PCM & Runtime Adaptation
Motivation

Elastic System Design: How to model it?
Modelling…?

- How do we represent the system, i.e., what is its meta-model?
  - Design-Time Structure:
    Types, potential connectors, potential deployments, …
  - Initial State:
    Initial component instances, connectors, deployment
  - Runtime State:
    Current configuration, component instances, actual deployments, *environment sensor data*

- How do we model the system changes?
  One solution: graph transformation systems (GTS)

- How do we check properties of these systems?
  One solution: graph-based model checking
Views and Viewpoints

- When defining a new modelling language
  - define viewpoint
  - and views

- Viewpoint
  - Modelled system aspect
  - Set of views for a specific purpose

- Views
  - Model system parts with specific focus
  - Defines notation and filter criteria
Views and Viewpoints

Static Viewpoint

- Potential component structure, types
- Monitoring probe locations

Adaptation Viewpoint

- Initial state view
- Adaptation rules view
Static Viewpoint

System Type View + Monitoring Specification

LoadBalancerNode

Load Balancer

ServerNode[i]

Server[i]

i:1..*<<leastUtil>>

<<monitoring>> ArrivalRate

<<monitoring>> Utilisation[i]
Initial State View

```
ln:LoadBalancerNode
  lb:Load Balancer

sn[1]:ServerNode[i]
  s[1]:Server[i]
```

Dashed line connecting `lb` and `s[1]` indicates the relationship.
Adaptation View

\[\text{lb:LoadBalancerNode} \quad \text{sn:ServerNode} \quad \text{s:Server} \]

\[<<\text{condition}>> \]
\[\exists i: \text{Utilisation}[i] > \text{ut} \]

\[<<\text{++}>> \]
\[\text{sn2:ServerNode} \quad \text{s2:Server} \]

\[<<\text{timing}>> \]
\[\text{clock}<10s\]
Adaptation Views

- Adaptation views are graph transformations
  - transform the Model@Runtime
  - span a graph language together with the initial state graph
- Special formalisms and tools exist
  - GTS
  - Story Charts (GTS with control flow)
  - Timed Story Charts
  - Prob. Timed Story Charts
- State space checker
Self-Healing Rules – Possible Parts

Context
(matching the runtime model)

s1:Server

<<assumption>>
Application Delay = 1 sec

Trigger Condition
(often refers to the monitoring model)

<<condition>>
Utilisation_CPU > 60%

Corrective Action
(updating runtime model and system)

s2:Server

<<expected>>
Utilisation_CPU < 50%
Deadline = 10 sec

Assumption in the “nothing changes” case
Control Loop

Update Monitoring

[Trigger Formula violated]

Apply Matching Graphreplacement

Predict Change Impact

[not ok] / rollback

[ok] /

Apply Matching Graphreplacement
Conclusion

Adaptive systems need new (architectural?) viewtypes

Static views is extended by additional information
- Cardinalities
- Connector types
- Monitoring locations

Adaptation views are a kind of graph grammar
- In-place model transformations, i.e., GTS
- Initial state

Future work
- Use such models to implement, predict, model-check (?), and manage systems at run-time