

Specifying Product Line Variability in Requirements Models

Cristian Anastasiu

Samuel Liniger

Fabian Hofstetter

Where does it fit?

Software Product Lines

**Feature-oriented
Domain Analysis**

Software Product Lines

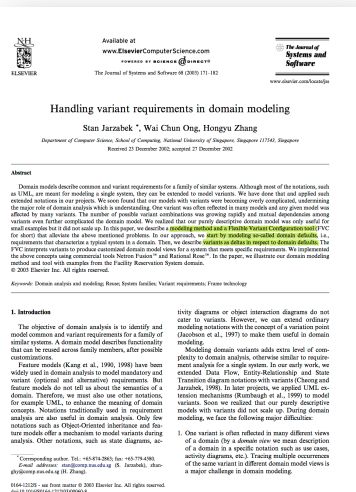
**Specifying Product Line
Variability in
Requirements Models**

Software Product Lines

**Product Derivation
with Requirements
Models**

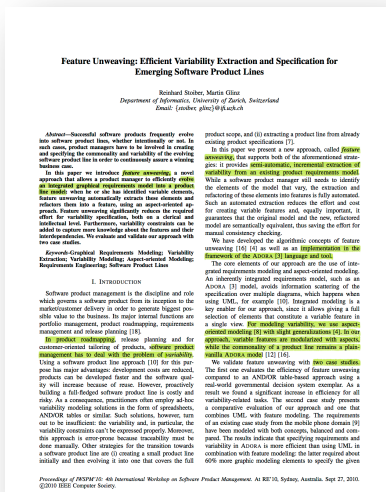
Literature

Handling variant requirements in domain modeling



[Stan Jarzabek *, Wai Chun Ong, Hongyu Zhang 2003]

Feature Unweaving: Efficient Variability Extraction and Specification for Emerging Software Product Lines



[Stoiber, Glinz 2010]

Agenda

- Product Line Variability
- FVC
- Feature Unveawing
- Observations

Product Line Variability

- Product Line Variability describes the variation (differences) between the applications that belong to a product line.
- In product roadmapping, release planning and for customer-oriented tailoring of products, software product management has to deal with *variability*.
- How do we model variability?

Modelling Challenges

- Different approaches used today to model variability
 - Feature Diagrams + UML extensions (variation points)
 - AND/OR Tables

- Models tend to become overcomplicated, do not scale
- Traceability has to be done manually
- **AND/OR Tables** are error-prone, no constraints between variable features

Proposed methods

FVC

(Flexible Variant
Configuraiton)

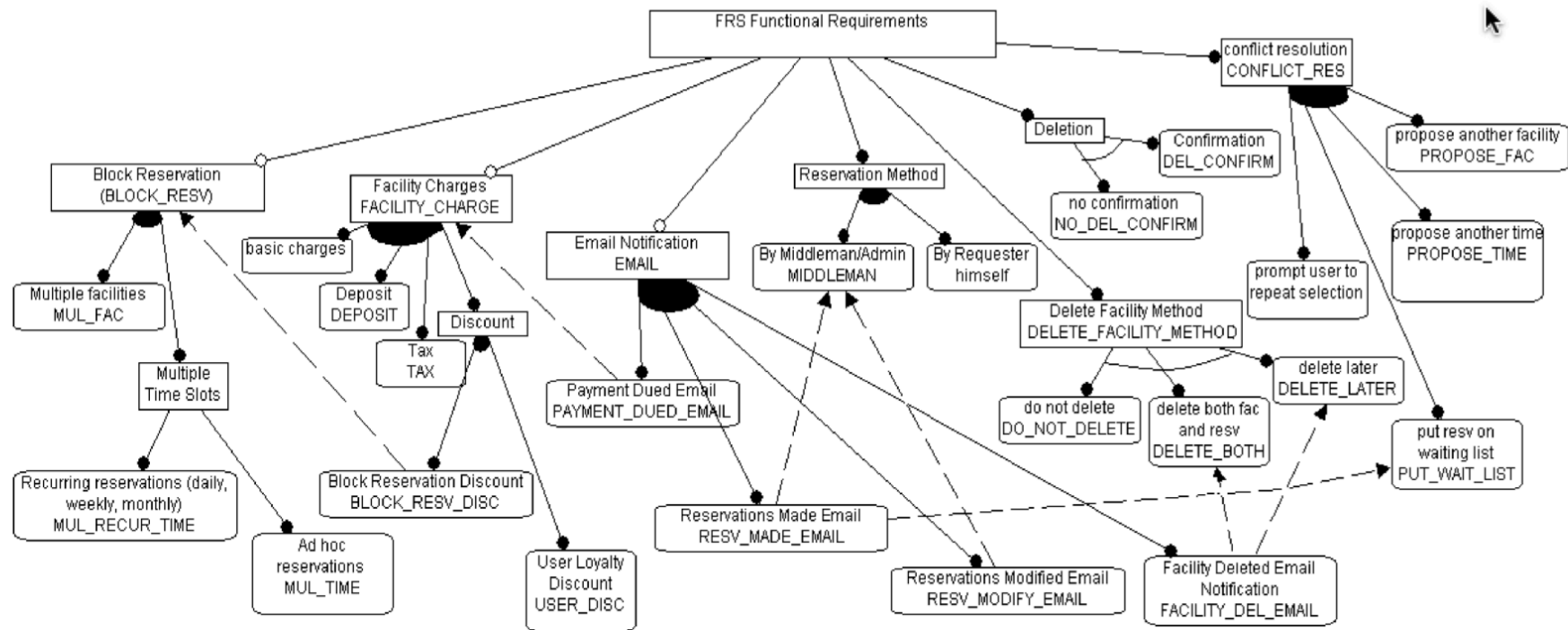
**Feature
unweaving**

Flexible Variant Configuration

- FVC uses frame technology¹ and Fusione toolset by Netron Inc.
- Main Steps:
 1. Start by modelling **domain defaults**
 2. Define variants as deltas from default requirements as **customization scripts**
 3. Use **FVC** tool to interpret customization script, and based on the selected variant, output the customized view

¹ [http://en.wikipedia.org/wiki/Frame_technology_\(software_engineering\)](http://en.wikipedia.org/wiki/Frame_technology_(software_engineering))

FVC



FVC

F-DEFAULT Make_Reservation_uc // suffix 'uc' indicates that the contents of this f-default is a use case

DEFAULT Make_Reservation_uc // suffix 'uc' indicates that the contents of this-default is a use case

1. Introduction

Make Reservation allows a user to select "FACILITY" and make new reservation.

2. Flow of events

2.1 The user is identified.

2.2 The system shows existing "FACILITY"s.

2.3 The user selects a "FACILITY" and supplies the time period of reservation.

2.4 If the "FACILITY" is reserved at the specified time, a warning message is prompted and user repeats step 2.2. Otherwise, the system confirms the new reservation.

3. Special Requirement

System should respond to user input within "RESPONSETIME".

Insert
Variation



Insert
Variation



Domain Defaults

FVC – Customization Scripts

```
CS MIDDLEMAN // Customization Script for MIDDLEMAN variant
```

```
// customize Make_Reservation_uc
```

```
.COPY Make_Reservation_uc
```

```
.INSERT~AFTER MIDDLEMAN
```

```
If the user is a middleman, then enter the requester's information.
```

```
.END-COPY Make_Reservation_uc
```

```
...
```

```
// customize Retrieve_Reservation_uc
```

```
.COPY Retrieve_Reservation_uc
```

```
.INSERT~AFTER MIDDLEMAN
```

```
If the user is a middle man, then when searching/retrieving reservations by requester, the user enters userid of the requester on behalf of the requester
```

```
.END-COPY Retrieve_Reservation_uc
```

```
...
```

FVC



Make Reservation use case with MIDDLEMAN variant:

1. Introduction

Allows a user to select **meeting room** and make new reservation.

2. Flow of events

2.1 The user is identified.

If the user is a middleman, then enter the requester's information.

2.2 The system shows existing **meeting rooms**.

2.3 The user selects a **meeting room** and supplies the time period of reservation.

2.4 If the **meeting room** is reserved at the specified time, a warning message is prompted, and user repeats step 2.2. Otherwise, the system confirms the new reservation.

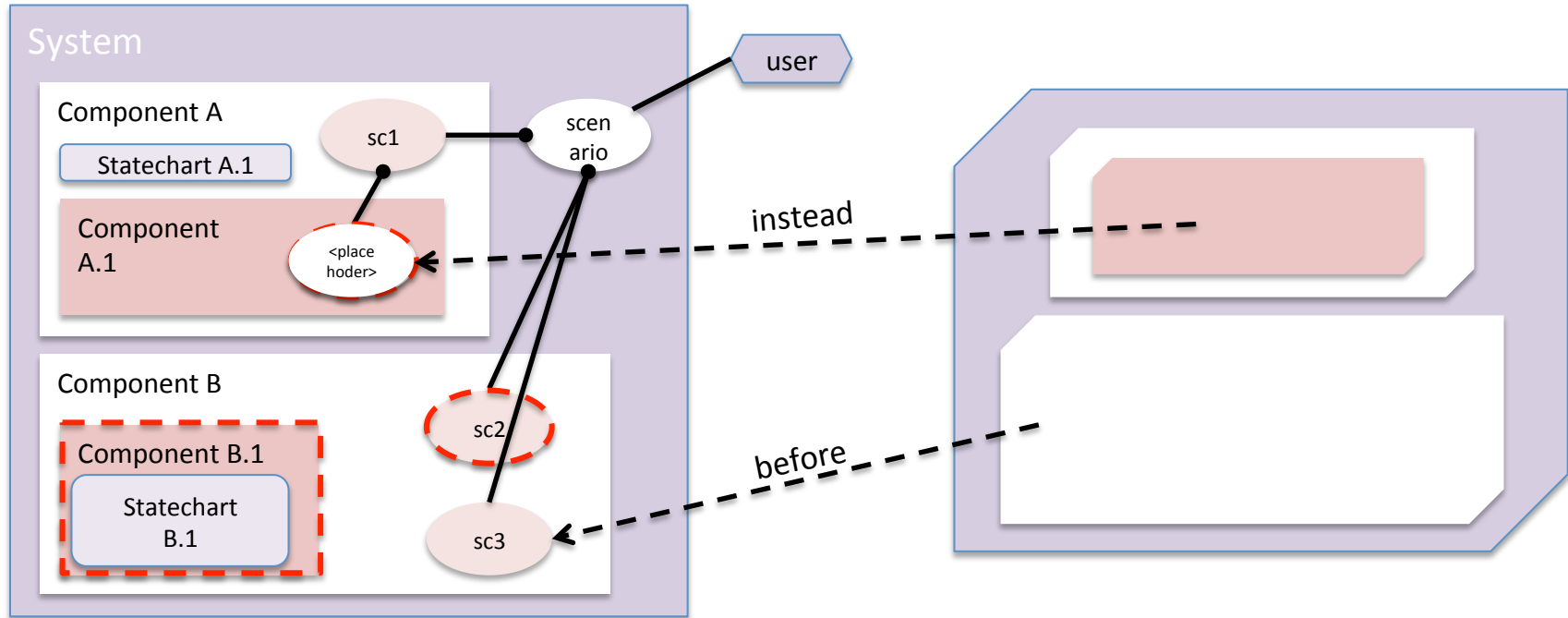
3. Special Requirement

System should respond to user input within **30secs**.

Feature Unweaving

- Based on the **ADORA** language and tool
- For modelling variability, it is using aspect oriented modelling
- Provides a **semi-automatic incremental extraction** of variability from an existing product requirements model

Feature Unweaving



Feature Unweaving Comparison

Two case studies where ADORA was compared with:

1. AND/OR Tables

- Feature Unweaving required **3 x less manipulations for specifying variability, deriving product variants and for propagating requirement changes**
- Feature unweaving simplified planning for new products and increased the quality of such planning

2. Sparx-Systems Enterprise Architect (EA) in combination with pure-systems pure::variants

- EA required 61.5% more elements for the requirements specification and 60% more elements for the variability modelling

Summary

Pros

Cons

FVC

- Reduces domain model complexity
- In theory could be used with different models

- As it is based on the frame technology, works with text files -> models need to be converted in text readable format (eg XML)
- Requires manual effort for the creation of domain defaults and scripts
- Does not scale with large number of variant interdependencies (?)
- Only in prototype phase

Feature Unweaving

- Comprehensive model overview
- Reduced modelling effort
- Simplifies planning for new products and increase the quality of such planning

- It is bounded to ADORA, cannot work with standards such as UML
- Need for many pseudo-elements (aspects)
- Model layout is changed by the addition of aspects or abstract objects

Discussion

What are the main issues you see when trying to manage variability?

Q & A