Assessing the Strategy Relevance of Delta Requirements
A Bottom-up Feature Generation Approach

Gabriele Zorn-Pauli , Barbara Paech

University of Heidelberg, Institute of Computer Science, Im Neuenheimer Feld 326, 69120 Heidelberg, Germany
{zorn-pauli, paech}@informatik.uni-heidelberg.de

Abstract. Strategic release planning is applied to decide which features to implement in which release to ensure competitive advantage. Therefore, continuously arriving changes to requirements at different levels of abstraction have to be aligned with business strategies. Especially, incrementally developed software systems have to cope with a flow of incoming delta requirements that specify enhancements to existing functionality. It is challenging for product managers to relate delta requirements to business strategies and to assess whether a delta requirement is relevant to business strategies scheduled for a specific release. The major idea of the proposed bottom-up feature generation approach is to handle these delta requirements at feature level to relate them more effectively to business strategies. The contribution of this paper is two-fold. First, a more detailed description of the already introduced Requirement Abstraction and Solution Model (RASM) to address requirements and solution abstraction. Second, a bottom-up feature generation approach that maintains RASM and which helps to assess the business strategy relevance of continuously arriving delta requirements.

Keywords: strategic release planning, roadmapping, feature generation, requirement bundling, delta requirement classification, requirement abstraction

1 Introduction

The development of software systems is characterized by continuous change to requirements at different levels of abstraction, 'responding to evolving requirements, platforms, and other environmental pressures' [2]. Passos et al. [7] postulated a vision that feature orientation of software design and of the software development process is able to handle change more effectively than previous methods. They stated the controlling and executing of change as the major challenge for most software projects. To align these changes with business strategies, strategic release planning (SRP), also called roadmapping, is applied to decide how features are scheduled for different releases to ensure competitive advantage. Therefore, SRP translates business strategies top-down into *business features* (BF), linking the business view with requirements engineering [6].

In an industrial case study [8], we explored that SRP in practice generates features also bottom-up by bundling continuously arriving delta requirements into *software features* (SF). These software features represent 'cohesive bundles of requirements ad-

dressing important capabilities of the system' [1]. *Delta requirements* specify enhancements to existing system functionality at a low level and require a relation to the existing system to provide an understanding of the delta [4]. Without knowing the relation between delta requirements and the existing system, it is difficult for release planning teams or product managers to understand the impact of delta requirements on existing BF structures and further release plans. Additionally, for a software product that is used globally, customers from different countries raise delta requirements that are motivated by their country-specific business strategies and processes. Another challenging task is to recognize delta requirements duplicates in large-scale projects.

This paper provides two contributions. First, a more detailed description of the already introduced Requirement Abstraction and Solution Model (RASM) [8] to handle requirements and solution abstraction during SRP. Second, a bottom-up feature generation approach that maintains RASM and which helps to assess the business strategy relevance of continuously arriving delta requirements.

The remainder of this paper is structured as follows: Section 2 provides a detailed description of the RASM using an application example. In Section 3, the bottom-up feature generation approach steps are introduced. Section 4 provides a discussion of RASM application and open issues. Finally, Section 5 concludes the paper and provides an outlook to future work.

2 RASM

In [8], we reported on an industrial case study to analyze the SRP process of a company in the health care domain, whereby we gathered four major requirements for feature generation support: *(FG1) support top-down and bottom-up feature generation, (FG2) support the aggregation of relevant changes into existing release plans, (FG3) support delta requirements handling, and (FG4) support feature classification and variability.*

The RASM has already been introduced briefly in [8] as a preliminary solution proposal to address these requirements. It extends the Requirement Abstraction Model (RAM) [3] by distinguishing explicitly between BF and SF to support top-down and bottom-up requirement and solution abstraction. In this section, we describe a RASM application example using real data from the company.

Business Strategies such as *Deal Compliance & Monitoring* and *Pricing Tool Enhancements*, shown in Figure 1 (a), represent business case initiatives with the different priorities 9 and 7, in a range from 1 (extremely low) to 9 (extremely high). In most cases, companies pursue several strategies at the same time and SRP aims at selecting features that optimally support highly ranked business strategies. Therefore, each priority change has an impact on existing release plans, which makes it necessary to keep business strategy priorities up-to-date.

Business Features (BF) represent business strategies at product level. The example provided in Figure 1 (a) illustrates two BFs derived in a top-down manner: *Instrument Freight Handling* and *Bloodgas Business*. They are related to a business strategy and describe business requirements at a high-level independently of the existing software system. In a globally operating company, there are several company sites in different countries (Ci) that have varying priorities (Pi) for BFs based on, for example different markets. To successfully apply SRP for a globally used software system, these BFs

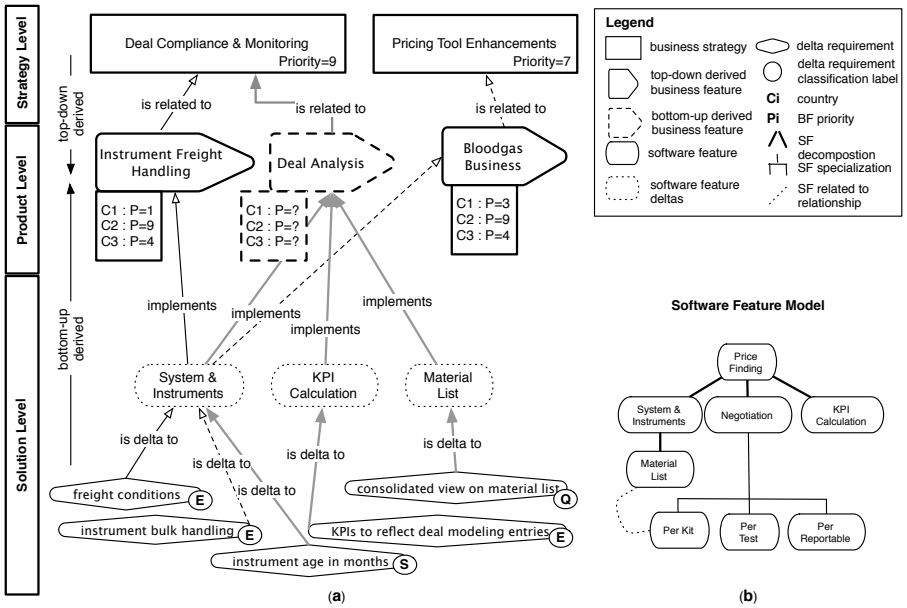

Fig. 1. (a) RASM application and (b) SF model example

have to be implemeted and provided first that guarantee the largest common overlap with respect to highly prioritized BFs by the different countries. Therefore, the country priorities of BFs have to be kept up-to-date and BFs must have a relation to one or more business strategies. The bottom-up generation of BFs is described in Section 3.

Software Features (SF) represent the high-level abstraction of an existing software system describing functional behavior or capabilities of an existing system. The different SFs are related to one another based on three different SF dependencies *decomposition*, *specialization*, according to [5], or *related to*. Figure 1 (b) illustrates a SF model excerpt of the software system developed by the company. In this example, the SF *System & Instruments* represents a decomposition of *Price Finding*, which means that *System & Instruments* refines the functional behavior of *Price Finding*. The *specialization* dependencies between *Negotiation* and SFs *Per Kit*, *Per Test* and *Per Reportable* indicate alternative functionalities of the SF *Negotiation*. The dependency *related to* indicates that the SF *Per Kit* and *Material List* are related to each other and a change to *Material List* has an impact on SF *Per Kit*.

Software Feature Deltas, as shown in Figure 1 (a), are those SFs that bundle one or more delta requirements, as for example *System & Instruments*.

Delta Requirements describe an enhancement to a specific SF [4]. We classify delta requirements using the delta requirement classification labels, described in Table 1. For example, in Figure 1 (a), delta requirement *instrument age in months* is linked with the SF delta *System & Instruments*. Since this delta comprises a specialization of an existing functionality, with the age of instruments managed in months instead of only years, the delta requirement is labeled with **S** for *specialization delta*.

Table 1. Delta requirement classification labels

Label	Description
Configuration delta	requires a configuration of the system that does not require an implementation of functionality.
Quality delta	addresses quality improvements, e.g. usability or performance.
Specialization delta	requires a specialization of existing functionality, with which the new and the existing functionality can be used optionally, e.g. filtering deals per day, week or also per month.
Enhancement delta	requires an enhancement of existing functionality.
Exclusive delta	requires a functionality that is conflicting with an existing functionality when behaving simultaneously.
Substitution delta	replaces existing functionality, where the old functionality is no longer available.

Applying RASM to structure SRP relevant information addresses the feature generation requirements FG1-FG4 and provides the following additional benefits. An *incremental documentation of a specific release* is provided, where business goals are related to solution specifications. Furthermore, a *business-oriented software feature-based representation of the whole system* is provided, which enables the comparison of different release versions based on any RASM meta-model element. For example, the following question can be answered: *What is the difference between release version X and Y with respect to SFs and corresponding delta requirements?* This can be a good basis to give a first estimation of the resulting end user training effort. Moreover, *SFs can be more appropriate to ask end users about SF usage or satisfaction*, because a SF model can be directly related to user manuals. Furthermore, the *identification of countries with similar markets*, based on their high priorities for the same BFs, can be easily supported. Additionally, *a measure for business strategy coverage* can be provided after feature selection is made, because the selected feature sets can be assessed concerning their strategy coverage. Finally, *the prioritization effort is reduced*, because the prioritization of low-level delta requirements is not necessary.

3 Bottom-up Feature Generation Approach

The major idea of the proposed approach is to handle continuously incoming low-level delta requirements at feature level to relate them more effectively to business strategies, where bug requests are not considered by the approach. These delta requirements are raised by the different company sites countries during support, test phases or roll-out projects and are bundled based on their SF belonging. Therefore, the approach includes the maintenance of RASM-relevant information in a bottom-up manner and assumes that an instance of the RASM exists. A RASM instance represents a specific increment (release) with respect to selected BFs and a SF model of the whole system. In the following the five steps of this approach are explained by describing goal and support of every step.

STEP1 Goal: ***The delta requirement is classified using delta requirement classification labels.*** Support: The classification of delta requirements is supported by the

given requirement classification catalogue in Table 1.

STEP2 Goal: ***The delta requirement is unambiguously related to one SF.*** The SF model-based representation of the existing system comprises "decompose" and "specialize" dependencies. If there are several potential SFs identified for a specific delta at the same level, the delta requirement is related to the parent SF. This indicates that a delta requirement potentially has an impact on all decomposed SFs below and the parent SF would be raised as a newly generated *software feature delta* in the model. For example, if the delta requirement *instrument age in months*, illustrated in Figure 1 (b), was identified as relevant to the SF *System & Instruments* and *KPI Calculation* the delta is related to *Price Finding*. Support: To support the identification of related SFs the requirement clustering approach according to [1] can be used to identify a relationship between the delta requirement under consideration and already bundled delta requirements.

STEP3 Goal: ***Delta requirement duplicates or conflicts are identified and removed.*** Support: Through the bundling of related delta requirements into the same SF group, duplicates are easier to detect. Additionally, the classification of delta requirements supports the recognition of conflicts between delta requirements.

STEP4 Goal: ***Software feature deltas (existing features that have to be changed) are identified, where these are the basis for release planning.*** Support: All SFs that were related to one or more delta requirements represent software feature deltas. Based on the classification of delta requirements, SF deltas can also be classified based on the deltas they bundle. For example, a SF delta comprising only delta requirements classified as configuration delta can be classified as a configuration SF delta, too.

STEP5 Goal A: ***The relation to the identified SF delta and a BF is assessed.*** Goal B: ***The relation to the identified SF delta and a business strategy is assessed and a new BF is generated.*** Support A and B: To identify a relation to a BF, already existing links between the SF, to which the delta requirement is related to, and BFs can be considered. Regarding the application example in Figure 1, the delta requirement *instrument age in months* is related to *System & Instruments*. We can see that there are already relations to two different BFs and it can be assessed whether the delta requirement is related to one of them or not. In the case of the application example, non of the existing BFs is sufficient, but a relation to the business strategy *Deal Compliance & Monitoring* is recognized. Therefore, a new BF *Deal Analysis* is generated that can be used for SRP purposes, e.g. gathering country priorities in order to assess the common priority.

4 Discussion

We have made some first experiences on generating a RASM instance at the company. The most challenging task was the generation of the SF model, representing the software system functionality at a high-level that enables to relate and further to understand delta requirements. Since, system solution specifications are only available incrementally from one release to another, user manuals and release notes are a good source to identify existing SFs and their relations. Once, a SF-based representation of the system exists, the proposed bottom-up feature generation approach maintains RASM by relating continuously arriving delta requirements to SFs and identifies SF deltas for release planning purposes. Furthermore, if SF deltas are recognized as business strategy

relevant, but cannot be related to existing BFs, a new BF is generated in a bottom-up manner. However, the top-down maintenance of the RASM, deriving BFs or SF deltas from business strategies, is an open issue.

Moreover, there are additional open issues that have to be discussed. It is not clear whether the SF-based representation of the system is detailed enough to understand delta requirements and to bundle them effectively. So far, the handling of delta requirements is not sufficiently addressed in requirements engineering research as stated by Herrmann et al. [4]. Further, it is not clear whether it is realistic that a suitable specification of the whole system is available in practice that can be used to sufficiently identify the belonging of delta requirements. Furthermore, we have not validated the suitability of the proposed delta requirement classifications to classify SF deltas. This requires, that SF deltas bundle a homogenous type of delta requirements, which is not always the case.

5 Conclusion and Future Work

In this paper we provided two contributions. First, we gave a more detailed explanation of the RASM, which copes with requirements and solution abstraction during SRP. Second, we provided a bottom-up feature generation approach to maintain the RASM and that helps to assess the business strategy relevance of continuously arriving delta requirements. Future work will include a case study in a large-scale industrial setting to evaluate the scaleability of RASM and the proposed bottom-up feature generation approach.

References

1. Chen, K, Zhang, W., Zhao, H.: An approach to constructing feature models based on requirements clustering. In: 13th IEEE International Conference on Requirements Engineering (RE05). pp. 31-40. (2005)
2. Godfrey, M.W., German, D.M.: The past, present, and future of software evolution. Frontiers of Software Maintenance (FoSM'08), pp. 129-138. (2008)
3. Gorschek, T., Wohlin, C. Requirements Abstraction Model. Journal of Requirements Engineering 11(1), pp. 79-101 (2005)
4. Herrmann, A.,Wallnoefer, A., Paech, B.: Specifying changes only - a case study on delta requirements. In: 15th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ'09), pp. 45-58 (2009)
5. Kang, C. K., Cohen,S., Hess,J., Nowak, W., Peterson, S.: Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University. (1990)
6. Lehtola, L., Kauppinen, M., Kujala, S.: Linking the business view to requirements engineering: long-term product planning by roadmapping. In: 13th IEEE International Conference on Requirements Engineering (RE05). pp. 439-443. (2005)
7. Passos, L., Czarnecki, K., Apel, S., Wsowski, A., Kaestner, C., Guo, J.: Feature-oriented software evolution. In: 7th International Workshop on Variability Modelling of Software-intensive Systems (VaMoS 13). pp. 1-8 (2013)
8. Zorn-Pauli, G., Paech, B., Beck, T., Karey, H.: Analyzing an industrial strategic release planning process - a case study at Roche Diagnostics. In: 19th International Working Conference on Requirements Engineering: Foundation for Software Quality (2013) (accepted to appear)