Ontology-based Requirements Specification Process

Hatime BENCHARQUI[1,*], *Saida* HAIDRAR[2], *and Adil* ANWAR[3]
[1]Mohammed V University in Rabat. EMI, Siweb Team, Morocco
[2]Mohammed V University in Rabat. EMI, Siweb Team, Morocco
[3]Mohammed V University in Rabat. EMI, Siweb Team, Morocco

Abstract. Requirements specification activity in a human-readable and formal representation is a crucial task while developing a complex system. If the meaning of requirements description and their fragments can be handled, Stakeholders can produce more effective requirement specifications of a higher quality. To that end, an ontology is developed to control capturing requirements in a formal way. The proposed ontology is built using Protege tool. This paper focuses on developing a generic ontology that can be used to define different concepts of requirements description. The suggested ontology can be used in many ways in analyzing requirements specification in any requirement engineering process. It would help system engineers to design requirement models susceptible to detect incomplete and conflicted requirements and a variety of other problems. It could also ensure consistency between many requirements specifications and increase communication between involved protagonists due to the use of the same terminology during all system specification process.

1. Introduction

An important factor for system success is the preliminary development of high-quality requirements formalization. This is a critical task given that each system domain uses complicated knowledge belonging to different abstraction levels. This can generate inconsistent, ambiguous, and error-prone requirements specifications.

In our previous works [1], we have developed a textual language, called ReqDL, devoted to formalizing requirements using a specific and common syntax. It proposes a structural description of the requirement's functionality using different operators. However, building a consistent requirement specification remains very challenging. In fact, the ReqDL syntax might be insufficient for tackling knowledge complexity related to the system domain, and this can produce ambiguous and inconsistent requirements. Moreover, ReqDL syntax uses the same structure to specify requirements even though they belong to different abstraction levels. This makes it difficult to detect conflicts and errors between requirements.

In order to handle this knowledge complexity, research has been recently interested in the use of ontology in requirements engineering areas [2, 3, 4]. In general, Ontology is a formal explicit description of concepts and properties in a specific domain, which aims at specifying its knowledge base [5]. In the context of requirements specification, Ontology can be used to tackle the knowledge used in the requirement engineering domain to make it exploitable to computers [6].

In this paper, we are going to enhance the requirements description within ReqDL syntax in order to be able to address requirements inconsistency problems. The proposed method is based on Ontology. Our method includes the following manner: 1) we build an ontology that allows formal description of concepts and relationships relevant to the system at different levels of abstraction. 2) we use this ontology to devise an intermediate DSL called OntoReq language in order to capture consistent concepts. 3) We import these concepts in ReqDL specification in order to facilitate analyzing requirements and detect susceptible conflicts.

The remainder of this paper is structured as follows: Section 2 introduces a background about ReqDL language. Ontology-based requirement specification Methodology is described in section 3. Section 4 illustrates the usage of our methodology by specifying requirements of the car cooling system, followed by related works in section 5. Finally, we give some conclusions and perspectives of this work.

2. Background

ReqDL vocabulary enables system engineers to clearly express requirements in a simple and easily understandable way in order to reduce natural language effects when documenting requirements. ReqDL addresses all kinds of requirements and documents information about the requirements as attributes as shown in figure 1

* Corresponding author: bencharqui@gmail.com

```
Traceable Requirement TemperatureManagement
{ identifier:R0
type: General
level:StakeholderReq
source : 'RSD'
author : 'Client'
description:
subject 'Vehicle' shall 'Maintain the engine with
    optimal temperature'}
```

Fig. 1. Example of a requirement specification using

ReqDL structure in this example highlights important data about requirements such as requirements source and author, requirement abstraction level, and related model elements. A fact that can facilitate many other tasks within the system development process such as requirements traceability and verification.

Obviously, requirements specifications must vary depending on their definition level. ReqDL denotes three abstraction levels where the requirements can be defined.

- Stakeholder requirement: specifies the Stakeholder needs.
- System requirement: indicates what the system would do avoiding reference to any unique design.
- Component requirement: indicates how the specific design satisfies the system requirements.

However, ReqDL keeps the same syntax for all these requirements, without differentiating between concepts according to each level. This can lead to specifying inconsistent and conflictual requirements.

Concerning the description field, it is still expressed in an informal manner. it is necessary to improve this point by finding a way to formalize the requirement expression without cluttering ReqDL syntax.

As a solution to this problem, we propose to use an ontology-based process for describing system domain knowledge. This ontology will be transformed into a pivot DSL called OntoReq. OntoReq is used to define shared concepts (constraints, data, event…) within system development activities. Those shared concepts will be imported while specifying requirements with ReqDL language.

3. Ontology-based Requirement Specification Methodology

3.1. Architecture Overview

Ontology-based specification requirement architecture offers the requirement engineers' team a formal method to elaborate a consistent requirements model as shown in Fig. 2. It represents how a given ontology is used to capture high-quality requirements.

- **Requirement description ontology:** Is used to describe system domain knowledge at different levels of abstraction and improve communication between different actors due to the use of common concepts. We have tried to define concepts related to a specific domain that can be reused and shared through requirements engineers' view points.
- **OntoReq:** Is a structured representation of the ontology in modelware space which will be used to collect common domain knowledge separately from requirements attributes. In fact, a requirement description references different system's artefacts such as constraints, Actors, objects, Events... OntoReq separates those elements from requirement description syntax, it acts as a shared dictionary.
- **ReqDL:** To provide additional controls and improve requirements' expressivity during the capture step. We have enhanced ReqDL syntax to import OntoReq elements. This helps avoiding possible errors and ensuring requirement consistency related to the writing of requirement attributes.

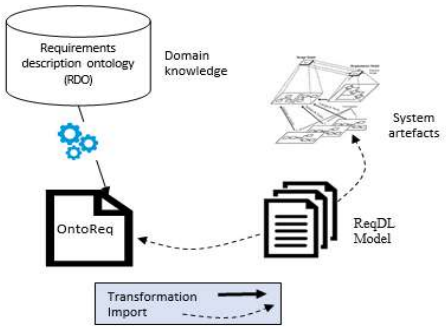

Fig. 2. ontology-based specification requirement

3.2. Requirement Ontology Design Process

Developing ontologies is guided or depended by the application domain. Therefore, there are different approaches for modeling an ontology. In [7], the authors classified these methods into two categories. The first category is based on the experience-based method, while the second category is Methontology which focuses on a set of activities and guidelines to build an ontology. The ontology that is designed here, is based on Methontology, and concerned the domain of requirement engineering, particularly, requirement description properties.

In the next subsection, we describe the main process steps:

- **Domain and Focus**

The target ontology will cover the shared knowledge that relies on Requirement Engineering, it will define concepts that are of interest within amenable requirement description. This will allow different actors to use in unambiguous ways the semantics of concepts and improve communication with users.

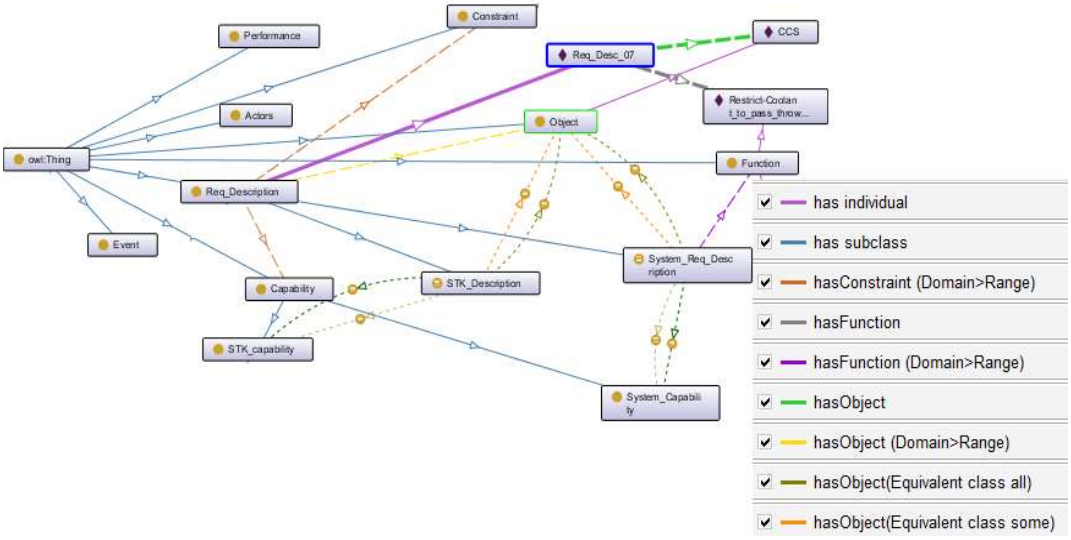

Fig. 3. Except of relations between domain concepts

- **Formulation of competence questions**

Competence questions give a general scope of the ontology and play an important role in the validation process[8]. Table 1 shows an excerpt of competency questions with answers used in the requirement description context.

Table 1. Competency questions and answers

N°	Competency Questions (CQs)	answers
CQ1	Which properties of the requirement descriptions should be considered for modeling?	Actors, function, constraint… (Table 2)
CQ3	What types of requirement description represents problem domain	STK_description (Stakeholder requirement)
CQ4	What types of requirement description represents solution domain	System description
CQ5	What types of description represents sub system Requirements	Component descrption

- **Ontology Domain description**

The proposed ontology separates the knowledge that lies in the problem domain from that in the solution domain. The concepts used can vary based on the level of requirements getting expressed. For example, capabilities and constraints needed by the stakeholders (problem domain) and functions, Event... that must be considered to specify the system requirements (solution domain). Some of these concepts are defined in Table 2. Indeed, any requirement must be defined with instances of those concepts called individuals. The object properties between two instances must be defined for each class. The potential interactions between instances of classes must also be determined.

Table 2. Part of concepts used in RDO

Concepts	Description
Capability	Identify the capability the be provided by the system
Event	A concept that allows collecting the external events of the system
Function	A concept used at the level of the requirements derived from the stakeholder requirements to express the functionalities expected by the system
Object	The object is the same as the one to which the action is directed.
Constraint	Constraint on the function or action Offered by the system
Actors	Allows capturing the elements that have a role in the system.

- **Ontology Implementation**

At this step we develop the class hierarchy of the ontology which starts with the definition of the most general concepts in the domain and then, specializes the subsequent ones [9]. Protege tool is used to implement the ontology with ontology web language (owl)[a]. Fig.3 shows an excerpt hierarchy of the main concepts. Different types of links are presented to distinguish relationships between defined concepts such as class, domain->range, individuals, object property ...

4. Illustrative Example

In this section, we illustrate our method by an example of a car cooling system (CCS). This example demonstrates the importance of using the developed ontology in the requirement specification process, and

[a] http://www.w3.org/TR/owl-guide/

thus by writing a consistent requirements model including different stakeholders and viewpoints.

The CCS consists of absorbing the engine's excess heat and avoiding possible deformation of the engine components. R07 presents an example of system's requirement:

"*R07: The CCS should restrict the coolant to pass through the radiator*"

Hereafter, we will explain how to formalize this requirement using the proposed ontology and how to derive a list of specifications with OntoReq and linking them to ReqDL language to constitute a consistent requirement.

4.1. Ontology Concepts Instances

Requirement elements are represented as a set of instances of the ontology's concepts depending on the level they belong to. Fig.4 shows how the requirement(R07) concepts are transformed into OWL instances. The main parts of this requirement are *Req_Description*, Object and function which correspond respectively to the individual values "*Req_Desc07*", "*CCS*" and "*Restrict-Coolant_to_pass_throw_radiator*". On line 7 and 8, the object's properties *hasFunction* and *hasObject* are used to associate their corresponding classes function and *Object*.

```
1  <owl:NamedIndividual
2    <rdf:about="CCS">
3    <rdf:type rdf:resource="Object"/>
4  </owl:NamedIndividual>        <!-- Req_Desc_07 -->
5  <owl:NamedIndividual rdf:about="Req_Desc_07">
6      <rdf:type rdf:resource="Req_Description"/>
7      <hasFunction rdf:resource="Restrict-Coolant_to_pass_throw_radiator"/>
8      <hasObject rdf:resource="CCS"/>
9  </owl:NamedIndividual>
10    <!-- Restrict-Coolant_to_pass_throw_radiator -->
11    <owl:NamedIndividual rdf:about="Restrict-Coolant_to_pass_throw_radiator">
12    <rdf:type rdf:resource="Function"/>
13 </owl:NamedIndividual>
```

Fig. 4. Requirement concepts instances model with OWL

4.2. OntoReq concepts specification

We instantiate the ontology to create a model of domain concepts instances. This model will be transformed into OntoReq model. This later model shows overall concepts such as constraints, Events, Actors of CCS Fig.5. For example, optimal temperature constraint is specified with unambiguous manner and set as 90 c°.

```
*CCS.onto
constraint optimal_Temp value 90 unit "c"
Event difficult_driving_climate
Event Raining_climate
Object collant value 20 unit "L"
Parameters Engine_Temperature
Actor CCS
Actor Supplier
Action restrict_coolant_to_pass_throw_Radiator
```

Fig. 5. Excerpt of Car cooling system Domain Concepts

4.3. Requirements specification with ReqDL

At this step, requirement specification activity is performed by means of ReqDL editor. Indeed, the requirement description is guided by OntoReq expressions already captured as separate models. This integration leads to constitute a requirements model with unambiguous or missing concepts. Furthermore, this can help to detect several types of requirements inconsistencies. Fig.6 shows an example of error while creating an Actor concept that does not exist in OntoReq repository. ReqDL editor can then offer instant feedback to the user as soon as the error is detected.

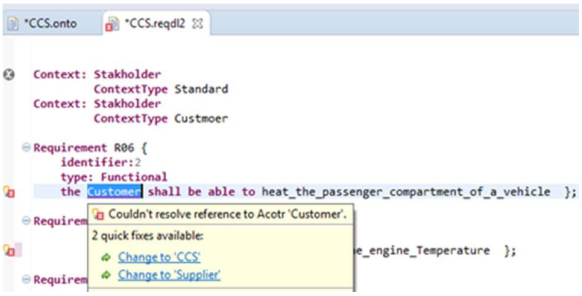

Fig. 6. Example of ReqDL editor with error detection and resolution

5. Related Works

This section discusses the studies that use ontologies in the Requirement engineering Context. Indeed, ontologies are one of the efficient methods in searching for inconsistency between requirements. This is ensured by Integrating formal concepts in writing text requirements. It is very important to automatically formalize the requirement description using a specific ontology. This ontology-based method is novel and hasn't been yet explored. In [10], the author provides a method to deal with conflicting requirements in the context of paradigm of conflict-oriented-requirements-Analysis (CORA). The presented requirement ontology describes the concepts of the requirements and their interactions. However, this approach is only concerned with stakeholder requirements and does not take into account other levels of abstraction.

A work similar to ours is [11], The authors proposed an ontology named OntRep which allows an automatic analysis. The main idea in OntRep ontology is to formulate the terminology of requirement as a dictionary. The OntRep analysis is based on two steps. The first step consists of collecting data, this is possible because requirement is specified with a specific structure. The second step of OntRep is intended to generate a conflicting requirement report. This work is still limited to the requirements of the software engineering area while our goal is to address requirements related to the system engineering field.

Other tools designate an automated process for revealing inconsistencies. For example, in [12], the authors use natural language text to specify requirements, xml is used to mark requirement elements.

The analyzer tool focuses primarily on <object>, <subject> and <relationship> to spot inconsistent requirements.

6. Conclusion

In this paper, we presented an ontology-based method dedicated to enhance requirement's specification process in terms of detecting inconsistencies and conflicts within requirements specifications. Our solution allows the use of Requirements Description Ontology concepts while specifying requirements models. Indeed, we devised a domain-specific language for capturing our Ontology concepts. ReqDL is then extended in order to be able to import these concepts. Finally, we prove the usefulness of this method by applying it to specify the requirements of the car cooling system.

Several perspectives on our work are under consideration. We are working on criteria for evaluating our proposed ontology using different case studies. We intend to generate a trace model of the conflicted requirements. So that we can effectively propose a resolution process for conflicts resolution based on the trace models.

1. S.Haidrar, H.Bencharqui, A.Anwar, J.M.Bruel & O.Roudies. REQDL: A requirements description language to support requirements traces generation. *In Modre workshop, 2017 IEEE 25th International Requirements Engineering Conference* (2017)
2. H. Alrumaih, A.Mirza, & H.Alsalamah. Domain ontology for requirements classification in *requirements engineering context. IEEE Access*, 8 (2020)
3. M.S. Murtazina, & T.V. Avdeenko. The detection of conflicts in the requirements specification based on an ontological model and a production rule system. *In CEUR Workshop Proceedings* (Vol. **2416**, pp. 63-73) (2019)
4. A. Crapo, A.Moitra, C.McMillan. & D.Russell. Requirements capture and analysis in ASSERT (TM). In *IEEE 25th Int. Requirements Engineering Conference* (2017)
5. T.E. El-Diraby, (2013). Domain ontology for construction knowledge. *Journal of Construction Engineering and Management,* 139(7), 768-784.
6. Ontologies with Python: Programming OWL 2.0 Ontologies with Python and Owlready2
7. G. Brusa, M. L. Caliusco, and O. Chiotti, ``A process for building a domain ontology: An experience in developing a government budgetary ontology*,"in Proc. 2nd Australas.Workshop Adv. Ontologies, Hobart, TAS,* Australia, vol. **72**, Dec, pp. 715 (2006)
8. M. Uschold and M. Gruninger, ``Ontologies: Principles, methods and applications," *Knowl. Eng. Rev.,* vol. **11**, no. 2, pp. 93136, (1996)
9. M.Uschold, & M,Gruninger, Ontologies: Principles, methods and applications. *technical report-university of edinburgh artificial intelligence applications institute aiai tr* (1996)
10. W.N. Robinson, S.Volkov, Requirement Conflict Restructuring *GSU CIS Working* (1999)
11. M. Heindl, T. Moser, D. Winkler and S. Biffl. Automating the Detection of Complex Semantic Conflicts between Software Requirements: An empirical study on requirementsconflict analysis with semantic technology Proc. *Int. Conf. on Software Engineering and Knowledge Engineering* (2011)
12. A. Sardinha, R.Chitchyan, J.Araújo, A.Moreira and A. Rashid. Conflict identification with EA-Analyzer *In Aspect-oriented requirements engineering* (pp. 209-224). Springer, Berlin, Heidelberg. (2013)