Application Server (Middleware)

RWTH AACHEN UNIVERSITY

Dinge, die eine Middleware leisten kann:

- **Load balancing (Clients werden auf wenig ausgelastete Server geleitet)**
- **Transparent failover (Bei einem Server-Crash übernimmt ein anderer Server)**
- **Clustering (State wird über Server repliziert, so dass transparent failover möglich wird)**
- **Dynamic redeployment (Software-Updates ohne das System stoppen zu müssen)**
- **Caching (Gemeinsamer Cache über alle Server hinweg)**
- **Und vieles mehr…**

Application Server (Middleware)

RWTH AACHEN UNIVERSITY

Explizite Middleware Ansteuerung

Transfer(Konto k1, Konto k2, long betrag)

1. **API: Starte eine Transaktion**
2. **API: Lade die Konten aus der Datenbank**
3. **Addiere/Subtrahiere den Betrag**
4. **API: Speichere die Konten in der Datenbank**
5. **API: Beende die Transaktion**

- **Viel Arbeit**
- **Schwer zu warten**

Application Server (Middleware)

RWTH AACHEN UNIVERSITY

Implizite Middleware Ansteuerung

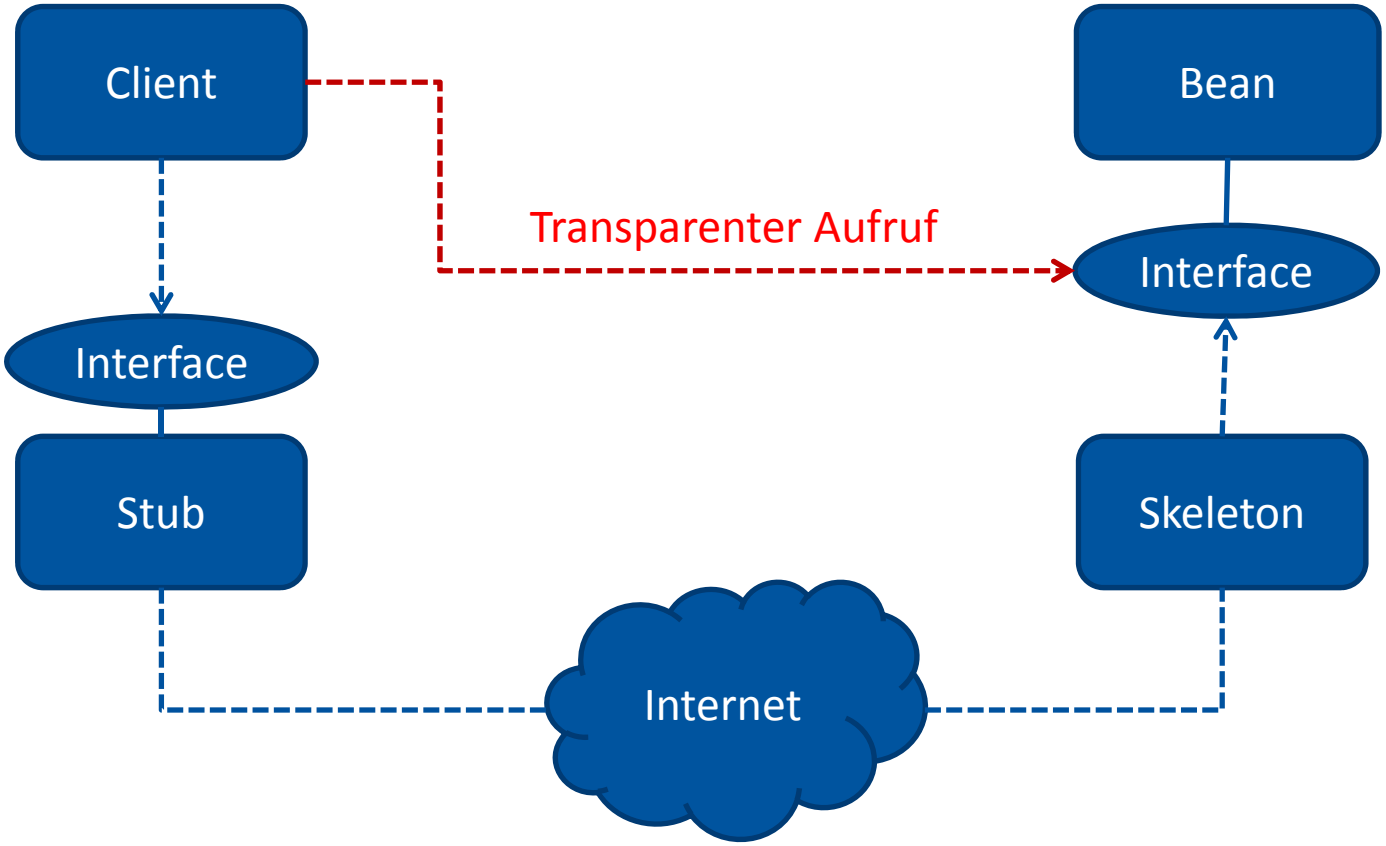
Transfer(Konto k1, Konto k2, long betrag)

1. **Addiere/Subtrahiere den Betrag**

- **Wenig Arbeit**
- **Leicht zu warten**
- **In EJB kann man die explizite Ansteuerung auch noch verwenden**

Application Server (Middleware)

- **Verbindung vom Client zur Middleware**



Session Bean

- **Kurzlebige Objekte, die eine „Session“ lang leben**
- **Die Dauer einer Session kann alles mögliche sein**
  - → Solange ein Browser-Fenster offen ist
  - → Solange ein Applet/eine Applikation läuft
- **Session Beans werden vom Server erzeugt und freigegeben**
  - → Keine Garantie, ob man dasselbe Bean-Objekt bei der nächsten Session nochmal bekommt
  - → Eine Session Bean existiert (aus Client-Sicht) von der Anforderung einer Referenz bis zu deren Freigabe.

Stateless Session Bean

- **Kein „Zustand“ (State) wird im Bean-Objekt vorgehalten (so dass dieser Zustand über mehrere Funktionsaufrufe hinweg vorhanden ist)**
- **Funktionsaufrufe als Services**
  - → Hole Artikel zu einer Kategorie
  - → Prüfe die Liquidität einer Person bzgl. eines Geldbetrags
  - → Buche einen Raum zu einem Termin
- **Es könnte sein, dass für jeden Funktionsaufruf eine neue Bean instanziiert wird und danach wieder freigegeben (Tatsächlich: Pool von Bean-Objekten)**

Stateless Session Bean

In der benutzten Eclipse J2EE Version:

- **In der Toolbar „New Server“ auswählen**
  - → Jboss AS 7.1 auswählen
  - → Weiter…
  - → „Home Directory“ auswählen

    C:\Local\Programme\EJBWork\jboss-as-7.1.1.Final
  - → Finish
- **Als Projekt „EJB Project“ nehmen**

Stateless Session Bean

- **Neue Session Bean**
  - → „Java package“: „server“
  - → „Class name“: „HelloBean“
  - → „State type“: Stateless
  - → Create business interface: Remote: „client.Hello“
  - → „No-interface View“ abschalten
- **Schnittstelle „Hello“:**
  - → String greet(String whom);

Stateless Session Bean: Client

- **Einstellungen, wo der Application Server zu finden ist (jboss-ejb-client.properties)**
- **Java Naming and Directory Interface (JNDI) Lookup, um die Session Bean zu finden**
- **Die Stub-Klassen werden vom Application Server Hersteller geliefert (jboss-client.jar)**

```
Hello hello = lookupHello();
String greeting = Hello.greet();
```

Stateless Session Bean

RWTH AACHEN UNIVERSITY

Aufgabe

- **Neue Klasse „Position“ mit X- und Y-Koordinate**
- **Erweitern der Schnittstelle „Hello“**

  → String greet(String whom, Position at);

- **Rückgabestring für**

```
greet( “you“, new Position(12, 65) );
```

  soll sein

```
Hello you at 12,65
```

- **Folgerung: Alles, was über das Netzwerk geht, muss serialisierbar sein!**