Decoupling validation UIs using Publish-Subscribe binding of instance variables in Overture

Luis Diogo Couto[1] Kenneth Lausdahl[2] Nico Plat[3]
Peter Gorm Larsen[2] Ken Pierce[4]

United Technologies Research Center, Ireland

Aarhus University, Department of Engineering, Denmark

West IT Solutions, The Netherlands

Newcastle University, School of Computing Science, UK

November 7, 2016 / 14th Overture Workshop

Outline

Outline

TEMPO Project

- TEMPO[1]
- Investigated collaboration between different Traffic Management Systems (TMSs)
- Traffic simulations produce a large amount of numerical data
- Required a 2D/3D visualisation library

[1] See `http://tempoproject.eu/`

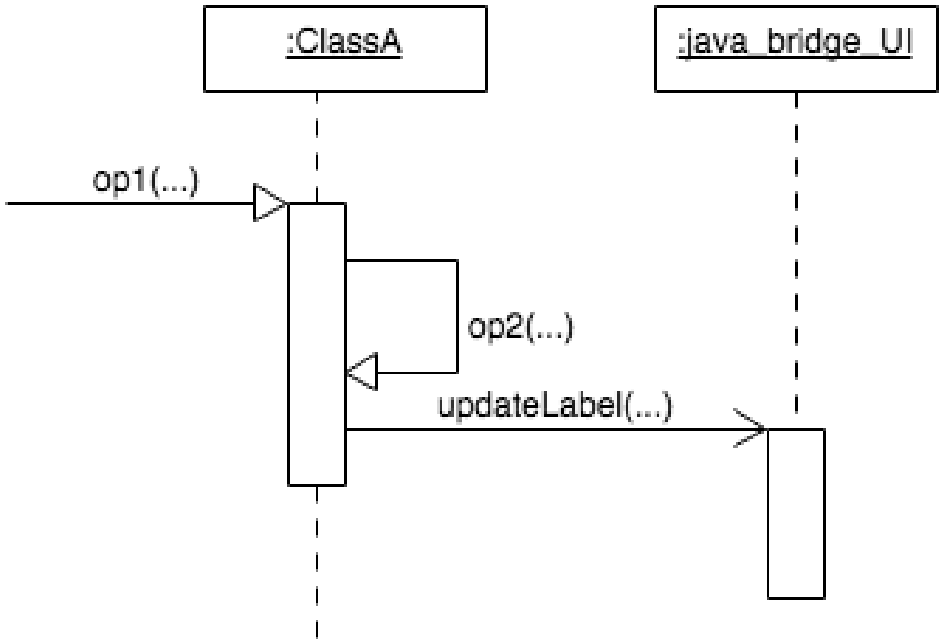
Coupling UI with Formal Models

Goal

- Enable domain experts to validate formal models
- Make it easy to create visual representations of formal models
- Loose coupling between UI and model

Existing Solutions

- Overture supports class path loaded instance link to:
  - `is not yet specified`
- VDM Tools has `DLModule`



Improving Existing Solutions

- Ease UI development with state of the art technologies
- Enable rapid UI development
- Reduce the model specific UI code

Improving Existing Solutions

- Ease UI development with state of the art technologies
- Enable rapid UI development
- Reduce the model specific UI code

Design Principles

1. The extension must enable the use of modern and fast UI technologies; and
2. the UI code must not pollute the VDM model.

Outline

Implementation

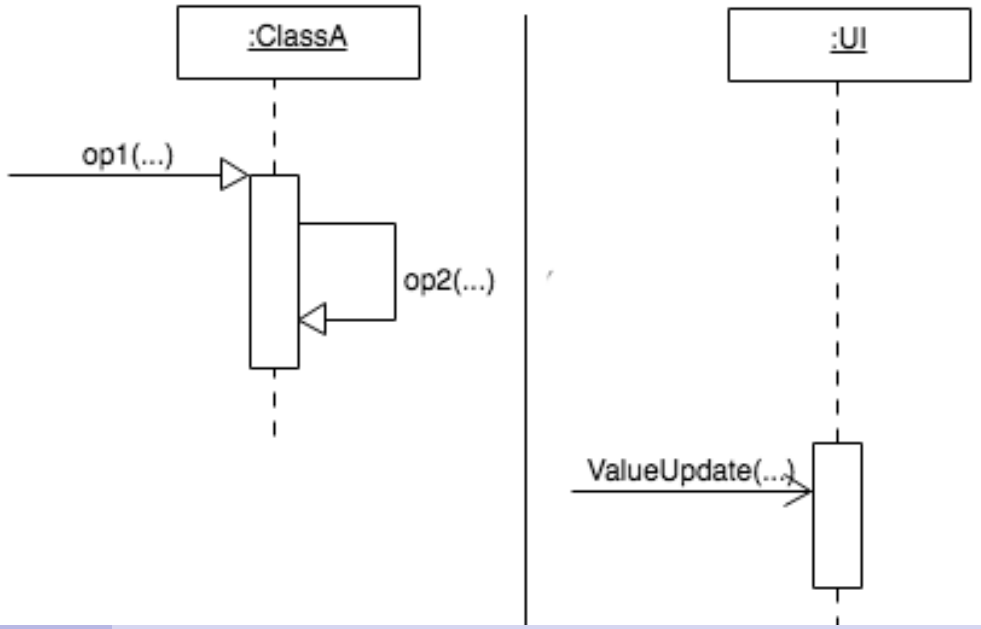
Solution

- Decouple Model and UI through *Value notifications*
- Decouple UI and interpreter through `RemoteInterpreter` with JSON using the public / subscribe design patter

Implementation

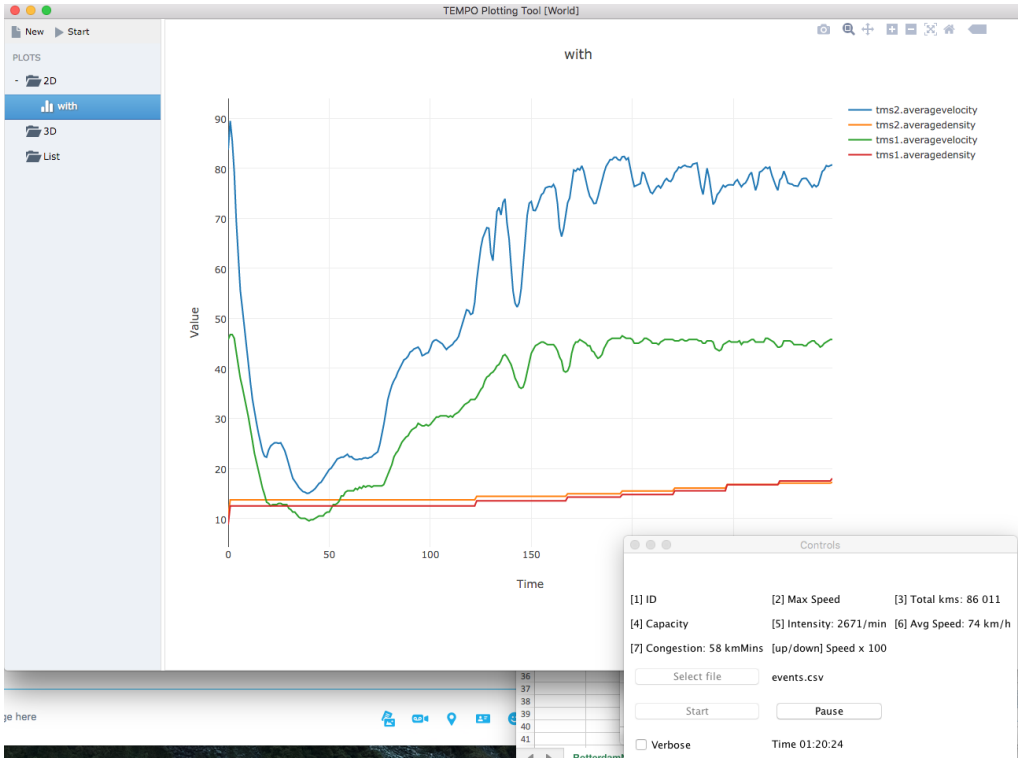
Solution

- Decouple Model and UI through *Value notifications*
- Decouple UI and interpreter through `RemoteInterpreter` with JSON using the public / subscribe design patter



Modern Graphical Platforms

- Faster prototyping than e.g. Swing
- Web-technologies is a good alternative
  - There exists many libraries for visualization
- Electron can wrap Web-technologies as native applications



Simple integration

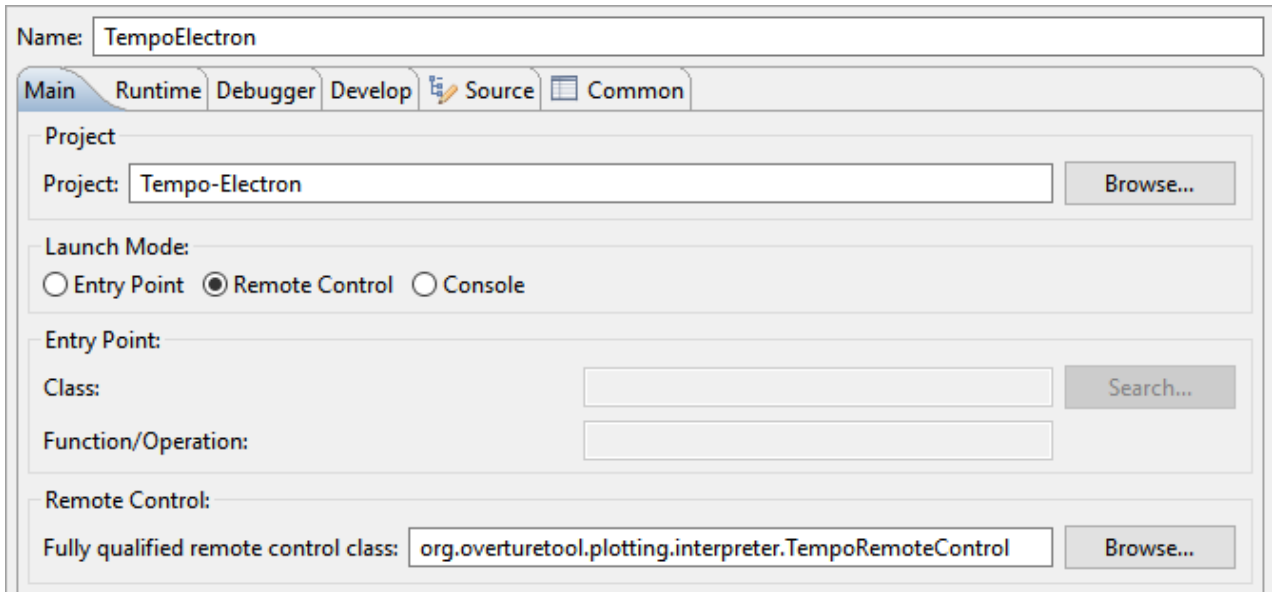


Figure: Overture launch configuration for remote control.

Demo

Demo

Demo
Model

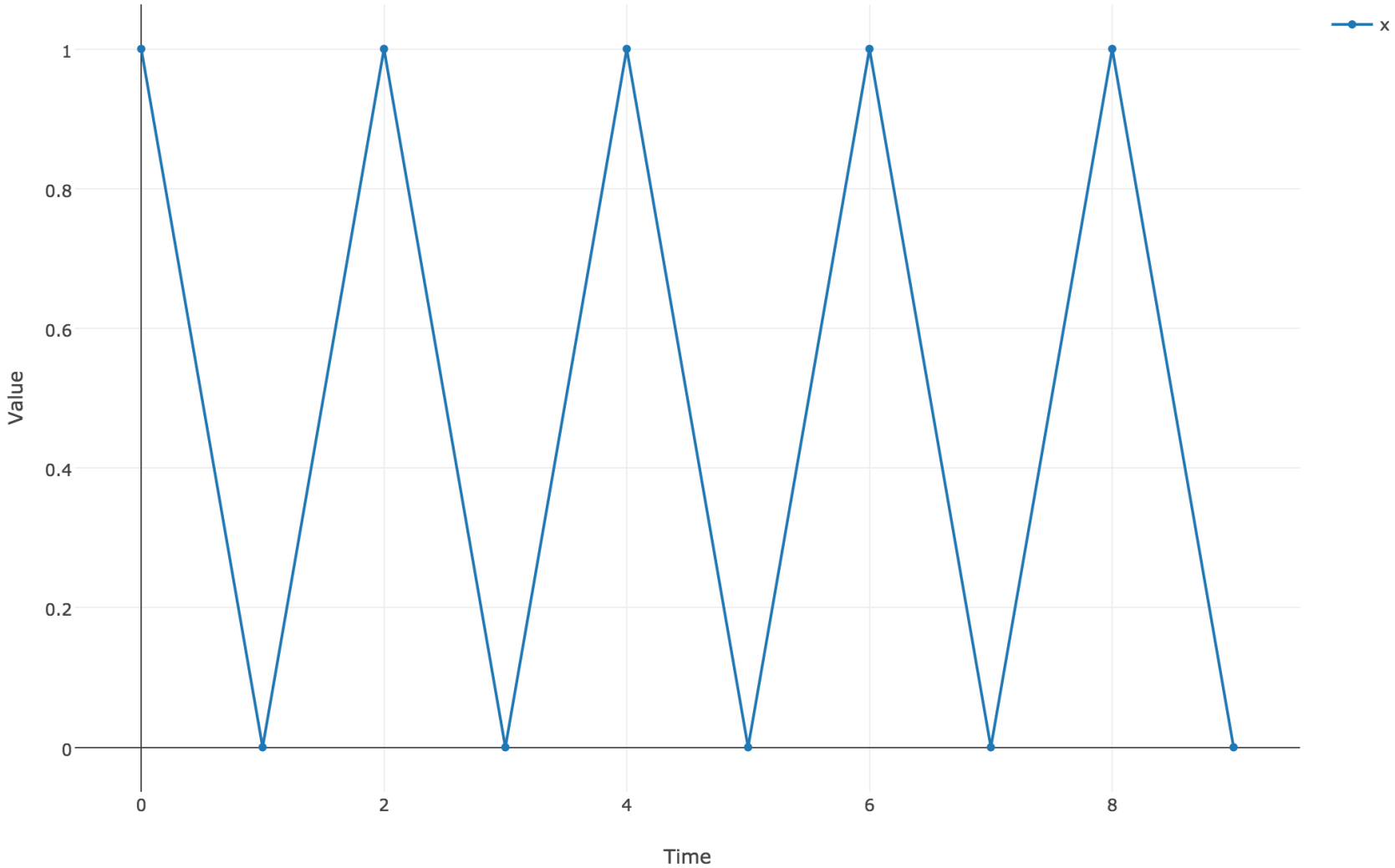
```
class A
end A

class B
instance variables
  x : int := 0;

operations

public op : () ==> ()
op() ==
for all i in set {1,...,10}
do x := x + if (i mod 2) > 0
            then 1
            else -1;
end B
```

Demo
Graph



Outline

Future Work

- Two way interaction through JSON
- Type based templates for complex types
- GUI builder

Summery

- Enabled link between UI and Model without model change
- Provided generic API for graphical interfaces using JSON

JSON Protocol

RunModel

SetRootClass

GetFunctionInfo

GetModelInfo

Subscribe

Execute

ValueUpdate

StopServer

JSON Protocol

Messages

```
1  // Obtain model classes
2  {"type":"REQUEST","data":{"request":"GetClassinfo"}}
3  {"type":"CLASSINFO","data": ["A","B"]}
4
5  // Set current root class
6  {"type":"REQUEST","data":
7    {"request":"SetRootClass","parameter":"B"}
8  }
9  {"type":"RESPONSE","data":"OK"}
10
11 // Get available functions or operations
12 {"type":"REQUEST","data":{"request":"Getfunctioninfo"}}
13 {"type":"FUNCTIONINFO","data": ["op"]}
14
15 // Obtain state info of root class
16 {"type":"REQUEST","data":{"request":"GetModelinfo"}}
17 {"type": "MODEL", "data":
18   {" rootClass": "mm", "name":"", "type":"",
19     "children": [
20       {"name": "x", "type":" int", "children": []}
21     ]}
```

JSON Protocol

Messages continued

```
1  // Subscribe to a variable change from the root class
2  {"type":"SUBSCRIBE","data":{"variableName":"x"}}
3  {"type":"RESPONSE","data":"OK"}
4
5  // Start simulation
6  {"type":"REQUEST","data":
7    {"request":"RunModel","parameter":"op"}
8  }
9
10 // Receive value updates
11 {"type":"VALUE","data":
12   {"variableName":"x","type":"int","value":"1"}
13 }
14
15 {"type":"VALUE","data":
16   {"variableName":"x","type":"int","value":"0"}
17 }
18
19 // Stop server
20 {"type":"REQUEST","data":{"request":"StopServer"}}
```

Implementation

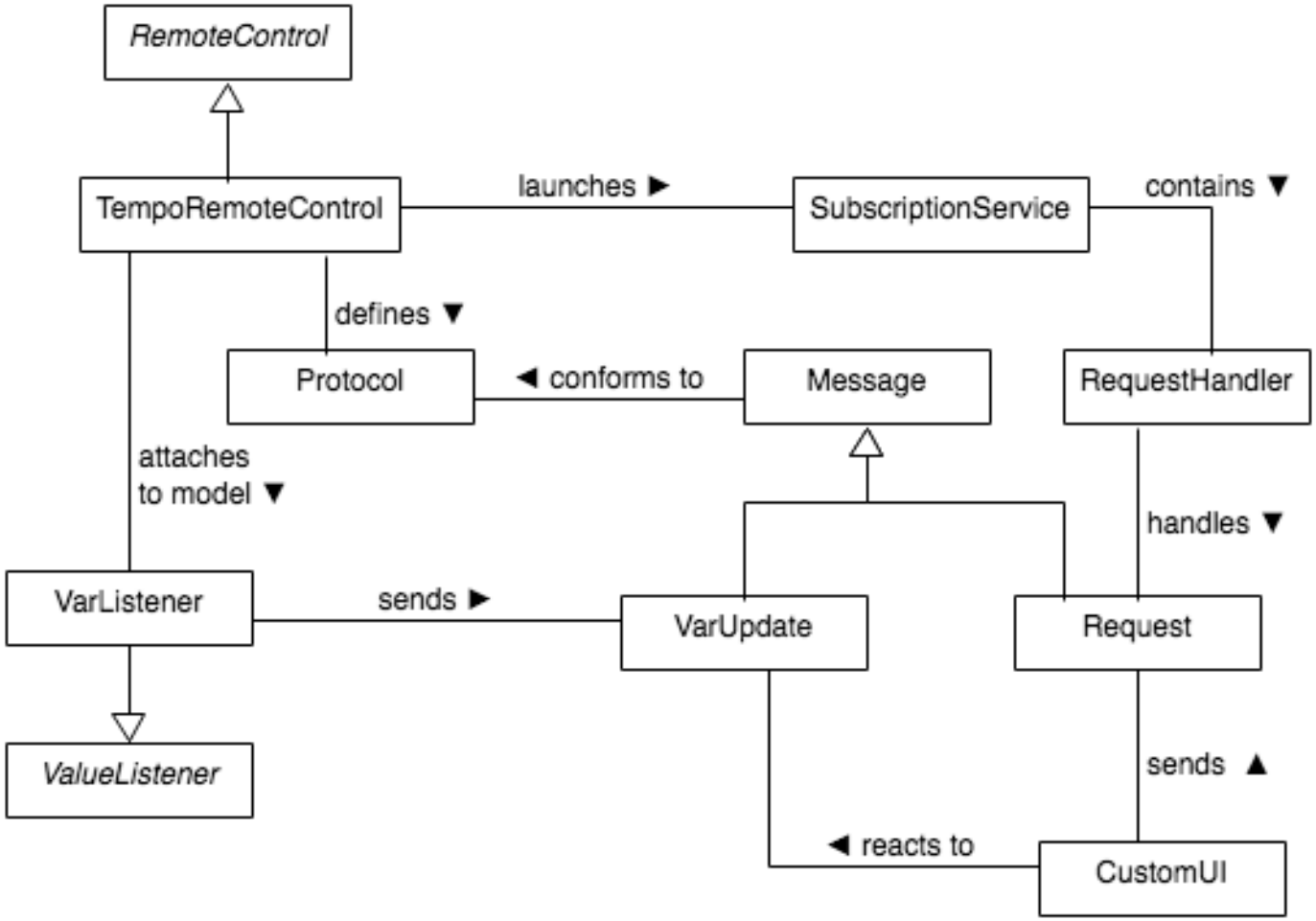


Figure: The main elements of the extension.