# Open-Source-Software als Katalysator im Technologietransfer am Beispiel des Monitoring-Frameworks Kieker

Wilhelm Hasselbring[1] & André van Hoorn[2]

[1] Kiel University (CAU)
Software Engineering Group
&
[2] University of Stuttgart
Reliable Software Systems Group

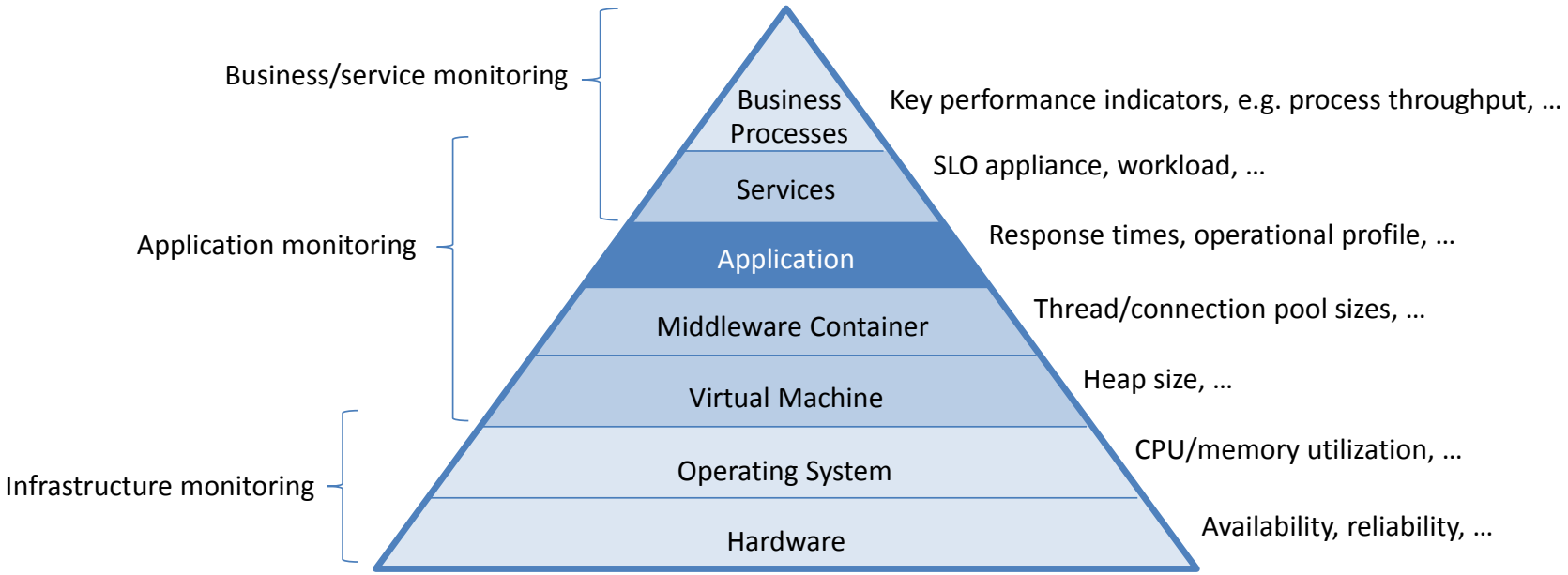

1. Kieler Open Source Business Konferenz

14. September 2015 @ Kiel

# Monitoring on different system layers



Christian-Albrechts-Universität zu Kiel




## Monitoring practice in the “real world” (based on what we’ve seen)

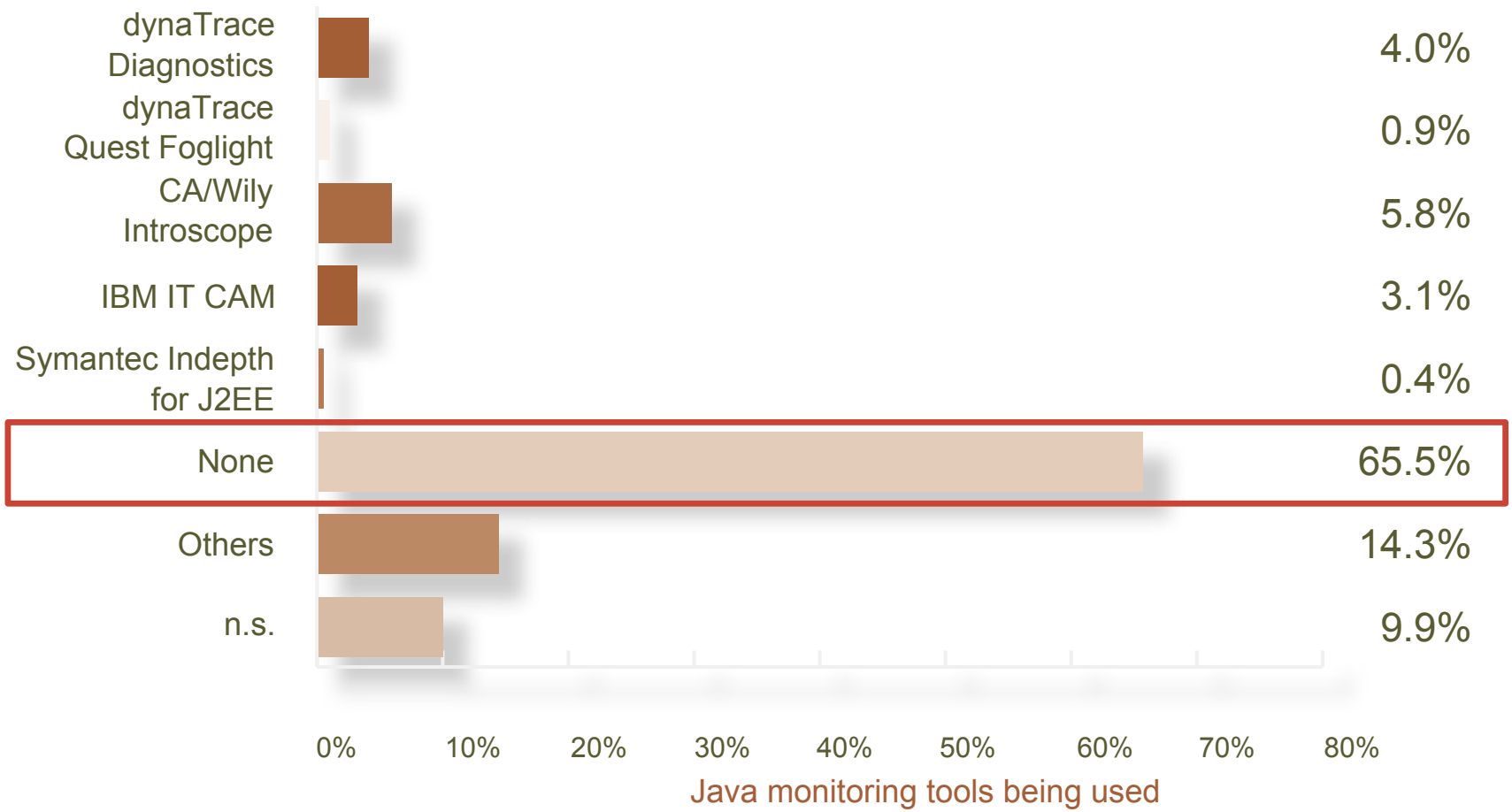
- Focus on system level (network availability, resource utilization) or business level (key performance indicators)
- No systematic instrumentation on application level
- Monitoring as an “afterthought”: probes are only added when problems occurred.

# Application-Level Monitoring in Practice . . . !?

## — Among Java Professionals —



Christian-Albrechts-Universität zu Kiel




"Java monitoring largely unknown."

[codecentric GmbH 2009]

## Scaling Facebook to 500 Million Users and Beyond

"Making lots of small changes and watching what happens only works if you're actually able to watch what happens. At Facebook we **collect an enormous amount of data** — any particular server exports **tens or hundreds of metrics** that can be graphed. This isn't just system level things like CPU and memory, it's also **application level statistics** to understand why things are happening.

It's important that the statistics are from the **real production machines** that are having the problems, when they're having the problems – **the really interesting things only show up in production**. The stats also have to come from all machines, because a lot of important effects are hidden by averages and only show up in distributions, in particular 95th or 99th percentile."

*Robert Johnson, Facebook Engineering Director*

Christian-Albrechts-Universität zu Kiel

# Kieker

- Modular, flexible, and extensible architecture (Probes, records, readers, writers, filters etc.)
- Pipes-and-filters framework for analysis configuration
- Distributed tracing (logging, reconstruction, visualization)
- Low overhead (designed for continuous operation)
- Evaluated in lab and industrial case studies

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

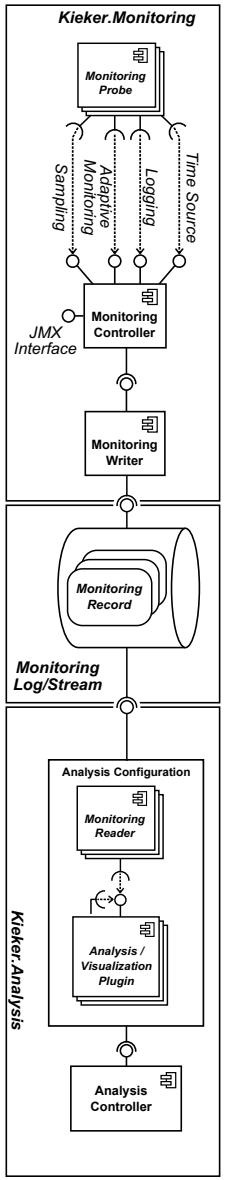
## Kieker is open-source software (Apache License, V. 2.0)

`http://kieker-monitoring.net`

## Recommended Tool of the 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



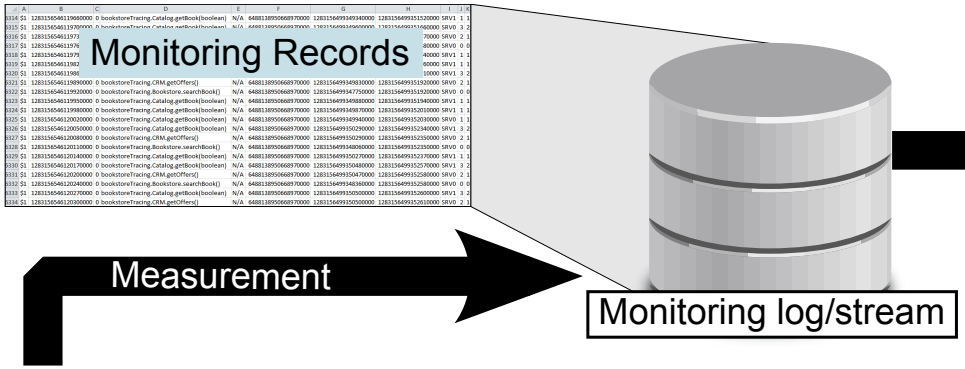

| | | | | | |
|---|---|---|---|---|---|
| Monitoring Probes/Samplers | Control-flow tracing | Manual | AspectJ | Servlet | Spring |
| | | CXF/SOAP | VB6, COBOL, ... | | |
| | | *<custom interception technology>* | | | |
| | Resource monitoring | Servlet | Sigar | CPU utilization | |
| | | | | Memory usage | |
| | | *<custom technology>* | | | |
| | *<custom monitoring probe>* | | | | |
| Time Sources | System time | | | | |
| | *<custom time source>* | | | | |
| Monitoring Writers | File system | Database (SQL) | TCP/IP | | |
| | Java Messaging Service (JMS) | | | | |
| | Java Management Ext. (JMX) | | | | |
| | Named pipe | Output stream | | | |
| | *<custom monitoring writer>* | | | | |
| Monitoring Records | Operation execution | | | | |
| | Control-flow events | | | | |
| | CPU utilization | Memory/swap usage | | | |
| | Resource utilization | | | | |
| | Current time | | | | |
| | *<custom monitoring record type>* | | | | |
| Monitoring Readers | File system | | | | |
| | Java Messaging Service (JMS) | | | | |
| | Java Management Ext. (JMX) | | | | |
| | Database (SQL) | TCP/IP | | | |
| | Named pipe | Real-time replayer | | | |
| | *<custom monitoring reader>* | | | | |
| Analysis/Visualization Plug-ins | Pipes and filters framework | Performance Analysis | | | |
| | | Architecture reconstr. | Trace analysis | Visualization | Dependency graphs |
| | | | | | Sequence diagrams |
| | | | | | Call graphs |
| | | | | | *<custom visualization>* |
| | | | | *<custom trace analysis>* | |
| | | | *<custom reconstruction plug-in>* | | |
| | | *<custom analysis plug-in/tool>* | | | |

# Dynamic Analysis with Kieker

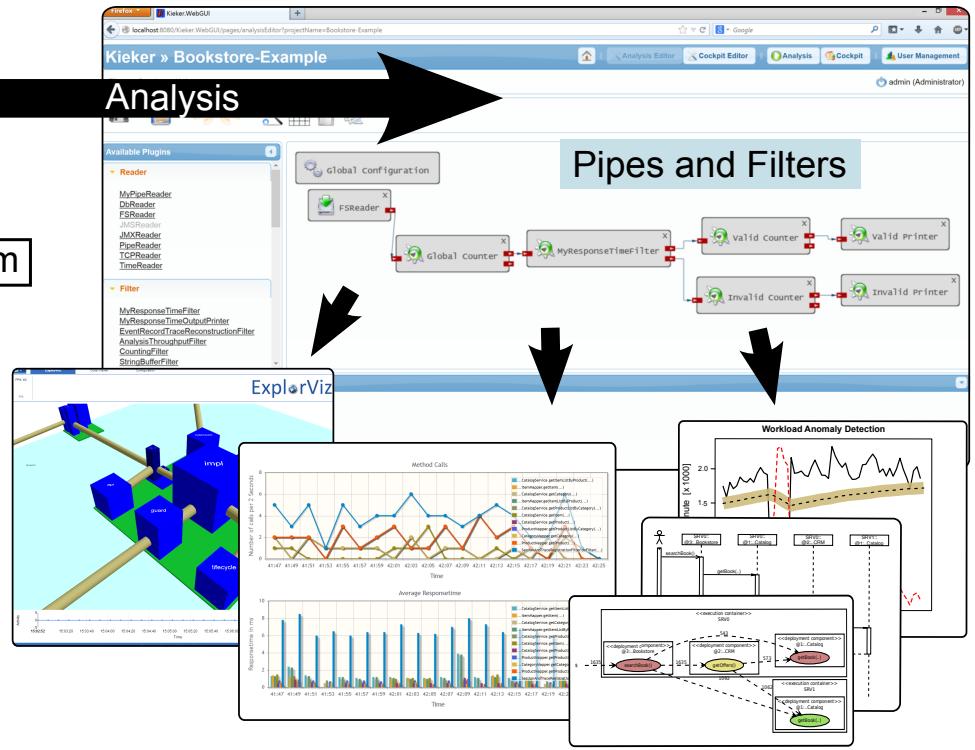
[van Hoorn et al. 2012]

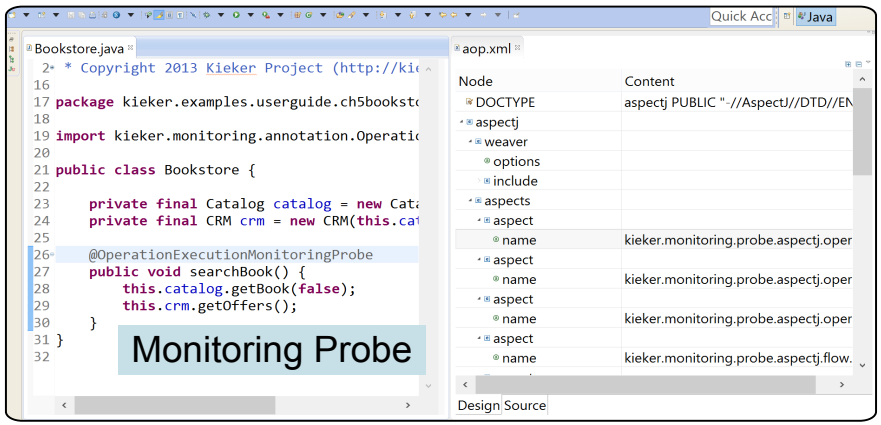
Overview

Christian-Albrechts-Universität zu Kiel




Analysis Configuration (via API and WebGUI)



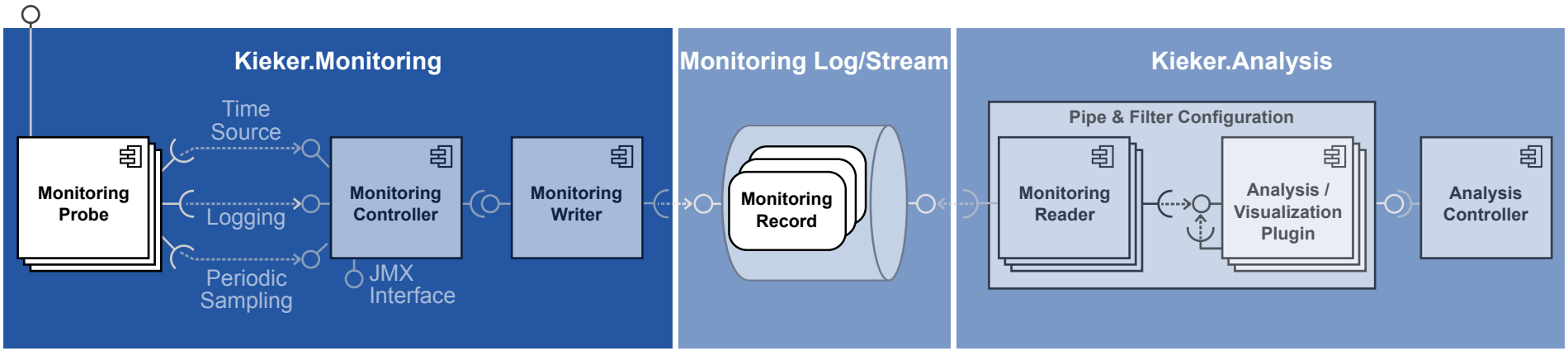




Software System with Monitoring Instrumentation

Online and Offline Visualization

# Core Framework Components

## Overview




### Java probes/samplers:

**Monitoring Probes/Samplers**

- Control-flow tracing
  - Manual instrumentation
  - AspectJ | Spring
  - Servlet | CXF/SOAP
  - *<your interception technology>*
- Resource monitoring
  - Servlet | Sigar
    - CPU utilization
    - Memory usage
  - *<your technology>*
- *<your monitoring probe>*

### + basic adapters for

- C#/.NET
- Visual Basic 6/COM
- COBOL

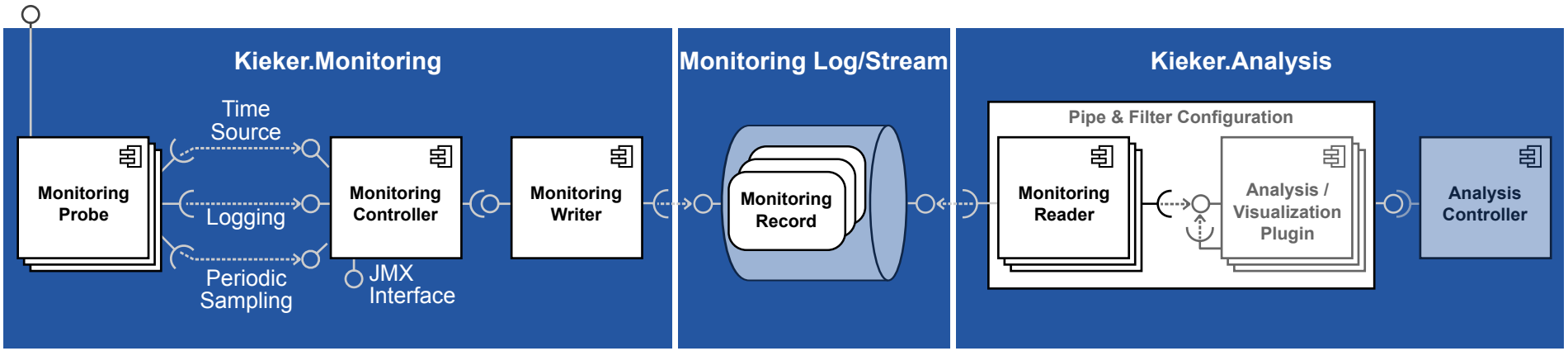
**Monitoring Records**

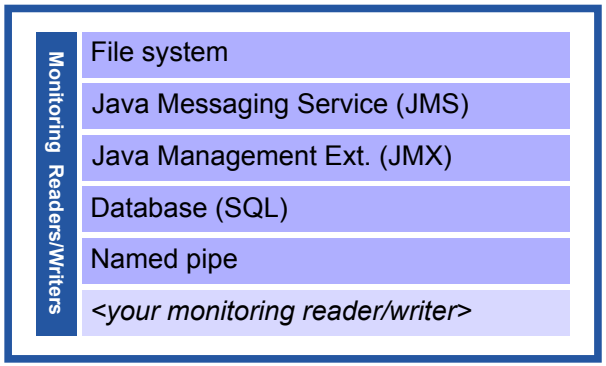
- Operation execution
- Control-flow events
- CPU utilization | Memory/swap usage
- Resource utilization
- Current time
- *<your monitoring record type>*

# Core Framework Components



Christian-Albrechts-Universität zu Kiel

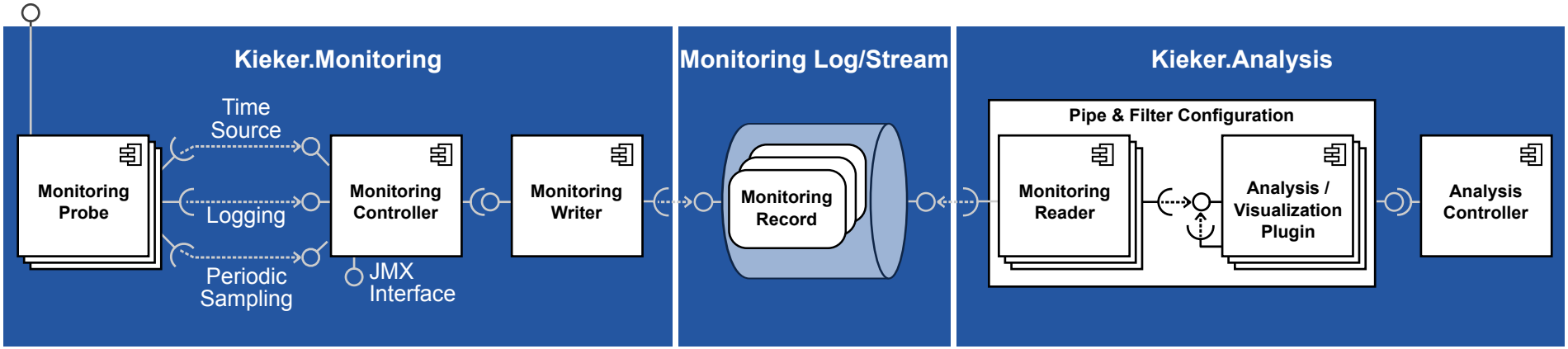


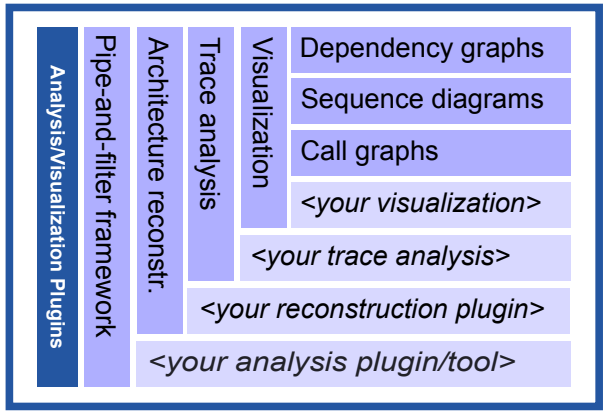



# Core Framework Components



Christian-Albrechts-Universität zu Kiel

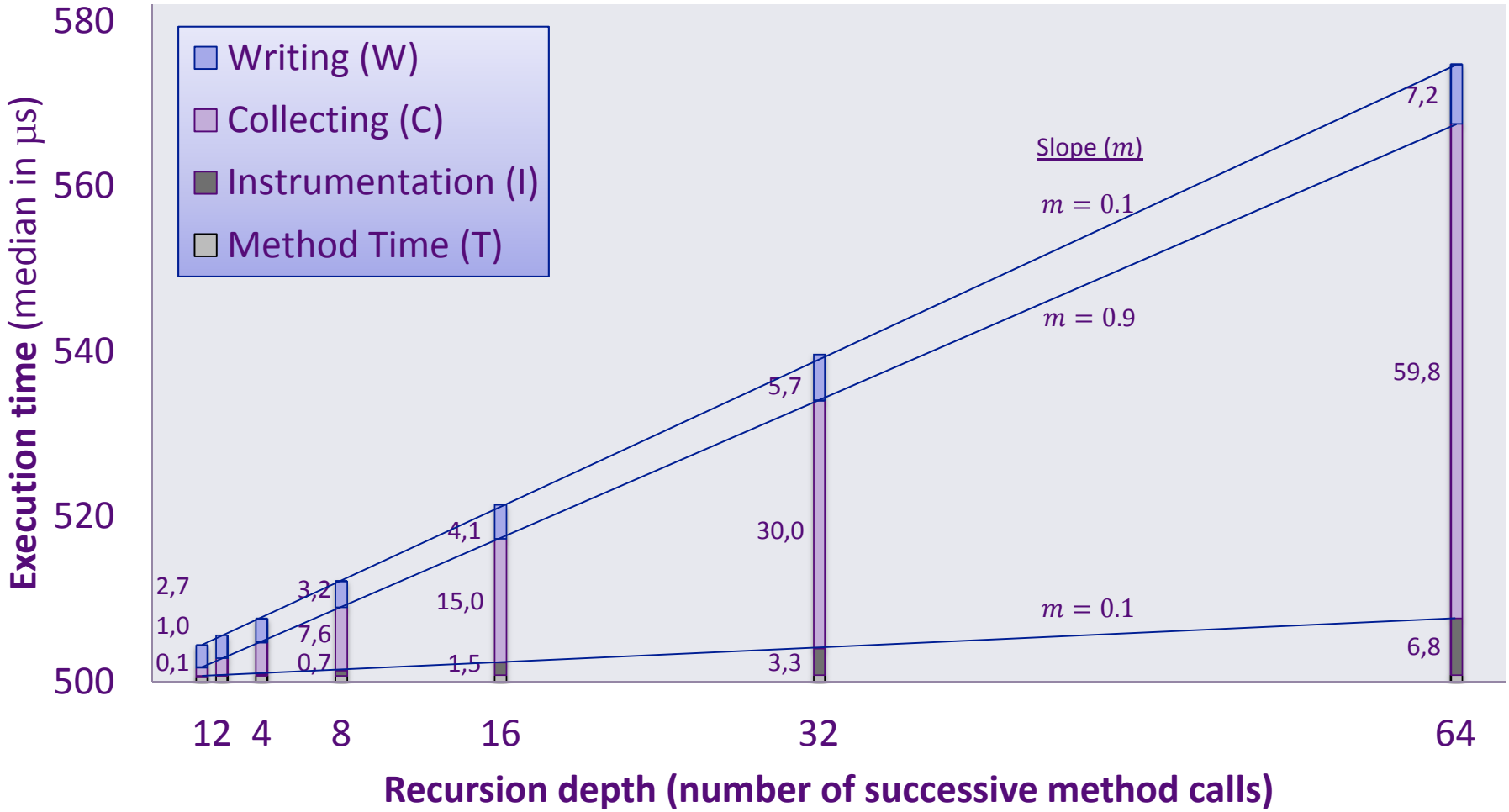


# Overhead Evaluation

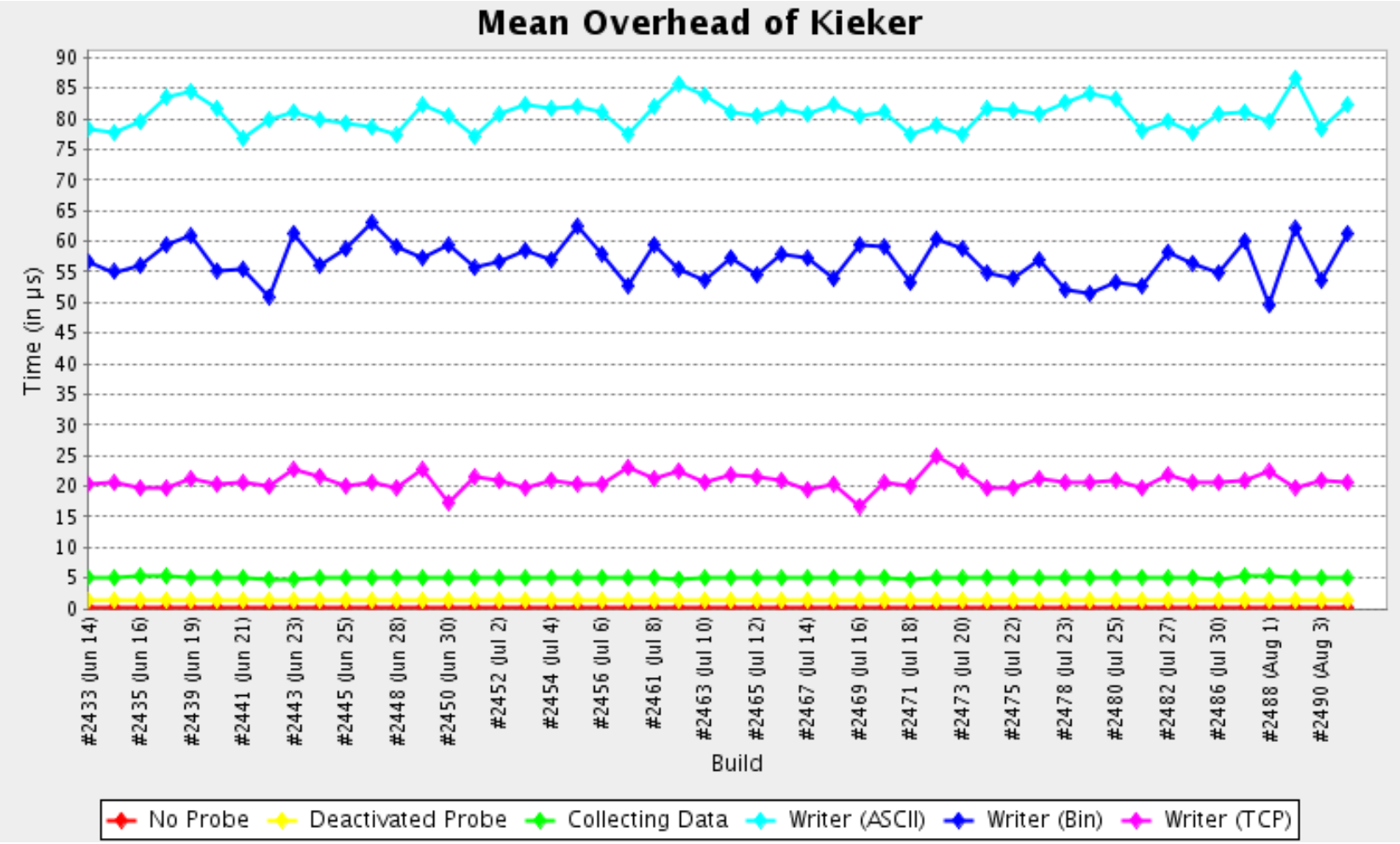
[Waller and Hasselbring 2012]






**Experiment similar to:**

| | | | |
|---|---|---|---|
| A5 | AsyncFS writer | 16 cores | whole system is available |




https://build.se.informatik.uni-kiel.de/jenkins/job/kieker-nightly-release/plot/

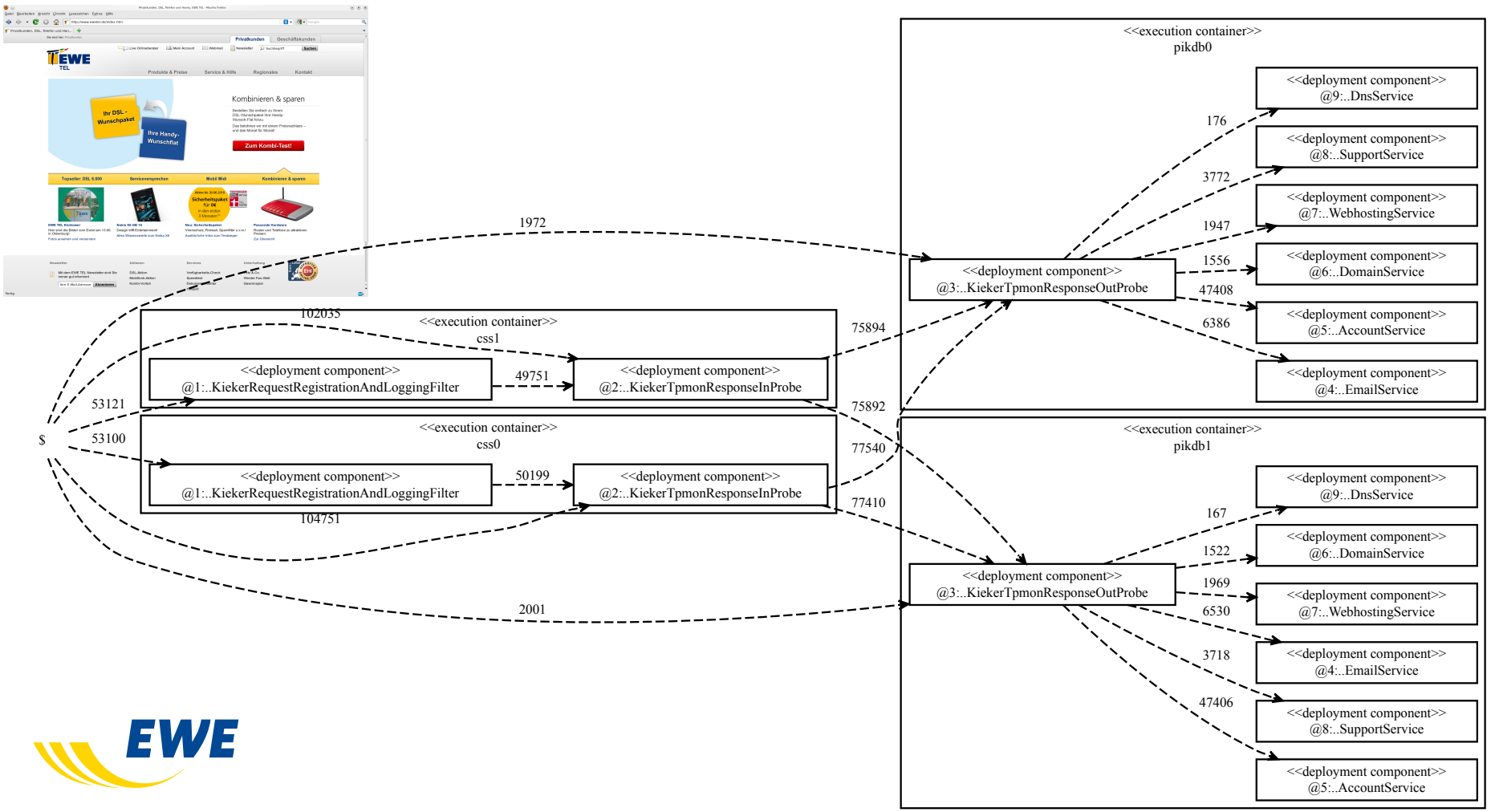
# Kieker Context: Application Examples



Christian-Albrechts-Universität zu Kiel

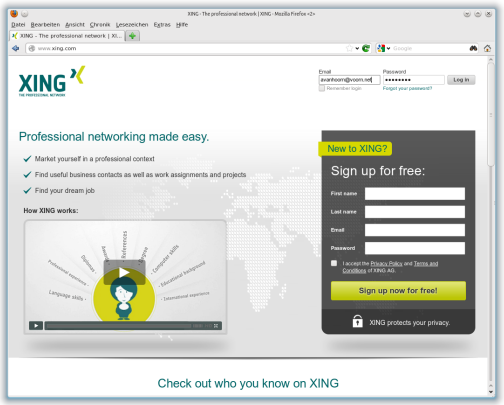
1. Architecture Discovery: Model Extraction and Visualization
2. Applicaton Performance Management: Anomaly Detection + Diagnosis

# Architecture Discovery: Model Extraction + Visualization (cont'd)

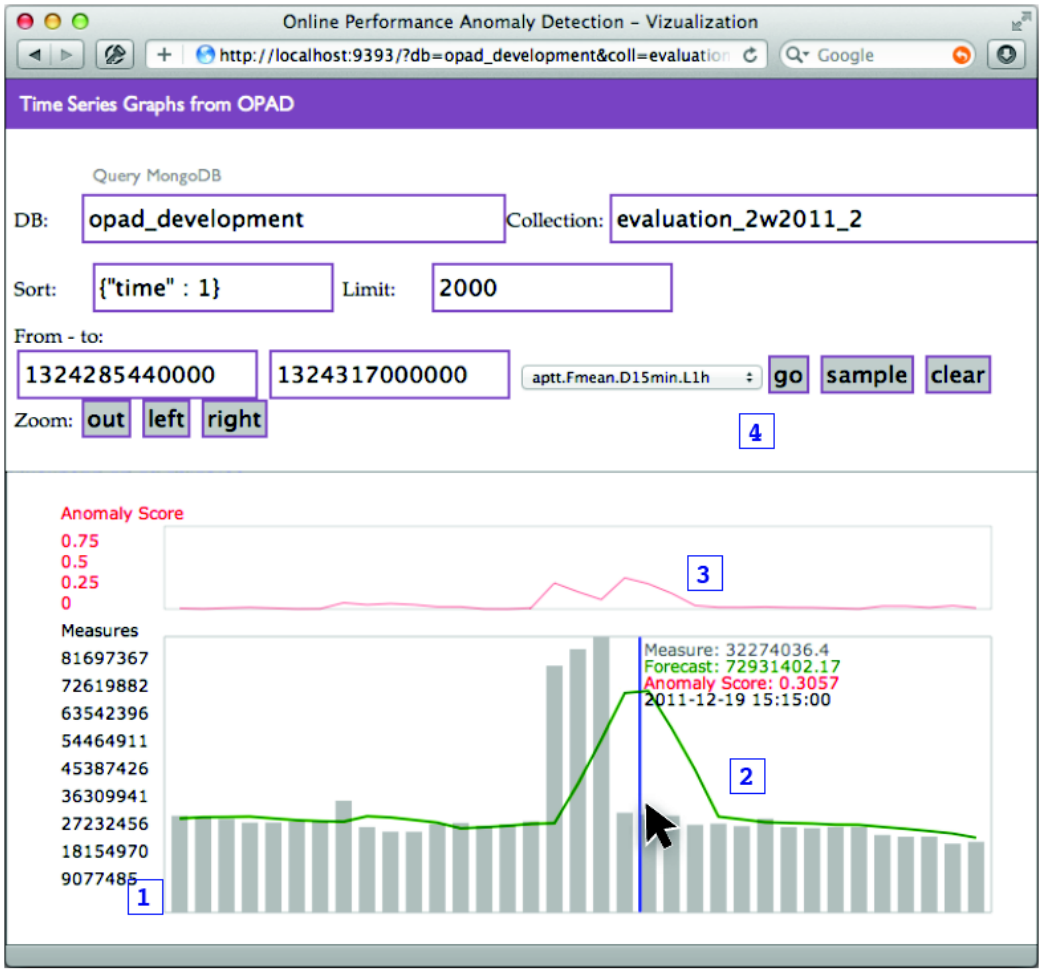



[van Hoorn et al. 2009]

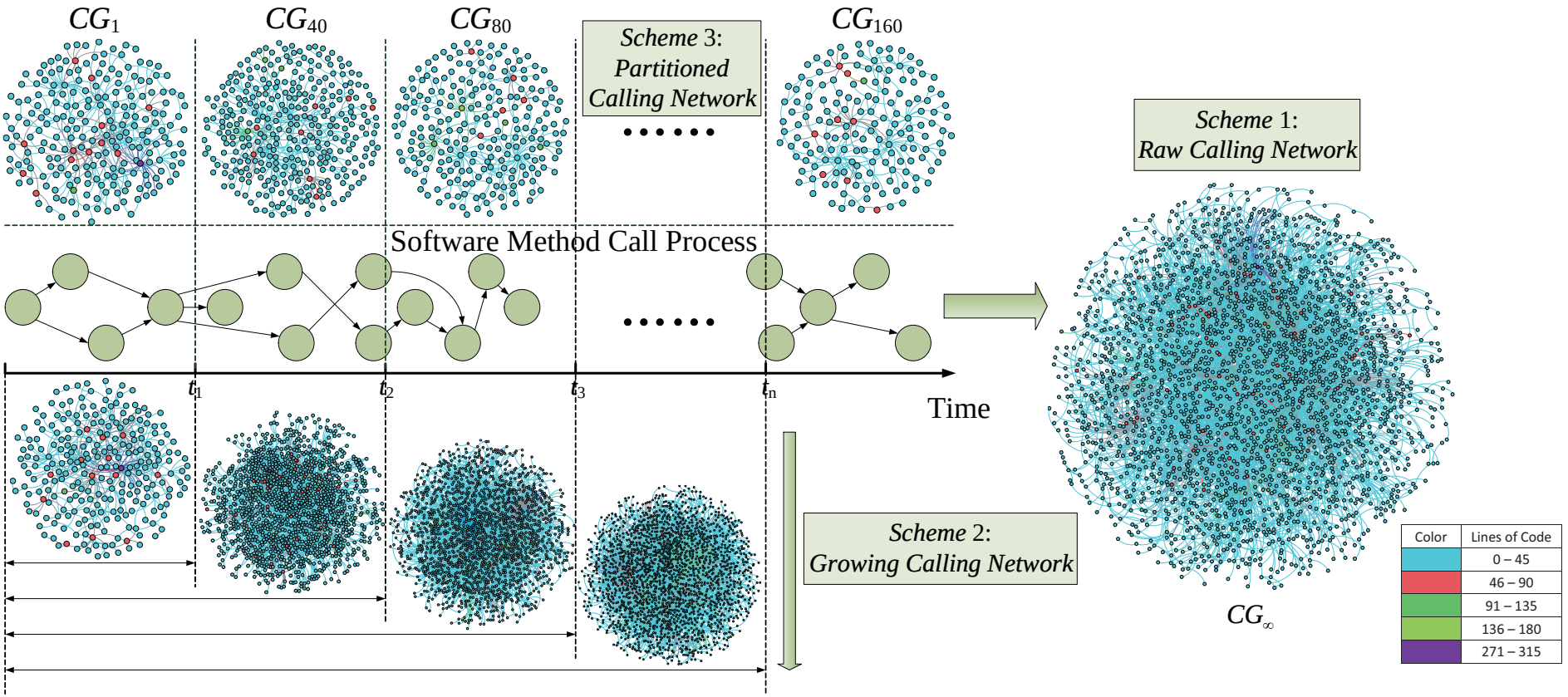
# APM: Anomaly Detection + Diagnosis (cont'd)




XING




[Bielefeld 2012, Frotscher 2013]



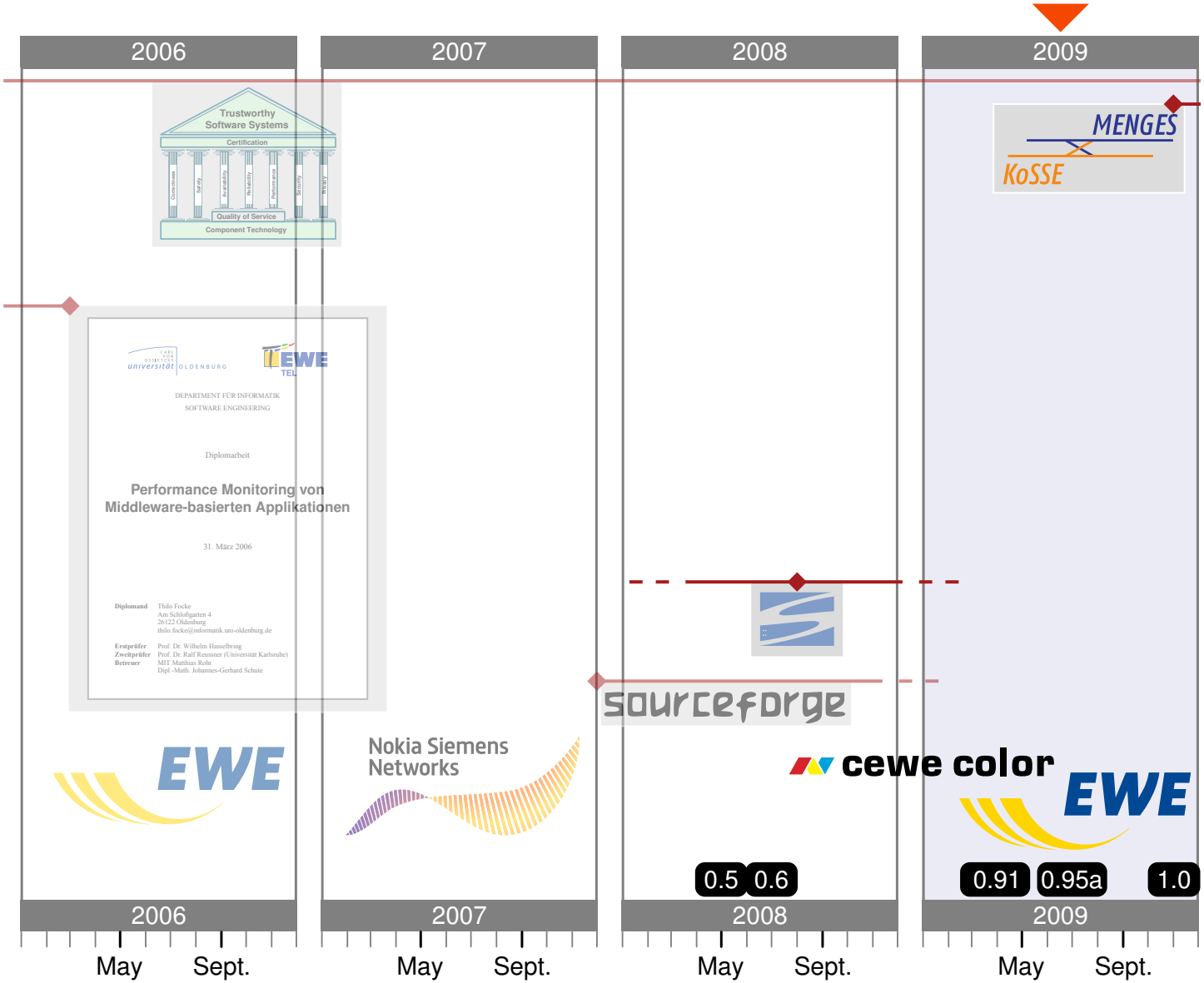

Xi'an Jiaotong University, Shaanxi
[Qu et al. 2015; 2014; 2013, Zheng et al. 2011]

# Agenda

# Looking back . . . 2006–2009



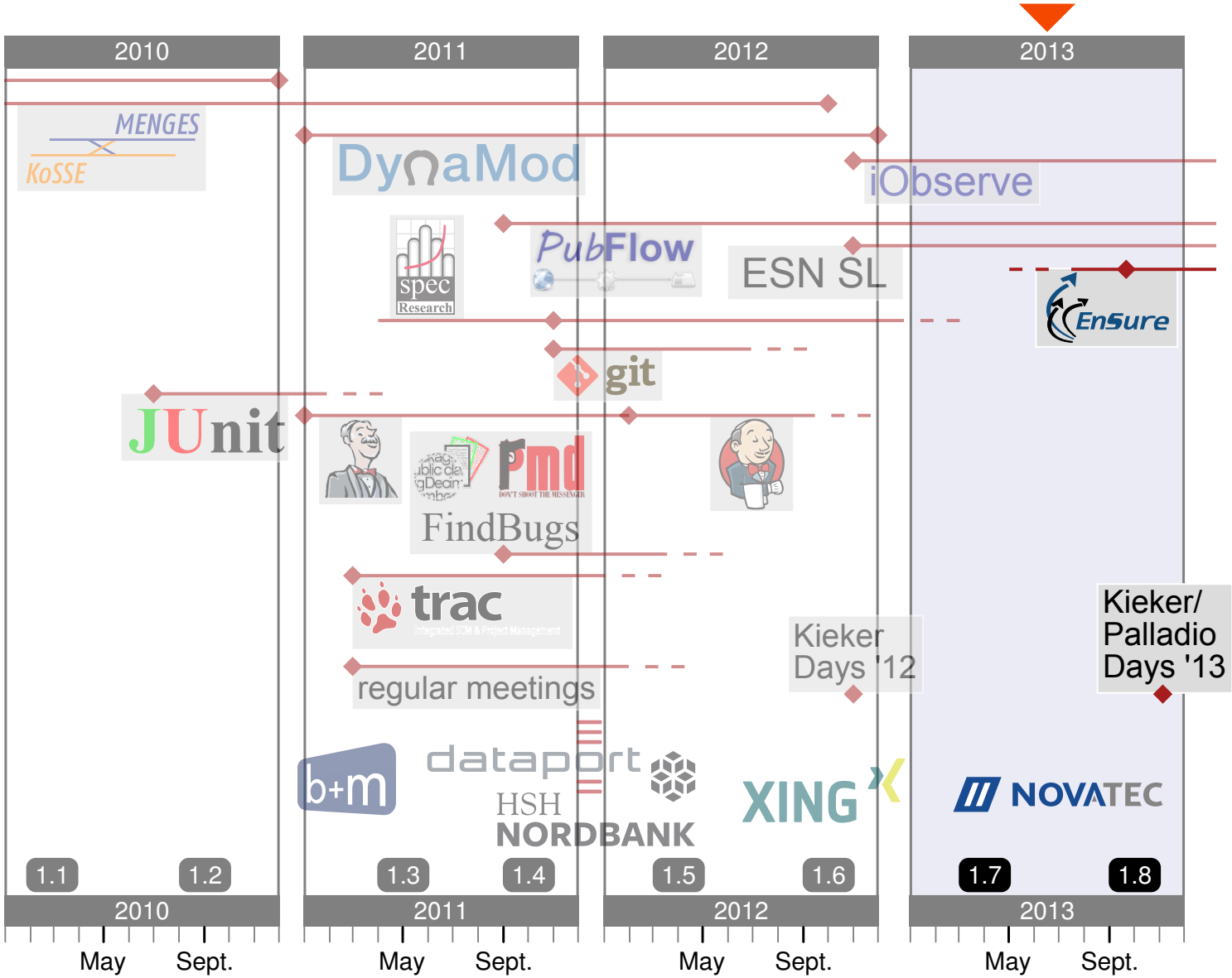




# Looking back . . . 2010–2013



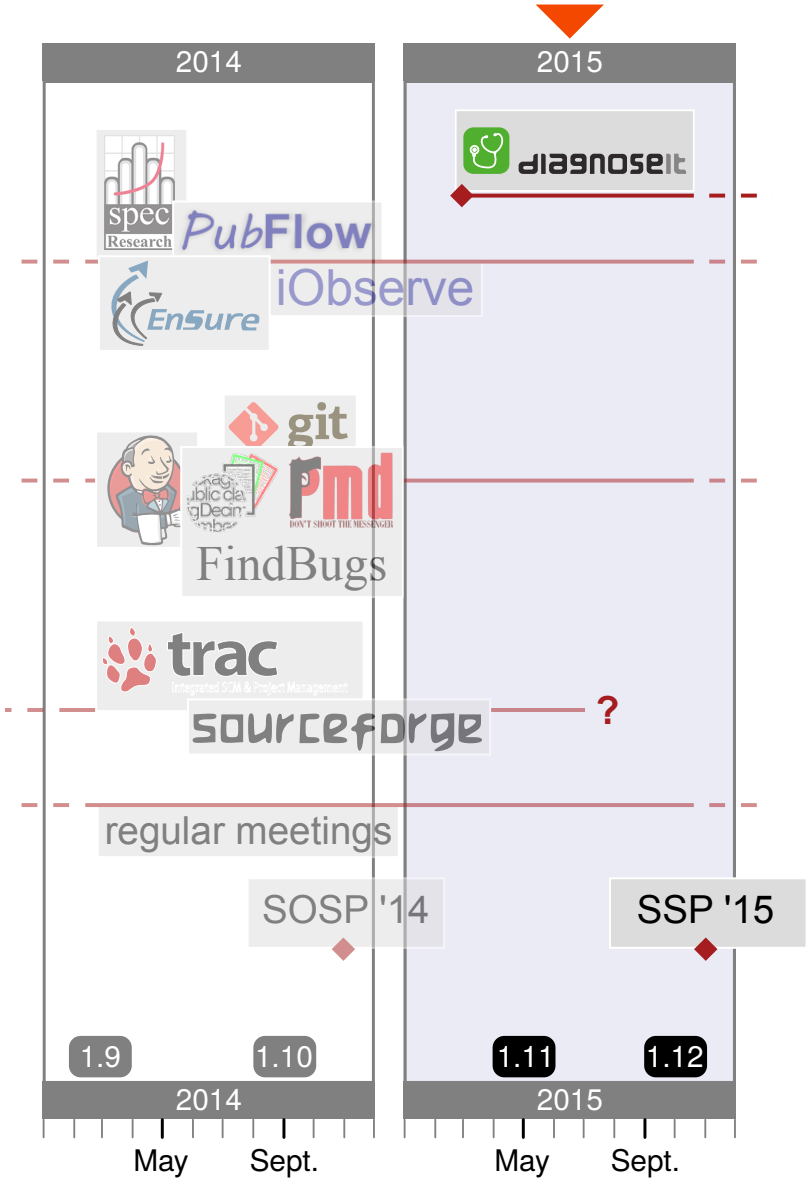
Christian-Albrechts-Universität zu Kiel



# Looking back . . . 2014–2015



Christian-Albrechts-Universität zu Kiel



# References: Research and Industry

http://kieker-monitoring.net/research/references/

Review

Christian-Albrechts-Universität zu Kiel

**Internal Researchers**

Kieker is currently maintained jointly by the following research groups from Kiel University and the University of Stuttgart as part of their teaching and research activities, including collaborators from other academic or industrial institutions.

Kiel University, Kiel, Germany – Researchers from the Kiel University's Software Engineering Group investigate innovative techniques and methods for engineering, evolving, and operating continuously running software systems (research projects).

University of Stuttgart, Stuttgart, Germany – Researchers from the University of Stuttgart's Reliable Software Systems Group investigate innovative quantitative QoS analysis and forecasting methods for distributed software-intensive systems (research projects).

Feel free to contact us if you are interested in any aspect of the Kieker framework.

**External Researchers**

Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany – Researchers from the KIT's Software Design and Quality Group are using Kieker for different purposes, e.g., for detecting and diagnosing performance problems in systematic experiments. We are also collaborating with KIT researchers in the context of the iObserve research project.

RWTH Aachen University, Aachen, Germany – Researchers from the RTWH Aachen University's Software Construction Group are using Kieker for monitoring-based architecture reconstruction in their ARAMIS project on model-based software architecture evolution and analysis.

University of Novi Sad, Novi Sad, Serbia – Researchers from the University of Novi Sad were using Kieker for adaptive monitoring of software systems in the context of performance problem detection and diagnosis.

University of Würzburg, Würzburg, Germany – Researchers from the University of Würzburg's Software Engineering Group are using Kieker for extracting performance models.

Warsaw University, Warsaw, Poland – Researchers from the Warsaw University employed Kieker for dynamic data acquisition of software architectures.

Xi'an Jiatong University, Xi'an, Shaanxi, China – Researchers from the Xi'an Jiatong University used Kieker for discovering architectural structures in software systems and to analyze software call graphs.

**Industry**

b+m Informatik AG, Melsdorf, Germany – With b+m, we collaborated in the context of the DynaMod and MENGES research projects. Moreover, Kieker is being used by b+m, e.g., for architecture discovery of large-scale COBOL mainframe systems. Contributions by b+m are part of the Kieker release.

CEWE COLOR AG & Co. OHG, Oldenburg, Germany – With CEWE COLOR, we collaborated in the context of the TrustSoft research project. CEWE COLOR provided a JavaEE-based web portal as a case study system for application performance monitoring. Contributions by CEWE COLOR are part of the Kieker release.

Dataport AöR, Altenholz, Germany – With Dataport, we collaborated in the context of the DynaMod research project. Dataport provided a VB6-based case study system for architecture discovery based on hybrid analysis with Kieker.

EPrints Services, Southampton, United Kingdom – With EPrints Services, we collaborated in the context of several thesis projects. We employ the EPrints system as a case study system for software performance analysis with Kieker for Perl-based systems. The Eprints team provides an integration of Kieker with EPrints as epkieker.

EWE TEL GmbH, Oldenburg, Germany – With EWE TEL, we collaborated in the context of the TrustSoft research project. EWE TEL provided a JavaEE-based web portal as a case study system for application performance monitoring. Contributions by EWE TEL are part of the Kieker release.

HSH Nordbank AG, Kiel, Germany – With HSH Nordbank, we collaborated in the context of the DynaMod research project. The HSH provided a C#-based function library for architecture discovery based on hybrid analysis with Kieker.

NovaTec GmbH, Leinfelden-Echterdingen, Germany – With NovaTec, we currently collaborate in the context of different teaching projects on application performance management. NovaTec published a nice blog article about their first experiences with Kieker.

SAP Research, Karlsruhe, Germany – Kieker is used as a tool to collect performance data for the Software Performance Cockpit. We are also collaborating with SAP Research in the context of the iObserve research project.

XING AG, Hamburg, Germany – With XING, we collaborated in the context of a Diploma thesis on online performance anomaly detection (OPAD). XING provided its core system xing.com for evaluating the Kieker-based OPAD implementation.

# Downloads and Citations

http://sourceforge.net/, https://scholar.google.de/

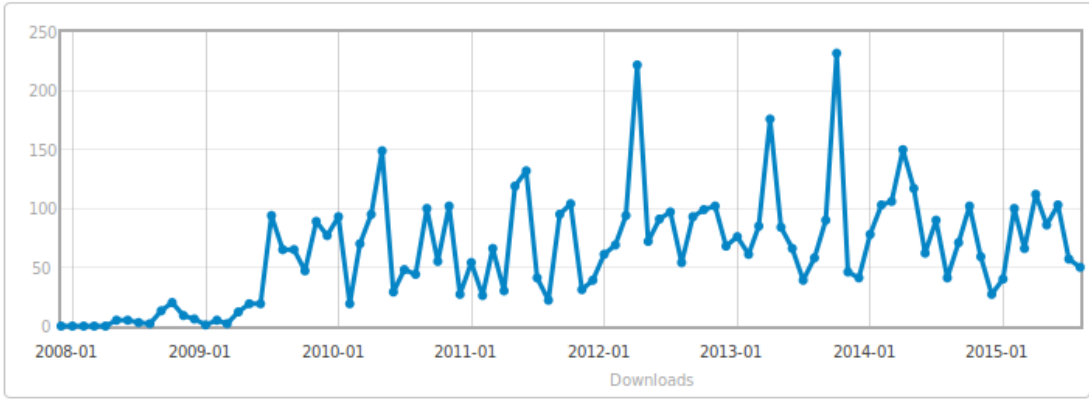
Review



Brought to you by: avanhoorn, janwaller, mat-rohr

Home (Change File)

Date Range: 2007-12-17 to 2015-08-31




**DOWNLOADS**

**5,944**

In the selected date range

**TOP COUNTRY ***

**Germany**

47% of downloaders

**TOP OS ***

**Windows**

60% of downloaders

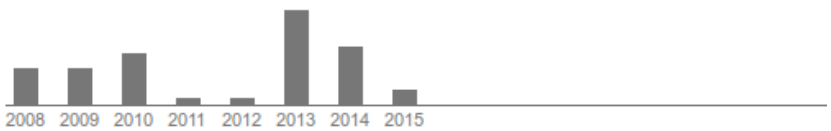
Cited by 115


Kieker: A framework for application performance monitoring and dynamic software analysis
A Van Hoorn, J Waller, W Hasselbring - Proceedings of the 3rd ACM/SPEC International ..., 2012
Cited by 115 - Related articles - All 4 versions

Cited by 70


Continuous monitoring of software services: Design and application of the Kieker framework
A van Hoorn, M Rohr, W Hasselbring, J Waller, J Ehlers... - 2009
Cited by 70 - Related articles - All 5 versions

Cited by 42



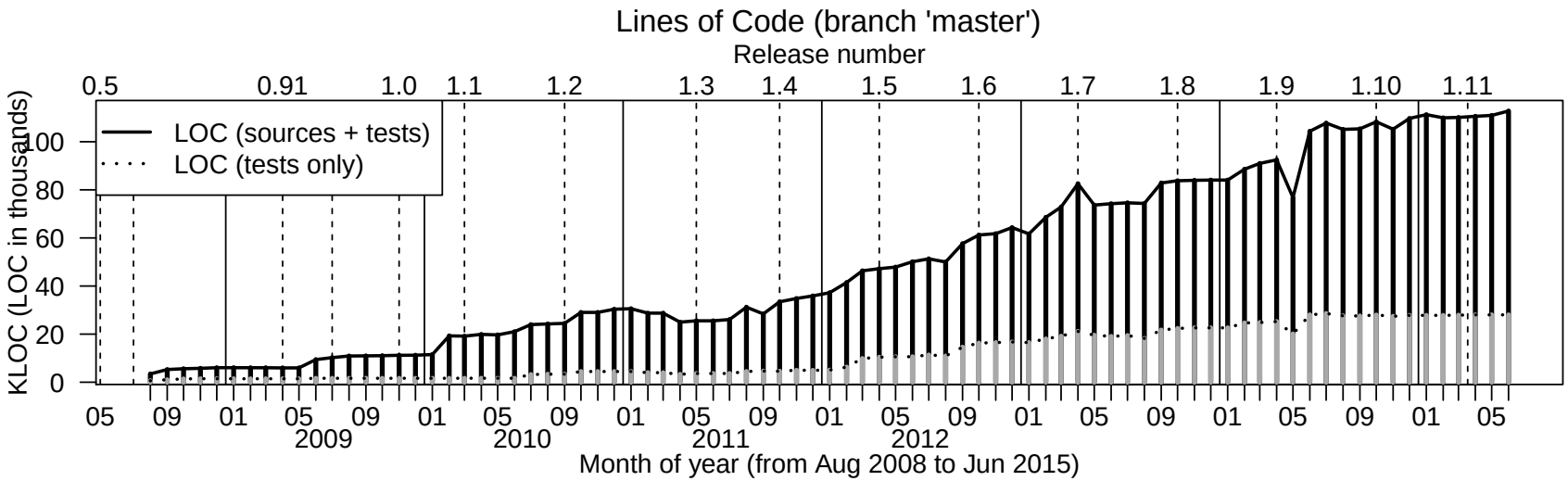

Kieker: Continuous monitoring and on demand visualization of Java software behavior
M Rohr, A van Hoorn, J Matevska, N Sommer... - 2008
Cited by 42 - Related articles - All 8 versions

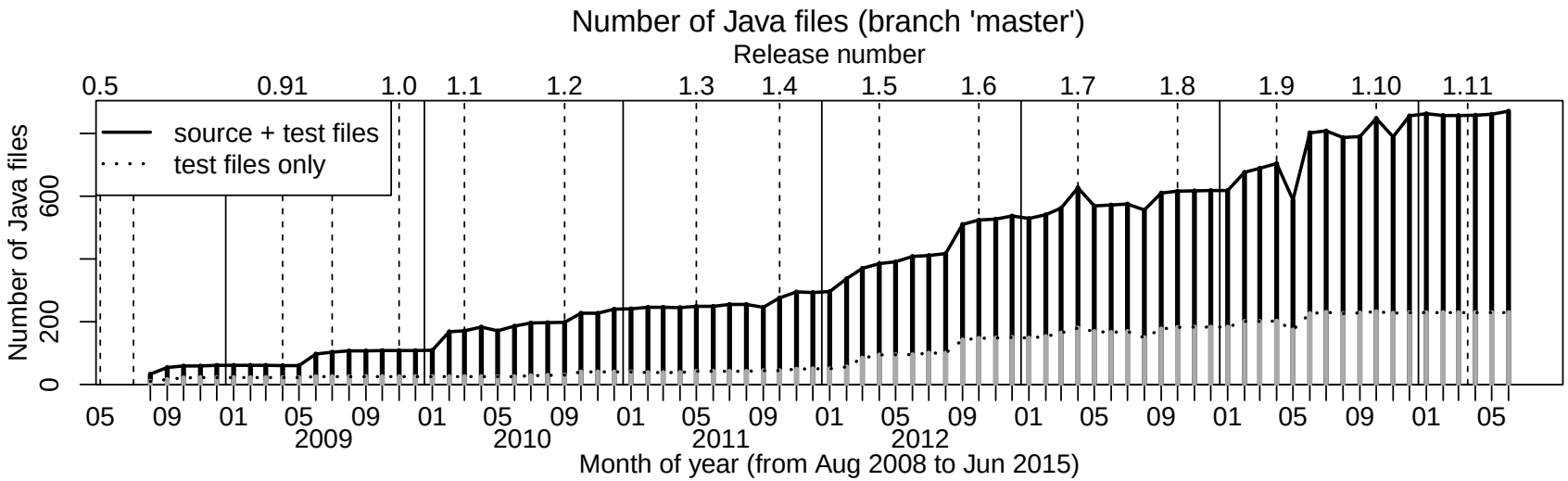
# Evolution of Kieker's Code Size ('master')

## LOC obtained via `wc -l <file>.java`

Review

Christian-Albrechts-Universität zu Kiel



# Development Phases

Phase 1: 2006 (Inception)

Phase 2: 2007–2009 (Production Systems)

Phase 3: 2009–2010 (Restructuring)

Phase 4: 2011–2012 (Quality Assurance, SPEC Review)

Phase 5: 2013–today (Distributed Development and Community Building)

# Agenda

# Lessons Learned

- Open source tool can increase visibility in academia and industry
- Funding for research projects is essential
  - Incubator for technology transfer.
- Besides projects, we also provide professional coaching and training for the software
- Kieker is also used as example and object for software engineering teaching
- Success factors
  - A crucial success factor for establishing Kieker was the early deployment in production systems (Phase 2)
  - Another boost came from the rigorous review process by the SPEC Research Group (Phase 4)
- Licensing is also relevant
  - Kieker is licensed under the Apache License, Version 2.0
  - Impact !

## Current Activities (Selection)

- Kieker Trace Diagnosis
  - Trace diagnosis tool to identify typical performance problems
- High-throughput infrastructure for Kieker Analysis
  - Based on TeeTime [Wulf et al. 2014], `http://teetime.sf.net`
- Interoperability between APM tools (open exchange formats)
- Docker-based Kieker example (NetflixOSS RSS reader application)
- Analysis of Kieker Development process and infrastructure
- Split Kieker into multiple, independent components [Hasselbring 2002]
  - under discussion ...

## Ticket System: Current/Upcoming Issues

`http://trac.kieker-monitoring.net`

# Thanks to All Contributors






Various additional people contributed to Kieker in the past years:

***Jan Beye, Tillmann (Till) Bielefeld, Peer Brauer, Thomas Düllmann, Philipp Döhring, Jens Ehlers, Nils Ehmke, Florian Fittkau, Albert Flaig, Thilo Focke, Sören Frey, Tom Frotscher, Henry Grow, Reiner Jung, Benjamin Kiel, Dennis Kieselhorst, Holger Knoche, Arnd Lange, Marius Löwe, Marco Lübcke, Felix Magedanz, Sören Mahmens, Nina Marwede, Robert von Massow, Jasminka Matevska, Teerat Pitakrat, Oliver Preikszas, Sönke Reimer, Bettual Richter, Matthias Rohr, Nils Sommer, Lena Stöver, Jan Waller, Nis Wechselberg, Robin Weiß, Björn Weißenfels, Matthias Westphal, Christian Wulf, Christian Zirkelbach***

—Alphabetic list of people who contributed in different form (source code, bug reports, promotion, etc) and intensity

## Accompanying Paper for this Talk

W. Hasselbring and A. van Hoorn. *Open-Source Software as Catalyzer for Technology Transfer: Kieker's Development and Lessons Learned.*

Technical Report TR-1508, Dept. of Computer Science, Kiel University, Kiel, Germany. Aug. 2015.

`http://eprints.uni-kiel.de/29463/`

**Open-Source Software as Catalyzer for Technology Transfer: Kieker's Development and Lessons Learned**

Wilhelm Hasselbring[1] and André van Hoorn[2]

[1] Kiel University, Department of Computer Science, 24118 Kiel, Germany
[2] University of Stuttgart, Institute of Software Technology, 70569 Stuttgart, Germany

**Abstract:** The monitoring framework Kieker commenced as a joint diploma thesis of the University of Oldenburg and a telecommunication provider in 2006, and grew toward a high-quality open-source project during the last years. Meanwhile, Kieker has been and is employed in various projects. Several research groups constitute the open-source community to advance the Kieker framework. In this paper, we review Kieker's history, development, and impact as catalyzer for technology transfer.

**1 Introduction**

The development of tools is common practice for researchers in order to demonstrate the practicality of developed research approaches and to qualitatively and quantitatively evaluate their research results. During the last years, there is an increasing trend that researchers make their tools publicly available under an open-source license, e.g., allowing a more thorough evaluation of work presented in research papers, as well as easing reproducibility of results and building on the work of others. The state of these tools ranges from proof-of-concept implementations to full-blown products. Popular examples of wide-spread and mature open-source tools originally developed and maintained by researchers include the probabilistic model checker PRISM [KNP11] and the R language and environment for statistical computing [R D08].

Since 2006, we have been developing the Kieker framework for dynamic analysis of software systems.[1] In this paper, we review Kieker's history, development, and impact as catalyzer for technology transfer. Parts of this paper have been published in a PhD dissertation [vH14, Chapter 15], which also includes a more detailed description of the framework (in addition to [RvHM+08, vHRH+09, vHWH12]) as well as its development process and infrastructure.

**2 Kieker's Development and Impact**

This section reviews the past years of Kieker development and gives some indication of the impact in terms of where and by whom Kieker has been developed and used.

[1] The Kieker framework's web site—including downloads, documentation, publications, and references—is available at `http://kieker-monitoring.net`

# Visit *http://kieker-monitoring.net*



Christian-Albrechts-Universität zu Kiel


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`

http://kieker-monitoring.net/blog/
kiekers-development-in-five-minutes-1-10/

**For a comprehensive list of publications, talks, and theses about Kieker, visit:**
`http://kieker-monitoring.net/research/`

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

codecentric GmbH. Performance survey 2008. `http://www.codecentric.de/export/sites/www/_resources/pdf/performance-survey-2008-web.pdf`, Mar. 2009.

P. Döhring. Visualisierung von Synchronisationspunkten in Kombination mit der Statik und Dynamik eines Softwaresystems. Master's thesis, Kiel University, Oct. 2012.

F. Fittkau, J. Waller, C. Wulf, and W. Hasselbring. Live trace visualization for comprehending large software landscapes: The ExplorViz approach. In *1st IEEE International Working Conference on Software Visualization (VISSOFT 2013)*, Sept. 2013.

F. Fittkau, S. Roth, and W. Hasselbring. ExplorViz: Visual runtime behavior analysis of enterprise application landscapes. In *23rd European Conference on Information Systems (ECIS 2015 Completed Research Papers)*. AIS Electronic Library, May 2015.

T. Focke. Performance Monitoring von Middleware-basierten Applikationen. Diplomarbeit, University Oldenburg, Mar. 2006.

T. Frotscher. Architecture-based multivariate anomaly detection for software systems. Master's thesis, CAU, AG Software Engineering, Oct. 2013.

W. Hasselbring. Component-based software engineering. In *Handbook of Software Engineering and Knowledge Engineering*, pages 289–305. World Scientific Publishing, 2002.

F. Magedanz. Dynamic analysis of .NET applications for architecture-based model extraction and test generation, Oct. 2011. Diploma Thesis, Kiel University.

N. S. Marwede, M. Rohr, A. van Hoorn, and W. Hasselbring. Automatic failure diagnosis support in distributed large-scale software systems based on timing behavior anomaly correlation. In *Proceedings of the 13th European Conference on Software Maintenance and Reengineering (CSMR 2009)*, pages 47–57. IEEE Computer Society, Mar. 2009.

Y. Qu, X. Guan, Q. Zheng, T. Liu, J. Zhou, and J. Li. Calling network: A new method for modeling software runtime behaviors. In *Proceedings of the Second International Workshop on Software Mining, Palo Alto, CA, USA*, 2013.

Y. Qu, Q. Zheng, T. Liu, J. Li, and X. Guan. In-depth measurement and analysis on densification power law of software execution. In *Proceedings of the 5th International Workshop on Emerging Trends in Software Metrics*, pages 55–58. ACM, 2014.

Y. Qu, X. Guan, Q. Zheng, T. Liu, J. Zhou, and J. Li. Calling network: A new method for modeling software runtime behaviors. *SIGSOFT Softw. Eng. Notes*, 40(1):1–8, Jan. 2015. doi: 10.1145/2693208.2693223.

B. Richter. Dynamische Analyse von COBOL-Systemarchitekturen zum modellbasierten Testen ("Dynamic analysis of cobol system architectures for model-based testing", in German), Aug. 2012. Diploma Thesis, Kiel University.

M. Rohr, A. van Hoorn, J. Matevska, N. Sommer, L. Stoever, S. Giesecke, and W. Hasselbring. Kieker: Continuous monitoring and on demand visualization of Java software behavior. In C. Pahl, editor, *Proceedings of the IASTED International Conference on Software Engineering 2008 (SE'08)*, pages 80–85, Feb. 2008.

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.

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.

J. Waller and W. Hasselbring. A comparison of the influence of different multi-core processors on the runtime overhead for application-level monitoring. In V. Pankratius and M. Philippsen, editors, *Proceedings of the International Conference on Multicore Software Engineering, Performance, and Tools (MSEPT 2012)*, volume 7303 of *Lecture Notes in Computer Science*, pages 42–53. Springer Berlin / Heidelberg, June 2012. ISBN 978-3-642-31201-4. doi: 10.1007/978-3-642-31202-1_5. URL http://eprints.uni-kiel.de/15425/.

J. Waller, N. C. Ehmke, and W. Hasselbring. Including performance benchmarks into continuous integration to enable DevOps. *SIGSOFT Softw. Eng. Notes*, 40(2):1–4, Mar. 2015. doi: 10.1145/2735399.2735416.

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.

C. Wulf, N. C. Ehmke, and W. Hasselbring. Toward a generic and concurrency-aware pipes & filters framework. In *Symposium on Software Performance 2014: Joint Descartes/Kieker/Palladio Days*, pages 70–82, Nov. 2014. URL http://eprints.uni-kiel.de/27382/.

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.