Towards Hybrid Model Persistence

Alfa Yohannis[1] Horacio Hoyos Rodriguez[*1]
Fiona Polack[**2] Dimitris Kolovos[1]

[1]Department of Computer Science, University of York, United Kingdom
[2]School 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
```

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

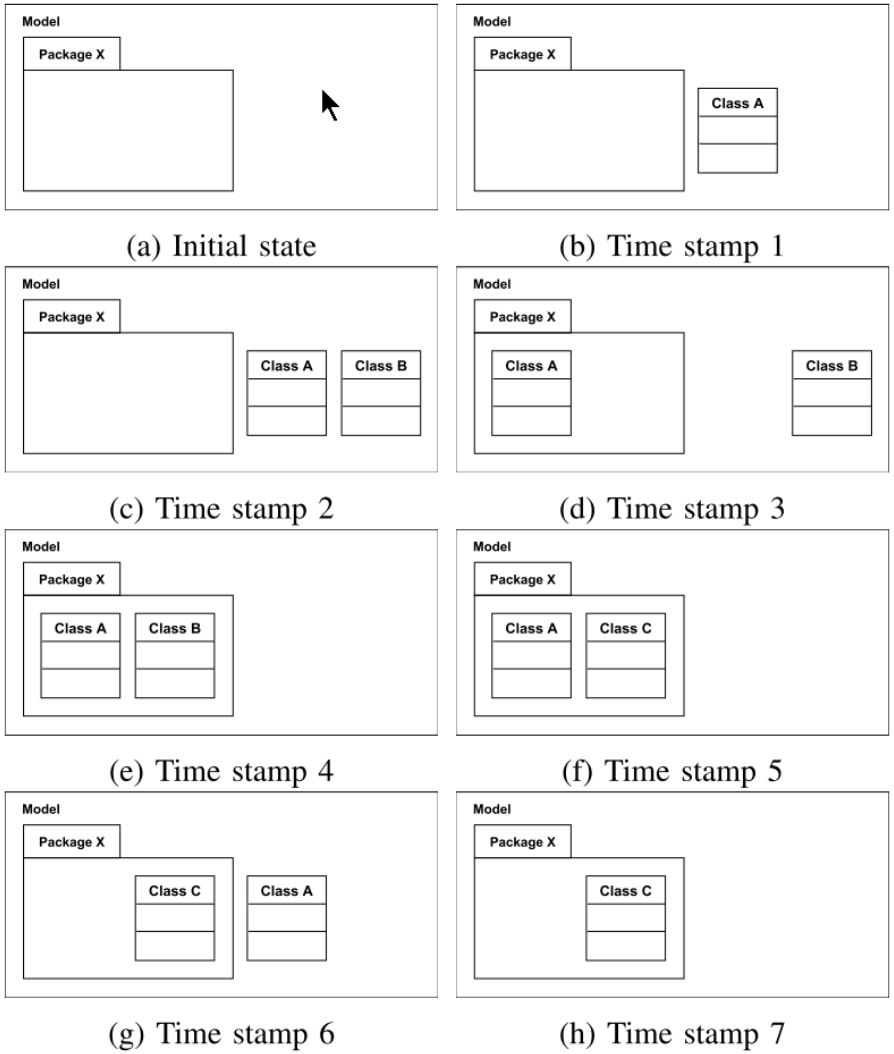
Introduction

- 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)



(a) Initial state (b) Time stamp 1

(c) Time stamp 2 (d) Time stamp 3

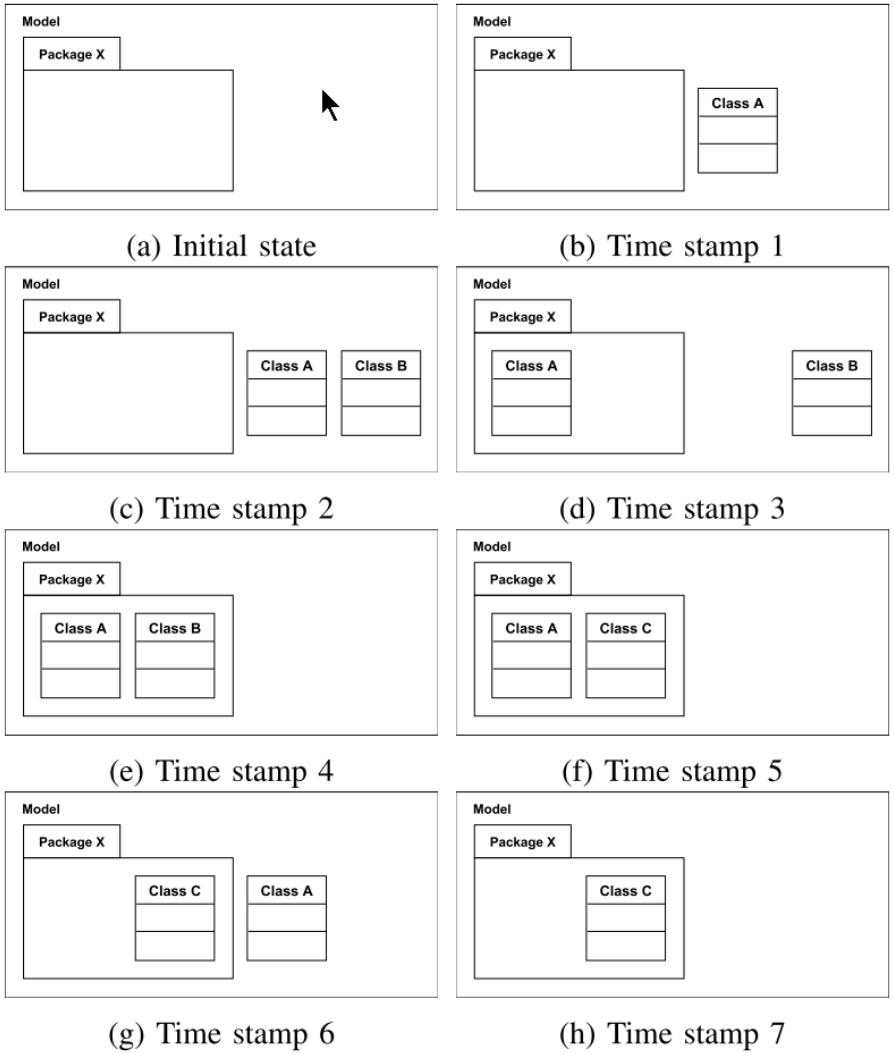
(e) Time stamp 4 (f) Time stamp 5

(g) Time stamp 6 (h) Time stamp 7

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

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

Change-based Persistence (CBP)



(a) Initial state (b) Time stamp 1

(c) Time stamp 2 (d) Time stamp 3

(e) Time stamp 4 (f) Time stamp 5

(g) Time stamp 6 (h) Time stamp 7

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

Dimesions	Change-based	State-based
Load Time	—	+
Save Time	+	—
Comparison Time	+	—
Storage Space	—	+

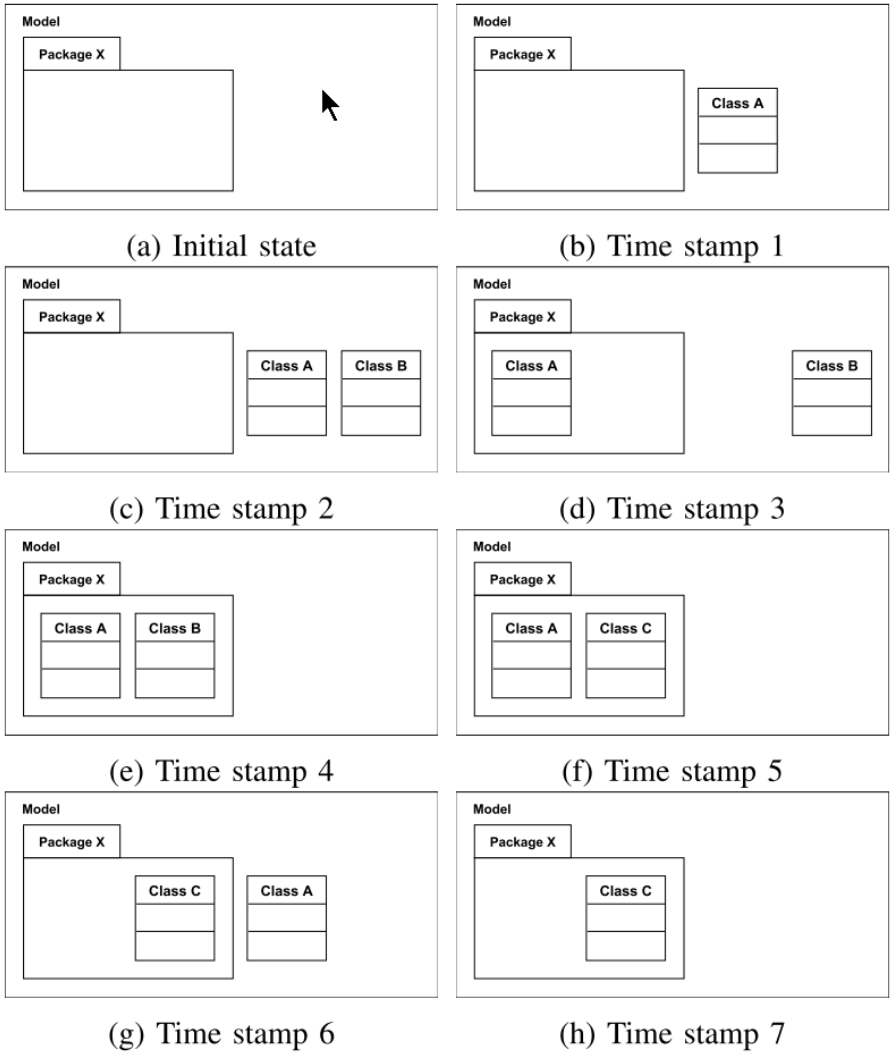
Introduce **hybrid model persistence** to address the **model loading time space** at the cost of additional space

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



(a) Initial state (b) Time stamp 1

(c) Time stamp 2 (d) Time stamp 3

(e) Time stamp 4 (f) Time stamp 5

(g) Time stamp 6 (h) Time stamp 7

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
```

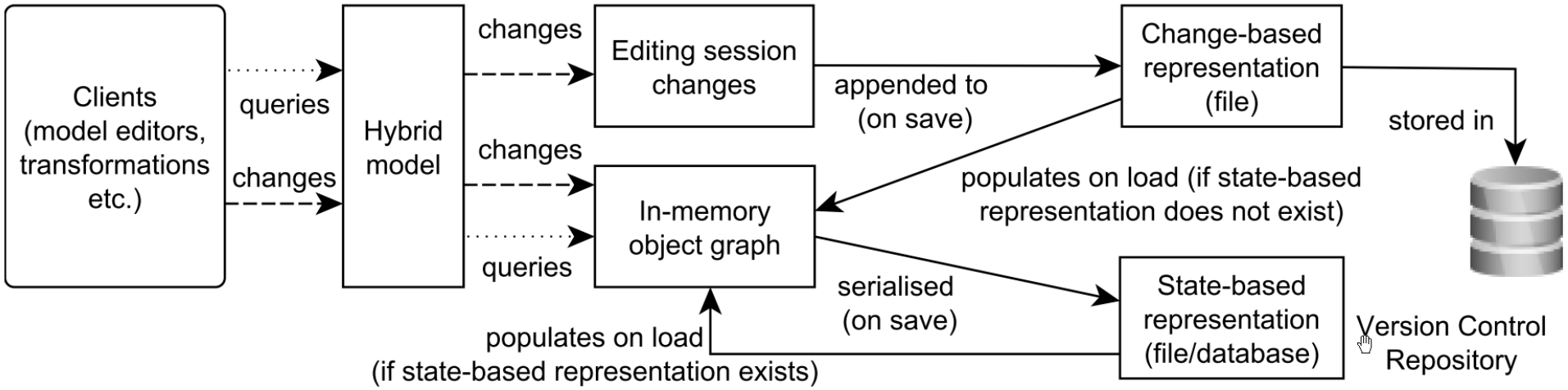
Related Work

- RDBMS & NoSQL model persistence
  - **Lazy loading**
  - i.e. CDO, Morsa, NeoEMF
- Hybrid model persistence?

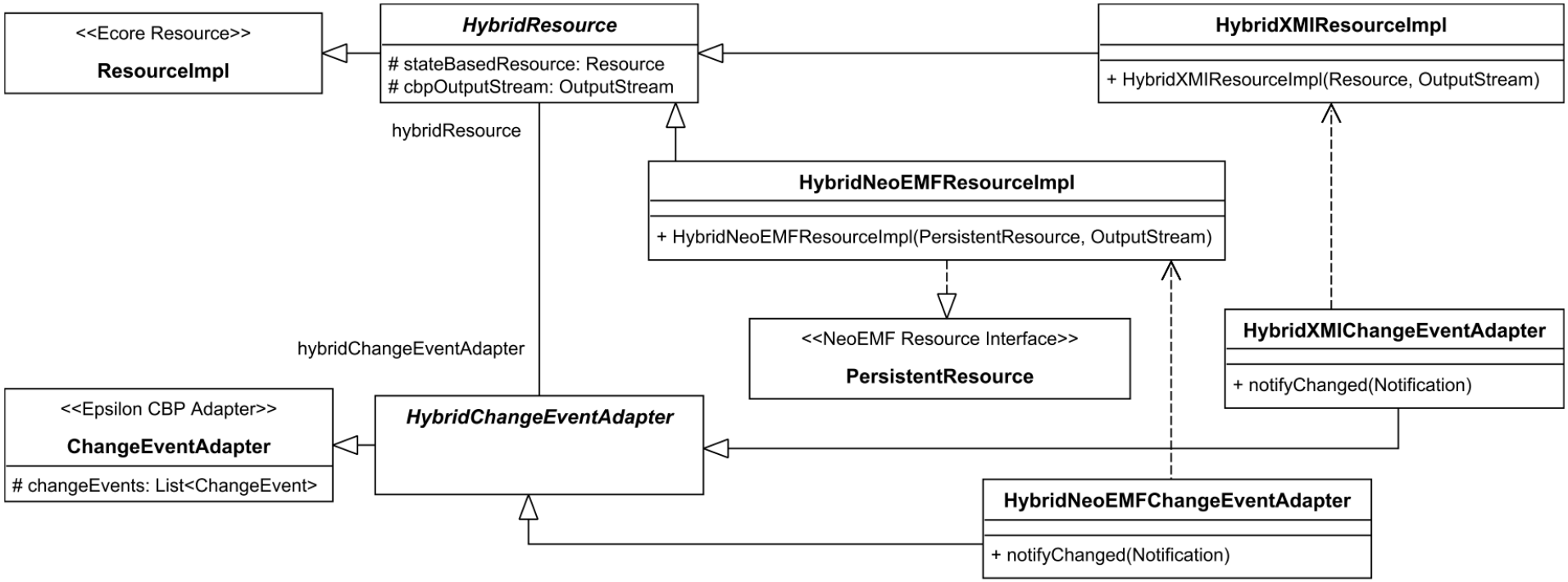
Related Work

- 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



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

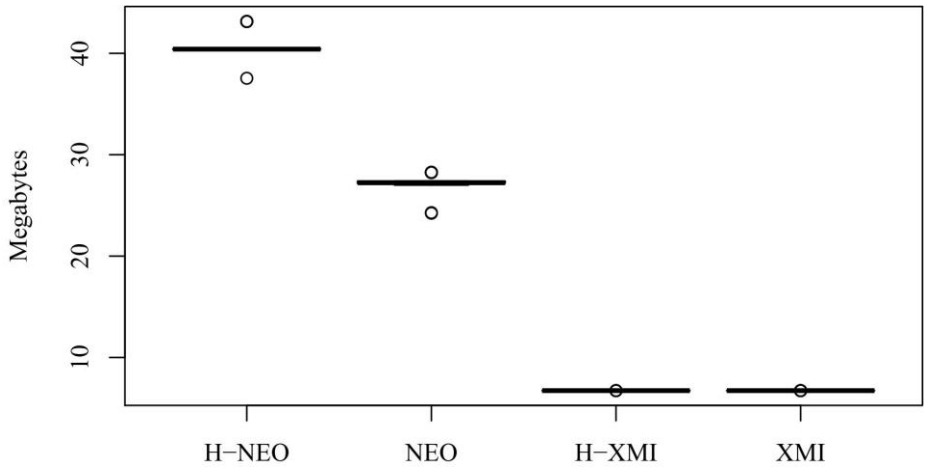
- Evaluation on:
  - Space Usage
  - Load Memory
  - Load Time
  - Save Memory
  - Save Time

Evaluation: Space Usage

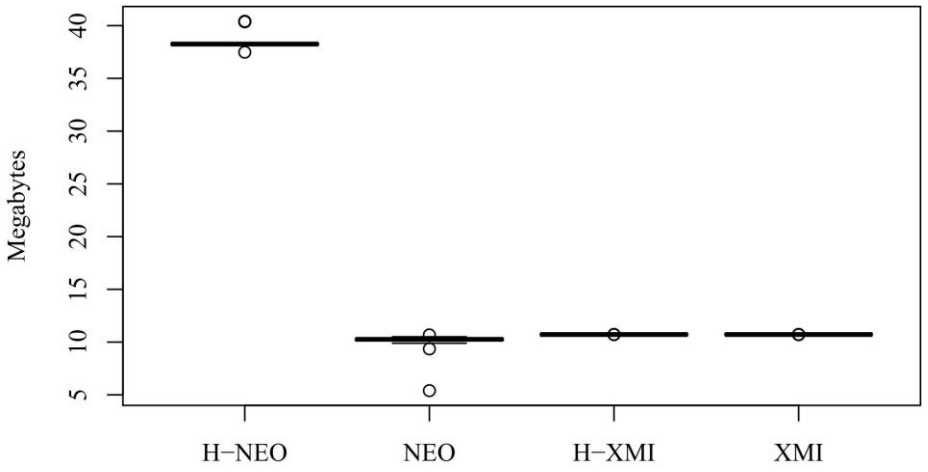
Case	**Epsilon**			**BPMN2**			**Wikipedia**		
Generated From	940 commits			192 commits			10,187 versions		
Type	XMI	NeoEMF	CBP	XMI	NeoEMF	CBP	XMI	NeoEMF	CBP
Element Count	88,020	88,020	—	62,062	62,062	—	13,112	13,112	—
Event Count	—	—	4.3 m	—	—	1.2 m	—	—	62.3 m
Space Size	9.44 MBs	188 MBs	406 MBs	6.55 MBs	134 MBs	109 MBs	1.28 MBs	31.8 MBs	5.85 GBs
Average Space Size	112 bytes/ element	2 KBs/ element	98 bytes /event	110 bytes/ element	2 KBs/ element	92 bytes /event	102 bytes/ element	2 KBs/ element	98 bytes /event

m = million events, MB = Megabytes, KB = Kilobytes

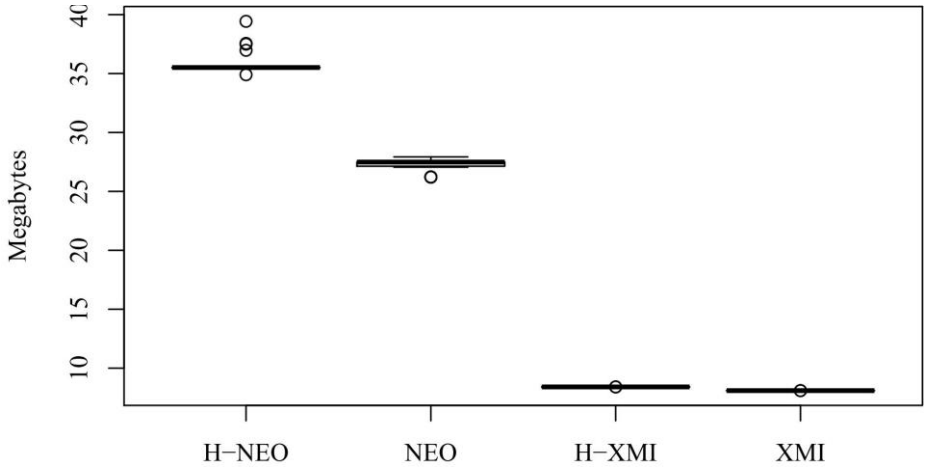
Evaluation: Load Memory



BPMN2

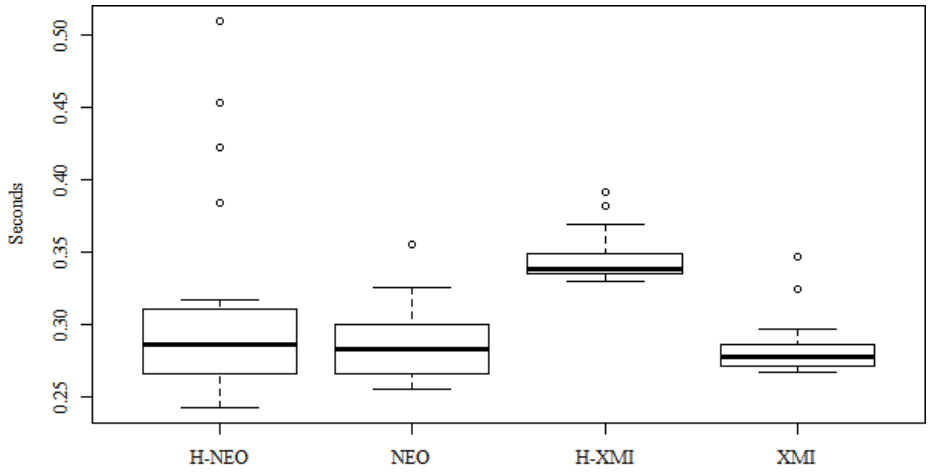


Epsilon

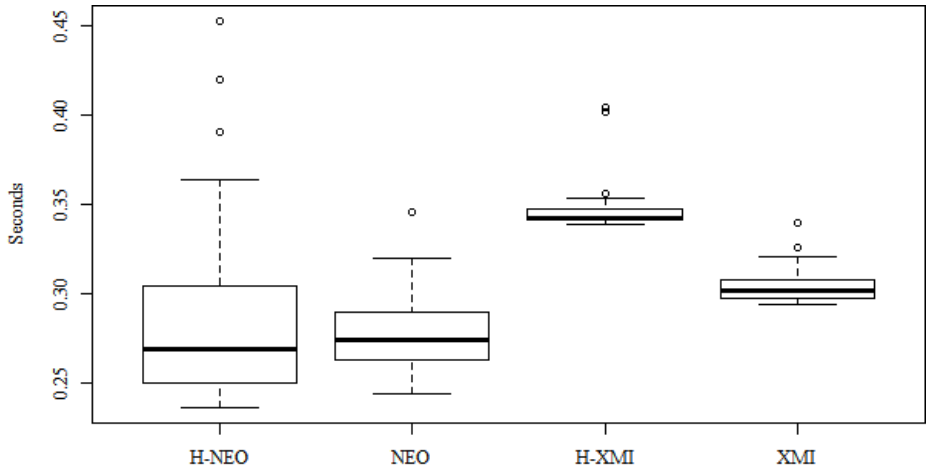


Wikipedia

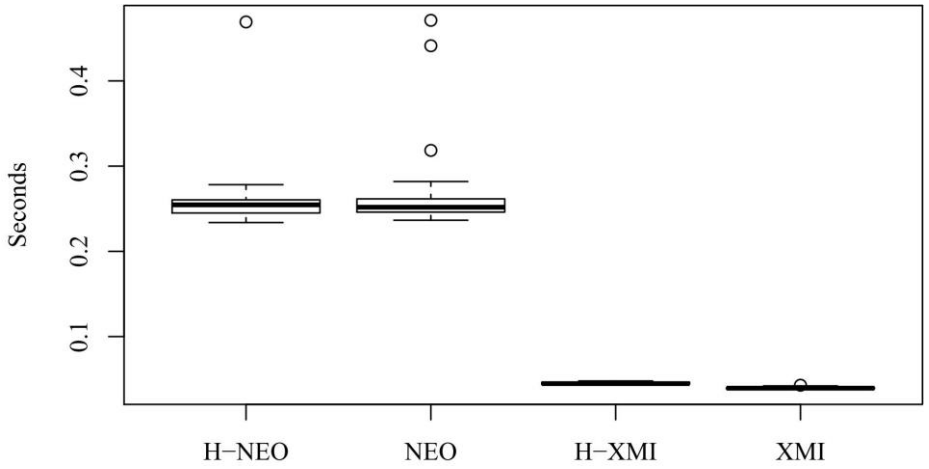
Evaluation: Load Time



BPMN2

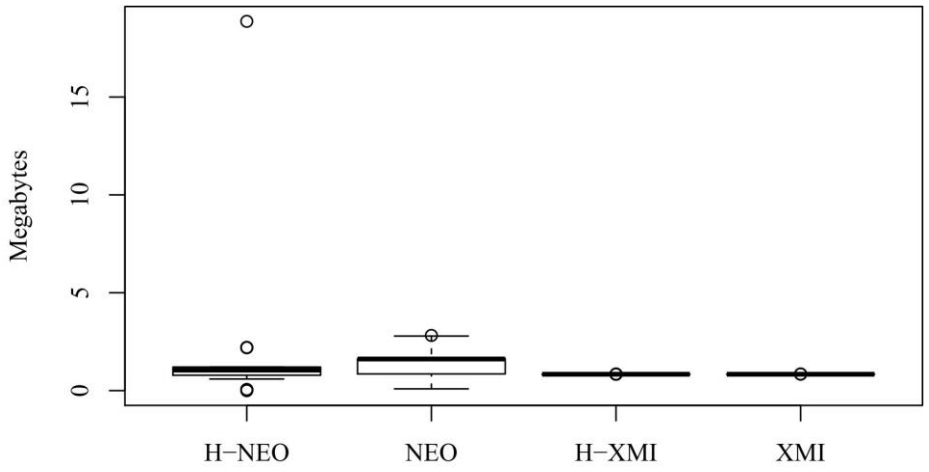


Epsilon

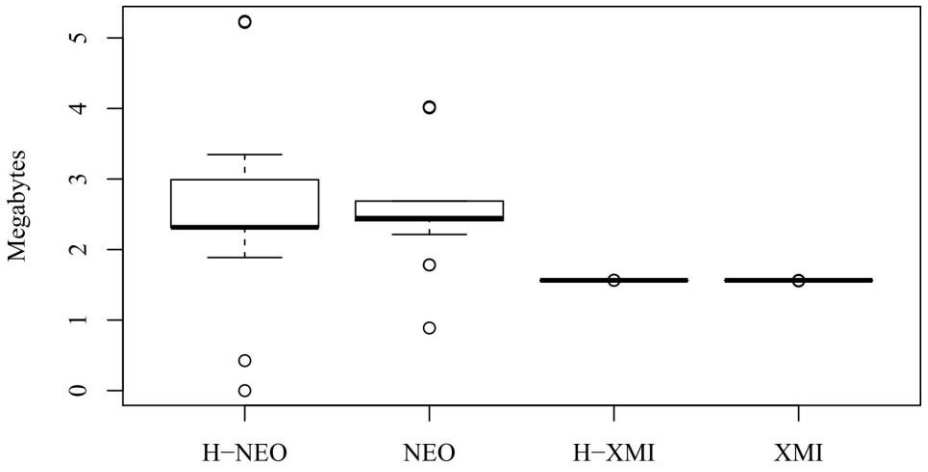


Wikipedia

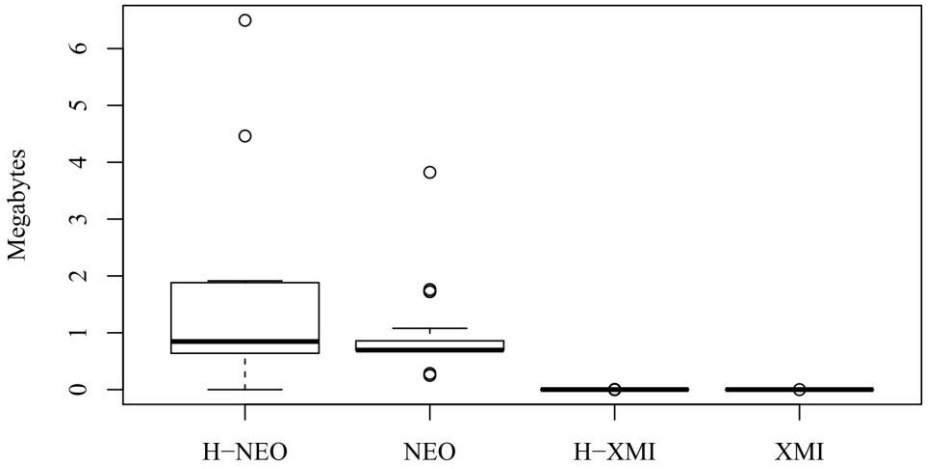
Evaluation: Save Memory



BPMN2

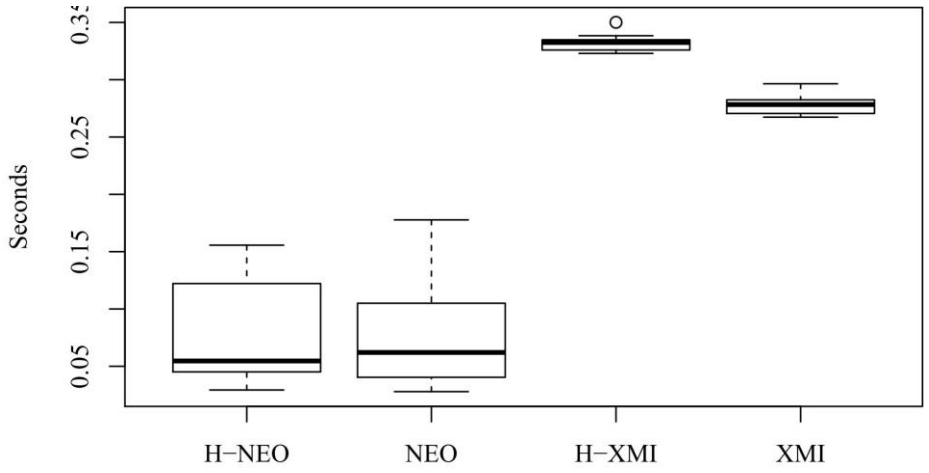


Epsilon

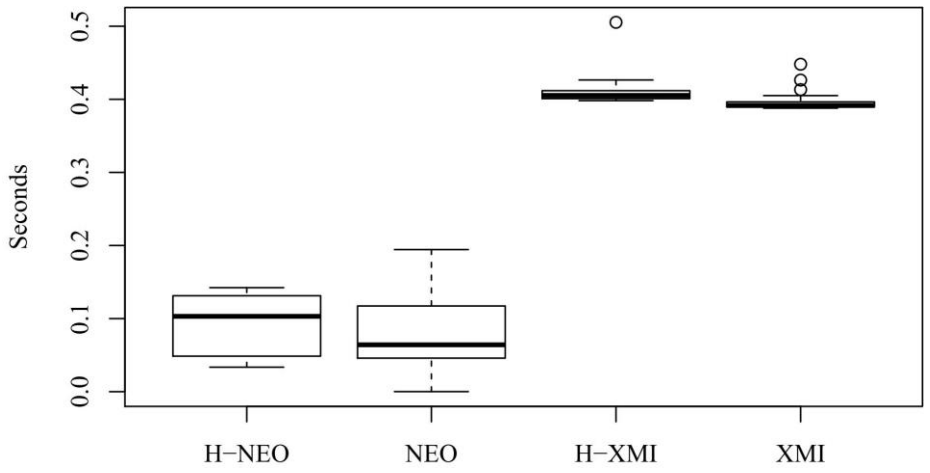


Wikipedia

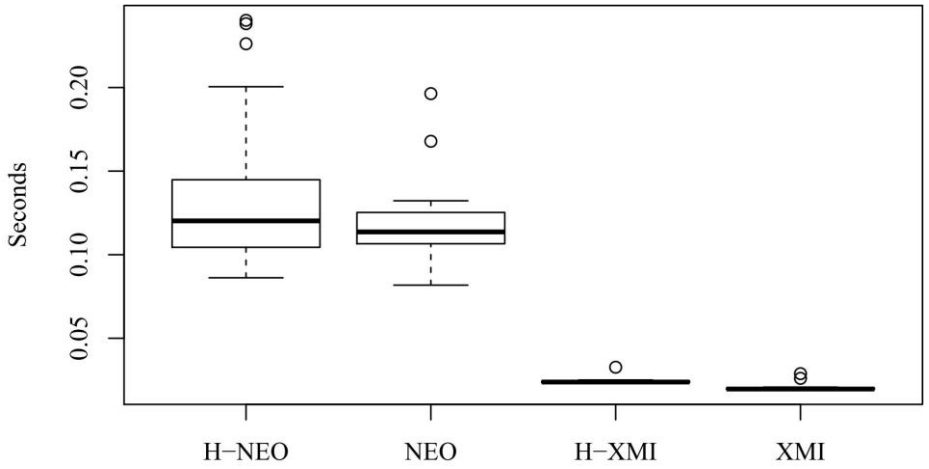
Evaluation: Save Time



BPMN2



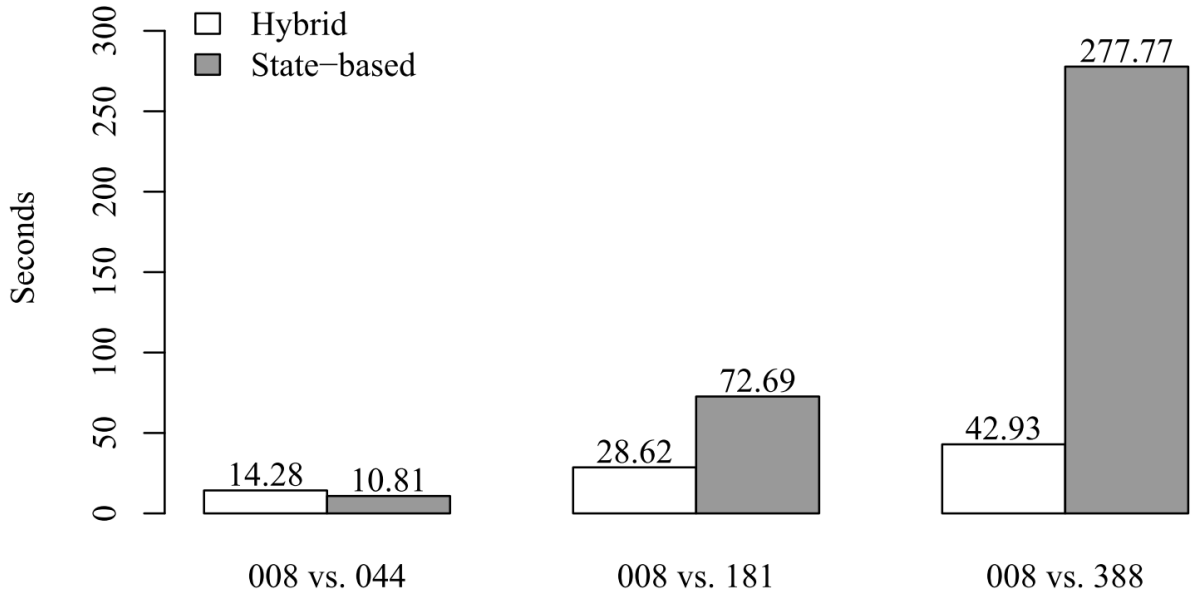
Epsilon



Wikipedia

Detecting Previous Versions' Deleted Elements:
UML2 Models of Epsilon Project

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



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