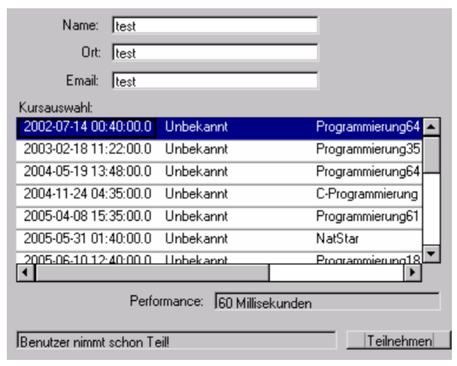
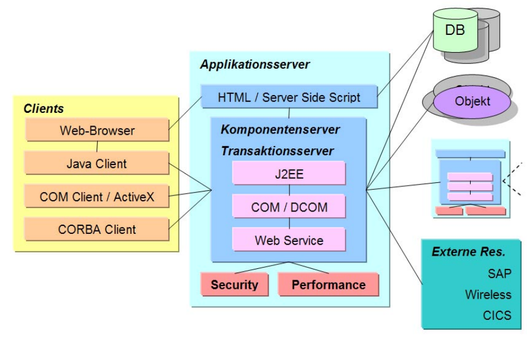
FACHHOCHSCHULE BRANDENBURG
FACHBEREICH INFORMATIK UND MEDIEN



Einsatz von Applikationsservern Untersucht am Beispiel des Sybase „Enterprise Application Server“

Diplomarbeit, vorgelegt von Andreas Schwarzbauer



Aufgabenstellung:

Der Einsatz von Applikationsservern ist im Rahmen mehrstufiger Architekturkonzepte zu einer praktisch relevanten Basistechnologie für Datenbankanwendungen geworden.

Es soll eine für Lehr- und Demonstrationsaufgaben geeignete Beispielanwendung für einen typischen Einsatzfall dieser Technologie entworfen und prototypmäßig implementiert werden.

Dabei sind die an der Einrichtung verfügbaren Sybase Softwareprodukte, insbesondere der Sybase Adaptive Server Enterprise, Release 12 und der Enterprise Application Server, Release 3.5 zu nutzen.

Die praktischen Arbeiten sind in allgemeine aus der Literatur und über das Internet ableitbare Aussagen zur Motivation für die Nutzung von Applikationsservern, deren Architektur und eine Übersicht zum Stand verfügbarer kommerzieller Produkte einzuordnen.

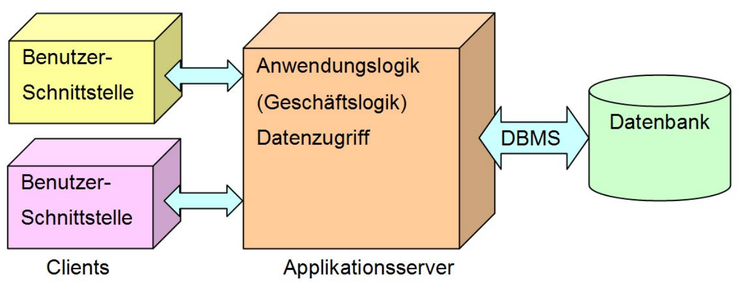
Aus den bei der Schaffung der Beispielanwendung gemachten Erfahrungen sollten Verallgemeinerungen und eigene Bewertungen abgeleitet werden.

Was ist ein Applikationsserver?

Eine Applikation ist eine ausführbare Anwendung.
Ein Server stellt in einem Netzwerk eine Anzahl von Diensten bereit. Folglich stellt ein Applikationsserver Anwendungen in einem Netzwerk zur Verfügung.
Der Begriff "Applikationsserver" umschreibt meistens die Anwendungslogik eines mehrschichtigen Client-Server-Systems.

Mehrschichtige Client-Server-Systeme:

Bei einem Drei-Schichten-Modell wird die eigentliche Anwendungslogik in eine zusätzliche Schicht ausgegliedert. Dadurch entsteht ein Applikationsserver, der z.B. höherwertige Funktionen in Form von Objekten und deren Methoden anbietet. Diesen Teil nennt man auch Geschäftslogik oder „Business Logic“. Die Software auf dem Applikationsserver nennt man „Middleware“.



Da der Client keinen direkten Zugriff auf den Datenbestand hat, wird sichergestellt, dass ein Client nur die für ihn freigegebenen Daten sehen und bearbeiten kann. Für die Kommunikation zwischen Client und Applikationsserver werden spezielle Middleware-Lösungen wie z.B. CORBA verwendet.

Applikationsserver:

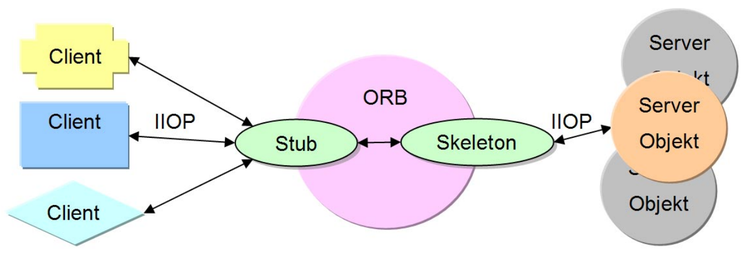
Die Anwendungsfelder für Applikationsserver wachsen ständig, und so gibt es immer mehr Anbieter von Applikationsserver-Lösungen. Die meisten Systeme basieren auf der Java 2 Enterprise Edition.
Es gibt aber auch Systeme, die nicht auf Java basieren und keine bzw. nur in einigen Schnittstellen eine Java-Unterstützung bieten, wie z.B. Mircosoft .NET.

Java 2 Enterprise Edition:

Die Enterprise Edition von Java 2 oder kurz J2EE wurde von Sun entwickelt und steht für eine auf JAVA 2 basierende Standard-Architektur zur Definition und Unterstützung von mehrschichtigen Programm-modellen.
Die Stärken von J2EE liegen in dem Zusammenspiel von kleinen Client-Applikationen (sog. „Thin-Client applications”) mit der Applikationslogik auf dem Server (z.B. über "CORBA"). Durch die Verlagerung der Applikationslogik benötigen die „Thin-Client applications” kein leistungsfähiges System bzw. Netz.

CORBA:

CORBA steht für „Common Object Request Broker Architecture” und wurde von der Object Management Group (OMG) entwickelt. Es ist ein System, welches einzelne Objekte über ein Netzwerk zur Verfügung stellt. Das Besondere daran ist, dass es die Kommunikation von Applikationen ermöglichen soll, egal wo sie im Netz liegen oder womit sie entwickelt worden sind.



Die zentrale Komponente bei CORBA ist der „Object Request Broker“ oder kurz "ORB". Der ORB sucht das vom Client aufgerufene Objekt, leitet den Aufruf und die übergebenen Werte weiter und gibt das Ergebnis an den Client zurück. Dazu benötigt der Client nur noch die Objektspezifikation, um die Ein- und Ausgabe korrekt auszuführen.

Die Praxis:

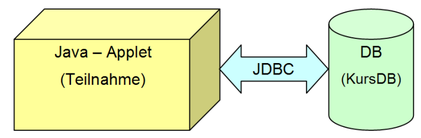
An einem praktischen Beispiel wird die Erstellung einer Client-Server-Anwendung und einer Java-Komponente veranschaulicht. Durch die Umsetzung in zwei verschiedenen Versionen werden die Unterschiede der klassischen Client-Server Programmierung zur komponentenorientierten Programmierung aufgezeigt.

Die Aufgabe:

Für einen Anbieter von Aufbau- und Lernkursen existiert eine Datenbank zur Speicherung der Kurse, Lehrer, Teilnehmer und Material.
Die Teilnahme an einem Kurs soll über ein Java-Applet möglich sein, welches über das Internet aufgerufen wird.

Die Client-Server Lösung:

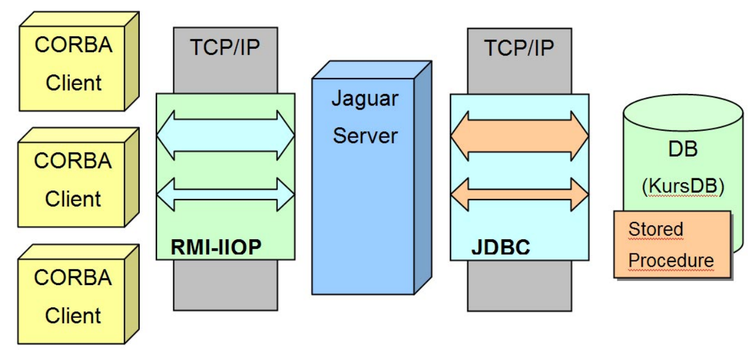
Die Teilnahme an einem Kurs läßt sich als Client-Server Anwendung umsetzten. Ein Java-Client nutzt eine Verbindung zum Datenbankserver über die JDBC-Schnittstelle, um Daten zu lesen und zu ändern.



Die komponentenbasierte Lösung:

Bei der zweiten Möglichkeit der Umsetzung wird keine JDBC-Schnittstelle benötigt. Es werden über den Komponentenserver „Jaguar“ mit der CORBA Schnittstelle zwei Methoden aufgerufen:

· Die Ausgabe der Angebote (Angebot_holen)
· Die Teilnahme (Teilnahme)



Die größten Vorteile dieser Implementierung sind die Wiederverwendbarkeit der Komponenten in anderen Clients und die erweiterte Sicherheit. Durch die festgelegte Infrastruktur können viele Programmteile durch geeignete Entwicklungswerkzeuge automatisch generiert werden. Dadurch sinkt der Programmieraufwand, besonders bei komplexeren Projekten.

Komplexität, Performance und Nutzen:

Die meisten Web-Applikationen werden heutzutage im Browser dargestellt und nutzen serverseitige Scriptsprachen oder Serversysteme, welche HTML-Output liefern. Diese Systeme stellen schon einfache Applikationsserver dar. Durch den geringen Kommunikationsaufwand können solche einfachen Systeme schneller sein als große, komplexe Applikationsserver. Die Interaktion über den Web-Browser ist jedoch stark eingeschränkt. In Zukunft werden diese Systeme mehr und mehr durch komplexere, komponentenbasierte Systeme ersetzt, was aber eine schnelle Verbindung zum Internet voraussetzt.
Komponentenbasierte Systeme kommen momentan nur in größeren Firmen zum Einsatz. Sie besitzen einen großen Kern an Funktionalität, der auf verschiedene Arten verwendet werden kann, sei es als Web-Seite oder als interner Client. Damit diese komplexen Systeme einwandfrei funktionieren, muß eine gut durchdachte Infrastruktur vorhanden sein. Ab einem gewissen Grad an Komplexität ist ein System auf Komponentenbasis besser zu handhaben als andere Systeme.
Es gibt mit der Enterprise Edition von Java jetzt schon einen weit verbreiteten Standard für die komponentenbasierte Programmierung, der von den meisten Applikationsservern unterstützt wird.

Betreuer: Prof. Dr. habil. M. Günther
Dipl.-Inform. I. Boersch

Abgabe: 22.04.2002