

Method for Managing Variability in Software Architecture

Mikko Raatikainen

Helsinki University of Technology (TKK)

Software Business and Engineering Institute (SoberIT)

www.soberit.hut.fi/mjraatik

www.soberit.hut.fi/preago



Objective / Outline

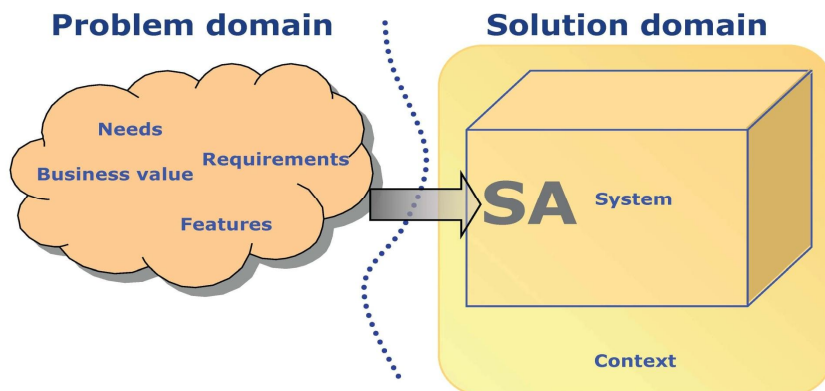
- To give an introductory overview of my research
- Most relevant background
 - Software architecture and variability
- Topic
 - Method for Managing Variability in Software Architecture
- Research methodology
 - Design science oriented approach
- Recent research results



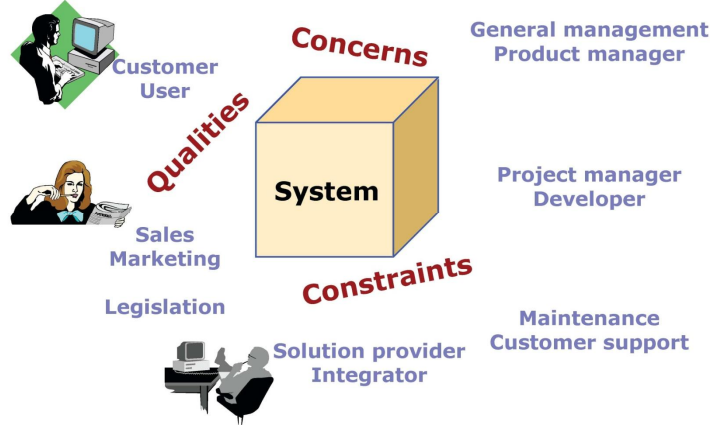
Background



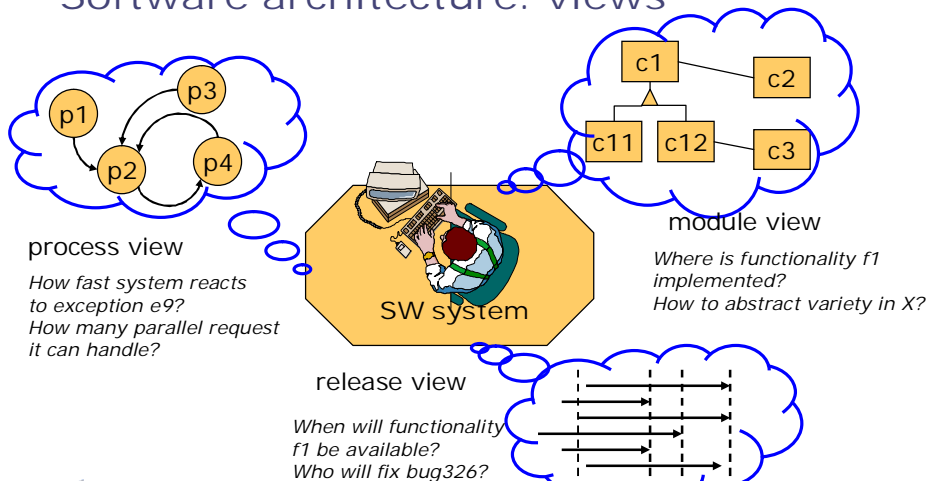
Software architecture



Software architecture stakeholders



Software architecture: views

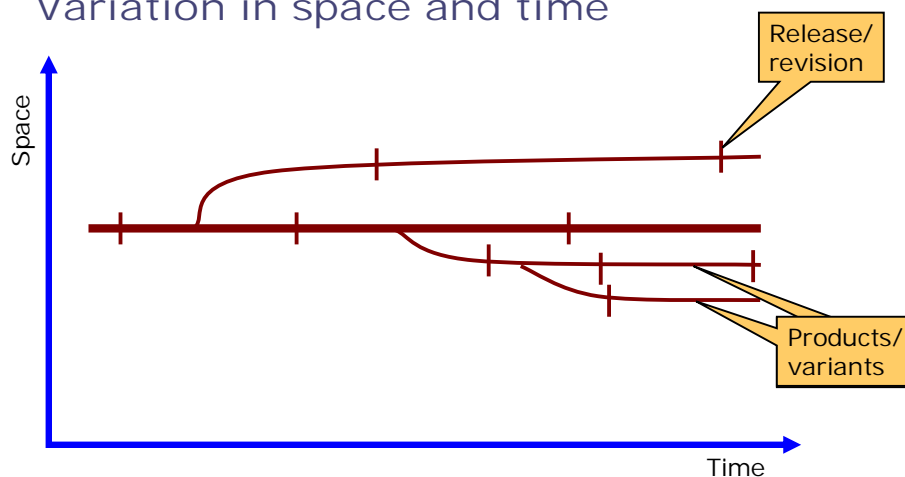


Software architecture

- Bass et al 2003
 - *The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships between them.*
- IEEE 1471
 - *the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution*

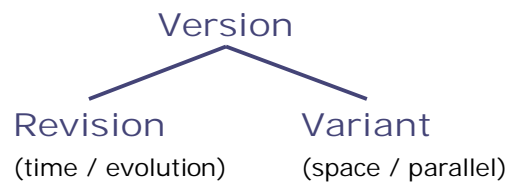


Variation in space and time

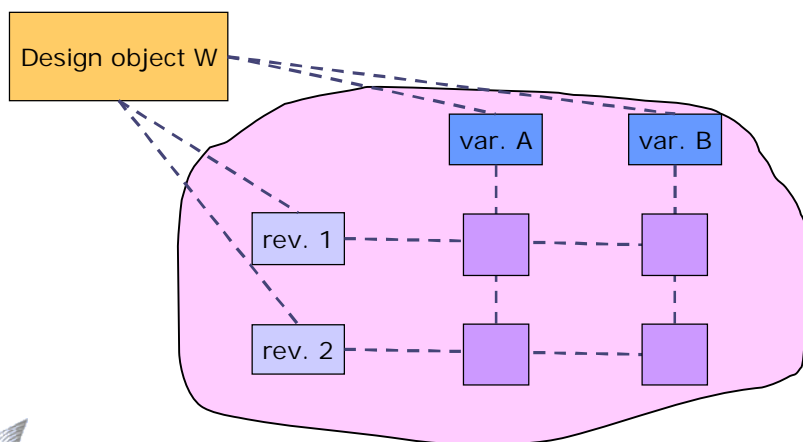


Variation in space and time - Terms

- Typical usage, although not always strictly followed



Variability in space and time

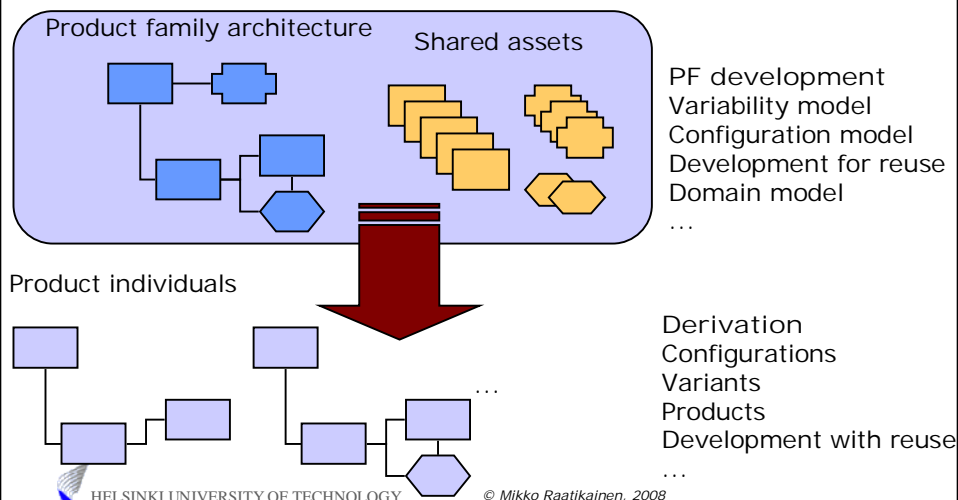


The idea of variability

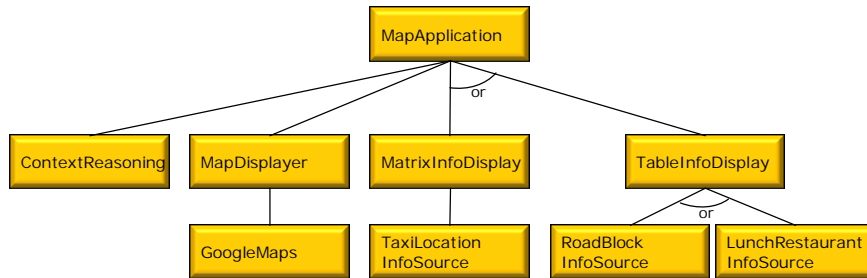
- A definition [Svahnberg, 2005]
 - Variability is the ability of a system to be efficiently extended, changed, customized or configured for use in a particular context
- Different reasons
 - multiple user segments, such as home and professional users
 - allow price categorization from low to high-end products
 - support various hardware platforms and operating systems
 - cultural variability, such as legislation, language, and market structures in different countries
- A trend: increase variability in software



Software product family



An example scenario – Map application



- Informal variability/domain/product family/line model consisting of 9 elements, relations between the elements, and rules for relations
- What are the correct variants?



Topic



Motivation - anecdotes

- A customer called helpdesk to fix a feature that was not operating correctly in her product
 - Helpdesk spent time to find the solution with engineers to find out that the customer had not bought the feature to her product
- A Scrum product owner needs to decide what features to deliver next
 - How to re-assign teams and convince them rationality of the decisions



Motivation

- Variability becomes complex due to exponential number of potential configurations and constraints
 - cf. e.g. packages in a Linux distribution: How many different combinations are possible, and how many of them are "correct"?
- Software engineering rarely takes variability into account
 - Rather, focus on a single product or project
 - Hence inadequate methodology per se for variability
 - Extensions, however, could be specified for variability



The problem

- Few existing variability modeling constructs
 - Feature models, Orthogonal variability modeling [Pohl 2005], ConIPF [Hotz 2006], Kumbang [Asikainen 2007] etc.
- However, variability modeling, e.g.,
 - Focus on concepts and constructs lacking a link to practical methodological support
 - Variability is an add-on or extension to existing models rather than studying stakeholder's needs for variability information
- Software variability research is still at its early stage in particularly from the practical application point of view



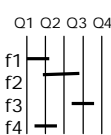
Objective

- A practical method for managing variability
 - The method adheres to the view based architecture description practices
 - Takes into account different stakeholders
 - Documentation / description and communication needs with respect to variability
 - Utilizes existing variability modeling constructs



Variability description – an example


- Stakeholders such as marketing, engineers, operators etc. share the concern of provided functionality
 - The same feature model can serve them
 - Their expectation of the representation, however, differ



Q1 Q2 Q3 Q4

f1
f2
f3
f4

Feature roadmap



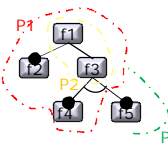
Product1

- f1
- f2
- f3
- f4

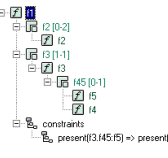
A feature list for a product

	P 1	P 2	P 3
f1	+	+	+
f2	+		+
f3	+	+	+
f4	+		
f5			+

A product – feature matrix



Informal feature tree




A more rigorous feature tree

```

feature type f1 {
  subfeatures
  f2 f2[0-2];
  f3 f3;
  constraints
  present(f3.f45.f5)
  =>
  present(f2);
}

feature type f3 {
  subfeatures
  (f5, f4) f45[0-1];
}
                    
```

Textual feature model (partial)



HELSINKI UNIVERSITY OF TECHNOLOGY

© Mikko Raatikainen, 2008

Method

Method

- Design science approach [March 1995] to develop a method
 - Case study research to capture descriptive account for background
 - Action research for evaluation



Our recent work

- Variability modeling tool suite called KumbangTools
 - GPL release available
 - SPLC'07, SVM-WS'07
- Variability modeling with QAs (especially security)
 - QoSA'07, EUROMICRO'07, Vamos 2007, Vamos 2008
- Architecture, variability and agile methods
 - Software architecture design in XP: CEE-SET 2007
 - Combine backlog and variability management: ICSE'08-SDG
- Combine variability modeling and modeling in SOA
 - SOAPL-WS in SPLC'07, SOSE'07
- (Theoretical basis
 - Mainly by Timo Asikainen and Tomi



References

- Svahnberg, M.; van Gurp, J. & Bosch, J. A taxonomy of variability realization techniques *Software - Practice and Experience*, John Wiley & Sons, Inc., 2005, 35, 705-754
- Hotz, L.; Wolter, K.; Krebs, T.; Deelstra, S.; Sinnema, M.; Nijhuis, J. & MacGregor, J. *Configuration in Industrial Product Families* IOS Press, 2006
- Pohl, K.; Böckle, G. & van der Linden, F. *Software Product Line Engineering: Foundations, Principles, and Techniques* Springer, 2005
- Asikainen, T.; Männistö, T. & Soininen, T. Kumbang: A domain ontology for modelling variability in software product families *Advanced Engineering Informatics*, 2007, 21, 23-40
- Bass, L.; Clements, P. & Kazman, R. *Software Architecture in Practice* Addison-Wesley, 2003
- March, S. T. & Smith, G. F. Design and Natural Science Research on Information Technology *Decision Support Systems*, 1995, 15, 251-266
- IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems -Description 2000

