Steffen Mazanek, Mark Minas

**Business Process Models as a Showcase for Syntax-based Assistance in Diagram Editors**

Denver, 7 October 2009
Assistance Features

• Correction and completion (of processes)
• Example (process) generation
• Correctness-preserving editing operations including replace and intelligent remove
• Diagram contraction (i.e., auto-link for processes)

Demo
Generic Approach

• Process models practically relevant example appl.
• Approach realized in the DiaGen meta-tool
• Editors with the same features for several languages:
  – Nassi-Shneiderman Diagrams

\[
\begin{array}{|c|c|}
\hline
n & :=0 \\
\hline
\text{while } x>1 & \\
\hline
y & \text{x even?} \\
\hline
n & \\
\hline
x:=x/2 & \\
\hline
x:=3x+1 & \\
\hline
n:=n+1 & \\
\hline
\end{array}
\]
Generic Approach

• Process models practically relevant example appl.
• Approach realized in the DiaGen meta-tool
• Editors with the same features for several languages:
  – Nassi-Shneiderman Diagrams
  – Sequence Diagrams
• Process models practically relevant example appl.
• Approach realized in the DiaGen meta-tool
• Editors with the same features for several languages:
  – Nassi-Shneiderman Diagrams
  – Sequence Diagrams
  – Alligator Eggs
  – …
• But: also restrictions (to be discussed later)
  – Here: only well-structured process models
Structure of the Talk

- Diagram editors and meta-tools
  - Classification of editors and meta-tools

- Hypergraphs and hypergraph patches
  - Correspondence diagram – hypergraph
  - Hypergraph grammars for syntax definition
  - Hypergraph patches

- Using hypergraph patches in diagram editors
  - Diagram analysis in DiaGen editors
  - Syntax-based user assistance

- Further assistance features
  - Correctness-preserving editing operations
  - Diagram contraction

- Restrictions
Meta-Tools

- Generation of diagram editors from a specification
- Different approaches for syntax definition:
Meta-Tools

- Generation of diagram editors from a specification
- Different approaches for syntax definition:

```
C1:Class ::=
  C1:Class
\> src
\> trg
  :Association
\> trg
  C2:Class
```

```
Class
\> src
\> trg
Association
```
Meta-Tools

- existing meta-tools provide
  - good support for the editor developer
  - less (generic) support for the editor user
Meta-Tools

usually:
better editor $\rightarrow$ more work for

with our approach:
better editor and less work for

- existing meta-tools provide
  - good support for the *editor developer*
  - less (generic) support for the *editor user*
Classification of Diagram Editors

- Free-hand editor
  - diagram components can be freely arranged without any restrictions
    - a lot of freedom

- Structure editor
  - predefined operations that transform correct diagrams into (other) correct diagrams
    - a lot of guidance
Structure of the Talk

• Diagram editors and meta-tools
  – Classification of editors and meta-tools

• Hypergraphs and hypergraph patches
  – Correspondence diagram – hypergraph
  – Hypergraph grammars for syntax definition
  – Hypergraph patches

• Using hypergraph patches in diagram editors
  – Diagram analysis in DiaGen editors
  – Syntax-based user assistance

• Further assistance features
  – Correctness-preserving editing operations
  – Diagram contraction

• Restrictions
Hypergraphs as a Diagram Model

- Hypergraph: edges connect an arbitrary number of nodes depending on their label
- Edges represent diagram components
- Nodes represent attachment areas
Hypergraphs as a Diagram Model

• Spatial Relationship Hypergraph
Hypergraphs as a Diagram Model

- Abstract Syntax Hypergraph
Syntax Def by Hypergraph Grammar

context-free rules
unambiguous
Hypergraph Patches

• Description of hypergraph modifications

- Merge nodes
- Add edges (and nodes)
Patch-computing Parser

- Conventional parser just constructs a derivation tree/DAG wrt a grammar
- Extended parser computes patches while parsing
  - **Input**: - hypergraph H
    - hypergraph grammar G
    - number of edges to add
  - **Output**: all possible correcting hypergraph patches of this size
- For patch construction only context-free rules are considered
Structure of the Talk

- Diagram editors and meta-tools
  - Classification of editors and meta-tools
- Hypergraphs and hypergraph patches
  - Correspondence diagram – hypergraph
  - Hypergraph grammars for syntax definition
  - Hypergraph patches
- Using hypergraph patches in diagram editors
  - Diagram analysis in DiaGen editors
  - Syntax-based user assistance
- Further assistance features
  - Correctness-preserving editing operations
  - Diagram contraction
- Restrictions
Assistance: Diagram Completion

- Compute patch and carry it over to diagram
DiaGen Editing Process

I need help!

Editor user

selects operation

asks for assistance

chooses

Diagram

Modeler

SRG

Reducer

ASG

Parser

Update translator

Hypergraph patches

Layout information

Layouter

Drawing tool

Assistance
Structure of the Talk

• Diagram editors and meta-tools
  – Classification of editors and meta-tools

• Hypergraphs and hypergraph patches
  – Correspondence diagram – hypergraph
  – Hypergraph grammars for syntax definition
  – Hypergraph patches

• Using hypergraph patches in diagram editors
  – Diagram analysis in DiaGen editors
  – Syntax-based user assistance

• Further assistance features
  – Correctness-preserving editing operations
  – Diagram contraction

• Restrictions
Assistance: Editing Operations

• Break and patch

![Diagram](image-url)
Assistance: Editing Operations

• Benefits:
  ☺ preservation of correctness
  ☺ no additional specification effort
  ☺ exhaustive

• Integration of free-hand and syntax-directed editing
  ☺ a lot of freedom
  ☺ powerful guidance
Assistance: Diagram Contraction

• Derive a correct diagram using layout information

• Auto-link for BPMs
Structure of the Talk

• Diagram editors and meta-tools
  – Classification of editors and meta-tools

• Hypergraphs and hypergraph patches
  – Correspondence diagram – hypergraph
  – Hypergraph grammars for syntax definition
  – Hypergraph patches

• Using hypergraph patches in diagram editors
  – Diagram analysis in DiaGen editors
  – Syntax-based user assistance

• Further assistance features
  – Correctness-preserving editing operations
  – Diagram contraction

• Restrictions
Applicability / Restrictions

• Context-free languages
  – only context-free part of diagram is considered for the computation of patches
  – many languages are inherently context-sensitive
  – similar support for context-sensitive rules
  → (too) many solutions

Example:

\[
\text{C1:Class} \quad \text{C1:Class} \\
\downarrow \text{src} \\
\text{:Association} \\
\downarrow \text{trg} \\
\text{C2:Class} \quad \text{C2:Class}
\]

\[n^2 \text{ solutions for } n \text{ classes}\]
Empirical Evaluation

• BPM editor accepted for demo-track of the 2009 Business Process Management Conference

• Demonstration of novel features

• Participants answered questions regarding
  – their level of experience and role in business process modeling
  – their preferred business modeling tool
  – and, of course, the demonstrated features
3.1 How useful are the demonstrated editor features in your opinion?

<table>
<thead>
<tr>
<th>Feature Description</th>
<th>Very useful</th>
<th>Useful</th>
<th>Neutral</th>
<th>Of little use</th>
<th>Not useful</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process Completion (derive missing components)</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Example Generation (browse all structurally different process models)</td>
<td>2</td>
<td>3</td>
<td>8</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Editing Operations (insert components into some user-selected context)</td>
<td>8</td>
<td>6</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intelligent Remove (remove and reconnect automatically)</td>
<td>12</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auto-Link (derive sequence flow from arrangement of components)</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conclusions

• Syntax-based assistance is useful
• Syntax-based assistance can be computed without much specification effort
• Generic approach purely based on syntax, i.e., semantics not considered
• Resulting features are just complementary
• However: Speed & Context-freeness
• Patch-computing parser is available as a library → can be used with other (meta) tools
Future Work

• Sketching editor with user-assistance
• User-assistance with metamodel-based syntax specification
  → AtoM³