Development and Evaluation of a Software Requirements Ontology

He Tan[1], Muhammad Ismail[1], Vladimir Tarasov[1] Anders Adlemo[1] and Mats Johansson[2]
[1]*Department of Computer Science and Informatics, School of Engineering, Jönköping University, Sweden*
[2]*Saab Avionics Systems, Jönköping, Sweden*
{*he.tan, muhammad.ismail, vladimir.tarasov, anders.adlemo*}*@ju.se,* {*mats.e.johansson*}*@saabgroup.com*

Keywords:
ontology, software requirements, ontology development, ontology evaluation, avionics software development

Abstract:
This paper presents an ontology which has been developed to represent the requirements of a software component pertaining to an embedded system in the avionics industry. The ontology was built based on the software requirements documents and was used to support advanced methods in the subsequent stages of the software development process. In this paper it is described the process that was used to build the ontology. Two pertinent quality measures that were applied to the ontology, i.e. usability and applicability, are also described, as well as the methods used to evaluate the quality measures and the result of these evaluations.

1 Introduction

One commonly cited definition of ontology is: "*an ontology is a formal explicit specification of a shared conceptualization*" (Studer et al., 1998), which merges the two definitions from Gruber (Gruber, 1995) and Borst (Borst, 1997). By explicitly modelling domain knowledge in a machine readable format, ontologies provide the possibility for representing, organizing and reasoning over the knowledge of a domain of interest, and can serve as a basis for different purposes.

Requirements engineering (RE) is an area concerned with elicitation, specification and validation of the requirements of software systems (Nuseibeh and Easterbrook, 2000). The use of ontologies in RE can date back to the 1990s, e.g. (Mylopoulos et al., 1990; Greenspan et al., 1994; Uschold and Gruninger, 1996). There exists a clear synergy between the ontological modelling of domain knowledge and the modelling of requirements performed by requirement engineers (Dobson and Sawyer, 2006). Recently, the interest in utilizing ontologies in RE, as well as software engineering in general, has been renewed due to the emergency of semantic web technologies (Dobson and Sawyer, 2006; Happel and Seedorf, 2006).

In this paper we present an ontology which has been developed to represent the software requirements of an application from the avionics industry. In the avionics industry, many of the systems are required to be highly safety-critical. For these systems, the software development process must comply with several industry standards, like DO-178B (RTCA, 1992). The requirements of the entire system, or units making up the system, must be analyzed, specified and validated before initiating the design and implementation phases. In our work an ontology was developed representing the requirements of a software component pertaining to an embedded system. The requirements are provided in natural language documents and have been manually validated by domain experts within the avionic industry. The requirements ontology developed based on these requirements is to be used in the subsequent stages of the software development process in order to support advanced approaches. To be more specific, the ontology is used to automate a part of the testing process in our work, namely the creation of test cases generation.

The evaluation of an ontology is very important to demonstrate its correctness and usefulness and many publications have been presented over the years within this area of ontology research. The focus of the evaluation in this work is on the usefulness of the requirements ontology. This is due to the fact that, in most cases, the users of an ontology are normally not the people who developed the ontology. This is also true for the requirements ontology presented in this paper where the users can be quality engineers or testers within the avionic industry. In our work, even a program can take the ontology as input to automate a part of the testing process. The evaluation results presented in this paper are focused on the evaluation of the user experiences when using the requirements ontology and, in particular, we consider the usability and applicability of the ontology.

The rest of the paper is organised as follows. Section 2 presents related work in the field of requirements ontologies, ontology development methods and ontology evaluations. In section 3 we present the developed requirements ontology. In section 4 we describe the ontology development method used. The evaluation of the ontology from a user perspective as well as the validation of the output when applying inference rules on the ontology are presented and discussed in section 5. Section 6 presents our conclusions and discussions on future work.

2 Related Work

In this section we present pertinent references to related work, specifically in the areas of requirements ontologies, ontology development methods and ontology evaluation.

2.1 Requirements Ontologies

A lot of research has been done on the application of ontologies in RE (e.g. (Lin et al., 1996; Mayank et al., 2004; Siegemund et al., 2011)). Much of the research deals with inconsistency and incompleteness problems in requirement specifications. For example, in (Lin et al., 1996), an ontology is proposed to support the cooperation between requirements engineers. The ontology provides a formal representation of the engineering design knowledge that can be shared among engineers, such that the ambiguity, inconsistency, incompleteness and redundancy can be reduced when engineers concurrently develop requirements for sub-systems of a complex artifact.

In other related work, such as described in (Greenspan et al., 1994; Mayank et al., 2004), ontologies are proposed to provide a framework for the requirements modeling in order to support the requirements engineering process. Such a framework can help mitigating the difficulties encountered in requirements engineering, such as negotiating a common understanding of concepts, dealing with ambiguity, and clarifying desires, needs and constraints.

Another direction of the application of ontologies in RE is to represent requirements in formal ontology languages in order to support consistency checking, question answering, or inferring propositions (e.g.(Mylopoulos et al., 1990; Moroi et al., 2012)). Most of the work within this direction is focused on analysis of consistency, completeness and correctness of requirements through reasoning over requirements ontologies. The work presented in this paper is also concerned with representing requirements in an ontology, but the ontology is mainly employed to support advanced methods in the subsequent stages of the software development process.

2.2 Ontology Development Methods

A number of methods have been put forward for building ontologies. Ontology 101 (Noy and McGuinness, 2001) proposes a very simple but practical guide for building an ontology using an ontology editing environment, such as Protége (Gennari et al., 2003). Another famous method in ontology development is METHONTOLOGY (Fernández-López et al., 1997). METHONTOLOGY contributes with a general framework for ontology development, which defines the main activities that people need to carry out when building an ontology, and outlines three different processes: management, technical, and supporting. The OTK Methodology (Fensel et al., 2000) is focused on application-driven ontology development. According to this method, engineering and industrial experts should actively be involved in the development of an ontology, in particular during the early stages of ontology engineering. There exist also other methods, e.g. (Presutti et al., 2009), that introduce Extreme programming, which initially was a software development method, to ontology engineering. These methods focus on a

collaborative, incremental, and iterative process of ontology development.

Unfortunately, no one single ontology development method was sufficient to guide us in our ontology development process. In section 4 we briefly present the development process used in our research, a process that combines features from the methods described above.

2.3 Ontology Evaluation

The challenge in ontology evaluation is to determine the quality features that are to be evaluated as well as the evaluation method. Many different evaluation criteria have been discussed in literature (e.g. (Gómez-Pérez, 2004; Gangemi et al., 2006; Vrandečić, 2009)). Which quality features to evaluate depend on various factors, such as the type of ontology, the focus of an evaluation and who is performing the evaluation. Burton-Jones et al. argue that different types of ontologies require different evaluation criteria (Burton-Jones et al., 2005). For example, for an application or domain ontology it is enough to evaluate the features important to the application or domain. Gangemi et al. (Gangemi et al., 2006) propose that the feature of ontology quality can be considered in three main dimensions: the structural dimension focusing on syntax and formal semantics, the functional dimension related to the intended use of an ontology and the usability-profiling dimension focusing on the communication context of an ontology.

Most ontology evaluation methods can fall into one of the following categories (Brank et al., 2005; Tan and Lambrix, 2009; Hlomani and Stacey, 2014): 1) Gathering statistics about the nature of ontologies. For example, in (Gangemi et al., 2006) the authors propose a set of metrics to measure the cohesion in an ontology, such as the depth, width, fan-out of an ontology, etc. However, it is often hard to validate how these statistics are capable of indicating the quality of an ontology. 2) Comparing an ontology to a gold standard. The limitation of this method is that it is difficult to establish the gold standard and furthermore the gold standard itself needs to be evaluated. 3) Comparing an ontology against a source of data stemming from the domain that the ontology strives to model. The assumption behind this method is that the domain knowledge is constant. However, the domain knowledge can at times be dynamic. 4) Evaluation performed by a human evaluator. The evaluation can be performed by an ontology expert, a domain expert, or a user of an ontology. The measurement of ontology quality is based on a set of predefined criteria, standards, or requirements. As a consequence, the evaluation results can be subjective because of the background of the evaluator. 5) Application-dependent evaluations. For example, in (Clarke et al., 2013) the quality of the gene ontology is evaluated regarding the effectiveness of the ontology and its annotations performed in the gene-set enrichment analysis experiment. The results from application-dependent evaluations are applicable to at least one context, but they cannot be generalized to other contexts. It is also expensive to perform application-dependent evaluations on many ontologies.

The ontology developed in this work is a domain ontology that has been used to support a particular application. In section 5 we will present the quality features we consider pertinent to evaluate as well as the evaluation methods used.

3 A Requirements Ontology in Avionics Industry

In the research project undertaken, an ontology representing the requirements of a telecommunication software component pertaining to an embedded system used in an avionics system has been developed. The software component is fully developed in compliance with DO-178B standard. As defined in DO-178B, the requirements of the software component have been prepared, reviewed and validated. The requirements ontology for the software component has been developed to support the automation of a part of the software development process.

The ontology includes three pieces of knowledge: 1) a meta model of the software requirements, 2) the domain knowledge of the application, e.g. general knowledge of the hardware and software, the electronic communication standards etc., and 3) each system requirements specifications.

The current version of the ontology contains 42 classes, 34 object properties, 13 datatype properties, and 147 instances in total. Figure 1 presents the meta model of the software requirements. As indicated in the figure, each requirement is concerned with certain functionalities of the software component. For example, a requirement may be concerned with data transfer. Each

requirement consists of at least requirement parameters, which are inputs of a requirement, requirement conditions, and results, which are usually outputs of a requirement, and exception messages. Some requirements require the system to take actions. Furthermore, there exists traceability between different requirements, e.g. traceability between an interface requirement and a system requirement. Figure 2 illustrates an ontology fragment of the domain knowledge about the telecommunication software component. Figure 3 shows the ontology fragment for one particular functional requirement, i.e. SRSRS4YY-431. The ontology fragments for all the other individual requirements are similar to this one. The functional requirement defines that if the communication type is out of its valid range, the initialization service shall deactivate the UART (Universal Asynchronous Receiver/Transmitter), and return the result "comTypeCfgError". In figures 1 to 3, the orange boxes represent the concepts of the ontology; the white boxes represent the instances; and the green boxes provide the data values of the datatype property for instances.

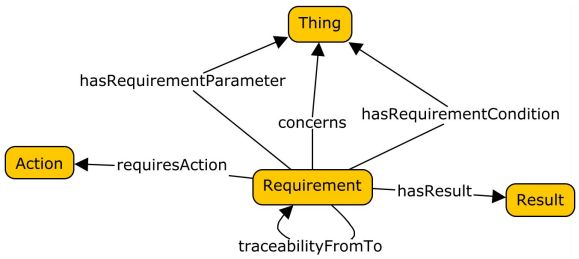

Figure 1: The meta model of the software requirements in the ontology

4 The Ontology Development Method

The general development process applied during the creation of the requirements ontology is illustrated in Figure 4. During the development of the ontology, the developers worked as a pair and followed an iterative and incremental process. In each iteration, the developers basically followed the steps suggested in Ontology 101, and considered the activities described in the supporting process in METHONTOLOGY. Lightweight competence questions (CQs) were used as a guidance to build the ontology. These CQs are simple and quickly prepared. In each iteration the developers had an opportunity to meet with the industry experts and discuss the issues they had encountered during the acquisition and specification steps. The developers also received feedback from the users of the ontology, and modified the ontology when needed.

The development tool used was the Protégé. HermiT reasoner (Shearer et al., 2008) was used to check the consistency of the ontology. Finally, the ontology was written in OWL (Bechhofer, 2009).

5 Evaluation of the Ontology

Any type of application domain model has to be evaluated, to demonstrate its appropriateness for what it was contemplated. As described in section 2, there exist a great number of different principles for the evaluation of the quality of an ontology. However, the focus in this paper is on two specific principles, the usability and the applicability of an ontology. The ontology that has been developed in the research project can be considered as being both a domain ontology and an application ontology, as illustrated in Figure 2 and 3. In this paper we define usability as a set of attributes that describe the effort needed by a human to make use of the ontology. The evaluation of the usability is performed by humans, i.e. typical potential users of the ontology. We define applicability as the quality of the ontology being correct or appropriate for a particular application domain or purpose. The evaluation of the applicability of the ontology is carried out by a developer of a software component that uses the ontology to implement its functionality.

5.1 Evaluation of Usability

The evaluation of the usability of a product or system is something that goes back in time. In 1986, Brooke developed a questionnaire, the System Usability Scale (SUS) (Brooke, 1996). During the years since then, it has been demonstrated that the SUS is applicable over a wide range of systems and types of technology and that it produces similar results as more extensive attitude scales that are intended to provide deeper insights into a users attitude to the usability of a systems. SUS also has a good ability to discriminate and identify systems with good and poor usability (Brooke, 2013). In this work we make use of the version of SUS introduced in (Casellas, 2009).

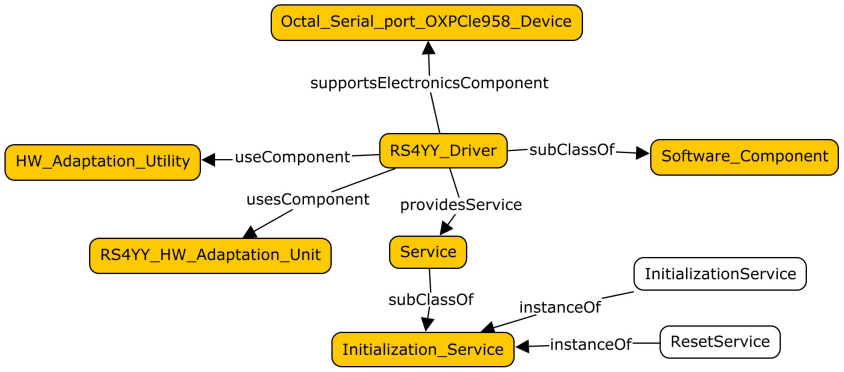

Figure 2: Ontology fragment for the domain knowledge

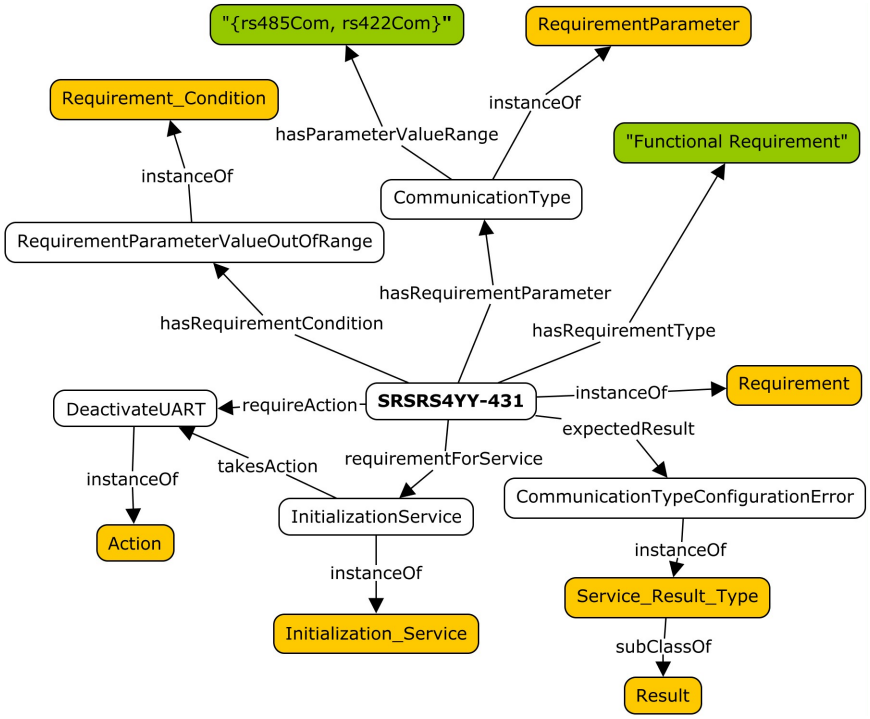

Figure 3: Ontology fragment for a requirement specification SRSRS4YY-431

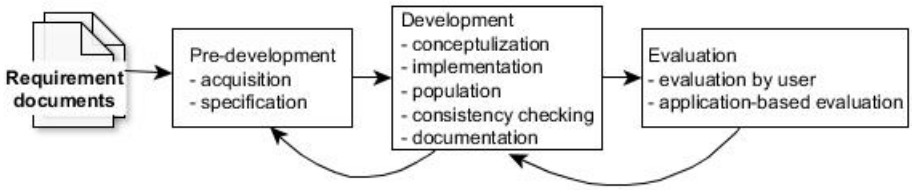

Figure 4: The ontology development process

Statements to evaluate the usability of the requirement ontology		AE	OE
1	I think that I could contribute to this ontology	3	5
2	I found the ontology unnecessarily complex	3	2
3	I find the ontology easy to understand	4	4
4	I think that I would need further theoretical support to be able to understand this ontology	2	1
5	I found that various concepts in this system were well integrated	4	4
6	I thought there was too much inconsistency in this ontology	2	1
7	I would imagine that most domain experts would understand this ontology very quickly	4	3
8	I find the ontology very cumbersome to understand	2	1
9	I am confident I understand the conceptualization of the ontology	4	5
10	I needed to ask a lot of questions before I could understand the conceptualization of the ontology	2	1

Table 1: Ontology usability evaluation. (AE:Application Domain Expert, OE:Ontology Expert, score: 1=strongly disagree, 2=disagree, 3=no preference, 4=agree, 5=strongly agree)

The scale, including its ten questions and the result, is presented in Table 1. The texts in the ten questions have only been slightly modified to adjust to the domain of ontologies.

The evaluation of the usability of an ontology is especially important when the ontology is going to be used by domain experts who are normally not ontology experts. Hence, the ontology was evaluated by two persons, one being an application domain expert (in software testing) and the other being an ontology expert, in order to compare their different views on the usability of the ontology. An evaluation of only two persons will only provide an indication of the usability of the ontology. However, Tullis and Stetson (Tullis and Stetson, 2004) has shown that it is possible to get reliable results with a smaple of 8-12 users. As a consequence, we are planning a more extensive evaluation in the near future.

The statements at odd numbered positions in Table 1 are all in positive form and the even numbered positions are all in negative form, as defined in SUS. The reason for this alternation is to avoid response biases, especially as the questionnaire invites rapid responses by being short; by alternating positive and negative statements, the goal is to have respondents read each statement and make an effort to reflect whether they agree or disagree with it.

The scoring is calculated as described in (Brooke, 1996). When applying the scoring procedure on the result presented in the table, the SUS scores indicate that the usability result for the application domain expert was 70 while the ontology expert had a result of 87.5. This implies that the usability of the ontology from the application domain expert's point of view was in the 57 percentile rank (i.e. seen as being "acceptable" (see the red line in Fig 5)) while the usability of the ontology from the ontology expert's point of view was in the 97 percentile rank (i.e. seen as being close to "the best imaginable" (see the blue line in the figure)) . A closer analysis of the answers from the two experts indicate the following.

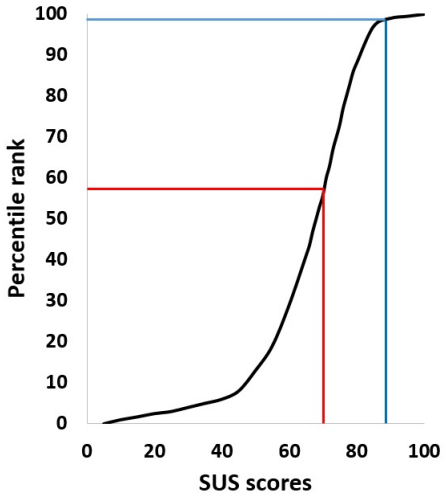

Figure 5: Percentile rankings of SUS scores (Sauro, 2011). (Red line: AE evaluation result, blue line: OE evaluation result)

Application domain expert: Due to inexperience in reading and using ontologies in testing activities at first it was not obvious how to apply the requirement ontology. However, after a reasonable amount of time, the almost one-to-one correspondence between the requirements identified in the software requirements specification

and the corresponding parts found in the ontology, made it quite straightforward to understand their interrelationship. Thus, the complexity of the ontology was not considered to be overwhelming. To overcome the initial problems in understanding some of the technicalities within the ontology, several questions had to be posed to the ontology experts. As a consequence, some extra capacitation in the field of ontologies would have been helpful. To sum up; "*it is my strong belief that I have a fairly good understanding of the conceptualization of the requirement ontology*".

Ontology expert: By using the ontology in the application, it was easy to determine changes that would improve the usability of the ontology for software testing. These changes were mostly about adding new instances and object and datatype properties. In general, it was easy to understand the ontology by exploring it in Protégé. But some parts required extra explorations by using visualization tools or even looking at the OWL functional serialization. As soon as the evaluator is an experienced ontology/knowledge engineer, no extra theoretical support is required. But it could be different for a less experienced ontology engineer. Most concepts were well integrated but some of them needed an extra integration, mainly through object properties. No inconsistencies were found, just some entities missing from the application viewpoint. Most ontology engineers would understand the ontology quickly but, still, they would need some extra time because the domain represented by the ontology is complex. The evaluator did not have to ask a lot of questions because the documentation was available in addition to the ontology itself. However, questions were needed to ask to understand how the ontology could be improved from an application viewpoint.

5.2 Evaluation through Application

According to our definition of applicability, the ontology should exhibit the quality of correctness or appropriateness when used for a particular application domain or purpose. To evaluate the applicability of the requirements ontology, it has been used for automatic generation of software test cases based on the requirements. The generation of test cases has been done by application of inference rules to the ontology. Each inference rule represents a part of a strategy for test case generation, and is formulated in terms of the entities in the requirements ontology. The entities are retrieved from the ontology to check conditions of the rules as well as to construct different test case parts. A test case is generated by applying several inference rules that traverse different ontology paths starting from the instance that represents the software requirement.

The inference rules were implemented in the Prolog language. To make it work, the requirements ontology was first saved in the OWL functional syntax (Motik et al., 2015). Then it was translated into the Prolog syntax(Bratko, 2001). As a result, OWL statements from the requirements ontology could be directly used in the inference rules. During the first experiment, 40 inference rules were used to generate 18 test cases for 15 requirements. 66 distinct entities form the ontology were used for the test case construction. The test cases were generated as plain text in English. The experiment showed an almost one-to-one correspondence between the texts in the generated test cases and the texts provided by one of our industrial partners.

The evaluation showed that the developed requirements ontology can fulfil its purpose, that is to support different stages of a software development process. The ontology has been used for automation of a part of the testing process and allowed for successful generation of test cases. The test cases generated by application of the inference rules to the ontology were almost the same as the ones manually constructed by the testers. The ontology allowed for a straightforward way of formulating inference rules. It was fairly easy to integrate the ontology in the OWL functional syntax in the Prolog program containing the inference rules. The OWL expressions were directly employed in the inference rules (after small syntactic changes in the translation phase) thanks to the availability of instances in the ontology. Exploring the ontology paths allowed to capture strategies for test case generation. The minor deficiencies in the ontology that were discovered during the development of inference rules were addressed in the following iterations.

6 Conclusion and Future Work

In this paper we have presented an ontology which was developed to represent the software requirements found in an embedded system in the avionics industry. The ontology was built from software requirements documents. From

an ontology development perspective, we propose a method which combines features from several different ontology development methods. The combined method is very practical and provides the necessary guidance for developing a requirements ontology from requirements documents for a particular application or purpose. One direction of the future work is to evaluate and improve the method through similar ontology development tasks.

Until now, very little work has been performed on the evaluation of an ontology from a user or an application's point of view. In this work we define two quality features pertinent to the evaluation of an ontology, i.e. usability and applicability, along with the evaluation method used. In the future we will continue the investigation of possible methods for evaluating the usability and applicability of ontologies. We would also like to develop guidelines to facilitate the evaluation of the quality of ontologies in connection with usability, applicability and other quality features with a focus on user experiences.

The ontology created from the software requirements documents is intended to support the automation of the different tasks in the subsequent stages of software development process, in order to reduce the cost of the software development. However, the development of an ontology itself is an expensive and difficult task. Hence, another direction in our future work is the automation of building ontologies from software requirements documents. Thus, when the ontology-based approach for software engineering is deployed in an industry environment, the real cost of the software development activities can be significantly reduced.

Acknowledgment

The research reported in this paper has been carried out in the project "Ontology-based Software Test Case Generation", which was financed by grant KKS-20140170 from the Knowledge Foundation in Sweden.

REFERENCES

Bechhofer, S. (2009). OWL: Web ontology language. In *Encyclopedia of Database Systems*, pages 2008–2009. Springer.

Borst, W. N. (1997). *Construction of engineering ontologies for knowledge sharing and reuse*. Universiteit Twente.

Brank, J., Grobelnik, M., and Mladenic, D. (2005). A survey of ontology evaluation techniques. In *Proceedings of the conference on data mining and data warehouses (SiKDD 2005)*, pages 166–170.

Bratko, I. (2001). *Prolog Programming for Artificail Intelligence*. Pearson Education, 4th ed. edition.

Brooke, J. (1996). SUS-A quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7.

Brooke, J. (2013). SUS: a retrospective. *Journal of usability studies*, 8(2):29–40.

Burton-Jones, A., Storey, V. C., Sugumaran, V., and Ahluwalia, P. (2005). A semiotic metrics suite for assessing the quality of ontologies. *Data & Knowledge Engineering*, 55(1):84–102.

Casellas, N. (2009). Ontology evaluation through usability measures. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 594–603. Springer.

Clarke, E. L., Loguercio, S., Good, B. M., and Su, A. I. (2013). A task-based approach for gene ontology evaluation. *Journal of biomedical semantics*, 4(1):1.

Dobson, G. and Sawyer, P. (2006). 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 at NPPs*, pages 27–29.

Fensel, D., Van Harmelen, F., Klein, M., Akkermans, H., Broekstra, J., Fluit, C., van der Meer, J., Schnurr, H.-P., Studer, R., and Hughes, J. (2000). On-To-Knowledge: Ontology-based tools for knowledge management. In *Proceedings of the eBusiness and eWork*, pages 18–20.

Fernández-López, M., Gómez-Pérez, A., and Juristo, N. (1997). METHONTOLOGY: from ontological art towards ontological engineering.

Gangemi, A., Catenacci, C., Ciaramita, M., and Lehmann, J. (2006). Modelling ontology evaluation and validation. In *European Semantic Web Conference*, pages 140–154. Springer.

Gennari, J. H., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubézy, M., Eriksson, H., Noy, N. F., and Tu, S. W. (2003). The evolution of protégé: an environment for knowledge-based systems development. *International Journal of Human-computer studies*, 58(1):89–123.

Gómez-Pérez, A. (2004). Ontology evaluation. In *Handbook on ontologies*, pages 251–273. Springer.

Greenspan, S., Mylopoulos, J., and Borgida, A. (1994). On formal requirements modeling languages: RML revisited. In *Proceedings of the 16th international conference on Software engineering*, pages 135–147. IEEE Computer Society Press.

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, 43(5):907–928.

Happel, H.-J. and Seedorf, S. (2006). Applications of ontologies in software engineering. In *Proc. of Workshop on Sematic Web Enabled Software Engineering"(SWESE) on the ISWC*, pages 5–9. Citeseer.

Hlomani, H. and Stacey, D. (2014). Approaches, methods, metrics, measures, and subjectivity in ontology evaluation: A survey. *Semantic Web Journal*, pages 1–5.

Lin, J., Fox, M. S., and Bilgic, T. (1996). A requirement ontology for engineering design. *Concurrent Engineering*, 4(3):279–291.

Mayank, V., Kositsyna, N., and Austin, M. (2004). Requirements engineering and the semantic web, part II. representaion, management, and validation of requirements and system-level architectures.

Moroi, T., Yoshiura, N., and Suzuki, S. (2012). Conversion of software specifications in natural languages into ontologies for reasoning. In *8th International Workshop on Semantic Web Enabled Software Engineering (SWESE'2012)*.

Motik, B., Patel-Schneider, P., and Parsia, B. (2015). OWL 2 web ontology language: Structural specification and functional-style syntax, 2nd edn.(2012).

Mylopoulos, J., Borgida, A., Jarke, M., and Koubarakis, M. (1990). Telos: Representing knowledge about information systems. *ACM Transactions on Information Systems (TOIS)*, 8(4):325–362.

Noy, N. F. and McGuinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology. Technical report, Stanford Knowledge Systems Laboratory. KSL-01-05.

Nuseibeh, B. and Easterbrook, S. (2000). Requirements engineering: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering*, pages 35–46. ACM.

Presutti, V., Daga, E., Gangemi, A., and Blomqvist, E. (2009). extreme design with content ontology design patterns. In *Proceedings of the 2009 International Conference on Ontology Patterns-Volume 516*, pages 83–97. CEUR-WS. org.

RTCA (1992). *Software Considerations in Airborne Systems and Equipment Certification*.

Sauro, J. (2011). *A practical guide to the system usability scale: Background, benchmarks & best practices*. Measuring Usability LLC.

Shearer, R., Motik, B., and Horrocks, I. (2008). HermiT: A highly-efficient owl reasoner. In *OWLED*, volume 432, page 91.

Siegemund, K., Thomas, E. J., Zhao, Y., Pan, J., and Assmann, U. (2011). Towards ontology-driven requirements engineering. In *Workshop semantic web enabled software engineering at 10th international semantic web conference (ISWC), Bonn*.

Studer, R., Benjamins, V. R., and Fensel, D. (1998). Knowledge engineering: principles and methods. *Data & knowledge engineering*, 25(1):161–197.

Tan, H. and Lambrix, P. (2009). Selecting an ontology for biomedical text mining. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, pages 55–62. Association for Computational Linguistics.

Tullis, T. S. and Stetson, J. N. (2004). A comparison of questionnaires for assessing website usability. In *Usability Professional Association Conference*, pages 1–12. Citeseer.

Uschold, M. and Gruninger, M. (1996). Ontologies: Principles, methods and applications. *The knowledge engineering review*, 11(02):93–136.

Vrandečić, D. (2009). Ontology evaluation. In *Handbook on Ontologies*, pages 293–313. Springer.