Towards Hybrid Model Persistence

Alfa Yohannis¹ Horacio Hoyos Rodriguez*¹
Fiona Polack**² Dimitris Kolovos¹

¹Department of Computer Science, University of York, United Kingdom
²School of Computing and Maths, Keele University, United Kingdom

{ary506, dimitris.kolovos}@york.ac.uk
*horacio_hoyos_rodriguez@ieee.org
**f.a.c.polack@keele.ac.uk

Models and Evolution@MODELS 2018
Tuesday, 16 October 2018
Copenhagen, Denmark
Change-based Persistence (CBP) & State-based Persistence (SBP)

1. session 1
2. create p1 type Package
3. set p1.name to "X"
4. create c1 type Class
5. set c1.name to "A"
6. create c2 type Class
7. set c2.name to "B"
8. add c1 to p1.packagedElement
9. add c2 to p1.packagedElement
10. session 2
11. set c2.name to "C"
12. remove c1 from p1.children
13. delete c1

```xml
<uml:Package xmi:id="1" name="X">
  <packagedElement xsi:type="uml:Class"
    xmi:id="3" name="C"/>
</uml:Package>
```
Introduction

• Model Change-Based Persistence (CBP) persists the complete history of changes of a model instead of its eventual state.
Main Goals

• Use the persisted changes to optimise model comparison, merging, and management (out of the scope of the paper)

• Support common text-based Version Control Systems (e.g. Git, SVN) to persist changes

• Support collaborative modelling

• **Challenge:** Loading Time
State-based Persistence (SBP)

The eventual state of the model (time stamp 7) persisted in XMI:

```xml
<uml:Package xmi:id="1" name="X">
    <packagedElement xsi:type="uml:Class"
        xmi:id="3" name="C"/>
</uml:Package>
```
The changes of the model persisted in a pseudo change-based format:

1. `session 1`
2. `create p1 type Package`
3. `set p1.name to "X"`
4. `create c1 type Class`
5. `set c1.name to "A"`
6. `create c2 type Class`
7. `set c2.name to "B"`
8. `add c1 to p1.packagedElement`
9. `add c2 to p1.packagedElement`
10. `session 2`
11. `set c2.name to "C"`
12. `remove c1 from p1.children`
13. `delete c1`
Change-based vs. State-based Persistence

• **Notable Advantages** of change-based over state-based persistence:
  • Speed up incremental model management activities
  • Enable novel model analytics
  • Faster model comparison and merging

• **Drawbacks** of change-based over state-based persistence:
  • Ever-growing of model files (size)
  • **Longer loading times**
## Change-based vs. State-based Persistence

<table>
<thead>
<tr>
<th>Dimensions</th>
<th>Change-based</th>
<th>State-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load Time</td>
<td>−</td>
<td>+</td>
</tr>
<tr>
<td>Save Time</td>
<td>+</td>
<td>−</td>
</tr>
<tr>
<td>Comparison Time</td>
<td>+</td>
<td>−</td>
</tr>
<tr>
<td>Storage Space</td>
<td>−</td>
<td>+</td>
</tr>
</tbody>
</table>

Introduce **hybrid model persistence** to address the **model loading time space** at the cost of additional space.
Previous Work

• Loading/replaying all events is **not efficient**

• Loading optimisation of change-based models
  • **Ignoring events** that have no effect on the eventual state of a change-based model
  • The optimisation only save up to around 50% of the un-optimised loading
  • The loading time is still **outperformed by** state-based models (less than 10% of the un-optimised loading)
Optimised Change-based Persistence

Ignoring events that have no effect when loading the model:

1. session 1
2. create p1 type Package
3. set p1.name to "X"
4. create c1 type Class
5. set c1.name to "A"
6. create c2 type Class
7. set c2.name to "B"
8. add c1 to p1.packagedElement
9. add c2 to p1.packagedElement
10. session 2
11. set c2.name to "C"
12. remove c1 from p1.children
13. delete c1
Related Work

• RDBMS & NoSQL model persistence
  • Lazy loading
    • i.e. CDO, Morsa, NeoEMF

• Hybrid model persistence?
Related Work

• EMFStore uses hybrid persistence approach:
  • Supports different backends
  • No support for common version control systems (VCS), i.e. Git, SVN to persist changes
  • Restricted to its own VCS
The Mechanism of Hybrid Model Persistence

Clients (model editors, transformations etc.)

Hybrid model

Editing session changes

In-memory object graph

Changes appended to (on save)

Change-based representation (file)

Populates on load (if state-based representation does not exist)

Serialised (on save)

State-based representation (file/database)

Stored in Version Control Repository
Hybrid Model Persistence Implementation
Evaluation

• Change-based Model Dataset was derived from version control repositories:
  • UML2 models of the BPMN2 project
  • UML2 models of the Epsilon project
  • ModiscoXML models of the Wikipedia’s United States article

• Evaluation on:
  • Space Usage
  • Load Memory
  • Load Time
  • Save Memory
  • Save Time
# Evaluation: Space Usage

<table>
<thead>
<tr>
<th>Case</th>
<th>Epsilon</th>
<th></th>
<th>BPMN2</th>
<th></th>
<th>Wikipedia</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generated From</td>
<td>940 commits</td>
<td>192 commits</td>
<td>10,187 versions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>XMI</td>
<td>NeoEMF</td>
<td>CBP</td>
<td>XMI</td>
<td>NeoEMF</td>
</tr>
<tr>
<td>Element Count</td>
<td>88,020</td>
<td>88,020</td>
<td>—</td>
<td>62,062</td>
<td>62,062</td>
</tr>
<tr>
<td>Event Count</td>
<td>—</td>
<td>—</td>
<td>4.3 m</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Space Size</td>
<td>9.44</td>
<td>188</td>
<td>406</td>
<td>6.55</td>
<td>134</td>
</tr>
<tr>
<td>Average Space</td>
<td>112</td>
<td>2</td>
<td>98</td>
<td>110</td>
<td>2</td>
</tr>
<tr>
<td>Size</td>
<td>bytes/element</td>
<td>KBs/event</td>
<td>bytes/element</td>
<td>KBs/event</td>
<td>bytes/element</td>
</tr>
<tr>
<td>m = million events, MB = Megabytes, KB = Kilobytes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evaluation: Load Memory

BPMN2

Epsilon

Wikipedia
Evaluation: Load Time

![Box plots comparing load times for BPMN2 and Epsilon on different platforms: H-NEO, NEO, H-XMI, XMI. The plots show the distribution of load times in seconds, with outliers represented as individual points.](image)
Evaluation: Save Memory

**BPMN2**

<table>
<thead>
<tr>
<th>H-NEO</th>
<th>NEO</th>
<th>H-XMI</th>
<th>XMI</th>
</tr>
</thead>
</table>

**Epsilon**

<table>
<thead>
<tr>
<th>H-NEO</th>
<th>NEO</th>
<th>H-XMI</th>
<th>XMI</th>
</tr>
</thead>
</table>

**Wikipedia**

<table>
<thead>
<tr>
<th>H-NEO</th>
<th>NEO</th>
<th>H-XMI</th>
<th>XMI</th>
</tr>
</thead>
</table>
Evaluation: Save Time

BPMN2

Epsilon

Wikipedia
Detecting Previous Versions’ Deleted Elements: UML2 Models of Epsilon Project

<table>
<thead>
<tr>
<th>Versions</th>
<th>Element Counts</th>
<th>Delta Elements (from ver. 8)</th>
<th>Event Counts</th>
<th>Delta Events (from ver. 8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>008</td>
<td>25,993</td>
<td>0</td>
<td>90,888</td>
<td>0</td>
</tr>
<tr>
<td>044</td>
<td>31,240</td>
<td>5,247</td>
<td>166,659</td>
<td>75,771</td>
</tr>
<tr>
<td>181</td>
<td>34,196</td>
<td>8,203</td>
<td>250,073</td>
<td>159,185</td>
</tr>
<tr>
<td>388</td>
<td>48,482</td>
<td>22,489</td>
<td>332,315</td>
<td>241,427</td>
</tr>
</tbody>
</table>

![Bar chart showing comparison of execution times between versions 008 vs. 044, 008 vs. 181, and 008 vs. 388. The times are: 14.28, 28.62, and 42.93 seconds respectively.]

21
Conclusions and Future Work

• Conclusions
  • Proposed a hybrid model persistence approach
  • Evaluated its impact on storage space usage and time and memory footprint for model loading and saving
  • The hybrid model persistence provides benefits on model loading time
  • Acceptable trade-off on memory footprint and storage space usage

• Future Work:
  • Utilise the persisted changes to optimise model comparison, merging, and management