Towards scalable information modeling of requirements architectures

Krzysztof Wnuk[1], Markus Borg[1], Saïd Assar[1, 2],

[1]Department of Computer Science, Lund University, Lund Sweden,
[2] Telecom Ecole de Management, France
{Krzysztof.Wnuk, Markus.Borg}@cs.lth.se, said.assar@it-sudparis.eu

Abstract. The amount of data in large-scale software engineering contexts continues to grow and challenges efficiency of software engineering efforts. At the same time, information related to requirements plays a vital role in the success of software products and projects. To face the current challenges in software engineering information management, software companies need to reconsider the current models of information. In this paper, we present a modeling framework for requirements artifacts dedicated to a large-scale market-driven requirements engineering context. The underlying meta-model is grounded in a clear industrial need for improved flexible models for storing requirements engineering information. The presented framework is created in collaboration with industry and initially evaluated by industry practitioners from three large companies. Participants of the evaluation positively evaluated the presented modeling framework as well as pointed out directions for further research and improvements.

Keywords: Large-scale requirements engineering, requirements architectures, empirical study, requirements modeling.

1 Introduction

Requirements engineering is an important part of the software development lifecycle as it helps to identify what should be implemented in software products to make them successful. As a knowledge intense part of the software development process, requirements engineering contributes to the generation of large amounts of information that need to be managed.

The size and complexity of software engineering artifacts continues to grow as a result of increasing complexity of software intensive systems. As a result, software development companies that operate globally often have to face the challenges of storing over 10 000 requirements in the requirements database [2,4]. The amount of information to manage increases even more if we consider additional software development information such as product strategies, design documents, test case descriptions and defect reports.

In a recent study, we introduced a classification of requirements engineering contexts based on the number of requirements and the number of interdependencies between requirements as a proxy for complexity [2]. We defined Very-Large Scale Requirements Engineering (VLSRE) as a context where the number of requirements and interdependencies exceeds 10 000 and manually managing a complete set of interdependencies among small bundles of requirements is unfeasible in practice. While empirically exploring challenges in VLSRE, we discovered that one of the challenges in VLSRE is to define and properly manage structures of requirements information, also called requirements architectures [10]. Defining a model for requirements related information could help in this and other related challenges of VLSRE. The challenge lies not only in dealing with the heterogeneity of artifacts structure that need to be managed all along the software project, but also in dealing with the frequent evolution of these structures during the lifetime of the software project.

In this paper we present a general modeling framework for requirements information in VLSRE projects created in close collaboration with industry. The underlying meta-model can describe not only requirements, but also any other pieces of relevant software development information, as suggested by our industry partners. The novelty of the approach lies in its capacity to explicitly involve external sources of information and in handling the temporal aspect related to the evolution of artifacts' structures. We conducted an initial validation of our approach with 5 practitioners from 3 companies to collect feedback, opinions and improvement proposals regarding the framework. All five respondents positively evaluated the general usefulness of the approach and provided insights and suggestions for further development and improvement of the modeling framework.

This paper is structured as follows: section 2 presents background, related work, outlines an example industrial context based of one of our industrial partners and explains the need for creating the modeling framework. Section 3 presents the research design of the study. Section 4 presents the modeling framework while section 5 presents the results of the initial evaluation of the model with industry practitioners. Section 6 discusses the limitations of the model, outlines future work and concludes the paper.

2 Large-scale requirements engineering and information landscape on an empirical example

Most work in an enterprise is accompanied by some form of knowledge representation in documents [11]. Documentation is fundamental in requirements engineering, as the lack of complete specifications is a major cause of project failures [12]. However, storing too much documentation risks burdening employers by an ever-increasing amount of information. *Information overload* occurs when an individual's information processing capabilities are exceeded by the information processing requirements, i.e. the individual does not have enough time or capability to process all presented information [14]. Several studies have found that the support for decision-making is positively correlated to the amount of presented information up to a certain point, and then it declines [13, 15, 16].

In software engineering projects, large amounts of formal and informal information is continuously produced and modified [5, 6]. Thus, an important characteristic of artifacts' information in software engineering projects is its findability, defined as "the degree to which a system or environment supports navigation and retrieval" [7]. Information seeking is an increasingly costly activity among knowledge workers in general [8]. Software engineering projects are no exceptions, as identified by previous case studies in this context [5, 9].

We present an example of a VLSRE context based on a longitudinal study we have been conducting at a large company since fall 2007. Focusing on feature tracking, we observed the structure of information related to product features and the associated detailed requirements, and the evolution of this structure over time and over projects. Together with observing the evolution of the information structure, we have in fall 2007 conducted 7 in-depth interviews to understand the role of information structures and their impact on the VLSRE context. Partial results from this study were published in [3, 10]. During these 7 interviews, we have conceptualized the following picture of the information landscape while managing requirements and features in a VLSRE context, see Fig.1.

In an example of a VLSRE context, we have key customers submitting their requirements specifications to the requirements repository, and suppliers receiving specifications based on interpretations of these key customers' wishes and market trends. Special teams of experts (STEs) together with product planning, assisted by requirements analysts and business analysts, create natural language descriptions of candidate future software features. The features are later refined by STEs to a set of more detailed system requirements and merged into the current requirements architecture. As it is indicated in Fig. 1, every new specification deliverable contains partial requirements structures that should fit within requirements architecture and be merged with the requirements repository.

The current number of features in the repository exceeds 8000 and the number of attributes associated with the features exceeds 50. Thus, the amount of information to manage is substantial. Moreover, the efficiency of requirements engineering and software development efforts depend on the accuracy, understandability and cohesion, robustness, extensibility and flexibility of the information structure [3].

3 Research design

To evaluate a modeling method or technique, a large set of approaches are possible such as feature comparison, conceptual investigation or empirical evaluation [17]. This study adopts an empirical perspective and has been conducted in an action research mode. In action research studies, researchers make an attempt to solve a real-word problem. At the same time, researchers investigate the experiences and problems encountered while trying to solve the problem [18]. In our case, a need for developing a model for requirements information was stated by our industry partners during the interviews in 2007. Following that authentic need, we have conducted several unstructured brainstorming sessions and discussion meetings with our industry partners where we further discussed the need for the model and the high-level content of it. Moreover, we have studied the current information models used at the case company and identified their strong and weak points that were also discussed during the brainstorming sessions. Based on the result of these empirical investigations, we propose a framework for requirements information modeling presented in section 4. This framework exploits a traceability meta-models developed previously [20].

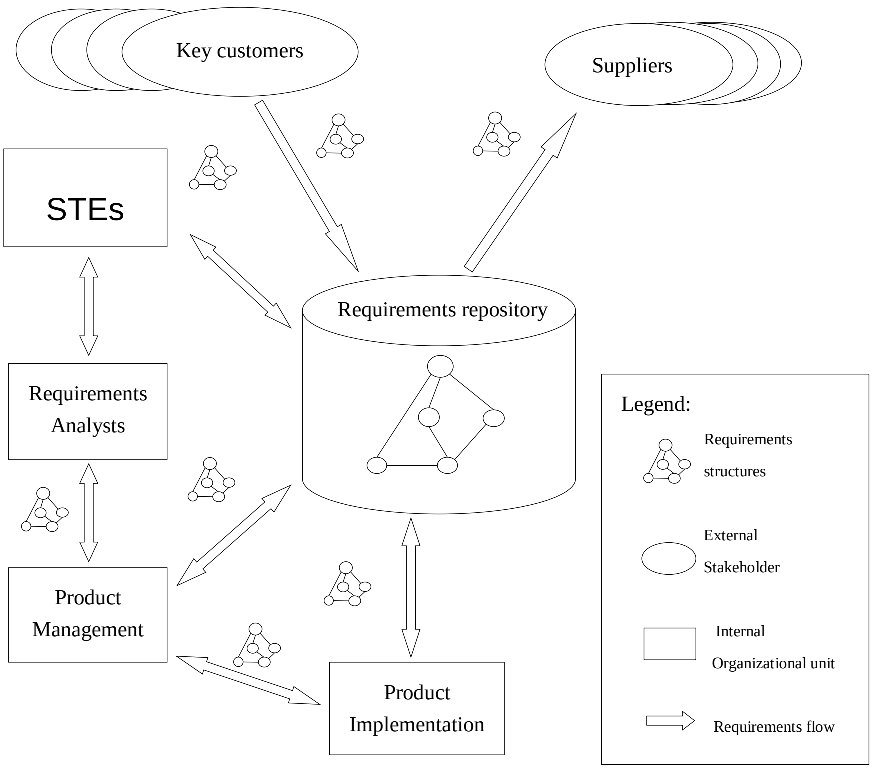

Fig. 1. An example of requirements engineering information flow in a VLSRE context.

We conducted 5 interviews at 3 companies to perform the initial validation of the model. The interviews were semi-structured which means that there was a possibility to discuss aspects not covered by the interview instrument [1]. Each interview took up to 60 minutes and was attended by one researcher; who moderated the discussion and took extensive notes; and one interviewee. At the beginning of each interview, the research goals and objectives were outlined to the interviewees. Next, we discussed the information model. Further, specific questions regarding the general usefulness of the modeling framework followed by specific questions regarding the elements of the underlying meta-model were asked. Finally, we collected the interviewees' opinions regarding the limitations of the model and suggestions for improvements of the modeling framework.

4 The iMORE framework

The core of the iMORE (*information Modeling in Requirements Engineering*) framework is the distinction between the external information structures and internal information structures, outlined in Fig. 2 by a dashed line. The importance of including external information structures was stressed several times by our industrial practitioners during the development of the modeling framework. This need for external information structures is caused by several sources of requirements and other information types that directly interact with the company, including competitors, suppliers, open source components and other partners. For all abstraction levels of the model, there is a need to be able to access external information while managing companies' internal information. For example, while looking at the source code, developers could check similar or associated open source solutions.

The structures of information are divided into three main blocks: the upstream, the requirements and the downstream blocks. In the 'upstream block' all 'high-level' information is stored, including the goals, strategies and business needs. In the 'requirements block' all requirements associated information is stored, including functional requirements, quality requirements, constraints, legal requirements and regulations. In the 'downstream' block the information related to the source code, is placed, including bug reports, code documentation, and the source code itself.

The last main element in the iMORE framework is handling temporal aspect of the information structure, depicted in the vertical arrow in Fig 2. The temporal aspects include capturing the evolution of the data models in terms of the evolution of the artifacts and their associated structures. To deal with this issue, the underlying meta-model defines 'Evolution' type of links between two artifacts. Using this category of links, users can handle the evolution over time of artifacts and their structure.

The information structure in each of the blocks is defined according to a simple traceability meta-model derived from related works [19] and previous research [20]. In this meta-model (Fig. 3), the structure of an element to be stored in the repository and to be traced in the software project is constructed using two generic concepts: artifact and attribute. An attribute can be an atomic element of information (e.g. owner, release date, version number, URL) or any complex structure (e.g. list of modification dates). The set of attributes is not only limited to a particular block of information but may also cover several blocks or even the entire information structure creating a set of 'global' attributes.

According to the user needs, any artifact in the repository can be linked to other artifacts. Five categories of links are predefined in the iMORE meta-model; they are briefly explained using the following examples:

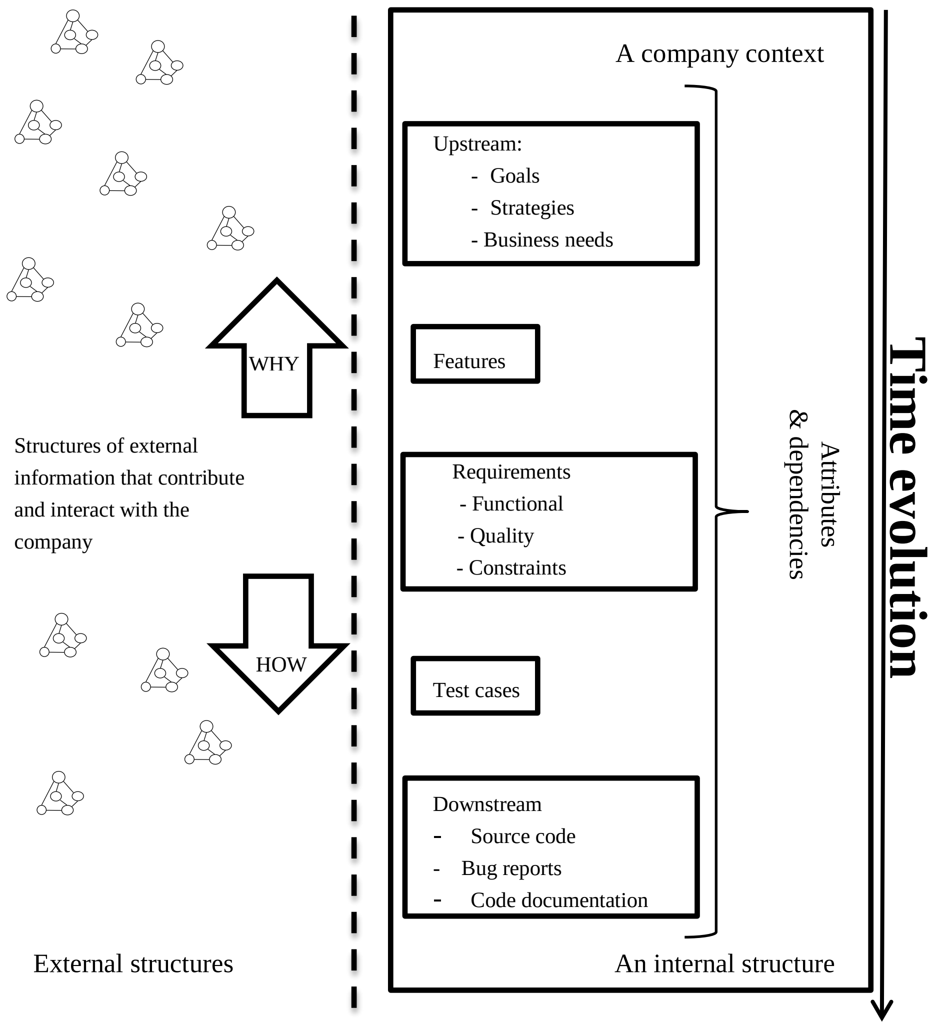

Fig. 2. The iMORE modeling framework.

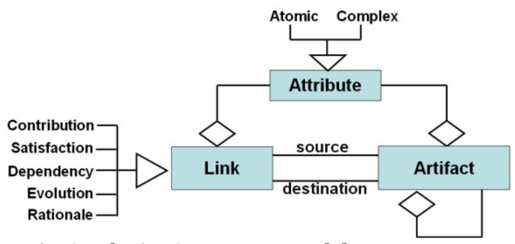

Fig. 3. The iMORE meta-model

- A requirement document A **contributes** to the specification of a design feature B
- A design feature A **satisfies** an external law based constraint B
- A design feature A **depends** on another design feature B
- A design specification A is the result of the **evolution** of a design specification B
- An external law based constraint A is the **rationale** for a requirement document B

These links are exploited in order to find linked elements and to navigate in the repository. If systematically provided by the users, such links can contribute to a full traceability system. Such pre-traceability is often difficult to implement [21], and recent works in requirements traceability advocate combining it with post-traceability based on information retrieval techniques [22]. However, full pre- and post-traceability is not the main goal of this proposal.

5 Discussion of the iMORE modeling framework with practitioners

We present the discussion of the iMORE approach based on five interviews conducted at three companies. During the interviews, we discussed not only the iMORE approach but also the relationships between the suggested meta-model and the challenges our practitioners face in their daily work. The results are outlined according to the interview instrument that can be accessed online [1].

The need for requirements information modeling. All five respondents confirmed the need for modeling requirements information in a more findable and understandable way. One respondent stressed that the need depends on the size of the company indicating that it is much more important for larger projects and companies to have effective requirements architectures in place. Another respondent indicated that the current rather high-level model has limited application and is more suited for high-level roles. Further, the same respondent stressed that the model may help to perform cross-analysis between the projects. Finally, one respondent stressed that the main goal for developing the model is to get better understanding of the knowledge of the market needs and other ‘upstream’ information.

The distinction between the internal and external information in the iMORE approach. Five respondents agreed to the distinction and stressed that external information currently dominated their daily work. Among the types of external information that our respondents need to browse are: standards and regulations, open source code and documentation, marketing resources available on the Internet etc. One respondent indicated that integrating regulations and laws to the model will be counterproductive and it will make the model hard to manage as regulations and laws can change frequently. Two other respondents mentioned that they access open source project information very often since their software product is mostly based on that solution. Those respondents also indicated that full integration of external open source project information is practically unfeasible as these projects change frequently. Further, one respondent indicated that external information is very important when “developing global services for large customers" which confirms our pre-understanding of the importance of external sources of information for projects in VLSRE. Finally, one respondent valued market and business related external information as the most valuable among the external sources. All respondent confirmed that in a large-scale MDRE context improved integration with external sources of information is important and desired.

Representation of attributes and dependencies in the iMORE approach. Two respondents agreed to the idea of separating attributes and dependencies from the requirements information. On the other hand, one respondent disagreed with this idea. Two respondents suggested that dependencies between requirements information elements are also a type of an attribute. Also, one respondent suggested that a "period of validity" attribute should be added. This attribute will improve managing the temporal aspect of the model by giving the engineers triggers and reminders about information becoming outdated that requires their attention. Another respondent indicated that the only important dependencies are one-way relations from visions to requirements and to code. Finally, one respondent suggested reusing patterns from data modeling to investigate which attributes are shared and which are unique to an instance.

Managing the temporal aspect of the information structure. Surprisingly, one respondent indicated that managing the temporal aspect of the information structure isn't so important. Another respondent suggested managing the temporal aspects of the information structure by creating an attribute for every entity called "period of validity". After the period of validity expires the information would need to be updated or deleted. Two respondents suggested implementing a similar system for managing changes based on triggers generated by changes to selected important attributes and entities in the information structure. Finally, one respondent suggested a method based on combining baselines and trigger-based updates. When it is important, a snapshot of external information should be taken and kept until it is no longer relevant. There should be a mechanism of finding the differences between the snapshot and the state of the information structure at the time when the snapshot became out of date. Changes to selected important entities of information should trigger actions, for example notification of substantial changes to the code base as oppose to bug fixes.

6 Conclusions and further work

Concise and quick access to information is critical in large organizations. Information overload impedes knowledge workers both in decision making and information seeking. In large-scale software development, challenging amounts of information are produced and modified throughout the entire development lifecycle. Successful management of requirements is one central activity that demands a robust information model. Increased dependence on external sources of information further stresses the situation. Thus, providing an efficient modeling framework could limit the consequences of information overload in software development projects.

In this paper, we present a modeling framework for requirements-related information for very-large market-driven requirements engineering contexts. The main novelty of our model lies in involving external sources of information and stressing the temporal aspect of the model. We evaluated our proposed with five industry practitioners from three companies. All respondents agreed with the main ideas behind the model. Moreover, they acknowledged that keeping and updating information structures in large scale software development projects is difficult. This finding is in line with previous research in knowledge intensive organizations in general [8] and the software development context in particular [5,9]. Also, our respondents confirmed that including external sources of information and the temporal aspect are strengths of our modeling framework.

Regarding the place of the attributes in the model our respondents gave inconsistent answers. However, attributes were considered as a way of handling the changes of the information structure over time. When queried about ways of managing the time evolution of the model, our respondents suggested creating an attribute for every entity called ‘period of validity’. In order to handle such needs, our approach is based on meta-modeling so that information structures can be defined, modified and managed all along the software project timeline. Our approach is in line with similar works that recognize the important role of abstraction meta-levels in dealing with information interoperability and traceability in software projects [23,24]. From an implementation perspective, it was suggested managing changes in the repository using triggers or by combining baselines and triggers together.

Future works is projected in two directions. First, the iMORE framework presented here can be seen as a set of high level requirement for information architecture and management in VLSRE context. As such, it can form the basis of an evaluation framework for studying and assessing existing information management tools for software engineering (e.g. Rational RequisitePro). A second direction is to further explore stakeholders' requirements concerning information management and integration in very large software projects. This would include enhancing and validating the information meta-model that is sketched in this paper.

References

1. Wnuk, K. The interview instrument can be accessed at http://serg.cs.lth.se/fileadmin/serg/II.pdf (2012)
2. Regnell, B., Berntsson Svensson, R., and Wnuk, K.: Can We Beat the Complexity of Very Large-Scale Requirements Engineering?. In: Paech, B., Rolland, C., (eds.) REFSQ 2008. LNCS, vol. 5025, pp. 123-128. Springer-Verlag, Berlin, Heidelberg, (2008)
3. Wnuk, K., Regnell, B., and Berenbach, B.: Scaling Up Requirements Engineering – Exploring the Challenges of Increasing Size and Complexity in Market-Driven Software Development. In: Berry, D., Franch, X., (eds.) REFSQ 2011, LNCS 6606, pp. 54-59. Springer-Verlag, Berlin, Heidelberg, (2011)
4. Berenbach, B., Paulish, D., J., Kazmeier, J., Rudorfer A.: Software & Systems Requirements Engineering: In Practice, McGraw-Hill, New York (2009)
5. Olsson, T.: Software Information Management in Requirements and Test Documentation. Licentiate Thesis. Lund University, Sweden (2002)

6. Cleland-Huang, J., Chang, C. K, Christensen, M.: Event-based traceability for managing evolutionary change. Trans. Soft. Eng. 29, 796-810 (2003)
7. Morville. P.: Ambient Findability: What We Find Changes Who We Become. O'Reilly Media, (2005)
8. Karr-Wisniewski, P., Lu, Y.: When more is too much: Operationalizing technology overload and exploring its impact on knowledge worker productivity. Computers in Human Behavior, 26, 1061-1072 (2010)
9. Sabaliauskaite, G., Loconsole, A., Engström, E., Unterkalmsteiner, M., Regnell, B., Runeson, P., Gorschek, T., Feldt, R.: Challenges in aligning requirements engineering and verification in a Large-Scale industrial context. In: Wieringa R., Persson, A, (eds.) REFSQ 2010. LNCS 6182, pp. 128-142. Springer-Verlag, Berlin, Heidelberg, (2010)
10. Wnuk, K., Regnell, B., Schrewelius, C.: Architecting and Coordinating Thousands of Requirements – An Industrial Case Study. In: Glinz, M., Heymans, P. (eds.) REFSQ 2009, LNCS 5512, pp. 118-123. Springer-Verlag Berlin Heidelberg (2009)
11. Zantout, H., Document management systems from current capabilities towards intelligent information retrieval: an overview. Int. J. Inf. Management. 19, 471-484 (1999)
12. Gorschek, T., Svahnberg M., and Tejle K.: Introduction and Application of a Lightweight Requirements Engineering Process Evaluation Method, Proc. of the 9th Int. Workshop on Requirements Eng.: Foundation for Software Quality (REFSQ 2003), 101-112 (2003)
13. Swain, M. R., and Haka, S. F.: Effects of information load on capital budgeting decisions. Behavioral Research in Accounting 12, 171–199 (2000)
14. Eppler, M., Mengis, J.: The Concept of Information Overload - A Review of Literature from Organization Science, Accounting, Marketing, MIS, and Related Disciplines. The Information Society. 20. 325-344 (2004)
15. Chewning, E. C., Jr., Harrell, A. M.: The effect of information load on decision makers' cue utilization levels and decision quality in a financial distress decision task. Accounting, Organizations and Society. 15, 527–542 (1990)
16. Cook, G. J. An empirical investigation of information search strategies with implications for decision support system design. Decision Sciences 24, 683–699 (1993)
17. Siau, K., Rossi, M.: Evaluation techniques for systems analysis and design modeling methods – a review and comparative analysis. Inf. Systems Journal. 21(3), 249-268 (2011)
18. Easterbrook, S., Singer, J., Storey, M-A., Damian, D.: Selecting Empirical Methods for Software Engineering Research. In: F. Shull et al. (eds.) Guide to Advanced Empirical Software Engineering, Springer, 285-311 (2008)
19. Ramesh, B., Jarke, M.: Toward reference models for requirements traceability. IEEE Transactions on Software Engineering. 27(1), 58–93 (2001).
20. El Ghazi, H., Assar, S.: A multi view based traceability management method. In 2nd Int. Conf. on Research Challenges in Inf. Science, 393-400. IEEE Computer Society, (2008)
21. Cleland-Huang, J., Settimi, R., Romanova, E., Berenbach, B., Clark, S.: Best Practices for Automated Traceability. Computer. 40(6), 27–35 (2007)
22. Borg, M., Pfahl, D.: Do better IR tools improve the accuracy of engineers' traceability recovery? Int. Workshop on Machine Learning Technologies in Soft. Eng., 27–34, (2011)
23. Terzi, S., Cassina, J., Panetto, H.: Development of a Metamodel to Foster Interoperability along the Product Lifecycle Traceability. In: Konstantas, D., Bourrières, J.-P., Léonard, M., et Boudjlida, N. (Eds.) Interoperability of Enterprise Software and Applications. 1-11. Springer, London, UK (2006)
24. Cavalcanti, Y.C., do Carmo Machado, I., da Mota, P.A., Neto, S., Lobato, L.L., de Almeida, E.S., de Lemos Meira, S.R.: Towards metamodel support for variability and traceability in software product lines. Proc. of the 5th VaMoS Workshop ACM, NY, (2011)