Requirements quality in the incremental design processes: problems and perspectives.

Claude Reytérou

Airbus Group Innovations
18, Rue Marius Terce
31025 Toulouse, France
+33 (0)561 168831

claude.reyterou@airbus.com

Abstract. Writing requirements is a critical step in designing aircrafts software-intensive systems. The latest requirement management and authoring tools, using current engineering based approaches, start to efficiently support the requirements quality and consistency checks for huge projects having long changes cycles. However, these solutions become limited facing the incremental design processes where frequent changes of requirements shall be handled. In this paper we will discuss on dedicated approaches to support requirement writing and checking based on boilerplates and semantic knowledge representations, in particular ontologies. The expected contributions are firstly to improve the quality of requirement writing and secondly to advance the current knowledge in the use of semantic technologies dedicated to quality management. We present the corresponding research issues, the relevance of these approaches and the main lines of the proposed research activities as well as the originality of the selected options.

Keywords: Requirement authoring, Boilerplates, Natural Language Processing, Ontology, Incremental design processes.

1 Introduction

As for many other complex systems, designing an aircraft is not an easy task; be it the new version of an existing model or creating from the scratch. Generally, the stage following the collect of customer needs is the requirements definition, starting by elicitation and analysis. This process continues with the cascading of requirements that often is a mirror of the product breakdown structure. Then, for each system level, requirements set are used to define the system elements. The other requirements are allocated to sub-system at the lower levels and the process continuing till the development of components. The market driven incremental product development and delivery (release) is becoming increasingly commonplace in software industry [3]. Incremental product development is planned and executed with the goal of delivering an optimal subset of requirements in a certain release (version of a product that is

distributed to customers) [2]. In parallel, 'Agile' methodologies are an alternative to traditional sequential development, addressing unpredictability response through incremental and iterative work cadences, known as sprints [4]. Incremental Design Processes is a promising emerging discipline related to, but not specifically a subset of, the market driven incremental product development and agile methodologies. These processes combine both approaches, trying to merge the agility methods and the incremental process to address the specificity of designing based variants products. A system design process could be said to be "incremental design process" if he is based on the use of high quality generic requirements instantiated into specific requirements dedicated to the definition of variants or increments of a system.
Our interest is to identify whether the incremental design process could be applied for the development of multiple variants of complex products (based on product line engineering), considering the current methods for creating high quality generic requirements. But idea of using incremental design processes raises challenges that could be interesting for the community of requirements engineering researchers.

2 Problem statement and experiences in industry

Our problem lies in the way we could formulate the high quality system requirements to be added in the incremental design processes. As the sources of the requirements vary it is not surprising that requirements come in different shapes and forms, at multiple levels of abstraction, and described on varying levels of refinement [1]. Requirements are specified for many different purposes and from many different engineering activities [5]. During authoring, natural language remains a universal means of expressing requirements and studies indicates that 89% of engineers [7] shown their preference to use of natural language requirements. However ambiguity or vagueness of requirements are the two main problems arising from the over-flexibility of the natural language [28], making their interpretation challenging for any natural language processing systems to reasonably understand the subject-matter.

The requirement writer in many occasions is skewed by their own personal experiences hence semantics, vocabulary and terms differ widely from one person to another as illustrated by Dickerson [6]. Not much research has investigated whether different domains need different kinds of semantic tools displaying different kinds of semantic relations. To address the challenge of writing the requirements right, we have identified two main industrial problems.

2.1 Poor quality of requirements while authoring

From the statistic survey established by Fanmuy and Foughali [7], it was found that the most common leading defects in the natural language requirements falls under: semantic contradiction, not verifiable, not complete, ambiguous, not understandable and not precise enough. On the conclusion of the survey, it was stated that the problems still persist despite the use of several requirement engineering approach like writing 'SMART' requirements right from the very first attempt. Dedicated to assist the system engineers, this approach leads to eliminate unnecessary information in requirements, to improve readability (i.e. text length, number of punctuation marks, etc.) and reduce complexity. Another approach: the formal notation and graphical representation based on models albeit offers alternative means to natural language. Gorschek proposes a Requirement Abstraction Model [1]. However, these approaches remain not so convenient to cover wide range of concepts, to manage compliance and to address the needs creativity of system engineers. In practice, the requirements do not exhibit all the acceptable characteristics of good requirements. As an example, this bad requirement "New and modified air distribution components shall be designed to minimize noise levels" should be replaced by "New and modified air distribution components installed in cabin areas shall emit less than 30 dB(A) of acoustic noise." The consequence of these mistakes is felt during all downstream activities such as architecting, design, implementation, and testing. Organizations struggle to bring consistency to their project and 47% of unsuccessful projects fail to meet goals due to poor requirements management [20].

2.2 The problems of requirements verification

Requirement Verification is about verifying sufficiently early in the development process whether the requirements have sufficient quality (i.e. they are well-formed according to the ISO/IEC/IEEE 29148 standard) to avoid many negative impacts subsequent activities. When eventually discovered, these defects will be significantly more expensive and take more time to fix them. The following picture shows the cost to extract defects.

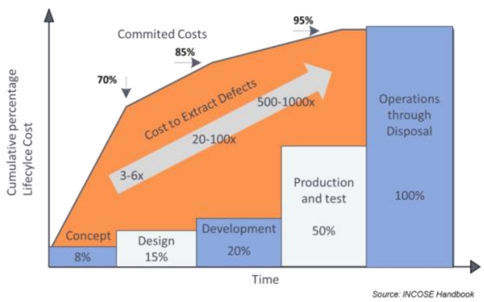

Fig. 1. Cumulative percentage of cost (source: INCOSE handbook)

Industrial projects handle up to thousands of requirements where human based verification process during peer reviews becomes extremely tedious, time consuming and expensive for the organization.

3 Proposed research activities

To address the hereunder issues, the following research tasks were identified.

3.1 Boilerplate for syntactic analysis of requirements

First of all, improvement of requirements quality is given through a better structuring of requirements sentences. This structuring goes through the definition of models allowing the creation of one or several types of requirements. To do so Mavin considers a simple and efficient set of sentence structures to improve drastically the quality of requirements [8]. Each sentence structure can be based on full sentence models as proposed by Al-Safadi [9] and in the Cesar project [10]. This study proves the interest to use sentences model based on predefined (frozen) or progressive, adaptive structures to support the authoring of requirements. This structure is generally called 'patterns' or 'boilerplate'. The concept of using boilerplates for writing statements of requirements is quite simple: choose an appropriate predefined pattern, and fill in the gaps. Each statement of requirement is then based on a boilerplate where the selected attributes have specific terms. Example of boilerplate: The <system> shall be able to <capability_verb> at a maximum rate of at least <quantity> times per <time unit>. The corresponding requirement could be: 'The light shall be able to flash at a maximum rate of at least 5 times per second.' The benefits of using boilerplates to structure sentence for the syntactic analysis of requirements are numerous. The most important ones are an aid in the articulation of sentence, a uniformity of language and a one-stop control over expression.

3.2 Ontologies for Semantic analysis of Requirements

The improvement of requirements is also possible thanks to an analysis of their meaning. Indeed, even if a requirement could be correctly written with boilerplate, this solution does ensure neither its intrinsic quality (i.e.: what is the requirement meaning?) nor its global quality (is the requirement redundant, contradictory or similar with regard to the other requirements?). The consistency quality of a requirement may rely on a semantic analysis of its sentences. To carry out this analysis we use outcomes given in the Cesar project [10], by Kof [11] and Jureta [12] which have already tackled these issues through different approaches, in particular with a domain ontology (like in some of the early works by Lin [13] Yu [14] and Cadihlac [16]). From all available definitions of ontologies, the best suitable for our purpose is given by Gruber [15]. In simple terms, ontology represents a domain of knowledge. To build and manage our ontologies, different tools exist on the market. Our aim is not to make a benchmark of these tools but to define a method to apply them. The process supporting this method of domain ontology definition is summarized in the next figure. Starting from the collection of corpus and requirements sets, the global process is based on five main sub-processes (the terminology, thesaurus, pattern definition, pattern matching and formalization) grouped together into two main phases.

The result of this process is double; a domain terms-based ontology (i.e. light ontology) and a set of structured formulation of sentences (i.e. boilerplates). Both are linked together, therefore the ontology is mapped on the boilerplates that support the analysis of quality. Indeed, only the combination of structured sentences and well known terminology ensures the definition of better quality requirements.

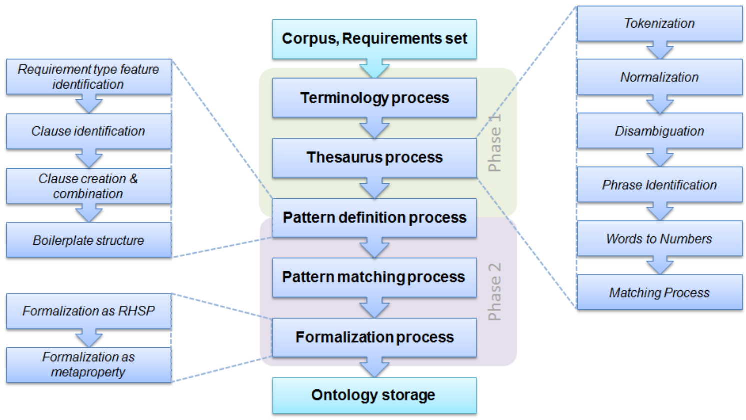

Fig. 2. Ontology definition process

4 Possible Perspectives

Future directions focus on extending our understanding of boilerplates and ontology techniques implemented for aircraft industry. Currently, we are in the process of constructive generic method to define clearly the process of ontology and boilerplate creations. So far we identified different requirement error taxonomy, semantic based requirement engineering concepts [17], formal expression language used in ontology [18], rules to construct domain ontology and issues concerning maintainability and interoperability of ontology [19]. Practical work covers the construction of thesaurus and its rules for ontology. Next activity will be to apply the process dedicated to the implementation and use of the boilerplates and ontology. However, the next problem will be the complexity of incremental design processes that requires the creation of high quality generic requirements. This drives to research issues: how ensuring requirements consistent and complete in incremental design processes? Which methods and techniques drive to requirements quality for product line processes? Rather than affirming at this stage, what shall be done in the next years down the line, we can only expect some ground breaking results thanks to new research activities.

Conclusion

The current practices and techniques of the requirement engineering are wide. Experiments, case studies, survey and action research will be evaluated by the end users in near future to determine suitability and interest of our boilerplate and ontology based method. As a result of the integration of our research methodology it is expected to create synergy and to contribute to the quality improvement of requirements. An interesting opportunity will be to carry out implementation of future methods and solutions to improve the quality of requirements for the purpose of incremental design processes.

References

1. Gorschek T, and Wohlin C: Requirements Abstraction Model. Requirements Engineering Journal, Vol. 11, No. 1, pp. 79-101, (2006)
2. Butscher SA, Laker M: Market-Driven Product Development. Marketing Management 9: pp. 48-53. (2000)
3. Gorschek T: Requirement Engineering Supporting Technical Product Management. Thesis of the Department of Systems and Software Engineering, School of Engineering; Blekinge Institute of Technology (2006).
4. Elis, G: Chapter 8 – Agile Project Management: Scrum, eXtreme Programming, and Scrumban. Project Management in Product Development, pp. 223-260 (2016).
5. Kaindl, H., Smiałek, M., Patrick Wagner, & Svetinovic, D.: Requirements Specification Language Definition. ReDSeeDS Project, Project Deliverable D (2009).
6. Dickerson, M.: Semantic dictionary and Concept Model. INCOSE insight, publication of the International Council on System Engineering, Vol 7 Issue 2. (2004)
7. Fanmuy, G., & Foughali, G.: A survey on Industrial Practices in Requirements Engineering. (2012).
8. Mavin A., Wilkinson P., Harwood A., Novak M. EARS (Easy Approach to Requirement Syntax). In proceedings of the 17th IEEE international Requirement Engineering Conference, pp. 317-322 (2009).
9. Al-Safadi L.: Natural Language Processing for Conceptual Modeling. International Journal of Digital Content Technology and its Applications Volume 3, Number 3 (2009).
10. Cesar project: Definition and exemplification of DSL and RMM. Public report D_SP2_R2.2_M2, version 1 (2010).
11. Kof L.: From Requirements Documents to System Models: Interactive Semi-Automatic Translation. Poster of IEEE international Requirement Engineering Conference (2010).
12. Jureta I., Borgida A., Ernst N., Mylopoulos J.: Techne. In proceedings of the 18th IEEE international Requirement Engineering Conference (2010).
13. Lin J., Fox, M., Bilgic, T.: A requirement ontology for engineering design. Concurrent Engineering: Research and Applications, 4, pp 279-291 (1996).
14. Yu, E.: Towards modeling and reasoning support for early-phase requirements engineering. In Proceedings of the IEEE International, Symposium on Requirements Engineering. IEEE, pp. 226-235 (2008).
15. Gruber; T.: A translation approach to portable ontology specifications. Knowledge Acquisition, pp. 199-220 (1993).
16. Cadilhac A. Aussenac-Gilles N., Benamara F.: Ontolexical resources for feature-based opinion mining: a case-study. ONTOLEX-Workshop at COLING 2010, Chinese Information Processing Society, p. 77-86 (2010).
17. Riechert, T., Lauenroth, K., & Lehmann, J.: Towards Semantic based Requirements Engineering. Requirements Engineering (2007).
18. Saeki, M.: Semantic Requirements Engineering. Requirements Engineering (2006).
19. Zong-yong, L. I., Zhi-xu, W., Ai-hui, Z., & Yong, X. The Domain Ontology and Domain Rules Based Requirements Model Checking. International Journal, pp. 89-100 (2007).
20. Bieg, D.: Requirements management, a core competency for project and programme success. PMI's Pulse of the Profession: Requirements Management — A Core Competency for Project and Programme Success (2014)