

Modelling variability in Software Product Lines

Outline

- Introduction
- A Meta-Model of Variability
- Instrumentations of models for enabling variability
- Conclusion

Introduction

Context: Product Lines

- Reuse between product lines is necessary [Van Ommering 2002]
 - Products share commonalities and variability
 - Products of one product line
 - Products across different product lines
- Explicit documentation of variability needed

Variability in Product Lines

- “Variability deals with differences between products of a product line.” [Tommy Myllymäki, University of Tampere, 2001]
- Use a meta-model for structuring variability information across product lines

A Meta-Model of Variability

Meta-Model of Variability

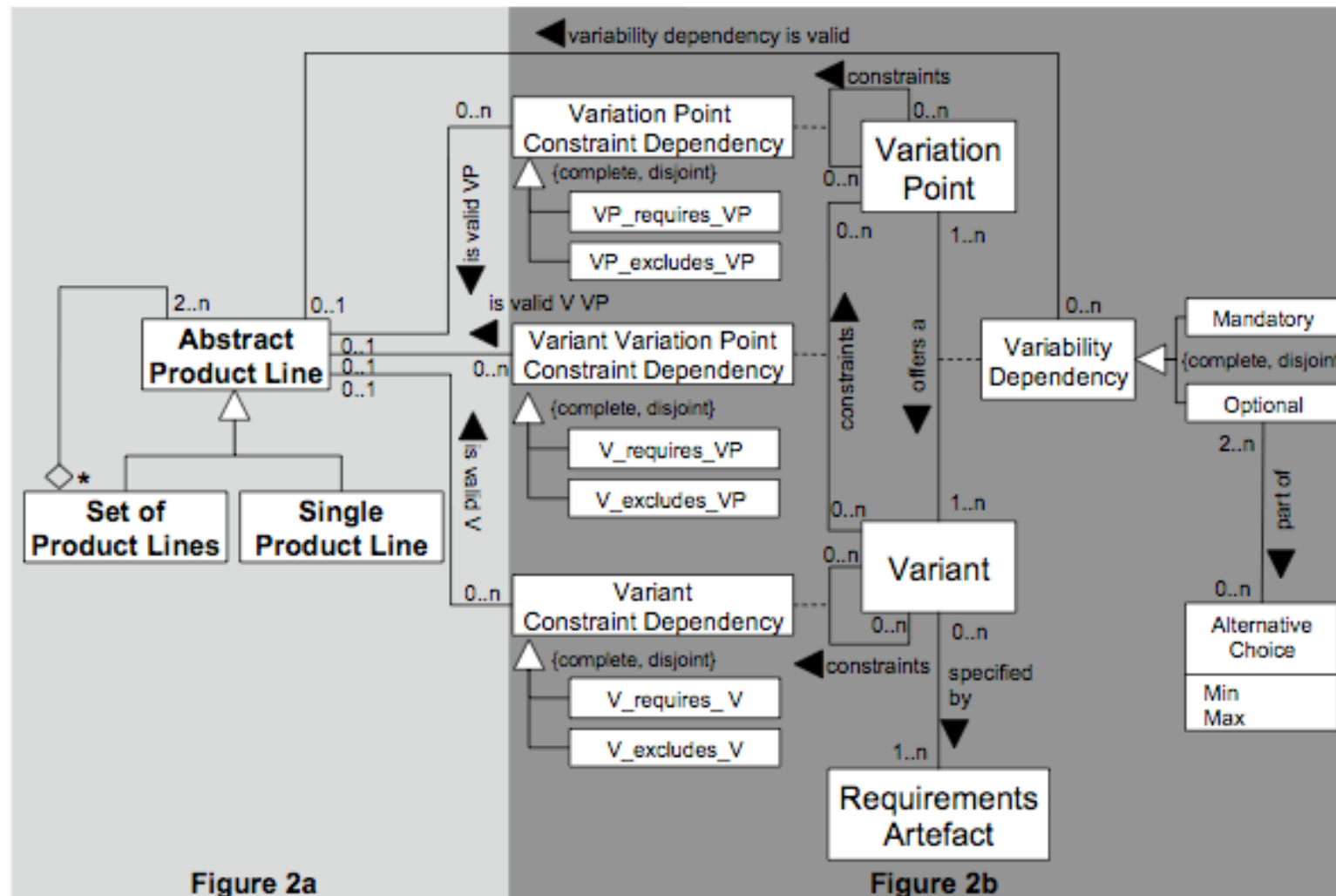


Figure 2: Extended variability meta model²

[Bühne et al. 2005]

Example of the Meta-Model

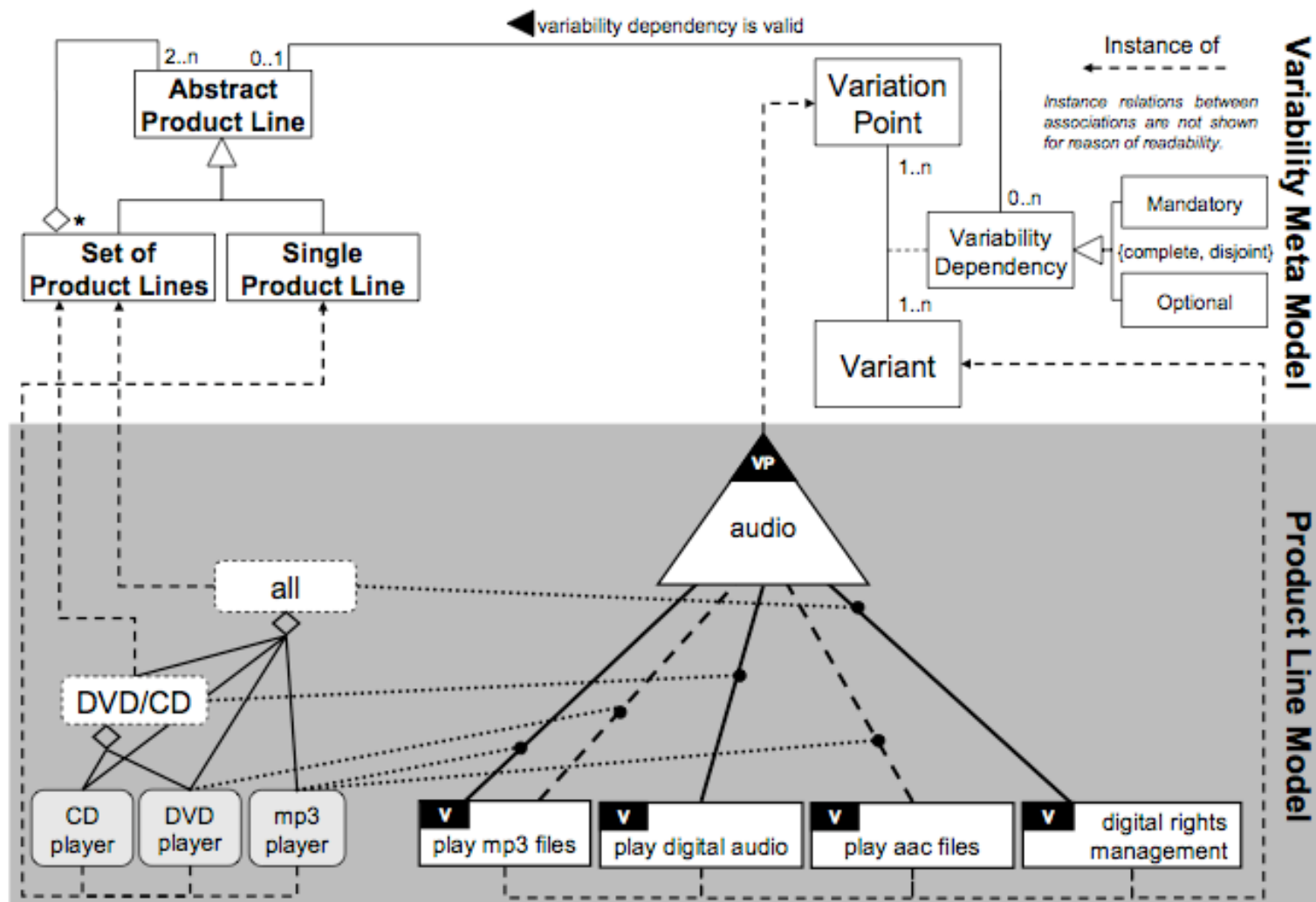


Figure 3: Example – instantiated meta model

[Bühne et al. 2005]

Instrumentation of models for enabling variability

Feature Tree

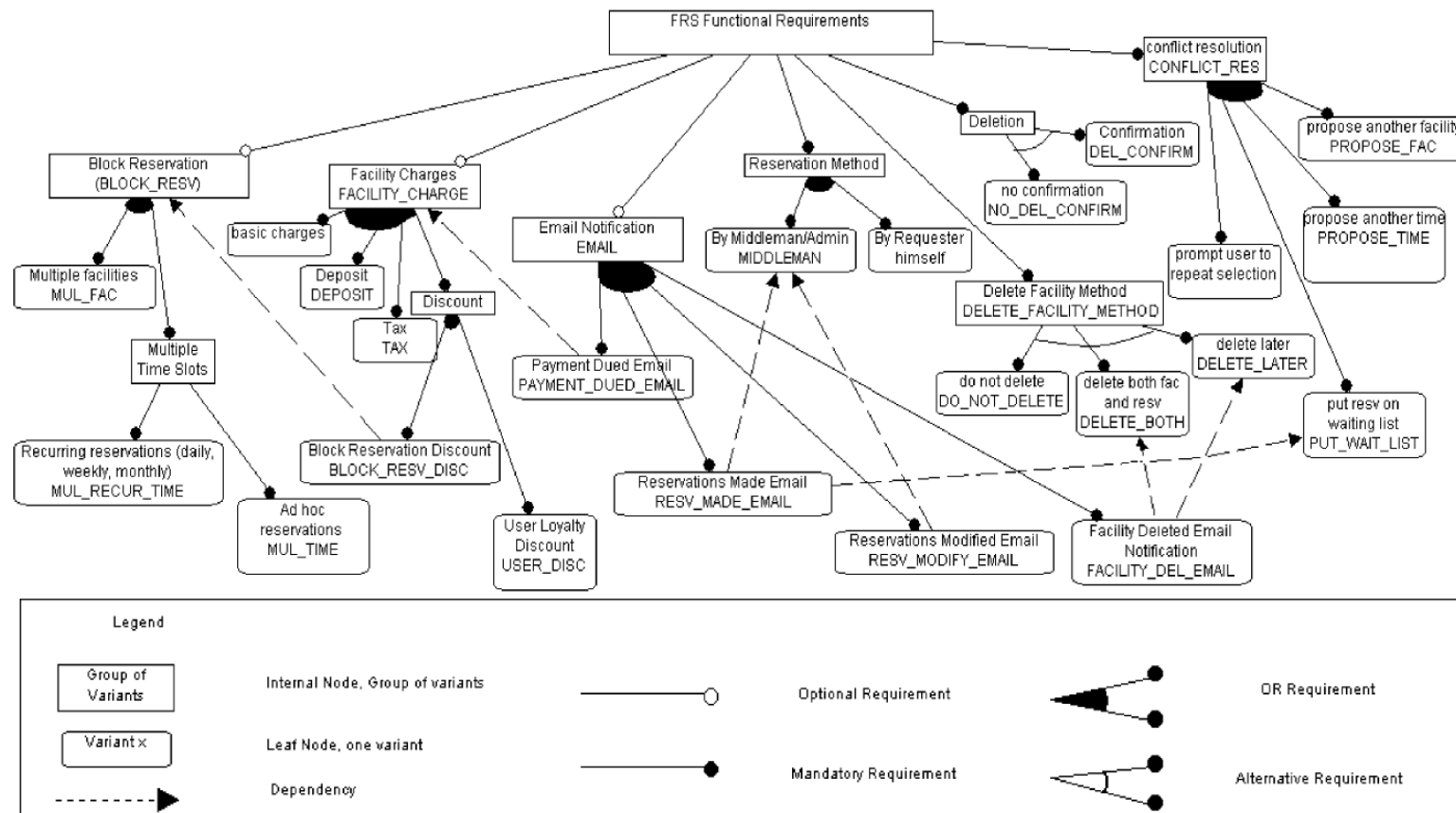


Fig. 3. FRS feature diagram with dependencies.

Flexible use cases 1 / 2

DEFAULT Make_Reservation_uc // suffix 'uc' indicates that the contents of this f-default is a use case
1. Introduction
<i>Make Reservation</i> 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".

Fig. 4. Default use case *Make Reservation*.

F-DEFAULT Make_Reservation_uc // suffix 'uc' indicates that the contents of this f-default is a use case
.REPLACE FACILITY BY "meeting room"
.REPLACE RESPONSETIME BY "30 secs"
1. Introduction
Allows a user to select FACILITY and make new reservation.
2. Flow of events
2.1 The user is identified.
.BREAK MIDDLEMAN //a breakpoint at which variant " reservation by middlemen" can be addressed
2.2 The system shows existing "FACILITY"s.
2.3 The user selects a "FACILITY" and supplies the time period of reservation.
.BREAK BLOCK_RESV_FAC //a breakpoint at which variant "select multiple facilities" can be addressed
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".

Fig. 5. f-default use case *Make Reservation*.

Flexible use cases 2 / 2

```
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

...
```

Fig. 6. A partial customization script for the MIDDLEMAN variant.

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.

Fig. 7. A customized *Make Reservation* use case.

Flexible workflows 1 / 2

- Using the same concepts from the use cases
- Store UML Diagrams as XML
- Insert the syntax there

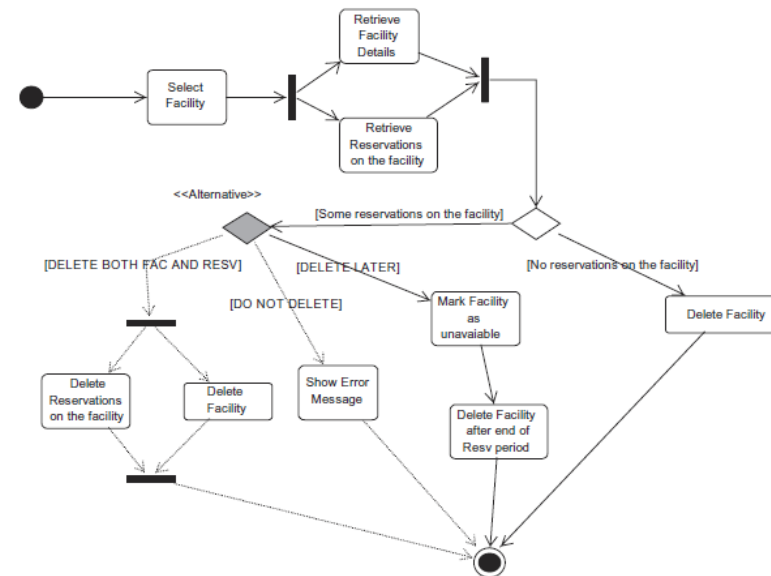


Fig. 8. Delete Facility workflow with variation point.

Flexible workflows 2 / 2

```
.SELECT ;DELETE_FACILITY_METHOD
.WHEN = "DELETE_LATER"
<Behavioral_Elements.State_Machines.ActionState xmi.id = 'G.100'>
  <Foundation.Core.ModelElement.name> Mark Facility as unavailable </Foundation.Core.ModelElement.name>
  ...
</Behavioral_Elements.State_Machines.ActionState>
<Behavioral_Elements.State_Machines.ActionState xmi.id = 'G.101'>
  <Foundation.Core.ModelElement.name> Delete facility after end of resv period </Foundation.Core.ModelElement.name>
  ...
</Behavioral_Elements.State_Machines.ActionState>
.WHEN = "DO_NOT_DELETE"
<Behavioral_Elements.State_Machines.ActionState xmi.id = 'G.100'>
  <Foundation.Core.ModelElement.name> Show Error Message </Foundation.Core.ModelElement.name>
</Behavioral_Elements.State_Machines.ActionState>
```

```
CS DELETE_FACILITY_METHOD // Customization Script for DELETE_FACILITY_METHOD variant
...
//Customize Delete Facility workflow
.COPY Delete_Facility_wf
.REPLACE DELETE_FACILITY_METHOD BY "DO_NOT_DELETE"
.END-COPY Delete_Facility_wf
...
```

Fig. 10. A partial customization script for the DELETE_FACILITY_METHOD variant.

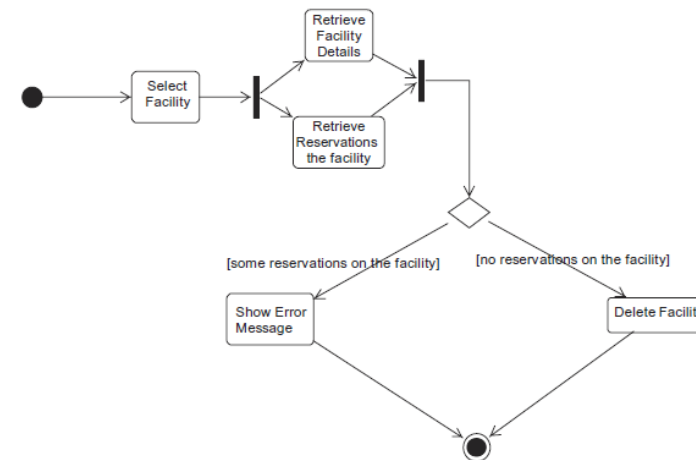


Fig. 11. A customized Delete Facility workflow.

Flexible Methods

ADVANTAGES

- Enhancement of an industry standard
- Small syntax
- Product line support added

DISADVANTAGES

- Standard does not support it
- Tools are required
- Requirements Engineer has to learn the syntax

Conclusion

- UML does not support product lines
- „Tools play the key role when installing new approach“ (Beuche 2007)
- „Flexible Methods“ are an elegant way

Additional References

Myllymäki, T. (2001). Variability Management in Software Product Lines. Technical Report. Tampere University of Technology. December 2001.

...thanks for your attention!