A Framework for an Ontology-Based Information System for Competence Management

Ştefan TRĂUŞAN-MATU
Research Institute for Artificial Intelligence, Romanian Academy, Bucharest, România
Department of Computer Science and Engineering, "Politehnica" University, Bucharest
Cristina NICULESCU
Research Institute for Artificial Intelligence, Romanian Academy, Bucharest, România
trausan@racai.ro, ncristina@racai.ro

The paper presents a framework for an intelligent information system for competence management based on ontologies in information technology organizations. There are described some theoretical concepts about ontologies, production rules and intelligent information systems. The information system will provide intelligent personalized access to concepts from an IT ontology and to a collection of relevant associated documents indexed according to the ontology's concepts. The declarative knowledge of the ontology and an associated set of production rules may be used for automatic inferences that will enable reasoning for getting intelligent answers to users' queries, under an expert system dialog. The paper uses examples from a first version of the ontology for Software Engineering concepts.
Keywords: *Competencies, ontology, management, information technology.*

1 Introduction

A competencies management system can be seen as a part of a Human Resource Management system, which provides the ability to store dispersed and unstructured corporate knowledge, such as corporate competencies characteristics.

In general, a competence management system (CMS) has to achieve three functions:

- to support the complete and systematic acquisition of knowledge about the competence of the members of an enterprise;
- to provide the knowledge about competence and the competence owners;
- to apply the available knowledge to serve a purpose.

A solution to the above problems is to develop an intelligent information system, based on a domain ontology, for example, Software Engineering in our case. The knowledge in this base should be both declarative and procedural. Declarative knowledge refers to what is known to exist in the domain, what are the concepts and how they are related. This kind of knowledge is usually constructed around a taxonomy and it may be viewed as an ontology of the domain.

In addition to domain knowledge, general knowledge may be also used, e.g. as a top level ontology, containing, for example general concepts describing human activities and regulations. Moreover, other kinds of knowledge is necessary. Procedural knowledge contains rules, recipes about how to use knowledge. For example, a general rule could say that, if you want to do an activity related to an object in a community, you should see what laws or regulations you should respect.

The paper continues with a section describing some theoretical concepts about ontologies and production rules. The following section introduces some original ideas about intelligent information systems. Section 4 presents the ontology for Software Engineering concepts. The paper ends with some conclusions and further development ideas.

2. Ontologies and rules

An ontology is, in the context of intelligent, knowledge-based systems, a declarative knowledge base containing the concepts and the relations that exist in a given domain, it is "a *specification of a conceptualization*. That is, an ontology is a description (like a formal specification of a program) of the concepts and relationships that can exist for an agent or a community of agents. This definition is

consistent with the usage of ontology as set-of-concept-definitions, but more general" [3]. The name is obviously inspired from philosophy, where it means a "branch of metaphysics concerned specifically with what (kinds of) things there are" [6].

From a knowledge representation perspective, ontologies are semantic networks that state what kinds of concepts exist and what abstraction-particularization relations hold among them. If a concept is a particularization of another concept, it has all the features of the more abstract concept and, maybe, some particular ones, For example, in figure 1 (the Protégé environment [7] was used for the development of the ontology), the fact that the concept "BridgesAndElevatedHighways" has "Crossovers", "Foot Bridge", "MobileBridge", "Overpasses", "RailBridge" and "RoadBridge", implicitly enumerates the only possible cases. Moreover, all these inherit properties (e.g. regulations) that belong to "BridgesAndElevatedHighways" or its ancestors.

The first version of an ontology for *Software Engineering* concepts (the domain ontology) was built using the *Guide to the Software Engineering Body of Knowledge* [9]. The purpose of the *Guide to the Software Engineering Body of Knowledge* is to provide a consensually validated characterization of the bounds of the software engineering discipline and to provide a topical access to the Body of Knowledge supporting that discipline. The Body of Knowledge is subdivided into ten software engineering Knowledge Areas (KA) plus an additional chapter providing an overview of the KAs of strongly related disciplines. The descriptions of the KAs are designed to discriminate among the various important concepts, permitting readers to find their way quickly to subjects of interest.

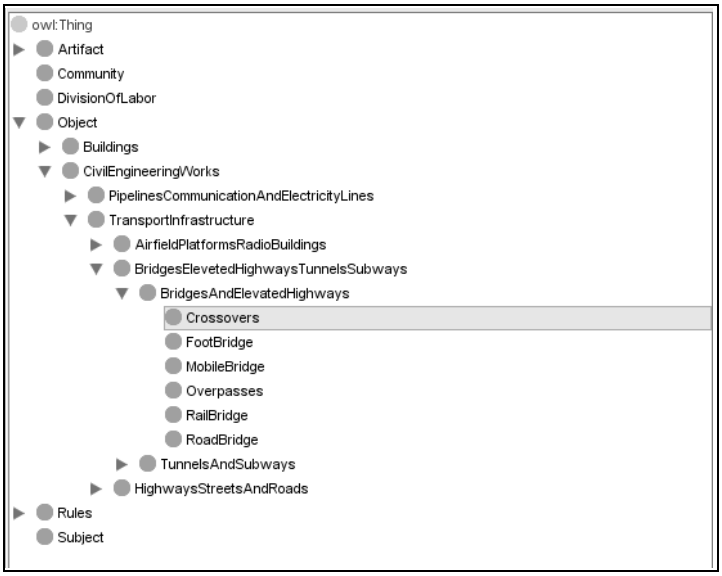

Fig.1. A fragment of an ontology

The ontology is developed following an integrated cognitive and socio-cultural approach. It contains a taxonomy of objects structured according to a project of the IEEE Computer Society Professional Practices Committee. This structured development facilitates further knowledge acquisition.

However, ontologies do not cover all kinds of knowledge representation. In addition to declarative knowledge representation, there is a need also for procedural knowledge, saying what to do in a given context. Such type of knowledge may be represented by production rules, which are pairs *condition – action*: IF *condition* holds, THEN PERFORM *action*. Conditions usually contain patterns and variables that may be linked to facts. A production rule system has a conflict resolution strategy that selects the rule that will be applied from the rules that may be applied.

The Jess production rule system [4] is one of the most known systems. In addition it may be plugged-in Protégé. A program in Jess is a collection of rules that can be matched to the

existing data in the working memory. Each rule has a first, matching (IF) part, and a second, action one (THEN), which modifies the working memory or prints something. A rule may have variables that are linked to values in the working memory using pattern matching. For example, a rule that prints the information that local authorities may provide is below exemplified. In this rule, the variables $?p, $?r, and $?c are matched to all the available data, in the working memory, regarding what the local authority provides, releases and controls.

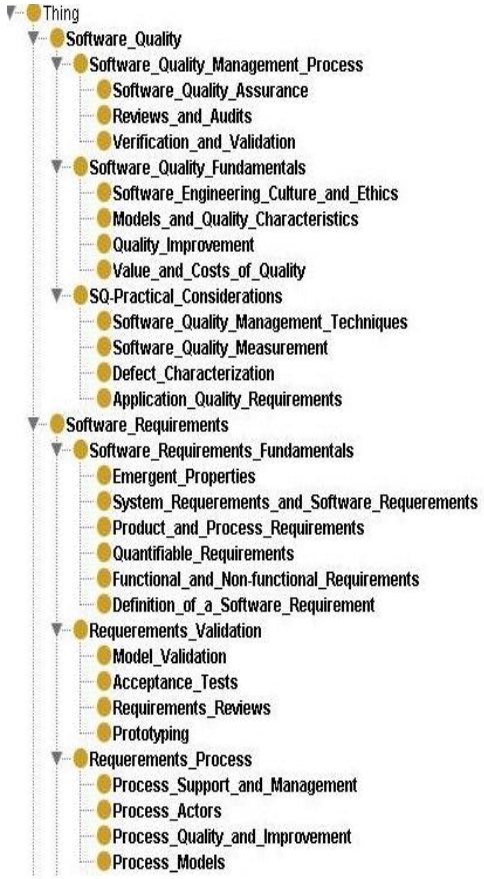

Fig.2. A fragment of the ontology for *Software Engineering* concepts

```
(defrule local_authority
 (declare (salience 1))
 (print go_to_local_authority)
 ?f <- (object (is-a Subject)
 (:NAME "LocalAuthority")
 (provides $?p)(releases $?r)
 (controls $?c))
 (not (answer ?))
 =>
 (printout t (slot-get ?f :NAME) " pro-
vides: " crlf)
 (foreach ?x $?p (printout t "       -
"(instance-name ?x)  crlf))
 (printout t "     releases: "  crlf)
 (foreach ?x $?r (printout t "       -
"(instance-name ?x) crlf))
 (printout t "     in accordance with:
"  crlf)
 (foreach ?x $?c (printout t "       -
"(instance-name ?x) crlf))
```

3. Intelligent information systems

An information system stores data, with the goal of providing information starting from that data, when needed. Therefore, we can identify three main features that characterize an information system: the way data is stored, the way new data is included, and the way information is provided as results to queries.

3.1 Storage alternatives

Many information systems are developed around a database. This case has the advantage of efficient implementations but, because data has a very precise and inflexible structure, any change in the conceptual schema and any intelligent dialogue are not possible.

In the context of the huge number of documents available now on the web, it is natural to consider the case of information systems using text as content instead databases. However, even if search engines like Google are able to provide a lot of useful information, a problem is that many times, such engines provide too much information, too much documents as a result to a query. This problem is due to the fact that documents are unstructured and that natural language processing is not able to handle ambiguity and context issues [11].

One of the ways to handle the problems of information retrieval from texts on the web is to add explicit knowledge descriptions (metadata) of the content of documents and to integrate this metadata in conceptual frameworks (ontologies) for each domain.

This approach is supported by the Semantic Web perspective, which aims at facilitating programs to process texts on the web [2]. A first step towards the Semantic Web is to annotate texts with XML [12]. Using RDF [8] to state facts about web resources is the second step. The next step is the development of ontologies using OWL [5].

3.2 Inclusion of new data

In databases, new data may be included only

as new instances, which strictly follow the fixed conceptual model. The case of texts is totally different, practically there are no restrictions. For example, the addition a new document on the web is not restricted by any conceptual model.

Ontology based storage, following the Semantic Web ideas, allows the addition of any document, but requires its metadata annotation. However, the conceptual model is not fixed, it can be changed, by modifying the ontology.

3.3 Intelligent query processing

In order to provide the needed data to various types of users and to different kinds of questions, an intelligent information system should provide several ways of interaction. For example, in the context of the web, a natural way is to browse pages containing useful information. In addition to classical browsing of a fixed structure of web pages, user's profile may be considered for generating a personalized structure of web pages, starting from the domain's ontology [10]. The ideal information system should be, however, able to enter into a dialog with the user, using his/her own language.

Any information act is, in fact, dialogistic. Moreover, as Bakhtin emphasized, any text is a dialog [1]. Even if you write something and you put something on the web, this is a potential dialog with the readers of the text.

Different ways of querying in information systems are, in our opinion, different ways of entering in dialog: a) database query, b) hypertext browsing, c) keyword-based search engine, d) intelligent search engine, e) expert system dialog, f) controlled natural language, g) question answering, h) natural language dialog.

From the above list, only natural language dialog and question answering are, at least for the moment, not satisfactory implemented. All the other ways of information querying are, more or less, possible to implement.

4. Conclusions and further developments

The paper has presented the framework of an intelligent information system that may be used for competence management. Declarative and procedural knowledge representation and processing may be performed under the Protégé system with a Jess plug-in.

The specific goals of a competence management system may be fulfilled by associating competencies in a domain with concepts in the domain. Moreover, more complex competence relations may be described as procedural knowledge using production rules. The resulting information system is offering advanced ways of user-system dialog, specific to expert systems, which are not available in usual applications, based solely on databases or simple information retrieval.

References

[1]Bakhtin, M.M., Speech Genres and Other Late Essays, University of Texas Press, 1986

[2]Berners-Lee, T., Hendler, J., and Lassila, O., The Semantic Web, Scientific American, May 2001.

[3]Gruber, T.R., http://www-ksl.stanford.edu/kst/what-is-an-ontology.html

[4]http://www.jessrules.com/jess

[5]Ontology Web Language - http://www.w3.org/2004/OWL/

[6]http://www.philosophy.umd.edu/deptwebsite/undergraduateprogram/resources/terms.html

[7]Protégé environment - http://protege.stanford.edu

[8]Resource Description Framework - http://www.w3.org/RDF/

[9]Guide to the Software Engineering. Body of Knowledge, www.swebok.org/ironman/pdf/SWEBOK_Guide_2004.pdf

[10] Trăuşan-Matu, Şt., Maraschi, D., Cerri, S., Ontology-Centered Personalized Presentation of Knowledge Extracted From the Web, Lecture Notes in Computer Science 2363, ISSN 03029743, Springer, 2002, pp 259-269

[11] Winograd, T., Flores, F., Understanding Computers and Cognition, Norwood, N.J.: Ablex, 1986.

[12] XML http://www.w3.org/XML/