Christian-Albrechts-Universität zu Kiel

Rekonstruktion von Softwarearchitekturen durch dynamische Analyse

Prof. Dr. Wilhelm Hasselbring

mit Beiträgen von
André van Hoorn, Nils Ehmke, Sören Frey, Reiner Jung,
Holger Knoche (b+m), Matthias Rohr, Jan Waller
und weiteren

Software Engineering Group, Univ. Kiel
`http://se.informatik.uni-kiel.de/`

GI-Fachtagung Architekturen 2012

Paderborn, 2. Juli 2012

Unterschiedliche Zielsetzungen und Werkzeuge, einige Beispiele:

Code Metriken	JavaNCSS
Programmierrichtlinien	Checkstyle
Fehlermuster	FindBugs
Architekturanalyse	Sotograph
Klonerkennung	Bauhaus

Statische Analyse kann bei Vorliegen des Quellcodes direkt angewendet werden um Qualitätseigenschaften zu überprüfen.

Dynamische Analyse des Laufzeitverhaltens

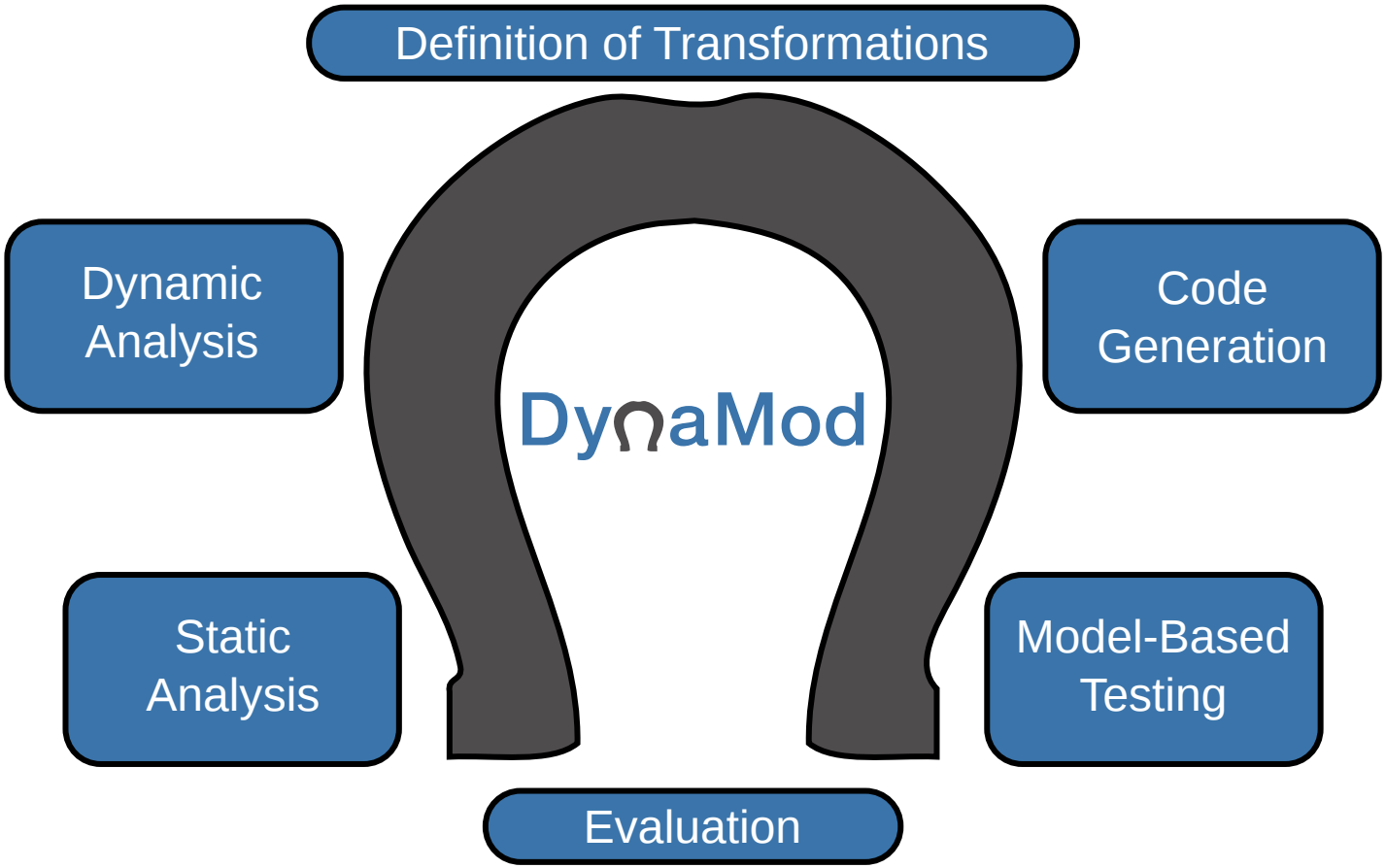
- Statische Analyse kann Einblicke in die Struktur eines Softwaresystems liefern,
  - das **tatsächliche** Laufzeitverhalten kann jedoch nur durch eine dynamische Analyse ermittelt werden.
- Profiling (in der Entwicklung) ermöglicht eine dynamische Analyse,
  - üblicherweise jedoch nicht unter realen Lastbedingungen.
- Monitoring (im Betrieb) ermöglicht die kontinuierliche dynamische Analyse
  - des tatsächlichen Laufzeitverhaltens unter realen Lastbedingungen.
- Ein Verständnis der Bedürfnisse des **Betriebs** von Softwaresystemen wird benötigt, um
  - im **Konstruktionsprozess** die richtigen Aktivitäten der
  - zum **Monitoring** für den jeweiligen **Einsatzzweck** benötigten Instrumentierung der Software durchzuführen.

Historisch gewachsene Architekturen

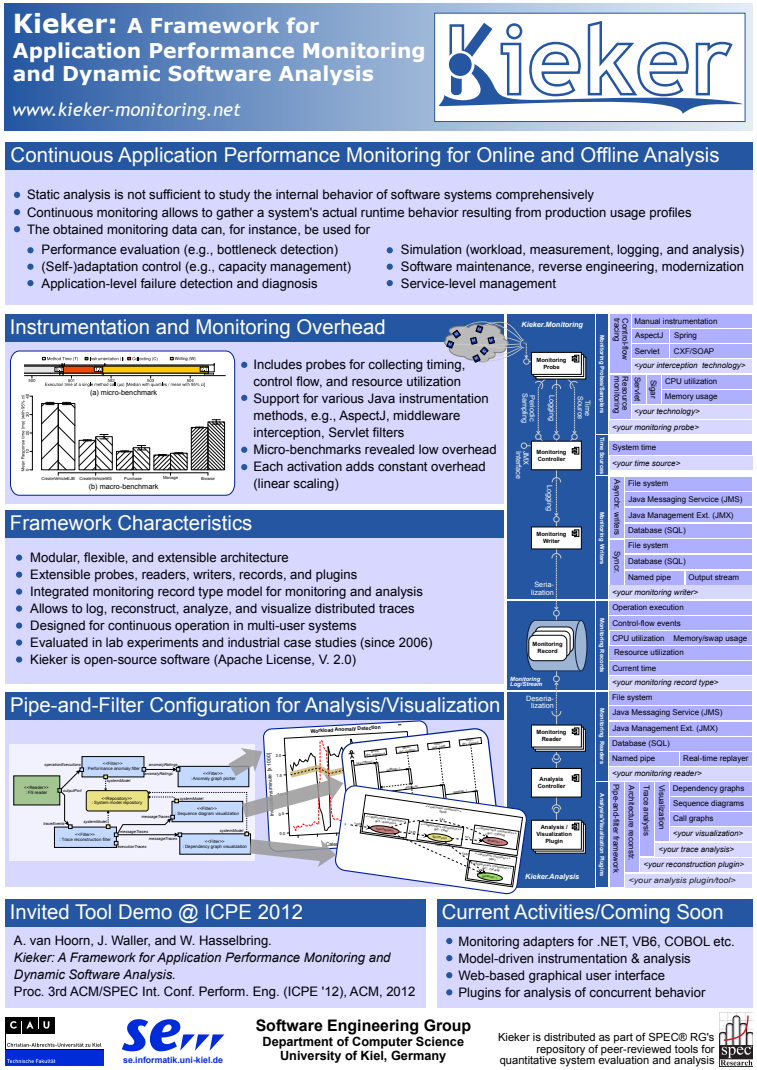
KoSSE-Projekt DynaMod

Dynamische Analyse zur modellgetriebenen Modernisierung

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

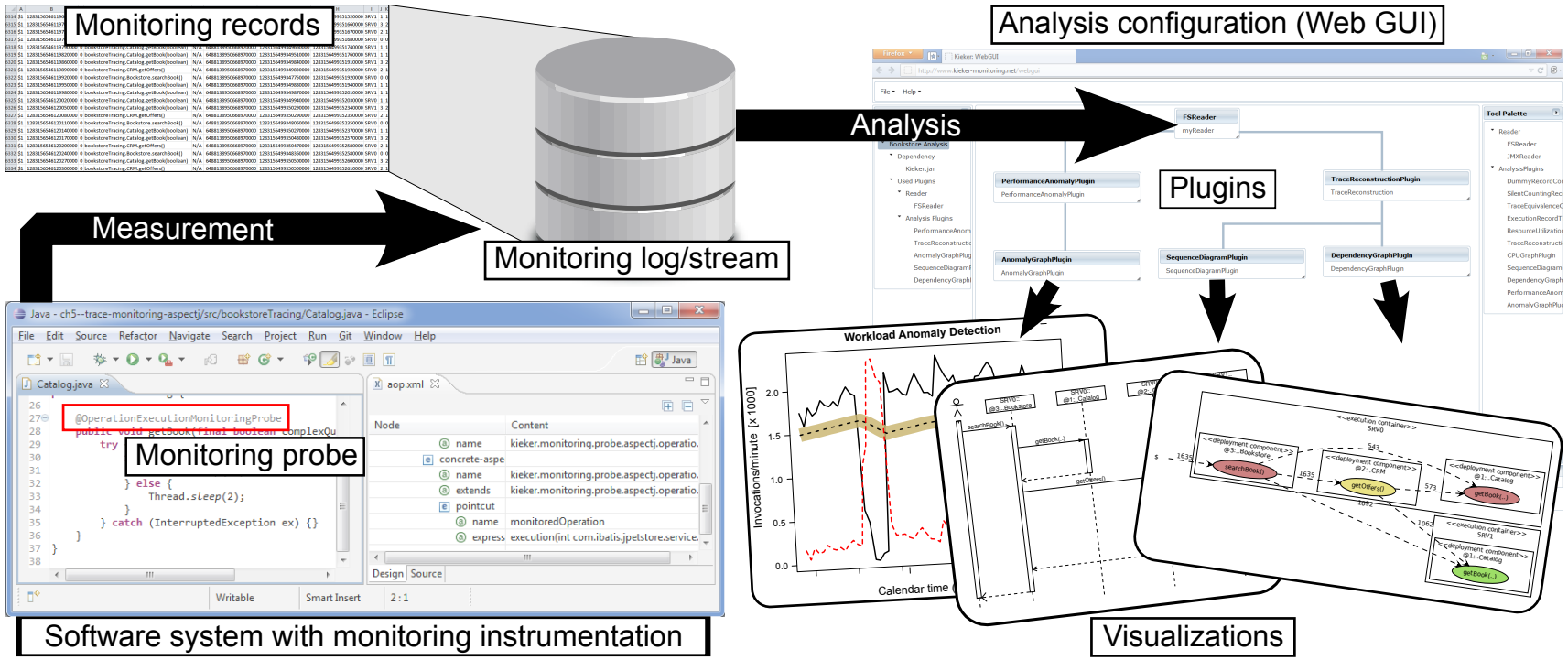

Definition of Transformations
Dynamic Analysis
Code Generation
DynaMod
Static Analysis
Model-Based Testing
Evaluation

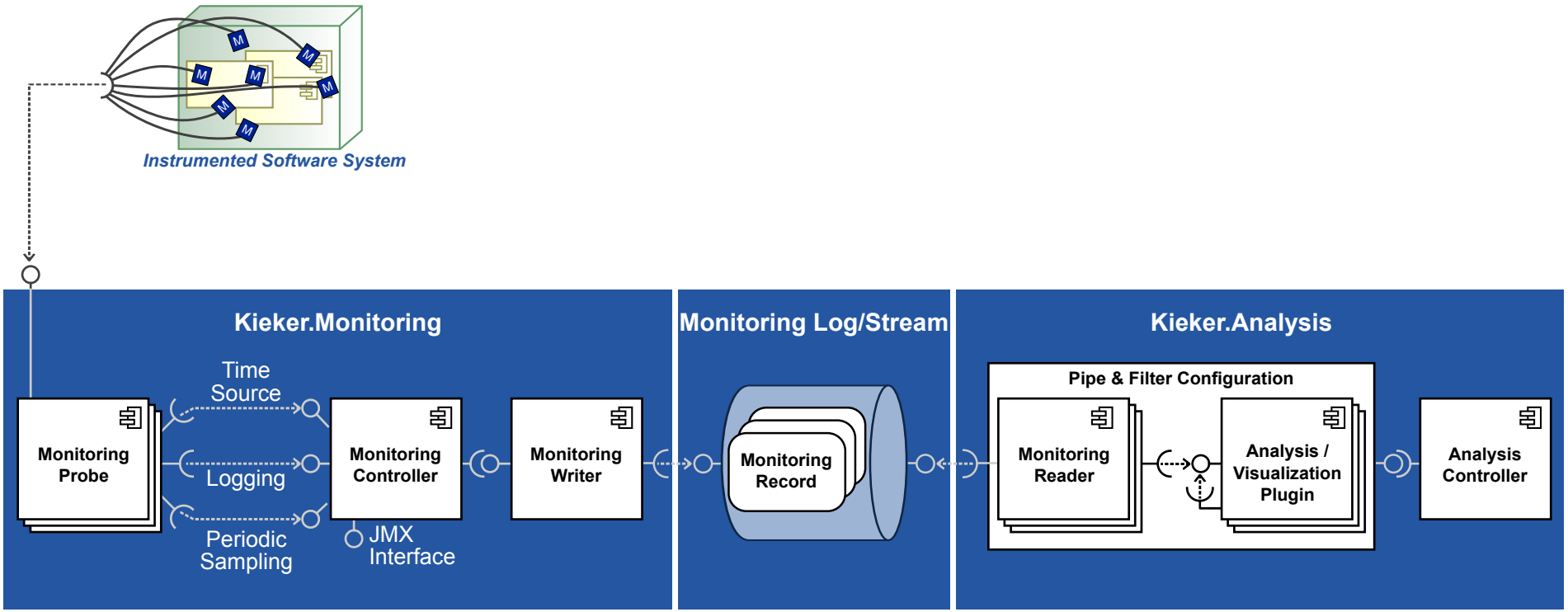
Agenda



Kieker: Nutzungsszenario

Christian-Albrechts-Universität zu Kiel




Instrumented Software System
Kieker.Monitoring
Time Source
Logging
Periodic Sampling
Monitoring Probe
Monitoring Controller
JMX Interface
Monitoring Writer
Monitoring Log/Stream
Monitoring Record
Kieker.Analysis
Pipe & Filter Configuration
Monitoring Reader
Analysis / Visualization Plugin
Analysis Controller

Kieker

- Flexible, modular erweiterbare Architektur (Messsonden, Log-Datenstrukturen, Analyse Plug-Ins, etc.)
- Verteiltes Tracing (Spring, SOAP, etc.)
- Niedriger Overhead (entworfen für den kontinuierlichen Betrieb)
- Evaluiert im Labor und in industriellen Umgebungen

b+m business IT management ALLGEIER GROUP · EWE · cewe color · dataport · HSH NORDBANK · XING · SAP

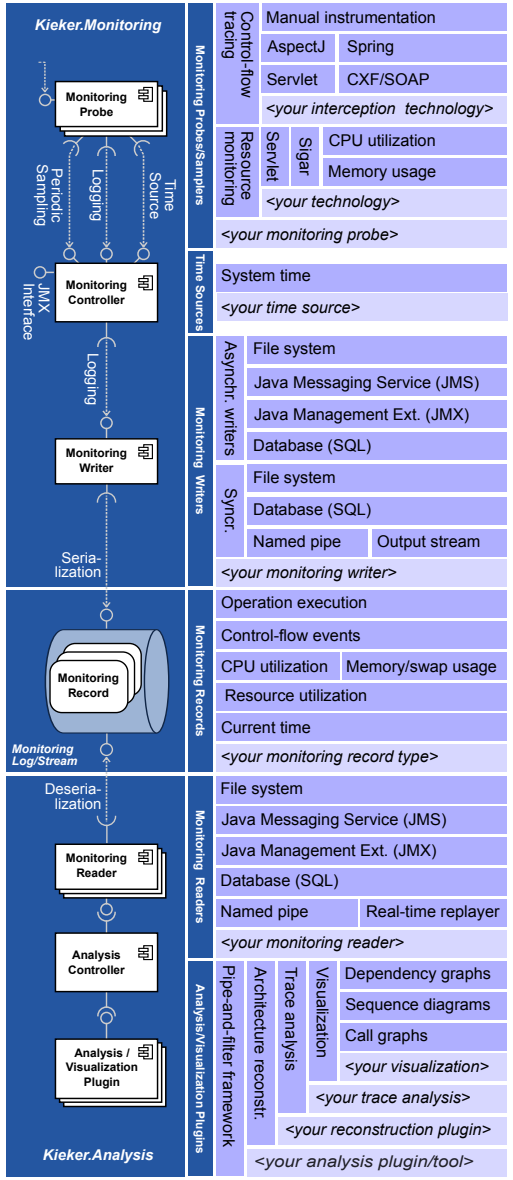
Kieker ist Open-Source Software (Apache License, V. 2.0)

`http://kieker-monitoring.net`

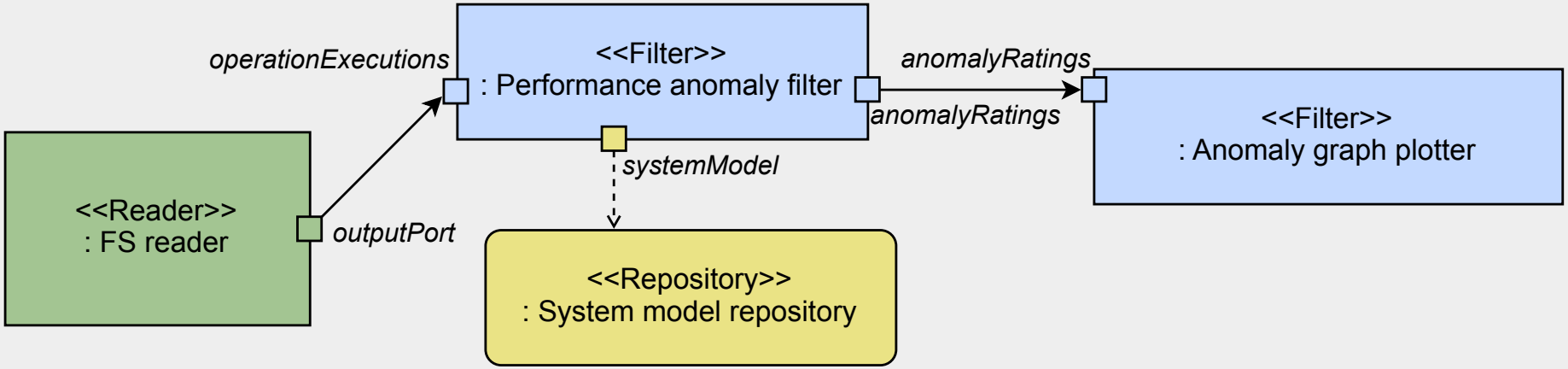
Empfohlenes Tool der SPEC Research Group

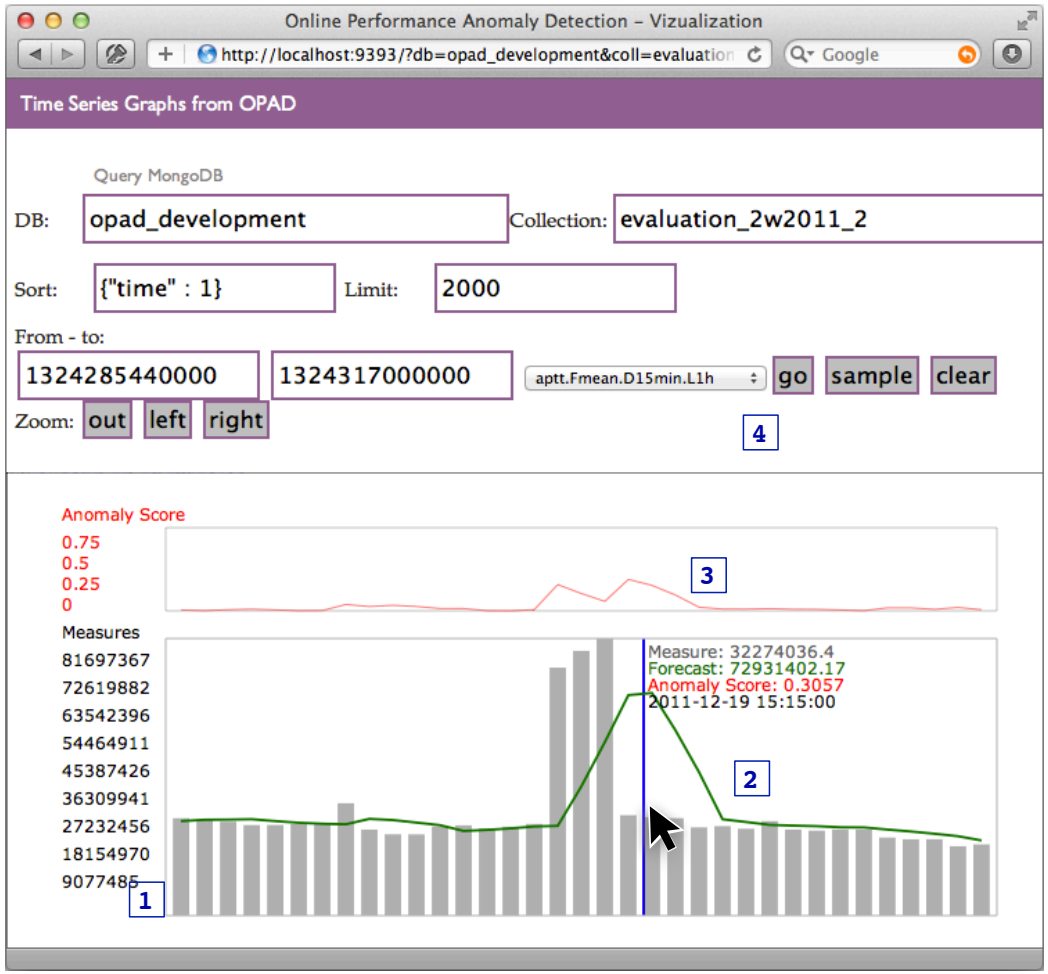
Kieker is distributed as part of SPEC RG's repository of peer-reviewed tools for quantitative system evaluation and analysis,
`http://research.spec.org/projects/tools.html`

spec® Research



Beispiel Pipe-and-Filter Konfiguration

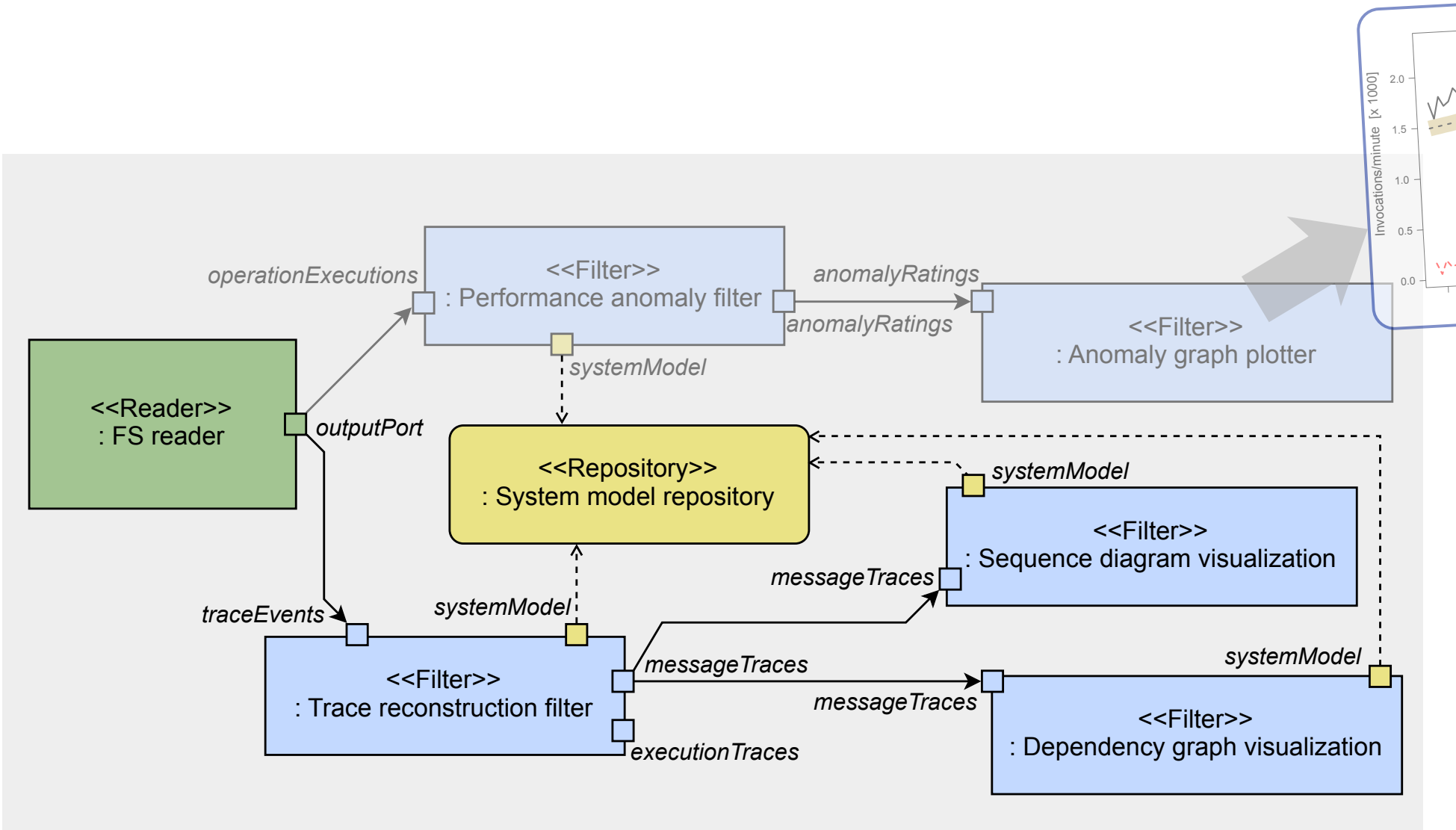




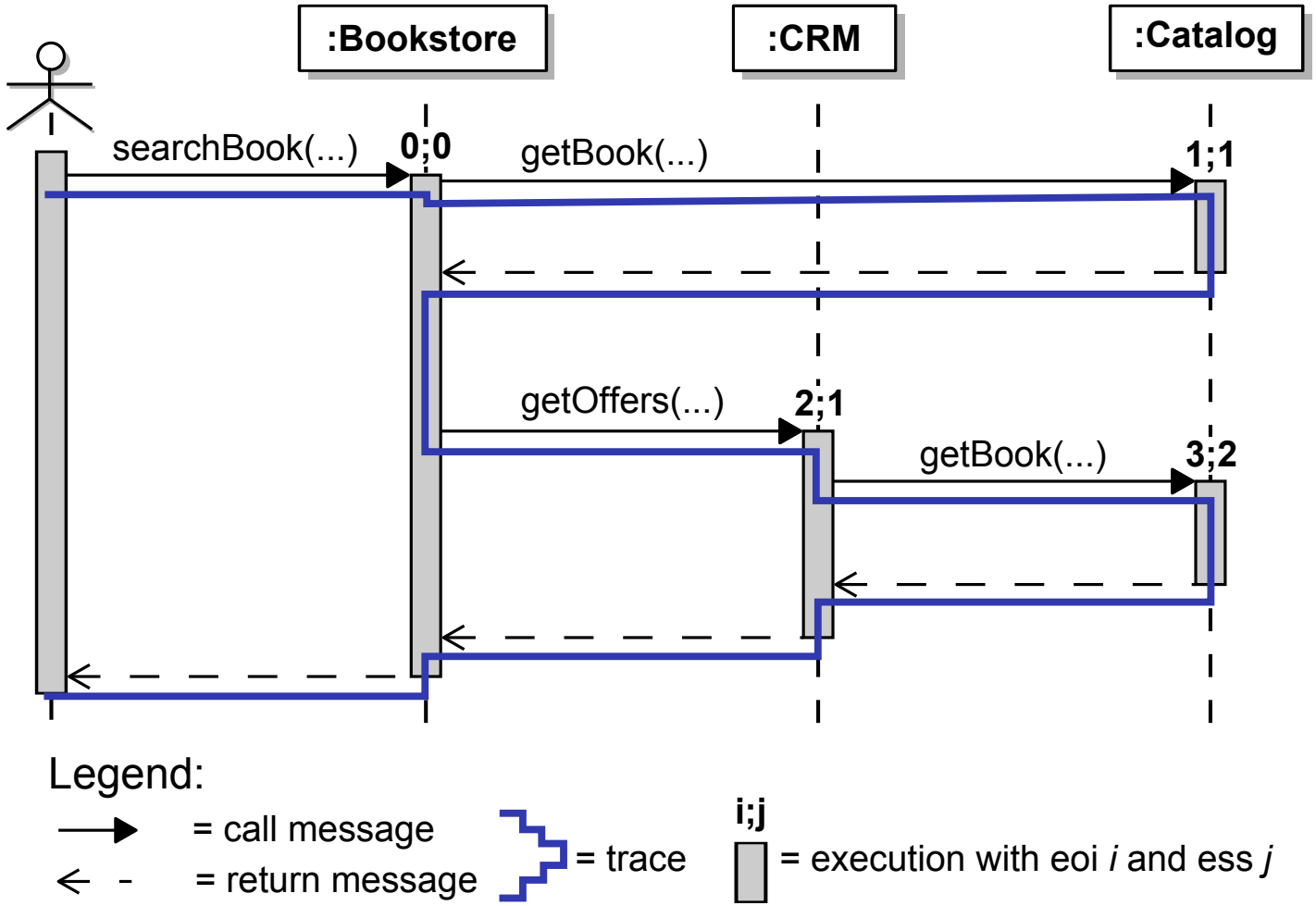
Entwickelt und eingesetzt bei der Xing AG [Bielefeld 2012].

Beispiel Pipe-and-Filter Konfiguration

Christian-Albrechts-Universität zu Kiel



Trace Monitoring

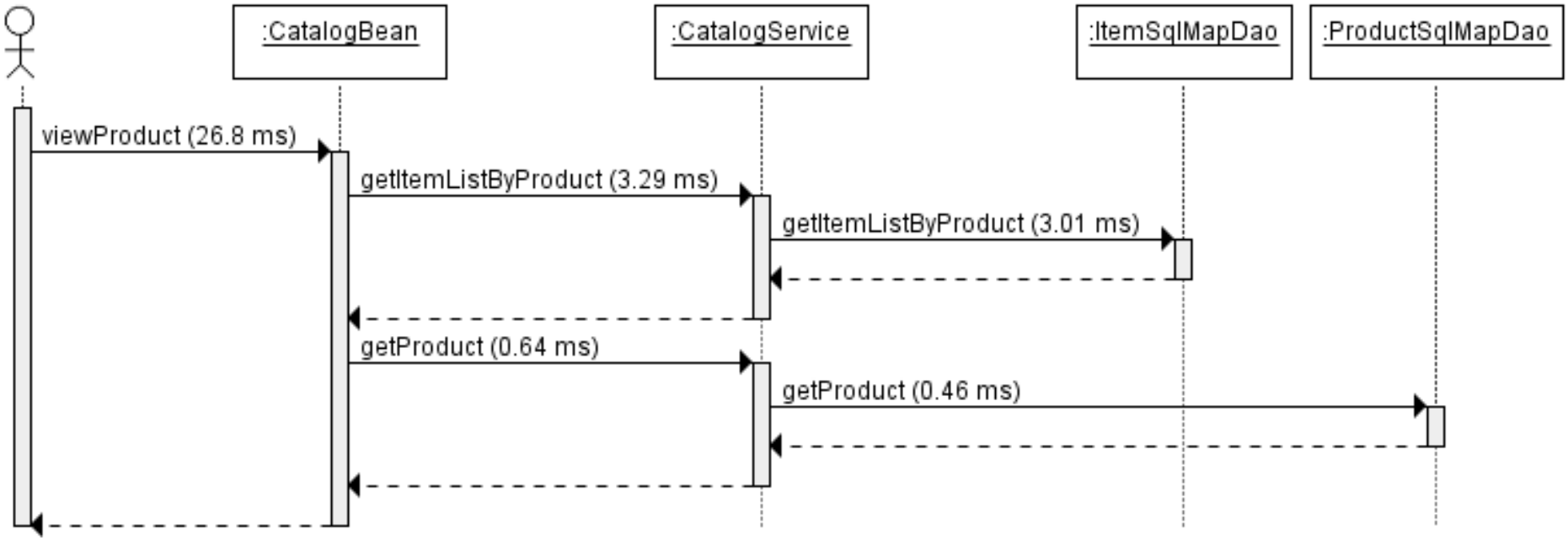


Execution order index (eoi) *i*: *i*-th started execution in a trace

Execution stack size (ess) *j*: execution started at stack depth *j*

Generiertes Sequenzdiagramm

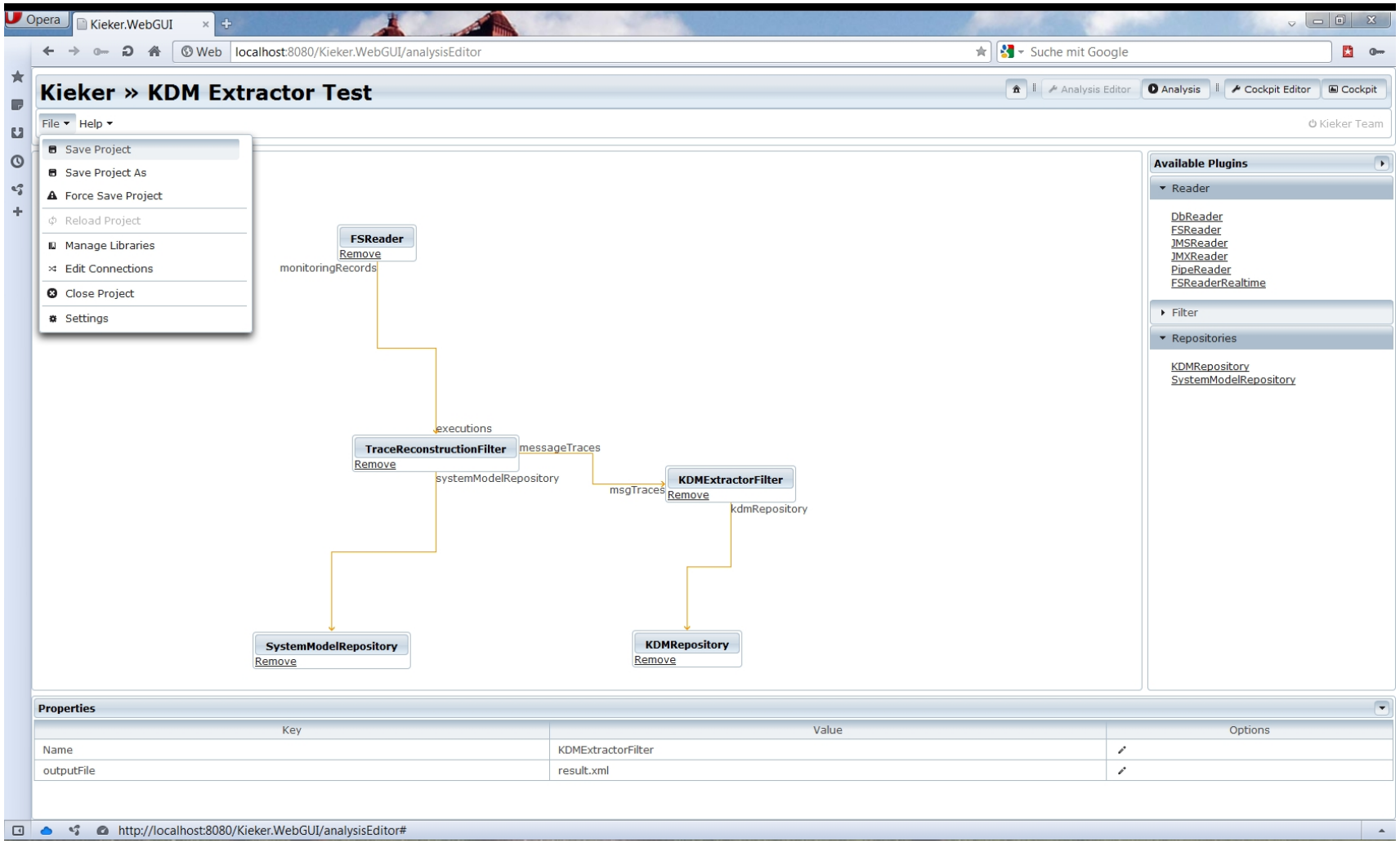
Visualisierungsbeispiel für die dynamische Analyse



Web GUI zur Konfiguration der Analyse

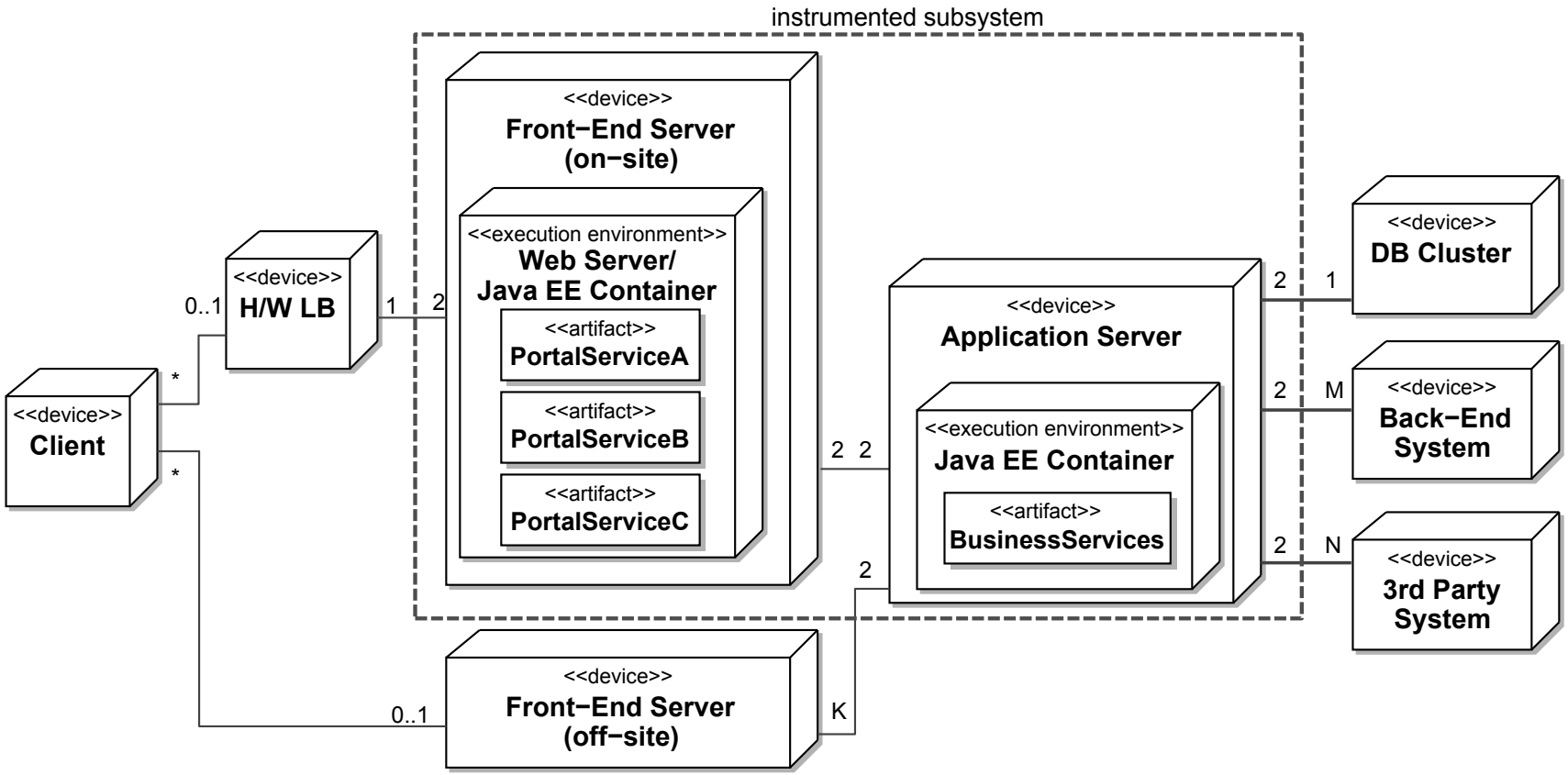
(in Arbeit)

Christian-Albrechts-Universität zu Kiel

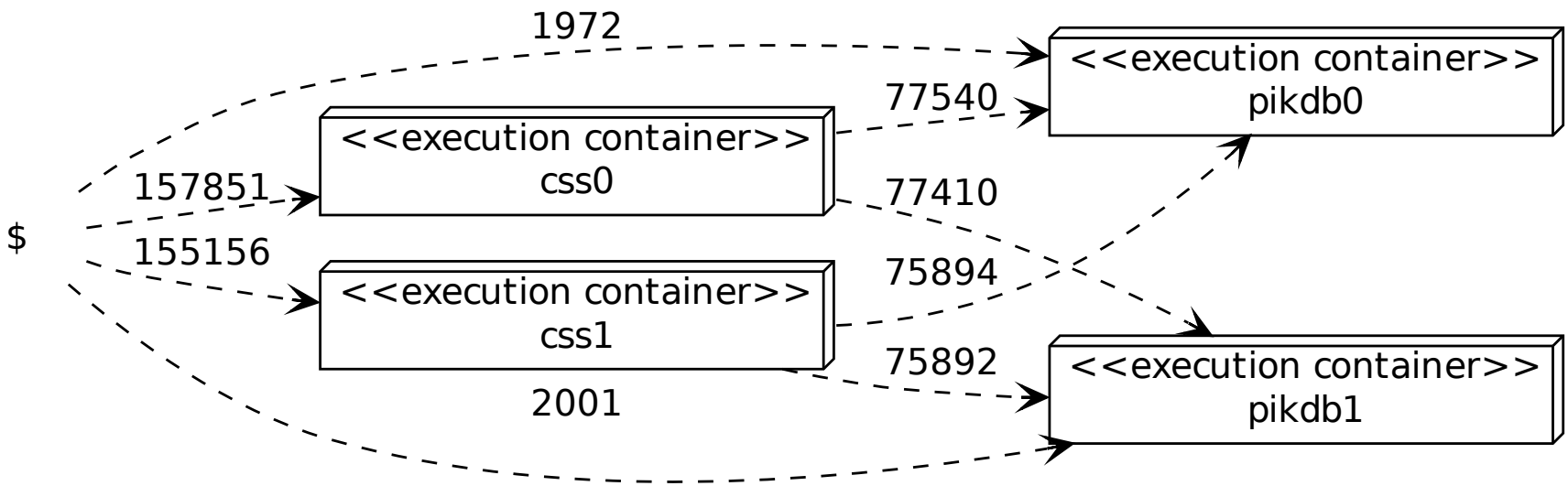


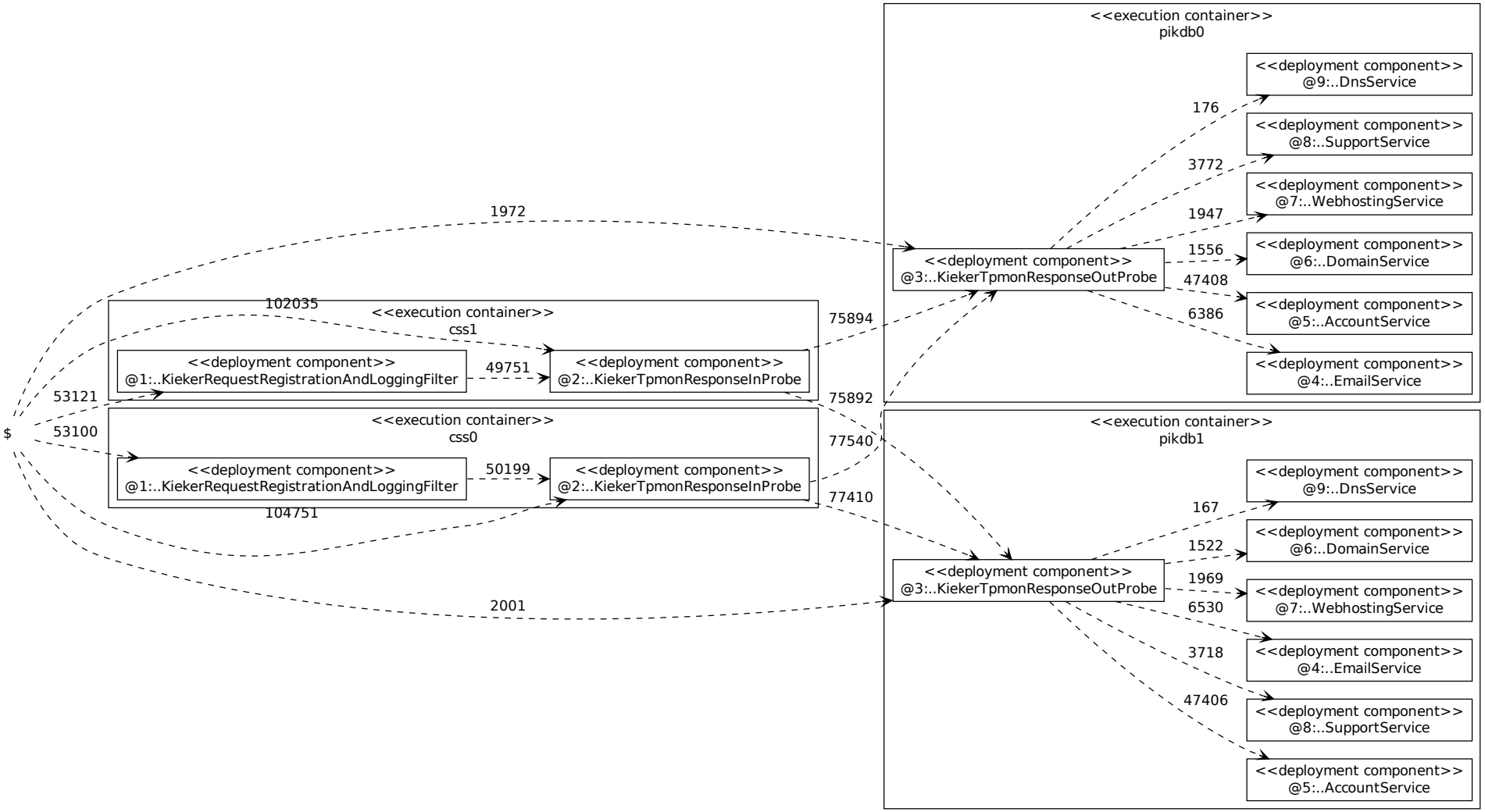
Agenda

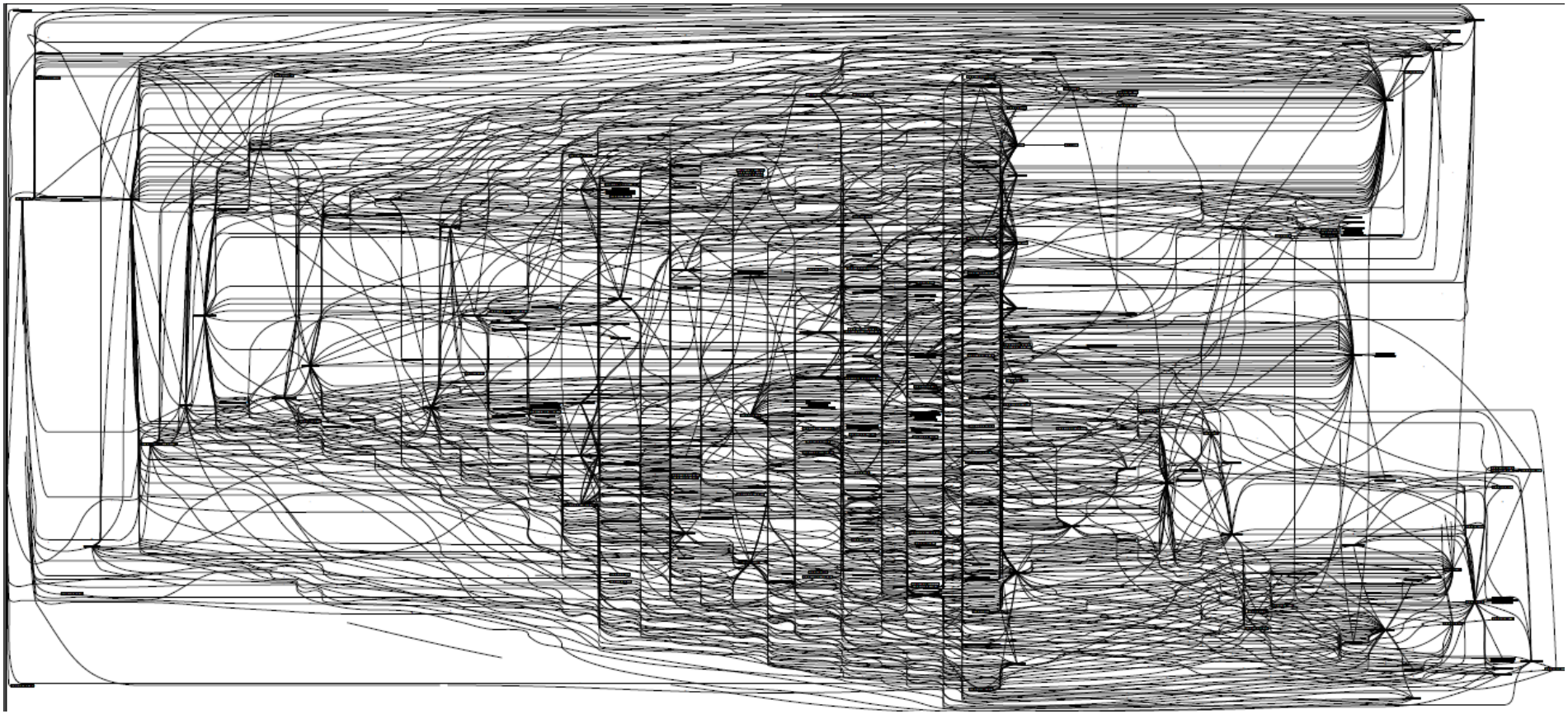
Verteiltes Enterprise Java System



Servlet, Spring und CXF/SOAP Messsonden

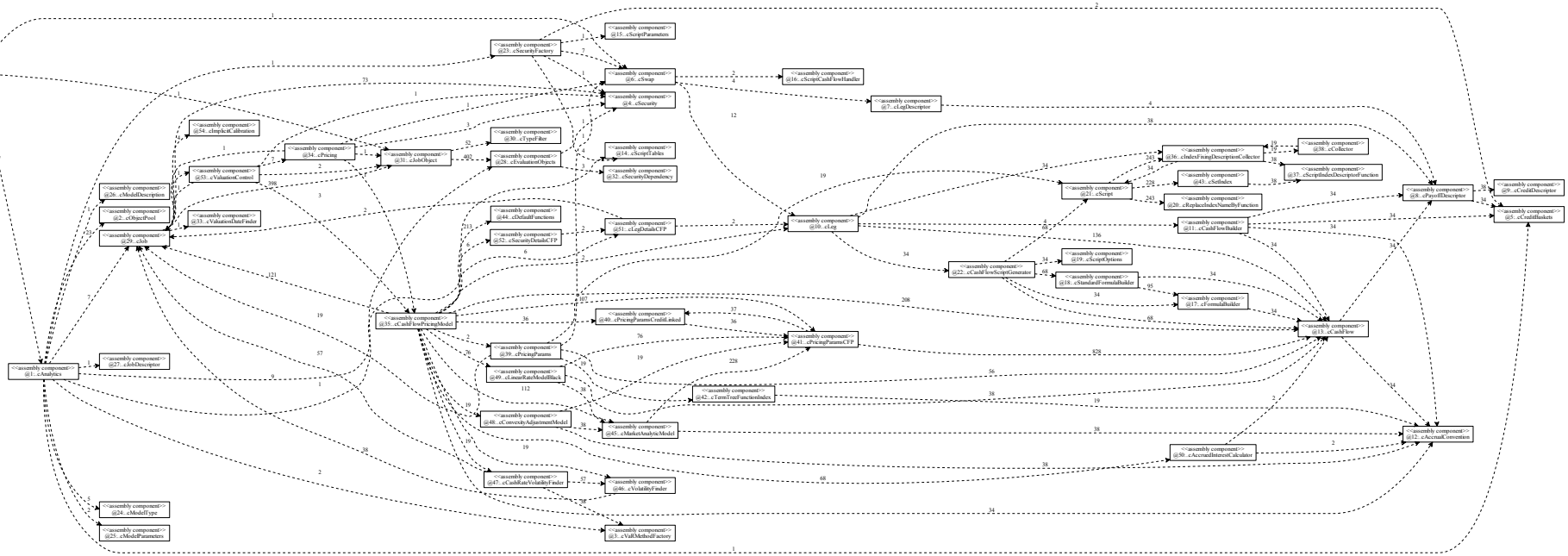

1972
77540
157851
77410
$
155156
75894
75892
2001
<<execution container>>
pikdb0
<<execution container>>
css0
<<execution container>>
css1
<<execution container>>
pikdb1


<<execution container>>
pikdb0
<<deployment component>>
@9:..DnsService
<<deployment component>>
@8:..SupportService
<<deployment component>>
@7:..WebhostingService
<<deployment component>>
@6:..DomainService
<<deployment component>>
@5:..AccountService
<<deployment component>>
@4:..EmailService
<<deployment component>>
@3:..KiekerTpmonResponseOutProbe
176
3772
1947
1556
47408
6386
1972
102035
<<execution container>>
css1
<<deployment component>>
@1:..KiekerRequestRegistrationAndLoggingFilter
49751
<<deployment component>>
@2:..KiekerTpmonResponseInProbe
75894
75892
53121
$
53100
<<execution container>>
css0
<<deployment component>>
@1:..KiekerRequestRegistrationAndLoggingFilter
50199
<<deployment component>>
@2:..KiekerTpmonResponseInProbe
77540
77410
104751
2001
<<execution container>>
pikdb1
<<deployment component>>
@9:..DnsService
<<deployment component>>
@6:..DomainService
<<deployment component>>
@7:..WebhostingService
<<deployment component>>
@4:..EmailService
<<deployment component>>
@8:..SupportService
<<deployment component>>
@5:..AccountService
<<deployment component>>
@3:..KiekerTpmonResponseOutProbe
167
1522
1969
6530
3718
47406



Fallstudie bei der HSH Nordbank AG.

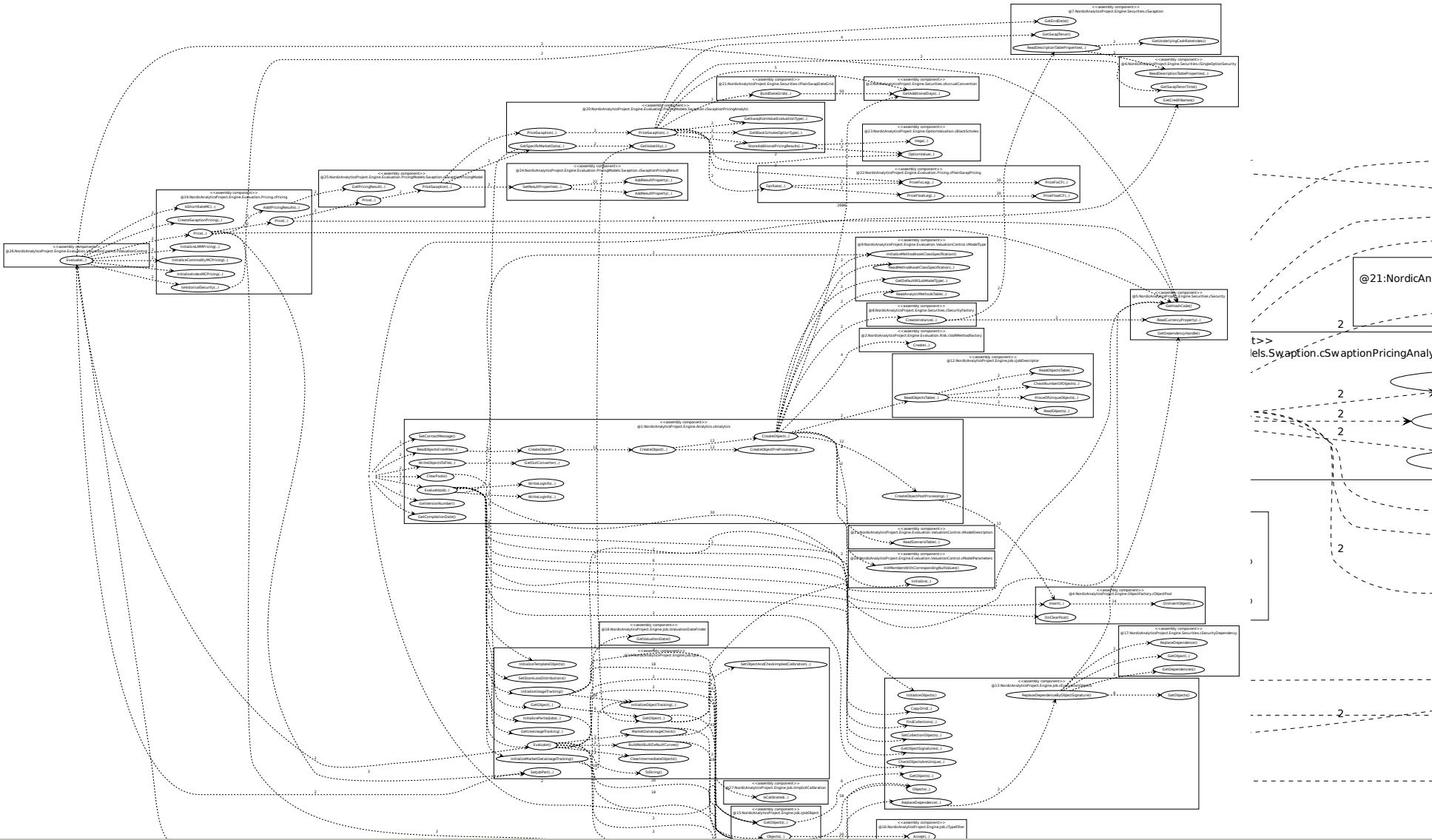
HSH NORDBANK



Zoom auf die Operationsebene

Reverse Engineering von C#

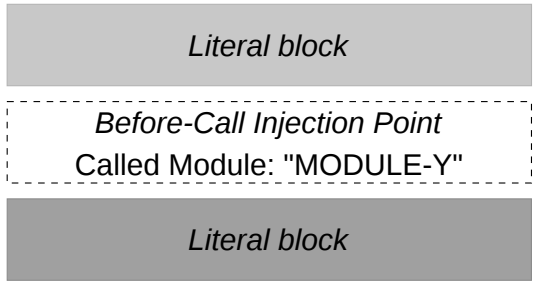
C A U
Christian-Albrechts-Universität zu Kiel



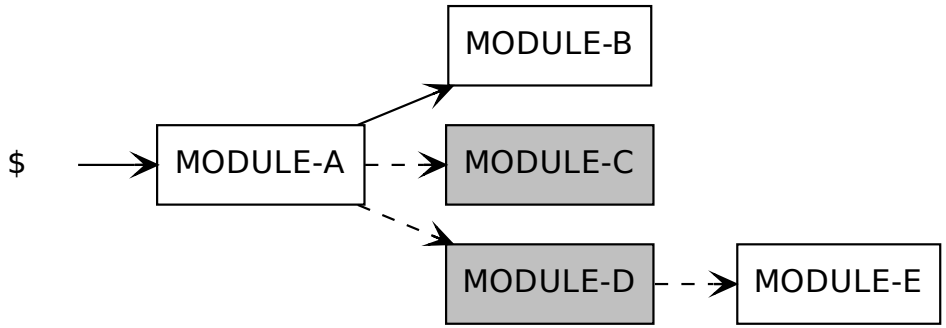
```
IDENTIFICATION DIVISION.
PROGRAM-ID. MODULE-X.

PROCEDURE DIVISION.

    CALL "MODULE-Y".
    GOBACK.
```



AOP-based COBOL instrumentation

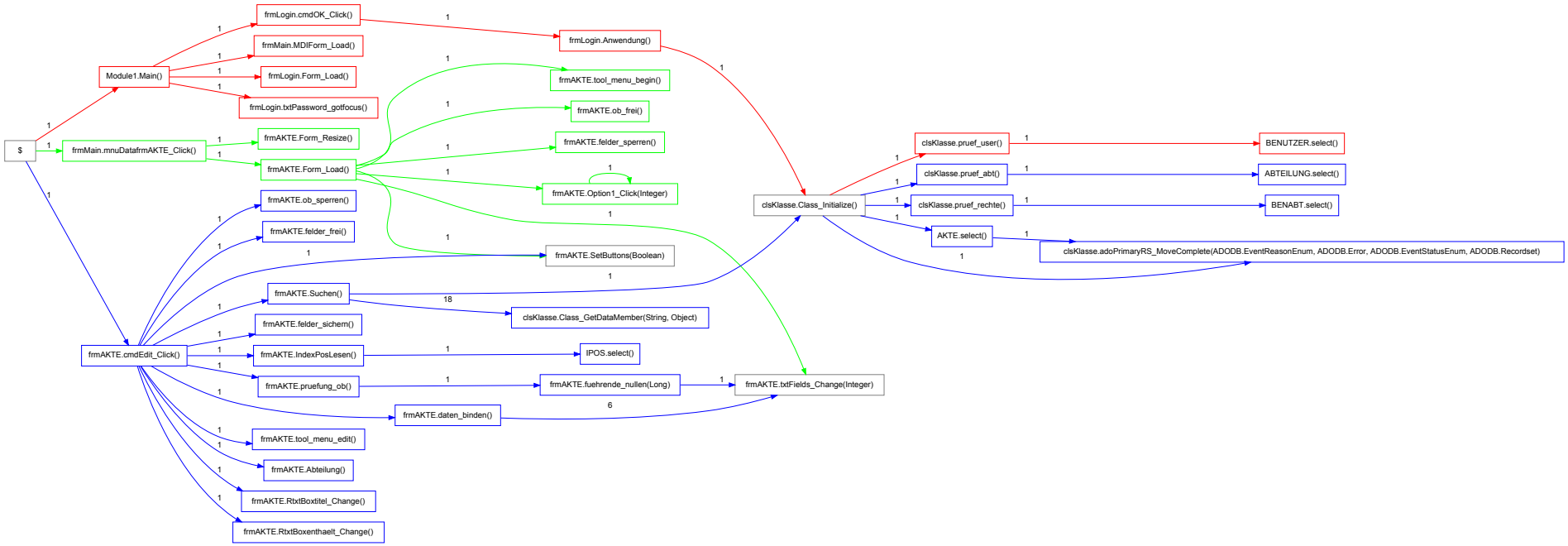


Module-level call dependency graph with assumed dependencies

Fallstudie der b+m Informatik AG (Kieker-Erweiterungen durch Holger Knoche):

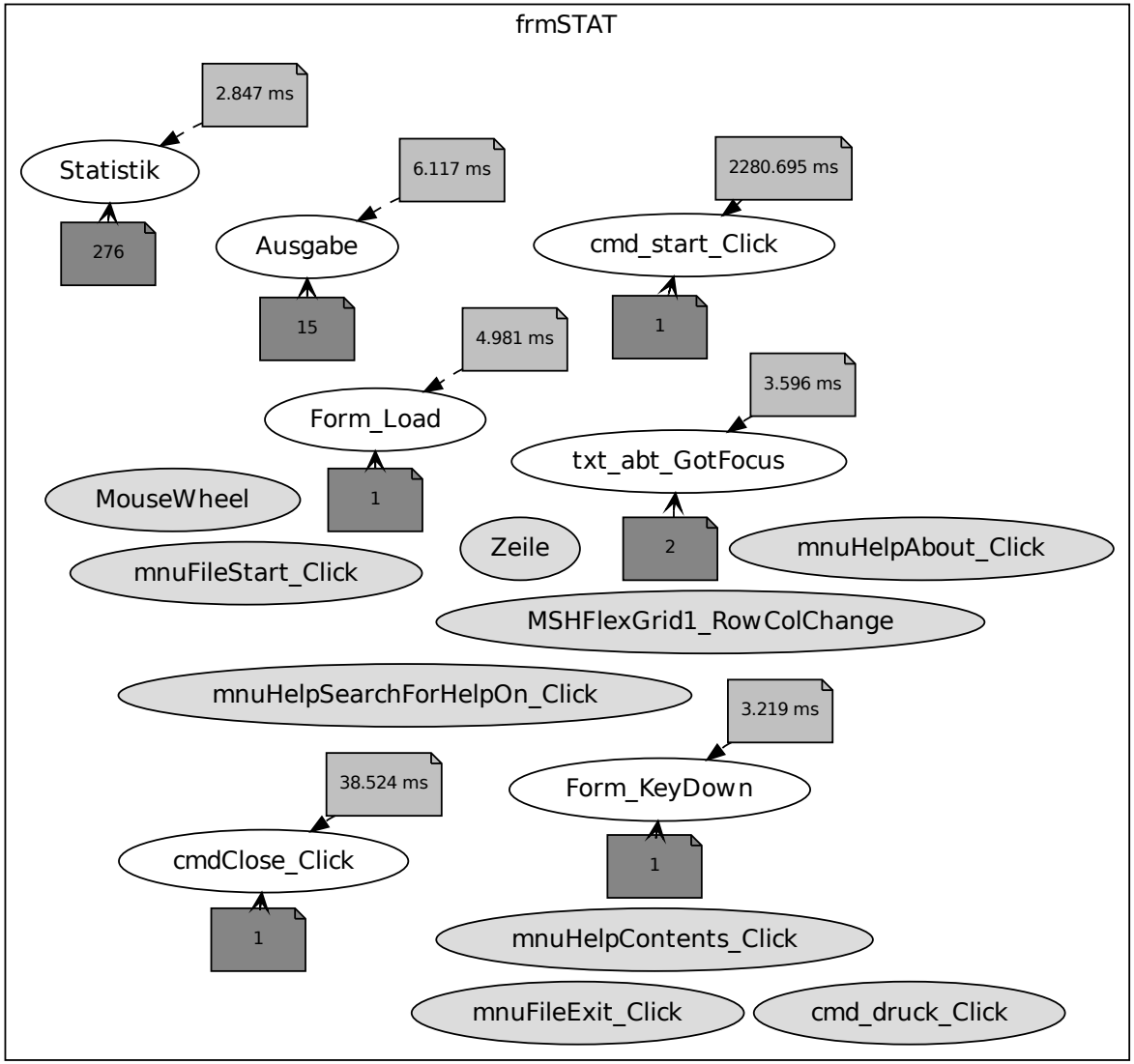
1. Industrieller Kontext mit mehr als 1.000 COBOL Modulen.
2. 140.351 Messpunkte mit 14 verschiedenen Arten von Messsonden.

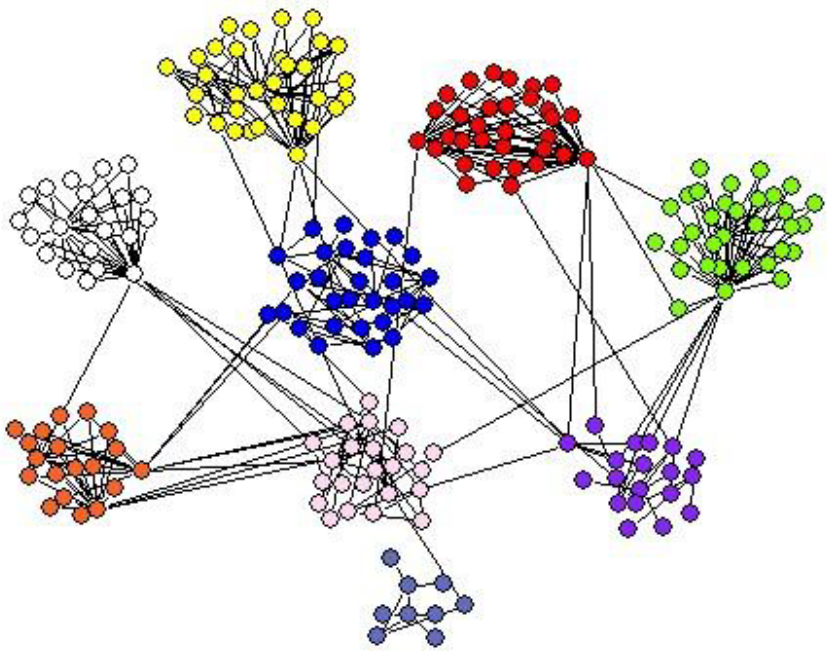
C A U
Christian-Albrechts-Universität zu Kiel



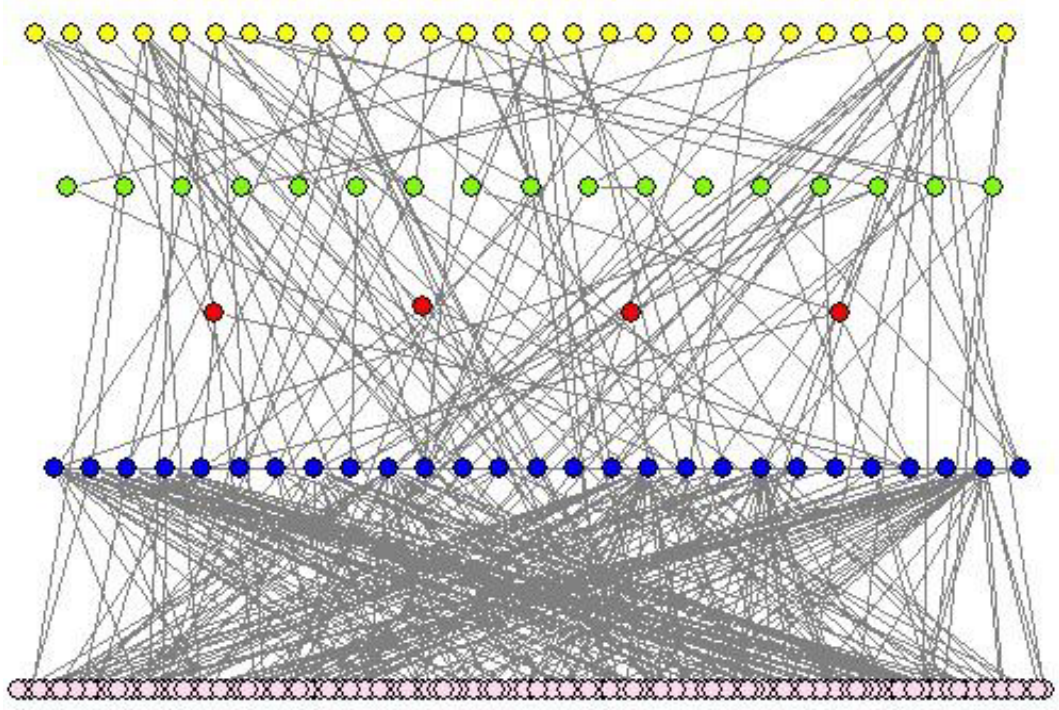
dataport

Fallstudie bei Dataport
(Kieker-Erweiterungen durch Holger Knoche).


frmSTAT
2.847 ms
Statistik
276
6.117 ms
Ausgabe
15
2280.695 ms
cmd_start_Click
1
4.981 ms
Form_Load
1
3.596 ms
txt_abt_GotFocus
2
MouseWheel
mnuFileStart_Click
Zeile
mnuHelpAbout_Click
MSHFlexGrid1_RowColChange
mnuHelpSearchForHelpOn_Click
3.219 ms
Form_KeyDown
1
38.524 ms
cmdClose_Click
1
mnuHelpContents_Click
mnuFileExit_Click
cmd_druck_Click

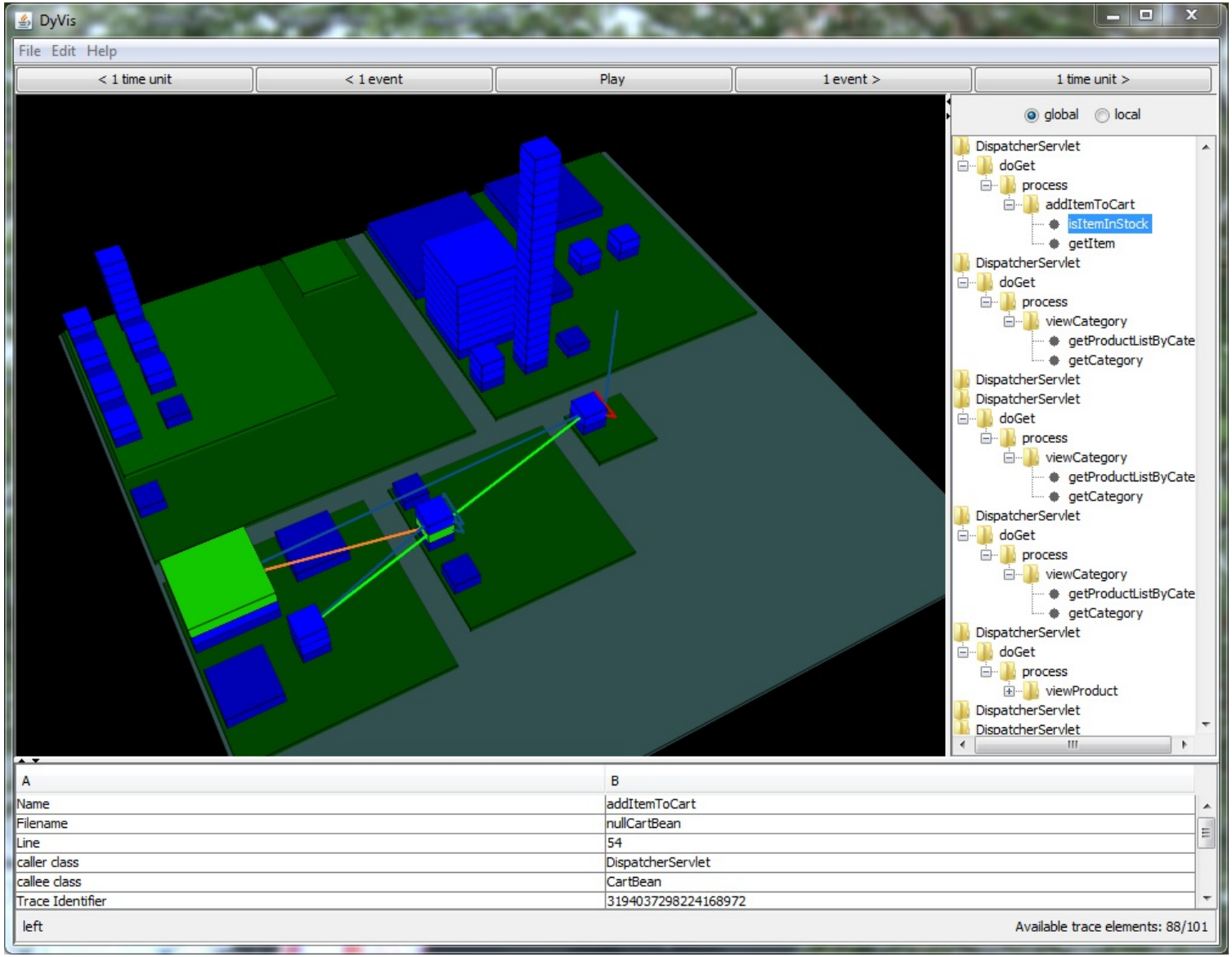


Community structure



Layered structure (hierarchy)

Xi'an Jiaotong University, Shaanxi [Zheng et al. 2011]


DyVis
File Edit Help
< 1 time unit
< 1 event
Play
1 event >
1 time unit >
global
local
DispatcherServlet
doGet
process
addItemToCart
isItemInStock
getItem
viewCategory
getProductListByCate
getCategory
viewProduct
A
B
Name
addItemToCart
Filename
nullCartBean
Line
54
caller class
DispatcherServlet
callee class
CartBean
Trace Identifier
3194037298224168972
left
Available trace elements: 88/101

Zusammenfassung

- Das Monitoring zur dynamischen Analyse liefert wichtige Daten über das **tatsächliche** Verhalten eines Systems.
  - Kombiniert mit statischer Analyse.
- Die Monitoring-Daten können zur **Rekonstruktion** der Softwarearchitekturen genutzt werden.
- Für die Architektur-Rekonstruktion sind **flexible Anpassungsmöglichkeiten** erforderlich.
- In der Entwicklung sind spezifische **Entwurfsentscheidungen** für das Monitoring erforderlich.

Aktuelle Arbeiten u.a.:

- Web-basierte Konfigurations-Oberfläche & Cockpit
- Model-driven instrumentation & analysis
- Einsatz für Workflow-Monitoring und zum Profiling eingebetteter Systeme

T. C. Bielefeld. Online performance anomaly detection for large-scale software systems. Diploma thesis, University of Kiel, Germany, Mar. 2012.

H. Knoche, A. van Hoorn, W. Goerigk, and W. Hasselbring. Automated source-level instrumentation for dynamic dependency analysis of COBOL systems. In *Proceedings of the 14. Workshop Software-Reengineering (WSR '12)*, pages 33–34, May 2012.

F. Magendanz. Dynamic analysis of .NET applications for architecture-based model extraction and test generation, Oct. 2011.

A. van Hoorn, M. Rohr, W. Hasselbring, J. Waller, J. Ehlers, S. Frey, and D. Kieselhorst. Continuous monitoring of software services: Design and application of the Kieker framework. Technical Report TR-0921, Department of Computer Science, University of Kiel, Germany, Nov. 2009. URL `http://www.informatik.uni-kiel.de/uploads/tx_publication/vanhoorn_tr0921.pdf`.

A. van Hoorn, J. Waller, and W. Hasselbring. Kieker: A framework for application performance monitoring and dynamic software analysis. In *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering (ICPE 2012)*, pages 247–248. ACM, Apr. 2012. ISBN 978-1-4503-1202-8.

C. Wulf. Runtime visualization of static and dynamic architectural views of a software system to identify performance problems. B.Sc. Thesis, University of Kiel, Germany, 2010.

Q. Zheng, Z. Ou, L. Liu, and T. Liu. A novel method on software structure evaluation. In *Proceedings of the 2nd IEEE International Conference on Software Engineering and Service (IEEE ICSESS 2011)*, pages 251–254. IEEE, July 2011. doi: 10.1109/ICSESS.2011.5982301.