enables users to see full hierarchy with all indirect relationships taken into consideration. It helps in finding inconsistencies across ontologies and other errors.
- Application modules – external elements using in practice ontologies developed and stored in described system. As such all external systems that connect to repository, both through OntologyManager and directly via EJB/WebServices, are treated.

Database

Proper selection of database engine and it's further optimization was a key element of system's design process. The following facts needed to be taken into consideration:

MOST WIEDZY Downloaded from mostwiedzy.pl

- ontologies can consist of very large number of triples,
- all versions of a given ontology should be stored allowing insight into the history of its changes at any point during ontology development,
- every act of creation of suggestion of changes, suggestion promotion, creation of new version or copying of ontology requires many operation on multiple records,
- users and experts can work simultaneously on many ontologies or on many versions of single ontology.

As a database engine IBM DB2 pureXML[15] database was selected. Usage of commercial database is supported by: big number of records that need to be stored by the system, need for fast access to data, good support in terms of administrative tools, remote access etc.

IBM DB2 supports compression of database tables which reduces number of physical read/write operations – less data needs smaller number of operations on physical storage which is treated as slow access memory. It also reduces space occupied on a hard disk.

IBM DB2 pureXML can also store ontologies as an XML file. This can be used to store stable versions of ontologies as a simple RDF/XML file. Than it can be easily exported to any other system without a need of performing costly conversion form triples to OWL API and then to XML file.

To further improve performance and lower disk space usage database structure was developed in such a way which maximizes the usage of many-to-many relations.

Each ontology can have many versions. Each version consists of version triples linking ontology with actual triples. This way one triple can belong to many ontologies or ontology version at the same time. When proposition is moved to a newer ontology version its triples are only joined with a new version instead of being copied, thus limiting disk space usage.

When a new triple is added two new records are stored: the triple itself and join between ontology version and the triple. When triple is removed from ontology version only the link is deleted. Triple itself is never deleted as it is used in some earlier versions.

Fig. 2 presents comparison of implementations in regard of disk usage when triples are being copied with each new version and when many-to-many implementation is used.

MOST WIEDZY Downloaded from mostwiedzy.pl

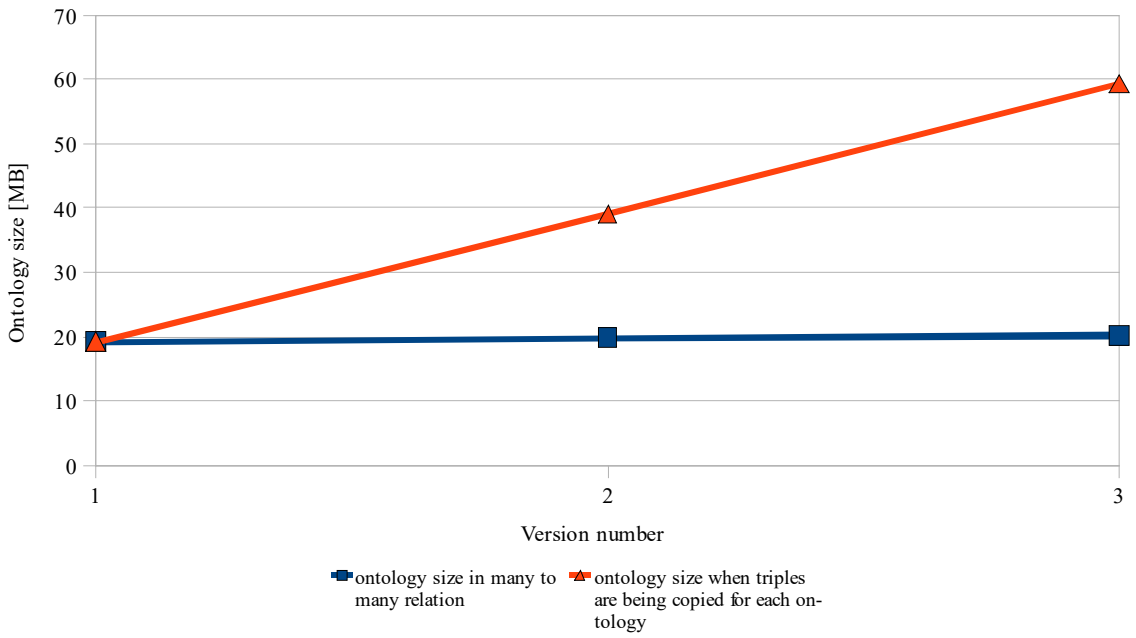


Fig. 2: Function of ontology size in respect of version number

The last step of database optimization was its deployment on application server that supported automatic transaction manager and enough cache memory to store hundreds of thousands of records. For that purpose Apache Geronimo 2.1[16] was used.

Ontology representation conversion

Ontology editor in OCS system uses OWL API library. During ontology development all work is done on its object representation. To store such ontology in a database, it needs to be exported. OWL API supports XML format but its application as way of storing data in a database is limited. Thus it was decided to use simple triple format as most basic way of storing ontologies. In this way a relational database can be used for data storage – each triple can be easily represented as a single row in appropriate table. Treating ontology as set of triples also simplifies ontology comparison, generation and storage of propositions of changes, etc.

Due to the above reasons, there is a need for representation conversion. During writing in a database ontology is converted form OWL API objects into triples. During checkout form repository reverse operation is performed.

When retrieving ontology from repository, a set of triples representing all axioms within ontology is being read as well as some additional data like ontology URI. An empty ontology is created as an OWL API object. Next, every triple from the set is converted into OWL API axiom and added to ontology created. earlier After all the triples were converted, in

MOST WIEDZY Downloaded from mostwiedzy.pl

the mentioned object fully restored OWL API ontology is contained and than returned as result of the parser.

Inverse conversion is done analogical. From object representation every axiom is being read and converted into matching set of triples. Those sets are than combined into one set and returned as a result of conversion.

Thanks to triples used as a basic format for ontology representation, implementation of creation, storage and modification of ontologies can be done easily and efficiently. When a new proposition of changes is created, simple comparison of triple sets before and after changes is performed. Sets of triples are compared to find out which triples were added and which were deleted. As a result, a list of triples is enriched with information whether given triple should be added to or removed from the previous version of the ontology. Propositions prepared in this way are very easy to store within database that was designed with triples in mind.

Storing propositions as triples also simplifies presentation of changes on the ontology. When proposition of changes is read, triples are divided into two groups depending on whether they should be added or removed from the ontology. Based on those groups, suitable axioms in the form of OWL API objects are created and applied to current ontology so that it would contain some or all changes suggested by users.

Summary

Ontologies are one of the means of enriching resources available through the Internet with a meaning understandable by computers. The need of having common concept layer, which would be widely used, cannot be fulfilled when ontologies are being developed individually without any concern for other groups' point of view. The system described in this publication was designed with collaboration in mind. It combines the possibility of influencing the shape of an ontology by any user with control over its future by ontology owner. This way ontology developers can take into account much winder concept base than when working without community, but still guarantee consistency of the created ontology. Such ontologies can be used in many applications not only the one it was designed for.

Bibliography:

1. Berners-Lee T., Hendler J., Lassila O.: *The Semantic Web*, Scientific American, May 2001
2. The Semantic Web Community Portal: http://semanticweb.org 2008

MOST WIEDZY Downloaded from mostwiedzy.pl

3. Semantic Web – W3C: http://www.w3.org/2001/sw/ 2001
4. OWL Web Ontology Language Overview: http://www.w3.org/TR/owl-features/ 2004
5. Stanford Center for Biomedical Informatics Research: http://protege.stanford.edu/ 2009
6. Gennari J. H., Musen M. A., Fergerson R. W., Grosso W. E., Crubézy M., Eriksson H., Noy N. F., Tu S. W.: *The evolution of Protégé: An environment for knowledge-based systems development*, Technical Report SMI-2002-0943, Stanford Medical Institute, 2002
7. Noy N. F., Fergerson R. W., Musen M. A.: *The knowledge model of Protégé-2000: Combining interoperability and flexibility*, Lecture Notes in Computer Science, Springer-Verlag, 2000
8. Stanford Center for Biomedical Informatics Research: http://protegewiki.stanford.edu/index.php/Collaborative_Protege 2009
9. Boiński T.: *Ontology portal: architecture of domain oriented ontology portal*, Gdańsk University of Technology, 2008 [in Polish]
10. CollabNet: http://subversion.tigris.org/ 2008
11. Horridge M., Bechhofer S., Noppens O.: *Igniting the OWL 1.1 Touch Paper: The OWL API*, OWLED 2007, 3rd OWL Experienced and Directions Workshop, 2007.
12. Bechhofer S., Lord P., Volz R.: *Cooking the Semantic Web with the OWL API*, 2nd International Semantic Web Conference, 2003
13. Prefuse.: http://prefuse.org/ 2009
14. Sirin E., Parsia B., Grau B. C., Kalyanpur A., Katz Y.: *Pellet: A practical OWL-DL reasoner*, Journal of Web Semantics, 2007
15. IBM: http://www-01.ibm.com/software/data/db2/9/ 2009
16. The Apache Software Foundation: http://geronimo.apache.org/ 2008

MOST WIEDZY Downloaded from mostwiedzy.pl