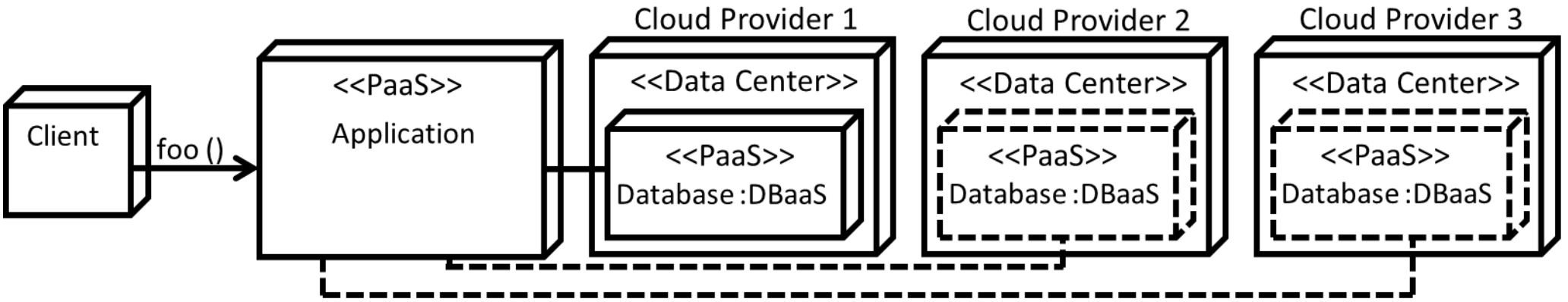
The iObserve Approach
--
Runtime Architecture Modeling and Visualization

Schedule of Events

09:00 – 09:10	Welcome and General Introduction
09:10 – 09:40	Study Foundations
09:40 – 10:00	Model-based Software Application Monitoring
10:00 – 10:30	**Runtime Architecture Modeling and Visualization**
10:30 – 11:00	Coffee Break
11:00 – 12:15	Introduction to the ExplorViz, Palladio, and iObserve Approaches with following Tool / Visualization Demos
12:15 – 12:30	Study Setup
12:30 – 14:00	Lunch
14:00 – 15:30	Comprehensibility Study
15:30 – 16:00	Coffee Break
16:00 – 16:30	Live Database Trace Visualization in Large Software Landscapes
16:30 – 17:00	Feedback and Open Discussion

A software application build by composing services



- \+ Flexibility, scalability, reusability
- \+ Economic use of resources
- − Complexity, fragility
- − Changes during operations unknown in development phase

DevOps practices

DevOps

DevOps is a set of practices of operators and developers participating together in the entire application lifecycle, from design through the development process to production support

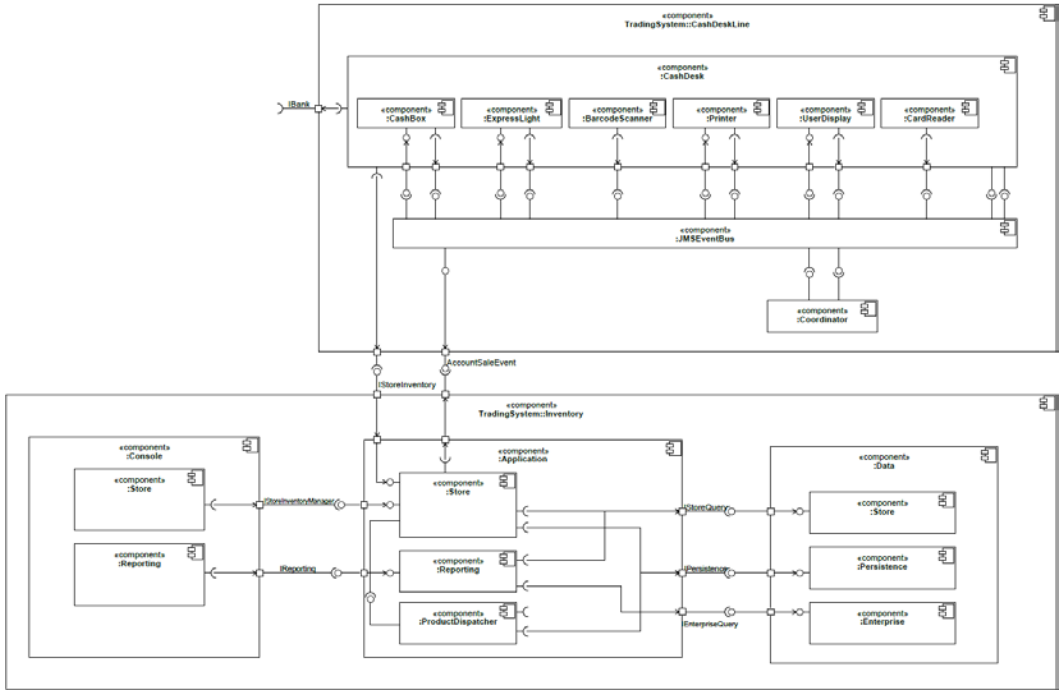
E. Mueller. What is DevOps?
http://theagileadmin.com/what-is-devops, 2016.

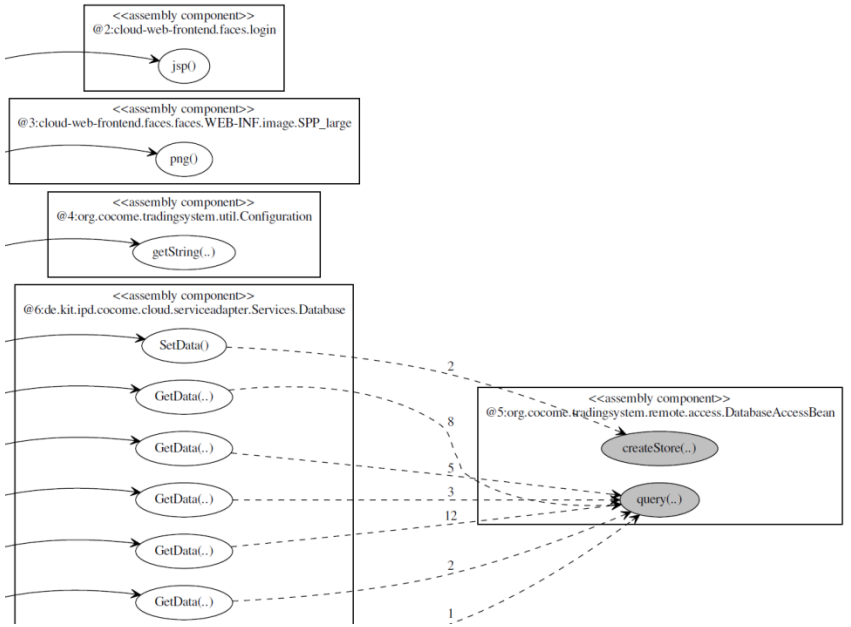
- Developers and operators must work more closely
- Feedback cycles from Dev to Ops and Ops to Dev
- Increased communication among developers and operators
- Integration of the role of developer and operator

➢ Differences in architectural models in development and operations

Differences in Architectural Models in Development and Operations

1 Level of abstraction (component-based vs. close to implementation level)



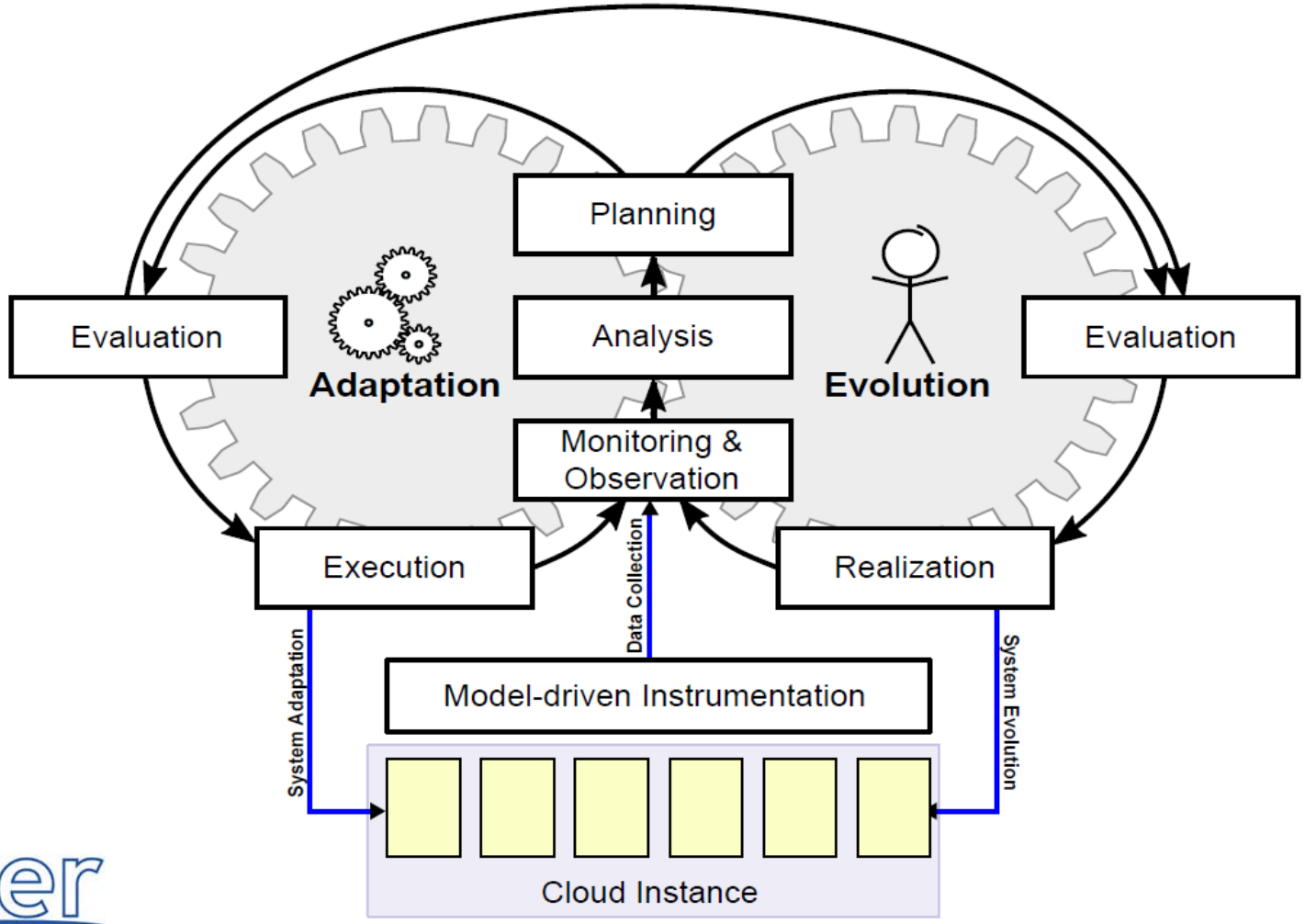


Differences in Architectural Models in Development and Operations

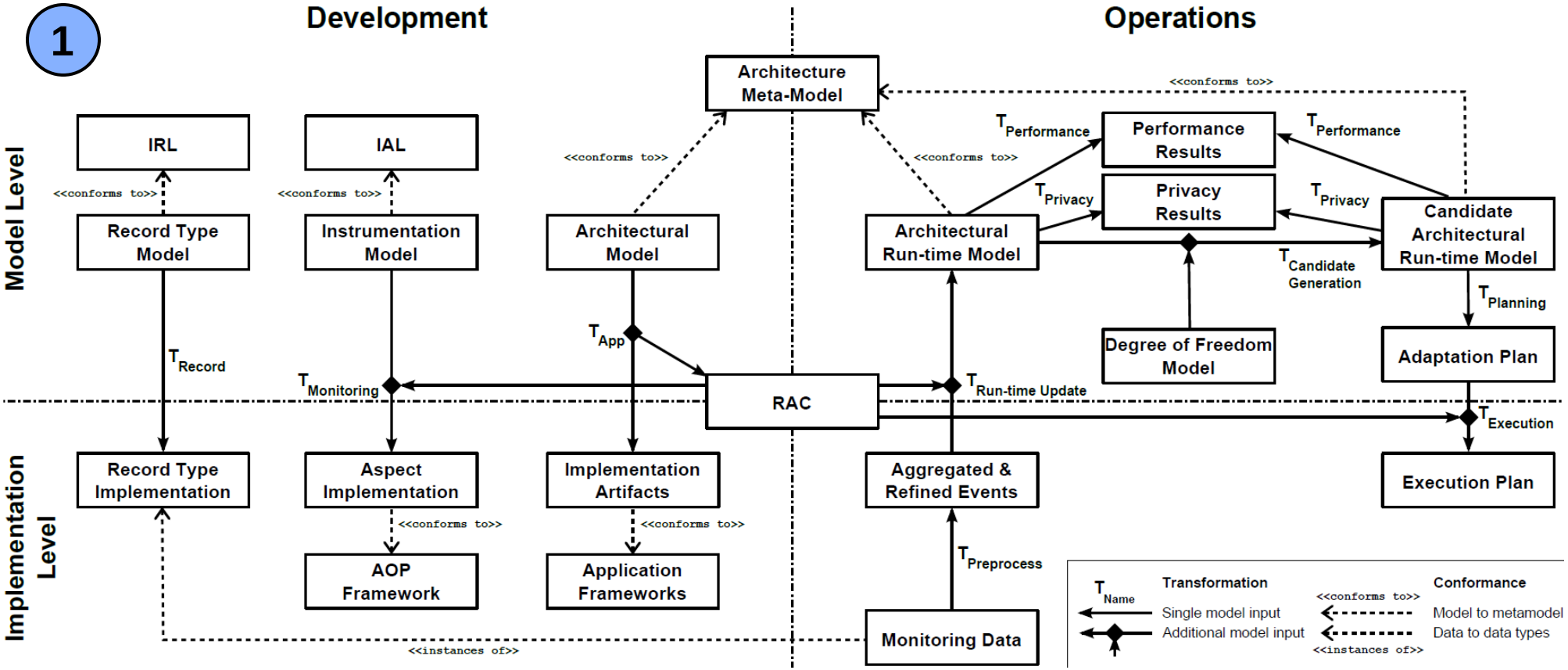
1. Level of abstraction (component-based vs. close to implementation level)
2. Purpose (finding appropriate design vs. reflecting current application configuration)
3. Content (static vs. dynamic)
   - Structure and design
   - In-memory objects and communication
   - Utilization of server

- Limited reuse of development models during operations
- Limited phase-spanning consideration of the software architecture

Overview of the iObserve Approach



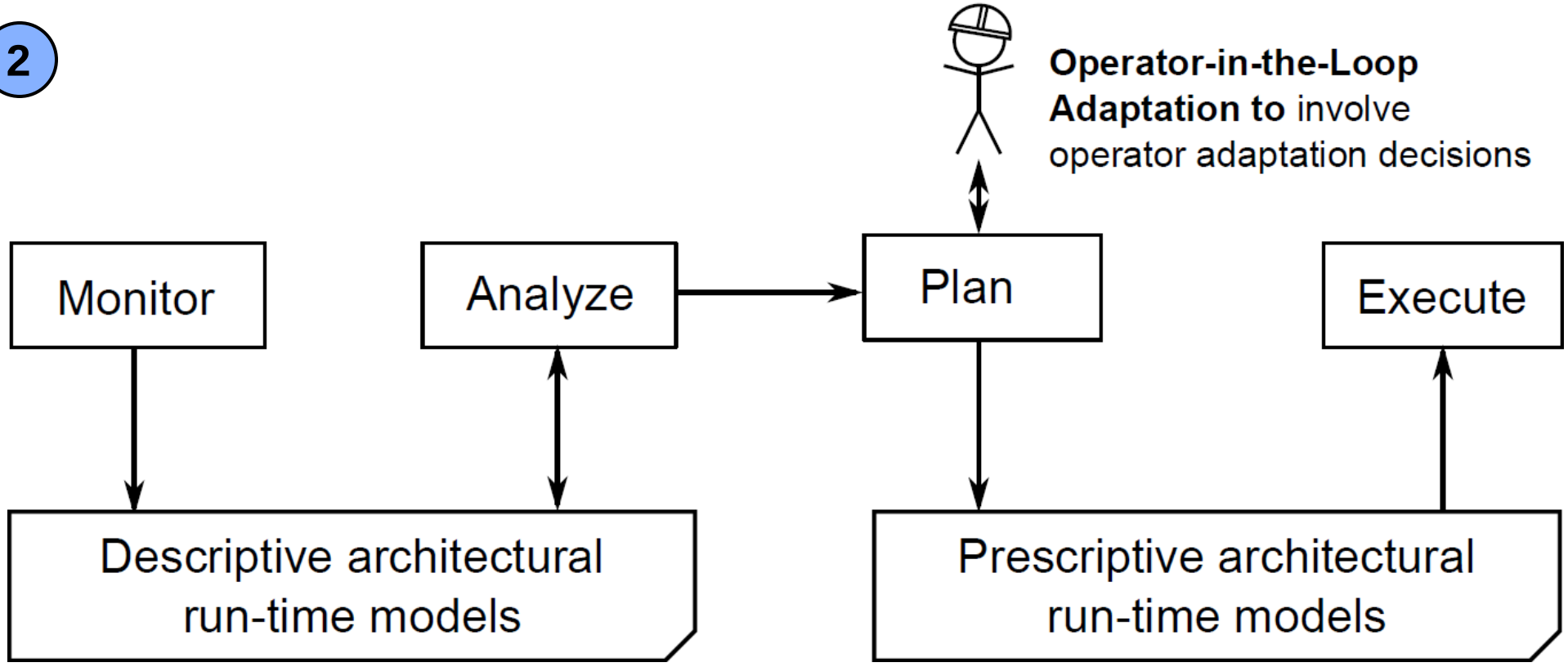
The iObserve Megamodel



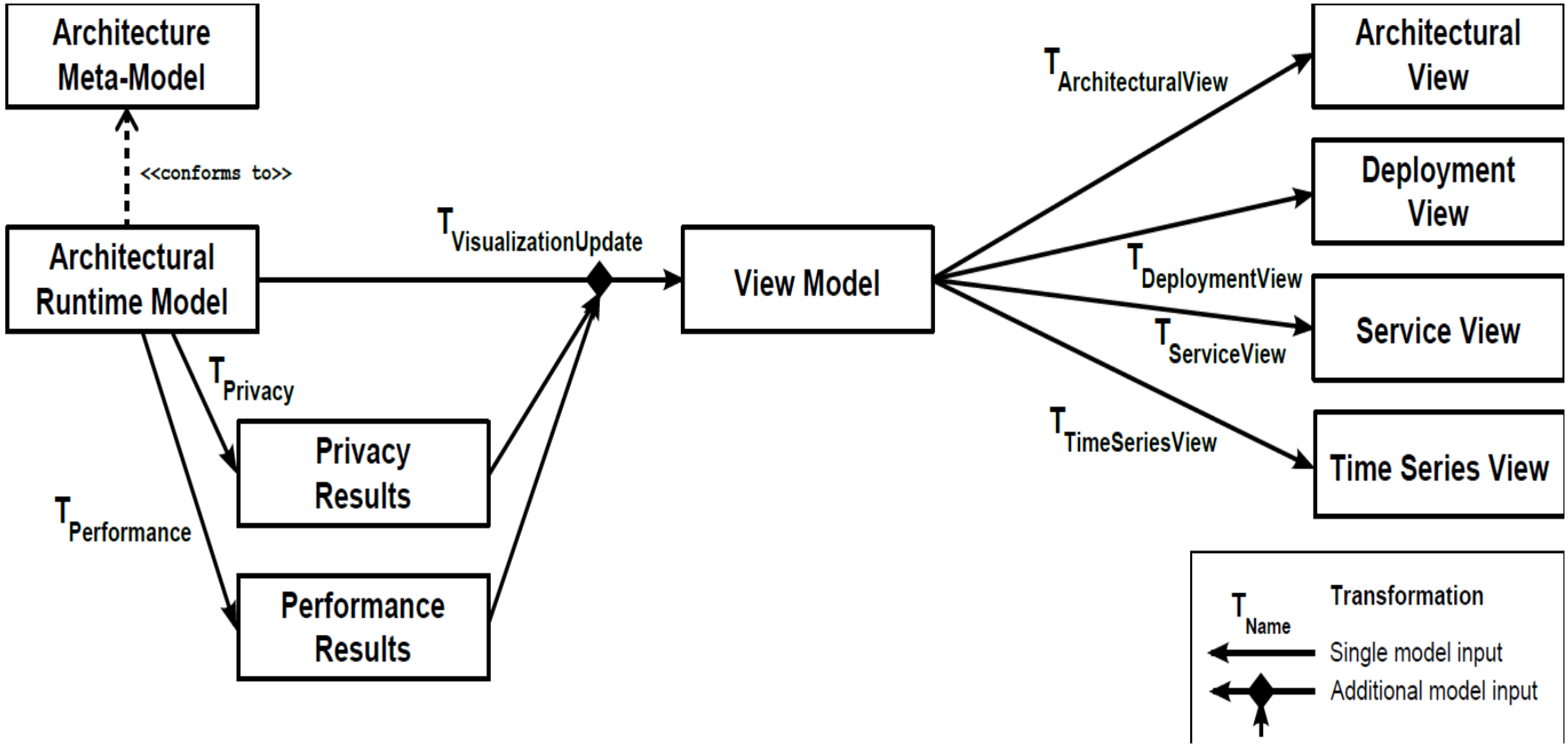
- Bridges the divergent abstraction levels in development and operations

R. Heinrich et al. An Architectural Model-Based Approach to Quality-aware DevOps in Cloud Applications. In Software Architecture for Big Data and the Cloud, Elsevier, 2017

2

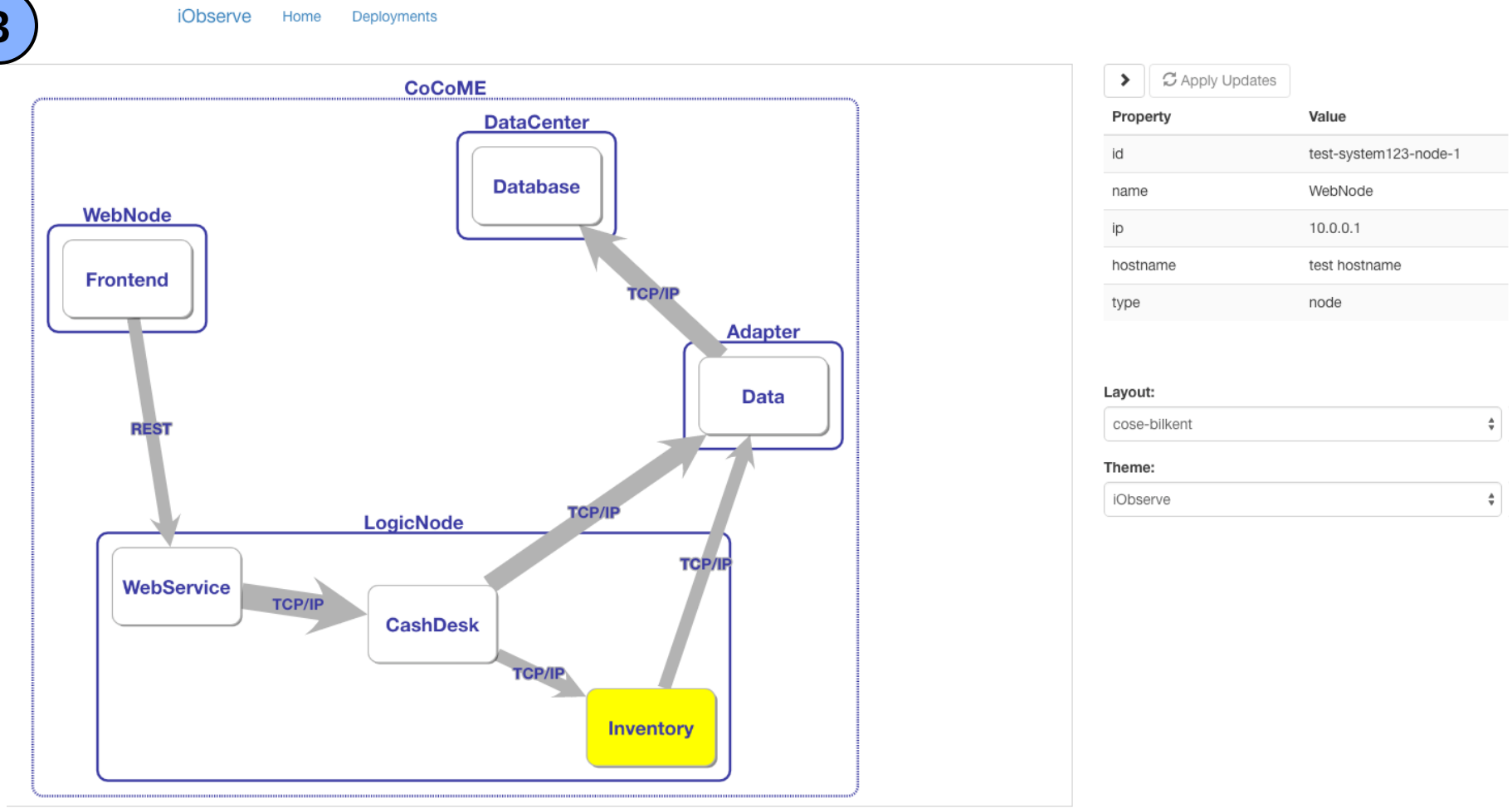


R. Heinrich et al. Architectural Run-Time Models for Operator-in-the-Loop Adaptation of Cloud Applications, 9th IEEE Symposium on the Maintenance and Evolution of Service-Oriented Systems and Cloud-Based Environments, 2015. IEEE.


Architecture Meta-Model
<<conforms to>>
Architectural Runtime Model
T VisualizationUpdate
View Model
T ArchitecturalView
Architectural View
Deployment View
T DeploymentView
Service View
T ServiceView
T TimeSeriesView
Time Series View
T Privacy
Privacy Results
T Performance
Performance Results
T Name
Transformation
Single model input
Additional model input

iObserve Live Visualization

3



Conclusion and Future Work

Differences in architectural models in development and operations

- Abstraction (component-based vs. close to implementation level)
- Purpose (finding appropriate design vs. reflecting executed application)
- Content (static vs. dynamic)

➢ iObserve is a first attempt to bridge differences in architectural models

Future Work

- Further investigate the planning and execution phases
- Improve live visualization of data contained in the iObserve megamodel
- Conduct experiments for evaluating scalability and usefulness

- Develop reference architecture for various quality aspects in monitoring, analysis and planning
- Support meta-model modularity, extension and evolution

https://github.com/research-iobserve

References

- L. Bass et al. DevOps: A Software Architect's Perspective. Pearson, 2015.
- F. Fittkau et al. Live trace visualization for comprehending large software landscapes: The ExplorViz approach. In VISSOFT. IEEE, 2013.
- R. Heinrich et al. A platform for empirical research on information system evolution. In SEKE, pages 415-420. KSI, 2015.
- R. Heinrich. Architectural run-time models for performance and privacy analysis in dynamic cloud applications. SIGMETRICS Performance Evaluation Review, 43(4):13–22, 2016.
- R. Heinrich et al. An Architectural Model-Based Approach to Quality-aware DevOps in Cloud Applications. In Software Architecture for Big Data and the Cloud, Elsevier, 2017
- R. Heinrich et al. The CoCoME platform for collaborative empirical research on information system evolution. Technical Report 2016,2; Karlsruhe Reports in Informatics, KIT, 2016
- K. Rostami et al. Architecture-based assessment and planning of change requests. In QoSA, pages 21-30. ACM, 2015.