Sketch-based Metamodel Construction

Research Internship II
Lucas Heer
lucas.heer@student.uantwerpen.be

31.01.2018
Motivation
Motivation
Solution

What if we start from instance models?
Brainstorm phase | Domain experts | Create sketches | Example models | Update

Solution
Solution
Solution

Brainstorm phase

Domain experts

Create sketches

Example models

Update

<<conforms to>>

Derive

Test suite

Metamodel

Validate

Refactor / update

Create

Modeling tool

Language engineers

Refinement phase

Modeling phase
Research goal

- Examine current state of research
- Identify challenges
- Analyze existing solutions using concrete test cases
- Compare solutions and find open questions
## Existing approaches

<table>
<thead>
<tr>
<th>Name</th>
<th>Year</th>
<th>Implemented</th>
<th>Readily available</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scribbler</td>
<td>2013</td>
<td>Yes (Standalone)</td>
<td>No</td>
</tr>
<tr>
<td>MLCBD</td>
<td>2012</td>
<td>Yes (Visio plugin)</td>
<td>No</td>
</tr>
<tr>
<td>metaBup</td>
<td>2013-2017</td>
<td>Yes (EMF plugin)</td>
<td>Yes</td>
</tr>
<tr>
<td>FlexiSketch</td>
<td>2012-2015</td>
<td>Yes (Android App)</td>
<td>Yes</td>
</tr>
<tr>
<td>Model Workbench</td>
<td>2013-2014</td>
<td>Yes (Standalone)</td>
<td>No</td>
</tr>
</tbody>
</table>
Scribbler:


MLCBD:


metaBup:


Resources (2)

FlexiSketch:


Model Workbench:


Example: metaBup

Step 1: Draw example models
Example: metaBup

Step 2: Transform to text model

```
shell
importedFragment fragment fragment1 {
  Home_1 : Home {
    attr name = "Elliott Smith"
    @ overlapping ref modem = CableModem_3
  }
  InternetServiceProvider_1 : InternetServiceProvider {
    attr name = "lemon"
    ref infrastructure = ISPNetwork_1, ISPNetwork_2
  }
  Home_2 : Home {
    attr name = "Damien Jurado"
    @ overlapping ref modem = CableModem_2
  }
  Home_3 : Home {
    attr name = "Laura Marlin"
    @ overlapping ref modem = CableModem_1
  }
  CableModem_1 : CableModem {
    attr ipBase = "251.12.211.6"
    ref isp = ISPNetwork_1
  }
  CableModem_2 : CableModem {
    attr ipBase = "251.12.210.56"
    ref isp = ISPNetwork_1
  }
  CableModem_3 : CableModem {
    attr ipBase = "251.12.210.48"
    ref isp = ISPNetwork_2
  }
  ISPNetwork_1 : ISPNetwork {
    attr tier = 3
    attr location = "MAD"
  }
  ISPNetwork_2 : ISPNetwork {
    attr tier = 4
    attr location = "BCN"
  }
}
```
Example: metaBup

Step 2: Transform to text model (MM)
Example: metaBup

Step 3: Derive metamodel

```shell
class Home {
  attr name : String
  @ overlapping ref modem: CableModem
}

class InternetServiceProvider {
  attr name : String
  ref infrastructure: ISPNetwork
}

class CableModem {
  attr ipBase : String
  ref isp: ISPNetwork
}

class ISPNetwork {
  attr tier : int
  attr location : String
}
```

```
\[\text{InternetService Provider}\]
\[\text{name : String}\]
\[\text{ISPNetwork}\]
\[\text{tier : int}\]
\[\text{ISPNetwork}\]
\[\text{ipBase : String}\]
```

```
\[\text{Home}\]
\[\text{name : String}\]
\[\text{ISPNetwork}\]
\[\text{0..1 overlapping}\]
\[\text{CableModem}\]
\[\text{isp}
```

```
Challenges

1) Sketch recognition
2) Metamodel inference
3) Evolution
4) Integration
Challenges

Brainstorm phase

Domain experts
Create sketches
Example models
Update

Refinement phase
Test suite
Validate

Metamodel

Refactor / update

Modeling phase
Language engineers

Create
Modeling tool
Evaluation

- Evaluate tools based on challenges in multiple dimensions
- Define test cases and criteria for each dimension
## Evaluation

### Unconstrained input

<table>
<thead>
<tr>
<th>Tool</th>
<th>Language type</th>
<th>Input</th>
<th>Entities and relationships</th>
<th>Textual annotations</th>
<th>Spatial relationships</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scribbler</td>
<td>Visual</td>
<td>Freehand</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MLCBD</td>
<td>Visual</td>
<td>Editor</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes (draw)</td>
</tr>
<tr>
<td>FlexiSketch</td>
<td>Visual</td>
<td>Freehand</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>metaBup</td>
<td>Visual</td>
<td>Editor</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Model Workbench</td>
<td>Textual</td>
<td>Editor (text)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes (textual)</td>
</tr>
</tbody>
</table>
## Evaluation

### Metamodel generation

<table>
<thead>
<tr>
<th>Tool</th>
<th>Metamodel generation</th>
<th>Advanced constructs</th>
<th>Automation</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLCBD</td>
<td>Implicit</td>
<td>-</td>
<td>Semi</td>
</tr>
<tr>
<td>FlexiSketch</td>
<td>Implicit</td>
<td>-</td>
<td>Full</td>
</tr>
<tr>
<td>metaBup</td>
<td>Explicit</td>
<td>Inheritance, Abstract classes, Compositions</td>
<td>Full</td>
</tr>
<tr>
<td>Model Workbench</td>
<td>Explicit</td>
<td>Inheritance, Abstract classes</td>
<td>Semi</td>
</tr>
</tbody>
</table>
## Evaluation

### Co-evolution

<table>
<thead>
<tr>
<th>Tool</th>
<th>Forward evolution</th>
<th>Backward evolution</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scribbler</td>
<td>No</td>
<td>Manual</td>
<td>None</td>
</tr>
<tr>
<td>MLCBD</td>
<td>Manual</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>FlexiSketch</td>
<td>Yes</td>
<td>Yes</td>
<td>None</td>
</tr>
<tr>
<td>metaBup</td>
<td>Yes</td>
<td>Yes</td>
<td>Non-breaking, Resolvable, Unresolvable</td>
</tr>
<tr>
<td>Model Workbench</td>
<td>Yes</td>
<td>Yes</td>
<td>Non-breaking, Breaking</td>
</tr>
</tbody>
</table>
## Evaluation

### Tool support

<table>
<thead>
<tr>
<th>Tool</th>
<th>Scribbler</th>
<th>MLCBD</th>
<th>FlexiSketch</th>
<th>metaBup</th>
<th>Model Workbench</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Implementation</strong></td>
<td>Java Application</td>
<td>MS Visio Plugin</td>
<td>Android App Java Application</td>
<td>EMF Plugin</td>
<td>Java EE Application</td>
</tr>
<tr>
<td><strong>Integration</strong></td>
<td>EMF</td>
<td>-</td>
<td>-</td>
<td>EMF metaDepth</td>
<td>Self-contained</td>
</tr>
<tr>
<td><strong>Usability</strong></td>
<td>Industrial user study</td>
<td>Case study</td>
<td>User study</td>
<td>User study</td>
<td>-</td>
</tr>
<tr>
<td><strong>Scalability</strong></td>
<td>?</td>
<td>?</td>
<td>Limited</td>
<td>Good</td>
<td>?</td>
</tr>
<tr>
<td><strong>Collaboration</strong></td>
<td>Client-Server</td>
<td>-</td>
<td>Client-Server</td>
<td>-</td>
<td>Client-Server</td>
</tr>
</tbody>
</table>
Analysis

Unconstrained input

Metamodel generation

Co-evolution

Tool support

0: No support
1: Weak
2: Good
3: Advanced

Scribbler
MLCBD
metaBup
FlexiSketch
Open research questions

- Concrete ↔ abstract syntax rigid
- Co-evolution mostly rudimentary
- Gap between metamodel design and usage phases
- Immature tools with little experience
- Semantics
How agile is it really?
Open research questions

- Concrete ↔ abstract syntax rigid
- Co-evolution mostly rudimentary
- Gap between metamodel design and usage phases
- Immature tools with little experience
- Semantics
Future work