Semantic Variability Modeling for Multi-staged Service Composition

Bardia Mohabbati\textsuperscript{1}, Nima Kaviani\textsuperscript{2}, Dragan Gašević\textsuperscript{3}

\textsuperscript{1}Simon Fraser University, \textsuperscript{2}University of British Columbia, \textsuperscript{3}Athabasca University, Canada

mohabbati@sfu.ca, nimak@ece.ubc.ca, dgaseavic@acm.org
Outline

• Objectives and Introduction
• Motivating Scenario
• Overview
  • Analysis and Design
  • Feature Model and Variability Modeling
  • Service Requirements Specification
  • Software Service Products Configuration
  • Service Discovery and Specialization
• Semantic Variability Modeling
  • Feature Model Ontology
  • Feature Model Annotation
  • Feature Model Specialization
• Conclusion

9/23/2009
Objectives

- Adaptive development of software-intensive systems for mobile and ubiquitous Computing

Pervasive/Ubiquitous computing (ubicomp) is about providing services and computing capabilities in heterogeneous environments

- Heterogeneous computing device
- Resource constraints of deployment platform

- Utmost functionality and satisfying non-functionality requirements (NFRs) in final product

- Service-Oriented Software Development and SPL
Objectives

We exclusively focus variability coming from the different delivery platform of service oriented architecture.
Motivating Scenario

Airport Large Screen Display

Ubiquitous Information Service

Mobile Devices

Service Provider

Remote Service (Web services)
Overview

Service Oriented Product Configuration

- Feature Model Specialization
- Configuration Validation
- Annotated Feature Model

- Software Service Product Specialization
- Feature Model Specialization

- Device Specification
- Request Specification

- User’s Requests and Preferences

Service Adaptation

- Domain Analysis
  - Product Family
  - Feature Extraction

- Representing Variability by Feature Modeling
- Feature Model Ontology
- Feature Model Annotation
Feature Model & Variability Modeling

**Functional Requirements**
- Core Services in the system
- Carry the objectives of the system

**Non-functional Requirements**
- Replaceable or alterable services
- Quality of Service (QoS) requirements
- Security requirements
Ontology is defined as a **formal specification of a conceptualization** and utilizes the representation of knowledge contained in feature models.

**Semantic representation of features:**
- Relations and dependencies
- Feature constrains

**Transformation:**
- Using the Web Ontology Language (OWL)
- Constructing mutual disjoint classes

\[
\begin{align*}
F_i & \sqsubseteq T \\
FM_i Rule & \sqsubseteq T \\
hasF_i & \sqsubseteq ObjectProperty \\
FM_i Rule & \equiv \exists hasF_i, F_i \\
F_i & \sqsubseteq \neg F_j, \text{ for } i \leq 1, j \leq n \land i \neq j
\end{align*}
\]
Feature Model Ontology enables us to take advantages of ontology annotation capabilities in order to enrich the definition of domain assets with the constraints concerning the non-functional requirements of a domain.

\[ AN \sqsubseteq \text{ObjectProperty} \]
\[ AN \sqsubseteq FM_m, \text{domain}(FM_m) \]
\[ \top \sqsubseteq \forall AN.DC_n, \text{range} (DC_n) \]

\[ \text{hasBluetooth}_AN \sqsubseteq \text{ObjectProperty} \]
\[ \text{hasBluetooth}_AN \sqsubseteq \text{Bluetooth}, \text{domain}(\text{BluetoothRule}) \]
\[ \top \sqsubseteq \forall \text{has Bluetooth}_AN.\text{BluetoothDevice}, \]
Software Service Products Configuration

- Device Ontology model reflect device capability (NFRs)
- NFRs as Integrity Constraints (IC)- which include and exclude the selection of services in annotated feature model

- Feature Model Specialization: Stage refinement process which constitute staged configuration
  - Specialization: The features don’t satisfy the NFRs of target deployment platform are discarded and pruning feature model
  - Ontology-based reasoning: consistency checking and verification
    - If instances of NFRs ontology are consistent with respect to the annotation properties defined in feature model
## Software Service Products Configuration

<table>
<thead>
<tr>
<th>Design Time</th>
<th>Run Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Creating the feature model</td>
<td>4. Platform Deployment</td>
</tr>
<tr>
<td>2. Incorporating NFRs into designing the feature model</td>
<td>Capability analysis</td>
</tr>
<tr>
<td>3. Binding Services to the NFR ontology</td>
<td>5. Feature selection</td>
</tr>
<tr>
<td></td>
<td>6. Service composition</td>
</tr>
<tr>
<td></td>
<td>7. Validation and Deployment</td>
</tr>
</tbody>
</table>

### Diagram

1. Feature Model
2. Device Ontology (Description Repository)
3. Feature Pruning
4. Product Configuration
5. Product Instance
6. Service Delivery
7. Configuration Validation Check
DEFINITION 1 Let $d \in \mathcal{DCO}$ be an instance of a device which has $n$ capabilities. Each capability for device $d$ is an instance $(i_k)$ of a concept $(C_k)$ from $\mathcal{DCO}$ such that $A \models C_k(i_k) \ (1 \leq k \leq n)$. $S_{dc} = \{C_1, \ldots, C_n\}$, represents the set of all concepts $C_k$ from $\mathcal{DCO}$ that $d$ supports.

DEFINITION 2 Let $S_{AF} \subseteq \mathcal{FM}\mathcal{O}$ be a set consisting of concepts $CF_i$ such that $\forall CF_i. \top \mid \exists AN_i \subseteq CF_i \sqcap \exists AN_i. C_k$ where $C_k \in \mathcal{DCO}$. 
AN_j \subseteq ObjectProperty

\[ SFA = \bigcup_{i=1}^{n} CF_i \]

for each \( C_k \in S_{dc} \)
for each \( CF_i \in S_{AF} \)
\[
\begin{align*}
& \text{if } (\exists AN_j. C_j \sqcap AN_j \subseteq CF_i \text{ and } C_j \in DCO) \\
& \quad \text{if } (C_j \in S_{dc}) \\
& \quad \quad \text{if } (A \neq CF_i(i)) \\
& \quad \quad \quad SFA \leftarrow SFA - \{CF_i(i)\} \\
& \quad \text{else} \\
& \quad \quad SFA \leftarrow SFA - \{CF_i(i)\} \\
& \text{return } (SFA)
\end{align*}
\]
During the process of the configuration validation, model constraints are checked against the derived configuration to provide proper assurance in terms of correct exclusion and inclusion of optional and mandatory features.

- **OWL-DL reasoning engines**: To support automated class subsumption, consistency reasoning and detection of possible inconsistencies in the specialized feature.

- **Consistency and conflicts detection**: Formulate and define a list of Semantic Web Rule Language (SWRL) rules to represent all invalid states
Service Discovery and Specialization

The result of consistency checking in the previous step can be two folded:
- Inconsistent ontology  \(\rightarrow\) Design Stage - family composition refinement
- Consistent ontology  \(\rightarrow\) Service Specialization based on soft constraints

Deployment of the service-oriented system:
- Web Service Modeling Ontology framework (WSMO)
- Transformation feature model to WSMO service (with ATL transformation language)
- Web service Execution Environment (WSMX) (the reference implementation for WSMO)
**Conclusion**

- **Ontologies as an underlying formalism for representation of feature models to enrich Feature Model with semantics**

- **Ontology-based approach for feature model annotation with NFRs ontologies**

- **Inference and Ontological Reasoning Over Feature Model**
  - validating (non-) functional requirements
  - Semi-automatic Product configuration
  - Product consistency check
Thank you 😊