Software Engineering for Self-Aware Computing

Samuel Kounev

University of Würzburg

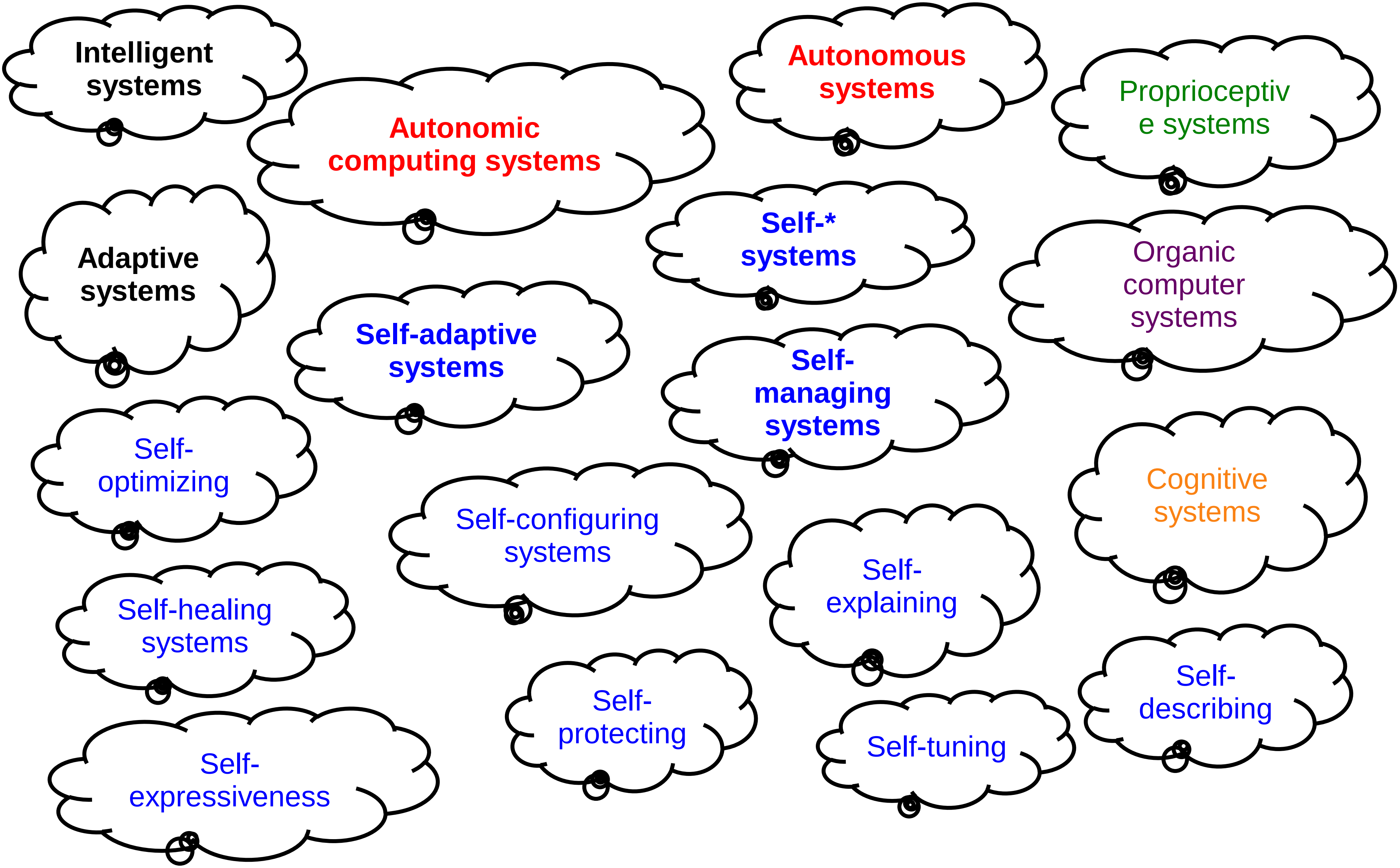
http://se.informatik.uni-wuerzburg.de/

Jan 19, 2015
Dagstuhl Seminar 15041:
“Model-driven Algorithms and Architectures for Self-Aware Computing Systems”

Agenda

- Self-aware computing and related terms
- Models in software engineering
- Modeling examples for self-aware computing
- Open issues and challenges
- Vision

Self-Aware Computing Systems?



Agarwal et al.

- **Def (Self-Aware):**
  - **Introspective**: can observe and optimise their own behaviour,
  - **Adaptive**: can adapt to changing needs of applications running on them,
  - **Self-healing**: can take corrective action if faults appear whilst monitoring resources,
  - **Goal-oriented**: attempt to meet user application goals,
  - **Approximate**: can automatically choose the level of precision needed for a task to be accomplished.

A. Agarwal, J. Miller, J. Eastep, D. Wentziaff, and H. Kasture, "Self-aware computing," MIT, Tech. Rep. AFRL-RI-RS-TR-2009-161, 2009.

SElf-awarE Computing (SEEC) Project

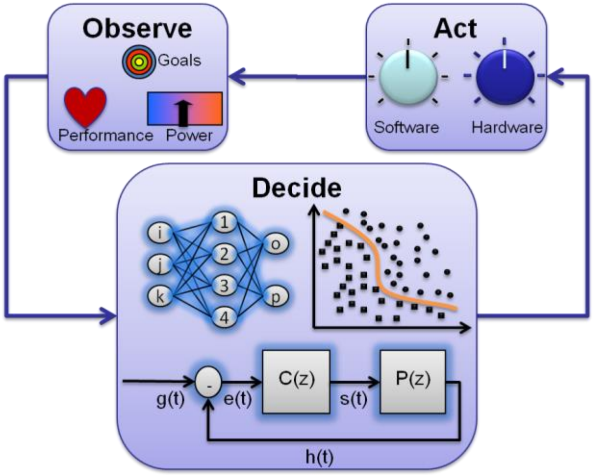
- **Def (self-aware):** Understand high-level goals and automatically adapt to meet those goals online
- Presence of **observe-decide-act (ODA) loops** in all system layers – hardware, compilers, OS, and applications
- Applications specify goals, system software specifies possible actions, and the SEEC framework decides how to use the available actions to meet the goals

SEEC: A General and Extensible Framework for Self-Aware Computing

by H. Hoffmann, M. Maggio, M. Santambrogio, A. Leva, and A. Agarwal

MIT CSAIL Technical Report, MIT-CSAIL-TR-2011-046, November 2011. (doi)

Project was named one of ten "World Changing Ideas" by Scientific American



ODA loop in the self-aware computing model

ASCENS EU Project

- Individual components and ensembles of components that are
  - **self-adaptive:** able to properly react on need by self-tuning their internal behavior and/or structure in an autonomic way –
  - **self-aware:** able to recognize the situations of their current operational context requiring self-adaptive actions
- Awareness of
  - not simply "what I am and what is happening in the world", but also
  - "what I could become and how the world could change accordingly"

On Self-adaptation, Self-expression, and Self-awareness in Autonomic Service Component Ensembles by F. Zambonelli, N. Bicocchi, G. Cabri, L. Leonardi, M. Puviani
2011 Fifth IEEE Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 3-7 Oct. 2011, DOI: 10.1109/SASOW.2011.24

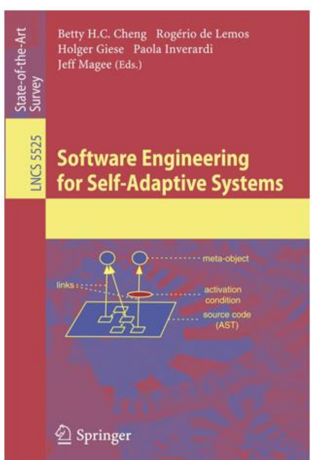
EPiCS EU Project

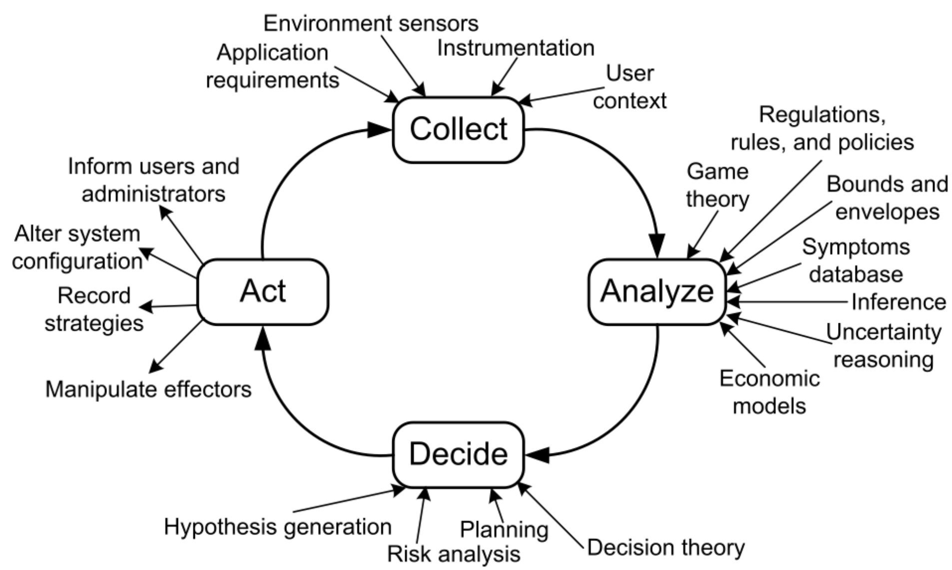
- Engineering Proprioception in Computing Systems
  - collect and maintain information about their state and progress, which enables them to reason about their behaviour (**self-awareness**)
  - and utilise this knowledge to effectively and autonomously adapt their behaviour to changing conditions (**self-expression**)

SEAMS Community

self-adaptive.org

- „Software Engineering for Self-Adaptive Systems“ Community
- **Def (self-adaptive systems):**
  - adapt at run-time to changing user needs, system intrusions or faults, changing operational environment, and resource variability
  - can configure and reconfigure themselves, augment their functionality, continually optimize themselves, protect themselves, and recover themselves, while keeping most of their complexity hidden from the user and administrator
- Generic Control Loop Model





Descartes DFG Project

Def (self-aware): possess, and/or are able to acquire at run-time, three properties, ideally to an increasing degree the longer they are in operation:

1. **Self-reflective**: Aware of their operational goals and of the aspects of their architecture and environment relevant to achieving these goals,
2. **Self-predictive**: Able to predict the effect of dynamic changes, as well as predict the effect of possible adaptation actions,
3. **Self-adaptive**: Proactively adapting as the environment evolves in order to ensure that their operational goals are continuously met.

http://descartes.tools

S. Kounev, F. Brosig, N. Huber, and X. Zhu. "Model-Based Approach to Designing Self-Aware IT Systems and Infrastructures“. Under review for IEEE Computer – available on request, 2015.

S. Kounev, F. Brosig, and N. Huber. „The Descartes Modeling Language“. Technical report, Department of Computer Science, University of Wuerzburg, October 2014. [bib | http | http | .pdf]

S. Kounev. Engineering of Self-Aware IT Systems and Services: State-of-the-Art and Research Challenges. In *8th European Performance Engineering Workshop (EPEW'11), Borrowdale, UK, October 12-13*, 2011. (Keynote Talk). [.pdf]

Goals

„The indisponsable

first step to getting the

things you want out of life

is this: decide what you

want".

-- Ben Stein

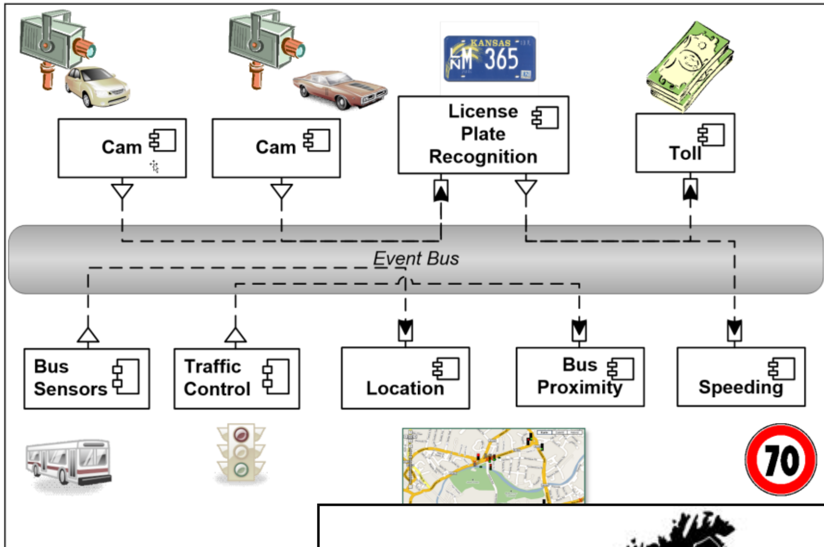
[Neshan Naltchayan, Wikipedia]

Stresses Explicit Awareness of

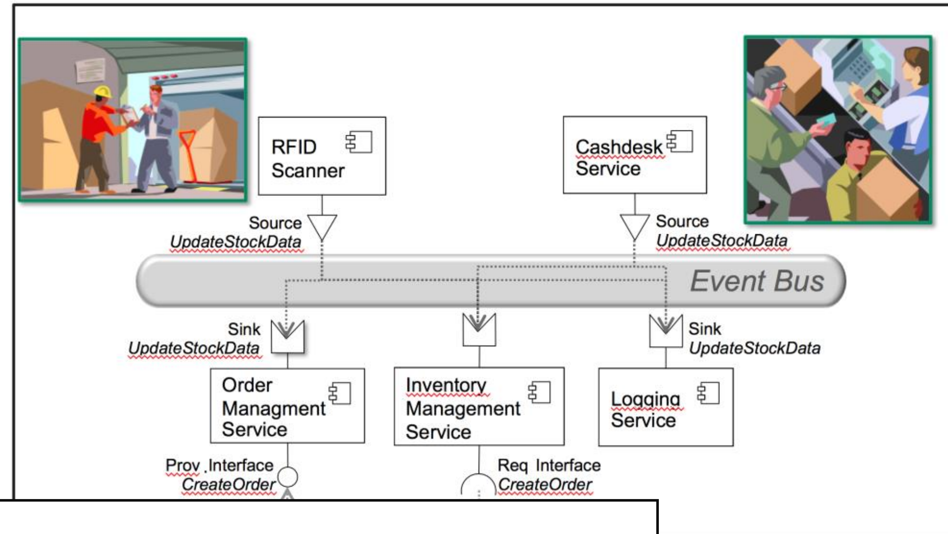
- What are my high-level (application) goals?
- What aspects of my **architecture** and my **environment** are relevant for achieving my goals?
- How well am I currently meeting my goals?
- What **changes** are **anticipated** that will have impact on my goals?
- What **possible adaptation actions** can I undertake? What would be the **impact of an adaptation** on my goals?
- How can I find a suitable adaptation tactic in time and **proactively adapt** to continue fulfilling my goals?

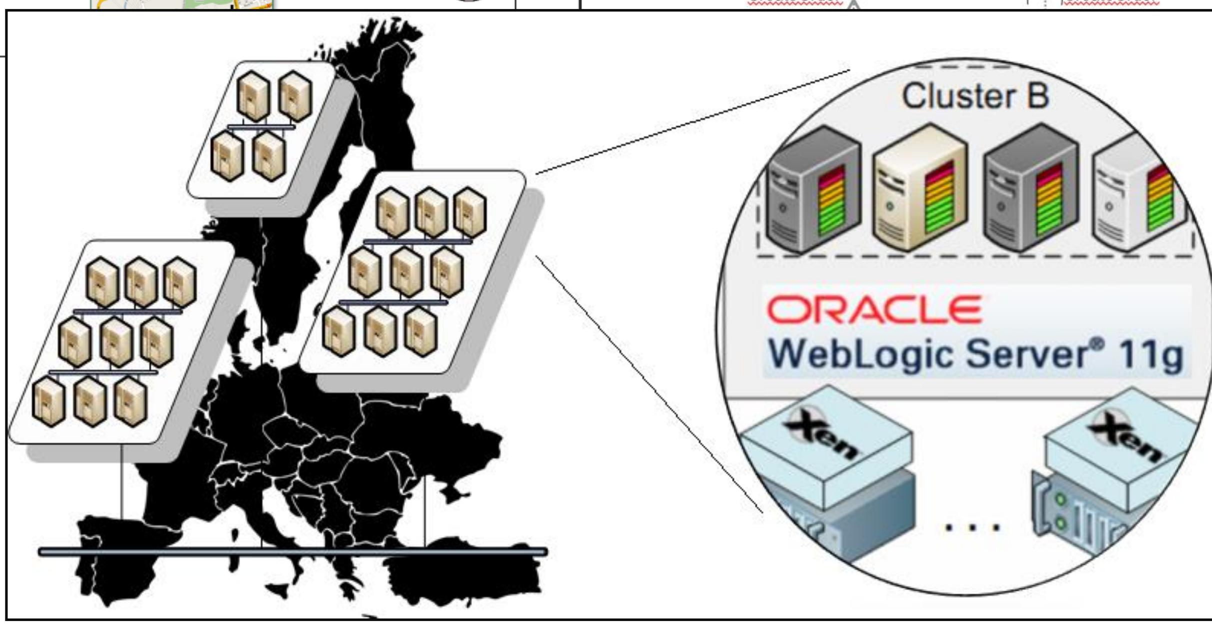
Examples of Modern Systems

Traffic Monitoring System



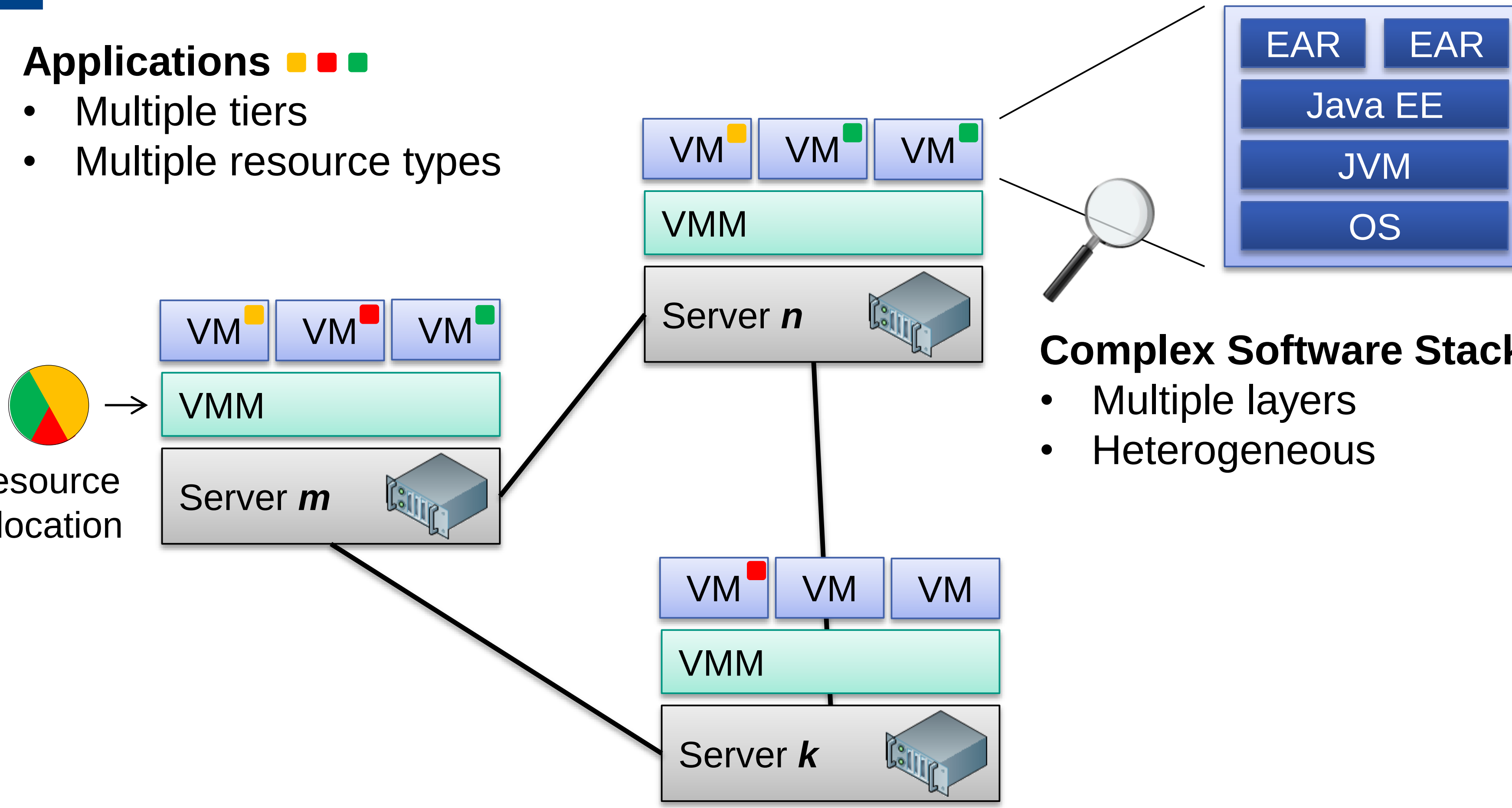
Inventory Management System





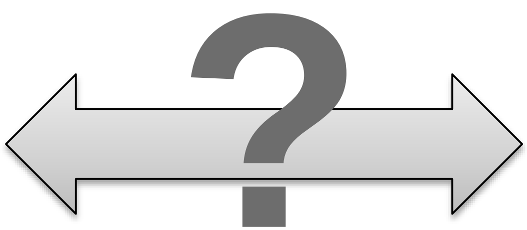
Semantic Gap Problem

Applications
- Multiple tiers
- Multiple resource types



Complex Software Stacks
- Multiple layers
- Heterogeneous

High-level Application Goals (e.g., SLOs)



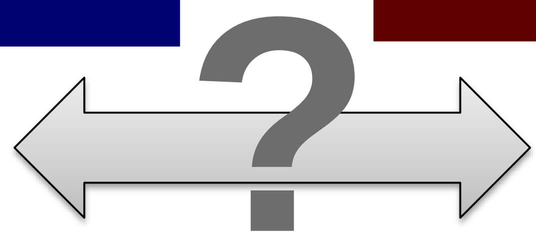
Configuration of System Components, Layers & Tiers

Semantic Gap Problem

- **Performance**
  - # requests that can be processed per sec > 1000
  - Response time of service x < 20 ms
  - Server utilization > 60% on average
  - ...
- **Availability / Reliability**
  - Time to recover after a server failure < 1 min
  - ...
- **Security**
  - Intrusion attempts are detected on time and prevented
- ...

- On which server to deploy software component y?
- How many vCPUs to allocate to VM n?
- How much memory to allocate to VM n?
- When exactly should a reconfiguration be triggered
- Which particular resources to scale / replicate / migrate?
- How quickly and at what granularity?

High-level Application Goals (e.g., SLOs)



Configuration of System Components, Layers & Tiers

Models in Software Engineering

Descriptive Models

- Capture relevant knowledge about the system and the environment in which it is running
- Describe selected aspects that have influence on the goal fulfilment

(Predictive) Analysis Models

- Allow to reason about the system behavior
- Predict the impact of changes on the goal fulfilment

Descriptive Models

Meta-Model

Node : Class
name : String numCPUs : Integer numVMs : Integer …

instance of

describes

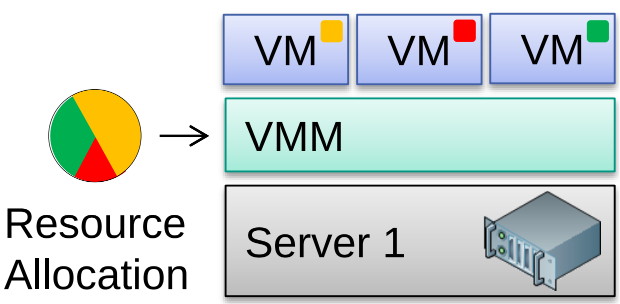
Model

Server 1 : Node
name = “Server 1” numCPUs = 3 numVMs = 3 …

instance of

describes

“Real world”



Meta-Models

A **meta-model** is a model precisely defining the parts and rules needed to create valid models.

It covers an abstract syntax, at least one concrete syntax, and static and dynamic semantics.

Parts → model elements

Rules → well-formed rules - when is a model valid?

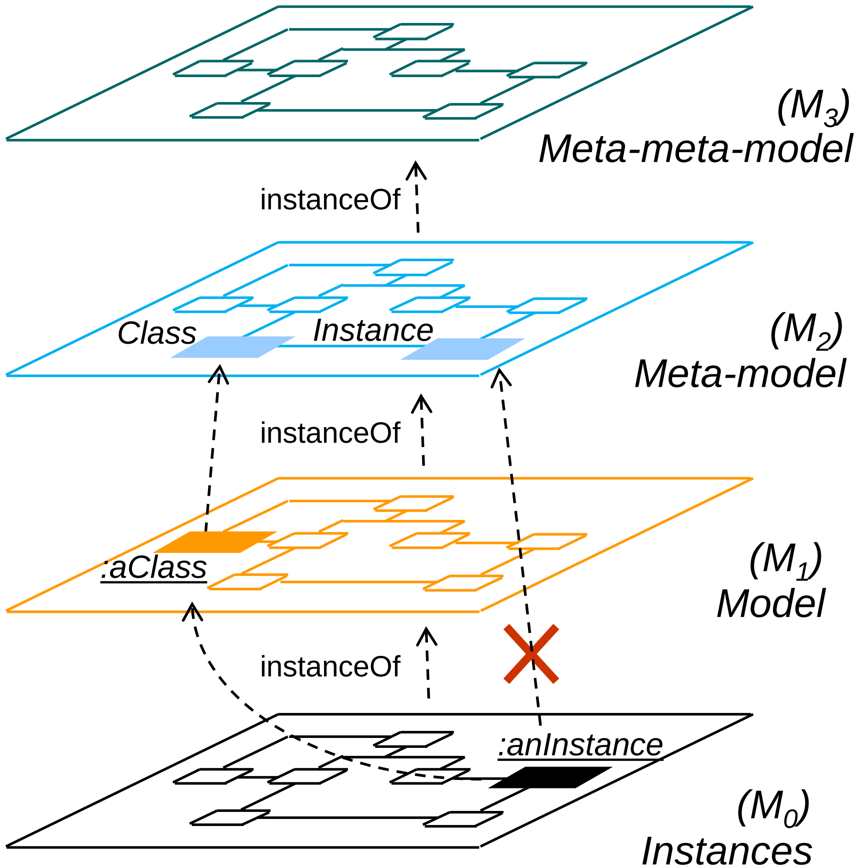
Abstract syntax: elements and their relations indep. of representation

Concrete syntax → representation of model-instances, e.g., in a file

Static semantics → semantics evaluable without executing the model

Dynamic semantics → what does the model mean/express?

Meta-Model Levels



OMG Four Level Infrastructure

Meta-Object Facility (MOF)

- Abstract language and framework for specifying, constructing, and managing technology neutral meta-models

- MOF is self-describing and has two parts
  - EMOF (Essential MOF, lightweight, subset of CMOF)
  - CMOF (Complete MOF, heavyweight)

- EMF (Eclipse Modelling Framework) can be seen as an implementation of EMOF
  - using the Ecore meta-model

- Example of a MOF-based meta-model → UML

Abstract vs. Concrete Syntax

Abstract

Server1 : Node
name = "Server 1" numCPUs = 3 numVMs = 3 …

Concrete

```
Node Server1 {
  String name = "Server 1"
  int numCPUs = 3;
  int numVMs  = 3;
  …
}
```

```
<Node
   nodeName="Server1">
   <attribute
      attributeName="name" value="Server 1"/>
   <attribute
      attributeName="numCPUs" value="3"/>
   <attribute
      attributeName="numVMs" value="3"/>
</Node>
```

```
Node Server1
(
   attributes
   (
      name    : "Server 1"
      numCPUs : 3
      numVMs  : 3
   )
)
```

Object Constraint Language (OCL)

- A declarative language for describing rules that apply to valid model instances

  "A constraint is a restriction on one or more values of an object-oriented model or system" [Warmer & Kleppe]

- Example:

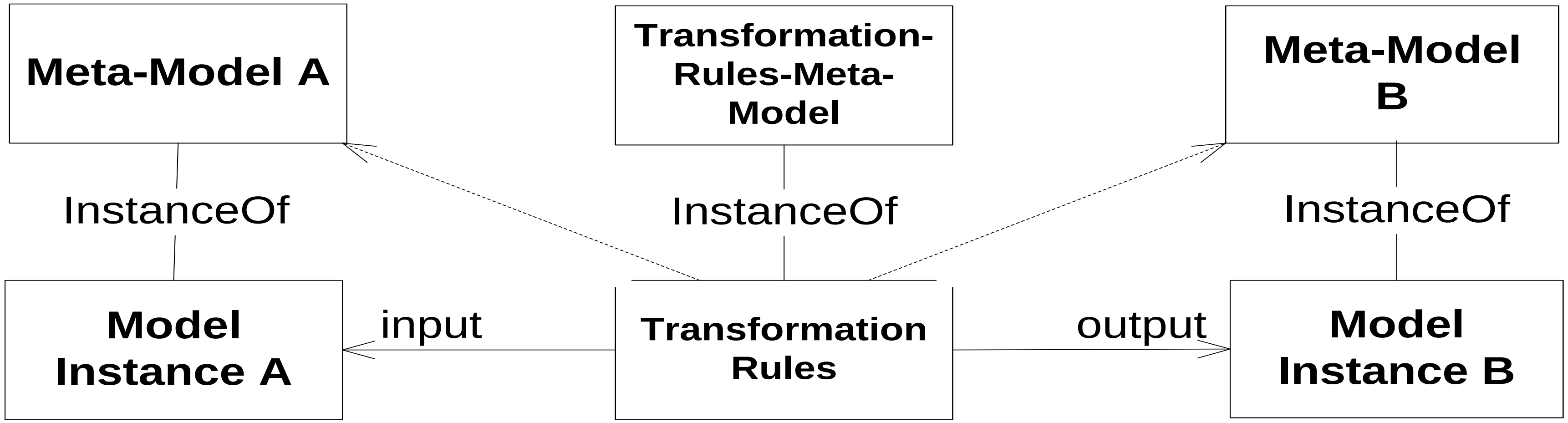
Node : Class
name : String numCPUs : Integer numVMs : Integer …

the attribute numCPUs of every Node must be greater than 0

```
context Node
inv CPUs: numCPUs > 0
```

Model-2-Model Transformations

- Transformations
  - Input: A model instance of meta-model **A**
  - Output: A model instance of meta-model **B**
  - Rules: How to transform meta-model elements of meta-model **A** into elements of meta-model **B**
  - Rule Engine: A system capable to execute the rules



Transformation Languages

imperative style
Xtend, QVT-O, Kermeta, XSLT…

declarative style
QVT-R, TGG…

supporting both paradigms
ATL, RubyTL, VIATRA…

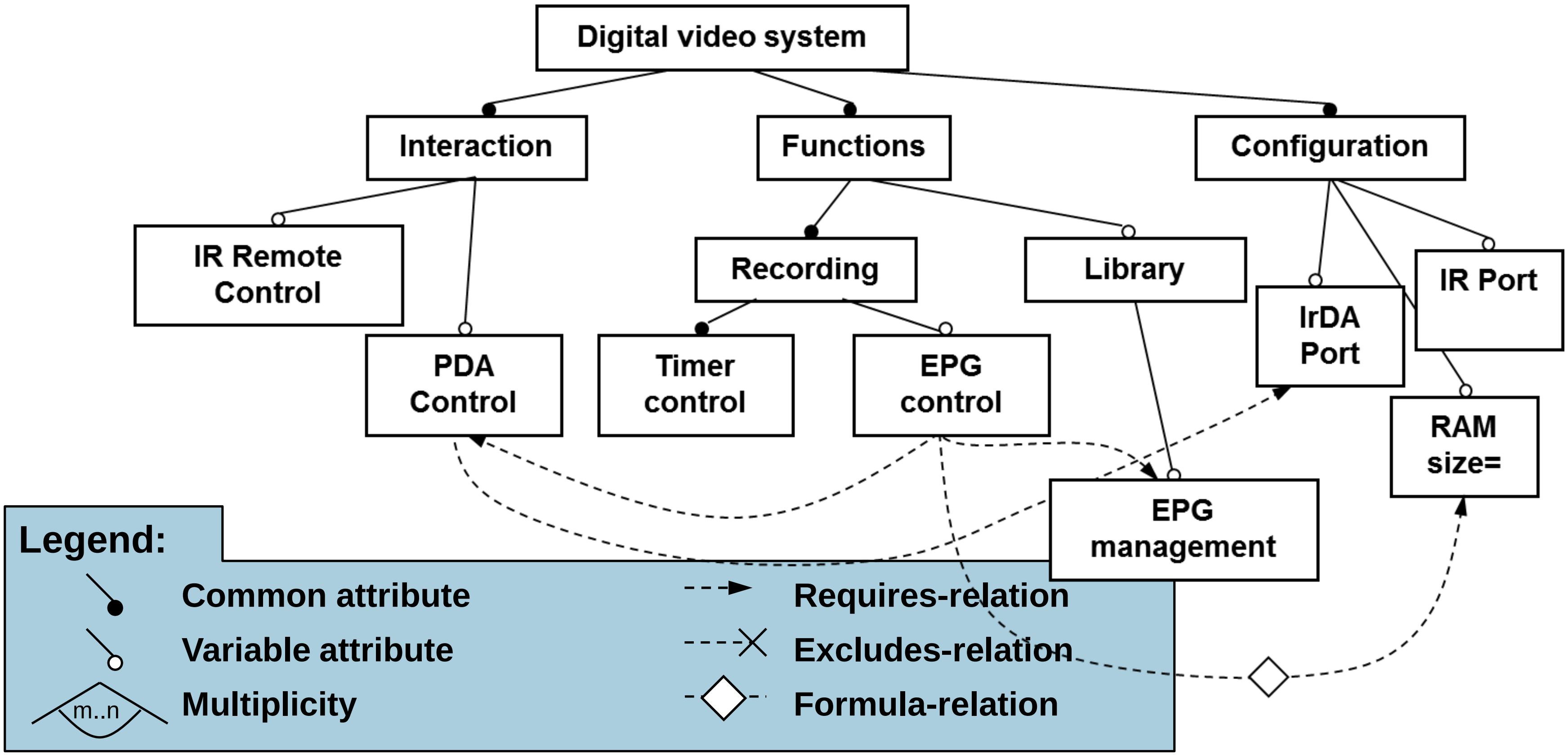
↔

general purpose
pragmatic

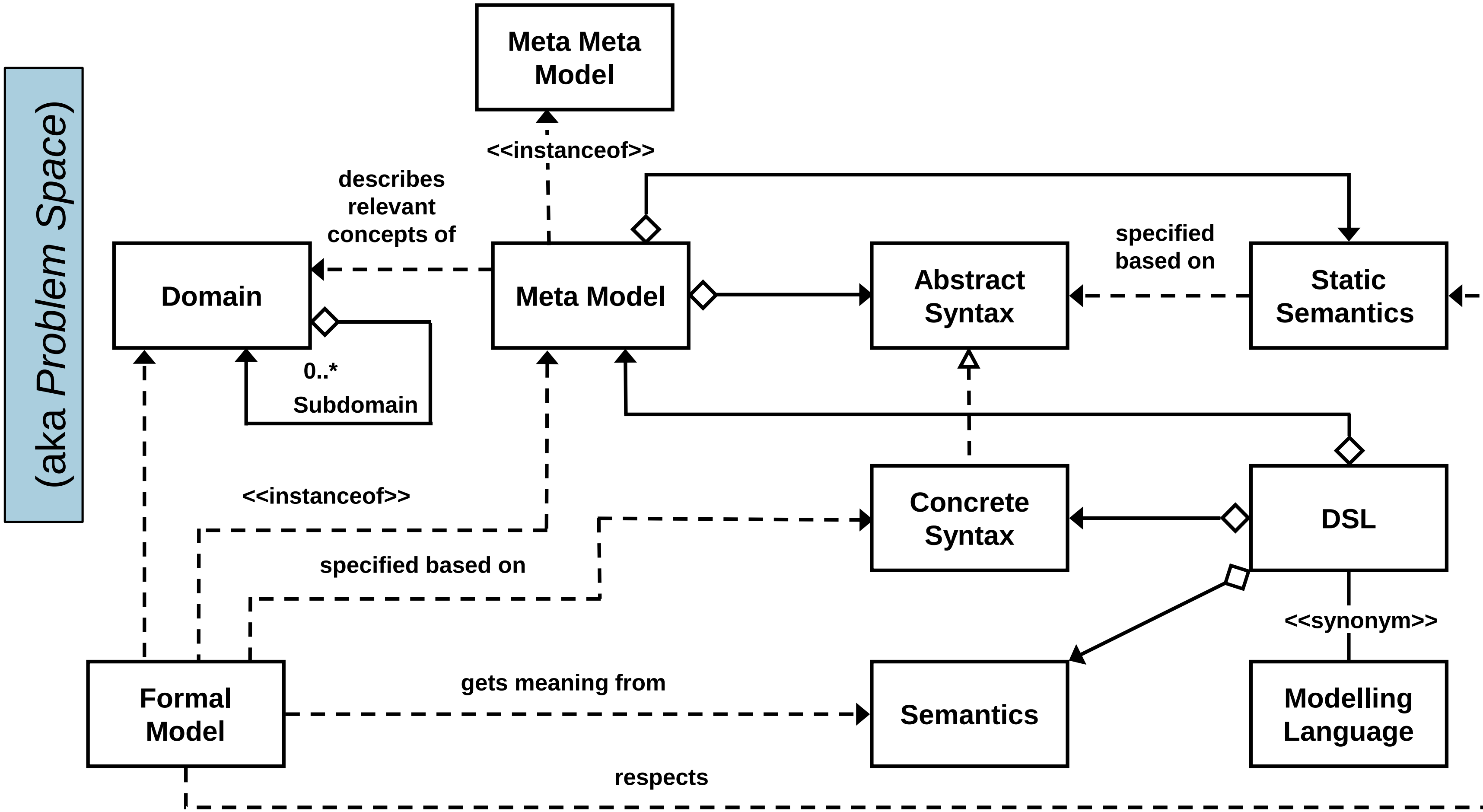
problem specific
formal and sometimes academic

Feature Models

- A feature is a choice you have
  - e.g. in a transformation
  - i.e. they model variation points that can be influenced via transformation parameters



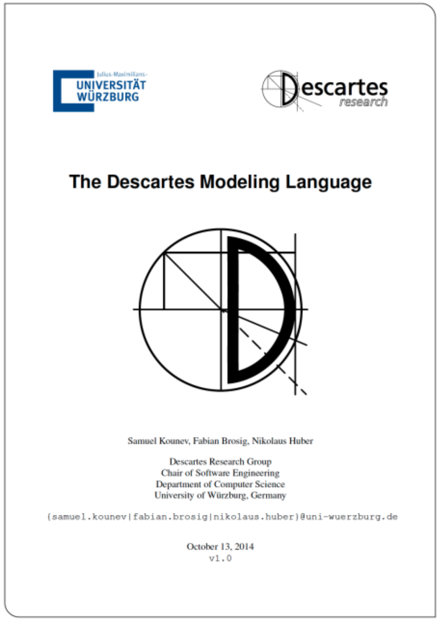
Concept Formation



cf. [Voe05]

Descartes Modeling Language (DML)

- Architecture-level modeling language for modeling QoS and resource management related aspects of IT systems and infrastructures
  - Prediction of the impact of dynamic changes at run-time
  - Current version focused on performance including capacity, responsiveness and resource efficiency aspects


Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Descartes
research

The Descartes Modeling Language

Samuel Kounev, Fabian Brosig, Nikolaus Huber

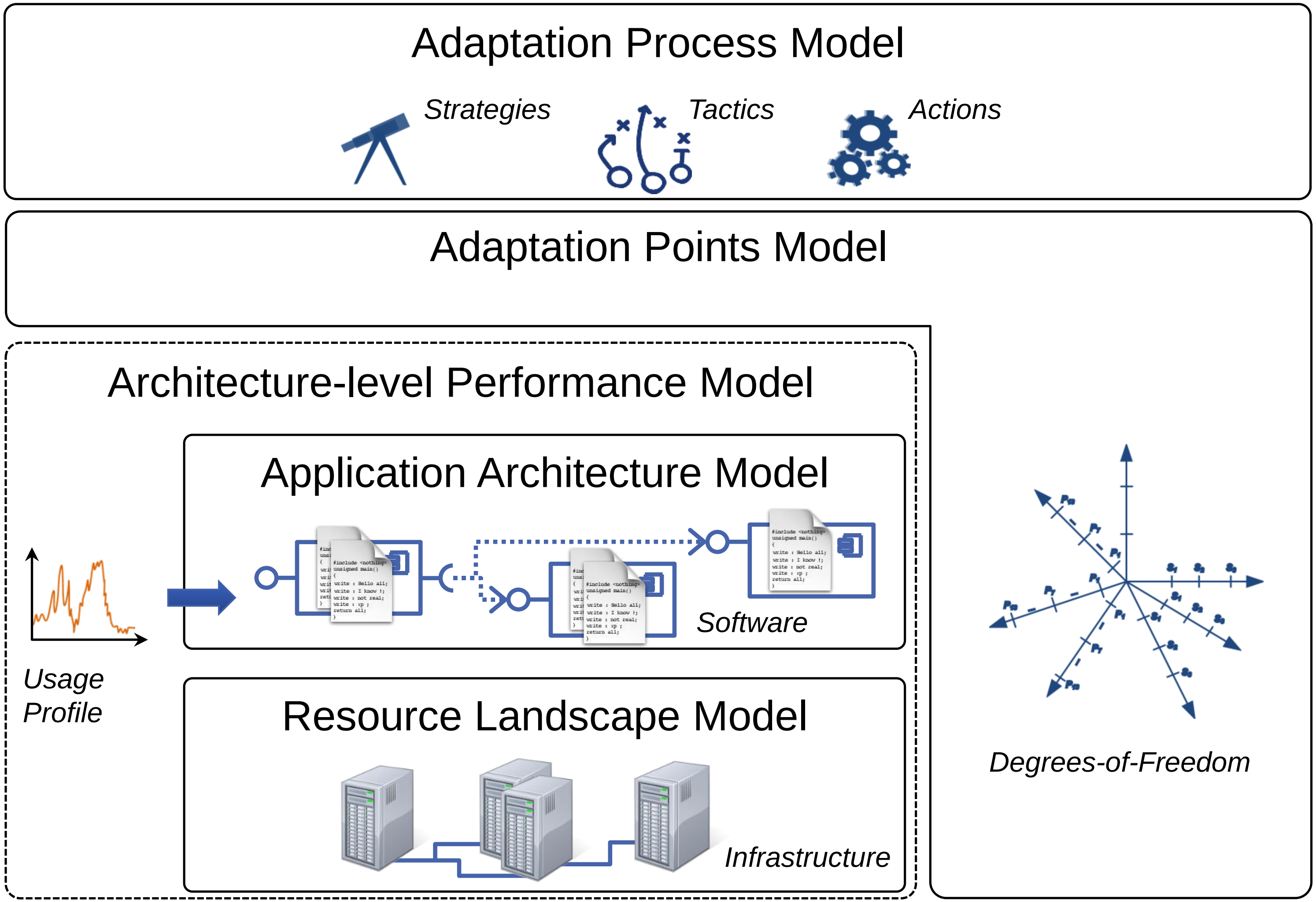
Descartes Research Group
Chair of Software Engineering
Department of Computer Science
University of Würzburg, Germany

{samuel.kounev|fabian.brosig|nikolaus.huber}@uni-wuerzburg.de

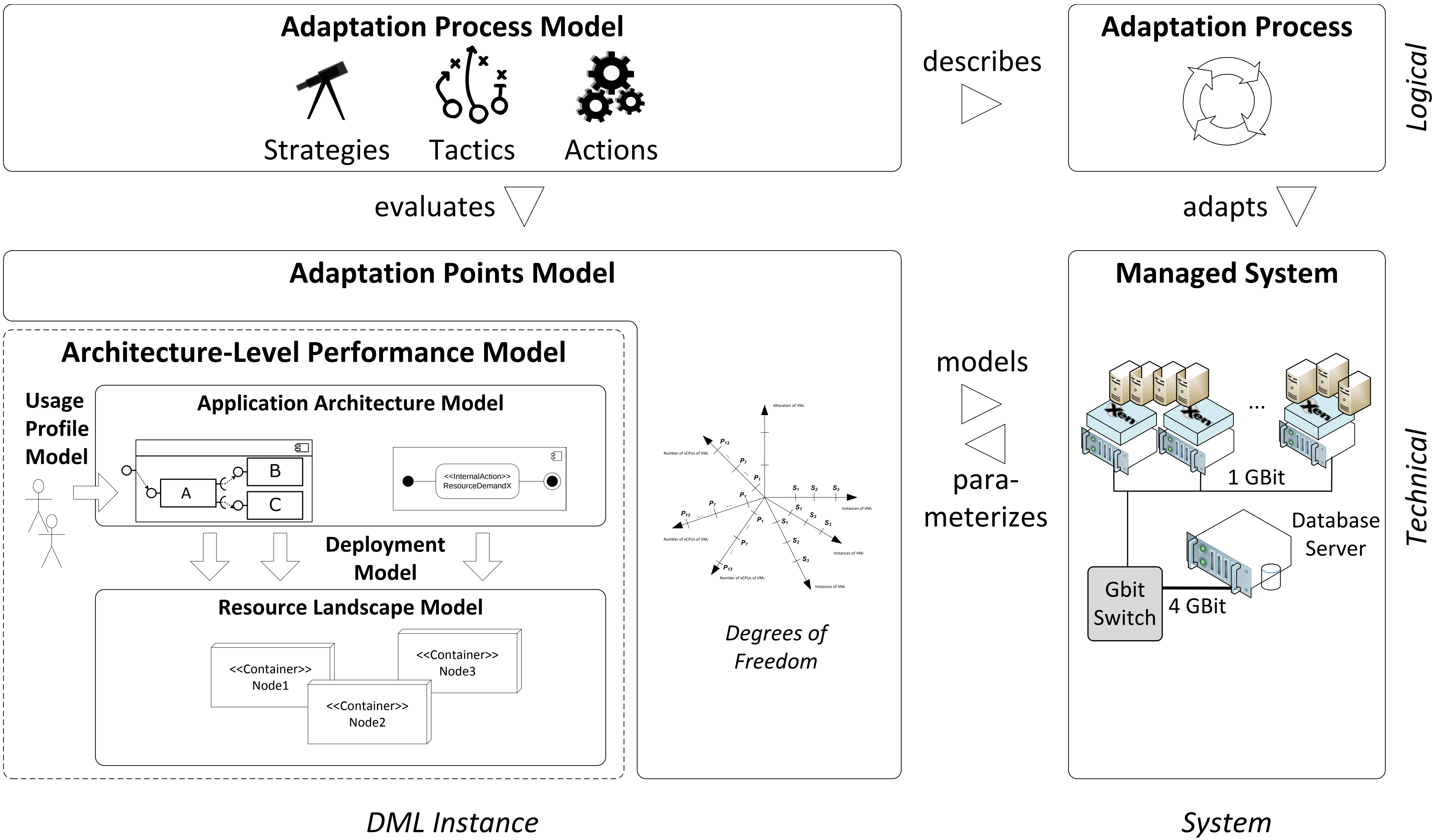
October 13, 2014
v1.0

http://descartes.tools/dml

Descartes Modeling Language (DML)



Big Picture


Adaptation Process Model
Strategies
Tactics
Actions
describes
Adaptation Process
Logical
evaluates
adapts
Adaptation Points Model
Managed System
Architecture-Level Performance Model
Usage Profile Model
Application Architecture Model
A
B
C
<<InternalAction>> ResourceDemandX
Deployment Model
Resource Landscape Model
<<Container>> Node1
<<Container>> Node2
<<Container>> Node3
Degrees of Freedom
models
para-meterizes
1 GBit
Database Server
Gbit Switch
4 GBit
...
Technical
DML Instance
System

References

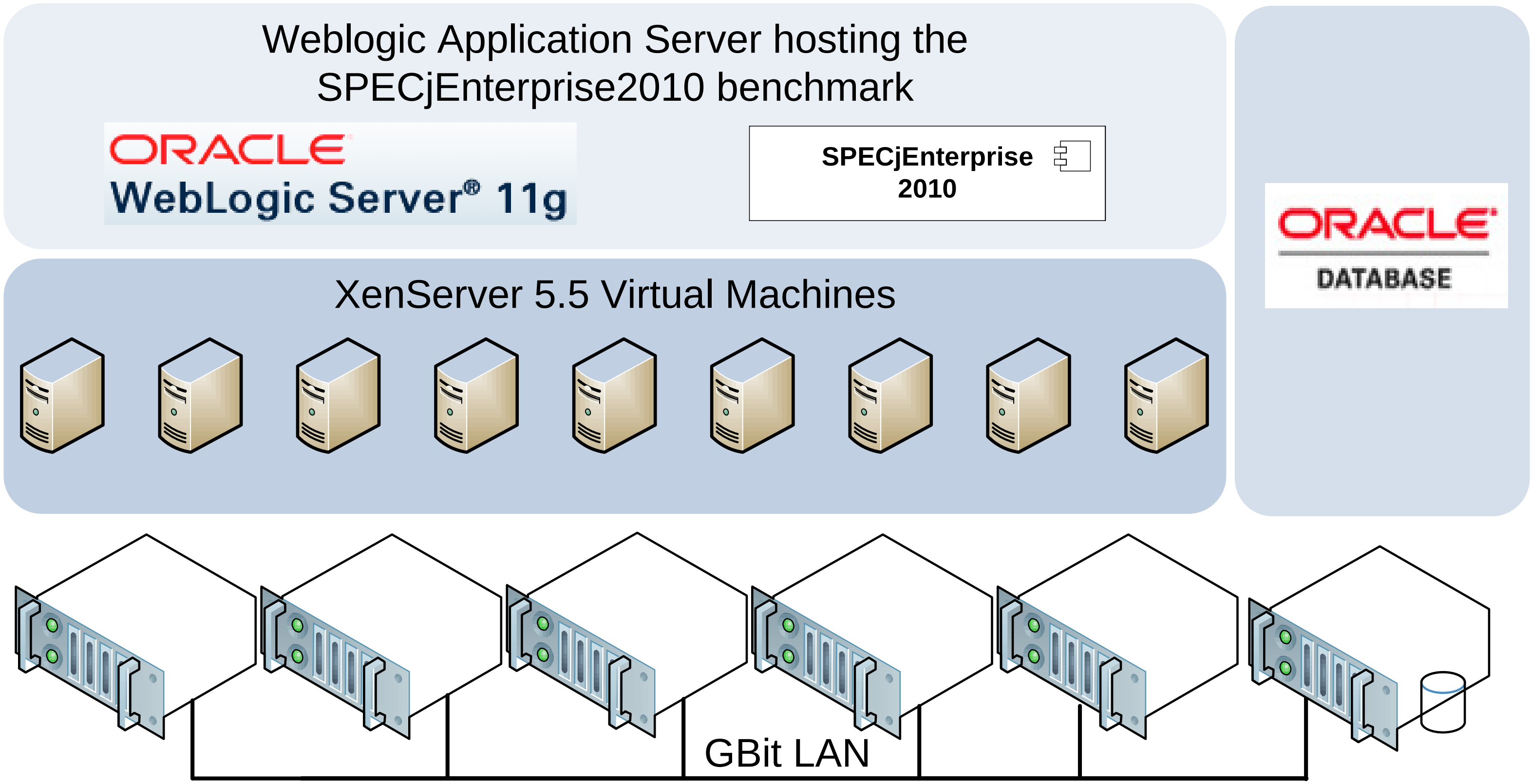
- S. Kounev, F. Brosig, N. Huber, and X. Zhu. **Model-Based Approach to Designing Self-Aware IT Systems and Infrastructures**. Under review. IEEE Computer Special Issue on Self-Aware and Self-Expressive Computing Systems, 2015. *Available on request.*
- S. Kounev, F. Brosig, and N. Huber. **The Descartes Modeling Language**. Technical report, Department of Computer Science, University of Wuerzburg, October 2014. [http | http | .pdf]
- F. Brosig, N. Huber, and S. Kounev. **Architecture-Level Software Performance Abstractions for Online Performance Prediction**. *Elsevier Science of Computer Programming Journal (SciCo)*, Vol. 90, Part B:71-92, 2014, Elsevier. [DOI | http | .pdf]
- N. Huber, A. van Hoorn, A. Koziolek, F. Brosig, and S. Kounev. **Modeling Run-Time Adaptation at the System Architecture Level in Dynamic Service-Oriented Environments**. *Service Oriented Computing and Applications Journal (SOCA)*, 8(1):73-89, 2014, Springer-Verlag. [DOI | .pdf]
- F. Brosig, P. Meier, S. Becker, A. Koziolek, H. Koziolek, and S. Kounev. **Quantitative Evaluation of Model-Driven Performance Analysis and Simulation of Component-based Architectures**. *IEEE Transactions on Software Engineering (TSE)*, 2014, IEEE, Preprint. [DOI | .pdf]
- F. Brosig, N. Huber, and S. Kounev. **Modeling Parameter and Context Dependencies in Online Architecture-Level Performance Models**. In *15th ACM SIGSOFT Intl. Symp. on Component Based Software Engineering (CBSE 2012), June 26-28, 2012, Bertinoro, Italy*, June 2012. [http | .pdf]
- N. Huber, F. Brosig, and S. Kounev. **Modeling Dynamic Virtualized Resource Landscapes**. In *8th ACM SIGSOFT Intl. Conf. on the Quality of Software Architectures (QoSA 2012)*, Bertinoro, Italy, June 25-28, 2012, pages 81-90. ACM, New York, NY, USA. June 2012. [DOI | http | .pdf]
- N. Huber, A. van Hoorn, A. Koziolek, F. Brosig, and S. Kounev. **S/T/A: Meta-Modeling Run-Time Adaptation in Component-Based System Architectures**. In *9th IEEE Intl. Conf. on e-Business Engineering (ICEBE 2012)*, Hangzhou, China, September 9-11, 2012, pages 70-77. IEEE Computer Society, Los Alamitos, CA, USA. September 2012. [DOI | http | .pdf]

References

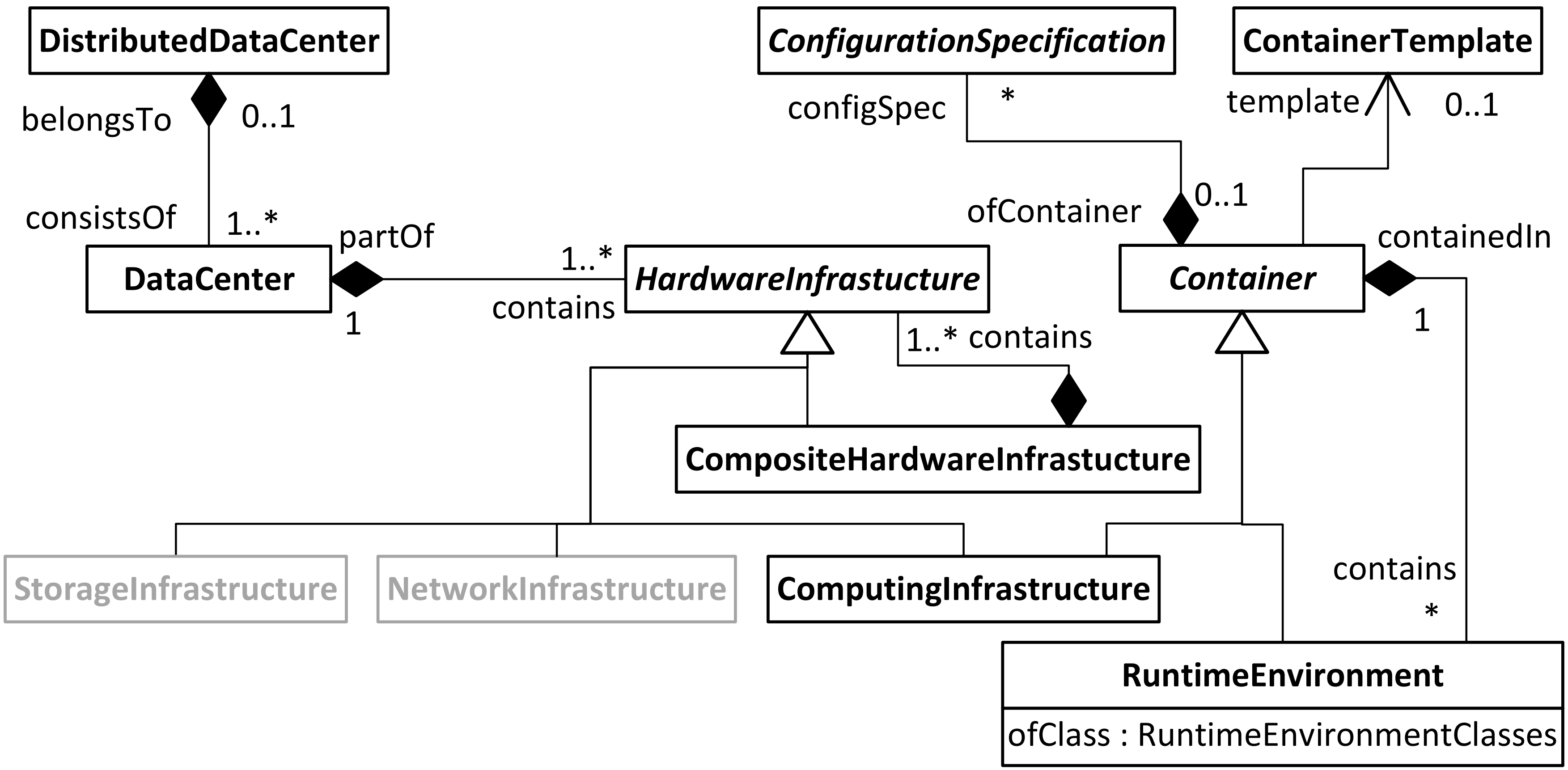
- **Fabian Brosig**. *Architecture-Level Software Performance Models for Online Performance Prediction*. PhD thesis, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany, 2014. [http | http]

- **Nikolaus Huber**. *Autonomic Performance-Aware Resource Management in Dynamic IT Service Infrastructures*. PhD thesis, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany, 2014. [http | http]

Example
(Resource Landscape)

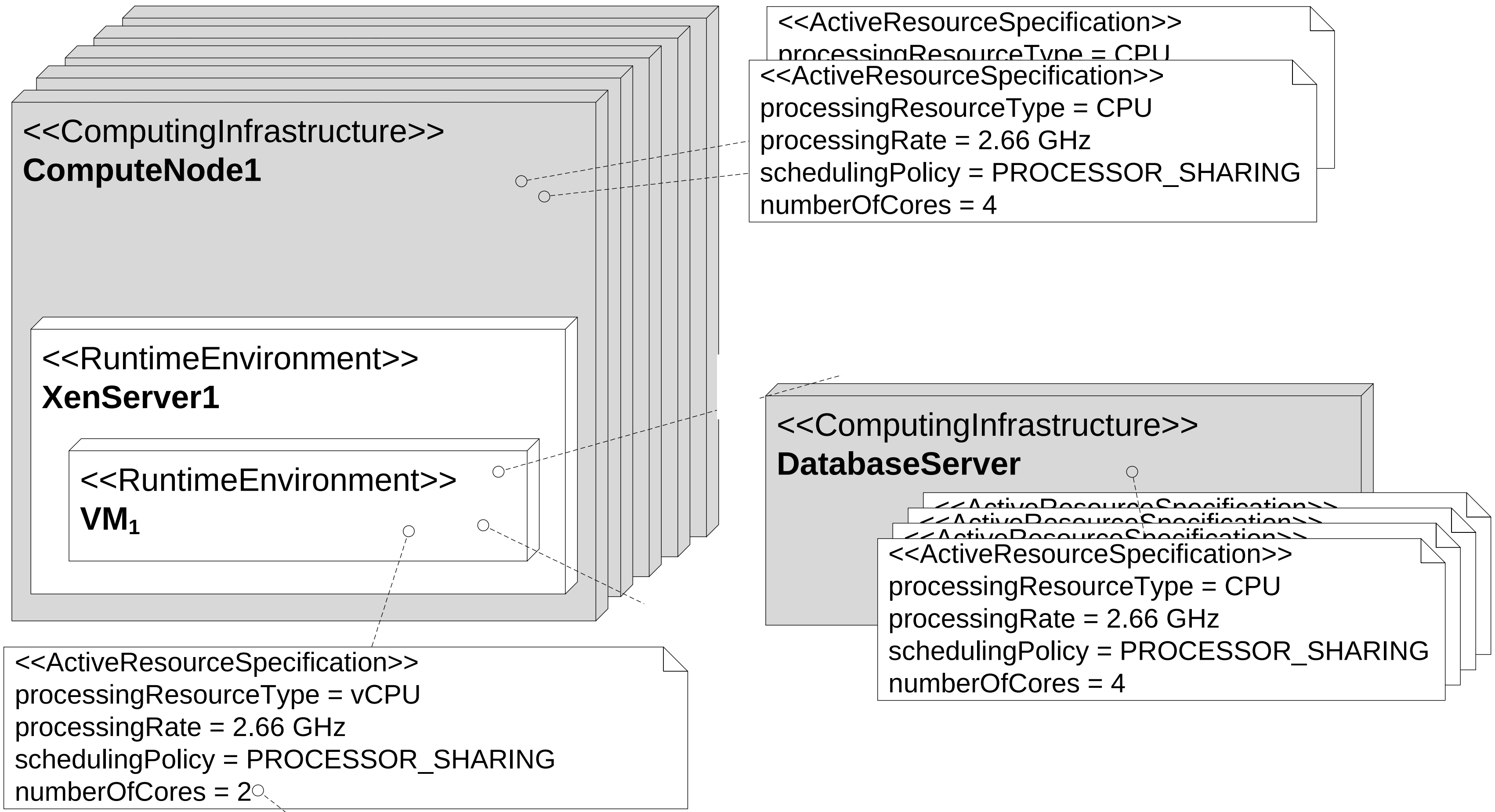


Resource Landscape Meta-Model
(Top Level Concepts)

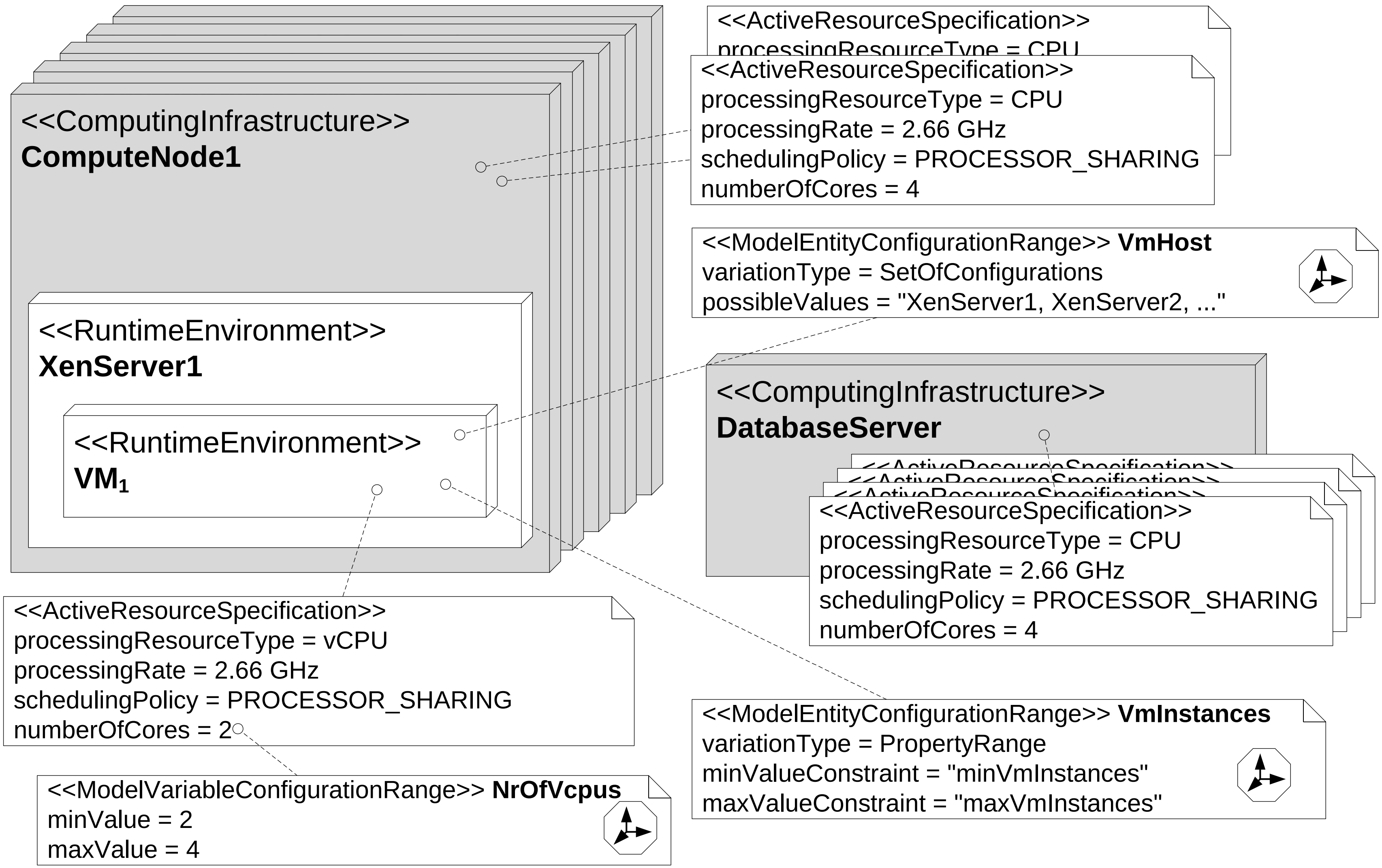


Example

(Resource Landscape Model)

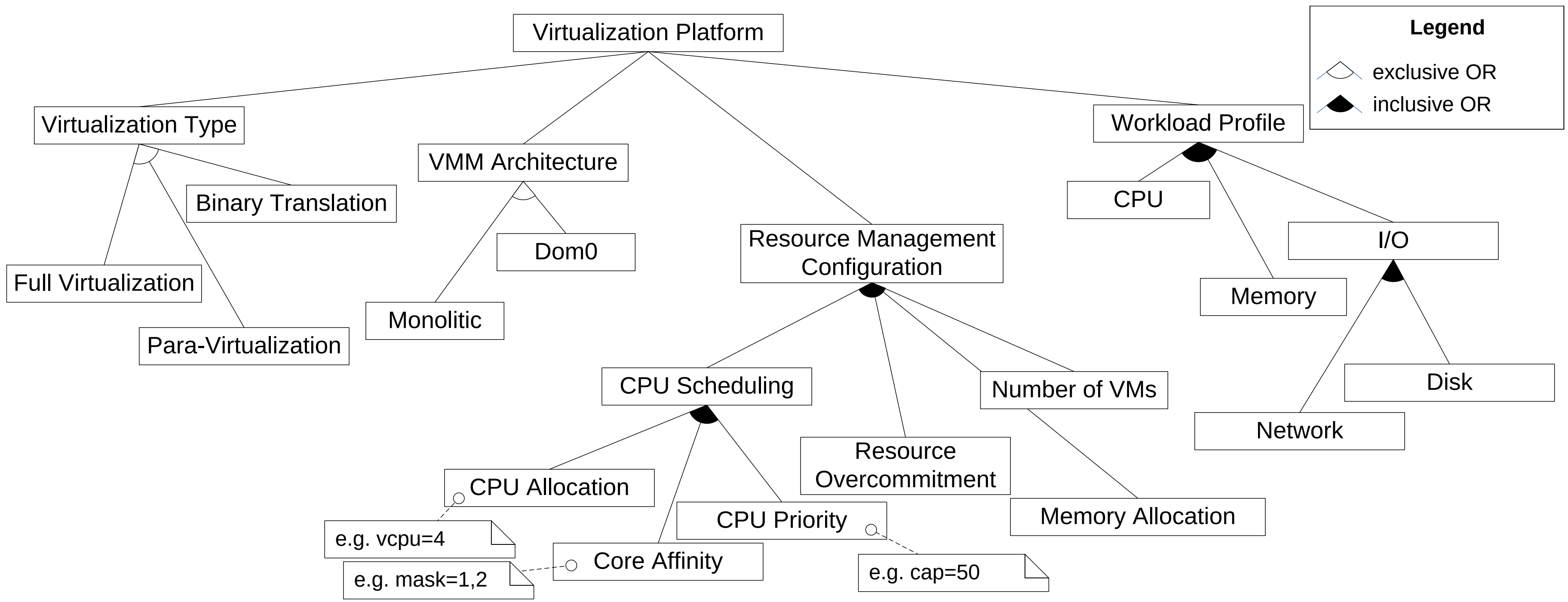


Example
(Resource Landscape Model) + (Adaptation Points Model)



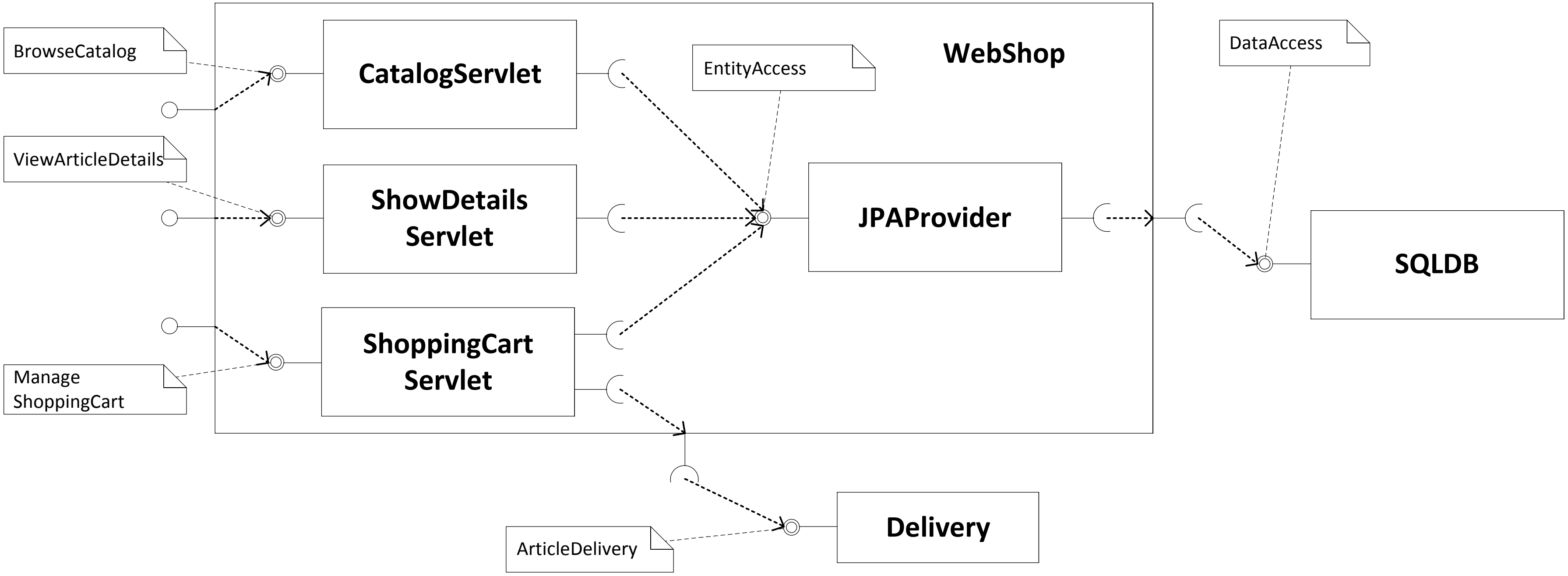
Example: Custom Configuration Model

(Feature Model for the Virtualization Platform)



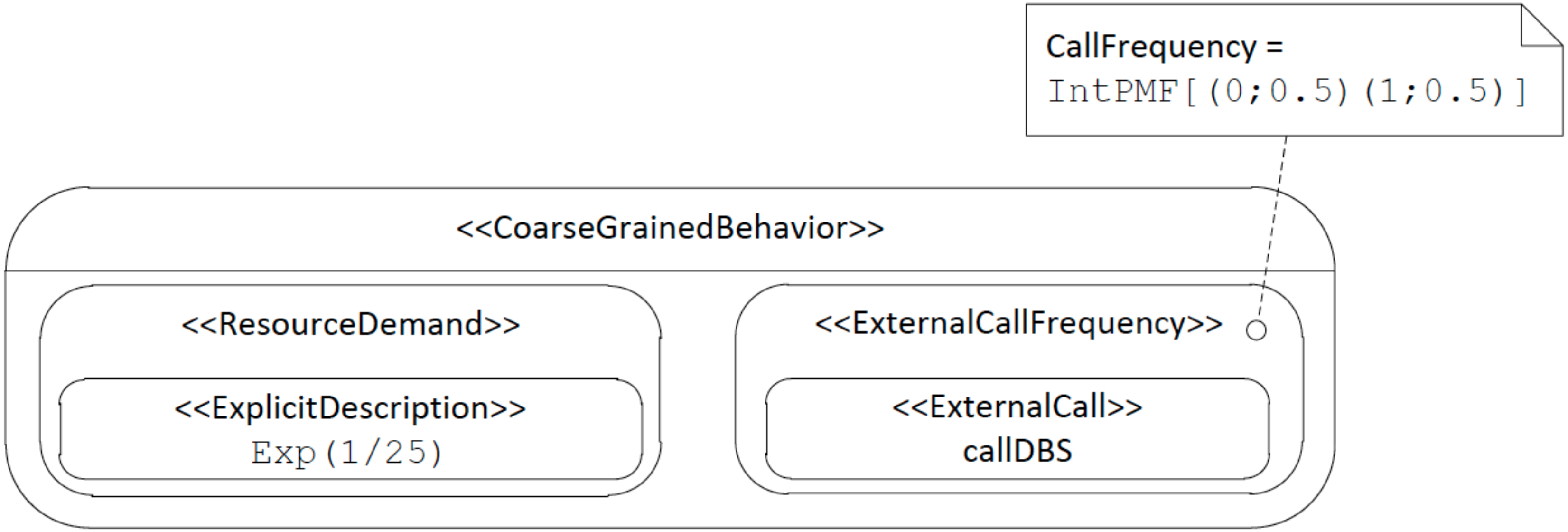
N. Huber, M. Quast, M. Hauck, and S. Kounev. **Evaluating and Modeling Virtualization Performance Overhead for Cloud Environments**. *International Conference on Cloud Computing and Services Science (CLOSER 2011), Noordwijkerhout, The Netherlands*, May 7-9, 2011. Best Paper Award.

Example
(Application Architecture Model)

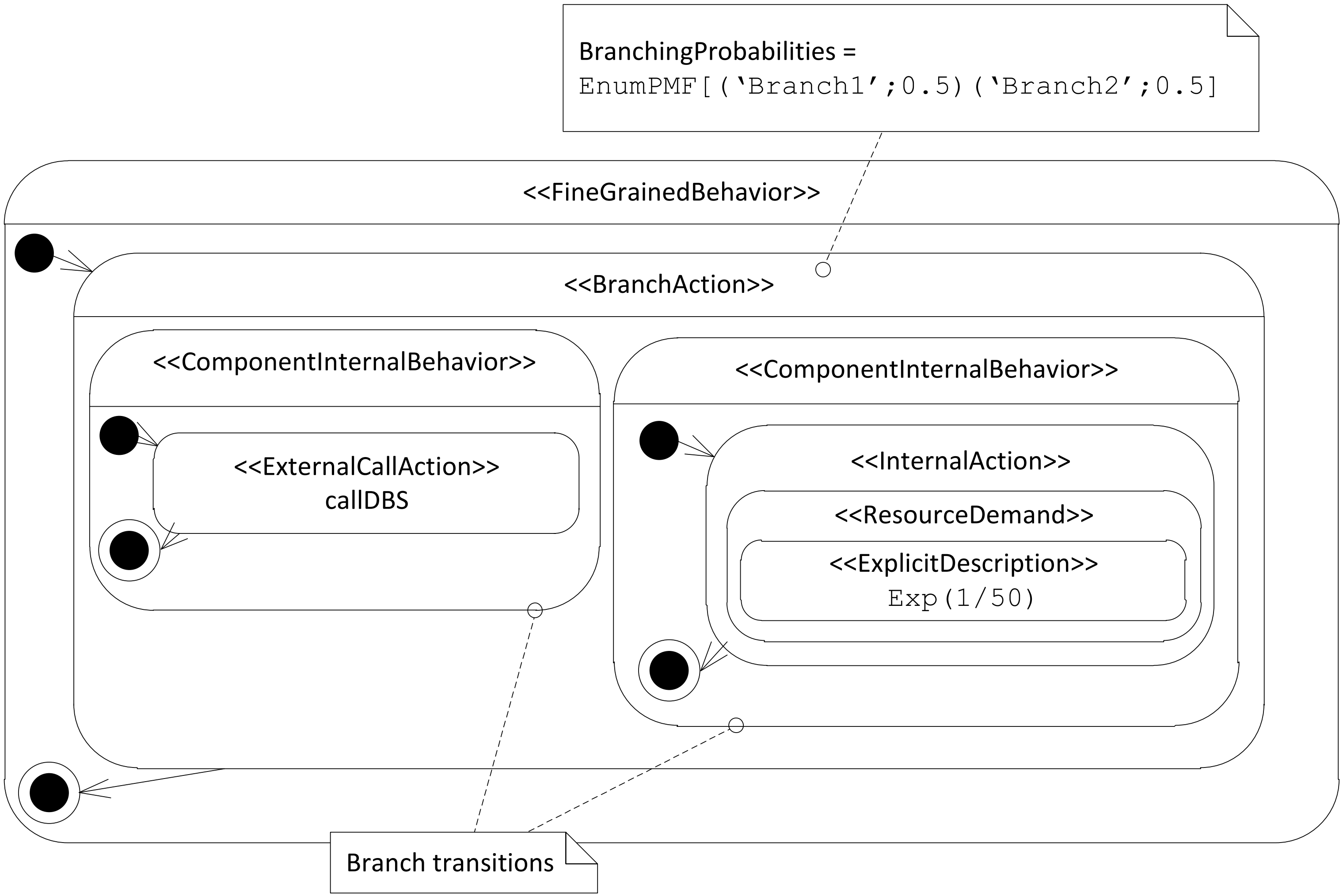


Example

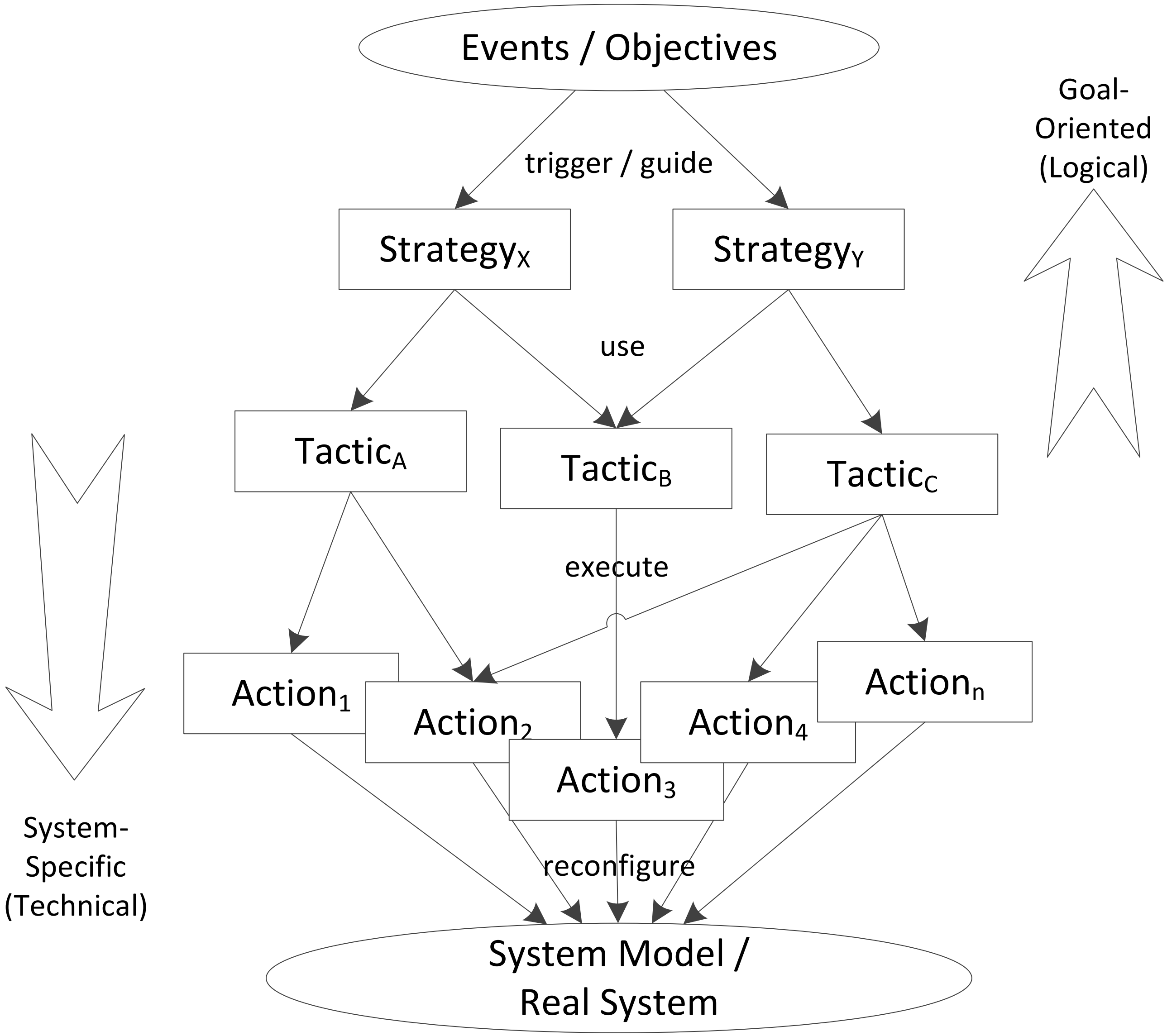
(Coarse-Grained Service Behavior Model)



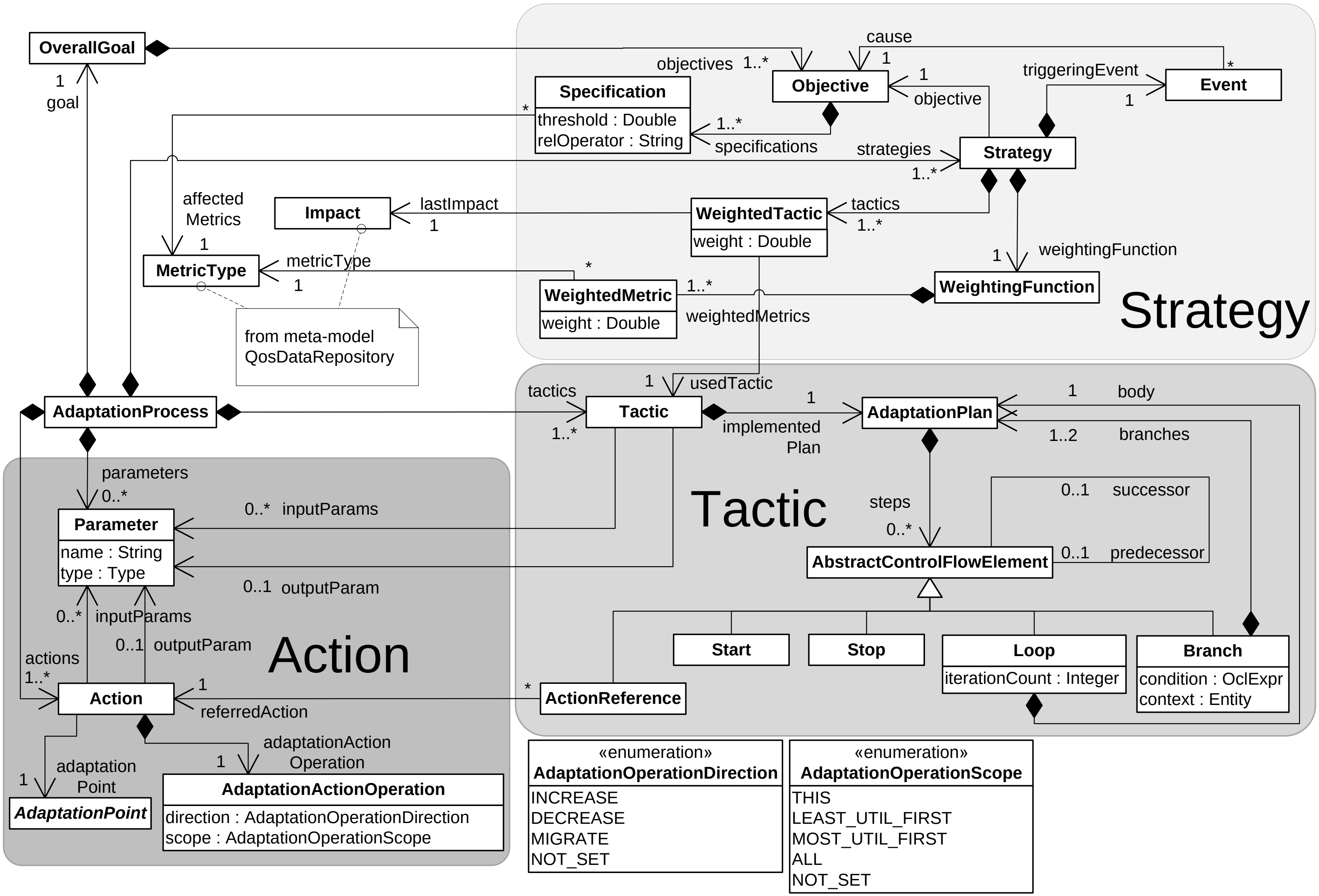
Example
(Fine-Grained Service Behavior Model)



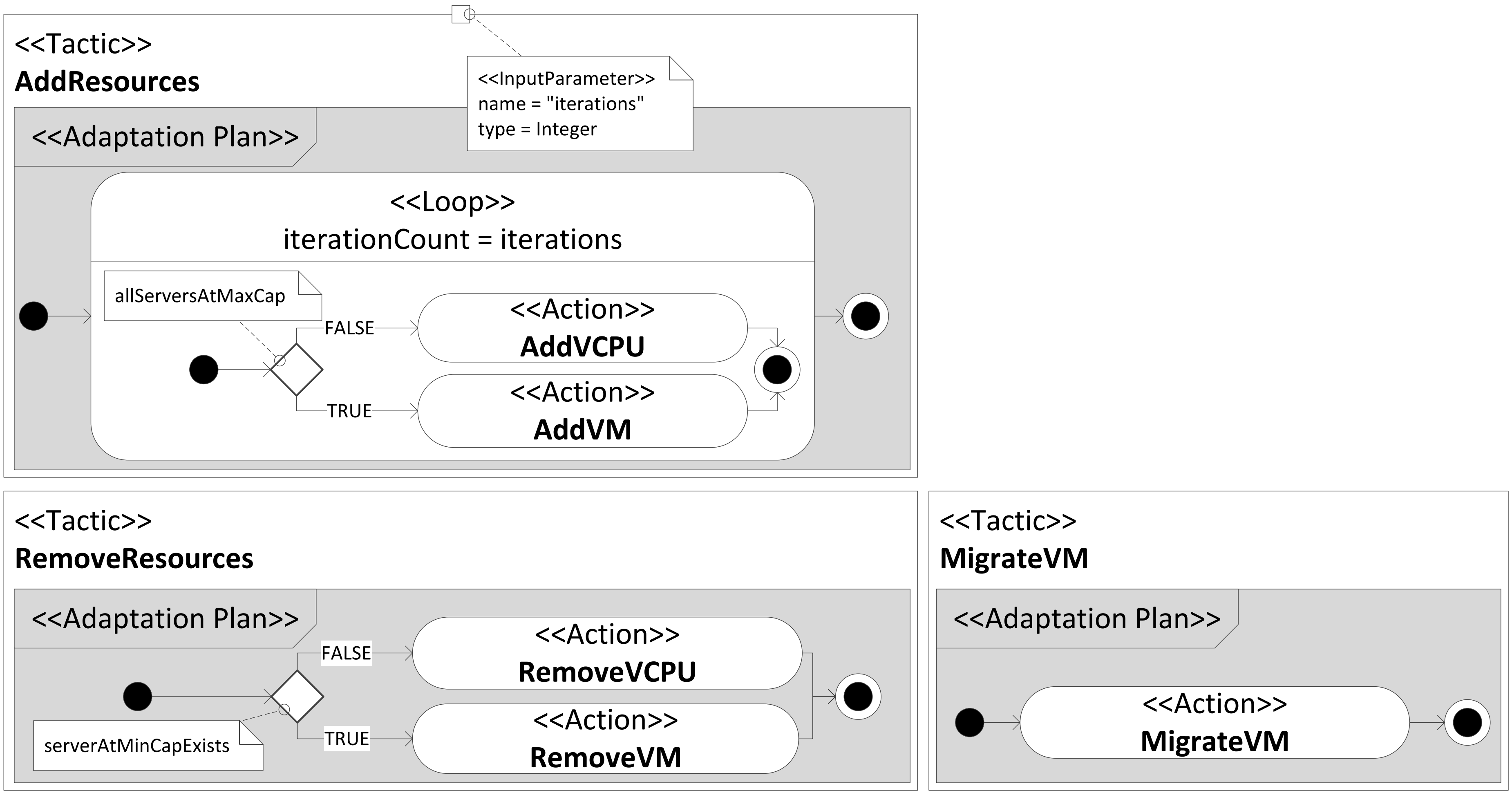
Adaptation Process



S/T/A Meta-Model (Strategies, Tactics and Actions)

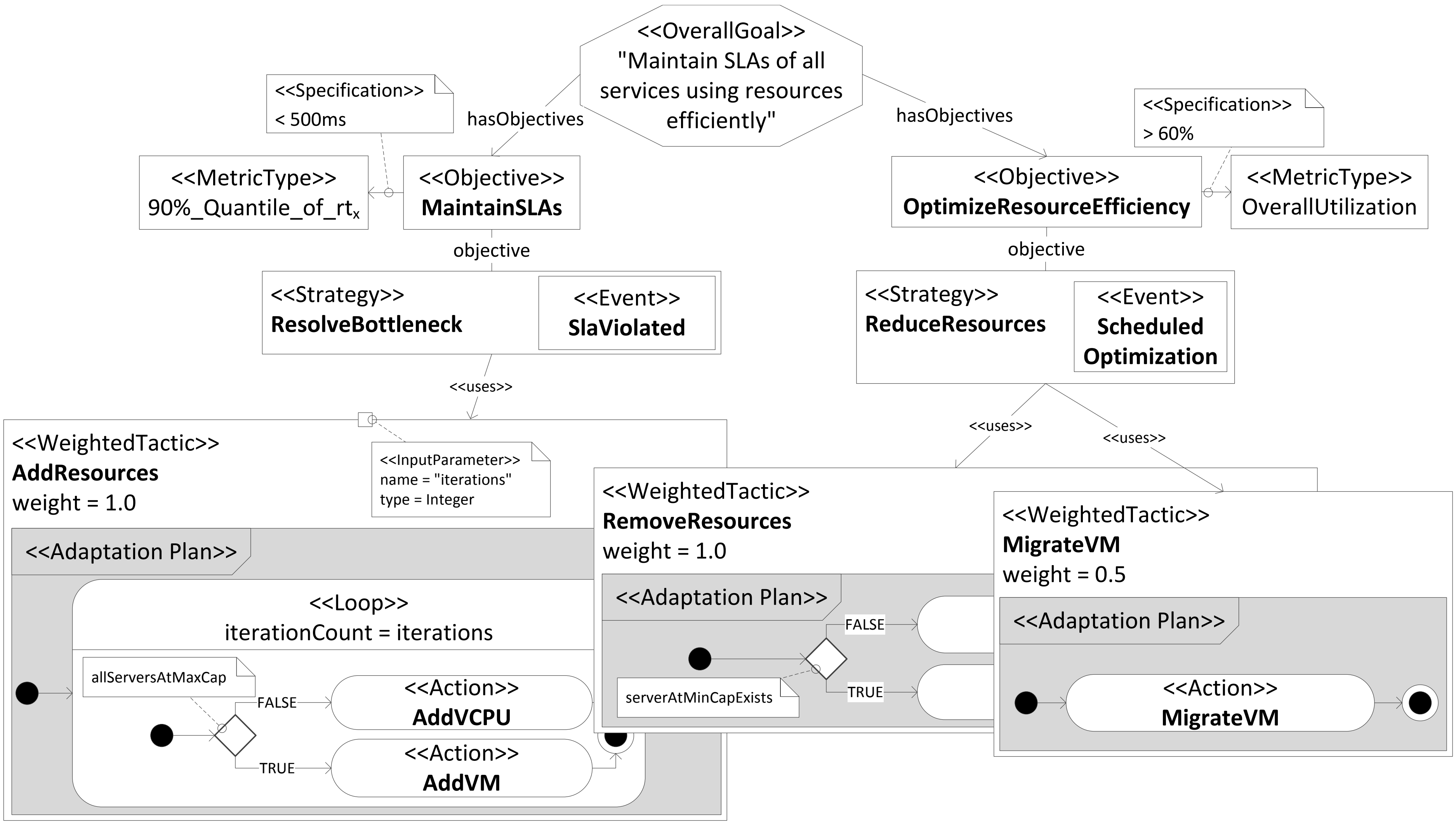


Example
(Tactics)

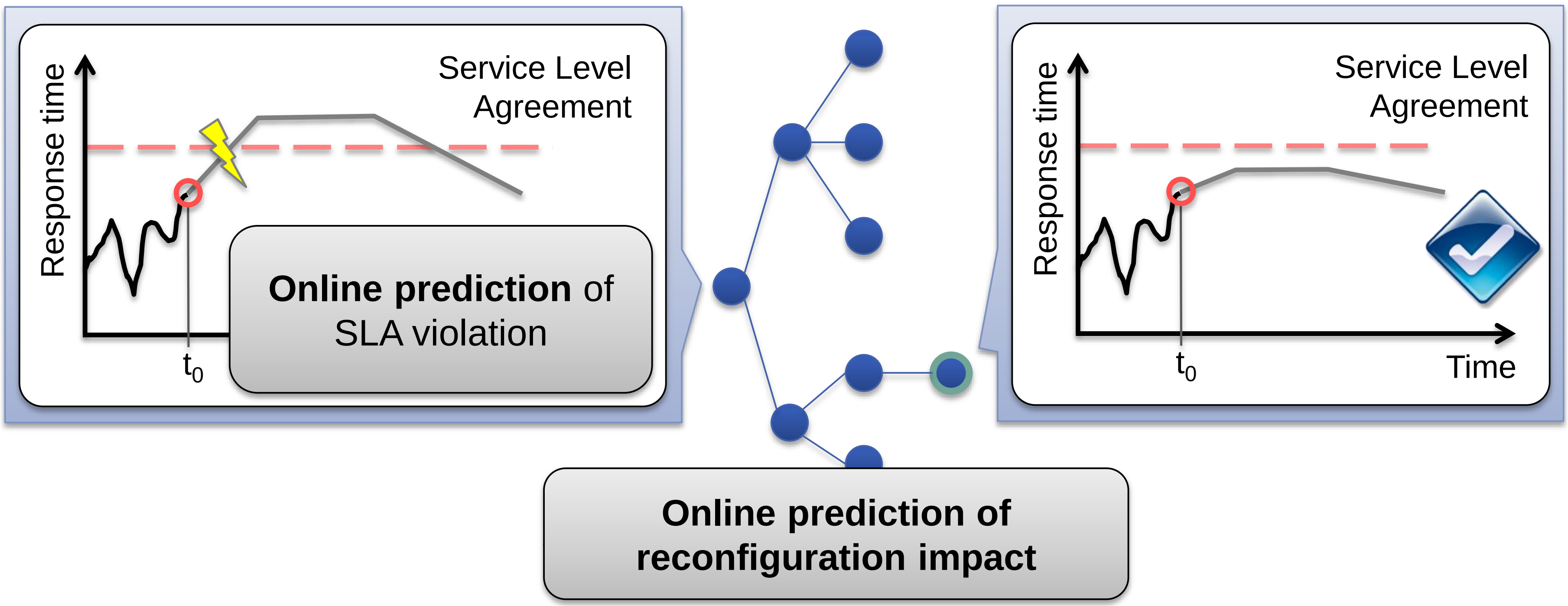


Example

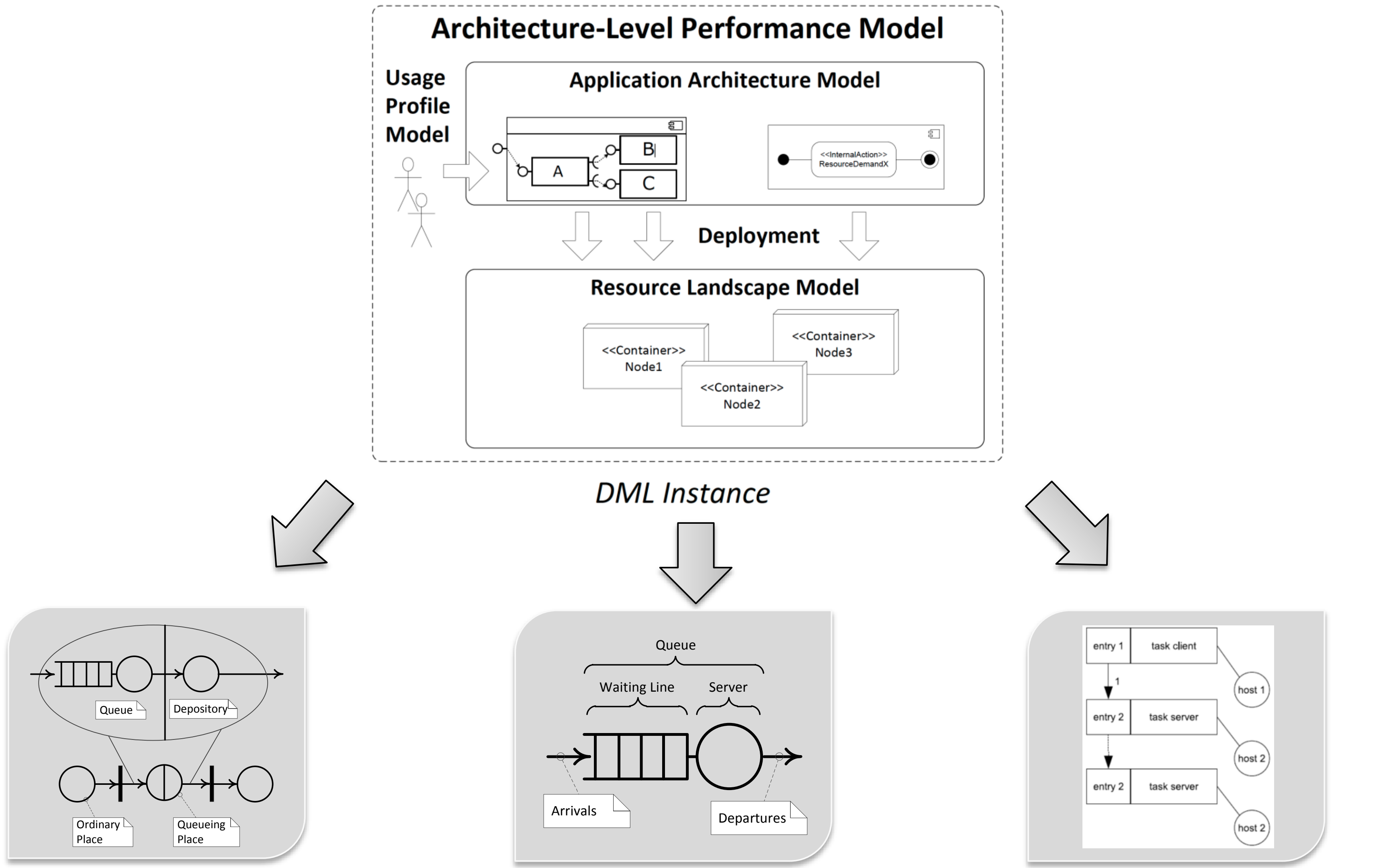
(S/T/A Model Instance)



Self-Predictive Property



Transformations to Predictive Models

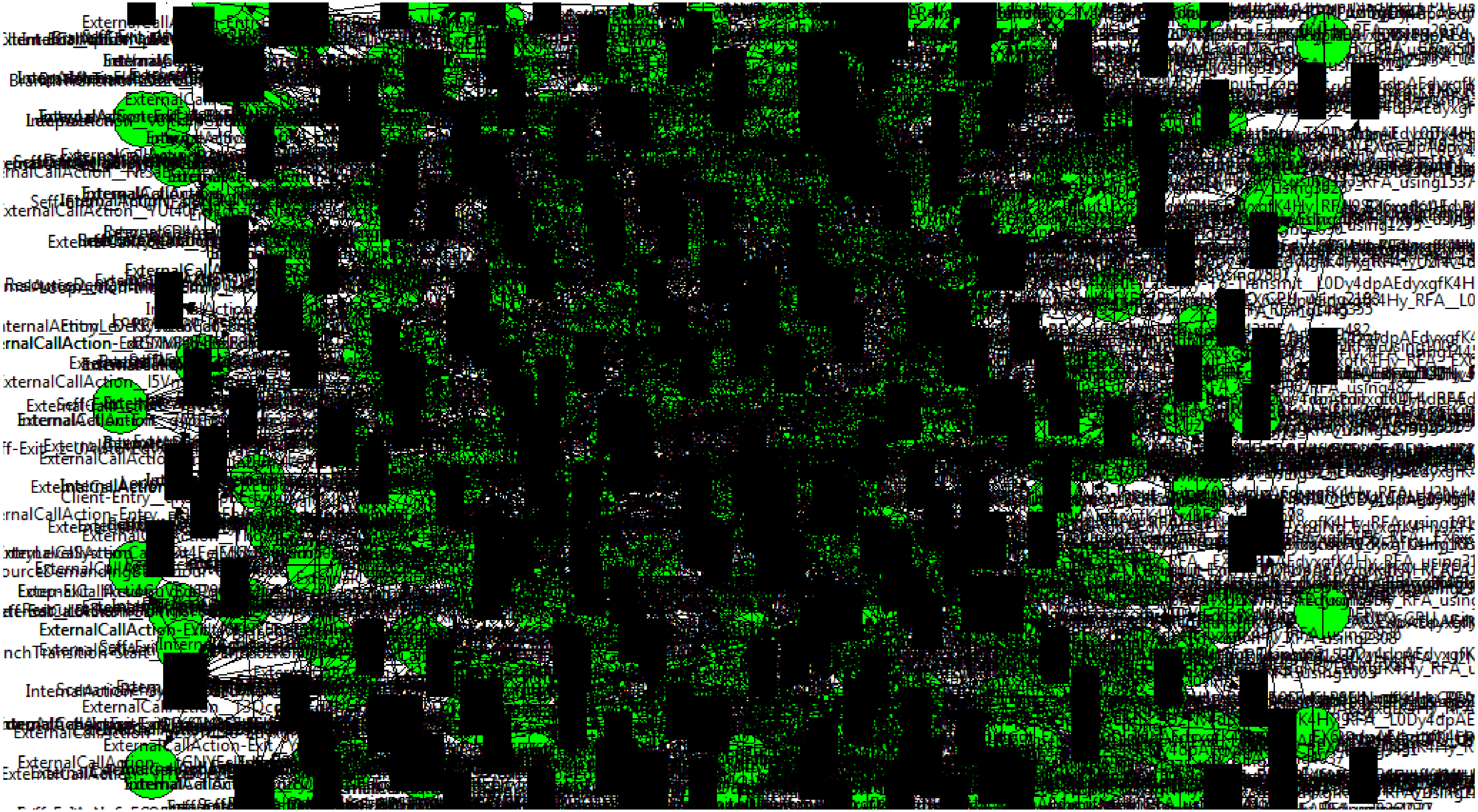


Queueing Petri Net

Bounds Analysis Model

Layered Queueing Network

Example: Automatically Generated QPN Model



P. Meier, S. Kounev, and H. Koziolek. **Automated transformation of component-based software architecture models to queueing petri nets.** In *19th IEEE/ACM Intl. Symp. on Modeling, Analysis and Simulation of Computer and Telecomm. Systems (MASCOTS), Singapore, July 25-27*, 2011. [bib | .pdf]

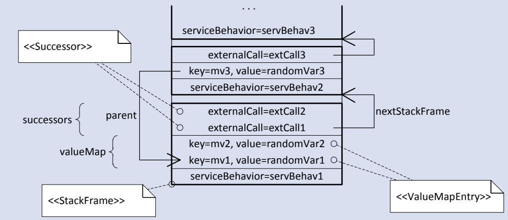
Tailored Model Solution

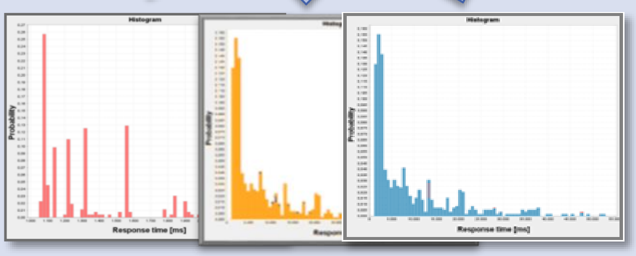
Analytical Analysis

$$R \geq \max\left[N \times \max\{D_i\}, \sum_{i=1}^{K} D_i \right] \qquad X_0 \leq \min\left[\frac{1}{\max\{D_i\}}, \frac{N}{\sum_{i=1}^{K} D_i} \right]$$

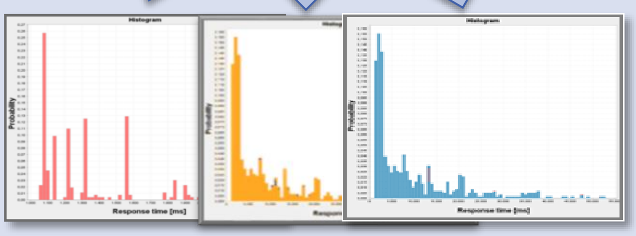
$$\frac{N}{\max\{D_i\}[K+N-1]} \leq X_0 \leq \frac{N}{avg\{D_i\}[K+N-1]}$$

Simulative Analysis





Analysis Results



Analysis Results

F. Brosig, P. Meier, S. Becker, A. Koziolek, H. Koziolek, and S. Kounev. **Quantitative Evaluation of Model-Driven Performance Analysis and Simulation of Component-based Architectures**. *IEEE Transactions on Software Engineering (TSE)*, 2014, IEEE, Preprint. [DOI | .pdf]

Overview of Applied Modeling Techniques

Descriptive Architecture-level Models

- OMG Meta Object Facility (MOF)
  - MOF-based meta-models
- (UML MARTE)
- (UML SPT)

Predictive Performance Models

- Bounding techniques
- Operational analysis
- Statistical regression models
- Stochastic process algebras
- (Extended) queueing networks
- Layered queueing networks
- Queueing Petri nets
- Reinforcement learning models
- Detailed simulation models

Workload Forecasting

- AR(I)MA
- Extended exp. smoothing
- tBATS
- Croston's method
- Cubic smoothing splines
- Neural network-based

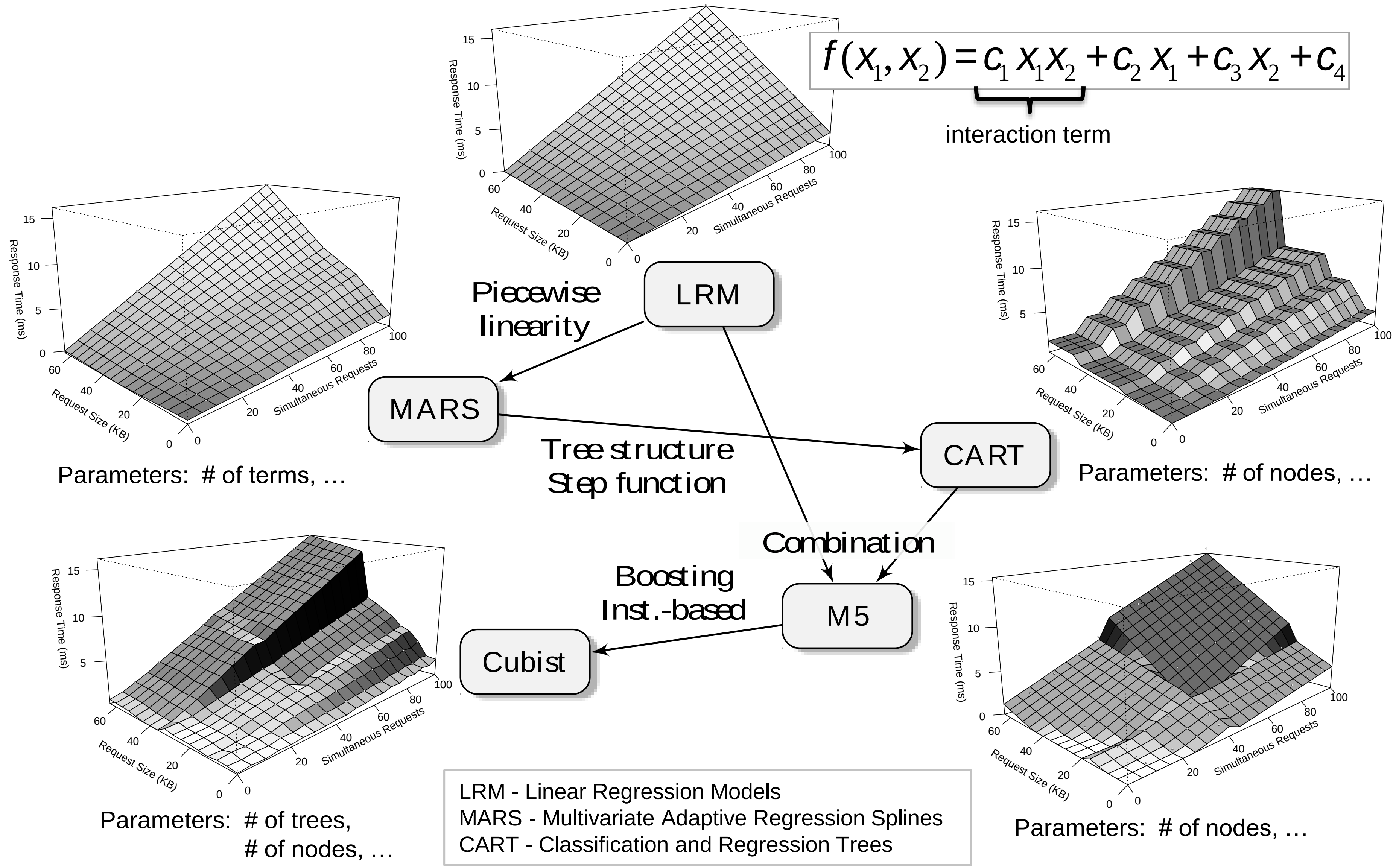
Resource Demand Estimation

- Regression-based techniques
- Kalman filter
- Nonlinear optimization
- Maximum likelihood estimation
- Independent component analysis

Regression Analysis

- MARS
- CART
- M5 trees
- Cubist forests
- Quantile regression forests
- Support vector machines

Example Statistical Regression Models



Challenges

- **Interoperability** of modeling languages
- **Automatic model extraction**, maintenance, refinement, and calibration during operation
- Supporting **flexible analysis** (accuracy vs. overhead)
- Scalable and **efficient algorithms** for system adaptation
- **Separation of responsibilities** in virtualized infrastructures
- Lack of **representative benchmarks** for evaluating self-awareness

Lack of Benchmarks

Reliable Metrics

- What exactly should be measured and computed?

Representative Workloads

- For which scenarios and under which conditions?

Sound Measurement Methodology

- How should measurements be conducted?

“*To **measure** is to **know**.*” -- Clerk Maxwell, 1831-1879

“*It is much easier to make **measurements** than to **know** exactly what you are measuring.*“ -- J.W.N.Sullivan (1928)

Standard-Performance-Evaluation-Corporation

SPEC Research Group

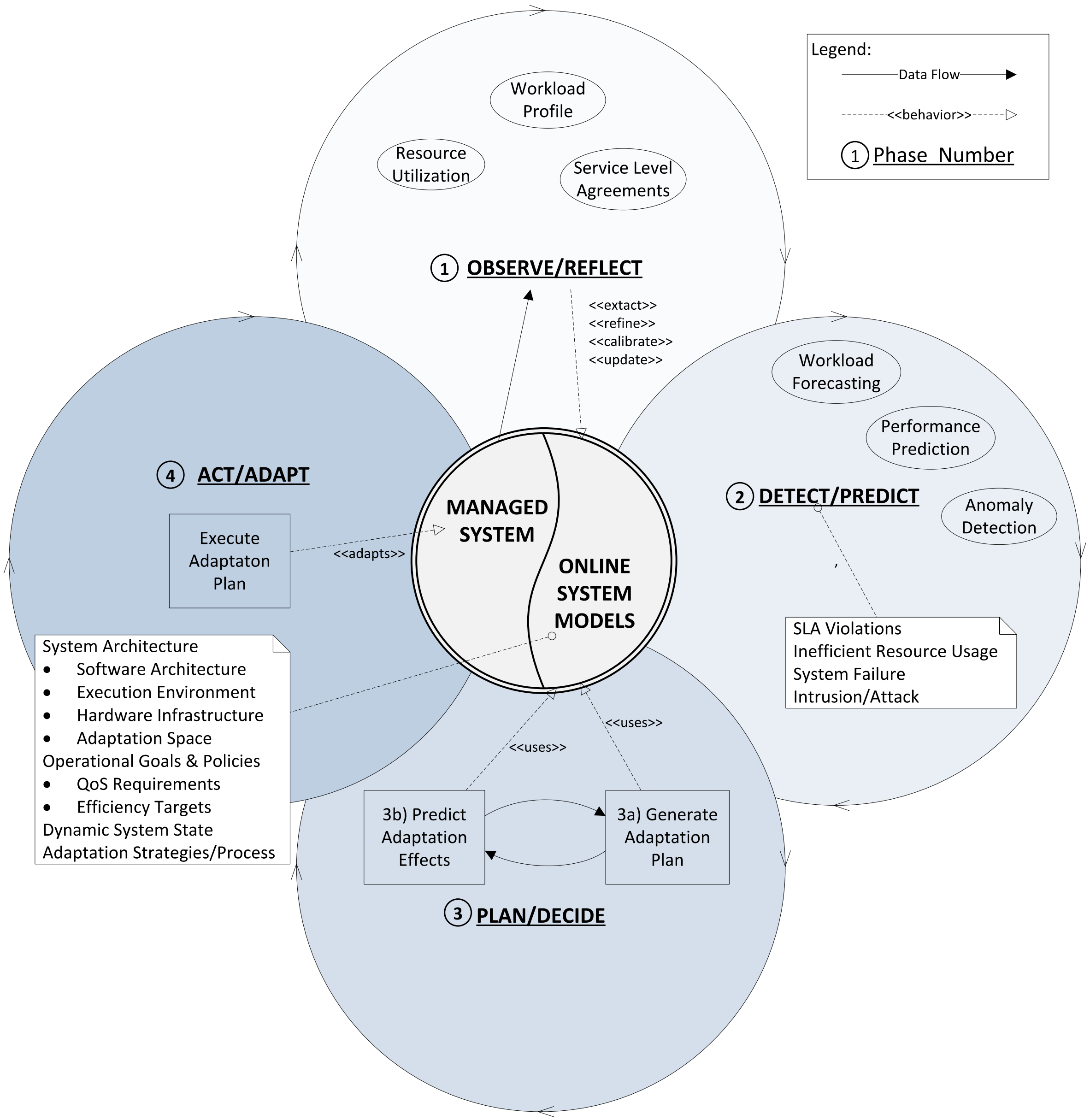
- **Open-Systems-Group (OSG)**
  - Processor and computer architectures
  - Virtualization platforms
  - Java (JVM, Java EE)
  - Message-based systems
  - Storage systems (SFS)
  - Web-, email- and file server
  - SIP server (VoIP)
  - Cloud computing
- **High-Performance-Group (HPG)**
  - Symmetric multiprocessor systems
  - Workstation clusters
  - Parallel and distributed systems
  - Vector (parallel) supercomputers
- **"Graphics and Workstation Performance Group" (GWPG)**
  - CAD/CAM, visualization
  - OpenGL

http://www.spec.org

SPEC Research Group (RG)

- Founded in March 2011: http://research.spec.org
  - Transfer of knowledge btw. academia and industry
- Activities
  - Methods and techniques for experimental system analysis
  - Standard metrics and measurement methodologies
  - Benchmarking and certification
  - Evaluation of academic research results
- Member organizations (Feb 2014)

Self-Aware Computing Vision



Questions?

Thank You!

skounev@acm.org
http://se.informatik.uni-wuerzburg.de