On the Interdependence and Integration of Variability and Architectural Decisions

Ioanna Lytra, Holger Eichelberger, Huy Tran, Georg Leyh, Klaus Schmid, Uwe Zdun

Software Architecture Research Group
University of Vienna, AT

Software Systems Engineering
University of Hildesheim, DE

Siemens AG
Erlangen, DE
Variability Decision

- ...is a decision related to the differences among the products that derive from a product line.

- From an architectural perspective, variabilities may reflect different architectural options considered during the design of the product line that are independent of the products’ features.
… is the result of the evaluation of alternative design options in terms of architectural elements such as patterns, components, or connectors and the selection of the best-fitting solution.

Architectural decisions refer to the software design both at product line and product level.

In the context of historian access to archive, ... facing data privacy regulations, ... we decided to encrypt historian database content ... to achieve confidentiality.
Variability decisions often overlap with or influence architectural decisions (e.g., resolving a variability may enable or prevent some architectural options).

The interdependence and integration of variability and architectural decisions have neither been studied nor addressed in a systematic way, yet.

This work intends to fill this gap and proposes the systematic integration of the two types of decisions along with tool support.
Motivating Case

Warehouse Management System

Place Order → Stock Determination → Transport

Picking → Packing → Shipping

Goods out process of a warehouse
# Architectural Decisions and Variability Decisions

<table>
<thead>
<tr>
<th>Variability Decision</th>
<th>Possible values</th>
<th>Binding Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Picking rate</td>
<td>High/Medium/Low</td>
<td>Design Time</td>
</tr>
<tr>
<td>Partial pallet strategy</td>
<td>Highspeed/Optimal reduction</td>
<td>Runtime</td>
</tr>
<tr>
<td>Stapler crane strategy</td>
<td>Single fork/Multiple forks</td>
<td>Design Time</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Architectural Decision</th>
<th>Options</th>
<th>Influenced by Variability Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPC type</td>
<td>Open source</td>
<td>Picking rate - Low</td>
</tr>
<tr>
<td></td>
<td>Medium Price</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Very Expensive</td>
<td>Picking rate - Medium/High</td>
</tr>
<tr>
<td>Deployment devices</td>
<td>Single server</td>
<td>Picking rate - Low</td>
</tr>
<tr>
<td></td>
<td>Multiple servers/round-robin</td>
<td>Picking rate - Medium</td>
</tr>
<tr>
<td></td>
<td>Multiple servers/load monitoring</td>
<td>Picking rate - High</td>
</tr>
<tr>
<td>Service identification</td>
<td>Business delegate proxy</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Business delegate adapter</td>
<td>-</td>
</tr>
</tbody>
</table>
...and their Interdependency

- **Low picking rate** implies *IPC open source* and *Single server*
- **Medium picking rate** implies *IPC very expensive* and *Multiple server with round-robin*
- **High picking rate** implies *IPC very expensive* and *Multiple server with load monitoring*

➤ *How to express and resolve these interdependencies systematically?*
**Proposed Approach**

1. Identify Variants
2. Define Architectural Decisions
3. Model Variability Decisions
4. Create Architectural Decision Model
5. Define Mapping
6. Product Line Level Architectural Decisions
7. Design Reference Architecture
8. Resolve Variability Decisions
9. Give Feedback
10. Product Level Architectural Decisions
11. Further Product Level Architectural Decisions
12. Design Product Architecture
Tool Support: EasyProducer

Product Configuration Editor: PL_WMS

<table>
<thead>
<tr>
<th>Decision Name</th>
<th>Current value</th>
<th>Freeze</th>
</tr>
</thead>
<tbody>
<tr>
<td>VP1</td>
<td>medium</td>
<td>freeze</td>
</tr>
<tr>
<td>VP2</td>
<td></td>
<td>freeze</td>
</tr>
<tr>
<td>VP3</td>
<td>single</td>
<td>freeze</td>
</tr>
<tr>
<td>VP4</td>
<td>computer</td>
<td>freeze</td>
</tr>
</tbody>
</table>

```java
project PL_WMS {
    version v0;
    enum PickingRateType {high, medium, low};
    enum PartialPalletStrategyType {highSpeed, optimalReduction};
    enum StaplerCraneStrategyType {single, multiple};
    enum UIDeviceType {computer, mobile};
    PickingRateType VP1;
    PartialPalletStrategyType VP2;
    StaplerCraneStrategyType VP3;
    UIDeviceType VP4;
    enum BindingTime {designTime, compileTime, initTime, runTime};
    attribute BindingTime bindingTime = BindingTime.designTime to PL_WMS;
    assign (bindingTime = BindingTime.runTime) to {
        PartialPalletStrategyType VP2;
    }
}
```
Tool Support: ADvISE

ADvISE - Interface Design - Eclipse

AD Questionnaire

- **AD1**
  Component-and-Connector: Decide how to design interfaces inside the VSP to
  - **Decoupling**
    Do you want to decouple operator application from the contract responsibility?
    - No
    - Yes
  - **Unified Interface**
    Do you need a unified interface for services inside the VSP?
    - No
    - Yes
  - **Hide complexity**
    Is the aim of the unified interface to hide interface or technological complexity?
    - Interface complexity
    - Technological complexity

- **AD2**
  Component-and-Connector: Decide the gateway details.
  - **Convert Interfaces**
Mapping between Variability and Architectural Decisions

1. `<mappings>
   ...
   <vp name="VP1">
     <relation type="excludes">
       <vd id="PickingRateType.medium"/>
       <add id="AD4.IPC type of software.IPC open source"/>
     </relation>
   </vp>
   ...
</mappings>

2. VP1 = PickingRateType.medium
   VP3 = StaplerCraneStrategyType.single

3. Product Configuration with EasyProducer

Updated Architectural Decision Model with ADvISE
Conclusions and Future Work

- Interdependencies between variability and architectural decisions exist but are mainly kept implicit.
- Formalizing the interdependencies requires additional efforts but leads to better automated support in integrating variability management and architectural decision making in the long run.
- Variability and architectural decisions remain consistent at product derivation.

Some open challenges
- Reconsidering a variability decision may cause inconsistencies to existing architectural decisions. How to deal with this?
- What other interdependencies exist? How to classify, formalize them?
- What is the impact of changing both decisions during the evolution and maintenance of product lines and products?