Software Process Validation: Comparing Process and Practice Models

Aldo de Moor
STARLab
Vrije Universiteit Brussel
ademoor@vub.ac.be

Harry Delugach
Dept. of Comp. Science
Univ. of Alabama in Huntsville
delugach@cs.uah.edu
Software Process vs Practice

• Software process: the normative, prescribed way of doing things in a software development organization
  – Providing coherence

• Software practice: the way things really get done
  – Learning, innovation, dealing with contingencies

• Process model: intended software process

• Practice model: consolidate individual lessons learnt

• Aim: detect and compare differences between process and practice (models)
  – How large are the differences?
  – How significant are the differences?
  – Where do they occur?
  – According to whose viewpoint?
Software Process Validation

Intended Process → Process Model Validation → First-hand observation → Actual Work Performance

Process Model

Preemptive (organization)

Practice Model

Model comparison

Preemptive (individuals)
Key concepts

• Workflows > **Steps** > Activities

- Activity pattern (definitional, schematic)
- Difference graphs: using color
<table>
<thead>
<tr>
<th>Difference operation</th>
<th>Name</th>
<th>To be highlighted</th>
<th>Where</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specialize concept</td>
<td>S-C</td>
<td>Concept</td>
<td>Both graphs</td>
</tr>
<tr>
<td>Specialize relation</td>
<td>S-R</td>
<td>Relation</td>
<td>Both graphs</td>
</tr>
<tr>
<td>Generalize concept</td>
<td>G-C</td>
<td>Concept</td>
<td>Both graphs</td>
</tr>
<tr>
<td>Generalize relation</td>
<td>G-R</td>
<td>Relation</td>
<td>Both graphs</td>
</tr>
<tr>
<td>Insert subgraph</td>
<td>I-G</td>
<td>Added subgraph</td>
<td>Result graph</td>
</tr>
<tr>
<td>Delete subgraph</td>
<td>D-G</td>
<td>Deleted subgraph</td>
<td>Source graph</td>
</tr>
<tr>
<td>Insert co-referent link</td>
<td>I-L</td>
<td>Linked concepts</td>
<td>Result graph</td>
</tr>
<tr>
<td>Delete co-referent link</td>
<td>D-L</td>
<td>Un-linked concepts</td>
<td>Source graph</td>
</tr>
<tr>
<td>Move into context</td>
<td>M-I</td>
<td>Moved elements</td>
<td>Result graph</td>
</tr>
<tr>
<td>Move from context</td>
<td>M-O</td>
<td>Moved elements</td>
<td>Source graph</td>
</tr>
</tbody>
</table>
Calculating difference sequences (ICCS 2005)

- Obtain the LCG of the two graphs
- Find a sequence of difference operations that transforms the first graph into the LCG
- Find a sequence of difference operations that transforms the LCG into the second graph.
- Concatenate the two sequences.

\[
DS( g_1, LCG )
\]

\[
DS( LCG, g_2 )
\]

\[
DS( G_1, G_2 ) :
\]

- **G-C:** Generalize `Reqts_Engineer: *r` to `Reqs_Engineer`
- **G-C:** Generalize `Reqts_Engineer: *s` to `Reqs_Engineer`
- **D-G:** Delete `notEqual` relation
- **S-C:** Specialize `Software_Engineer` to `Software_Engineer: Jerry`
- **S-C:** Specialize `Reqts_Engineer` to `Reqts_Engineer: Jerry`
A formal method for software process validation

- Create activity patterns for the model
- Create two models
  - Models can be either process or practice models
  - Identify model steps
  - Build models by instantiating or specialising general activity patterns for each step
- Compare the models
  - Identify activities in the model, and delineate the steps in a sequence
  - Compare steps between the models
  - Create difference graphs
- Interpret differences between the models
  - Present difference graphs to software professionals
  - Adapt process and/or practice models
  - Launch change program
Case: Aerospace Software Engineering

- Small internal software development group (10-20 persons)
- Multiple roles per person
- Proj. B: mission-critical propulsion system softw.
- Examined documents, interviewed manager
- Compared process models Proj. A and B (EMMSAD 2006)
Difference graph Proj A-B process models (Proj A perspective)
Conclusions

- Software process validation essential
- Formal comparison of process-practice models is key
- Identified differences interpreted by software professionals help identify problems, opportunities for improvement

Research agenda
- Ontology? What core patterns (activity etc.) needed?
- Knowledge representation and analysis:
  - Conceptual graph features (concept/relation type hierarchies, context, actors...)
  - Automating analysis (mappings, algorithms)
- Visualization (delineating steps etc.)?
- Elicitation / interpretation methodology