AN ARCHITECTURE FRAMEWORK OF
ONTOLOGY DEVELOPMENT AND
DEPLOYMENT

Gang Zhao, Jijuan Zheng, Robert Meersman

affiliation
keywords
number
date
corresponding author
status
reference

ontology server architecture
STAR-2003-01
21/01/2003
Gang Zhao
accepted
E-Society 2003 (IADIS International Conference)
AN ARCHITECTURE FRAMEWORK OF ONTOLOGY DEVELOPMENT AND DEPLOYMENT

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, at STARLab, VUB. It has been adopted for ontology modeling and the development of ontology facilities in such EU R&T projects as NAMIC, ONTOWEB, FF POIROT. Based on this framework, an MCU architectural paradigm is put forward in consideration 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
With fast growth of information available electronically and computing resources, more and more challenging and complex requirements and expectations are made of software information systems. One of them is the capability of semantic processing to harness the explosion of information and complexity of system functionality with available software and hardware potentials.

STARLab, VUB has developed an ontology development framework in the DOGMA\textsuperscript{1} project and other EU projects (ex. NAMIC\textsuperscript{2}, ONTOWEB, FF POIROT\textsuperscript{3}, SCOP\textsuperscript{4}). This paper presents the DOGMA framework and introduces our exploration of one of the architectural paradigms of the framework, with focus on the problems and requirements in some of on-going research projects [STARLab, VUB].

It is organized into six sections on the subjects of requirements, DOGMA framework, MCU architecture, ontology development and service portal and future work

2. REQUIREMENTS
In this section, we shall review briefly some major requirement issues for our current study. It is not intended as a comprehensive coverage of the requirements encountered in our projects in STARLab, VUB or in ontology research communities. By ontology development, we mean ontology modeling, content authoring and application development for information or knowledge systems using ontology.

2.1 Modular, scalable collaborative and evolutionary modeling

\textsuperscript{1} Developing Ontology-Guided Mediation for Agents
\textsuperscript{2} News Agencies Multilingual Information Categorisation
\textsuperscript{3} Financial Fraud Prevention-Oriented Information Resources using Ontology Technology
\textsuperscript{4} Semantic Connection of Ontologies to Patient record data, part of the EUREKA project: Secure Care On-line Pocket
Ontology modeling is to recast a conceptualization of domain into particular conceptual constructs and express them in a given language. The main actor is the ontology engineer. The output is a representation of conceptualization, a symbolic system in the sense of semiotics. The representation language needs to be declarative, first-order to capture the primary conceptual constructs to capture entity and relationship [Meersman, R., 1999]. Its supporting tools and infrastructure for managing the resultant symbolic system 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
- collaborative effort, mostly distributed.

### 2.2 Interoperability across systems

Here ‘system’ means semiotic systems including any natural and artificial data or communication processes, such as database schemas, natural and formal languages, ontologies, terminology bases, network systems. Below are listed some interoperability issues encountered in our projects:

- Multi-linguality with ontology
- Information retrieval from heterogeneous databases
- Translation of semiotic systems for communication between agents and processes.

### 2.3 Reusability of ontology

The reuse of ontology is different from reuse in component-based software development. It seems to be a stroke of luck that the scope of reuse follows the interface line pre-defined by the ontology author in the first place, due to infinite justifiable perspectives of the amorphous semantic space. The ontology reuse tends to be thus partial and dynamic.

The scope of reuse must be explicitly defined as given views of ontology. Reuse can be

- Adoptive, without transforming or modifying the semiotics
- Adaptive, translating or modifying the semiotics.

### 2.4 Development of ontology-based applications

Besides Jasper R. and Uschold, M.’s four main use cases of ontology (1999), 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.

In this case, 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 [Peter et al, 2002]. It will be the basic constructs of ontology development architecture.

#### 3.1 Key concepts

#### 3.11 Lexon
Lexon is a quintuple \(<\gamma, t_1, r_1, r_2, t_2>\) where \(\gamma \in \Gamma\) 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. \(\Gamma, T\) and \(R\) are strings over an alphabet, \(A^+\).

The semantic reference of \(t\) in a lexon is of two kinds concept and label types:

\[
C = \text{ct} (g, t) \mid g \in \Gamma, t \in N, N \subset T
\]

\[
L = \text{lt} (g, t) \mid g \in \Gamma, t \in A, A \subset T
\]

where \(N \cap \Lambda = 0\).

It is worth distinguishing the ‘lexical’ terms from the rest, the ‘nonlexical terms’, for its special semantic operations. The lexical term refers to semantic symbols which themselves are strings over the same alphabet, \(A\). Its signification, \(S\), is

\[
S = \Gamma \times A \times P \mid \Gamma \subset A^+, A \subset T, P \subset T, A \cap P = 0.
\]

Given a context \(g \in \Gamma\), the label, \(l \in A\), yields the reference, \(p \in P\) as its semantic interpretation. It is worth noting that lexical terms are used for the interpretation of non-lexical terms.

We can distinguish two types of lexons:

\[
i = <\gamma, t_1, r_1, r_2, t_2> \text{ where } \gamma \in \Gamma, r_1 \in R, r_2 \in R, t_1 \in C, t_2 \in C
\]

\[
b = <\gamma, t_1, r_1, r_2, t_2> \text{ where } \gamma \in \Gamma, r_1 \in R, r_2 \in R, t_1 \in L, t_2 \in C
\]

It is assumed that with a context identifier \(g \in \Gamma\) and \(t \in T\), its concept is uniquely identified. Context is a set of sources identifying some documents. Here documents are interpreted in an abstract sense, similar to text in semiotics, a structure expressed in a given language. Intuitively, a given document source \(g \in \Gamma\) expresses a valid relationship between two concepts. The validity of the relationship is established by user agreeing on the sources.

3.1.2 Lexon base

Based on the above-mentioned formulation, a lexon base, \(\Omega\), is a set of expressions composed from a ordered pair \(<A, I>\) where

\[
A \text{ is the alphabet.}
\]

\[
I = \Gamma \times C \times R \times C \text{ is the set of lexons, expressing facts as relationship (in a given language)}
\]

\[
\Gamma, R, C \text{ are sets of strings over } A^+
\]

3.1.3 Model-theoretic approach

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], especially ontology scalability in size and complexity during both engineering and processing time.

3.1.4 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 and others are ‘left to …a mediation layer between ontology proper and the application’ [Meersman, R., 2000].

It is an interface layer between the model, the ontology proper, and applications. It is introduced into an otherwise monolithic representation of declarative and operational semantics. It is a separation between a model of fact types in lexons and lexon spaces and their constraints, rules and inferences on the mediation layer, interfaces between ontology and its application.

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 in the sense of databases. It is a subset of ontology base, defined above as $<A, F>$. $C$ means commitment to models, seen in task-oriented perspective of the ontology base. $U$ is use for system-specific configuration of commitments for a given application. It is one of the existing and possible architectural paradigms based on DOGMA ontology representation framework.

4.1 Architectural dimension and viewpoints

The ontology base is translated into models whereas commitments and uses are the two dimensions on the commitment layer of the ontology representation framework.

Across the three dimensions of model, commitment and use, there are four architectural viewpoints, called ‘perspectives’.  
- Ideational perspective  
- Structural perspective  
- Task perspective  
- System perspective

There are four corresponding context types: ideational, structural, task, application contexts.

Associated with the perspectives, are constructs, processes, design, such as lexons, models, commitments, contexts, signification, configuration, pipe-and-filter architecting patterns.

Based on these perspectives, methodology of ontology development and use can be developed to capture the best practices, guidelines, procedures and development management on the work along three dimensions.

Three types of users are under consideration.

- Model developers (M-developer)  
- Application developers who develop and architect commitments into uses in the light of tasks and systems in an application (A-developer)  
- Content developers who author contents according to a particular model in a particular view of it (C-developer)

Figure 1 shows its layout of mortar, bricks and marbles.

4.2 Model

This section examines lexons from the point of view of semiotics to define the system boundary. The structure of model can be defined recursively. A model constitutes one or more lexon spaces made up of lexons. It can include one or more models as its constituents. The structure of the model is that of recursive embedding. An ontology base is thus a semiotic system that consists of self-contained subsystems, models.

4.2.1 Lexon space

Lexon space is a set of lexons which form a (directly or indirectly) connected graph. The network of relationship between the lexons in the structure can be interpreted on different levels. The semantics of a lexon is grounded in the lexon space where it occurs. The lexon space pertains to ontology modeling rather to ontology use. It is partitioned during modeling. Ontology commitments, with respect to a task or task types, is scoping with virtual spaces over lexon spaces.

4.2.2 Lexon space as semiotic system

In a lexon space, each lexon enters into direct or indirect relationship with the rest symbolic entities in this finite semiotic space. Its definition derives from its relationship or its semiotic place in this self-contained system, not dependent any entity outside the system. The traffic lights are a simple semiotic system in which the three symbols are defined relative to each other. In this semiotic space, “… 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].
4.2.3 Structural and ideational aspects of lexons and lexon spaces

Lexons and lexon space are signs. Viewed in terms of structuralist semiotics, they consist of two elements: *signifiant* and *signified* [De Saussure, F., 1966]. This distinction has important suggestion on ontology modeling as well as architecting. Lexons and lexon space can be examined in two perspectives.

In a structural perspective, the lexon, $<\gamma, t_1, r, t_2>$, is a quadruple of four identifiers coming from three sets of symbolic primitives, *context*, $\Gamma$, *terms*, $T$, and *relationships*, $R$. Context on the structural level is *s-context*. It identifies a particular lexon space where the lexon is defined. In other words, it constitutes the structural context in which lexons are defined systemically. *Terms* and *relationships* are any arbitrary identifiers over a given alphabet.

The *signified* of our lexon spaces is the conceptualization [Guarino, N., 1998] of states of affairs in the domain or applications. In an ideational perspective, lexons embodies ‘concepts and relations’ via another semiotic system, namely, natural language terminology. $\Gamma$, $T$, and $R$ draw on natural language terminology and thus $\gamma$, $t_1$, $r$, and $t_2$ in $<\gamma, t_1, r, t_2>$ acquire natural language semantics. The context, $\gamma$, at this level, refers to a space in world semantics. We shall call it *i-context*. $t_1$, $r$, and $t_2$ besides being identifiers on the structural level, are assigned denotations of natural language semantics with reference entity types in the world.

---

5 The lexical entity type is a special case in that instead of drawing on natural language semantics it references to a close enumerable set of entities expressed by the same alphabet as ontology semiotic system itself.

The relationship of the structural and ideational aspects of lexons and lexon spaces suggests one of the two kinds of signification important in the current context. One is the assignment of world semantics to lexon spaces. In this case,

- The signification of the *signified* (world semantics) by the *signifiant* (lexons and lexon spaces) is arbitrary [De Saussure, F., 1966].
- The signification is an essential activity in ontology modeling.
- The signification is negotiation between the ‘conceptual grid’ and lexon spaces, where the structure of the latters must be agreed on.
- The signification, in fact, involves the semantic mappings among three semiotic systems, ‘conceptual grid’, system of lexical semantic relationship in natural language and ontology model.6 Though the methodology of translating conceptual and linguistic semantics into lexons is our focus of attention in architecting ontology development, the process itself is not part of our architecture proper.

Our development of ontology development platform mainly starts from the structural perspective of ontolog.

The other sense of signification is the transference or mappings between *signifiants*, formal semiotic systems, such as from ontology model into logical database schema. Such semantic transference takes place in software processes. The world semantics remains implicit and is expressed through operational behaviour of the artifact. This will be addressed in our research on ontology application development framework in the section about commitments to models.

4.3 Commitments

Commitments are mappings between lexons and lexon spaces in the model, on the one hand, and task-oriented and application specific semantic knowledge constructs, on the other. It implies selecting a view of model and process on the selected lexons, such as adding more constraints, transformations.

4.3.1 Task-oriented commitment

Commitments are the first level on the mediation layer. Their projectors are in fact Ω interpreters of lexons and lexon spaces. They act as interfaces between the heterogeneous model and various tasks of semantics-based processing. The commitment is designed and its translation developed according to tasks types or specific tasks.

4.3.2 Commitment transformation

The process of generating commitment is commitment transformation. It is a function that sends a set of semiotic objects to another set. It is dedicated to the second sense of signification of *signified* by *signifiant*, i.e., transference between two *signifiants*. In other words, it operates on lexons and lexon spaces on the structural rather than ideational sense.

From the point of view of the ontology model, the commitment transformation has two directions: interpretation and rendering. The input to the model interpretation is commitments of ontology model, one or virtual lexon spaces. The process transforms a set of the lexons into a set of different objects or constructs as required for particular type of tasks. The translation from ontological semiotics into any other arbitrary formal semiotics can be adoptive – total function with theoretically reversible transformation. It can be adaptive when the transformation is irreversible with possible non-monotonic, destructive translations.

Rendering is a process of transformation, verbalization, translation from any formal semiotics, and outputs a view of the ontology model. The input to the model rendering is *requests* and output a view of model, specific lexons and lexon spaces.

The transformation ranges from processes with configuration of parameter setting to those that rely on a set of rules requiring sophisticated means of specification.

---

6 The triangle of mappings among the three systems means ontology model draws on natural language terminology but it is not equivalent to lexical semantic relationships in natural language system.
For example, an interpretation of lexon model for database query or model traversal can be specified in RIDL [De Troyer O. et al, 1988]. It is worth noting that the rule base for interpretation and rendering can be itself a model of ontology, a meta-ontology for the interpreted or rendered models.

4.3.3 Commitment transformation and commitment library

A classification of commitments and their transformation can lead us to build up a software library that supports commitment transformation by basic task types and is extensible for specifics of particular tasks. There are four main features of commitment transformation that should be part of consideration in designing the transformation development framework:

- Commitments can be layered on each other
- Transformation process need to be configurable in predetermined formalism
- Transformation forms a hierarchy by types
- Transformation need to be extensible

They point us to
- a utility suite of parser and compilers for transformation configuration, ranging from parameter setting to rule specification,
- a class inheritance hierarchy of transformations,
- a framework on the basis of design patterns, such composite, visitor, decorator to allow for layering, configurability and extensibility.

On the basis of such software development framework, transformations can be developed by such guideline as

- No transformation depends on the other for its operation.
- No transformation shares a state with others.
- A transformation has no knowledge of pre- and post-processing transformations.
- The layers of transformation are not determinants on the final result of the layered transformations.

The commitment library is a storage of outcomes of the transformations. They can be model schema in different structure or format, design intermediate results, rules as part of knowledge bases, etc. They serve content reuse and share for such purposes as system interoperation on a lower abstraction level, content integration into multiple systems.

The transformation development framework and commitment library intended for application developers and knowledge architects. The transformation library will also be used by content authors who annotate their texts and terminology.

4.3.4 Commitment types

Comments can be classified in different ways according to its purposes. Here are some examples of commitment types along the functional dimensions:
- Data population commitment, where transformation introduces integrity constraints on lexons with respect to a data population. A model of lexons can be translated into an view of ER, ORM script [Demey, J. et al, 2002] or SQL scripts, or vice versa.
- Commitment for Aggregation, where a set of lexons are objectified, nested or packaged into another abstract schema constructs, such as frames.
- Commitment for Alignment, where two models of lexons are projected into one model with derived virtual lexons connecting the two models.
- Linguistic commitments, where the expression of models in natural languages is explicitly handled. It is a two-way transformation: natural language understanding onto a model of lexons and natural language generation from a model of lexons. It can also be transformation from a view of model into C++ or Java programs.
- Import/export commitment, for the purpose of ontology interoperability.

4.4 Use

We can follow C. S. Peirce and C. Morris beyond structuralist semiotics to introduce a second level on the mediation layer about ‘interprétant’, i.e. specification and processing with respect to the user. By ‘user’, we mean both software and human agents. See Spyns, P. et al for argument for pragmatics of ontology [Spyns, P. & Meersman R., 2002]. The first level of the mediation layer views the ontology model in a context of purposes or tasks. The second level considers the model and its configuration on the why-level in a context of users, benefitors or recipients.

The use dimension of the MCU paradigm of the DOGMA framework supports development based on the task requirements analysis for a given user. The development is largely an activity of
- Configuring the existing transformations: compiling rule-specification for a transformation to generate a required commitments
- Reusing and adapting commitment from commitment library for final or further transformation
- Pipelining transformations to create layered commitments. The principle of the architecture pattern ‘filter and pipes’ [Shaw, 1996] can be best applied here.

The purpose is to group existing or newly developed commitment into a software artifact interfacing between ontology and applications and enabling the customized use of generic semantic models. The existence of the developed transformation and commitment library puts semantics and its processes in a perspective of assembly. With the possibility of customized extension of transformation by the transformation development framework, the use dimension is cast in the light of component-based software engineering process and methodology.

4.5 Context

Context can be classified into two major categories along the separation of ontology and mediation layer: ontology context and application context.

4.5.1 Ontology context

Ontology context is an essential element for semantic interpretation of lexons and lexon spaces both ontology engineers and software processes.

The ideational context is the conceptualized spaces pertaining to particular lexons or lexon spaces. The structural context, where concepts, relations and lexons are defined, is $\gamma \subseteq \Gamma$ indicating a lexon space. A second sense of context lies in the embedding structure of the model, described above. Since the containing semiotic system is the context of the contained subsystem, we can derive the hierarchical structure of contexts from the recursive embedding structure of the model.
The system of context for a model is a quadruple \( <M, \Sigma, P, \mu> \), where \( M \) is a finite set of context variables, \( \Sigma \) is a finite set of constants denoting lexons and lexon spaces, \( P \) is a production that generate an element, \( e \in V \times (V \cup \Sigma) \), \( \mu \in M \) is the embracing context variable at the start of derivation.

4.5.2 Application Context

The context on the mediation layer is viewed in two perspectives: tasks and systems of applications. Commitments are specified and transformed with respect to task types. The task context, such as a typology of task types, pre-and post-conditions, are part of commitment requirements specification.

Uses are developed with assembly of commitments in the light of system interfaces. The context of consideration is the application context. This wider context of ontology use is the background to architect knowledge of the system with commitment configuration and interfaces between knowledge modules and between knowledge and systems.

4.6 Significance to ontology development

The DOGMA approach advocates the independence of semantic models from information systems with a layer that mediates between ontology and applications. This encourages decoupling general semantics from specific application logic as development methodology, thus enabling the reuse and share of semantic models and processes across systems, applications and developments. With given commitments to ontology, models or part of models are ‘highlighted’ as foci of semantic universe of discourse, commitments and uses will be developed or reused for various needs of application tasks and systems.

This approach provides means to make semantic knowledge from diverse application sharable. The model of lexons captures sharable semantics which is viewed and used for diverse tasks, applications and systems. This achieves a form of semantic independence for IT applications that may be compared to the earlier notion of data independence for such applications using databases.

The further modularization along MCU dimensions into lexon spaces, commitments, and uses, with corresponding processes, in contexts, enhances the flexibility of modeling, processing and reuse. It enables the developer to see semantic space in strata of objects and processes in a framework of configuration for applications. It is important facility to ontology scalability in terms of model size and complexity and along the dimension of various task and system needs.

5. ONTOLOGY DEVELOPMENT PORTAL

This section discusses an ontology portal design on the MCU paradigm. The portal is not only a portal to support distributed ontology modeling in a community of ontology engineers, but also provides facilities for developing applications of ontology with shared model transformation, commitments, uses, software framework. Since the development portal touches the area of ontology deployment for ontology-based content developments and mediation, the discussion also includes one of our future researches on the infrastructure of ontology-based services.

The architecture constitutes two separated but related parts, ontology development and deployment. 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.
5.1 Ontology development

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 layer 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 layer 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 layer is also responsible for client and communication management.

5.2 Ontology-based service

Semantic processing also needs data and rule population, i.e. instances of ontology schemas and types. Adjacent to the ontology development portal, there is an ontology service portal where deployed are ontology artifacts such as lexon model, transformation, commitments and use. It provides instance-related ontology operations via ontological level: such as ontology-based agent mediation, content annotation, semantic operation and validation of contents with ontology.

Lexon models, commitments and uses are deployed for two main purposes.

- Ontological content creation: data annotation
- Ontological content deployment: instance-level processing for mediation or querying
The second layer of the service portal manages

- Collection of instance data
- Semantics-based query and browsing of instance population
- Validation of instance data against ontology
- Synthesis of instance data for heterogeneous tasks with component
- Mediation in inter-application communication
- Mediation-based access to heterogeneous data sources in different data formats and in real time.

The following figure shows an example implementation. The components, Content Synthesis Content Validation, Querying Service and Browsing Service are implementation of the middle layer of the service portal. The mediation service is yet to be implemented.

![Figure 5. An example of Ontology Development Portal (lower triangle) with Ontology Deployment Portal (higher triangle)](image)

### 5.2.1 Collaborative ontology development

The ontology developer uses Ontology Development Studio connected to the ontology development portal, to develop check in and check out models, commitments and uses, i.e. MCU data. Ontology developers may be grouped with dedicated operational scope or type over MCU data. It acts as pivot point for collaboration among m-developers, a-developers, c-developers engineers together to form a collaborative engineering environment. It also imposes centralized versioning of MCU data and helps synchronize ontology update to all participants.

For content developers, one of the important issues in practical semantic content engineering is reference resolution out of the developer’s working scope. Ontology service portal can support reference resolution on the instance level in the contents via ontology. It helps the c-developer identify the external ‘address’ among distributed heterogeneous data sources with his/her knowing details irrelevant to the task.
6. USE CASE

The ontology representation framework and MCU architecture are developed based on the insights and lessons learned from, among others, NAMIC [Basili R. et al, 2002] [De Bo J. et al, 2002] and OntoWeb. In NAMIC, an ontology server in the DOGMA framework provides the platform for modelling concepts of news subjects. The ontology is used for defining user profiles in a news web portal. In OntoWeb, a semantic portal with ontology-based querying and browsing facilities with DOGMA ontology server, where we identified collaboration in ontology and content development and scalability of ontology issues of big practical importance.

We are continuing our effort in SCOP and FF POIROT. In SCOP, DOGMA framework is used to provide search and retrieval from a set of independent databases, mediated on the semantic level with ontology. In FF POIROT, ontologies will be mined from the texts and modeled of financial, legal and forensic domains in a distributed, collaborative environment. DOGMA framework will used for representation and MCU infrastructure and tools, such as ontology alignment, will be evolved. The ontologies will be used to build multilingual terminology base to support linguistic interoperability. Applications of the ontologies will be developed to fit into the existing information and organizational systems to help detect and prevent financial frauds. Based on the framework and architecture, ontology engineering methodology will be evolved from practical work.

7. CONCLUSION

Based on the previous and current European R&T projects, 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], 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 at different semantic levels, scalability in size and complexity, development methodology with specific conceptual framework and architecture.

We shall devote attention to classification of lexons, taxonomies of commitments, knowledge design and architecting patterns with uses, agent communication and mediation distributed and open semantic engineering environment in DOGMA framework and MCU architecture.

ACKNOWLEDGEMENT

We wish to thank our colleagues in STARLab, VUB and partners in our EU research projects for their direct and indirect contribution to the current idea of the architectural paradigm.

REFERENCES


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