

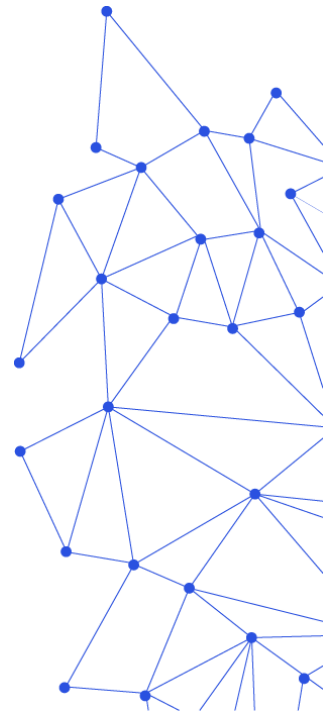
SPECjEnterprise2010 & Java Enterprise Edition (EE) PCM Model Generation DevOps Performance WG Meeting 2014-07-11

Andreas Brunnert

Performance & Virtualization Group, Information Systems Division

fortiss GmbH

An-Institut Technische Universität München



Agenda

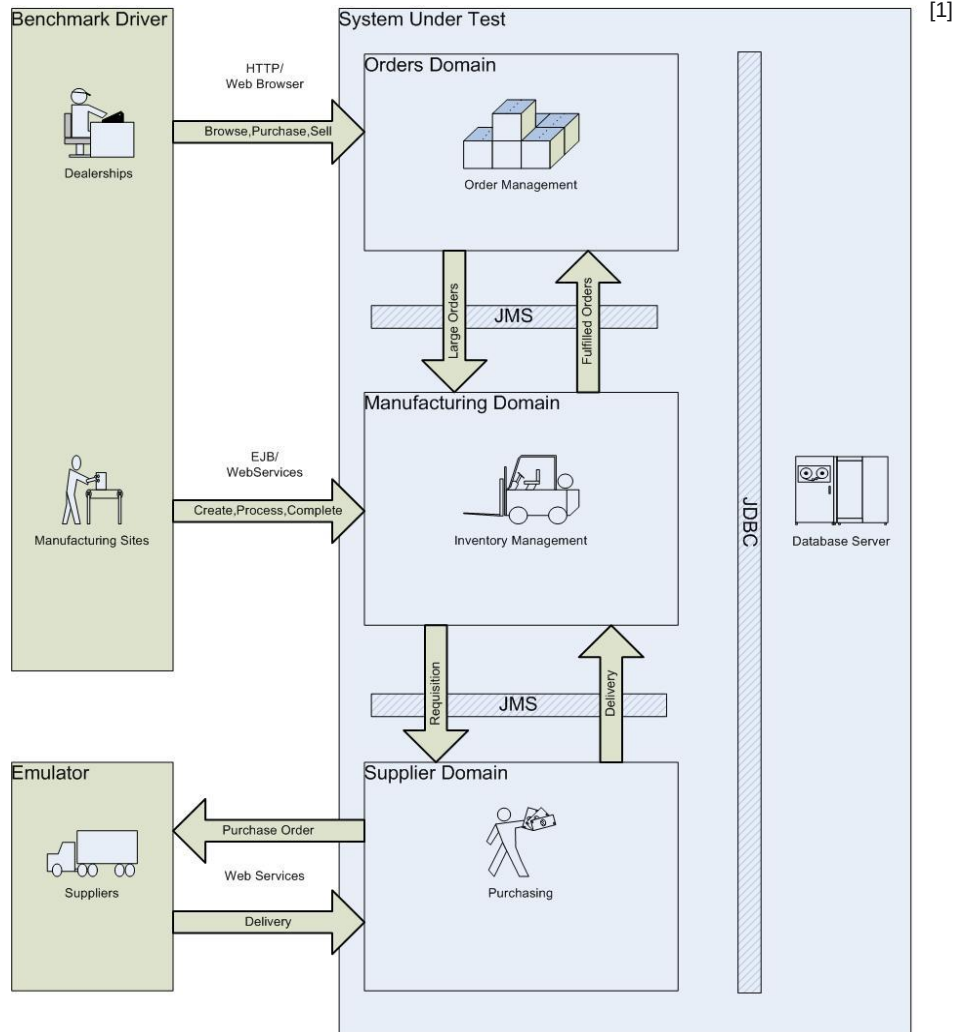
- SPECjEnterprise2010
 - Architecture
 - Orders Domain Workload
 - Benchmark Execution Procedure
 - Pitfalls
- Automatic Performance Model Generation
 - Data Collection
 - Data Aggregation
 - Model Generation
- Next Steps

Agenda

- SPECjEnterprise2010
 - Architecture
 - Orders Domain Workload
 - Benchmark Execution Procedure
 - Pitfalls
- Automatic Performance Model Generation
 - Data Collection
 - Data Aggregation
 - Model Generation
- Next Steps

SPECjEnterprise2010

Architecture



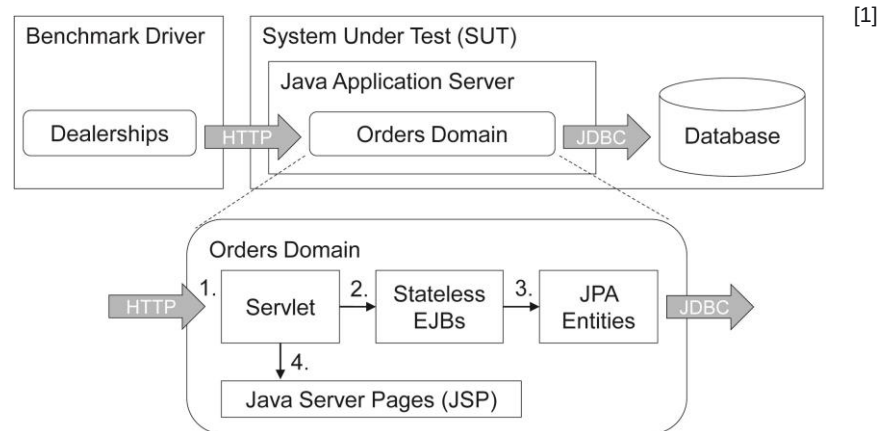
[1] Image Source: http://www.spec.org/jEnterprise2010/docs/DesignDocumentation.html#1_2_Business_and_Application_Env

[2] Details about components, db tables: <http://www.spec.org/jEnterprise2010/docs/DesignDocumentation.html>

SPECjEnterprise2010

Architecture

- Orders Domain (CRM): Java EE Web Application

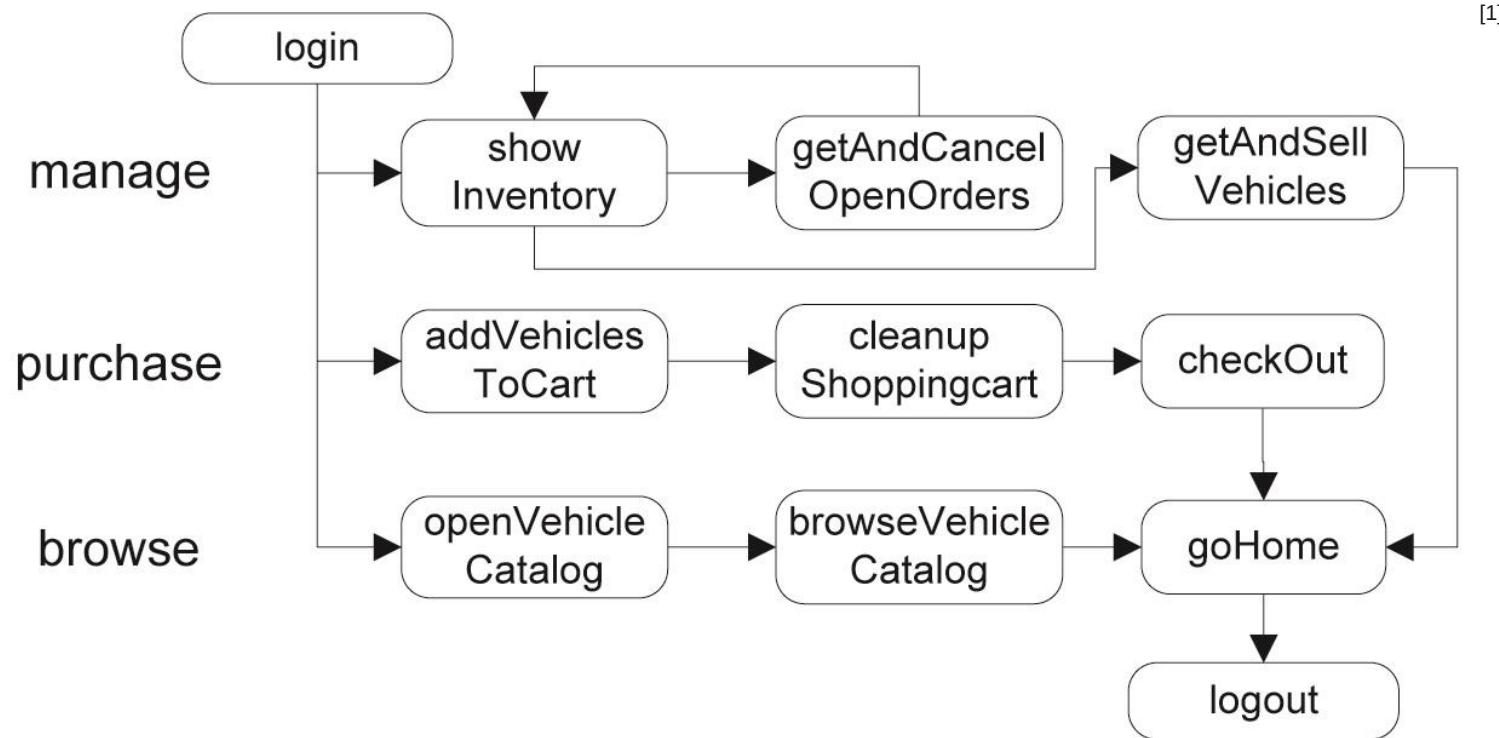


- Manufacturing Domain
 - Enterprise JavaBeans (EJBs) driven by the Orders Domain and a special Manufacturing Domain Workload to produce cars
- Supplier Domain (SCM)
 - Web-Services used by the Manufacturing domain to order parts required during the manufacturing process → no specific driver only indirectly

[1] Image Source: Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

SPECjEnterprise2010

Orders Domain Workload



[1] Image Source: Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

SPECjEnterprise2010

Benchmark Execution Procedure

- SPECjEnterprise uses the Faban Benchmarking Harness to specify the workload and to execute load on the system
- Most important input variable: **Scale (Needs to be at least 50)**
- Scale specifies:
 - The number of users; Scale*10
 - Their behavior - in terms of how many entities they access during the execution
 - The number of items in the database

© 2005-2009 Faban

Java	Driver	DealerDriver	Mfg Driver	Servers
Description				
App-Monitored (Update 17) SPECjEnterprise2010 (Faban-VM-Cores: 8; SPECj-VM-Cores: 6; 300/600/150;				
Host				
SUSEDRIVERJBOS				
Stats Collection				
Interval				
Run time Stats				
Run time Stats Interval				
Scale				

SPECjEnterprise2010

Benchmark Execution Procedure

- SPECjEnterprise2010 checks the SUT configuration (e.g. database content, caching behavior of JPA) before every benchmark execution!!!
 - **Therefore, one always needs to reload the database before executing a benchmark!**
- Afterwards, the Benchmark can be executed – however, one still needs to specify a Ramp-Up, steady state and Ramp-Down phase for the benchmark execution. This time depends on:
 - Scale/Number of Users
 - JVM settings such as Xms/Xmx
 - The minimum ramp-up time can be calculated based on the delay that is specified between thread-starts (default: 100ms)
- We recommend ~10 minutes for ramp-up (even for up to 4300 users) and a thread start delay of 100ms.

SPECjEnterprise2010

Benchmark Execution Procedure

Java Driver DealerDriver Mfg Driver Servers © 2005-2009 Faban

Description	App-Monitored (Update 17) SPECjEnterprise2010 (Faban-VM-Cores: 8; SPECj-VM-Cores: 6; 300/600/150;
Host	SUSEDRIVERJBOSS
Stats Collection Interval	30
Run time Stats	false
Run time Stats Interval	30
Scale	70
Tools	NONE
Delay between thread starts (ms)	100
Start Threads simultaneously	false
Start Agents in parallel	true
Audit	true
Stop if audit failed	true

Run Control	
Ramp Up	300
Steady State	600
Ramp Down	300

SPECjEnterprise2010

Pitfalls

- Not fully loaded databases:
 - SPECjEnterprise2010 allows a small tolerance for the database content
 - This means that one is able to re-run a benchmark if a previous one has been canceled after a few minutes (as not many items have been modified)
 - However, this might change your results
- Different JPA implementations (Hibernate vs. EclipseLink)
 - Check your persistence.xml carefully to see if caching is disabled in the correct – technology specific - syntax
- „Stateful“ JPA – Some JPA Implementations (e.g. the one in JBoss 7.1.1.) are quite sensitive to schema changes, so **two** reboots might be required to get the atomicity tests running

SPECjEnterprise2010

Pitfalls

- Full garbage collections during the benchmark execution
 - Either make sure that the memory is big enough to handle one benchmark run or let the benchmark run long enough to reduce the impact of single spikes in the results
- For Linux: set all language and time settings to en_us!!!
 - Otherwise you might not be able to see the graph results in your benchmark harness
- Use systems with the ability to measure time intervals below 10 ms (e.g. CentOS, openSuse, Solaris, AIX, ...) and not Windows or Ubuntu as you might get incorrect results
 - See dissertation of Michael Kuperberg for more details
- Make sure that the hostname in the Driver configuration matches the one in hosts file of the driver OS

Agenda

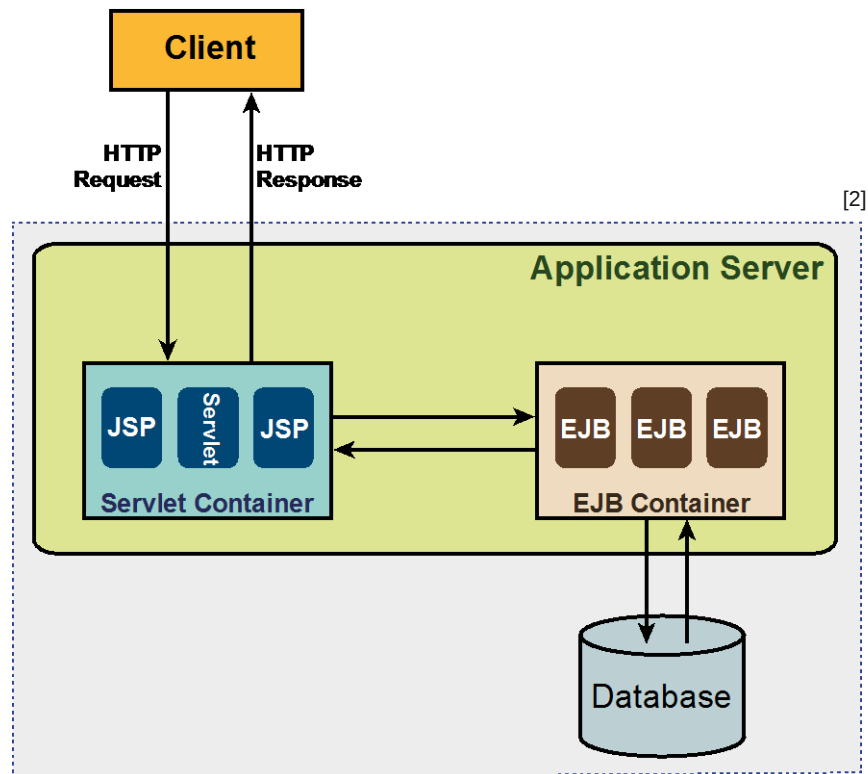
- SPECjEnterprise2010
 - Architecture
 - Orders Domain Workload
 - Benchmark Execution Procedure
 - Pitfalls
- Automatic Performance Model Generation
 - Data Collection
 - Data Aggregation
 - Model Generation
- Next Steps

Automatic Performance Model Generation

Java Enterprise Edition (Java EE)

- Typical Java EE Application Architecture (Example):

- Components according to the Java EE Specification:^[1]
 - Servlets / Java Server Pages (JSP) handle requests/responses
 - Enterprise Java Beans (EJB) contain Business-Logic
 - Database contains persistent data



[1] Content Source: Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

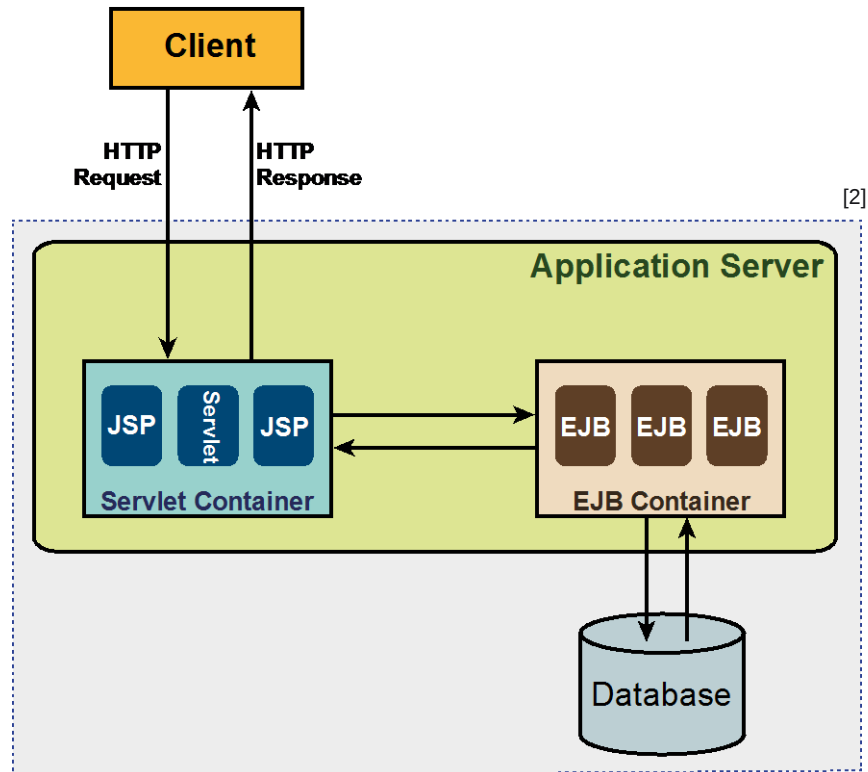
[2] Image Source: Neubig, Stefan (2014): "Evaluierung der Ressourcenabschätzung einer beispielhaften Unternehmensanwendung auf Basis eines automatisch generierten Performance-Modells." Bachelor's Thesis, Technische Universität München 2014.

Automatic Performance Model Generation

Data Collection^[1]

- To represent these predefined component types the following data needs to be collected

- Servlets, JSPs, EJBs names
- Component operation names
- Component relationships on the level of their public operations
- Resource demand of these component operations
- External systems such as databases



[1] Content Source: Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

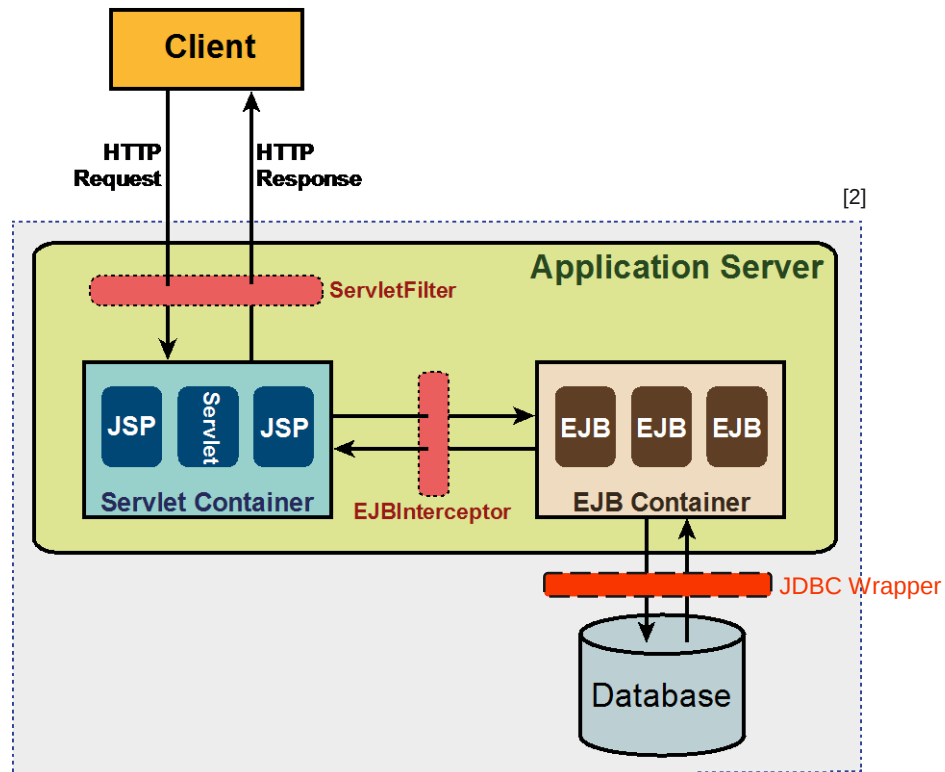
[2] Image Source: Neubig, Stefan (2014): "Evaluierung der Ressourcenabschätzung einer beispielhaften Unternehmensanwendung auf Basis eines automatisch generierten Performance-Modells." Bachelor's Thesis, Technische Universität München 2014.

Automatic Performance Model Generation

Data Collection^[1]

- To represent these predefined component types the following data needs to be collected

- Servlets, JSPs, EJBs names
- Component operation names
- Component relationships on the level of their public operations
- Resource demand of these component operations
- External systems such as databases



[1] Content Source: Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

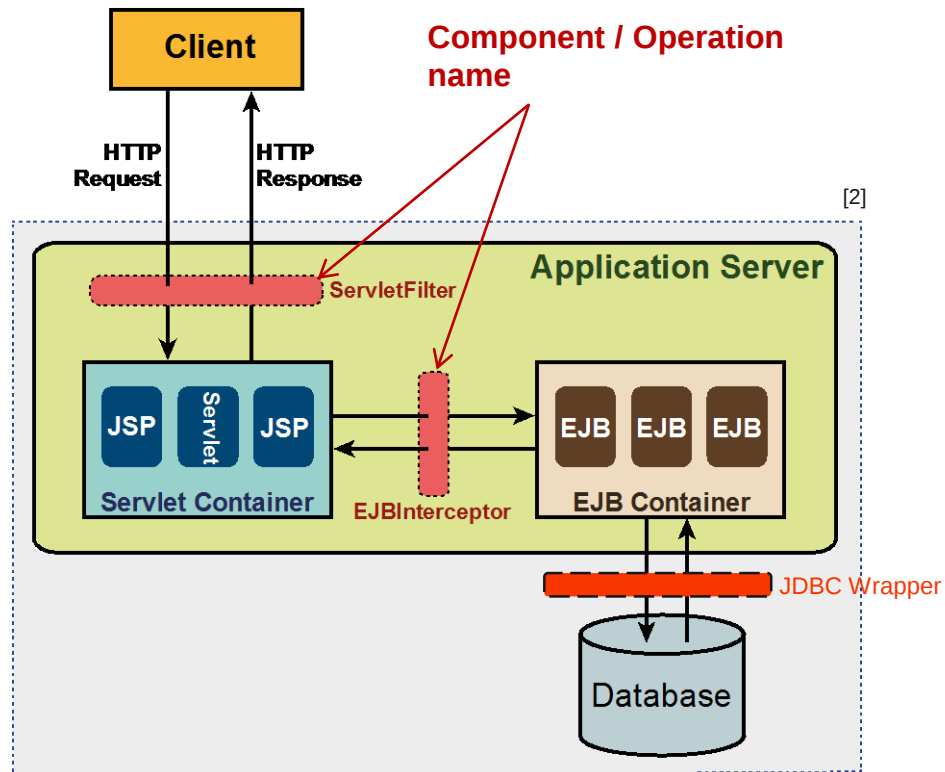
[2] Image Source: Neubig, Stefan (2014): "Evaluierung der Ressourcenabschätzung einer beispielhaften Unternehmensanwendung auf Basis eines automatisch generierten Performance-Modells." Bachelor's Thesis, Technische Universität München 2014.

Automatic Performance Model Generation

Data Collection^[1]

- To represent these predefined component types the following data needs to be collected

- Servlets, JSPs, EJBs names
- Component operation names
- Component relationships on the level of their public operations
- Resource demand of these component operations
- External systems such as databases



[1] Content Source: Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

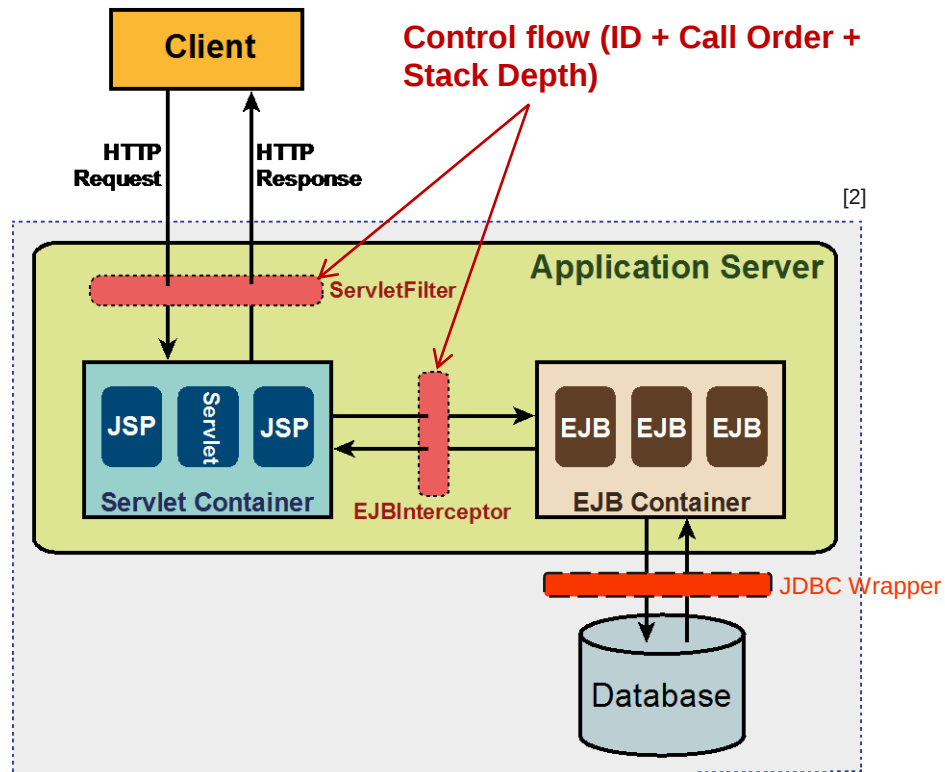
[2] Image Source: Neubig, Stefan (2014): "Evaluierung der Ressourcenabschätzung einer beispielhaften Unternehmensanwendung auf Basis eines automatisch generierten Performance-Modells." Bachelor's Thesis, Technische Universität München 2014.

Automatic Performance Model Generation

Data Collection^[1]

- To represent these predefined component types the following data needs to be collected

- Servlets, JSPs, EJBs names
- Component operation names
- Component relationships on the level of their public operations
- Resource demand of these component operations
- External systems such as databases



[1] Content Source: Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

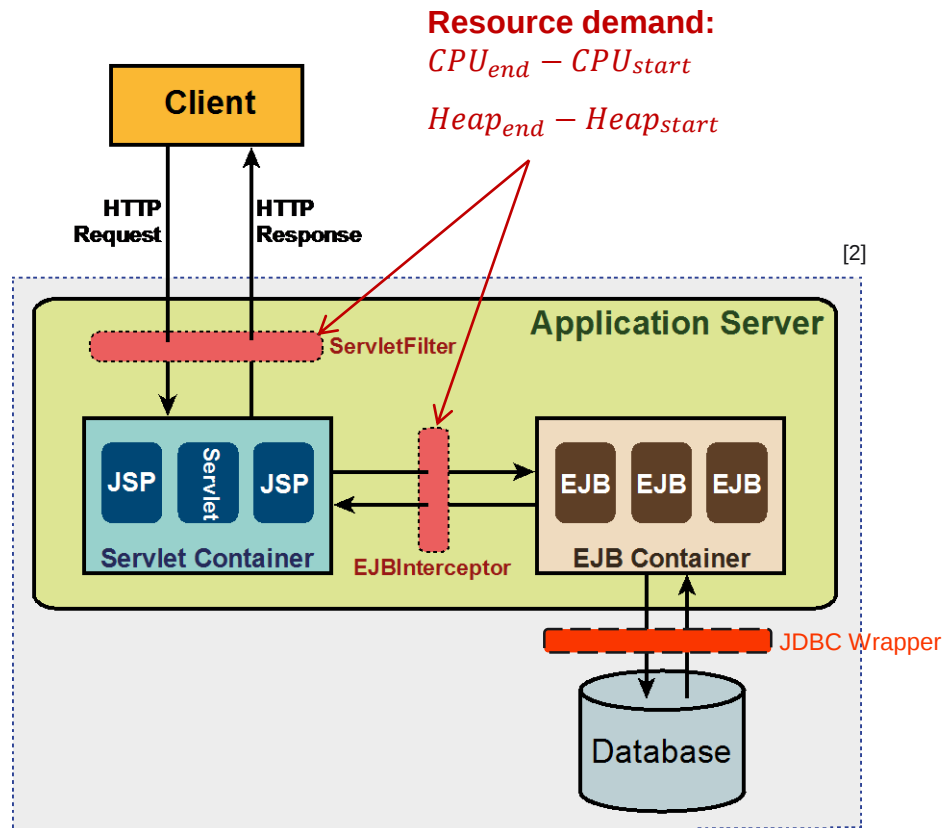
[2] Image Source: Neubig, Stefan (2014): "Evaluierung der Ressourcenabschätzung einer beispielhaften Unternehmensanwendung auf Basis eines automatisch generierten Performance-Modells." Bachelor's Thesis, Technische Universität München 2014.

Automatic Performance Model Generation

Data Collection^[1]

- To represent these predefined component types the following data needs to be collected

- Servlets, JSPs, EJBs names
- Component operation names
- Component relationships on the level of their public operations
- Resource demand of these component operations
- External systems such as databases



[1] Content Source: Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

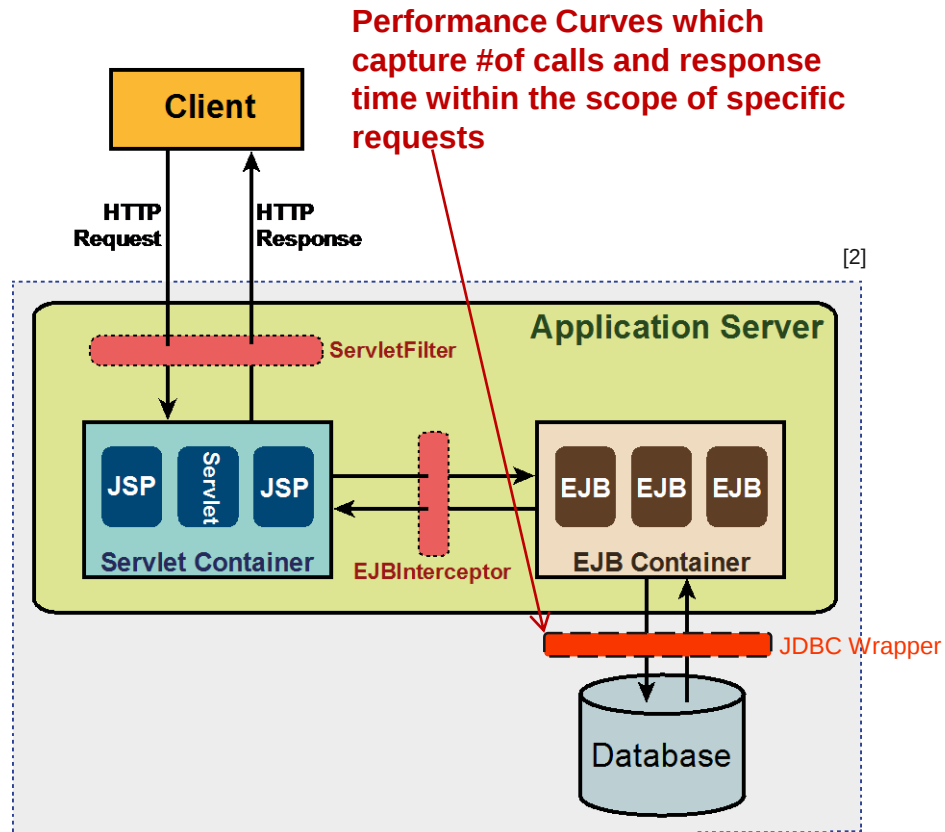
[2] Image Source: Neubig, Stefan (2014): "Evaluierung der Ressourcenabschätzung einer beispielhaften Unternehmensanwendung auf Basis eines automatisch generierten Performance-Modells." Bachelor's Thesis, Technische Universität München 2014.

Automatic Performance Model Generation

Data Collection^[1]

- To represent these predefined component types the following data needs to be collected

- Servlets, JSPs, EJBs names
- Component operation names
- Component relationships on the level of their public operations
- Resource demand of these component operations
- External systems such as databases



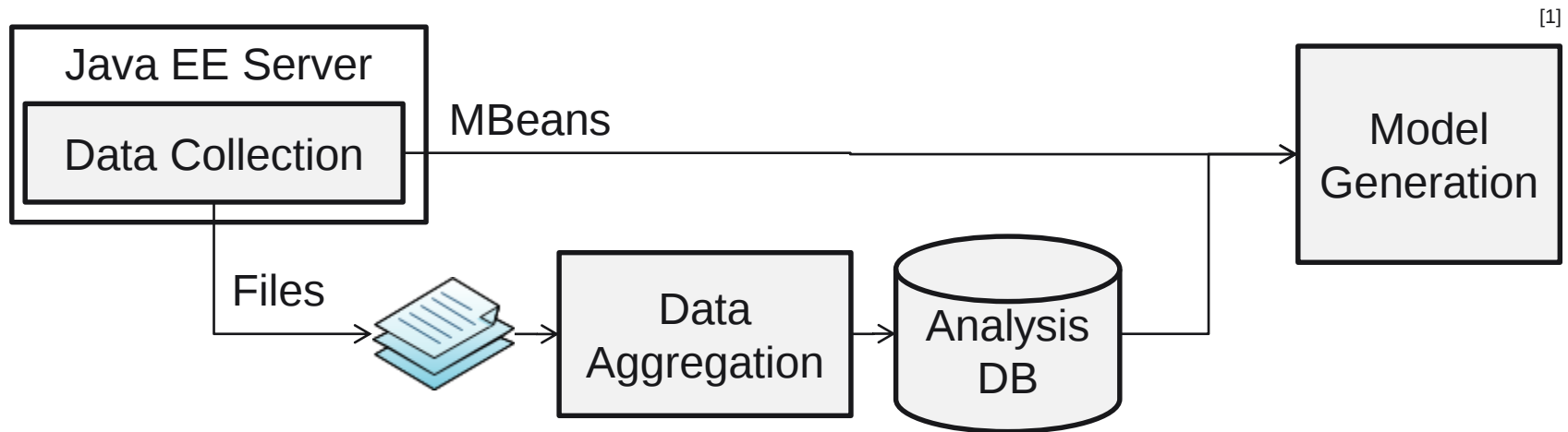
[1] Content Source: Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

[2] Image Source: Neubig, Stefan (2014): "Evaluierung der Ressourcenabschätzung einer beispielhaften Unternehmensanwendung auf Basis eines automatisch generierten Performance-Modells." Bachelor's Thesis, Technische Universität München 2014.

Automatic Performance Model Generation

Data Aggregation

- The Data Collection and Model Generation Components can work with either work with InvocationBasedPersistence Services (Files) or AggregatedPersistenceServices (MBeans, DBs)



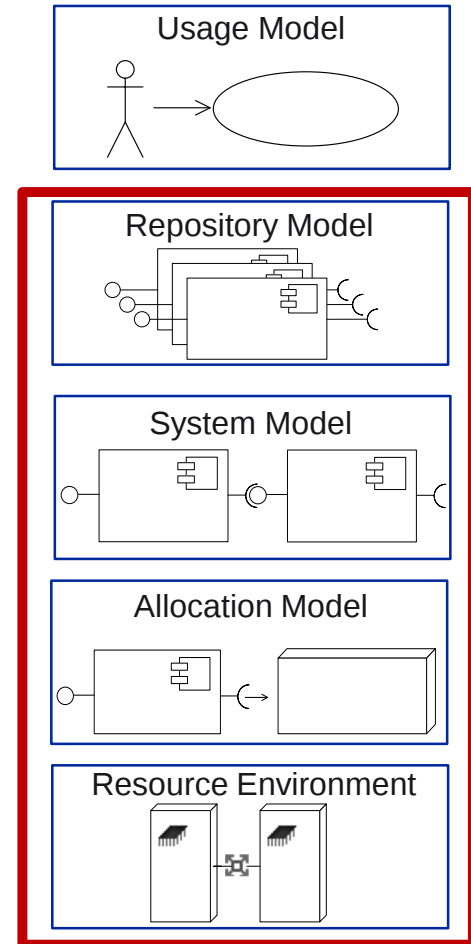
[1] Image Source (adapted): Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

Automatic Performance Model Generation

Model Generation

- Java-based client
- Based on Data:
 - Stored in Runtime Mbeans
 - Stored in Files or an Analysis Database
- Default repository, system, allocation and resource environment are generated automatically
- PCCurve Specification file is also generated automatically and is also associated with the system model and its corresponding repository model interfaces

[1]

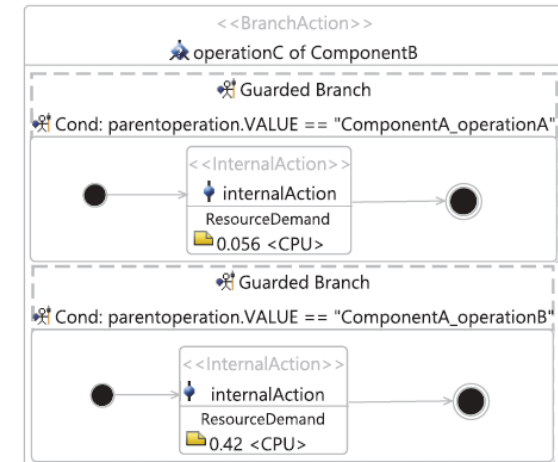
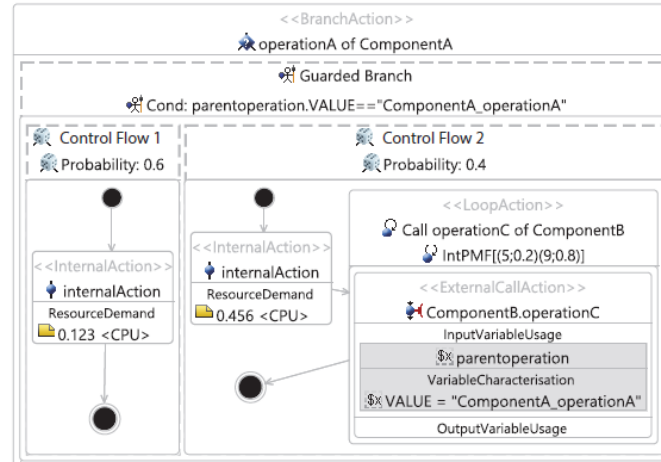
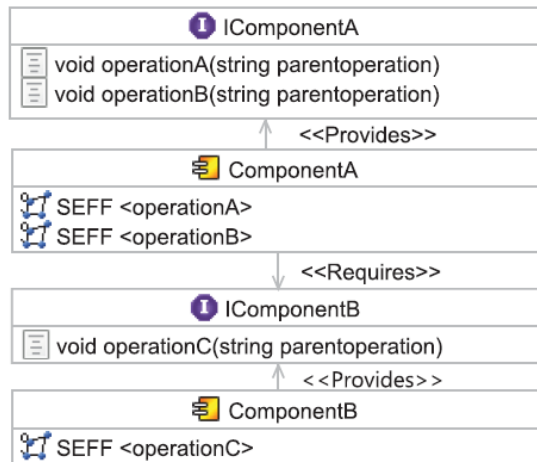


[1] Image Source: Wischer, Kilian (2014): "Abbildung von Ressourcen-Profilen für Unternehmensanwendungen durch Performance-Modelle." Master's Thesis, Technische Universität München 2014.

Automatic Performance Model Generation

Model Generation

[1]



(a) Repository model example [4] (b) RDSEFF of ComponentA.operationA [4] (c) RDSEFF of ComponentB.operationC

[1] Image Source (adapted): Brunnert, Andreas; Vögele, Christian and Krcmar, Helmut (2013): "Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications." In: Computer Performance Engineering 10th European Workshop on Performance Engineering, EPEW 2013, Venice, Italy, S. 74-88.

Agenda

- Introduction
- Motivation
- Automatic Performance Model Generation
 - Data Collection
 - Data Aggregation
 - Model Generation
- Evaluation
 - SPECjEnterprise2010 Deployment
 - Model Generation
 - Measurement and Simulation Results in Comparison
- Related Work
- Outlook

Outlook

- Usage Model Generation
 - Master Thesis ongoing based on session traces
- Setting thread limits for each tier (Web, EJB, JDBC Conn. Pools)
- Representing Distributed systems
 - Distributing the transaction ID within distributed Java EE environments using the Application Response Measurement (ARM) standard or similar means
 - Aggregating data from multiple Java EE instances
- Modeling Java EE server clusters
- Supporting new component types:
 - JSF, web-services, Message driven beans, Other Java EE 7.0 enhancements, e.g. Batch Jobs, ..

Thanks for your attention!

Questions?

Contact // Andreas Brunnert

fortiss GmbH

An-Institut Technische Universität München

Guerickestraße 25 • 80805 München • Germany

tel: +49 89 3603522 510 **mail:** andreas.brunnert@fortiss.org

fax: +49 89 3603522 50 **web:** www.fortiss.org