AN ARCHITECTURE FRAMEWORK FOR ONTOLOGY DEVELOPMENT

Gang Zhao, Jijuan Zheng and Robert Meersman
STARLab, Vrije Universeiteit Brussel
Pleinlaan 2, Brussels, Beligum

ABSTRACT

This paper identifies requirements for an ontology development platform to facilitate methodical ontology engineering and ontology application development. It introduces the DOGMA ontology framework, developed with insights from semantic modeling and methodology in database engineering. Based on this framework, an architectural paradigm is put forward in view of ontology engineering and development of ontology applications and a development portal designed to support ontology engineering, content authoring and application development with a view to maximal scalability in size and complexity of semantic knowledge and flexible reuse of ontology models and ontology application processes in a distributed and collaborative engineering environment.

KEYWORDS

Ontology, semantic processing, ontology portal, architecture, methodology

1. INTRODUCTION

This paper summarizes the DOGMA (**D**eveloping **O**ntology-**G**uided **M**ediation for **A**gents) framework and our exploration of one of the architectural paradigms of the framework, with focus on the problems and requirements in the current research projects (STARLab, VUB). It is organized into six sections on the subjects of requirements, DOGMA framework, MCU architecture, ontology development portal.

2. MAJOR REQUIREMENTS OF ARCHITECTURE

This section reviews briefly some major requirement issues for the current study. *Ontology* is an approximate linguistic representation of agreed conceptualization concerning a subject matter. *Ontology development* means ontology modeling, content authoring and application development.

2.1 Modular, scalable, collaborative and evolutionary modeling

The supporting tools and infrastructure of ontology modeling needs to support managing model complexity with modularization and stratification, scalability in complexity and size of knowledge, managing evolution and change in an iterative modeling cycle and collaborative distributed effort.

2.2 Interoperability across systems

Here *system* means any semiotic system, such as database schemas, languages, ontologies, network systems. Interoperability issues encountered in our study are multi-linguality with ontology, information retrieval from heterogeneous databases, communication between agents and processes.

2.3 Reusability of ontology

The reuse of ontology is different from reuse in component-based software development. Besides needs for semantic transparency, the ontology reuse tends to be partial and dynamic. It can be adoptive, without transforming or modifying the semiotics, augmentative and adaptive, translating or modifying the semiotics.

2.4 Development of ontology-based applications

We are working on the scenario of using ontology in knowledge system applications, such as ontology-based information extraction, ontology-based decision support, ontology-based HCI and natural language processing. Ontology is seen as abstract model underlying database or knowledge base to facilitate a large-scale information and knowledge engineering. In addition, it can also be regarded as part of knowledge base, contributing to inferencing, problem-solving search, justifying and explaining an automatic conclusion and advice.

3. DOGMA REPRESENTATION FRAMEWORK

This section describes the key concepts of DOGMA framework for ontology representation. It will be the basic constructs of ontology development architecture.

3.1 Lexon and lexon base

Lexon is a quintuple $< g , t_1, r_1, t_2, r_2>$ where $g \in \Gamma$ is a *context* identifier, $t_1 \in T$ and $t_2 \in T$ are *terms* over alphabet A, $r_1 \in R$ and $r_2 \in R$ are roles in the semantic relationship. Γ, T and R are strings over an alphabet, A^+. With a context identifier $g \in \Gamma$ and $t \in T$, its concept can be uniquely identified. Context is a set of sources referring to some documents. Intuitively, a given document source $g \in \Gamma$ contextualizes ideationally the relationship between two concepts. The semantic validity of the relationship is established through the developer/user agreement on the sources. In fact, the context is ideational. A lexon base, Ω , is a set of expressions composed from a ordered pair $<A, I>$ where $I = \Gamma \times T \times R \times T \times R$ is the set of lexons.

Lexons captures fact types rather than their instances. The ontology base is an ontology of fact types underpinning knowledge and data bases. This model-theoretic paradigm of semantic modeling has great significance to ontology engineering as to database systems (Meersman, R. 2000) (Spyns P. et al, 2002), especially ontology scalability size and complexity during both engineering and processing time.

3.2 Commitment layer

Lexons are not the specification of application-related information about data population or rule-based processes, unlike database schemas or rule bases. They are declarative and static description of relation types underlying application domains. In the DOGMA representation framework, application-specific business rules, data integrity constraints are left to an interface layer between the model, the ontology proper, and applications (Meersman, R., 2000). It is introduced into an otherwise monolithic representation of declarative and operational semantics. In addition, it enables decoupling semantic knowledge from the IT applications, so that the latter makes use of an external ontology for its semantic processing. A comparable approach to data independence brought about successful development of database systems and their client applications.

4. MCU ARCHITECTURE

M stands for *model* as in database terminology. It is a subset of ontology base and a way of structuring the lexon population. *C* is *commitment* to the le xon models with respect to task or task types. *U* is *use* for system-specific configuration of commitments and its processing for a given application.

4.1 Architectural dimension, viewpoints, processes

The ontology base is structured into *models* whereas *commitments* and *uses* are the two architectural dimensions based on the commitment layer of the ontology representation framework. The *model* and pertain to ontology modeling whereas *commitment* and *use* are concerned with ontology application. The former defines what semantics is whereas the latter how the semantics is applied. *Commitment* acts as interface between heterogeneous and general models of ontology and various task-specific intelligent processes in diverse applications.

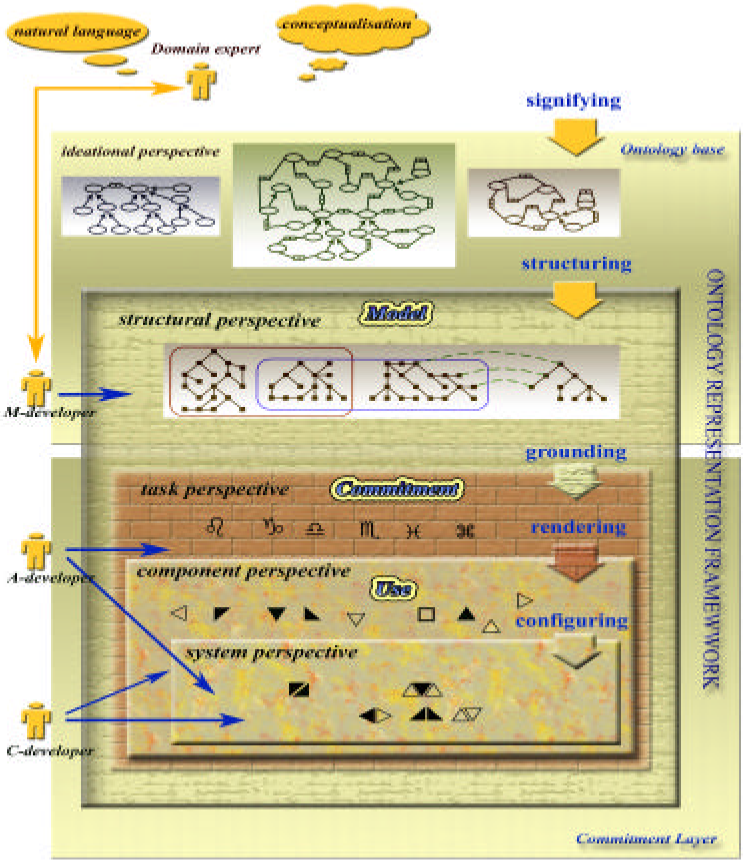

Figure 1. DOGMA MCU Architecture

As depicted in figure 1, the ontology development gives four ontology deliverables: models, commitments, processing components and their assembly, from four development processes: *signifying*, *structuring*, *grounding* and *configuring* in four development contexts: *ideational*, *structural*, *task* and *application*. Across the dimensions of *model*, *commitment* and *use*, there are five architectural viewpoints, *ideational, structural, task*, *component* and *system*. Methodology of ontology development and use are developed to capture the best practices, guidelines, procedures and development management in these viewpoints. The targeted users are model developers (M-developer), application developers who develop and architect commitments and their processing components into uses (A-developer) and content developers who author contents with ontology (C-developer).

4.2 Model

The model is a device to structure the lexon base. It can be defined recursively. A model consists of one or more lexons. It can include one or more models as its constituents. Each lexon enters into direct or indirect relationship with the rest of symbolic entities in this finite semiotic space where "… elements … do not have intrinsic meaning as autonomous entities but derive their significance from oppositions which are in turn related to other oppositions in a process of theoretically infinite semiosis." (Culler, J., 1981).

4.3 Commitment

Committing to ontology is the adoption and interpretation of lexon models to enable particular behaviour in the software agent for a given task in given applications. The adoption is selective (Gruber, T., 1993) and 'a partial semantic account' (Guarino, N. & Giaretta, P., 1995). The interpretation imposes constraints and instantiations on lexon models so that the models are semantically consistent and unambiguous in the context of tasks and applications. A commitment consists of a set of selected lexons and their constraints and instantiations concerning particular tasks. For example, a commitment of ontology for database operations imposes the integrity constraints or other rules over the selected lexons. Commitments can be specified in a commitment specification language, for example, ? -RIDL (extended from RIDL (Meersman, R., 1981)) for ontology-based query over multiple databases.

4.4 Use

The architectural dimension of *use* features software processes using or interpreting commitments. It is divided into two dimensions: *components* and their *assembly*. The components fall into two categories: commitment interpretation and conversion with respect to task types. Interpreting executes commitments and generates the interpretation of lexons with respect to application tasks, thereby, lexons are grounded on particular task semantics in a given application process. Conversion translates, encodes or transforms from one semiotic into another on the basis of commitments. The semiotics can be an ontology model or any other task-specific semantic codes. One example is the translation from lexon models or commitments into formally different but equivalent semiotics, such as ontology exporting, natural language verbalization, translation into SQL or C++ codes aggregation. Another example is understanding or annotation of texts or database schemas in terms of ontology. These components can form into a development framework or library to allow systematic reuse and extension. They are associated processing task or task types. On the *assembly* dimension, these components are configured or pipelined into 'filter and pipes' (Shaw, 1996) to constitute composite semantic operation.

5. ONTOLOGY DEVELOPMENT PORTAL

The ontology portal based on the MCU paradigm is not only a portal to support distributed ontology modeling, but also provides facilities for developing applications of ontology with shared model transformation, commitments, uses, software framework. As depicted in figure 2, it is presented in a style of layered architecture of functional components. It is 3-tiered: presentation, business logic and data. The middle tier, itself, is of n-tier structure.

The ontology development portal supports ontology engineering and development of ontology applications and facilitates collaborative content and model authoring as well as development of knowledge components for a given software system. The presentation tier consists of a set of client-based facilities for the ontology developer to author, design, check in and out, query, browse, program, configure, import and export, merge, share lexon models, their task-oriented commitments and uses for a particular application system. The main functionality of the second tier is data interpretation and management. It processes the requests and operations on the presentation layer, synthesize, validate, translate, and guarantee the integrity of models, commitments and uses. The tier is also responsible for client and communication management.

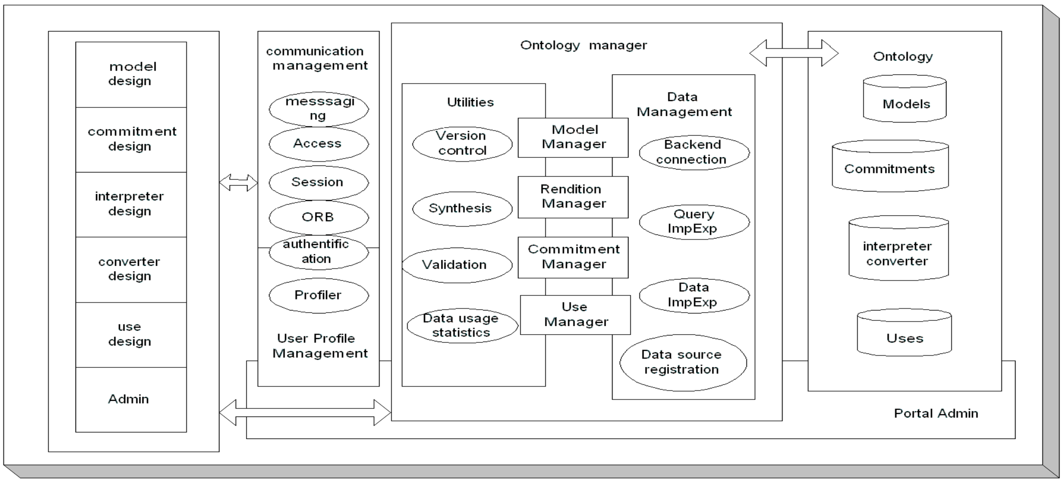

Figure 2. Architecture of Ontology Development Portal

6. CONCLUSION

We introduce the DOGMA ontology representation framework and its MCU architecture. We illustrate their significance in ontology engineering and application with an architecture ontology development portal and service portal, deploying developed outputs. It not only supports ontology sharing as SEAL (Maedche A. et al, 2002), Ontobroker (Fensel D. et al, 1998) and OntoWeb (Zheng J. et al, 2001), but also provides means and layout for specific knowledge application development and ontology deployment for application services, such as mediation in semantic search, agent communication. Among the core requirements of semantic modeling and processing, we emphasize issues such as distributed collaborative engineering, share and reuse of knowledge components on different architectural dimensions, in different development perspectives, scalability in size and complexity, development methodology with specific conceptual framework and architecture.

REFERENCES

Culler, J., 1981. *The Pursuit of Signs: Semiortics, Literature, Deconstruction*. Cornell University Press, New York.

Fensel, D., Decker, S., Erdmann, M., and Studer, R., 1998. *Ontobroker: How to enable intelligent access to the WWW. In AI and Information Integration*, Technical Report WS-98-14, 36-42. Menlo Park, CA: AAAI Press.

Gruber, T., 1993. *Toward Principles for the Design of Ontologies Used for Knowledge Sharing*, Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford University.

Maedche, A., Staab, Studer, R., Sure, Y., and Volz, R., 2002. *SEAL— tying up information integration and web site management by ontologies*. IEEE Data Engineering Bulletin, 25(1):10–17.

Meersman R., 1981. *The RIDL Language*. Technical Report, Control Data DMRL, Belgium.

Meersman R., 2000. *Reusing certain database design principles, methods and techniques for ontology theory, construction and methodology*. STAR Lab Technical Report, Brussel. Available from: - http://www.starlab.vub.ac.be/publications/STAR-2001-01.pdf

Meersman R., 2001. *Ontologies and Databases: More than a Fleeting Resemblance*, in d'Atri A. and Missikoff M. (eds), OES/SEO 2001 Rome Workshop, Luiss Publications.

Shaw, M., 1996. *Some patterns for software Architecture*. In John M. Vlissides, James O. Coplien, & Norman L. Kerth (Eds.), Pattern Languages of Program Design, Vol 2, pp. 255-269. Reading, MA: Addison-Wesley.

Spyns P., Meersman R. & Jarrar M., 2002. *Data modelling versus Ontology engineering*, in Sheth A. & Meersman R. (ed.), SIGMOD Record Special Issue on Semantic Web, Database Management and Information Systems

STARLab, VUB. *Systems Technology and Applications Research Laboratory*, http://www.starlab.vub.ac.be/

Zheng J., Meersman R., Studer R., Sure Y., & Volz R,.2001. *Creation of a browsable prototype of the portal*, OntoWeb Deliverable #6.2, Brussel. Available from: - http://www.starlab.vub.ac.be/research/projects/ontoweb/OntoWebD62.pdf