Model-Driven Engineering of Information Integration Systems

Susanne Busse
Information Engineering Group
Jönköping
March 2007

Outline
• Information Integration Systems (IIS)
• Model-Driven Software Engineering
• Model-Driven Engineering of IIS
  • Computational-Independent Design of an IIS: Reference Architecture
  • Platform-Independent Design of an IIS: Vocabulary Definitions and Integration Algorithms
• Conclusions

Information Integration Systems (IIS)

Engineering of an IIS

Computational-Independent Design
- Which reference architecture fits the requirements?
- How do we ‘apply’ the reference architecture?

Platform-Independent Design
- Which data models / languages / algorithms are appropriate?
- How to integrate them into the design?

Platform-Specific Design
- Which middleware technology is appropriate?
- How is the platform-independent model realized with this technology?
Model-Driven Software Engineering

- Requirements
- Model 1
- ...
- Model n
- Code
- Running System

- Well-defined engineering process
- Traceability
- Consistent evolution
- Documentation
- Reusability

Software Engineering with Variants

- Model i
  - IIS Design
- Variants: Feature Model
  - Platform Requirements
- Transformation
  - Rules
- Model i+1
  - SOA-based IIS
  - J2EE-based IIS

Parameterized Model Transformation

- Define model i
- Select feature model
- Select features of the feature model
- Transformation
- Generated model i+1

Parameterized Model Transformation – Example

- Selected Features
- PIM
  - UML
- PSM
  - UML
Software Engineering with Variants

Model i
IIS Design

Transformation
Rules

Model i+1
SOA-based IIS

Model i+1
J2EE-based IIS

Variants: Feature Model
Platform Requirements

What

How

Outline

• Information Integration Systems (IIS)
• Model-Driven Software Engineering
• Model-Driven Engineering of IIS
  • Computational-Independent Design of an IIS: Reference Architecture
  • Platform-Independent Design of an IIS: Vocabulary Definitions and Integration Algorithms
• Conclusions

Design of IIS – Feature Model

Design: Selecting a Reference Architecture

Variants IIS

Variants MBIS

Variants IR-IIS
Matching Algorithm

Matching of Feature Nodes

<table>
<thead>
<tr>
<th>Node-match</th>
<th>Bound</th>
<th>Common</th>
<th>Undecided</th>
<th>Uncommon</th>
<th>Removed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bound</td>
<td>+1</td>
<td>+0.5</td>
<td>-0.5</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>Removed</td>
<td>-1</td>
<td>-0.5</td>
<td>+0.5</td>
<td>+1</td>
<td></td>
</tr>
<tr>
<td>Undecided</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>±</td>
</tr>
</tbody>
</table>

Similarity of Feature Trees

\[
\text{match}(f_{F1}, f_{F2}) = \text{node-match}(f_{F1}, f_{F2}) \cdot \frac{1}{n} \cdot \text{match}(f_i_{F1}, f_i_{F2})
\]

- Subfeatures have the same priority
- Details of undecided features are not considered
- Details of removed features are not considered

Design: Selecting a Reference Architecture

Design based on Reference Architectures

Outline

- Information Integration Systems (IIS)
- Model-Driven Software Engineering
- Model-Driven Engineering of IIS
  - Computational-Independent Design of an IIS: Reference Architecture
  - Platform-Independent Design of an IIS: Vocabulary Definitions and Integration Algorithms
- Conclusions
Selecting Ontology Languages/Algorithms

- Vocabulary Definition
- Concepts
- Attributes
- Relationships between Concepts
- Relationships between Attributes
- Instances
- Supported Processes
- Design
- Query
- Source Integration
- Annotation
- Mapping
- Extension Point Language
- Extension Point Supported Processes
- Vocabulary Definition

RDM / OWL / ...

Algorithm to ...

Requires

<table>
<thead>
<tr>
<th>Required Language Concepts</th>
<th>Thesaurus</th>
<th>Semantic Net</th>
<th>Log. Domain Theory</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Type) Concepts</td>
<td>(X)</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Relationships</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>formal is-a</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>binary relationships</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>n-ary relationships</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Axioms</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

Voc. Definition

Feature Model

Selected Features

Matching

Thesaurus / ...

OWL / ...

Extraction ...

• for combining existing languages
• for combining existing languages and algorithms
• for using existing languages and algorithms for a specific IIS

Using Ontology Languages / Algorithms

Automatic Generation / Integration → Model Transformation

Patterns to Solve Conflicts → Model Transformation

Component Library for Manual Composition → Validation, Dependency Checking
Correspondences – Solving Conflicts

Selected Features

LaV Correspondence Language

Mismatch

LaV-Specification + Query Restrictions

Using Ontology Languages / Algorithms

Automatic Generation / Integration → Model Transformation

Patterns to Solve Conflicts → Model Transformation

Component Library for Manual Composition → Validation, Dependency Checking

Model-Driven Engineering of IIS – Conclusions

Model-Driven Engineering?

Feature Modelling allows to deal with Variants!

- structures features and specifies dependencies (What is important to clarify?)
- allows to characterize existing concepts (Which is appropriate?)
- fills the gap between requirements and ‘solution’ (How to use ..?)

Model-Driven Engineering of IIS

- Computational-Independent Design: Selecting a reference architecture
- Platform-Independent Design: Adding languages and algorithms
- Platform-Specific Design: Realizing with an appropriate middleware

Current and Future Work

Platform-Independent Design of IIS

- Integration of Data Sources
- Query Processing and Optimization
- Information Integration / Object Fusion

Tool Development

- Feature Modelling
- Model Management
- Model Transformation
- Feature Discovery

Susanne Busse

sbusse@cs.tu-berlin.de

http://cis.cs.tu-berlin.de/~sbusse/