Proposed Ontology for Requirements Reuse

Sugandh Wafai[1], Zohaib Jan[2]

[1, 2], Shaheed Zulfikar Ali Bhutto Institute of Science and Technology (SZABIST) Karachi Pakistan

[1]sugandhwafai@gmail.com

[2]zohaib.jan@szabist.edu.pk

Abstract: **Requirements engineering is a crucial phase in system development. If the requirements are correct, the entire project will succeed but the failure of requirements will have negative impact on the project. In this research, ontology for requirement reuse is proposed either across different projects or within a single project. The requirements once elicited, designed, verified and validated, produce sets of artefacts that are used to build the system. These requirements can be useful if stored in a repository and searched for similar criteria in future scenario thus, saving time and effort. This paper proposes ontology for the reuse process of the requirements.**

***Keywords*— Requirements Engineering, Requirements Reuse, Ontology, Requirements Reuse Ontology.**

I. Introduction

Ontology is used to define the "entity" as it exists in the environment. Before it was applied to computer science, it was widely used in philosophy to represent the idea of "being" or more specifically it deals with the "possibilities" and "conditions" of being.

On the other hand, the computer scientists take ontology simply as defining the entity, its various attributes and the relationships it has with other entities.

The term ontology was first applied to the field of artificial intelligence in 1970's to represent the varying concepts of particular areas of knowledge as needed. [1]

The main reasons for using ontologies in software engineering are (i) they facilitate the semantic interoperability and (ii) they facilitate the machine reasoning.

The benefits of using ontologies for defining the process of requirements engineering are that they prevent the occurrence of common problems like; ambiguous requirements, insufficient specifications, incomplete and dynamic requirements. Using the ontologies for defining the requirements makes easier to keep track of what is happening to the requirements. [2] Ontologies are used to collect or elicit the requirements [3]. They are used for validating requirements [4] and for the software reuse [5].

Requirements Engineering (RE) is a phase of formal software engineering process that is concerned with the elicitation, analysis, design and verification and validation of the requirements. A requirement is simply described as a "desirable" feature of the intended system. It can be any functionality in the system or simply some quality feature. Requirements engineering process is important in the aspect that if the requirements are clear and well understood, the system is going to be complete and satisfactory product for all the stakeholders. But if the requirements are incomplete, ambiguous, contradictory, or dynamic, then the success of the system will be questionable.

The requirements reuse is the process that arises after the requirements are fully completed, validated and verified and transformed into a design. There are various methods for requirements reuse. According to 'methods and tools' an online software development magazine [6], the requirements are reused by the person creating the requirements without the context of the requirements and that solely depends on his experience. The sharing of the requirements is included in this context.

Requirements are often copied for the similar projects without keeping track of the origin of the requirement and where it came from.

Then, there is a reuse in which required requirements are kept with the record of their origin. It is called requirements reuse with heritage.

Requirements are sometimes used formally with all the information and the change management record. This is reuse with annotations.

When the requirements are reused for the new project with changes, then it is the reuse with the change notification.

That is keeping track of the original requirements and the changes it went through before being stored in the repository and after its retrieval whatever changes it went through are also the part of the requirement reuse. This is the most sophisticated form of the requirement reuse and said to be requirements reuse with the change notification and annotations.The benefits of reusing requirements are, it saves time, effort and cost.

This paper focuses on creating an ontology that caters to requirements reuse by conceptualizing the process that should be used for formal requirements reuse. The rest of the paper is organized in the following way; section 2 deals with the related work done in the field, section 3 deals with the requirement lifecycle, section 4 is the proposed ontology for requirements reuse. Section 5 is the conclusion and future work.

II. Literature Review

The existing literature has ample examples of ontologies used for requirements engineering processes. There exist ontologies for every individual process in requirements engineering cycle as well as for entire process. There are domain specific and application specific ontologies.

Lee *et al* [3] in their paper discuss the short comings of many different ways of eliciting requirements; they can be incomplete, incoherent, contradictory and redundant. The proposed framework "Ontology-based Active Requirements Engineering framework" adopts a mixed-initiative approach to elicit, represent and analyse the diversity of factors associated with software-intensive systems. The framework integrates various re-modelling techniques with complementary semantics in unifying ontological process. They conclude the paper by discussing the case study of The DITSCAP Automation where this framework was applied.

Wolter *et al* [5] in their paper discuss the idea of mapping MOF-based software representations and description logic-based mechanisms for facilitating software reuse. They combine all software representations (requirements specifications, design models, code etc.) resulting from one project in one "software case" and store it in a repository for later retrieval. They map the requirements specifications to an ontology using the WordNet [6] taxonomy, which makes the implicit relations between different levels of requirements explicit. Doing so also reduces the distance between the initial definition of requirements and the one already stored in repository. They conclude their paper by discussing the various successful experiments using the above discussed approach to create successful requirements models.

Farfeleder *et al* [7] in their paper use the "boilerplate" method of requirements elicitation and transform it into an ontology to make the boilerplate more domain oriented. They presented the prototype implementation of a semantic guidance system used to assist requirements engineers with capturing requirements using a semi-formal representation. Their proposed semantic guidance system uses concepts, relations and axioms of domain ontology to provide a list of suggestions the requirements engineers can use to define requirements. They concluded the paper by presenting their results that were 85% accurate.

Siegemund *et al* [8] in their paper present a meta model for ontology-driven and goal-oriented requirements engineering. They argued that to avoid the problems caused by incomplete, inconsistent, ambiguous and faulty requirements ontology tasks can be used for consistency checking and rule-driven completeness test to measure the validity and coverage of the evolving requirements model. They proposed ODRE (Ontology Driven Requirements Engineering) model which was used for checking completeness and consistency of the requirements. ODRE was evaluated within the MOST Projects and found the results satisfactory to carry on further research. In the section of future works, it was proposed that working on the traceability and creating ontology for non-functional requirements can be carried out.

Happel and Seedorf [9] discussed the application of ontologies in software engineering. They discussed the application of ontology to requirements engineering phase of software engineering to make it more formal and convenient.

Additionally, Hesse [1] discussed the benefit of applying ontologies in the software engineering process as a whole, making it more reusable, component driven and structured. The argument presented is that using the ontology can help in understanding interoperability and partial reuse of the requirements. It should be as much detailed-oriented as possible but should not provide the implementation details.

Kossman *et al* [10] in their paper presented OntoREM, an Ontology-driven Requirements Engineering Methodology, that aims to improve the quality of requirements while reducing the time and cost needed to develop, maintain and reuse requirements. They attempted to show the importance and efficiency of OntoREM by applying it to the aerospace industry and suggest that this methodology should be applied to the larger industry to see its true potential. They also pointed out that by adopting OntoREM, domain ontology will be developed which will serve as the formal repository of validated domain knowledge which allows the reuse of the domain knowledge which in turn could be used for future projects with less effort than the first project.

In a report by Jureta *et al* [11], they endeavoured to rediscover the meanings and definitions of various requirements engineering terms to make a standard for applying ontological practices to the process of requirements engineering. They called it "CORE" ontology for requirements engineering. Their emphasis was that at times one term is used for many objects and sometimes one object is called by many terms, this creates ambiguity and contradiction in requirements. To avoid such situation, it is necessary to agree on one definition of each term and how it will be used in the given context.

Jureta [12] found problems with Zave and Jackson's core ontology for defining the "requirement problem". They discussed in their previous report [9] that because of the many definitions and meanings of one term, it becomes difficult to understand the intention and requirement of a user. Furthermore, the ways of communication also differ, thus creating problems with the fulfilment of "non-functional" requirements or "nice-to-have" features. To eradicate these issues, the authors of this paper proposes an ontology that not only takes into consideration requirements, specifications and domain knowledge but also takes into consideration the plan, quality, soft goals and justified approximation. Therefore, making the understanding and recording of "non-functional" requirements easy.

In 2009, Jureta [13] did further research on CORE ontology for requirements, defining the various acts of reasoning and how they can be interpreted by requirements engineers to modulate the requirements. Because of being people intensive task, it is not always easy to understand and interpret the requirements. This part of Jureta's CORE ontology emphasizes on the acts of communication and the mental state of the speaker about what is being communicated. It is important as it will help the engineer to understand what does speaker literally means as thinking of "always true," "already there," "desirable," "critical," and "crucial."

III. REQUIREMENTS LIFECYCLE

According to BABOK [14-15], the requirements go through these phases until they are turned into a product.

A requirement starts with being “stated” after it has been documented as a result of an elicitation activity but it is

The requirement is then confirmed by the requirements engineer or business analyst that by further interviewing or and confirms to the actual desired or intentions of the stakeholder.

After the confirmation, requirement is communicated that is during the various formal and informal sessions, the requirement is presented to various stakeholders to clarify their understanding of the requirements. This step is necessary to avoid the conflict in future requirements and the stakeholders.

After they are communicated, it is important to keep the traceability clear and maintained because it helps to trace the requirement back to its origin and forward to its implementation. The tracing also helps in covering the relationships between the requirements.

After the initial phases, the requirements are approved. It is done by signing off from the authorized stakeholders.

After being approved by the stakeholders, the requirements are prioritized. The priority of the requirement depends on the value the requirement delivers to the business, the risk, difficulty and urgency of the task.

After the prioritization, the requirements are analysed. It is important to note that a requirement can be analysed before the prioritization as well. The requirements are modelled for the various tasks and the applicability etc.

The requirements are formally verified, prioritized and analysed. The requirements are verified to make sure that requirement specification is internally consistent and work accordingly. Validating if requirements come after it has been verified and it is done to ensure that the requirements fulfils the criteria it was set for, or in other words, it is doing what it was supposed to do. The validated requirements achieve their goals. After the validation and verification of the requirements, they are implemented. The implementation comes with its own set of rules. The implementation of the requirement depends on the release cycles, available resources and on any other constraints.

Another phase that is part of the “requirements eleven” (figure 1) is maintaining requirements for reuse. It is completely decoupled from the rest of the requirement lifecycle. The maintenance phase can be applied to the requirements in any other phase and it does not affect the implementation of the requirement (the system to-be) but maintaining requirements for re-use can serve the organization in the long run.

important to commemorate that it is still unconfirmed, that is, it is still not validated that it actually matches the stakeholder’s understanding of the problem.

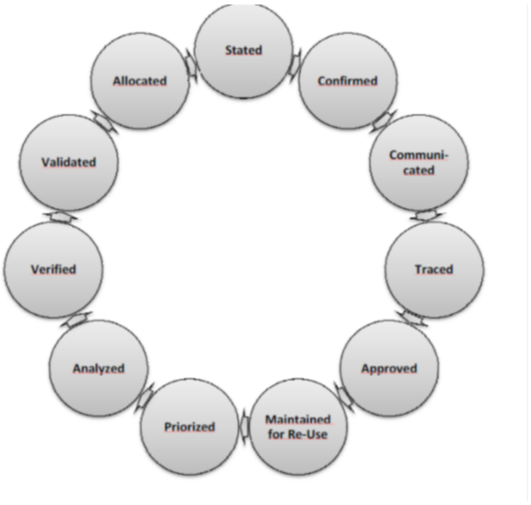

Fig. (1). Requirements Eleven by BABOK [14]

1V. REQUIREMENTS REUSE PROCESS

There is no formal process for requirements reuse; the OPF [16] states the three steps for requirements reuse.

1. Standardize the requirements.
2. Put in a repository.
3. Reuse.

Standardization of the requirement is not easily achieved, as everyone have their own standard and definition, as Jureta [13] stated. Documenting the requirements is just one way of standardizing them. An example can be used as use-cases.

A use case can be simply described as a discrete and standalone activity that a user performs to achieve some task/goal.

The requirement repository can be an ordinary database table to a complex data warehouse where the documented artifacts and the external links can be stored and retrieved for some matching criteria as Kathrina [5] performed.

Reuse is re-employing already documented, tested, validated and verified requirements in some other project or phase of the same project. The reuse requires analysis and re-analysis to make sure that retrieved data is indeed relevant to the search criteria. The requirements can be reused within the project (release cycles/ products) or across the projects (similar but different project)

The traceability is usually important for requirement reuse because matching the use cases or subject-verb-object sentences will be futile if there is nowhere to go from there. Requirement traceability is defined as *“ability to describe and follow the life of a requirement, in both forward and backward direction, ideally through the whole system lifecycle.”* [17]

The traceability is achieved by linking the related information objects for example, requirements with:

- System Objectives that it needs to achieve.
- Change proposals and affected requirements.
- Decisions, rationales and assumptions made during the elicitation or analysis process.
- Test cases validating the requirements; and,
- System components and resources needed to implement those requirements.

V. Requirement Maintenance for Reuse

A requirement goes through various phases before it is finalized for implementation. BABOK [13] suggest that maintaining the requirement can make the impact analysis of the new or proposed changes to the business, reduce analysis time and effort, facilitate in maintenance of previously implemented solutions and support other activities, including training, corporate governance and standard compliance.

Hull *et al* in their book [18] wrote that management issues for requirements development arises from one of the following areas:

- Planning
- Monitoring Progress
- Controlling changes

Planning is associated with the earlier activities of requirements lifecycle i.e. identification of the stakeholders, allocation of time and resources, elicitation of requirements, placing the requirements within the given structure, identification of constraints etc. Care should be taken to assess any planning constraints to ensure that they are feasible and sensible. Monitoring starts once the plan has been put into action. The monitoring plan should address the issues of structure of requirements specification, required attributes for requirements and the review process. Once the monitoring structure is ready, identification of any missing points or "holes" would be easy and it will also be very feasible to monitor the progress of the plan. Changes occur when the stakeholders realize shortcomings in the original requirement or face feasibility or structure issue. Managing change is a vital activity in requirements development. Whatever stage a project is at, the following steps are required in a change control process.

- Recording the suggested change
- Identification of the impact of change on other requirements
- Decision on the acceptability of change, and;
- Decision of the implementation of change.

Whatever the change may be, it is important to maintain the traceability of the requirement.

VI. An Ontology For Requirements Reuse Process

In this section, ontology for the reuse of the requirements gathered by the organization to complete projects is introduced. From the literature review, it has been elicited that the requirements reuse is a very common practice and is carried out by every major organization in every capacity to save time, effort, cost and to build efficient, accurate and more realistic requirements. But most of these requirements are domain specific, i.e. they only apply to one specific area of study or are application/project specific. For the reuse of the requirements, it is necessary to understand what the requirement needs to do, its "goal" [11] and then the metadata about that requirement, i.e. who proposed the requirement, who approved it, what changes it went through, what artefacts were produced from it. All of this "goal" and metadata needs to be placed in repository. The artefact repository can be a database or data warehouse but it should facilitate the retrieval of information in various ways, lexical and graphical. The matching criteria for requirements reuse can differ from one project to another but it is agreed by all authors that it should be able to match the nearest possible requirements with the set criteria according to the "goal" or "project type", "activities the requirement will trigger" etc. Figure 2.

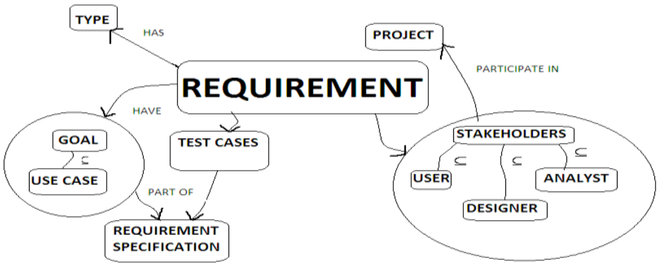

Fig. (2). Requirements Attributes

The following ontology is an independent and generic presentation of the requirement as an entity, the various attributes that it will require to become "searchable" and the relationships of requirement with other domains.

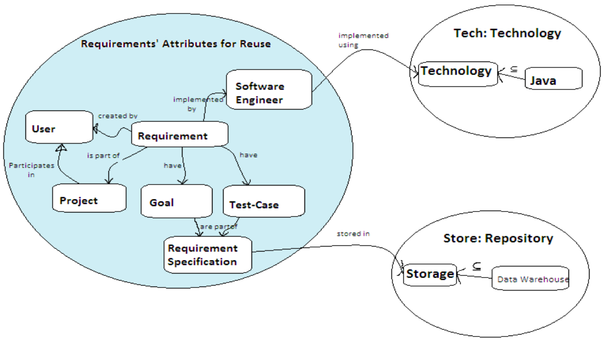

Fig. (3). An Ontology for Requirement Reuse Process

The above figure 3 represents a part of the reuse process, the attributes of every requirement to standardize it and build a traceability grid to make it easier for reuse with reference.

It is important to agree upon and maintain the granularity of the information in metadata because the search criteria will depend on the information provided. The searching of the repository is dependent on the application domain and the implementation of the repository. The common consensus is on the use of XML based tags, Query languages and ontological terms. The search results are dependent on human analysis but because they are carefully selected based on the close association of the query and the result produced, it is less tedious and more methodical.

VII. Conclusion

Requirement reuse is an undeniable fact. Various people and organizations reuse requirements in one way or another. The need is to streamline the process so that it becomes easier to understand and follow. The ontology presented here is a very first attempt to stream line the process of requirements reuse. The matching of the existing requirements with the requirements of a future project has been the subject of the various researches. In this paper, the requirements reuse process has been presented using ontology so that the standardization of the requirement and the relationship with the other entities is clear.

References

[1] W. Hesse, "Ontologies in the Software Engineering Process," In *Proceedings of Workshop on Enterprise Application Integration* (*EAI)*, 2005.

[2] L. Ballejos, "The Use of Ontologies in Requirements," *Global Journal of Researches in Engineering*, vol. 10, no. 6, pp: 2-8, 2010.

[3] S. W. Lee and R. A. Gandhi, "Ontology-based active requirements engineering framework," In *Proceedings of 12th Asia-Pacific Software Engineering Conference*, 2005.

[4] L. Kof, R. Gacitua, M. Rouncefield, and P. Sawyer, "Ontology and Model Alignment as a Means for Requirements Validation," In *Proceedings of 2010 IEEE Fourth International Conference on Semantic Computing*, 2010, pp: 46–51.

[5] K. Wolter, M. Smialek, L. Hotz, S. Knab, and J. Bojarski, "Mapping MOF-based requirements representations to ontologies for software reuse," In *Proceedings of 2nd Workshop on Transforming and Weaving Ontologies and MDE*, USA, 2009.

[6] D. Akers, "Real Reuse for Requirements," 2008. Reference: Available from: http://www.methodsandtools.com/archive/archive.php?id=68. Accessed: 21-Apr-2014.

[7] S. Farfeleder, T. Moser, and A. Krall, "Ontology-driven guidance for requirements elicitation," In *Proceedings of the 8th extended semantic web conference on The semanic web: research and applications*, 2011, pp: 212–226.

[8] K. Siegemund, E. J. Thomas, Y. Zhao, J. Z. Pan and U. Assmann "Towards ontology-driven requirements engineering," In *Proceedings of the 10th International Semantic Web Conference*, 2011.

[9] H. Happel and S. Seedorf, "Applications of ontologies in software engineering," In *Proceedings of 2nd International Workshop on Sematic Web Enabled Software Engineering,* 2006.

[10] M. Kossmann and M. Odeh, "Ontology-driven requirements engineering–A case study of OntoREM in the aerospace context," In *Proceedings of INCOSE Conference,* 2010.

[11] I. Jureta, J. Mylopoulos, S. Faulkner, and P. Schobbens, "Core ontology for requirements engineering," Inf. Manage. Res. Unit, Univ. Namur, Belgium, Tech Rep, 2007.

[12] I. Jureta, J. Mylopoulos, and S. Faulkner, "Revisiting the Core Ontology and Problem in Requirements Engineering," In *Proceedings of 16th IEEE International Requirements Engineering Conference*, 2008, pp: 71–80.

[13] I. Jureta, J. Mylopoulos, and S. Faulkner, "A core ontology for requirements," *Applied Ontology*, vol. 4, pp: 169–244, 2009.

[14] *Requirements Lifecycle according to BABOK®*, masVenta Business GmbH, Germany, 2011, pp. 8–9.

[15] *A Guide to the Business Analysis Body of Knowledge ® (BABOK ® Guide)*, 2nd ed., International institute of Business Analysis, Whitby, ON, Canada, 2009.

[16] Open Process Framework (OPF). Reference Available from: http://www.opfro.org/index.html?Components/WorkUnits/Tasks/RequirementsEngineering/RequirementsManagement.html~Contents .

[17] A. Aurum and C. Wohlin, *Engineering and managing software requirements*. Springer, 2005.

[18] E. Hull, K. Jackson, and J. Dick, *Requirements Engineering*, 3rd ed. Springer, 2011.