A Traceability Semantics Approach for Supporting Product Value Analysis

Angelina Espinoza
Technical University of Madrid (UPM),
E.U. Informática, Ctra. de Valencia Km. 7
28031 Madrid, Spain
aespinoza@syst.eui.upm.es

Jessica Díaz
Technical University of Madrid (UPM),
E.U. Informática, Ctra. de Valencia Km. 7
28031 Madrid, Spain
yesica.diaz@upm.es

ABSTRACT

Product requirements prioritization approaches identify the most valuable requirements according to customer value, requirements risk, volatility, cost or other market parameters. However, it is still a challenge in Value-Based Engineering to manage the requirement values for incrementing the product value. The analysis of the product value requires a better understanding of interdependencies among various artifacts of the software development lifecycle, particularly, requires to get a better understanding of how the value chain is preserved from the *problem space* to the *solution space*. In this position paper our aim is to study the feasibility of using traceability as the backbone to preserve the value chain between the prioritized product requirements and the product architecture. We propose traceability semantics to address this aim. Some initial findings from our research are presented.

Categories and Subject Descriptors

D.2.1 [**Software Engineering**]: Requirements/Specifications; D.2.9 [**Software Engineering**]: Management—*Lifecycle, Cost Estimation*; D.2.11 [**Software Engineering**]: Software Architectures—*Languages*

General Terms

Management, design, documentation

Keywords

Traceability semantics, product value, product value chain, product features, product architecture

1. INTRODUCTION AND MOTIVATION

The requirements prioritization process is fundamental for successfully planning the product releases and for supporting the product risk-management. In Value-Based Engineering, the product requirements prioritization process is even more important and complex because it is used not only for the product planning and control, but also, for the planning and control of the value delivered to stakeholders [1]. There are a lot of requirements prioritization techniques [6] to identify the most valuable requirements according to various criteria such as customer value, requirements risk, volatility, cost or other market parameters.

Those value-based prioritized requirements are then further allocated in product features. During this task some approaches could be used to preserve the value of the requirements, for instance our approach is to calculate the weight among the values of the requirements related to a feature. This weight in the simplest form could be an average among the requirement values. Then, this weight will be assigned to the product features for transferring the value from requirements. As we could realize, the value chain can be preserved in a practical manner in some extent. The management of the values is still feasible because we are still in the problem space.

However, how the value chain is preserved from product features to product architecture is a current challenge. The reason is the distance between the *problem space* and the *solution space*, motivated by the exponential complexity number arisen when the product architecture's artifacts are built. The lack of an efficient solution to preserve the values assigned to the requirements through the subsequent development phases, may directly affect the analysis for determining the global product value.

The analysis of the product value requires a better understanding of interdependencies among various artifacts of the software development lifecycle, particularly, requires to get a better understanding of how the value chain is preserved from *problem space* to the *solution space*. Thus, the specific research question introduced in this position statement is about how to preserve the values from the prioritized requirements to the product architecture with the aim of supporting the analysis of the product value. In particular, we propose the use of traceability to preserve the value chain. A proper approach towards this direction would also provide the basis for determining the resulting value of the final product when changes are introduced into the prioritized product requirements. Value-Based Product Evolution and Release Planning would directly result benefited with this approach.

2. PRODUCT VALUE ANALYSIS BASED ON TRACEABILITY SEMANTICS

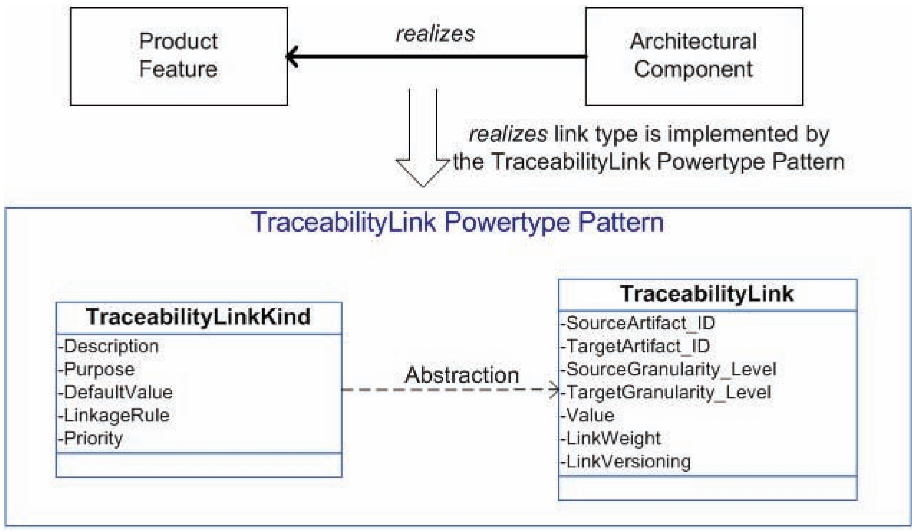

Figure 1: *TraceabilityLink* powertype pattern supporting traceability semantics: the class *TraceabilityLinkKind* is used to define the link *types*, and the *TraceabilityLink* class is used to create links during the project development. The *value* concept is present in both classes.

Traceability is the ability to describe and follow the life of a software artifact and a means for modeling the relations between software artifacts in an explicit way [5]. Traceability is an efficient tool to facilitate communication, improve the system development processes performance; and the most important for this position statement approach, traceability is a widely known technique to preserve the knowledge through the product development lifecycle [2, 7, 5].

Our approach is based on defining traceability links which includes the value concept in its semantics definition. In this sense, the assigned value to a prioritized requirement will be included as part of the data of the traceability link among a specific requirement, the product features and the further product architecture's artifacts (components, interfaces, services. etc.) which realize the requirement.

The proposed approach has been implemented by means of a *Traceability Link* powertype pattern [3, 4], as a template to create the traceability links among the value-based prioritize requirements, the product features, and the product architecture's artifacts which *realize* those requirements (see Figure 1). The Traceability Link powertype pattern is an entity composing of two classes: one is for supporting the definition of traceability link *type* (see Figure 1 *TraceabilityLinkKind*), and the other for supporting the creation of the links during the project development (see Figure 1 *TraceabilityLink*).

Our approach enriches the Traceability Link pattern's classes by means of adding the value to this traceability semantics. As it is shown in Figure 1, an attribute for assigning a *default value* to the *TraceabilityLinkKind* class has been added (*DefaultValue* attribute in Figure 1). Later, this default value will be automatically assigned to all links to be created from defined link types. Additionally, the *TraceabilityLink* class includes another attribute for changing the default value of the link *type* (*Value* attribute in Figure 1), according to the value of the particular relation between two artifacts (SourceArtifact_ID and TargetArtifact_ID attributes in Figure 1).

Therefore, if a value of a prioritized product requirement is assigned to a specific traceability link *type*, e.g. the *realizes* link type in Figure 1; then the assigned value is automatically assigned to all the links created from such *realizes* type. Obviously, this default value can be changed to introduce specific values, for instance, the value of a link between a product feature and the architectural component which realizes the feature. For this purpose the *Value* attribute of the specific link will be used.

Moreover, our approach supports that the assigned values to traceability links are weighted when a given product architecture's artifact partially realizes several prioritized requirements. Here, the traceability link value should be weighted (see in Figure 1 the *LinkWeight* attribute of the *TraceabilityLink* class) between all the values of the prioritized product requirements which are partially realized by one architecture' artifact, e.g. a component partially realizing more than one of the product features. Finally, *TraceabilityLink* class also includes other attributes to keep the version and granularity levels (coarse-grained or fined-grained levels) as it is shown in Figure 1.

3. FURTHER WORK

The next research step is as follows:

1. To develop a traceability metamodel between product features and architecture model, based on the traceability semantics previously explained.
2. To define the process to perform the product value analysis, based on such metamodel.
3. To develop a case study, considering the application scenarios for managing the product value, taken from software industry.

Some benefits that we expect are:

1. The improvement of the value chain management during the development lifecycle of a software product. Specifically, the improvement is obtained by preserving the value from the requirements to architecture.

2. This approach will be used for determining the resulting value of the final product when changes are introduced into the prioritized product requirements. Value-Based Product Evolution and Release Planning would directly result benefited with this approach.

Acknowledgments

This research work has been partially supported by the projects INNOSEP TIN 2009-13849 (MICCIN, Spain), VULCANO TSI-020301-2009-1 (MITYC, Spain) and UPM under their Researcher Training program.

4. REFERENCES

[1] B. Boehm. Value-based software engineering. *ACM SIGSOFT Software Engineering Notes*, 28-2:1–12, 2003.

[2] A. Egyed and P. Grunbacher. Automating requirements traceability: Beyond the record and replay paradigm. In *Proceedings of the International Conference on Automated Software Engineering (ASE)*, pages 163–171. IEEE, 2002.

[3] A. Espinoza, G. Botterweck, and J. Garbajosa. A formal approach to reuse successful traceability practices in SPL projects. In *Proceedings of SAC 2010*, 2010.

[4] A. Espinoza and J. Garbajosa. Tackling traceability challenges through modeling principles in methodologies underpinned by metamodels. In *CEE-SET WiP 2008 Proceedings*, pages 41–54, Brno, Czech Republic, 2008. Oficyna Wydawnicza Politechniki Wroclawskiej.

[5] P. Lago, H. Muccini, and H. van Vliet. A scoped approach to traceability management. *J. Syst. Softw.*, 82(1):168–182, 2009.

[6] L. Lehtola. Providing value by prioritizing requirements throughout software product development. state of practice and suitability of prioritization methods. Technical report, Doctoral Thesis dissertation. Helsinki University Of Technology, 2006.

[7] B. Ramesh and M. Jarke. Toward reference models for requirements traceability. *IEEE Transactions on Software Engineering*, 27(1):58–93, Jan 2001.