Quality Analysis of QVTo Model Transformations

Kick-off Presentation

Christine Gerpheide
Advisors: Alexander Serebrenik (TU/e)
Ramon Schiffelers (ASML)
Problem Statement

- Many resources are available to improve quality in GPLs

- Such resources don’t exist for QVTo
  - ...the OMG standard for M2M transformations
Background on QVT

- Query, View, Transformation (2007)
  - “Fantastically complex” [1]

- QVTo
  - Imperative
  - Composition, abstraction, inheritance, extension
  - Intermediate properties and classes
Research Questions

• Q1: How do we assess quality in QVTo model transformations?
  a) Which metrics exist which suggest quality attributes?
  b) What best practices can be identified?

• Q2: How do we improve quality in QVTo model transformations?
Ideal deliverables

- A tool for QVTo which helps improve code quality
- Distilled best practices
- Areas for future research
Approach

- MDE Quality Framework by Mohagheghi [3]

1. Identify quality goals/sub-goals
2. Identify target objects
3. Identify quality-carrying properties & product/project characteristics
4. Specify evaluation method/metrics
5. Quality engineering
6. Review
7. Execute
Applying the framework

1. **Identify quality goals**

2. **Identify target objects**
   - Transformation code and transformation process

3. **Identify quality-carrying properties**
   - Exploratory study

4. **Specify how to evaluate**

5. **Specify association between quality goals and properties**

6. **Review and evaluate for practice**

7. **Execute (implement) and re-evaluate model**
   - Validation
Exploratory Study

• Find information on:
  • Common issues
  • Design patterns
  • Difficulties
  • Metrics
  • Tooling
  • Testing
  • ...

• To finalize **quality goals and find quality-carrying properties in QVTo**
A peak at results so far...

• **Existing material**
  • Mixing notations (text, graphical, black-boxes) reduces understandability. [5,6]
  • Leveraging OCL syntax increases interoperability, conciseness and often understandability [10,11]
  • Exist QVTo tools for dependency analysis and unit testing already. [7,8,9]
• **4 ASML QVTo developers interviewed**
**Timeline**

- **Project dates:** 4/12/13 – 4/6/14 (26 weeks)

<table>
<thead>
<tr>
<th>Week</th>
<th>Activities</th>
</tr>
</thead>
</table>
| 1    | Research software quality  
      | Formalize approach |
| 2    | Learn QVTo |
| 3    | Prepare exploratory research |
| 4    | Christmas, mostly |
| 5    | Exploratory: Review existing material |
| 6    | Exploratory: Perform interviews and receive case studies |
| 7    | Process exploratory findings |
| 8…   | ...Specify how to evaluate quality properties, implement tool, validate! |
Questions?

• References

## Quality Framework Example

<table>
<thead>
<tr>
<th>Target Object</th>
<th>Quality Goal</th>
<th>Quality-carrying Property</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Select appropriate transformation approach [3]</td>
<td>Measure performance</td>
</tr>
<tr>
<td>Transformation model / rules</td>
<td>Preservation of consistency</td>
<td>Enforce consistency by tools [7]</td>
<td>Consistency analysis tool, measuring consistency before and after transformation</td>
</tr>
<tr>
<td>Reusability</td>
<td></td>
<td>Modularization, i.e. specialize and chain transformations, and rule inheritance</td>
<td>Inspection</td>
</tr>
<tr>
<td>Simplicity</td>
<td></td>
<td>Few number of rules, i.e. modularization</td>
<td>Measure complexity in the number or size of rules</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appropriate algorithm</td>
<td>Measure the complexity of algorithms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Simple output models</td>
<td>Measure complexity and size of the output model [22]</td>
</tr>
</tbody>
</table>