Model-driven Monitoring

ICSA 2017, Kiel

Reiner Jung

3rd April 2017

iObserve

Kieker

Motivation

Why?

- Observing system operation
- Detect anomalies, like bottlenecks
- Detect altering user behavior/acceptance of new UIs
- Right size servers

What?

Hasselbring et al. 2013

- Resource Utilization
- Allocations
- Deployments
- User interaction
- Call and message traces
- Networks
- Database interactions

Instrumentation of software systems

Multiple languages Java, Javascript, C, Perl, Python, Fortran

Multiple instrumentation technologies

- AspectJ, EJB Interceptors, Servlet Listeners
- AspectC
- Compiler based weaving, e.g., `gcc -finstrument-functions -g`
- Perl Sub::WrapPackages

Multiple data sampler custom, SNMP

Multiple platforms phones, servers, IoT

Hoorn et al. 2012

Instrumentation

Draw backs

- Different data formats
- Different ways to instrument
- Specialized analysis tools

Solution Modeling instrumentation

- Unified data model
- Unified specification of pointcuts and advice
- Reuse of analysis tooling

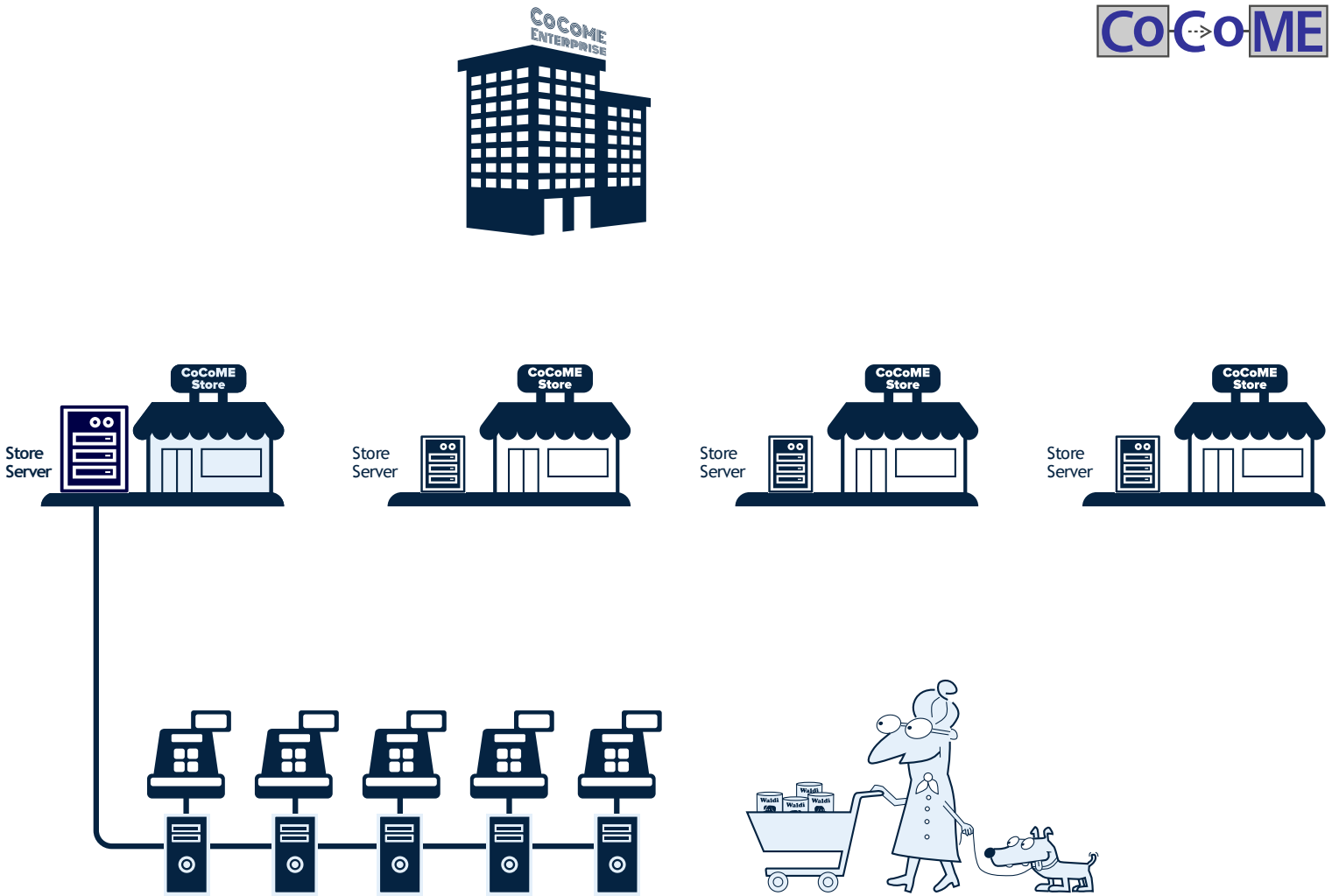
Common Component Modeling Example

CoCoME

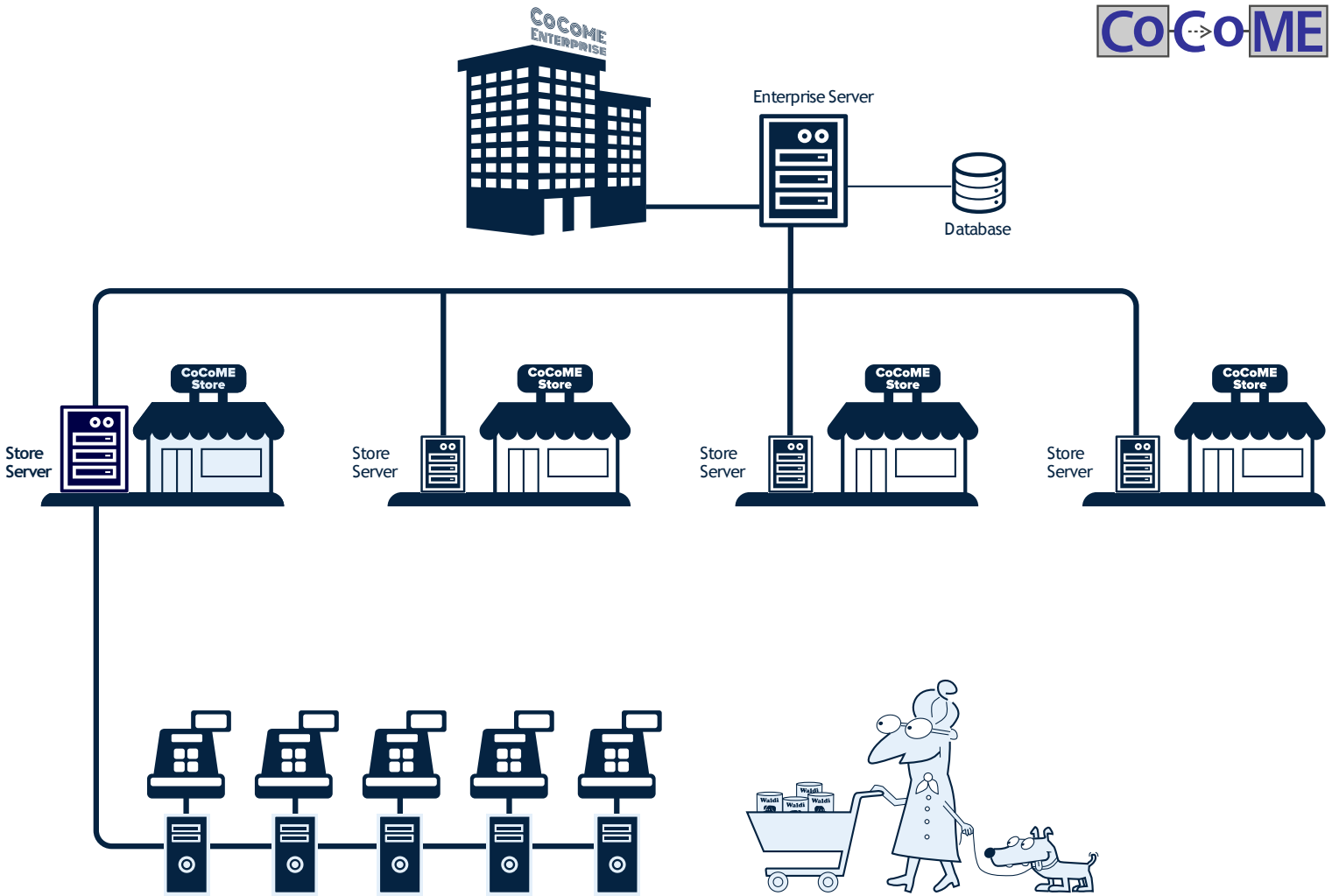
Heinrich et al. 2016

Common Component Modeling Example

Common Component Modeling Example

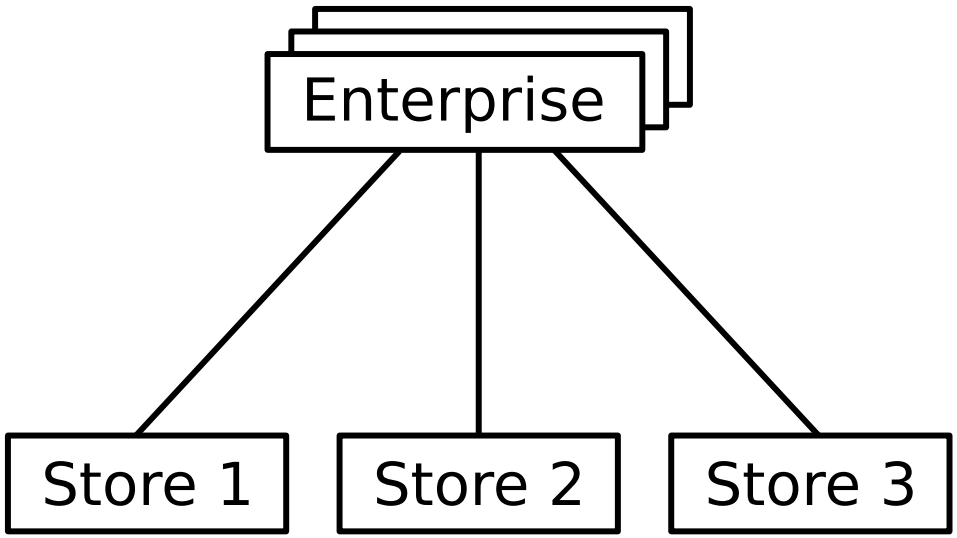


Common Component Modeling Example

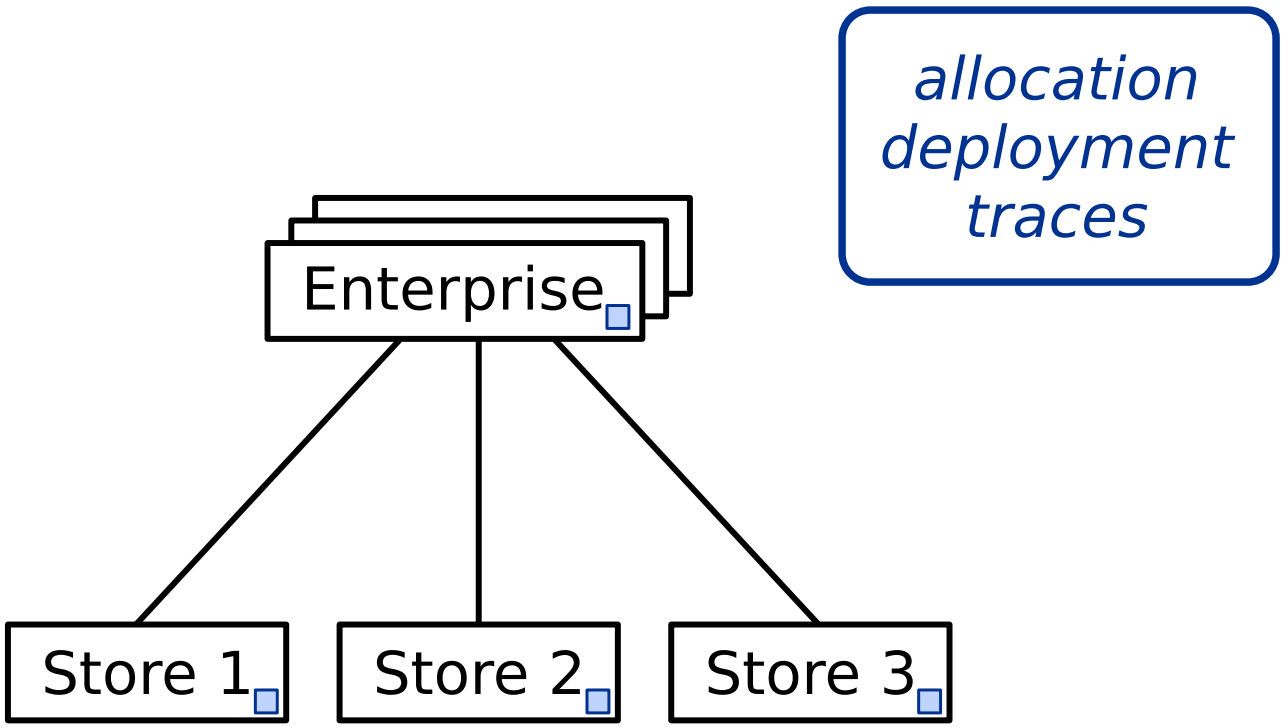


Heinrich et al. 2016

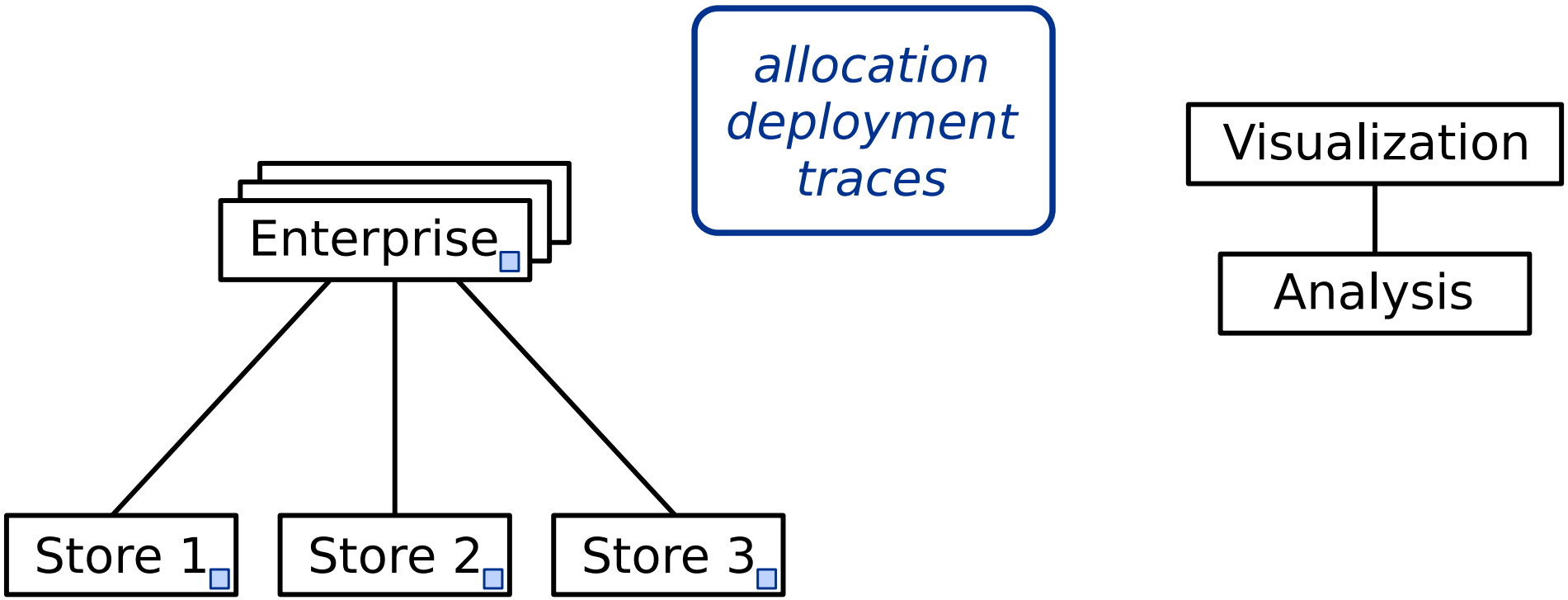
CoCoME Monitoring Example



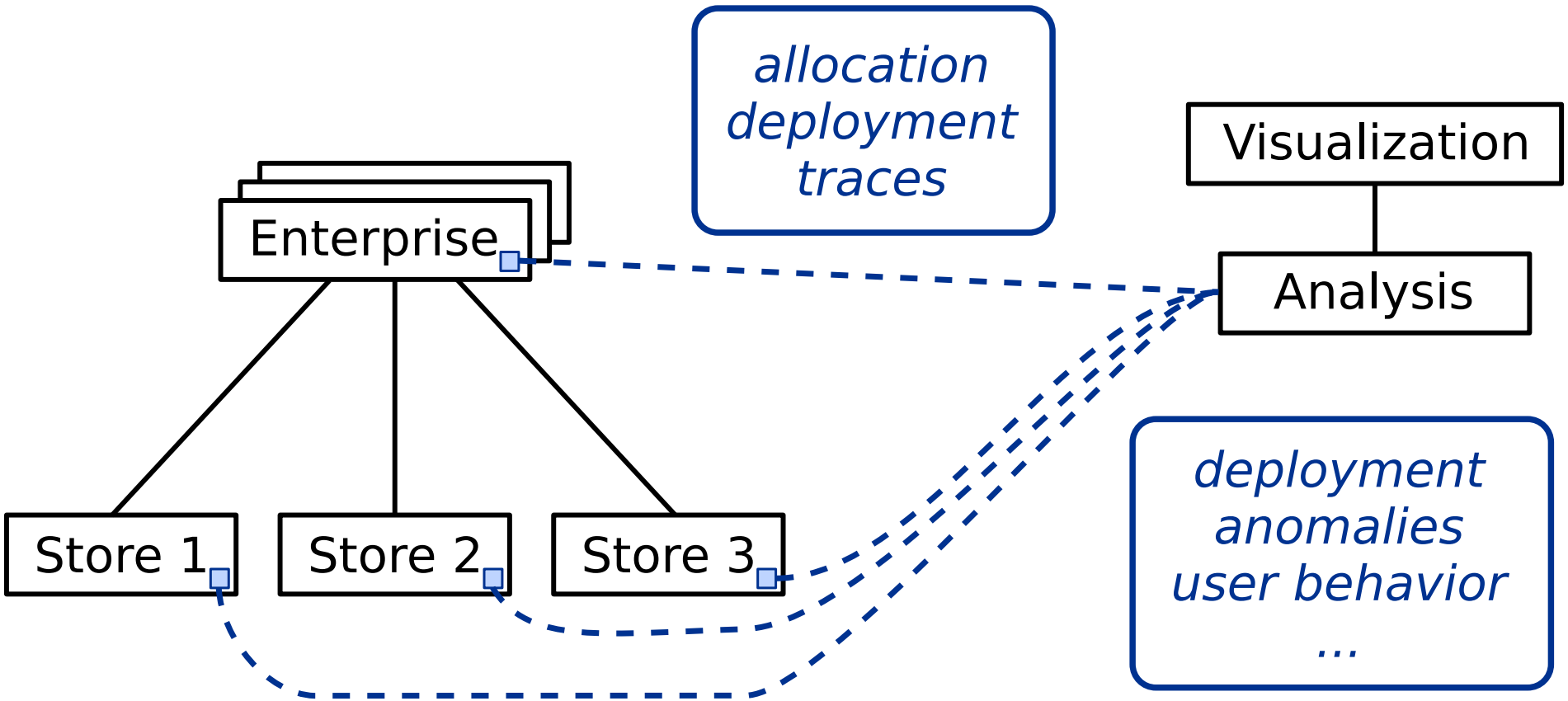
CoCoME Monitoring Example



CoCoME Monitoring Example

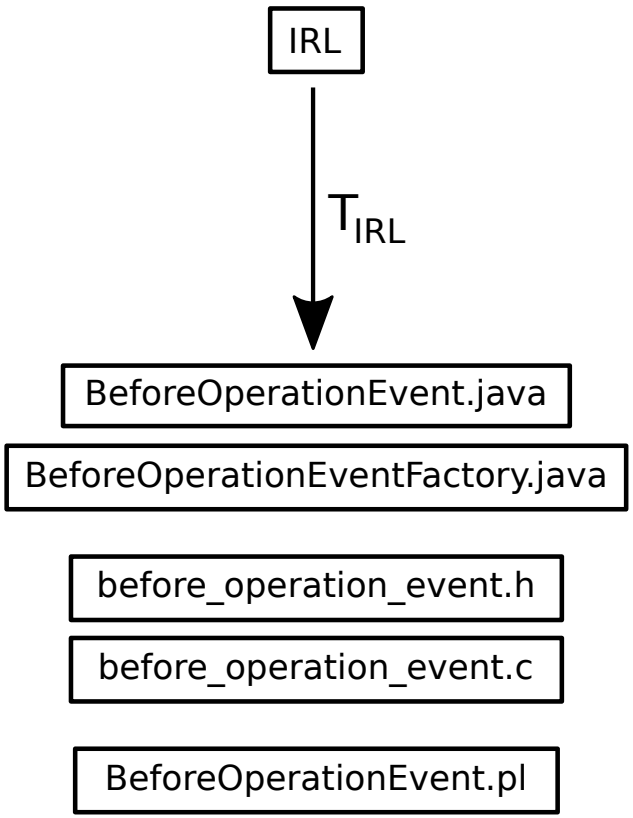


CoCoME Monitoring Example



Data Modeling

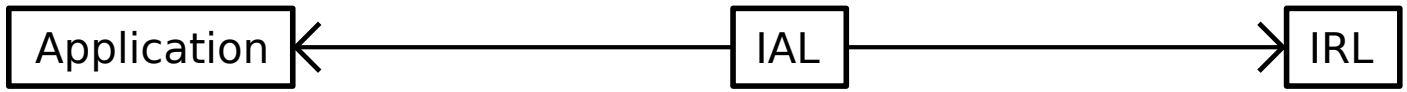
Instrumentation Record Language




```
package kieker.common.records

template ITraceRecord {
  int traceId : trace—id
  int orderIndex : order—index
}

abstract event AbstractEvent {
  long timestamp : time
}

abstract event AbstractOperationEvent
  extends AbstractEvent {
  string classSignature : class—signature
  string operationSignature : operation—signature
}

event BeforeOperationEvent
  extends AbstractOperationEvent : ITraceRecord
```

Aspect and Pointcut Modeling

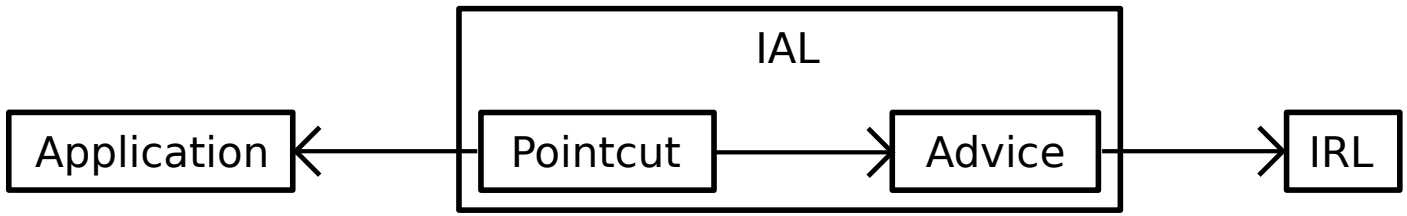
Model



Jung and Wulf 2016; Jung, Heinrich, et al. 2016; Jung 2016

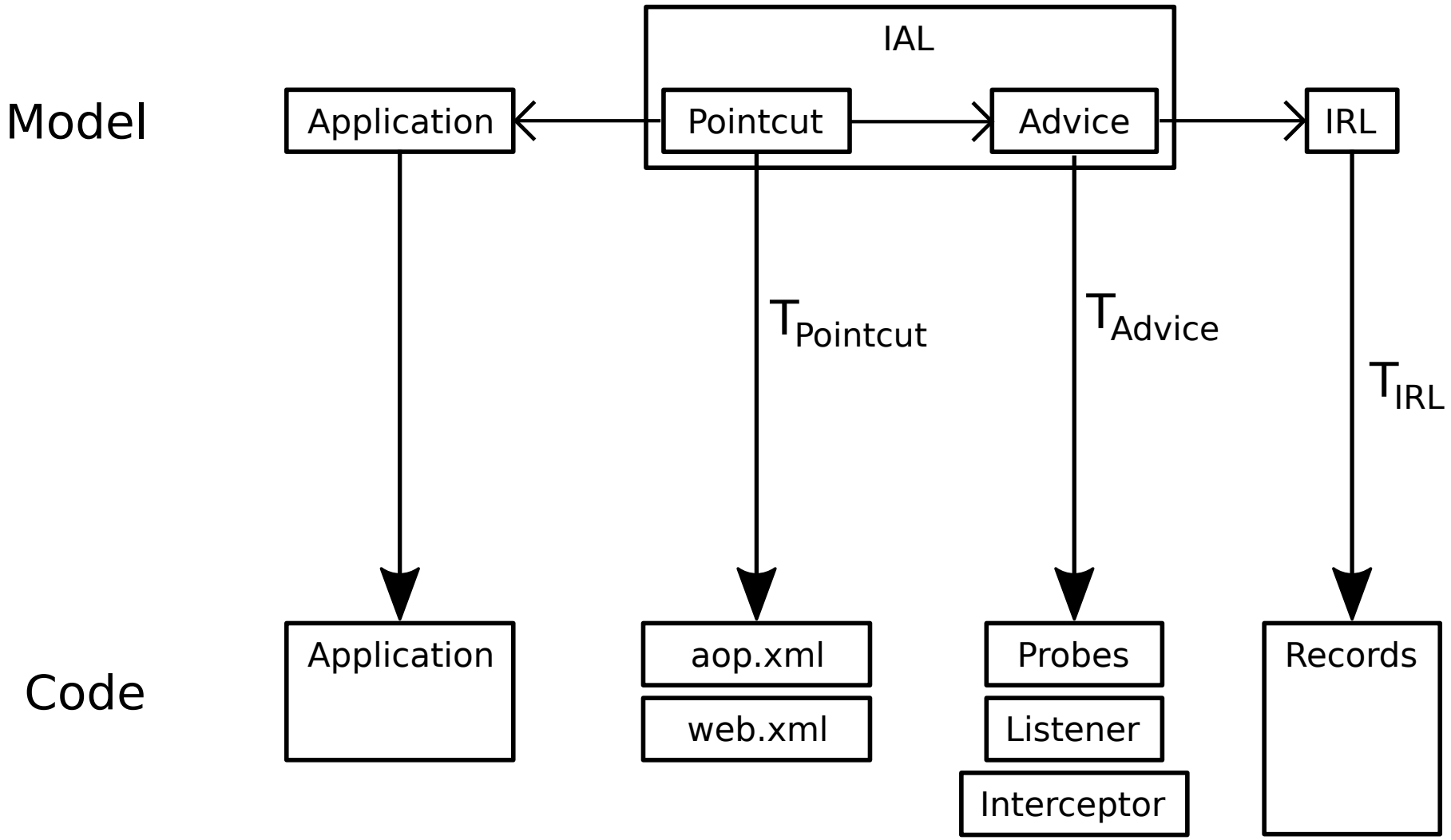
Aspect and Pointcut Modeling

Model



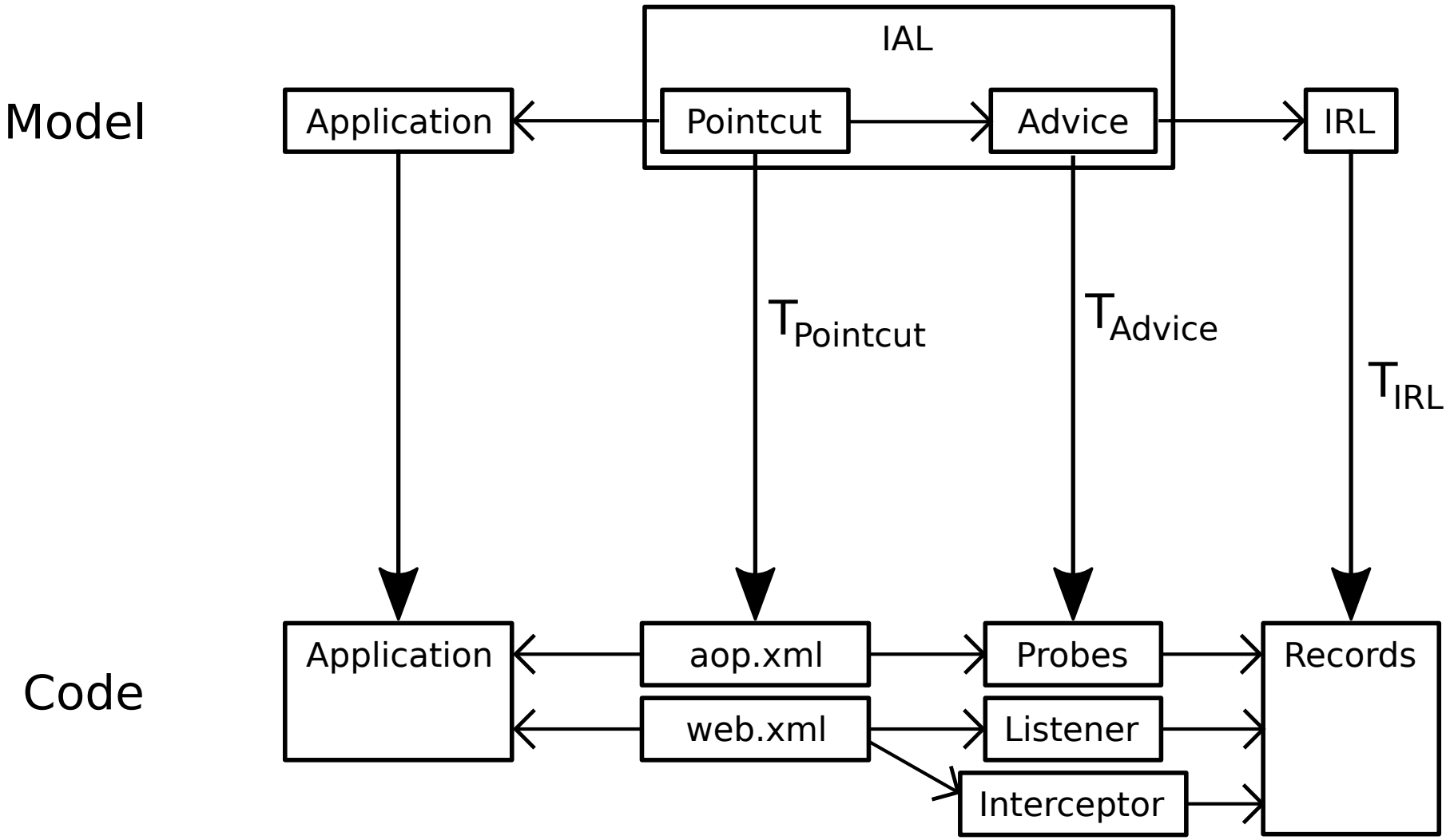
Jung and Wulf 2016; Jung, Heinrich, et al. 2016; Jung 2016

Aspect and Pointcut Modeling



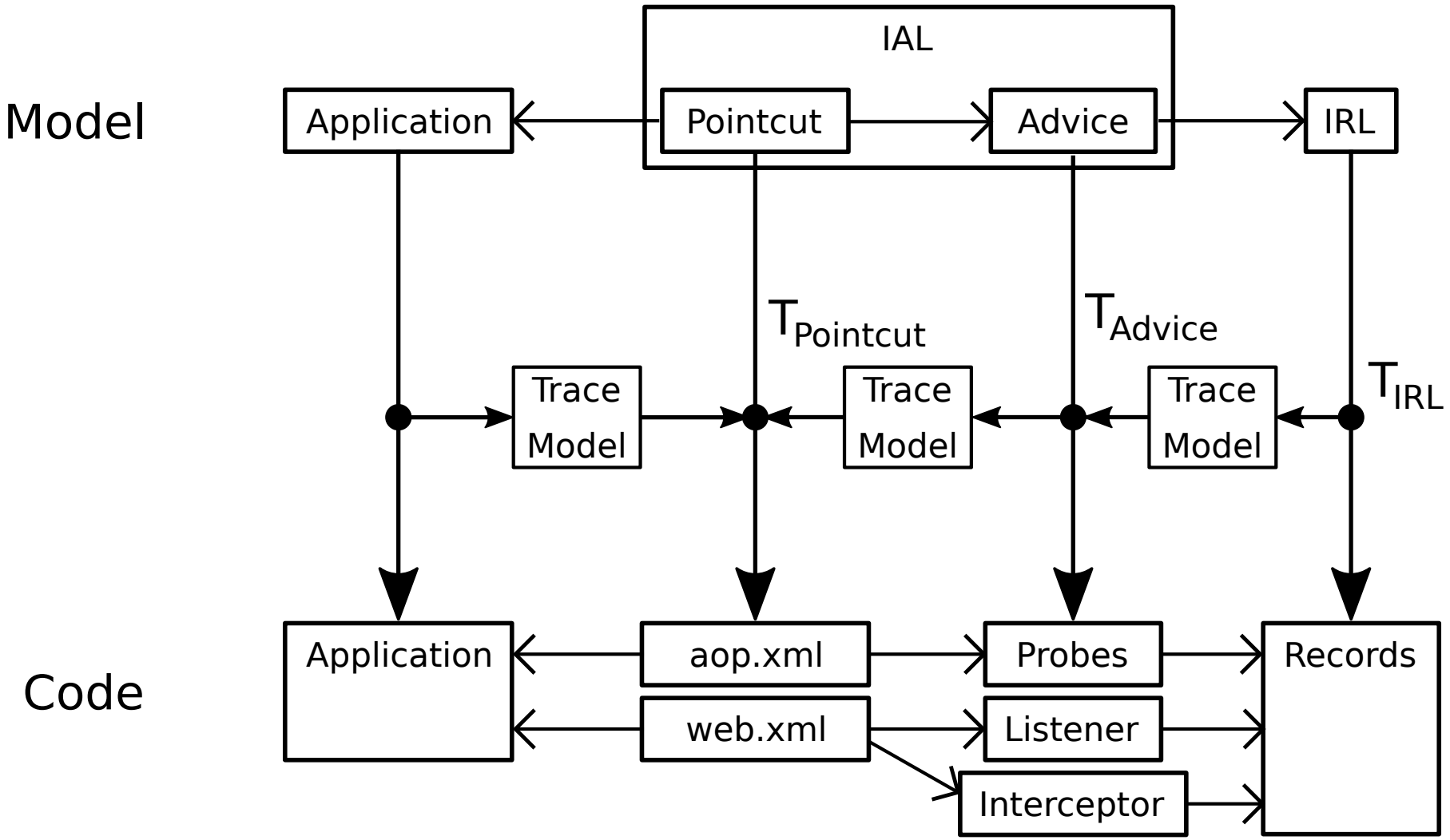
Jung and Wulf 2016; Jung, Heinrich, et al. 2016; Jung 2016

Aspect and Pointcut Modeling



Jung and Wulf 2016; Jung, Heinrich, et al. 2016; Jung 2016

Aspect and Pointcut Modeling



Jung and Wulf 2016; Jung, Heinrich, et al. 2016; Jung 2016

Specifying Instrumentation

```
package kieker.monitoring

import kieker.common.records.BeforeOperationEvent
import kieker.common.records.AfterOperationEvent
import kieker.common.records.AfterOperationExceptionEvent

map pcm model cocome "cocome-model.system" trace "cocome.rac"

aspect p1,p2 : exampleAdvice

pointcut p1 class cocome.org.cocome.tradingsystem.*
pointcut p2 class cocome.org.cocome.web.* {
    exclude cocome.org.cocome.web.data.*
}

advice exampleAdvice {
    before BeforeOperationEvent
    after AfterOperationEvent
    exception AfterOperationExceptionEvent
}
```

Specifying Instrumentation (Java)

```
package kieker.monitoring

import kieker.common.records.BeforeOperationEvent
import kieker.common.records.AfterOperationEvent
import kieker.common.records.AfterOperationExceptionEvent

map java model cocome "cocome-model.system"

aspect p1,p2 : exampleAdvice

@technology ejb
pointcut p1 class cocome.org.cocome.tradingsystem.*

@technology servlet
pointcut p2 class cocome.org.cocome.web.* {
    exclude cocome.org.cocome.web.data.*
}

advice exampleAdvice {
    before BeforeOperationEvent
    after AfterOperationEvent
    exception AfterOperationExceptionEvent
}
```

Conclusion

Summary

- Modeling monitoring data
  - Instrumentation Record Language
  - Used to specify all Kieker monitoring records
- Specifying monitoring aspect

Open Tasks

- Adaptation of IAL generators to new API (plug-ins)
- Complete PCM and Java IAL support

Github

`https://github.com/kieker-monitoring/instrumentation-languages`

Eclipse update site - snapshot

`https://build.se.informatik.uni-kiel.de/eus/mdm/snapshot/`

Bibliography I

Hasselbring, Wilhelm et al. (2013). *iObserve: Integrated Observation and Modeling Techniques to Support Adaptation and Evolution of Software Systems*. Technical Report. Kiel University.

Heinrich, Robert et al. (2016). *The CoCoME Platform for Collaborative Empirical Research on Information System Evolution*. Technical Report 2016,2; Karlsruhe Reports in Informatics. Karlsruhe Institute of Technology. URL: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000052688.

Hoorn, André van et al. (2012). "Kieker: A Framework for Application Performance Monitoring and Dynamic Software Analysis." In: *Proc. of the 3rd ACM/SPEC Int. Conf. on Performance Engineering*, pp. 247–248.

Jung, Reiner (2016). *Generator-Composition for Aspect-Oriented Domain-Specific Languages*. Vol. 2016/4. Kiel Computer Science Series. ISBN: 978-0-01-219888-9. URL: http://eprints.uni-kiel.de/33602/.

Bibliography II

Jung, Reiner, Robert Heinrich, et al. (2016). "GECO: A Generator Composition Approach for Aspect-Oriented DSLs." In: *Theory and Practice of Model Transformations - 9th International Conference, ICMT 2016*, pp. 141-156. DOI: 10.1007/978-3-319-42064-6_10.

Jung, Reiner and Christian Wulf (2016). "Advanced Typing for the Kieker Instrumentation Languages." In: *Softwaretechnik-Trends* 36.4. URL: http://pi.informatik.uni-siegen.de/stt/36_4/./01_Fachgruppenberichte/SSP2016/ssp-stt/18-Advanced_Typing_for_the_Kieker_Instrumentation_Languages.pdf.