The Mythical Adequate Level of Details

Bundling Requirements for Large Scale Market Driven Engineering

Daniel Lucas-Hirtz
Exibri, www.exibri.com
Rennes, France
daniel@exibri.com

Mahvish Khurum
Blekinge Institute of Technology
Karlskrona, Sweden
mahvish.khurum@bth.se

***Abstract*— requirements bundling is known to help coping with the complexity of large scale requirements engineering. The multiplicity of the dimensions of this bundling is not often emphasized, yet causes issues which remain unsolved.**

Keywords - market driven requirements; requirements bundling; software product lines; structure of requirements engineering artifacts

I. INTRODUCTION

As shown in [1], the structure of requirements engineering artifacts is a major challenge within large scale (aka product lines or product families) market driven requirements engineering. We name “requirements bundle” the unit of organisation for this structure.

This paper proposes to orthogonally organise requirements bundling across five major engineering concerns namely “input requirements”, “governance”, “architecture”, “variability”, and “project”. These bundles are often not explicitly prescribed to be scoped independently, driving the impression that requirements bundling is a one dimension problem; where key is the "adequate level of details" - and that a well scoped “feature” will kill all flies in one shot. However, we have experienced in industry that this approach does not scale up.

Therefore, in this paper 1) we discuss the concept of requirements bundles based on the essential engineering concerns and 2) we state the corresponding issues which need further exploration.

II. FIVE ENGINEERING CONCERNS

"Requirements" are a central input to the following five concerns of market driven engineering organisations:

TABLE I. ENGINEERING CONCERNS

Concern:		Purpose:
1	Input requirements	State the problem (customer & market requirements, etc.)
2	Governance	Decide (change control, release planning, etc.)
3	Architecture	Develop & maintain the reusable assets
4	Variability	Customize the platform into a customer dedicated product
5	Project	Manage resources and milestones

III. FIVE REQUIREMENTS BUNDLES

In our experience of large scale contexts, the actual unit of management of the five concerns above is not the "requirement", but orthogonal requirements bundles, one for each concern. These bundles are explained in TABLE II:

TABLE II. REQUIREMENTS BUNDLES

Requirements bundle:		Unit of:	Examples
1	Request	Documentation of the market requirements	need, goal, requirements document, e-mail, phone call
2	Delta (as in [2])	Decision, commitment, governance	release, feature, change request
3	Component	Organisation of the reusable assets	reusable specification document
4	Variation point	Documentation of the variability	variant, parameter, option, flex
5	Task	Allocation of the resources	task scope specification, project plan

IV. PROBLEM STATEMENT

We foresee following issues to be resolved:

A. *Moving away from the "everything is requirement" paradigm*

Large scale engineering, when focused on unitary requirements, risks breaking apart, reason being 1) an increasing number of requirements, and 2) an attempt to stress out requirements processes as a way to keep control. We propose here to shift the focus from “requirements” towards higher level requirements bundles (see for example the RAM in [4]). This change of habits is sometimes overlooked in the literature.

As an example, release planning of a mobile phone with thousands of requirements is very complex, and one may investigate sophisticated prioritization techniques to address this complexity. We believe that prioritization, as a tool for decision, should focus on "units of decision" (which we call “deltas”). We therefore propose to shift the main focus of release planning from "how to evaluate the priority" towards "how to scope the deltas". Once these are correctly scoped and their number decreases drastically, their actual evaluation is likely to be far simpler.

B. Moving away from the "everything is feature" paradigm

The “feature” is probably the most popular requirements bundle, as pointed out in [5]. The term “feature” often qualifies both the unit of customer requirements (features seen as “selling units” in [6]), the unit of decision (as in release planning), the unit of variability (as in FODA’s “feature model” in [7]) and sometimes the unit of domain engineering specification (ex. “feature specifications”). Using the same term risks to let one think that the various concerns stated above probably need a single requirements bundle. We, however, have experienced in industry that this does not scale up and may lead to severe failures (as in [8], see also [3]). We therefore avoid the term “feature” for requirements bundles.

C. Scoping requirements bundles

A fundamental challenge of working with requirements bundles is to scope requirements correctly into bundles.

Let us again take the example of the “units of decision”. In [9], Smits proposes “5 Levels of Agile Planning”, from “product planning” up to “daily scrum”; and the associated units of decision, from "feature" to "user story" and "task". The scoping of these various units of decision becomes the central planning activity, and the hierarchical planning allows delaying decisions whenever possible. Similarly, Fricker proposes in [10] to model the set of "features" as a tree rather than a flat list, to ensure that the actual decision process can hierarchically first focus on the most important and meaningful decisions, while keeping the ability to progressively refine the focus on smaller grained decisions.

In both cases above, working with bundles requires a fundamental process change from “taking the right decision” towards “scoping the proper unit of decision”. This change will greatly impact the overall engineering organisation.

D. Keeping them all together

Once the bundles are scoped, the challenge lies in “keeping them all together” to maintain a holistic repository throughout the life of the product line. As suggested by Lamsweerde in [11], traceability, as the art of “documenting for evolutions”, may be the way to go. However, the challenge is far from being addressed. Requirements engineering is already known to be complex, and traceability hard to maintain. In a five dimensions space, the combination of the relations is likely to explode, and traceability might no longer be possible.

It is yet to be investigated whether the above complexity increase is balanced by some simplifications in either the requirements or in their relations.

Figure 1 below shows the example context and relationships between the proposed bundles (given in TABLE II) along with the associated engineering concerns (as given in TABLE I).

References

[1] K. Wnuk, B. Regnell, B. Berenbach: Scaling Up Requirements Engineering - Exploring the Challenges of Increasing Size and Complexity in Market-Driven Software Development, Requirements Engineering for Software Quality, March 2011, Essen, Germany, dates, p54-59

[2] P. Carlshamre, B. Regnell: Requirements Lifecycle Management and Release Planning in Market-Driven Requirements Engineering Processes, DEXA Workshop 2000

[3] S. Bühne, K. Lauenroth, K. Pohl, Why is it not Sufficient to Model Requirements Variability with Feature Models? AURE04, 2004

[4] T. Gorschek and C. Wohlin, Requirements Abstraction Model, Requirements Engineering Journal, Vol. 11, No. 1, pp. 79-101, 2006

[5] A. Classen, P. Heymans, P.Y. Schobbens, What’s in a Feature: A Requirements Engineering Perspective, ETAPS 2008

[6] G. Ruhe, The Battle for the Right Features or: How to Improve Product Release Decisions ?, 2010

[7] K. Kang, S. Cohen, J. Hess, W. Nowak and S.Peterson, "Feature-Oriented Domain Analysis (FODA) Feasibility Study", Technical report 1990.

[8] D. Lucas-Hirtz, Practical requirements reuse - How to initiate improvement [..] REFSQ 2011, http://filesocial.com/a7lml5

[9] H. Smits, 5 Levels of Agile Planning: From Entreprise Product Vision to Team Stand-up, Rally Software

[10] S. Fricker, S. Schumacher, Variability-based Release Planning, ICSOB 2011

[11] A. van Lamsweerde, "Requirements Engineering - From System Goals to UML Models to Software Specifications" Wiley 2009

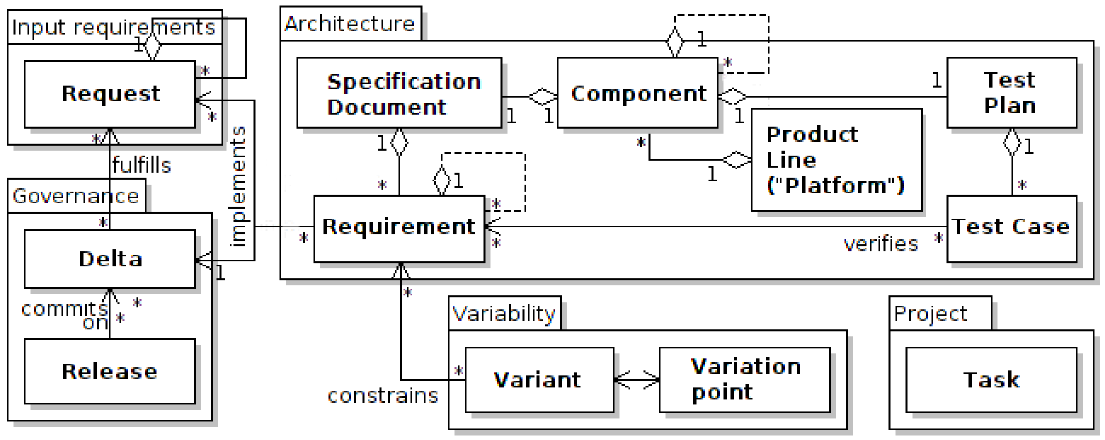

Figure 1. Example representation of five engineering core concerns, their associated units of management, and some possible relations.