Application Server

- Begriff und Einordnung
- Basistechniken
- Enterprise JavaBeans (EJB)

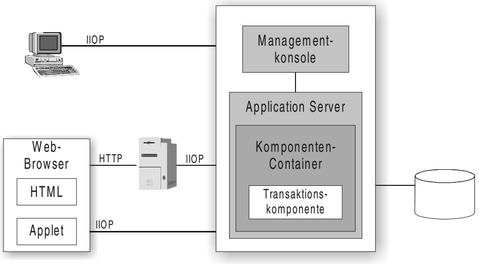
Application Server: Motivation

- Geschäftsanwendungen im Internet
  - mehrstufige Architekturen (Geschäftslogik)
  - Integration von Legacy-Systemen
- komplexe Entwicklung
  - Sicherheitsanforderungen
  - Transaktionsunterstützung
  - heterogene Umgebung (DBMS, Verteilungsplattformen, Benutzerschnittstellen)

Application Server: Begriff

- Vereinigung von Middleware-Technologien
  - Web-Server
  - Transaktionsmonitore
  - Object Request Broker
  - Sicherheitslösungen
  - Datenbankzugriffslösungen
- Entwicklungsumgebung
- Managementlösung

Application Server: Aufbau



Application Server: Basistechniken

- Unterstützung verschiedener Clients
  - HTML, Java-Applets, Standalone-Applikationen
  - Trennung von Anwendungslogik und Benutzerschnittstelle
  - HTML-Generierung
- Entwicklungsunterstützung
  - Application Server als Entwicklungsumgebung
  - Entwicklung von Web-Benutzerschnittstellen
  - Installation von Geschäftskomponenten

Basistechniken

- Gewährleistung von Skalierbarkeit und Verfügbarkeit
  - Verbindungsmanagement
  - Lastausgleich
  - Fehlermanagement
- Sicherheitstechniken
  - Bereitstellung von Verschlüsselungstechniken für
    - Signierung (digitale Unterschriften)
    - sichere Kanäle (Secure Socket Layer)

Basistechniken

- Transaktionsmanagement in verteilten Umgebungen
  - Sicherstellung von Transaktionen über verteilte Objekte
    - 2-Phasen-Commit-Protokolle
    - Transaktionsmonitore
  - Sessionmanagement für Web-Clients
- Datenbankzugriff
  - Einbindung von Datenbanken zur persistenten Speicherung von Geschäftsobjekten
  - Zugriff auf Legacy-Daten

Basistechniken

- Performance-Monitoring
  - Überwachung des Laufzeitverhaltens der Applikationen
  - Aufspüren von Engpässen
- Applikationsmanagement und -konfiguration
  - zentrale Verwaltung und Fehlerüberwachung
  - Installation und Pflege von Applikationen
  - Lastausgleich

Technologien

- Verteilung:
  - CORBA, DCOM
- Transaktionsmanagement:
  - CORBA Transaction Service (OTS)
- Sicherheit:
  - X509-Zertifikate, Secure Socket Layer
- DBMS:
  - RDBMS, Object-Relational-Mapping
- Komponenten:
  - JavaBeans, ActiveX-Controls

Systeme

- WebLogic (BEA Systems)
- NetDynamics (Sun)
- Web Application Server (Inprise)
- Web Application Server (Oracle)
- Domino (Lotus)
- WebSphere (IBM)
- PowerTier (Persistence)
- Zope (Digital Creations)

Enterprise JavaBeans (EJB)

- Serverseitige Komponenten für Geschäftsanwendungen
- Trennung von Geschäftslogik und Standardfunktionen
- Anwendungsgebiet: Applikationsserver
  - Unterstützung verschiedener Protokolle (HTTP, RMI, IIOP)
  - Standarddienste

JavaBeans-Komponentenmodell

- Bean:
  - wiederverwendbare Softwarekomponente, die in einem visuellen Softwarekompositionswerkzeug (Builder) verwendet und manipuliert werden kann
  - besitzt *Eigenschaften*: z.B. Kontostand
  - meldet *Ereignisse*: Zustandsänderungen
  - stellt *Methoden* bereit: z.B. Überweisung
- Implementierung:
  - einzelne Klasse
  - Menge von Klassen + Ressourcen (Archiv)

JavaBeans

- Manipulation bzgl.
  - Eigenschaften: Lesen, Ändern
  - Ereignissen: Festlegen des Empfängers, Definieren des Handler-Codes
  - Methoden: Verbinden mit Ereignissen
- Builder:
  - ermittelt und manipuliert Bean-Eigenschaften (Introspection bzw. spezielle Eigenschafteneditoren)
  - generiert notwendigen Adapter-Code zur Verbindung von Ereignissen mit Methoden

EJB: Charakteristika

- EJB: Komponente, die Geschäftslogik auf Geschäftsdaten implementiert (Kunde, Konto, Bestellung)
- EJB-Instanzen werden durch Container erzeugt und verwaltet
- Anpassung (Customizing) durch Manipulation der Eigenschaften
- separate Bereitstellung der Standarddienste
- Vermittlung des Zugriffs über Container

EJB: Potentiale

- Portabilität bei Konformität zur EJB-Spezifikation
- Anpassung ohne Änderung von bzw. Zugriff auf Quellcode
- Konzentration auf Geschäftslogik
- Komponentenorientierte Entwicklung (Wiederverwendbarkeit)

EJB: Standarddienste

- Lebenszyklus
  - Verwaltung des Lebenszklus von Beans (Erzeugung, Aktivierung, ...)
- Zustandsmanagement
  - Repräsentation von Zuständen und Sitzungen
- Sicherheit
  - Vergabe und Verwaltung von Zugriffsberechtigungen
- Transaktionen
  - Unterstützung von Transaktionen über mehrere Methodenaufrufe hinweg
- Persistenz
  - Speichern und Wiederherstellen der Daten von Beans in bzw. aus Datenbanken

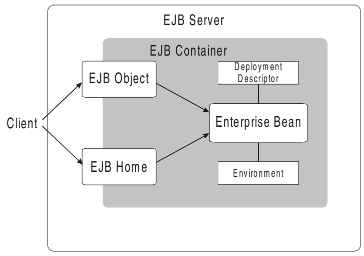
EJB: Aufteilung

- Server:
  - Serverprozeß für ein oder mehrere Container
  - Bereitstellung der Standarddienste
- Container:
  - Ablaufumgebung
  - Abbildung der Bean-Attribute auf Datenspeicher
- Client:
  - Inanspruchnahme von Diensten (durch Beans bereitgestellt)

EJB: Aufteilung

- Beans: Geschäftsobjekte
  - SessionBeans
    - leichtgewichtige, nicht-persistente Komponenten
    - repräsentieren nicht direkt Datenobjekte
    - werden vom Client erzeugt
  - EntityBeans
    - persistentes Geschäftsobjekt
    - repräsentiert Eintrag in der Datenbank
    - können durch mehrere Clients genutzt werden

EJB: Architektur



EJB: Architektur

- EJB: als JAR-Archiv geliefert
  - Sammlung von Interfaces, Klassen und Ressourcen
    - Home Interface: Lifecycle-Schnittstelle zum Auffinden, Erzeugen und Löschen
    - Remote Interface: Vereinbarung der eigenentlichen „Geschäfts“-Methoden
    - Deployment Descriptor: Verhalten bzgl. der Standarddienste (Persistenz, Transaktionen)
    - Bean-Klasse + generierte Klassen

EJB: Home Interface

```
import java.rmi.RemoteException;
import javax.ejb.*;

public interface AccountHome extends EJBHome {
    public abstract
          Account create (String s, double d)
          throws CreateException, RemoteException;

    public abstract
          Account findByPrimaryKey (AccountPK pk)
          throws FinderError, RemoteException;
}
```

EJB: Remote Interface

```
import java.rmi.RemoteException;
import javax.ejb.*;

public interface Account extends EJBObject {
    public abstract double deposit (double d)
          throws RemoteException;

    public abstract double withdraw (double d)
          throws RemoteException;

    public abstract double balance ()
          throws RemoteException;
}
```

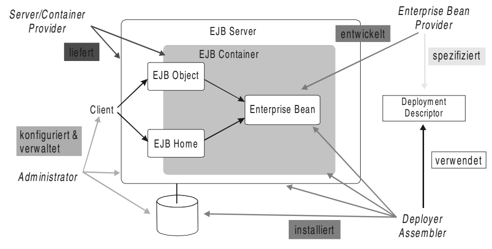

EJB: Ablauf

- Installation
  - Installation eines EJB im Container
  - Registieren des Home Interface in einem Namensdienst
- Nutzung
  - Suchen des Home Interface über Namensdienst
  - evtl. Erzeugen einer Instanz
  - Aufruf der Methoden des Remote Interface über Container
    - Weiterleitung und Realisierung der Dienste (Persistenz, Sicherheit, Transaktionen)
    - Schnittstellen zwischen Bean und Container: automatisch generierte Verbindungsklassen

EJB: Persistenz

- EntityBeans repräsentieren persistente Daten
  - Ausführung persistenzspezifische Funktionen beim Erzeugen, Aktivieren, Deaktivieren und Löschen
  - Art der Persistenzimplementierung in Deployment Descriptor
- Bean-verwaltete Persistenz:
  - Datenspeicherung und -wiederherstellung wird vom Bean-Entwickler implementiert (z.B. über JDBC)
- Container-verwaltete Persistenz:
  - Bereitstellung und Aufruf der Persistenzfunktionen durch Container

EJB: Entwicklung



EJB: Technologien

- Namens- und Verzeichnisdienst:
  - JNDI (DNS, NDS, NIS+, LDAP, COS Naming)
- Kommunikation:
  - RMI, IIOP (CORBA), Java Messaging
- Transaktionen:
  - Java Transaction API, Java Transaction Service API
- Datenbankzugriff:
  - JDBC