EOS: Making the Epistemic Impact of Ontologies in Knowledge Processing Explicit

Wolfgang Wohner[1]

[1]Bavarian Research Center for Knowledge Based Systems
Orleansstraße 34, D-81667 Munich, Germany
wohner@forwiss.de

Abstract. EOS provides a formal framework for automated knowledge processing. The EOS framework combines theoretical foundations derived from epistemology and ontology with recent advances in the fields of knowledge representation and processing. As such it allows for an explicit modeling of domain knowledge on the ontological as well as on the epistemic layer, which ensures a flexible, modular system architecture that overcomes the limitations of current systems. We will introduce the overall EOS architecture and discuss its practical applicability by presenting an actual EOS system for intelligent knowledge retrieval in semi-structured data.

1 Introduction

Knowledge processing (KP) has become a major field of interdisciplinary research. Knowledge is being identified, acquired and analyzed, then formalized using various representation models. Resulting representations lay the groundwork for processes that incorporate knowledge, be it on a general level (e.g. for supporting workflows within organizations) or in practical applications (e.g. for assisting specific workflow components). In this paper we will concentrate on the latter use of knowledge, and particularly on machine supported applications, i.e. knowledge *processing* in its strict sense.

At present the predominant means for representing knowledge in KP systems are *formal ontologies*. But despite their widespread use the notion of formal ontologies remains fuzzy. Gruber's commonly accepted definition of an ontology as a '*specification of a conceptualization*' [8] does not account for the semantic implications of ontologies although the object of such conceptualizations is supposed to be knowledge. Consequently, the term 'ontology' is attributed to a large variety of formalizations that differ greatly in their expressive power. What is lacking is an explicit specification of the semantics an ontology is providing and how it may be utilized, i.e. a 'meta-specification' describing the epistemic impact of ontologies. This way KP systems could make immediate use of the semantics inherent in knowledge representations. The benefits of such meta-information on ontologies are:

- On the application side reasoning processes within KP systems do not have to be tailored to particular tasks or domains as ontology-specific inference rules are provided by the epistemic meta-information. Recognizing epistemics in KP therefore allows for flexible, self-adapting systems. Additionally, exchanging ontologies among different systems is being greatly facilitated by providing their inherent application semantics.

- On the conceptual level epistemic meta-information enriches the expressiveness of knowledge representation languages. The explicit specification of epistemic semantics leads to a clear, two-layered knowledge representation design that respects the distinction between ontological and epistemic facts.

In order to give a deeper conceptual motivation for the EOS framework the following paragraphs of this section will briefly discuss two different perspectives on knowledge, namely organizational knowledge management and the philosophical theory of knowledge, and contrast these approaches with the particular notion of knowledge in machine-supported knowledge processing. Based on these preliminary considerations Section 2 introduces a general framework for knowledge processing systems that serves as a basis for the EOS framework. An actual knowledge processing system complying with the EOS framework is being presented in Section 3 in order to stress its practical applicability. Section 4 comprises related work that is being discussed by referring to the conceptual and practical implications of the EOS framework. Finally, concluding remarks and future research directions can be found in section 5.

1.1 Using Knowledge: The Knowledge Management Business Model

Knowledge management (KM) has become a vital economic factor for organizations as commercial success depends on a proper understanding of internal processes leading to increased productivity and innovation, external processes, e.g. concerning market perspectives, and interactive processes among organizations and their customers as in e-commerce environments. All of these processes require knowledge in order to be mastered successfully. Knowledge, here, is seen in the context of the organizational memory (OM) that comprises the intellectual potential of employees (i.e. skills, experience, expertise), document archives (electronic as well as print media) and all further information relevant to the organization (e.g. inherent in workflow processes) [11]. Therefore, KM tasks concentrate on capturing and organizing OM semantics. Predominant problems in this area are how to implement ways to acquire knowledge, particularly implicit knowledge, and how to incorporate it into the organization's workflows.

The KM business model understands knowledge as a valuable resource that should be exploited in order to supplement the success of an organization. Thus KM offers no explicit theory of knowledge nor does it promote a particular methodology for representing knowledge. Rather, it provides guidelines for identifying and using relevant information (about and within business processes) and its actual and potential benefits for an organization.

1.2 Defining Knowledge: The Philosophical Approach

Theories of knowledge have a long tradition in philosophy, in fact, an entire philosophical discipline, *epistemology*, is dedicated solely to the study of knowledge. Epistemology focuses on questions about the nature of (human) knowledge, i.e. what is knowledge and what can be known. Modern analytical philosophy stresses the propositional structure of knowledge and uses mathematical logic for arguing about propositions. The general idea is that true propositions describe situations in the world, which presupposes objective truth that may be attributed to propositions. In order to turn a true proposition into knowledge its truth has to be proven, or justified. This leads to the most prominent definition of knowledge as ‘justified true belief’ that has been given by the Greek philosopher Plato and is still at the center of current debates.

While epistemology examines the *nature of knowledge* itself, the philosophical discipline of *ontology* is concerned with the *nature of objects* of knowledge. This way ontology serves as a basis for epistemic theories as it gives a notion about being and truth which are fundamental to knowledge.

1.3 Representing Knowledge: The Knowledge Processing Approach

During the past two decades the notion of knowledge in the AI community experienced a notable shift from a primarily functional view that was focussed on modeling human rationality towards a new perspective that put emphasis on modeling systems in the world [1], [9]. We call the former understanding of knowledge the *narrow view* as it is foremost task-driven, i.e. solely knowledge relevant to a specific, pre-defined problem is taken into consideration. Opposed to this notion is the *general view* of the latter approach where knowledge is expected to describe not only details for particular tasks but an entire problem *domain*. Thus the general view is closely related to the objective reality of the problem domain and in itself independent of possible applications. It therefore gives an ontological (in the philosophical sense of the word) perspective on an application area.

Knowledge representations are generally applied to tasks where computer systems need additional input (domain knowledge) in order to adequately process data, e.g. texts in natural language. The system is regarded as possessing knowledge about a problem domain via its formal representation that it accepts as an input. Knowledge in this sense consists of data while *application* logics, i.e. knowledge on how this data may be used, is being considered at most on a restricted level, e.g. languages based on description logics, like DAML+OIL [15], inherently provide a basis for reasoning on the concepts defined but cannot explain any further use of this inferred knowledge within the system, e.g. for query processing. Thus knowledge representations actually address (philosophically) ontological aspects while the definition of epistemic processes that use elements of these representations are to a great extent part of the algorithmic implementation of the system.

2 Processing Knowledge

Knowledge processing (KP) in computer systems comprises all tasks and methods concerned with modeling, representing and employing domain knowledge for enabling a desired system performance, e.g. intelligent information management. In this section we will present a commonly agreed upon design for KP systems and point out its shortcomings. We will use this discussion for motivating a novel, epistemic perspective on KP and present how it may be implemented by the EOS framework.

Example System

In order to motivate these considerations we will first briefly sketch an exemplary KP system that may be used for intelligent information retrieval from heterogeneous information sources such as the Web, interpreted as a vast knowledge base. Representative systems falling into this category are [3] and [12]. The internal KR, typically a formal ontology, models some domain of interest (e.g. the Enterprise Ontology, a collection of terms and definitions relevant to business enterprises [13]) and is used to extract domain specific information from external sources (e.g. Web pages). This information is being stored in the system's own knowledge base and serves as a repository for answering ad-hoc user queries.

2.1 The General Picture

Computer systems that use an explicit modeling of domain knowledge generally exhibit a basic layout as depicted in figure 1. From a functional point of view a KP system will accept input data and process it according to the internal knowledge representation (KR). As indicated the system returns structured information that may on its own part, again, serve as input to the system.

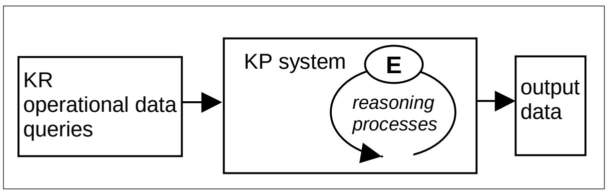

Figure 1. A General Framework for Knowledge Processing Systems

The different system components are:

- **Input data**. Naturally, there are different kinds of input data serving different purposes:
  – ***Knowledge representation (KR)***: essential to the overall performance is a formalized KR that models the system's application domain. The KR is held persistent within the system and serves as a basis for advanced tasks such as reasoning processes.
  – ***Operational data***: any type of documents (e.g. structured or unstructured text files, graphics, etc.) containing information that corresponds to the internal KR is regarded as operational data. For performance reasons it may be stored and indexed separately, e.g. inside a knowledge base attached to the system.
  – ***Queries***: user interaction with the system as well as automated processes may trigger queries against the KP system. Queries may address the system's KR itself or be directed towards the structural or semantic content of operational data, resulting in newly created conceptual knowledge that becomes part of the KR, or operational data.

- **Reasoning processes**. The reasoning capacities of the system utilize the internal KR and produce structured information from the previously received input data. Inference semantics and all associated mechanisms for processing an KR instance, i.e. the epistemic layer (E), are integral parts of the system.

- **Output data**. The system returns structured information computed from operational data and/or the internal KR, e.g. specific information held within documents, classifications and indices, or return values of queries.

As depicted by this general layout, the formalization of application semantics (E) is hard-coded into the algorithmical implementation of resulting systems. This leads to several drawbacks:

- KP systems following this general layout must be designed for very specific tasks and domains as the corresponding semantics may differ to a great extent, e.g. natural language processing and deduction on chemical data require very different application semantics.
- Additionally, these systems are highly inflexible regarding conceptual changes of the knowledge representation, i.e. only a restricted class of KRs can be processed by a particular system. Increasing the expressive power of a KR (e.g. by introducing facilities for incorporating axiomatic terms to a given KR model) must therefore result in a costly system redesign.
- For similar reasons exchanging KRs among KP systems poses serious problems. Again, foreign KRs must comply with native application semantics and modeling paradigms of a given KP system in order to render it capable of processing it.

– Finally, even KRs exhibiting the syntactical makeup a KP system can process may be interpreted incorrectly as there is no direct coupling between the objects of the KR and their semantic impact. For example the notion of some relation ‘part-of’ can be different for two KRs, but a KP system will always process ‘part-of’ according to its own implementation.

2.2 The EOS Framework

Based on the preceding considerations we will now introduce the EOS framework for KP systems shown in figure 2. The EOS framework refines and extends the system layout of figure 1 by introducing different semantic layers to the internal knowledge representation.

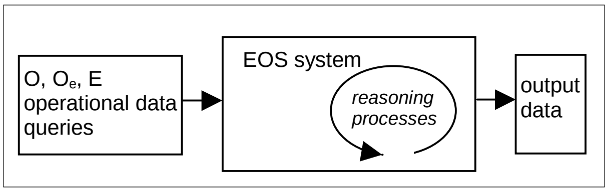

Figure 2. The EOS Framework

We differentiate between two levels of the ontological domain model and an superordinate epistemic layer:

- ***Ontological objects of knowledge (O)*****:** a fundamental set of entities the system can identify. This is the ontological basis of the computer system. The general term we use for depicting such ontological entities is that of a *concept*. The basic assumption is that any abstract or concrete real-world entity is being represented by a concept of its own and that no two concepts refer to the same entity (unlike natural language terms that may be used synonymously). This way, concepts allow for a semantically disambiguous modeling of natural objects, their attributes and qualities, as well as relations among objects (e.g. O(business) may contain objects like EMPLOYEE, PROJECT and WORKS-IN, etc.) within some domain of interest.

- ***Onto-epistemic objects of knowledge (O_e)*****:** a set of rules or axioms referring to particular concepts of **O**. An example rule of **O(**business**)** could be paraphrased as ‘each employee works in at least one project’. Onto-epistemic objects complete the ontological domain model by providing domain-specific details to the simple objects of **O** .

- ***Epistemic objects of knowledge (E)*:** the set of explications about the objects of **O** and $\mathbf{O_e}$. This is the epistemic layer of the system. Unlike with the general framework **E** is here treated as additional input to the system. This establishes an important shift from leaving application semantics hidden within the system to explicitly modeling these semantics into the KR. Explications mould a body of laws that specify how ontological and onto-epistemic objects should be processed by the system. The notion and function of laws will be further elaborated in the following paragraphs.

Formal ontologies in current systems usually cover objects of **O** and to some extent of $\mathbf{O_e}$ (e.g. [4]). To the best of our knowledge there is no system or methodology using epistemic objects as we understand them, i.e. meta-level descriptions about application semantics of ontology objects. Thus, such a formalization defines metadata about the ontology, foremost semantic processing rules we call *laws* [14]. Again, laws have to be understood and executed by software components but the invaluable benefit they could provide is a homogeneous formal description of the semantic and syntactic implications of such processes. Laws may be regarded as function templates that accept *cases* (e.g. a query) and contain formalized descriptions how to solve them.

Generally, laws provide application semantics about the objects of an KR. As such they render the epistemic impact of an KR explicit while remaining part of the KR (as its metadata). This way the shortcomings of current KP systems complying with the general KP framework can be overcome. The EOS framework allows for flexible, self-adaptable KP systems as knowledge about the semantics of reasoning processes (**E**) is modeled outside these systems (in the form of laws). For example conceptual changes can be expressed by laws which are incorporated into the KR they are describing. Consequently, only the epistemic layer of a KR (i.e. the input data fed to the system) has to be adapted for the system to function correctly. Naturally, processing KRs and passing them over to other EOS systems poses no problems because the application semantics of KR objects is being supplied along with the KR.

Areas of application for laws are:

Ontological semantics

The expressive power of an KR (e.g. if it is possible to define axiomatic terms) is made explicit by laws. Thus they state the representational limits of a KR such as the scope of the ontology or its level of granularity. Therefore an EOS system is aware of representational capacities of a KR it is processing.

Inference semantics

Inference rules may differ greatly between concepts or groups of concepts (e.g. relations ‘is-a’ and ‘part-of’ are both transitive but may be treated differently during query processing). As laws can be general or attributed to single concepts or classes of concepts, they can be used to express inference semantics.

Another aspect of inference semantics concerns *fuzzy concepts* (e.g. closeness) that have to be interpreted according to their context. The meaning of e.g. the term 'close' depends on the context of a query or reasoning process, as there are different notions of closeness in the context of houses and, say, atoms. In such cases techniques are needed to establish context which requires laws that describe how the desired information can be deduced.

Query semantics

Automated *semantic query rewriting* is a promising technique for improving query return values. Using ontology knowledge an original query may be transformed into a set of refined queries. The excerpt of an XML document shown below does not contain an <Address> tag, so a query restricted to searching addresses would omit this document:

```
<Person>
<Name> Smith </Name>
<Phone> (222) 333-4444 </Phone>
<Profession> philosopher </Profession>
</Person>
```

By contrast, laws provide rules for extending the scope of the query from addresses to e.g. phone numbers, street names and other address components known to the ontology. This would yield Mr. Smith's phone number, valuable information that the original query could not have produced.

Uncertainty may also play an important role in the context of iterative document querying, i.e. reasoning on grounds of intermediate results extracted from documents. From the XML example shown above it can be inferred that 'philosopher' is an instance of the concept PROFESSION. The value 'philosopher' can now be interpreted as a concept as well. But as this information has been derived from the textual content of a document it must be regarded as uncertain knowledge. Uncertain knowledge is an omnipresent factor in intelligent information management and we will intensify our research efforts in that direction.

3 The EOS Knowledge Processing System

Based on the considerations of the preceding sections we will now turn to introducing the architecture of the EOS knowledge processing system, a practical application of the EOS framework. Our EOS system will be used for providing access to heterogeneous semi-structured data sources. Its three main components are (i) an Information Manager, (ii) an Ontology Manager and a (iii) Query Manager.

Basic assumptions about the system are:

- As a preliminary assumption the EOS system possesses a unified interface, i.e. all input and output data is coded using a semi-structured format. In particular, data generated by the system can, again, serve as direct input to the system.
- The system requires a KR that comprises formalized knowledge about the application domain. The ontology fed to the system is expected to contain ontological and onto-epistemic objects of knowledge as well as epistemic laws.
- There is a set of heterogeneous semi-structured documents (e.g. XML documents) covering topics of that domain. These documents serve as operational data of the EOS system.
- There exists a mapping between markup tags of the documents and the concepts of the ontology, i.e. the ontology can 'understand' markup semantics in a sense that the concepts involved are part of its formal model.

We decided to concentrate on semi-structured data for several reasons. Besides the most promising perspective that XML-based representation formats will become the predominant means for electronic information exchange and the widespread tool-support for managing and querying XML documents, semi-structured data offers a variety of advantages over unstructured data (e.g. plain text files). The main benefits stem from the distinction between content and metadata which allows for more sophisticated reasoning procedures.

Information Manager

The Information Manager accepts input data (KR and operational data) which is being stored and indexed inside the system's own internal repository. The stored data must be ready for efficient access, e.g. for reasoning procedures. Operational data will naturally be considerably large, so indexing, along with an efficient linkage between operational data elements and ontology concepts is an essential requirement.

Query Manager

The Query Manager accepts user queries and converts them into queries against the internal repository. Return values can be document fractions, or complete documents, as well as purely ontological data. In order to retrieve valid results from operational data the Query Manager first has to understand the semantics of the query and then make use of the ontology's domain knowledge for exploring the particular structures of the documents.

Ontology Manager

The domain ontology is being accessed by the Ontology Manager. Its task is to evaluate the epistemic content of ontology laws in order to assist the other system components. Laws are used to control the analysis and processing of objects of $\mathbf{O}$ and $\mathbf{O_e}$. This includes simple ontological reasoning as well as managing more advanced epistemological processes such as semantic query rewriting.

The general task of the EOS system is to derive information (semantics) from semi-structured data (data conforming to syntax). There are some properties of semi-structured data the EOS system may particularly take advantage of. We will illustrate this by referring to XML syntax:

- ***Syntax definition***: the syntax definition of markup elements used within an XML document is known via its DTD, so the system is aware of all element names, their attributes and subelements.
- ***Concepts***: the semantics of the structuring elements (tags) are known to the system because of the mapping between elements and ontology concepts.
- ***Context***: markup elements are organized hierarchically thus establishing contexts (e.g. by nesting tags like <Name> and <Address> into <Person>) which can be interpreted semantically.
- ***Types***: in a weak sense each markup element represents a type of its own but it is also possible to introduce primitive or derived element datatypes using e.g. XML Schema.

In summary, semi-structured data offers the possibility to establish a direct linking between the system's knowledge representation and the operational data it has to process. The knowledge representation itself is structured as follows:

- **O**: ontological objects are being defined in a *dictionary of concepts*. On the one hand the dictionary serves as a complete listing of all domain entities the system knows of, along with their definitions in a human readable form. On the other hand it contains a mapping between the concepts and the vocabulary of the problem domain: while ontological concepts are unique and disjunctively refer to single real-world entities the vocabulary used by a community may be ambiguous, i.e. different groups within the same community might also differ in the language they use. Thus, the dictionary provides a mapping for synonymous vocabulary terms depicting the same ontological concept.
  A *domain model* formalizes the structure of the problem domain, i.e. its entities and their interrelationships. The domain model is made up of the concepts defined in the dictionary.
- $\mathbf{O_e}$: axioms about the concepts of O are specified in a *body of rules*. Following [2] this can be efficiently accomplished using Frame-Logic [10] but DAML+OIL [15] may proof itself feasible as well.
- **E**: application semantics are modeled using *laws*. It is still an open question how laws should be formalized for practical purposes but Frame-Logic and various description logic languages offer a promising starting point for further research.

In conclusion, EOS uses formal ontologies coupled with their inherent application semantics in order to implement intelligent information retrieval and management of semi-structured data. EOS makes use of the notion of laws for operating on ontological information, e.g. for supporting automated reasoning processes and intelligent query processing.

4 Related Work

The discussion of related work will focus on conceptual aspects concerning the formalization of epistemic processes and actual KP systems dealing with semi-structured data.

Modeling Epistemic Processes

A recent approach to bridge the gap between formal ontologies and reasoning processes has been that of Problem-Solving Methods (PSMs). PSMs describe the reasoning process of a knowledge based system in an implementation- and domain-independent way [7]. They are abstract, task-oriented methods designed for facilitating knowledge-engineering processes and address as such similar problems as laws. But, unlike PSMs, laws are an integral part of the formalization of domain knowledge and may be distributed and applied as such. Laws only exist in the context of ontologies and give instructions on how to process their formal objects correctly, i.e. laws are metadata about ontologies while PSMs are about specific (types of) tasks. For knowledge processing purposes we promote the use of laws as they integrate naturally into the formal body of ontologies and KP systems. Consequently, the immediate application of laws poses no difficulty whereas finding an appropriate PSM for a domain-specific task (i.e. choosing a PSM that suits the problem and operates on the correct level of generality) is not trivial. However, PSMs can be useful for designing laws tailored to a particular domain model.

Knowledge Processing on Semi-structured Data

There are two general approaches to combine ontologies and markup languages: (i) defining new markup which is directly related to the ontology or (ii) translating foreign markup into native concepts of the local ontology. The first approach has been propagated by SHOE [12] and Ontobroker [3] but its drawback is obvious. Since their markup methods did not evolve to become widely accepted standards, only a small portion of Web documents use them. For this reason current research besides EOS, e.g. [4], [5], is focused on making the second approach work. Systems such as On2broker [4] use ontologies as the overall structuring principle that drives query processing and inference mechanisms. However, these dynamic processes are mostly hidden within algorithms of software components like query and inference engines. Despite the modular architecture of On2broker (e.g. decoupling of inference and query engines) a notion similar to that of laws is lacking.

5 Conclusions and Future Work

We have motivated and discussed the importance of introducing epistemic metadata to knowledge representations, such as ontologies, used by KP systems. For explanatory purposes we presented a general framework for KP systems and extended this framework to meet this requirement. The resulting EOS framework exhibits a clear conceptual distinction between ontological, onto-epistemic and epistemic representational objects. We characterized epistemic objects as laws that model application semantics for objects of knowledge on the subordinate ontological and onto-epistemic levels. Finally, we described the architecture of a practical EOS system for intelligent information retrieval. Its main characteristics are, besides the employment of laws, the special notion of concepts as unique representatives of real-world entities and the proposed mapping between these ontological concepts and markup tags of the problem domain.

Our future research will concentrate on a formal specification of laws, and their implementation into our EOS system. This way we hope to establish a sound methodology for the application of laws which will enable us to give out practical guidelines for incorporating them into KP systems. Another interesting aspect of the EOS system is that of semantic query rewriting. We will intensify our efforts in that field and evaluate the capabilities and shortcomings of XML query languages (e.g. XQuery) in that respect.

6 References

1. W. J. Clancey. The Knowledge Level Reinterpreted: Modelling Socio-Technical Systems. International Journal of Intelligent Systems, 8: 33-49, 1993.
2. S. Decker, D. Fensel, M. Erdmann, R. Studer. The Technical Core of Ontobroker. Draft, http://citeseer.nj.nec.com/149833.html.
3. D. Fensel, S. Decker, M. Erdmann, R. Studer. Ontobroker: The Very High Idea. In Proceedings of the 11 th International Flairs Conference (FLAIRS-98), Sanibel Island, Florida, USA, pp. 131 -135, May 1998.
4. D. Fensel, J. Angele, S. Decker, M. Erdmann, H.-P. Schnurr, S. Staab, R. Studer, A. Witt. On2broker: Semantic Access to Information Sources at the WWW. In Proceedings of IJCAI-99 Workshop on Intelligent Information Integration, Stockholm, 31 July 1999.
5. D. Fensel et al. OIL in a nutshell In: Knowledge Acquisition, Modeling, and Management, Proceedings of the European Knowledge Acquisition Conference (EKAW-2000), R. Dieng et al. (eds.), Lecture Notes in Artificial Intelligence, LNAI, Springer-Verlag, October 2000.
6. A. Farquhar, R. Fikes, & J. Rice. The Ontolingua Server: A Tool for Collaborative Ontology Construction. Knowledge Systems Laboratory, September, 1996.
7. A. Gomez-Perez, V.R. Benjamins. Overview of Knowledge Sharing and Reuse Components: Ontologies and Problem-Solving Methods. In Proceedings of the IJCAI-99 workshop on Ontologies and Problem-Solving Methods (KRR5), Stockholm, Sweden, August 2, 1999.
8. T. R. Gruber. A translation approach to portable ontologies. Knowledge Acquisition, 5(2):199-220, 1993.
9. N. Guarino, Formal Ontology, Conceptual Analysis and Knowledge

Representation. International Journal of Human and Computer Studies, special issue on The Role of Formal Ontology in the Information Technology edited by N. Guarino and R. Poli, vol 43 no. 5/6, 1995.
10. M. Kifer, G. Lausen, J. Wu: Logical Foundations of Object-Oriented and Frame-Based Languages, Journal of the ACM, 42, 1995.
11. R. Klemke. Context Framework – an Open Approach to Enhance Organizational Memory Systems with Context Modelling Techniques. Proceedings of the Third Int. Conf. On Practical Aspects of Knowledge Management (PAKM2000), Basel, Switzerland, 30-31 Oct. 2000.
12. Luke, S., Heflin J. SHOE 1.01. Proposed Specification. SHOE Project. February 2000. http://www.cs.umd.edu/projects/plus/SHOE/spec1.01.htm.
13. Mike Uschold, Martin King, Stuart Moralee and Yannis Zorgios (1998) The Enterprise Ontology The Knowledge Engineering Review , Vol. 13, Special Issue on Putting Ontologies to Use (eds. Mike Uschold and Austin Tate).
14. W. Wohner. A Modest Proposal: Reasoning Beyond the Limits of Ontologies. In Proceedings of IJCAI-01 Workshop on Ontologies and Information Sharing, Seattle, Washington, August 4-5, 2001.
15. Reference description of the DAML+OIL (March 2001) ontology markup language, http://www.daml.org/2001/03/reference.html.