

Allgemeines

<i>Dozent:</i>	Prof. Dr. Gerd Beneken
<i>Verantwortlich:</i>	Prof. Dr. Gerd Beneken
<i>Studiengang:</i>	Bachelor
<i>Pflicht/FWPF:</i>	Pflicht
<i>Voraussetzungen:</i>	Programmieren 3, Datenkommunikation, Betriebssysteme
<i>Sprache:</i>	Deutsch
<i>Lehrform:</i>	4 SWS seminaristischer Unterricht und Übungen
<i>Arbeitsaufwand:</i>	60 Stunden Präsenzzeit, 90 Stunden Selbststudium
<i>Leistungspunkte:</i>	5
<i>Medienform:</i>	Vortrag mit Overhead-Projektor und Laptop mit Beamer, Skript
<i>Prüfung:</i>	Mündliche Prüfung

Lernziele und Inhalt

Richtziel

Die Studierenden kennen die wichtigsten Architekturprobleme verteilter Systeme und können die technische Komplexität derartiger Systeme einschätzen. Sie können die wichtigsten Architekturmuster, Algorithmen und Techniken anwenden und damit selbstständig verteilte Systeme in einer objektorientierten Programmiersprache implementieren. Die Studierenden können selbstständig serviceorientierte Systeme, Systeme auf der Basis einer anderen synchronen Middleware sowie Systeme auf der Basis einer asynchronen Middleware entwerfen und implementieren.

Inhalt

Die Grundideen verteilter Verarbeitung werden anhand von den folgenden Technologien beschrieben: Sockets, RMI, Web-Services, REST, JMS und EJB, die alle in Java (JSE, JavaEE) zur Verfügung stehen. Diese Technologien werden mit verwandten Technologien von Microsoft insbesondere WCF und Standards, insbesondere CORBA verglichen und in Beziehung gesetzt. Im Vordergrund steht nicht die eher zufällige Syntax des jeweiligen API, sondern die Frage, wie man durch eine sinnvolle Softwarearchitektur die Abhängigkeiten von einer gegebenen Basistechnik minimiert. Die wichtigsten Entwurfsmuster für verteilte Systeme werden vorgestellt.

Inhalt

1. Wiederholung: Thread-Programmierung, Java-Streams, (XML-)Serialisierung
2. Grundlagen der Netzwerkprogrammierung: (Java-)Sockets
Funktionsweise der Java Sockets
Architurentwurf mit Sockets: Marshaller, Remote-Proxy, Forwarder-Receiver
3. Entfernter Methodenaufruf: RMI und Web-Services
Funktionsweise von RMI, Web-Services, CORBA und Co.
Architurentwurf mit RMI: Schnittstellen Design, Data Transfer Object, Referenzfreiheit
4. Ressourcen-Orientierte Architekturen: REST über HTTP
5. Nachrichtenorientierte Middleware (MOM): JMS, MSMQ
Funktionsweise von JMS (Java Messaging Service)

Architekturentwurf mit MOM: Asynchrone Verarbeitung, Publisher/Subscriber, Enterprise Integration Patterns

6. Architektur verteilter Systeme

Architekturschemata: Client/Server, Multi-Tier, SOA, Peer-to-Peer, ...
Idee des Application-Servers / Containers, Inversion-of-Control, Interceptor

7. Mehrwertdienste in verteilten Systemen

Namensdienste am Beispiel LDAP über JNDI
Sitzungsmanagement am Beispiel Servlet und Enterprise JavaBeans
Transaktionsmanagement und verteilte Transaktionen
Management verteilter Systeme und JMX

In den Übungen werden mehrere Fallbeispiele (Nummernserver, Remote Shell und Homebanking) mit verschiedenen Basistechniken realitätsnah implementiert.

Literatur

Besonders empfohlen

1. Hohpe, Woolf: *Enterprise Integration Patterns*, Addison-Wesley (2003)
2. Völter, Kirchner, Zdun: *Remoting Patterns*, Wiley, (2004)
3. Völter, Schmid, Wolf: *Server Component Patterns*, Wiley (2002)

Zu den Java-Technologien: vgl. aktuelle Empfehlungen in der Vorlesung

Zusätzlich empfohlen:

4. Lea, D.: *Concurrent Programming in Java*. Addison Wesley 2nd Edition (1999)
5. Fowler, M: *Patterns of Enterprise Architecture*. Addison-Wesley (2002)
6. Buschmann, Meunier, Rohnert, Sommerlad: *Pattern-oriented Software Architecture* (Vol 1), Wiley, (1996)
7. Kirchner, Jain: *Pattern-oriented Software Architecture* (Vol 3):
Patterns for Ressource Management, Wiley, (2007)
8. Buschmann, Henny, Schmidt: *Pattern-oriented Software Architecture* (Vol. 4):
A Pattern Language for Distributed Computing, Wiley, (2007)
9. Stevens, W. R.: *Unix Network Programming*. Prentice Hall (1990)
10. Tilkov: *REST und HTTP*, dpunkt (2009)