Supporting User Roles in Ontology Fuzzification

[1]Manolis Wallace, [2]Panos Alexopoulos, [1]Ioannis Papafragkos and [1]Costas Vassilakis

[1]Department of Computer Science and Technology,
University of Peloponnese, Tripolis, Greece
Email:{wallace,pcst0924,costas}@uop.gr
[2]IMC Research
Fokidos 47, 11527, Athens, Greece
Email:palexopoulos@imc.com.gr

Abstract

Manual ontology development is clearly a strenuous task. Whilst a variety of ontological engineering methodologies exist, their actual application is far from trivial, mainly due to the widely diverse nature of the tasks involved. In this work we study these tasks and identify the different types of human experts that are best suited to perform each one. As a result, we present a cooperative version of an ontological engineering methodology, together with a graphical tool that supports it.

Keywords

ontology development; domain expert; knowledge engineer; user roles; fuzzification;

1 Introduction

Ontologies and ontology engineering are far from new and emerging as a field. They have been around for quite a while, and their role in modern computer science is by now well established [1][2]. With their applications ranging from semantic annotation [3], and document clustering [4] to decision support [5] and knowledge management [6] to list just a few, a great deal of attention has been given to them.

The field started out with a focus on designing ontology representation languages capable of describing the semantics of common and not so common human knowledge [7][8][9]. Important questions in this direction include the consistency of the representations [10] and the computability regarding its implicitly contained knowledge. At the same time, plenty of focus was also given to the practical aspect of developing ontology enabled semantic applications [11]. In the beginning these comprised mainly conventional applications remodeled in an ontology driven framework, while more recently genuinely semantic applications have emerged.

But the more progress was made in the technical side of ontological computing, the more evident it became that we were lacking on the methodological side. Although we had the languages to represent human knowledge and the algorithms and tools to exploit it, we were missing the ability to develop extensive, detailed, complete, consistent and correct ontologies. As a response to this gap, we have seen the development of a sequence of methogogologies that formalize the ontology development, extension and adaptation processes by organizing them in specific steps and tasks [12][13][14][15].

But even with these methodologies in hand, the actual development of a sizeable ontology remains a challenging task, not only because the tasks comprising these methodologies are quite abstract in their nature, but also because they are quite diverse. Whilst it may be easy to identify experts who can identify the fine differences between different types of red wines or others who can select the ideal ontological description structures for every situation, it is quite difficult to find people who combine such skills.

With this in mind, in this paper we shift our focus to the skills and characteristics involved in the process of ontological engineering. Examining a specific ontology engineering methodology (IKARUS-Onto [16]), we identify the distinct user roles that are associated with it and describe the ways in which such distinct users might cooperate in developing an ontology. Our work concludes with the presentation of a graphical tool designed and developed in order to support such diverse users in cooperatively developing an ontology.

The remainder of this paper is organized as follows: In section II we briefly discuss the different types of roles that are inherently associated with ontological engineering. Building on this, in section III we review the methodology that we will focus on and associate the tasks in comprises to the user roles of section II. Section IV presents the graphical tool that we developed in order to support users in this cooperative ontology engineering task whilst section V lists our concluding remarks.

2 Types of experts and their roles

An ontology engineering methodology may be viewed as an abstract description of a process that transfers knowledge from a human to a machine readable formalized structure. Intuitively we may conclude that for the process to be successful we need at least a person that holds the knowledge ontology, a way to extract this knowledge and a way to formalize the knowledge into an ontology.

The person holding the knowledge at the first place is commonly referred to as the "domain expert" and is an absolutely integral part of the process. This is the individual (or group of individuals) who is able to manually perform tasks similar to those that we hope an ontology driven machine/algorithm will be as a result of our work. Since semantic operations quite often rely on the identification of the finer differences in the subject at hand, a domain expert cannot really be substituted by a person with lesser understanding of the domain in question without sever damage to the quality of the resulting ontology.

On the other hand, the choice of the ontology representation language itself is also an important task that has a major influence on the applicability and effectiveness of the resulting ontology driven application. What is needed is an expert that will be able to select from the variety of available representation options (OWL, RDF, f-SHIN, Fuzzy OWL, etc) the one that best fits the requirements of the application to be developed as well as the nature of the knowledge that is available. This person is commonly referred to as the "knowledge engineer".

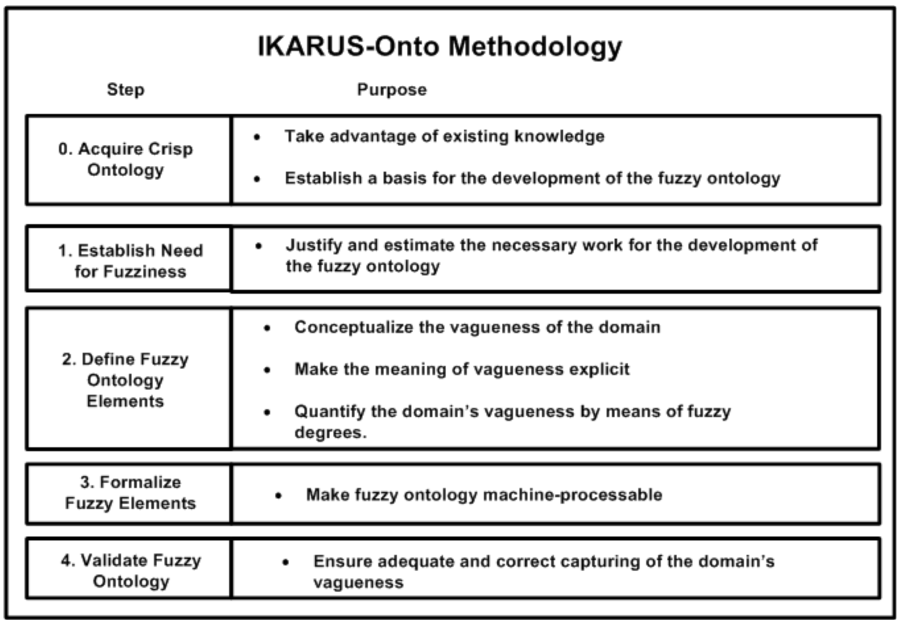
Clearly a combination of a domain expert and a knowledge engineer, although better than any of the two on their own, is still not sufficient for successful ontological engineering since having access to a domain expert is quite different to having the knowledge itself. Knowledge elicitation is a known bottleneck in knowledge systems engineering since the ideal domain expert is not always the ideal person to formally describe that knowledge as well. For example, when building a medical ontology we cannot expect that people who are leading physicians will necessarily also be able to formalize their knowledge so that it can be easily mapped to an ontological structure. Therefore we also need a person and/or tool to bridge the gap between the knowledge engineer and the domain expert.

Although "intermediate experts" have been considered (people who understand to some degree both the domain at hand and the ontology engineering procedure and can facilitate the exchange of information) a methodological tool may also prove useful. A methodological tool, i.e.

a formal ontology development methodology, that is well defined and designed in a way that is understandable and applicable for both types of experts can help them understand each other's role and needs in the process, so that their cooperation may be facilitated.

3 IKARUS-Onto and expert roles

In this work we focus on IKARUS-Onto, a methodology for the development of fuzzy ontologies. IKARUS-Onto assumes that a conventional ontology is available and describes the actions needed in order to generate its extended fuzzy version. Whilst at first this may strike one as a limited example of the aforementioned approach, it is worth noting that IKARUS-Onto is quite similar to conventional ontology engineering methodologies. In fact, it inherits the structure of METHONTOLOGY, which is one of the most acknowledged and applied ontology engineering methodologies [12].



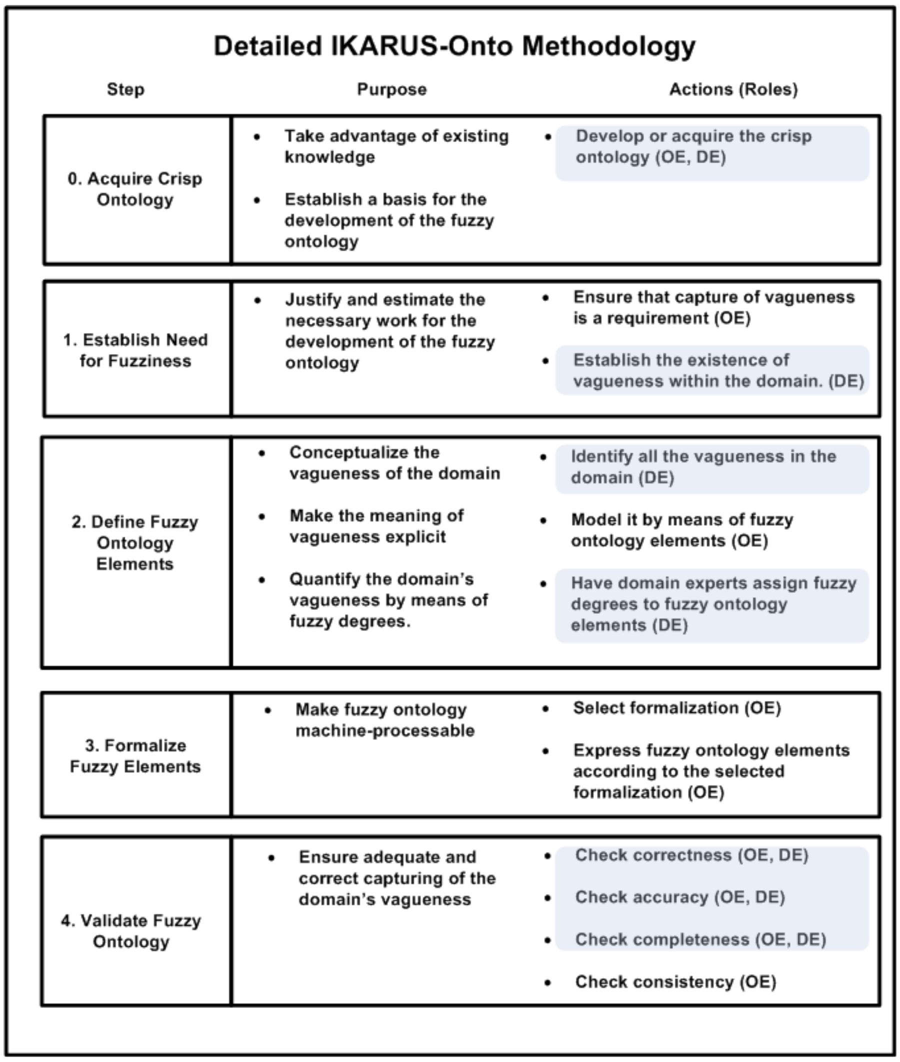
IKARUS-Onto Methodology

Step	Purpose
0. Acquire Crisp Ontology	• Take advantage of existing knowledge • Establish a basis for the development of the fuzzy ontology
1. Establish Need for Fuzziness	• Justify and estimate the necessary work for the development of the fuzzy ontology
2. Define Fuzzy Ontology Elements	• Conceptualize the vagueness of the domain • Make the meaning of vagueness explicit • Quantify the domain's vagueness by means of fuzzy degrees.
3. Formalize Fuzzy Elements	• Make fuzzy ontology machine-processable
4. Validate Fuzzy Ontology	• Ensure adequate and correct capturing of the domain's vagueness

Figure 1: Outline of the IKARUS-Onto methodology

In figure 11 we can see in summary the steps that comprise IKARUS-Onto. Step 0 corresponds to the development of the original conventional ontology and therefore falls outside the core of the methodology. It is worth noting though that it is a step that, as has already been mentioned earlier, cannot be perfectly executed by either an ontology engineer of a domain expert alone; their combined expertise is required.

Step 1 refers to establishing the need for fuzziness, and therefore to the need to actually apply the rest of the methodology and develop a fuzzy version of the ontology. Broken into distinct actions, this step includes a check that the intended application of the ontology is one where vagueness would play a role, i.e. a task for the ontology engineer, and a check that the domain does include vagueness, i.e. a task for the domain expert.



Detailed IKARUS-Onto Methodology

Step	Purpose	Actions (Roles)
0. Acquire Crisp Ontology	• Take advantage of existing knowledge • Establish a basis for the development of the fuzzy ontology	• Develop or acquire the crisp ontology (OE, DE)
1. Establish Need for Fuzziness	• Justify and estimate the necessary work for the development of the fuzzy ontology	• Ensure that capture of vagueness is a requirement (OE) • Establish the existence of vagueness within the domain. (DE)
2. Define Fuzzy Ontology Elements	• Conceptualize the vagueness of the domain • Make the meaning of vagueness explicit • Quantify the domain's vagueness by means of fuzzy degrees.	• Identify all the vagueness in the domain (DE) • Model it by means of fuzzy ontology elements (OE) • Have domain experts assign fuzzy degrees to fuzzy ontology elements (DE)
3. Formalize Fuzzy Elements	• Make fuzzy ontology machine-processable	• Select formalization (OE) • Express fuzzy ontology elements according to the selected formalization (OE)
4. Validate Fuzzy Ontology	• Ensure adequate and correct capturing of the domain's vagueness	• Check correctness (OE, DE) • Check accuracy (OE, DE) • Check completeness (OE, DE) • Check consistency (OE)

Figure 2: Detailed IKARUS-Onto methodology with reference to user roles

Step 2 is concerned with the actual specification of the vagueness in the domain in fuzzy terms. This is in fact the core of the work to be performed. When analyzed into distinct tasks, this step is broken down into:

- The identification of the areas of the original ontology where vagueness actually exists, which is a task ideally performed by the domain expert.
- The selection of the most mathematical model and ontological structure that best matches each case of vagueness, when considering both the semantics of the vagueness and the limitations or requirements of the intended application, which can only be performed by the ontology engineer.
- The specification of the exact fuzzy degree(s) that should be associated with each case of vagueness, which can only be performed by a domain expert, assuming of course that the domain expert is aware of the meaning that these degrees are expected to carry and the way in which these degrees will be interpreted when the ontology is put into practical application.

Clearly, this step cannot be performed by an ontology engineer or a domain expert alone.

Step 3 refers to the selection of the most suitable ontology representation language for the generated ontology. This selection is determined by the nature of the ontological structures that have been used in the previous steps as well as by the technical specification of the application in which the fuzzy ontology will be used. This is a clearly technical step that can be performed by an ontology engineer alone.

Finally, step 4, often omitted as optional, is the validation step, in which the output of the aforementioned tasks is checked for correctness, consistency etc. These checks range from purely technical ones, such as the consistency check, to heavily semantic ones, such as the accuracy check, and are therefore again performed by a combination on domain experts and ontology engineers.

In figure 22 we can see a graphical representation of the IKARUS-Onto methodology, on which the tasks to be implemented by domain experts are highlighted. One can easily see that the role of the domain expert is not limited to a single continuous segment of the process but is instead closely intertwined with the work of the ontology engineer.

Hence, the need for facilitated cooperation between users performing the two roles is evident.

4 Ontology fuzzification tool

In order to observe the application of the theory proposed herein in an experimental setting we have developed a graphical tool that implements the IKARUS-Onto methodology, while also taking into consideration and supporting distinct user roles. Specifically, the toll aims to organize the ontology fuzzification work around the IKARUS-Onto methodology, while at the same time supporting the domain experts in their role. Emphasis is put on the domain expert since on one hand there is already an abundance of tools to support the ontology engineer in his task and on the other the ontology engineer needs far less support in his work.

The ontology fuzzification tool is set up as a web interface in which a crisp ontology may be loaded. This ontology is visualized, so that users may graphically review it and specify the required updates and extensions. The visual approach to ontological editing makes it possible for domain experts who are laymen when it comes to computing to participate in the process. Additionally, the visual portion of the ontological editing process is not bound to a specific notation or ontology description language. Therefore the domain expert does not need to comprehend or even worry about specific language characteristics or limitations.

As we have already mentioned, step 0 of the methodology, as presented in section 3, is a preparatory step that is outside the core scope of IKARUS-Onto. In fact, within IKARUS-Onto step 0 is meant as the process in which a suitable base ontology is identified, so that the fuzzification task may be based on it. As far as the developed tool is concerned, step 0 corresponds to a UI that allows the user (may he be a domain expert or an ontology engineer) to load a conventional ontology into the tool; supported ways to load the ontology are directly copying its contents and providing a URI to the file that contains it.

The tasks included in steps 1 and 2 of the methodology are supported in greater detail as they are the core tasks of the ontology fuzzification process. As can be seen in figure 22 the work in steps 1 and 2 of the methodology can be organized as four intermitted phases of ontology engineer and domain expert work. This exact structure is followed by the developed tool. Specifically, the visualized ontology is first presented to the ontology engineer, so that he may assess the need to capture the related vagueness. Assuming the decision is to go ahead with the fuzzification process, the visualized ontology is presented to the domain expert in the UI that is shown in figure 33. In this UI the domain expert can visually specify the elements of the ontology for which some type of vagueness will need to be captured and modeled in the ontology.

Following the structure of the IKARUS-Onto methodology, the work is then transfered to the UI presented in figure 44. Here the ontology engineer is presented with the elements that have been “highlighted” by the domain expert. For each one of them, the ontology engineer can select the most suitable structure to model its vagueness. Additionally, the ontology engineer “annotates” his work by explaining the meaning that the specified degrees have and the way they will be interpreted when the ontology is put to actual use. It is exactly this information that will assist the domain expert in the next step to specify the fuzzy degrees in a meaningful, consistent and efficient manner.

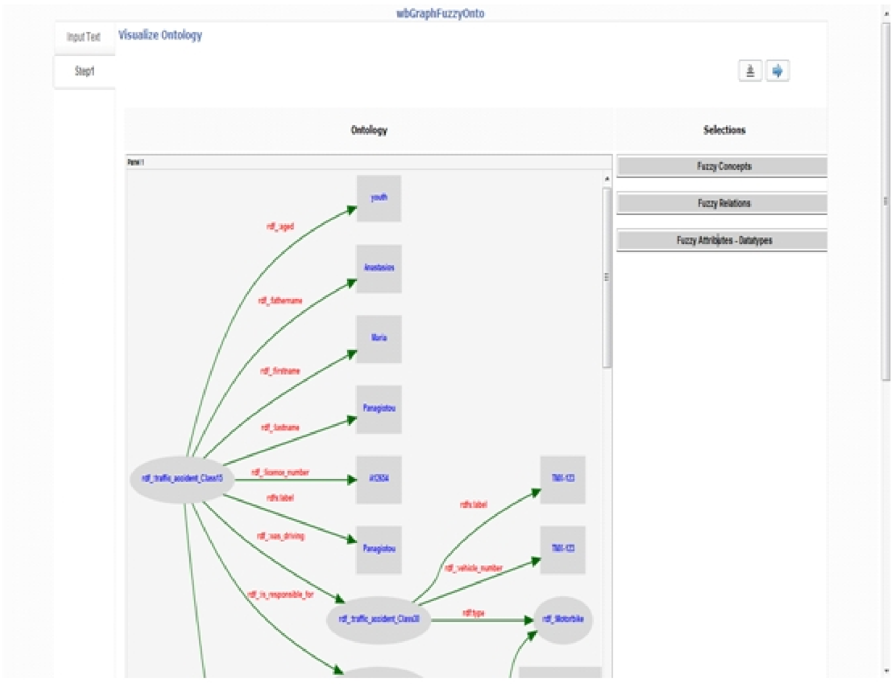
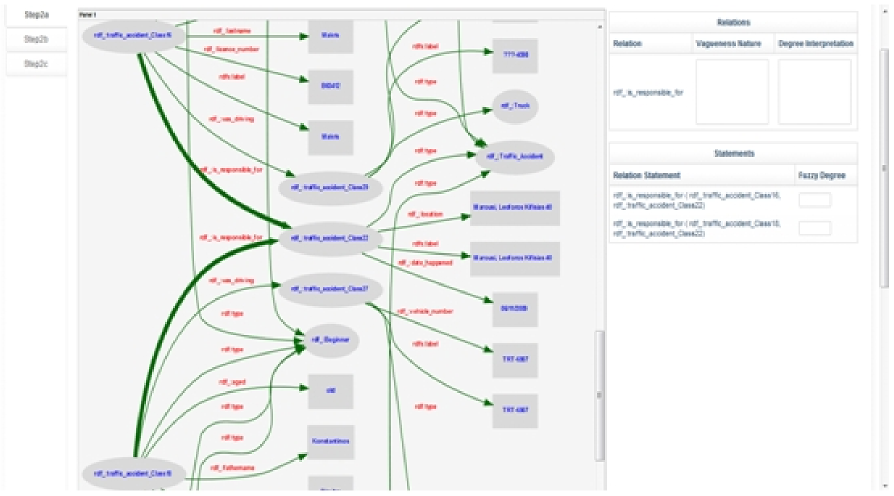


Figure 3: The domain expert is assisted in identifying the elements to fuzzify



Relations		
Relation	Vagueness Nature	Degree Interpretation
rdf_is_responsible_for		

Statements	
Relation Statement	Fuzzy Degree
rdf_is_responsible_for (rdf_traffic_accident_Class16, rdf_traffic_accident_Class22)	
rdf_is_responsible_for (rdf_traffic_accident_Class18, rdf_traffic_accident_Class22)	

Figure 4: The ontology engineer specifies the type of fuzziness for each element and its meaning

5 Discussion and conclusions

In this paper we discussed the different types of experts that are involved in the ontology engineering process. Continuing, we examined the different types of tasks involved specifically in the IKARUS-Onto methodology and identified the types of expert users that are most suited to perform each of the tasks. The result of this analysis has been an updated version of the methodology that is designed specifically for cooperative and interdisciplinary ontological engineering, as well as a first version of the graphical tool that supports it.

It is worth noting that, although much of the analysis has been focused specifically on IKARUS-Onto, the core of our proposal is directly applicable in any ontological engineering methodology. In fact, as part of our immediate future work we plan to apply our cooperative and interdisciplinary modifications to other methodologies, such as METHONTOLOGY. We also intend to develop the corresponding versions of our tool.

As far as the tool itself is concerned, as we have already mentioned the implementation of step 3 of the IKARUS-Onto methodology is a priority. We also have plans to develop our own generic formalization to store and export the intermediate results of the different ontological engineering steps (which are now kept in memory), so that distant users can cooperate asynchronously.

Finally, we intend to frame and offer our finalized tool as a free online service for any researcher and/or developer to use. This public version is intended to contain all the aforementioned extensions, so that it can be used in the widest possible range of situations.

References

[1] Borst W.N., Construction of Engineering Ontologies, Centre for Telematica and Information Technology, University of Tweenty, Enschede, The Netherlands, 1997.

[2] Chandrasekaran B., Josephson J.R., Benjamins V.R., What are ontologies and why do we need them, IEEE Intelligent Systems, pp 20-26, 1999.

[3] Sanchez D., Isern D., Millan M., Content annotation for the semantic web: an automatic web-based approach, Knowledge and Information Systems, pp. 1–26, 2010.

[4] Fodeh S., Punch B., Tan, P.-N., On ontology-driven document clustering using core semantic features, Knowledge and Information Systems, pp. 1–27, 2011.

[5] Bouamrane M.-M., Rector A., Hurrell M., Using OWL ontologies for adaptive patient information modelling and preoperative clinical decision support, Knowledge and Information Systems, pp. 1–14, 2010.

[6] Jurisica I., Mylopoulos J., Yu E., Ontologies for Knowledge Management: An Information Systems Perspective, Knowledge and Information Systems, vol. 6(4), pp. 380-401, 2004.

[7] Guarino N., Formal ontology, conceptual analysis and knowledge representation, International Journal of Human-Computer Studies, vol. 43(5-6), pp. 625–640, 1995.

[8] McGuinness D.L., Fikes R., Hendler J., Stein L.A., DAML+OIL: An Ontology Language for the Semantic Web, IEEE Intelligent Systems, vol. 17(5), pp. 72-80, 2002

[9] Horrocks I., Patel-Schneider P.F., van Harmelen F., From SHIQ and RDF to OWL: the making of a Web Ontology Language, Web Semantics: Science, Services and Agents on the World Wide Web, vol. 1(1), pp. 7-26, 2003.

[10] Baclawski K., Kokar M.M., Waldinger R.J., Kogut P.A., Consistency Checking of Semantic Web Ontologies, Proceedings of the International Semantic Web Conference, 2002.

[11] Analyti A., Antoniou G., Damásio C.V., Wagner G., Computability and Complexity Issues of Extended RDF, Proceedings of the 18th European Conference on Artificial Intelligence, 2008.

[12] Fernandez Lopez M., Gomez Perez A., Juristo N., Methontology: From ontological art towards ontological engineering, Proceedings of the Spring Symposium on Ontological Engineering of AAAI, 1997.

[13] Vrandecic D., Pinto H.S., Sure Y., Tempich C.,The DILIGENT knowledge processes, Journal of Knowledge Management, vol. 9(5), 2005.

[14] Jarrar M., Meersman R., Ontology Engineering -The DOGMA Approach, Advances in Web Semantics, vol I, LNCS 4891, Springer, 2008.

[15] Kotis K., Vouros G., Human-Centered Ontology Engineering: the HCOME Methodology, International Journal of Knowledge and Information Systems (KAIS), vol. 10, pp. 109-131, 2006.

[16] Alexopoulos P., Wallace M.,Kafentzis K. and Askounis D., A Methodology for Developing Fuzzy Ontologies, International Journal of Knowledge and Information Systems, in press.