The Use of Ontologies in Requirements Engineering

GJRE Classification (FOR) 091599

Verónica Castañeda[1], Luciana Ballejos[2], Ma. Laura Caliusco[3], Ma. Rosa Galli[4]

Abstract **- With the advent of the Semantic Web and the technologies for its realization, the possibilities for applying ontologies as a means to define the information and knowledge semantics become more and more accepted in different domains. The nature of requirements engineering involves capturing knowledge from diverse sources, including many stakeholders with their own interests and points of view. There are, therefore, many potential uses of ontologies in Requirements Engineering (RE). The purpose of this paper is to comprehensively review and present these uses. The main contribution is the classification of approaches that include ontologies within RE, with the aim of clarifying the way in which traditional RE techniques can benefit from them. Furthermore, future trends are identified**.

Keywords- Requirements Engineering, Ontologies, Framework

I. Introduction

Ontology can be defined as a specification of a conceptualization [1]. More precisely, ontology is an explicit formal specification of how to represent the entities that exist in a given domain of interest and the relationships that hold among them [2]. In general, for an ontology to be useful, it must represent a shared, agreed upon conceptualization.Ontologies have been used in many contexts and for many purposes throughout the years due to, principally, the advent of the Semantic Web [3]. Recently, the use of ontologies in software engineering has gained popularity for two main reasons: (i) they facilitate the semantic interoperability and (ii) they facilitate machine reasoning. Researchers have so far proposed many different synergies between software engineering and ontologies [4]. For example, ontologies are proposed to be used in requirements engineering [5], software implementation [6], and software maintenance [7] [8].There is an increasing amount of research devoted to utilizing ontologies in software engineering, and Requirements Engineering in particular. Thus, the main objective of this paper is to further examine this trend. The remainder of the paper is structured as follows: Section 2 presents the main concepts related to Requirements Engineering and Ontological Engineering. Section 3 analyzes the benefits of applying ontologies in Requirements Engineering and presents a framework for integrating ontologies in Requirements Engineering. In Section 4, ontologies in RE are presented. Finally, in Section 5, the conclusions and future trends are discussed.

About[1] -CIDISI- UTN - Facultad Regional Santa Fe vcastaneda@frsf.utn.edu.ar
About[2] -CIDISI- UTN - Facultad Regional Santa Fe – lballejos@santafe-conicet.gov.ar
About[3] -CIDISI- UTN - Facultad Regional Santa Fe – mcaliusc@frsf.utn.edu.ar
About[4] -INGAR-CONICET-UTN. – mrgalli@santafe-conicet.gov.ar

II. Background

a) Requirements Engineering

The primary measure for an information system to be successful is the degree in which it meets the intended purpose. Requirements Engineering (RE) is the process of discovering that purpose by identifying stakeholders and their needs, and documenting them for their future analysis, communication, and subsequent implementation [9]. RE is understood as a subtask of Software Engineering, which proposes methods and tools to facilitate the definition of all desired goals and functionalities of the software.Figure 1 shows an iterative cycle of core activities executed in RE [9]. All tasks presented in this figure generate diverse deliverables, in order to document obtained results along the RE process. There are diverse requirements specifications. They are mainly created in the "Requirements Representation" stage in Figure 1. These specifications are generally complementary, and very difficult to define. Thus, software engineers are often faced with the necessity to redesign and iterate due to the lack of information and differences in interpretation [10].Diverse other challenges must be faced during RE activities in order to generate, at early stages of software development, consistent and complete requirements and to efficiently feed subsequent stages. One of those challenges is the management of participating organizations (through their stakeholders) in requirements gathering, considering the frequent lack of technical knowledge.

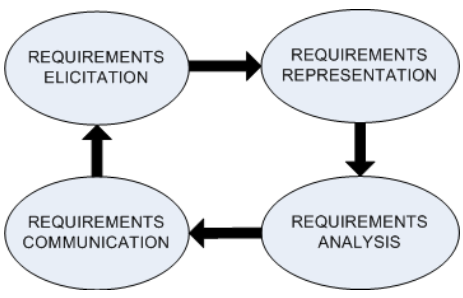


Fig.1. Requirements Engineering Activities.

Therefore, effective tools must be provided to achieve a complete analysis, considering particular and general needs and to manage requirements as a complete collaborative

process [11].Moreover, in RE processes there is a continual need for efficiently managing the great volume of information and knowledge generated and used during all activities presented in Figure 1. Thus, ambiguous requirements must be minimized since they produce waste of time and repeated work. They arise, for example, when different stakeholders produce different interpretations for the same requirement during the "Requirements Analysis" activity.

b) *Ontological Engineering*

The word ontology comes from the Greek ontos (being) and logos (word). It denotes the science of being and the descriptions for the organization, designation and categorization of existence [1]. Carried over to computer science in the field of artificial intelligence and information technologies, an ontology is understood as a representational artifact for specifying the semantics or meaning about the information or knowledge in a certain domain in a structured form [12]. Then, an ontology is used to reason about the properties of that domain, and might be used to describe the domain.Ontologies can be classified according to the task they are meant to fulfill [13]. Knowledge representation ontologies describe the modeling primitives applicable for knowledge formalization. Top-level ontologies, also called upper-level ontologies, try to comprehensively capture knowledge about the world in general, describing for example: space, time, object, event or action, and so forth, independently of a particular domain. Domain ontologies and task ontologies contain reusable vocabularies with their relations describing a specific domain or activity. They can specialize the terms of top-level ontologies.Several methodologies for developing ontologies have been described during the last decade [14] [15]. The objective of these methodologies is to define a strategy for identifying the key concepts that exist in a given domain, their properties and the relationships that hold between them; identifying natural language terms to refer to such concepts, relations and attributes; and structuring domain knowledge into explicit conceptual models. Two groups of methodologies can be figured out. The first one is the group of experience-based methodologies represented by the Grûninger and Fox methodology defined in the TOVE project [16] and by the Uschold and King methodology based on the experience of developing the Enterprise Ontology [17]. The second one is the group of methodologies that propose a set of activities to develop ontologies based on their life cycle and the prototype refinement, such as the METHONTOLOGY methodology [13], the Ontology Development 101 Method [18] and the methodology defined by Brusa et al. [19]. Usually, the first group of methodologies is appropriate when the purposes and requirements of the ontology are clear, while the second group is useful when the environment is dynamic and difficult to understand, and the objectives are not clear from the beginning [20].For ontology representation in a machine-interpretable way, different languages exist. Ontology languages are usually declarative languages commonly based on either first-order logic or on description logic. Ontology languages based on first-order logic have high expressive power, but computational properties such as decidability are not always achieved due to the complexity of reasoning [21]. The most popular language based on description logic is OWL DL, which have attractive and well-understood computational properties [22]. Another relevant language in Ontological Engineering is the Resource Description Framework (RDF). RDF was originally meant to represent metadata about web resources, but it can also be used to link information stored in any information source with semantics defined in an ontology. The basic construction in RDF is an (Object, Attribute, Value) triplet: an object O has an attribute A with value V. A RDF-triplet corresponds to the relation that could be written as *(O, A, V)*, such as for example *(Professor; teaches; ArtificialIntelligence)*.

III. Benefits of Applying Ontologies in RE

The study of an information system requirements should result in the establishment of well-defined functionalities and attributes agreed by the stakeholders. If the functionalities are defined as incomplete or incorrect, the software may not meet the expectations of users. Factors that could lead to an inadequate process of requirements elicitation can be:

- *Ambiguous Requirements*: which produce lost of time and repeated work. Their origin resides in the diverse stakeholders, who produce different interpretations of the same requirement. Moreover, one stakeholder can interpret the same requirement in diverse ways. The ambiguity conduces to mistaken product tests.
- *Insufficient Specifications*: they produce the absence of key requirements. This conduces to developers' frustration, because they base their work in incorrect suppositions and, so, the required product is not developed, which displeases the clients.
- *Requirements not completely defined*: they make impossible the project secure planning and its monitoring. The poor requirements understanding leads to optimistic estimations, which return against when the agreed limits are surpassed.
- *Dynamic and changing requirements*: which require constant requirements revision in order to help to understand new clients needs and to identify how they can be satisfied.

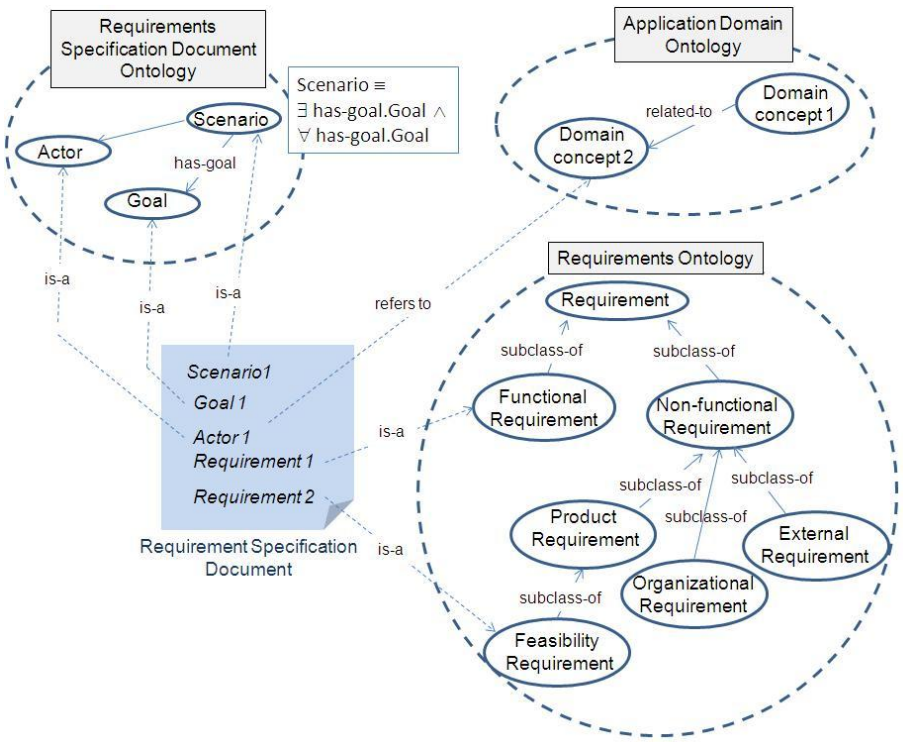
In order to reduce the negative effects of the previous factors on the RE processes, the ontologies can be used. The potential uses of ontologies in RE include the representation of: (i) The requirements model, imposing and enabling a particular paradigmatic way of structuring requirements, (ii) Acquisition structures for domain knowledge, and (iii) The knowledge of the application domain. Figure 2 shows a framework that depicts the interrelations between the ontologies previously described and a requirement

specification document. In this figure the following ontologies can be identified:

- *Requirements Ontology*. Requirement specifications are the descriptions of the desired software characteristics specified by the customers. This model can be defined using an upper-level ontology. For example, Figure 2 shows a portion of an ontology that describes the non-functional requirements defined by Sommerville [23]. This ontology can be used during the elicitation process to reduce ambiguous requirements and avoid incomplete requirements definitions. Restrictions about requirements can be defined in this ontology. They help in requirements validation and verification.
- *Requirements Specification Document Ontology*. In RE different approaches are used as intermediate steps for obtaining requirements. One of these approaches is the technique of scenarios [24], which are exemplary descriptions of the usage of the planned system to reach a defined goal. In Figure 2, a portion of an ontology that represents the semantics related to the scenario approach is presented. The use of ontologies for describing the structure of requirements specification documents reduce the insufficient requirements specifications. Furthermore, they can greatly help in the definition of several structures for showing the same knowledge, in order to, for example, involve all stakeholders in the analysis of elicited requirements. Moreover, they can also help in reusing structured representation for diverse objectives or projects, only changing their content.
- *Application Domain Ontology.* This ontology represents the application domain knowledge and business information required for building software applications in a specific domain. It also includes the semantic relationships established among their concepts from a real-world point of view. An application domain ontology is useful to identify dynamic and changing requirements since it helps to understand the domain.

The arrows between the requirements specification document and the ontologies in Figure 2 represent conceptual dependencies. These dependencies can be defined using the RDF language. For example:
(Scenario1, is-a, Scenario)
(Goal1, is-a, Goal)
(Actor1, refersto, DomainConcept2)
(Requirement1, is-a, FunctionalRequirement)
(Requirement2, is-a, FeasibilityRequirement)
Thus, defining the requirements by using the previous framework makes possible to trace dependencies among them, their sources and implementations. In Figure 2, portions of ontologies are presented as examples. In the following section research made towards ontologies in RE is presented. Diverse results can be used for implementing the proposed framework.

Fig. 2. Ontology-based framework for supporting semantics based Requirements Engineering



IV. Ontologies in RE

A. *Ontologies for Describing Requirements Specification Documents*

A well-characterized requirements specification is important to the design stage of software development and to the evaluation and reuse of elicited requirements. Specifications are formed of both, the document structure and its content. In this sense, Groza et al. [25] affirm that the structure of a document has a very important influence in the perception of its content. Reuse is one of the most required features for any software product. It is based on the form in which requirements are specified, documented and structured. Nevertheless, the reuse faces several challenges. These challenges are caused by insufficient support for its steps, such as search, evaluation and adaptation. One way of exchanging reusable requirements specification documents is through Wiki systems, which allow the self-organized reuse since the community provides and organizes the artifacts to be reused [26].The analysis of Wikis as solutions in this area is a very novel approach. The proposals conclude that requirements specification documents can specially benefit from ontologies, moreover when the content of those documents grows in a chaotic way. One way of solving this issue is structuring the knowledge by enriching the documents with additional metadata and finding interrelated useful content adding semantics to the documents extending the wiki with RDF, this way the semantic is expressed in a machine-understandable format. This solution is known as Semantic Wiki and can be considered as a lightweight platform.Another advantage of this approach is the automatic reasoning support and communication of used concepts.Furthermore, reuse cannot be possible if requirements documents do not have two main attributes carefully balanced, as described by Hull et al. [27]: readability and processability. They can be greatly enhanced by the use of ontologies in requirements documentation. One clear example is adapted by Decker et al. [26] from the Use Case approach. They add diverse documents and new structures to the traditional Use Cases documentation. These new documents are known as templates and allow to capture knowledge. Each one has metadata, besides the ontology of the documents. The authors also allow the extension of the ontology linking different Use Cases to facilitate the search of documents of the same type with other projects.Another approach that uses templates is proposed by Groza et al. [25]. They describe a solution for generating different representations of the same document, known as templates, based on the metadata created by using a particular authoring and annotation framework. Proposals like this can be of great help in order to represent RE specifications structures, thus promoting the reuse of RE specification content using diverse structures representations.As mentioned before, it is widely demonstrated that the use of ontologies helps stakeholders to clarify their information needs and comes up with semantic representations of documents. Dragoni et al. [28] for example, present an approach for the ontological representation and retrieval of documents and queries for Information Retrieval Systems using a vector space model which use concepts instead of terms, where the documents are represented in a conceptual
way, and the importance of each concept is calculated.
All these approaches can be, in some way, integrated in order to define an ontology for capturing the RE documents structures, and so, promoting the adaptation of the same content in diverse formats in order to be understandable by all stakeholders. Moreover, an ontology with this goal, can be reutilized in diverse projects in order to structure knowledge proper for each one.

B. *Ontologies for Formally Representing Requirements*

The use of ontologies for the representation of requirements knowledge has been under study since a long time ago. One of the initial approaches in this area was presented by Lin et al. [29]. They propose a generic solution that provides an unambiguous, precise, reusable and easy to extend terminology with dependencies and relationships among captured and stored requirements. The proposal can be applied to any kind of product to reach diverse requirements: communication, traceability, completeness, and consistency. It also supports the detection of redundant or conflicting requirements. The developed ontology is implemented using Prolog. The authors propose the use of first order logic to identify the axioms and capture the definition, constraints and relationships among the objects. They also allow integrity checking of the design knowledge. Besides being a very complete proposal, one of its disadvantages is that the involved terminology is only shared by the engineers of the project, and thus, the customer is not aware of it. This way, some requirements might stand ambiguous.The relationships among captured and stored requirements defines the traceability of the RE process. Traceability is the ability to describe and follow the life of software artifacts in Software Engineering [30]. More specifically in RE, those artifacts are the requirements. Thus, in order to trace requirements to their sources and to the intermediary and final artifacts generated from them all over the development process, it is mandatory to consider and represent information related to their source and the requirement's history.Traceability also facilitates the reuse of the requirements and the related information. In this sense, and promoting requirements reuse, Veres et al. [31] define diverse requirements models and give rules for the mapping and traceability among them. Also Decker et al. [26] promote reuse by establishing a common requirements structure to be considered along Software Engineering activities. This is related to which Brewster et al. [32] affirm, that to build systems that solve real-world tasks, not only conceptualizations must be specified, but also, clarity over the problem solving must be given.In this way, Riechert et al. [33] present a semantic structure for capturing

requirements relevant information, in order to support the RE process semantically and to promote the collaboration of all stakeholders in software development processes. They also apply and evaluate the proposal in an e-government case study.The KAOS (from Knowledge Acquisition in autOmated Specification) methodology is a goal-oriented requirements engineering approach with a rich set of formal analysis techniques [34]. KAOS is described as a multiparadigm framework that allows to combine different levels of expression and reasoning: semi-formal for modeling and structuring goals, qualitative for selection among the alternatives, and formal when needed for more accurate reasoning [35].All goal-oriented approaches are more applicable for complex systems. They are commonly based on the not easy task of identifying goals. Then, nonfunctional requirements (NFRs) are derived from them. Their analysis and management is much more difficult than the functional requirements ones.As a more specific approach for using ontologies for representing NFRs knowledge, Dobson and Sawyer [5] propose an ontology for representing dependability between requirements. It considers diverse NFRs, such as: availability, reliability, safety, integrity, maintainability, and confidentiality. Meanwhile, another proposal in this area is given by Kassab [36] who develops an ontology which provides the definition of the general concepts relevant to NFRs, without reference to any particular domain. He describes, through the proposed ontology, diverse glossaries and taxonomies for NFRs. The first ones are used for generalization to the common NFRs concepts.Considering the importance of knowledge reuse and its application in Requirements Engineering, Wouters et al. [14] point out that one of the biggest problems in reusing use cases was to find similar or related ones to reuse. Thus, and in order to accomplish the reuse, they propose a semiformal description which, used together with a "human" format, can make it possible the reuse of use cases. The defined ontology has three categories of information: labels,concepts and relations. With these concepts diverse rules and queries can be created which, under a logic inference machine and together with algorithms, make it possible to find similar use cases.

C. *Ontologies for Formally Representing Application Domain Knowledge*

Domain ontologies are specific, high-level models of knowledge underlying all things, concepts, and phenomena of a given domain of discourse. As with other models, ontologies do not represent the entire world of interest. Rather, ontologists select aspects of reality relevant to their task [37]. Then, the selection of the methodology to be used for developing an ontology depends on the application that ontologists have in mind and the extensions that they anticipate. In software development, an ontology can be used at development time or at run time [38]. Using an ontology during the development stage enables designers to practice a higher level of knowledge reuse than is usually the case in software engineering. At run time, an ontology may enable, for instance, the communication between software agents or be used to support information integration. In both cases, the creation of the ontology starts at the RE process.Any software development process implies multiple stakeholders which collaborate with a common goal. At development time, a domain ontology can be used as a way of facilitating the understanding between stakeholders. Pohl [39] affirms that RE must elicit and understand the requirements from the relevant stakeholders and develop the requirements together with them. Thus, in order to maximize environment comprehension, a common understanding of the involved concepts must be carried out. This means, the requirements analysts should be endeavored and must work towards understanding the language used in the universe of discourse, to then initiate its modeling. A model of the environment represents the reality and considerably improves its comprehension.Thus, a crucial part of RE is the establishment of a common terminology by diverse stakeholders. To this aim, the methodologies described in Section 2.2 can be used at the first stage of the software development process.The traditional methodologies for developing ontologies appear to be unusable in distributed and decentralized settings, and so the systems that depend on them will fail to cope with dynamic requirements of big or open user groups [40]. In this sense, Breitman and Sampaio do Prado Leite [41] propose a process for building an application ontology during the requirements process based on the Language Extended Lexicon (LEL). The lexicon will provide systematization for the elicitation, model and analysis of ontology terms. The underlying philosophy of the lexicon falls in the contextualism category, according to which particularities of a system's use context must be understood in detail before requirements can be derived. This approach is new to ontology building, which traditionally associates generalization and abstraction approaches to the organization of the information. Application ontologies are much more restricted than domain ontologies and have a much more modest objective. The authors see the ontology of a web application as a sub-product of the requirements engineering activity.

V. Conclusions and Future Trends

The paper describes the diverse challenges that must be faced during RE activities. As mentioned before, RE involves several activities to generate consistent and complete requirements representation and specification, but due to the fact that stakeholders belong to different backgrounds, in addition to the great volume of information that must be managed, the need of a framework that helps in the whole process is noticeable. It also synthesizes diverse specific proposals based on ontologies, which were developed in order to help in diverse RE aspects. Moreover, as shown in the article, these proposals can be clearly divided into three application areas, such us: the description of requirements specification documents, the formal representation of the application domain knowledge, and the

formal representation of requirements.Although the approaches show an advance towards the demonstration of the importance of implementing technologies in certain circumstances and RE activities, more effort is still needed in order to generate an integrated framework, capable of addressing these challenges in an integrated way, and of being applied all over the RE process and its activities. This is even more important if the persistence of requirements in all the software development process is considered.This framework and its predominant characteristics were simply described in this paper. Once developed and implemented, it will be useful in requirements consistent management, specification, and knowledge representation activities during the entire software development project.Thus, future work will be focused on generating support for the framework in order to enhance and integrate requirements structure ontology generation, requirements content ontology generation and requirements domain ontology generation. This will allow the collaboration of all stakeholders in the definition of requirements along all involved tasks, and moreover, to define a common structure and knowledge representation format, capable of being used in the entire software development process.

VI. References

1) Grûber, T.: A translation approach to portable ontology specification. Knowledge Acquisition 5(2) (1993) 199-220.
2) Smith, B., Kusnierczyk, W., Schober, D., Ceusters, W.: Towards a reference terminology for ontology research and development in the biomedical domain. In: Proc. of the 2nd Int. Workshop on Formal Biomedical Knowledge Representation: \Biomedical Ontology in Action". (2006) 57-66.
3) N. Shadbolt, W.H., Berners-Lee, T.: The semantic web revisted. IEEE Intelligent Systems 21(3) (2006) 96-101.
4) Happel, H.J., Seedorf, S.: Applications of ontologies in software engineering. In: International Workshop on Semantic Web Enabled Software Engineering (SWESE'06). (2006).
5) Dobson, G., Sawyer, P.: Revisiting ontology-based requirements engineering in the age of the semantic web. In: Proceedings of the International Seminar on Dependable Requirements Engineering of Computerised Systems. (2006).
6) Eberhart, A., Argawal, S.: Smartapi - associating ontologies and apis for rad. In: Proceedings of Modellierung. (2004).
7) Ankolekar, A.: Towards a Semantic Web of Community, Content and Interactions. PhD thesis (September 2005).
8) Dameron, O.: Keeping modular and platform-independent software up-to-date: Benets from the semantic web. In: Proceedings of the 8th International Protégé conference. (2005).
9) Nuseibeh, B., Easterbrook, S.: Requirements engineering: A roadmap. In: International Conference on Software Engineering - Conference on the Future of Software Engineering. (2000) 35-46.
10) Noppen, J., van den Broek, P., Aksit, M.: Imperfect requirements in software development. In P. Sawyer, B.P., (Eds.), P.H., eds.: Requirements Engineering for Software Quality - REFSQ 2007. Volume LNCS 4542, Springer-VerlagBerlinHeidelberg (2007) 247-261.
11) Lang, M., Duggan, J.: A tool to support collaborative software requirements management. Requirements Engineering Journal 6 (2001) 161-172.
12) Allemang, D., Hendler, J.A.: Semantic web for the working ontologist: Modeling in RDF, RDFS and OWL. Elsevier, Amsterdam (2008).
13) Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering. Springer, New York. USA. (2004).
14) Wouters, B., Deridder, D., Van Paesschen, E.: The use of ontologies as a backbone for use case management. In: Proceedings of European Conference on Object-Oriented Programming (ECOOP 2000). (2000).
15) Corcho, O., Fernández-López, M., Gómez Pérez, A.: Methodologies, tools and languages for building ontologies: where is the meeting point? Data and Knowledge Engineering 46 (2003) 41-64.
16) Grüninger, M., Fox, M.: Methodology for the design and evaluation of ontologies. In: Proceedings of the Workshop on Basic Ontological in Knowledge Sharing. (1995).
17) Uschold, M.: Building ontologies: towards a unifed methodology. In: Proceedings of 16th Annual Conference of the British Computer Society Specialists Group on Expert Systems. (1996).
18) Noy, N., McGuinness, D.: Ontology development 101: A guide to creating your first ontology (2001).
19) Brusa, G., Caliusco, M.L., Chiotti, O.: Towards ontological engineering: a process for building a domain ontology from scratch in public administration. Expert Systems: The Journal of Knowledge Engineering 25(5) (2008) 484-503.
20) Cristiani, M., Cuel, R.: A comprehensive guideline for building a domain ontology from scratch. In: Proceedings of International Conference on Knowledge Management (I-KNOW 04). (2004).
21) Brachman, R., Levesque, H.: Knowledge Representation and Reasoning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2004).
22) Smith, M., Welty, C., McGuiness, D.: Owl web ontology language guide. Recommendation W3C 2(1) (2004).
23) Sommerville, I.: Software engineering (5th ed.). Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA (1995).
24) Hadad, G., Doorn, J., Kaplan, G.: Explicitar Requisitos del Software usando Escenarios. In: Proceedings WER'09, Workshop on Requirements Engineering. (2009).
25) Groza, T., Schutz, A., Handschuh, S.: Salt: A semantic approach for generating document representations. In:

Proceedings of the 2007 ACM symposium on Document engineering. (2007) 171-173.
26) Decker, B., Rech, J., Klein, B., Hoecht, C.: Selforganized reuse of software engineering knowledge supported by semantic wikis. In: Proceedings of Workshop on Semantic Web Enabled Software Engineering (SWESE). (2005).
27) Hull, M., Jackson, K., Dick, A.: Requirements Engineering, Practitioner Series. Springer, New York. USA. (2002).
28) Dragoni, M., da Costa Pereira, C., Tettamanzi, A.: An ontological representation of documents and queries for information retrieval systems. In: Proceedings of the 1st Italian Information Retrieval Workshop (IIR10). (2010).
29) Lin, J., Fox, M., Bilgic, T.: A requirement ontology for engineering design. Manuscript, Enterprise Integration Laboratory, University of Toronto (1996).
30) Winkler, S., von Pilgrim, J.: A survey of traceability in requirements engineering and model-driven development. Software and Systems Modeling Theme Section (2009).
31) Veres, C., Sampson, J., Bleistein, S., Cox, K., Verner, J.: Using semantic technologies to enhance a requirements engineering approach for alignment of it with business strategy. In: Proceedings of 2009 International Conference on Complex, Intelligent and Software Intensive Systems. (2009).
32) Brewster, C., O'Hara, K., Fuller, S., Wilks, Y., F.E., Musen, M., Ellman, J., Buckingham Shum, S.: Knowledge representation with ontologies: The present and future. IEEE Intelligent Systems 19(1) (January/February 2004) 72-81.
33) Riechert, T., Lauenroth, K., Lehmann, J., Auer, S.: Towards semantic based requirements engineering. In: Proceedings of the 7th International Conference on Knowledge Management (I-KNOW). (2007).
34) Lapouchnian, A.: Goal-oriented requirements engineering:an overv iew of the current research. Technical report, University of Toronto, Department of Computer Science (2005).
35) van Lamsweerde, A., Letier, E.: From Object Orientation to Goal Orientation: A Paradigm Shift for Requirements Engineering. In: Radical Innovations of Software and Systems Engineering in the Future. Volume 2941 of Lecture Notes in Computer Sciences. Springer Berlin / Heidelberg (2004) 153- 166.
36) Kassab, M. Non-Functional Requirements – Modeling and Assessment. VDM Verlag Dr. Müller Aktiengesellschaft, 2009.
37) Devedzic, D.: Understanding ontological engineering. Communications of the ACM 45(4) (2002) 136-144.
38) Fonseca, F.: The double role of ontologies in information science. American Society for Information Science and Technology 58(6) (2007) 786-793.
39) Pohl, K.: Requirements Engineering. Dpunkt Verlag (2007).
40) Davies, J., S.R., Warren, P.: Semantic Web Technologies: Trends and research in ontology-based systems. John Wiley, England (2006).
41) Koogan Breitman, K., Sampaio do Prado Leite, J.: Ontology as a requirements engineering product. In: Proceedings of the 11th IEEE International Requirements Engineering Conference. (2003).