Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation

D3S Seminar/Department Meeting
Department of Distributed and Dependable Systems
Charles University, Prague

André van Hoorn
Software Engineering Group
University of Kiel, Germany
avh@informatik.uni-kiel.de

June 20, 2012 @ Prague
Motivation & Overall Goal

Introduction

André van Hoorn (Univ. Kiel)
»Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«
June 20, 2012 @ Prague
The searchBook service must respond in $\leq 0.5$ seconds in 95% of all cases.
The searchBook service must respond in \( \leq 0.5 \) seconds in 95\% of all cases.

The Bookstore system must be available 99.9999 \% of all time.

André van Hoorn (Univ. Kiel)  
«Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation»  
June 20, 2012 @ Prague
Motivation & Overall Goal

Introduction

André van Hoorn (Univ. Kiel)
»Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«
June 20, 2012 @ Prague

Varying Workload (intensity + usage profile)
Motivation & Overall Goal

Introduction

Static capacity management
Overprovisioning of resources
Motivation & Overall Goal

Introduction

Goal: Increase resource efficiency while meeting SLAs

SLAstic: Architecture-based online capacity management

André van Hoorn (Univ. Kiel)  »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«  Christian-Albrechts-Universität zu Kiel

June 20, 2012 @ Prague  2 / 18
André van Hoorn (Univ. Kiel) »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation« June 20, 2012 @ Prague 3 / 18
Affiliation

- **University of Kiel, Software Engineering Group**
- Before: University of Oldenburg (DFG RTG TrustSoft)
About Me

Introduction

Affiliation

- University of Kiel, Software Engineering Group
- Before: University of Oldenburg (DFG RTG TrustSoft)

Research Interests

1. **Software performance engineering & self-***
   - {Model,Architecture}-{based,driven} SPE techniques (+ tools)
   - Online performance management (monitoring, analysis)
   - (Architectural) performance models @ runtime

2. **Software architecture**
   - Component- & service-based software systems, MDSD
   - QoS (particularly, performance and resource efficiency)
   - Runtime reconfiguration/adaptation, self-***

Current Projects

1. **SLAStic** — Model-Driven Online Capacity Management for C-B Software Systems
2. **Kieker** — Application Performance Management and Dynamic Software Analysis
About Me

Introduction

Affiliation

- University of Kiel, Software Engineering Group
- Before: University of Oldenburg (DFG RTG TrustSoft)

Research Interests

1. **Software performance engineering & self-**
   - \{Model,Architecture}\{-based,driven\} SPE techniques (+ tools)
   - Online performance management (monitoring, analysis)
   - (Architectural) performance models @ runtime

2. **Software architecture**
   - Component- & service-based software systems, MDSD
   - QoS (particularly, performance and resource efficiency)
   - Runtime reconfiguration/adaptation, self-*

3. **Software re(verse)-engineering**
   - Dynamic and hybrid (legacy) software analysis
   - Extraction of architectural models and usage profiles
   - Architecture-based software modernization

Current Projects

1. **SLAStic** — Model-Driven Online Capacity Management for C-B Software Systems
2. **Kieker** — Application Performance Management and Dynamic Software Analysis
3. **DynaMod** — Dynamic Analysis for Model-Driven Software Modernization
Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation

1. Introduction

2. SLAStic — Architecture-Based Online Capacity Management

3. Kieker — Application Performance Monitoring and Dynamic Analysis

4. Conclusion
SLAStic — Architecture-Based Online Capacity Management

André van Hoorn (Univ. Kiel)  »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«  June 20, 2012 @ Prague

instrumented, runtime reconfigurable s/w system

architecture technology
SLAStic Framework

SLAStic — Architecture-Based Online Capacity Management

- instrumented, runtime reconfigurable s/w system
- raw monitoring records
- monitoring events
- SLAStic model
- Analysis
- online adaptation engine

SLAStic.MONITORING

SLAStic.CONTROL
SLAStic Framework

SLAStic — Architecture-Based Online Capacity Management

André van Hoorn (Univ. Kiel)  »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«  June 20, 2012 @ Prague  6 / 18
SLAStic Framework

SLAStic — Architecture-Based Online Capacity Management

André van Hoorn (Univ. Kiel) »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation« June 20, 2012 @ Prague 6 / 18
Model-Driven Instrumentation

SLAStic — Architecture-Based Online Capacity Management

instrumented, runtime reconfigurable s/w system

SLAStic

SLAStic.MONITORING

SLAStic.RECONFIGURATION

SLAStic.CONTROL

André van Hoorn (Univ. Kiel)  »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«  June 20, 2012 @ Prague
Extraction of Architectural Models (Dyn. Analysis)

SLAStic — Architecture-Based Online Capacity Management

instrumented, runtime reconfigurable s/w system

SLAStic.MONITORING

SLAStic.RECONFIGURATION

SLAStic.CONTROL

online adaptation engine

André van Hoorn (Univ. Kiel)  »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«  June 20, 2012 @ Prague 6 / 18
SLAStic.CONTROL — Architecture-Based Online Capacity Management

André van Hoorn (Univ. Kiel) »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation« June 20, 2012 @ Prague
SLAStic Meta-Model

SLAStic — Architecture-Based Online Capacity Management

System Partition (also used as runtime model)

1. Type repository (e.g., component types, interfaces, connector types, execution container types)
2. Component assembly (e.g., assembly of components via connectors)
3. Execution environment (e.g., execution containers and interconnection via links)
4. Component deployment (mapping: assembly components → containers)
**System Partition** (also used as runtime model)

1. **Type repository** (e.g., component types, interfaces, connector types, execution container types)
2. **Component assembly** (e.g., assembly of components via connectors)
3. **Execution environment** (e.g., execution containers and interconnection via links)
4. **Component deployment** (mapping: assembly components → containers)

---

Example type repository

```
<<Provides>>
<<ComponentType>>
BookstoreT
<<Interface>>
IBookstore
<<Provides>>
Book searchBook()
<<ConnectorType>>
ConnectorTIBook

<<Requires>>
<<Interface>>
ICatalog
<<Provides>>
Book getBook()
<<ConnectorType>>
ConnectorTICatalog

<<Provides>>
<<ComponentType>>
CatalogT
<<ExecutionContainerType>>
WebSrvT

<<Provides>>
<<ComponentType>>
AppSrvT
```
**System Partition** (also used as runtime model)

1. **Type repository** (e.g., component types, interfaces, connector types, execution container types)
2. **Component assembly** (e.g., assembly of components via connectors)
3. **Execution environment** (e.g., execution containers and interconnection via links)
4. **Component deployment** (mapping: assembly components → containers)

Example component assembly
**System Partition** (also used as runtime model)

1. **Type repository** (e.g., component types, interfaces, connector types, execution container types)
2. **Component assembly** (e.g., assembly of components via connectors)
3. **Execution environment** (e.g., execution containers and interconnection via links)
4. **Component deployment** (mapping: assembly components → containers)

---

**Example component deployment**

```
<<ExecutionContainer>>
web1: WebSrvT
<<AssemblyComponent>>
bookstore
<<ComponentType>>
BookstoreT

<<ExecutionContainer>>
web2: WebSrvT
<<AssemblyComponent>>
bookstore
<<ComponentType>>
BookstoreT
```

```
<<ExecutionContainer>>
app1: AppSrvT
<<AssemblyComponent>>
catalog
<<ComponentType>>
CatalogT

<<ExecutionContainer>>
app2: AppSrvT
<<AssemblyComponent>>
catalog
<<ComponentType>>
CatalogT
```
(De-)Replication of Software Components

André van Hoorn (Univ. Kiel)  "Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation"  June 20, 2012 @ Prague
(De-)Replication of Software Components

- replicate (component: AssemblyComponent, to: ExecutionContainer)
- dereplicate (component: DeploymentComponent)
1 (De-)Replication of Software Components
1. (De-)Replication of Software Components

2. Migration of Software Components
1. (De-)Replication of Software Components

2. Migration of Software Components

3. (De-)Allocation of Execution Containers
Completions/Decorations

SLAStic Meta-Model (cont’d)

SLAStic — Architecture-Based Online Capacity Management

Completions/Decorations

- **Adaptation / Reconfiguration** (e.g., plans, operations, capabilities, properties)
- **Measurement** (e.g., workload, timing, utilization)
- **Usage** (e.g., operation call frequencies, calling relationships)
- ...
Completions/Decorations

- **Adaptation / Reconfiguration** (e.g., plans, operations, capabilities, properties)
- **Measurement** (e.g., workload, timing, utilization)
- **Usage** (e.g., operation call frequencies, calling relationships)
- ...
Evaluation Methodology

SLAStic — Architecture-Based Online Capacity Management

1 Online analysis (production/lab system)
Evaluation Methodology

SLAStic — Architecture-Based Online Capacity Management

1. Online analysis (production/lab system)

Monitoring log from case study

JMeter/Markov4JMeter

Workload Generator

Instrumented, runtime reconfigurable s/w system (or lab system)

SLAStic.MONITORING

SLAStic.RECONFIGURATION

SLAStic.CONTROL

André van Hoorn (Univ. Kiel) »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation« June 20, 2012 @ Prague
Evaluation Methodology

SLAStic — Architecture-Based Online Capacity Management

1 Online analysis (production/lab system)
2 Offline analysis (log replay)
Evaluation Methodology

SLAStic — Architecture-Based Online Capacity Management

1. Online analysis (production/lab system)
2. Offline analysis (log replay)
3. Simulation-based analysis
Agenda

1. Introduction
2. SLAStic — Architecture-Based Online Capacity Management
3. Kieker — Application Performance Monitoring and Dynamic Analysis
4. Conclusion
Kieker: Example Workflow and Use Cases
[van Hoorn et al. 2012]
Kieker — Application Performance Monitoring and Dynamic Software Analysis

Selected use cases in research and industrial practice (+ teaching):
• Online/offline performance evaluation and feedback, e.g.,
  • Continuous monitoring of application behavior and usage
  • Performance anomaly detection and diagnosis
  • (Self-)adaptation control
  • Extraction of software architectural (performance) models and visualizations
  • Simulation (replaying previously monitored stimuli; measurement, logging, and analysis)

Monitoring probe
Software system with monitoring instrumentation
Kieker: Example Workflow and Use Cases
[van Hoorn et al. 2012]

Kieker — Application Performance Monitoring and Dynamic Software Analysis

Software system with monitoring instrumentation

Java probes/samplers:

- Manual instrumentation
- AspectJ
- Spring
- Servlet
- CXF/SOAP
- <your interception technology>
- Sigar
- CPU utilization
- Memory usage
- <your technology>
- <your monitoring probe>

+ basic adapters for
- C#/.NET
- Visual Basic 6/COM
- COBOL
Selected use cases in research and industrial practice (+ teaching):

- Online/offline performance evaluation and feedback, e.g.,
- Continuous monitoring of application behavior and usage
- Performance anomaly detection and diagnosis
- (Self-)adaptation control
- Extraction of software architectural (performance) models and visualizations
- Simulation (replaying previously monitored stimuli; measurement, logging, and analysis)

André van Hoorn (Univ. Kiel)
»Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«
June 20, 2012 @ Prague
Selected use cases in research and industrial practice (+ teaching):

- Online/offline performance evaluation and feedback, e.g.,
- Continuous monitoring of application behavior and usage
- Performance anomaly detection and diagnosis
- (Self-)adaptation control
- Extraction of software architectural (performance) models and visualizations
- Simulation (replaying previously monitored stimuli; measurement, logging, and analysis)
Kieker: Example Workflow and Use Cases
[van Hoorn et al. 2012]
Kieker — Application Performance Monitoring and Dynamic Software Analysis

Monitoring records

Measurement

Monitoring log/stream

Monitoring Records

Current time

Resource utilization

CPU utilization

Operation execution

Control-flow events

Memory/swap usage

File system

Java Messaging Service (JMS)

Java Management Ext. (JMX)

Database (SQL)

Named pipe

<your monitoring reader/writer>

Selected use cases in research and industrial practice (+ teaching):

• Online/offline performance evaluation and feedback, e.g.,

• Continuous monitoring of application behavior and usage

• Performance anomaly detection and diagnosis

• (Self-)adaptation control

• Extraction of software architectural (performance) models and visualizations

• Simulation (replaying previously monitored stimuli; measurement, logging, and analysis)

Software system with monitoring instrumentation
Selected use cases in research and industrial practice (+ teaching):

• Online/offline performance evaluation and feedback, e.g.,
  - Continuous monitoring of application behavior and usage
  - Performance anomaly detection and diagnosis
  - (Self-)adaptation control
  - Extraction of software architectural (performance) models and visualizations
  - Simulation (replaying previously monitored stimuli; measurement, logging, and analysis)
Kieker: Example Workflow and Use Cases

[van Hoorn et al. 2012]

Kieker — Application Performance Monitoring and Dynamic Software Analysis

Analysis configuration (Web GUI)

Monitoring records

Measurement

Analysis log/stream

Analysis

Plugins

Software system with monitoring instrumentation

Selected use cases in research and industrial practice (+ teaching):

• Online/offline performance evaluation and feedback, e.g.,
  • Continuous monitoring of application behavior and usage
  • Performance anomaly detection and diagnosis
• (Self-)adaptation control
• Extraction of software architectural (performance) models and visualizations
• Simulation (replaying previously monitored stimuli; measurement, logging, and analysis)

André van Hoorn (Univ. Kiel)

»Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«
June 20, 2012 @ Prague
Kieker: Example Workflow and Use Cases
[van Hoorn et al. 2012]
Kieker — Application Performance Monitoring and Dynamic Software Analysis

Analysis configuration (Web GUI)

Monitoring records

Measurement

Monitoring log/stream

Analysis

Plugins

Sequence diagrams
Dependency graphs
Call graphs
Visualization
Architecture reconstr.
Pipe-and-filter framework
Analysis/Visualization Plugins

Visualizations

Selected use cases in research and industrial practice (+ teaching):

• Online/offline performance evaluation and feedback, e.g.,
• Continuous monitoring of application behavior and usage
• Performance anomaly detection and diagnosis
• (Self-)adaptation control
• Extraction of software architectural (performance) models and visualizations
• Simulation (replaying previously monitored stimuli; measurement, logging, and analysis)

André van Hoorn (Univ. Kiel)  »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«  June 20, 2012 @ Prague
Selected use cases in research and industrial practice (+ teaching):
Selected use cases in research and industrial practice (+ teaching):

- Online/offline performance evaluation and feedback, e.g.,
  - Continuous monitoring of application behavior and usage
  - Performance anomaly detection and diagnosis
  - (Self-)adaptation control
Selected use cases in research and industrial practice (+ teaching):

- **Online/offline performance** evaluation and feedback, e.g.,
  - Continuous **monitoring** of application behavior and usage
  - Performance **anomaly detection and diagnosis**
  - *(Self-)*adaptation control
- **Extraction of software architectural** (performance) models and visualizations
Selected use cases in research and industrial practice (+ teaching):

- Online/offline performance evaluation and feedback, e.g.,
  - Continuous monitoring of application behavior and usage
  - Performance anomaly detection and diagnosis
  - (Self-)adaptation control
- Extraction of software architectural (performance) models and visualizations
- Simulation (replaying previously monitored stimuli; measurement, logging, and analysis)
Core Kieker Framework Components

Kieker — Application Performance Monitoring and Dynamic Software Analysis

Instrumented Software System

Monitoring Probe

Monitoring Controller

Monitoring Writer

Monitoring Log/Stream

Monitoring Reader

Analysis Controller

Analysis / Visualization Plugin

Kieker.Monitoring

André van Hoorn (Univ. Kiel)  »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«  June 20, 2012 @ Prague
Core Kieker Framework Components

Kieker — Application Performance Monitoring and Dynamic Software Analysis

Instrumented Software System

Monitoring Probe

Monitoring Controller

Logging

Periodic Sampling

JMX Interface

Time Source

Monitoring Writer

Monitoring Record

Monitoring Log/Stream

Monitoring Reader

Analysis Controller

Analysis / Visualization Plugin

Kieker.Monitoring

Kieker.Analysis
Core Kieker Framework Components

Kieker — Application Performance Monitoring and Dynamic Software Analysis

Instrumented Software System

Monitoring Probe → Monitoring Controller → Monitoring Writer

Monitoring Log/Stream

Kieker.Monitoring

Monitoring Controller

JMX Interface

Periodic Sampling

Logging

Time Source

Analysis Controller

Analysis / Visualization Plugin

Kieker.Analysis

André van Hoorn (Univ. Kiel) «Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation» June 20, 2012 @ Prague
Core Kieker Framework Components

Kieker — Application Performance Monitoring and Dynamic Software Analysis

Instrumented Software System

Kieker.Probe

Monitoring Controller

Monitoring Writer

Monitoring Record

Monitoring Log/Stream

Kieker.Analysis
Core Kieker Framework Components

Kieker — Application Performance Monitoring and Dynamic Software Analysis

Instrumented Software System

Monitoring Probe

Monitoring Controller

Monitoring Writer

Monitoring Log/Stream

Monitoring Record

Analysis Controller

Analysis / Visualization Plugin

André van Hoorn (Univ. Kiel)  »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«  June 20, 2012 @ Prague
Core Kieker Framework Components

Kieker — Application Performance Monitoring and Dynamic Software Analysis

Instrumented Software System

Monitoring Probe

Monitoring Controller

Logging

Monitoring Writer

JMX Interface

Periodic Sampling

Monitoring Record

Monitoring Log/Stream

Monitoring Reader

Analysis Controller

Analysis / Visualization Plugin

Kieker.Monitoring

Monitoring Log/Stream

Kieker.Analysis
Kieker.Analysis example pipes-and-filters configuration

- Performance anomaly detection and visualization
- Architecture and trace reconstruction/visualization
Kieker.Analysis example pipes-and-filters configuration

- Performance anomaly detection and visualization
- Architecture and trace reconstruction/visualization
Example Pipe-and-Filter Configuration

Kieker — Application Performance Monitoring and Dynamic Software Analysis

Kieker.Analysis example pipes-and-filters configuration

- Performance anomaly detection and visualization
- Architecture and trace reconstruction/visualization

Andre van Hoorn (Univ. Kiel) » Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation « June 20, 2012 @ Prague 14 / 18
**Example Pipe-and-Filter Configuration**

Kieker — Application Performance Monitoring and Dynamic Software Analysis

---

**Kieker.Analysis example pipes-and-filters configuration**

- Performance anomaly detection and visualization
- Architecture and trace reconstruction/visualization

---

Andre van Hoorn (Univ. Kiel) «Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation» June 20, 2012 @ Prague
Kieker.Analysis example pipes-and-filters configuration

- Performance anomaly detection and visualization
- Architecture and trace reconstruction/visualization
Example Pipe-and-Filter Configuration

Kieker — Application Performance Monitoring and Dynamic Software Analysis

Kieker.Analysis example pipes-and-filters configuration
- Performance anomaly detection and visualization
- Architecture and trace reconstruction/visualization

<<Reader>>
: FS reader

<<Repository>>
: System model repository

<<Filter>>
: Trace reconstruction filter

<<Filter>>
: Performance anomaly filter

<<Filter>>
: Anomaly graph plotter

operationExecutions

systemModel

traceEvents

messageTraces

executionTraces

anomalyRatings

André van Hoorn (Univ. Kiel)
»Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«
June 20, 2012 @ Prague
Kieker.Analysis example pipes-and-filters configuration

• Performance anomaly detection and visualization
• Architecture and trace reconstruction/visualization
Kieker.Analysis example pipes-and-filters configuration

- Performance anomaly detection and visualization
- Architecture and trace reconstruction/visualization
• 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
**Kieker — Application Performance Monitoring and Dynamic Software Analysis**

- 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

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

http://kieker-monitoring.net

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

André van Hoorn (Univ. Kiel)  »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«  June 20, 2012 @ Prague  15 / 18
Telecommunication provider, distributed setting

- **Goal**: Workload characterization and performance evaluation
- **Java-based technology**: Servlet, Spring, and CXF/SOAP
- **Continuous Monitoring** (utilizing Kieker):
  - Probes for collecting distributed application-level trace and performance data
  - Continuous monitoring in production use since 12/2009
- **Model extraction examples** (316,980 traces):

  Container dependency graph (with calling frequencies)
Deployment-level component dependency graph (with calling frequencies):

André van Hoorn (Univ. Kiel)  »Architecture-based Online Capacity Management: Modeling, Monitoring, and Adaptation«  June 20, 2012 @ Prague
What I’ve Talked About

Conclusion

SLAStic

- Self-adaptive capacity management based on application-level workload
- Adaptation goal: Increased resource efficiency while meeting SLAs
- Means for adaptation: Architectural runtime reconfiguration
- Use of architectural (performance) models at development & runtime
- Domain: Business-critical distributed component-based software systems

Kieker

- Modular 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

Feel free to contact me: avh@informatik.uni-kiel.de


