Orientation in Objects

# EJB 2.1 Persistenz und JDO im Vergleich

**Orientation in Objects GmbH**
Weinheimer Str. 68
68309 Mannheim
http://www.oio.de
info@oio.de

**Dirk M. Sohn**
<sohn@oio.de>

**Tobias Kieninger**
<kieninger@oio.de>



## Web Polemik

Orientation in Objects

„While it was widely used in designs a year ago, JDO will probably go down in history as the proverbial chicken that crossed the road when the CMP2.0 truck came along.

BLUE "Why I Love EJBs" by Marc Fleury
http://www.jboss.org/modules/html/blue.pdf

"Thomas Hobson, a seventeenth-century liveryman in Cambridge, England, told every customer he could have any horse he wanted as long as it was the one nearest the door. ...EJB's choice is Hobson's Choice. If you don't like EJB's locking, caching, performance, failure or persistence mechanisms, then write your own, says JavaSoft."

EJB's 101 Damnations by Matt Stephens
http://www.softwarereality.com/programming/ejb/conceptual.jsp

## Automatische Objekt-Persistenz

Orientation in Objects

- Persistenz von Objekten
  - Objektexistenz (Identität / Zustand ) über Programmlaufzeit hinaus
- Persistenz-Schicht
  - Architekturschicht für diese Aufgabe (manuell / automatisch)
- Aufgaben der Persistenz-Schicht
  - einfach, Objektstruktur möglichst wenig beeinflussen
  - Transaktionsunterstützung anbieten
  - Speichern, Laden und Abfragen ermöglichen
  - Mapping auf nicht OO-Speichermodelle anbieten

## Agenda

Orientation in Objects

- EJB 2.1 Persistenz
  - Container Managed Persistence
  - Local Interfaces
  - Container Managed Relationships
  - EJB-QL
  - Auswirkungen aufs Design
- JDO Einführung
- Vergleich der beiden Modelle
- Architekturvarianten

## 2 Arten von Entity Beans

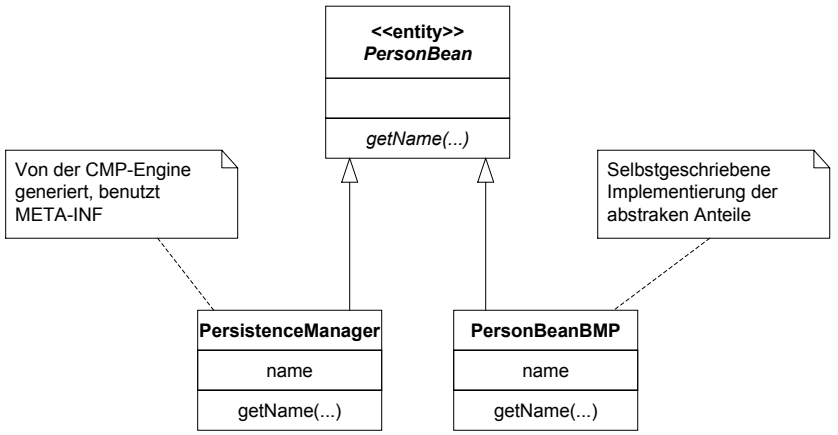
Orientation in Objects

- Der Entwickler kann entscheiden, ob er einen der mitgelieferten Persistenzmechanismen verwenden möchte:
  - Bean Managed Persistance (= BMP)
    - **Eigene Persistenzmechanismen implementieren:**
    - **Missverständlicher Name: nicht Bean-managed sondern „Bean Provider managed“**
    - **Das bedeutet, der Geschäftslogik Entwickler ist gleichzeitig Persistenzexperte (und entwickelt deshalb eine DA-Schicht)**
    - **Oder er kauft ein O/R-Tool**
  - Container Managed Persistance (= CMP)
    - **Fertige Persistenzmanager nutzen**

## Harmonischer Umgang CMP / BMP

Orientation in Objects



## CMP EJB 1.1 - öffentliche Attribute

Orientation in Objects

```
public class PersonBean implements EntityBean {

   public long id;
   public String name;

...
}
```

## CMP 2.0 - Code

Orientation in Objects

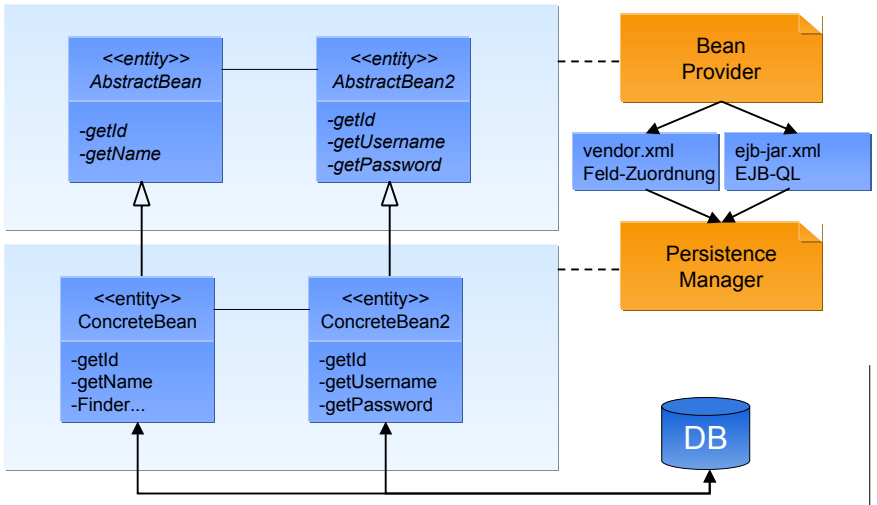
```
public abstract class PersonBean implements EntityBean {

   public abstract long getId();
   public abstract void setId(long Id);
   public abstract String getName();
   public abstract void setName(String name);
   public abstract String getChristianName();
   public abstract void setChristianName(String cname);

}
```

## CMP 2.0 - abstrakte Zugriffsmethoden



## CMP 2.0 - ejb-jar.xml

```
<entity>
   <ejb-name>Person</ejb-name>
   ..
   <persistence-type>Container</persistence-type>
   <cmp-version>2.x</cmp-version>
   <abstract-schema-name>Person</abstract-schema-name>
   <cmp-field>
       <field-name>personId</field-name>
   </cmp-field>
   <cmp-field>
       <field-name>name</field-name>
   </cmp-field>
   <primkey-field>personId</primkey-field>
..
</entity>
```

## CMP 2.0 - jbosscmp-jdbc.xml

Orientation in Objects

```
<entity>
   <ejb-name>Person</ejb-name>
   <table-name>PERSON</table-name>
   <cmp-field>
        <field-name>personId</field-name>
        <column-name>PERSON_ID</column-name>
   </cmp-field>
   <cmp-field>
        <field-name>name</field-name>
        <column-name>NAME</column-name>
   </cmp-field>
</entity>
```

## Agenda

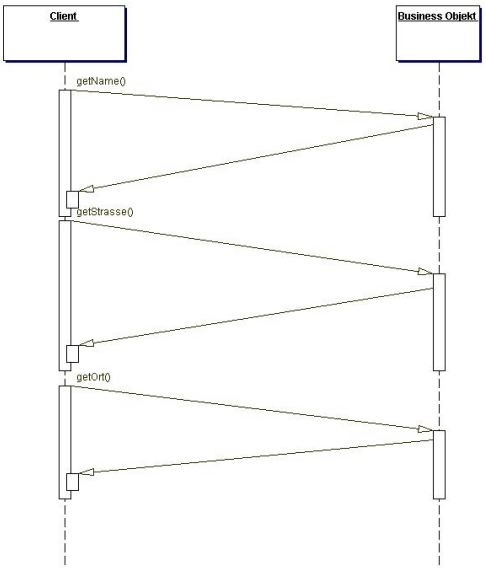
Orientation in Objects

- EJB 2.1 Persistenz
  - Container Managed Persistence
  - Local Interfaces
  - Container Managed Relationships
  - EJB-QL
  - Auswirkungen aufs Design
- JDO Einführung
- Vergleich der beiden Modelle
- Architekturvarianten

## EJB Nutzung ohne Value Objects

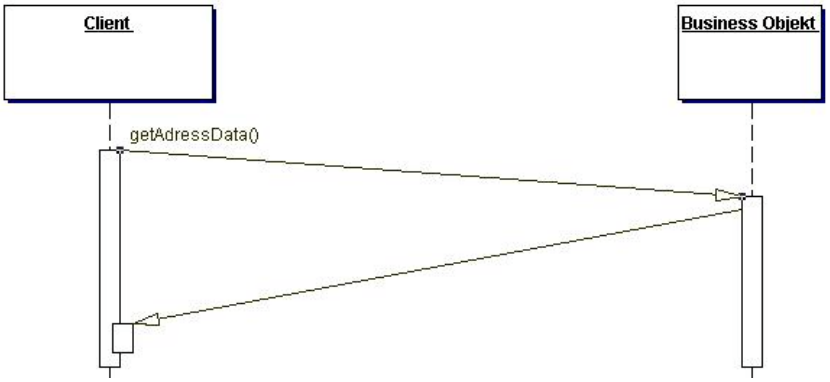
Orientation in Objects



## EJB Nutzung mit Value Objects

Orientation in Objects



## Local Interfaces - Vorgeschichte

Orientation in Objects

- Konsequenzen von ausschließlichen remote-Aufrufen
  - nicht viele einzelne Aufrufe - jeder ist teuer
- Konsequenzen von "call by value"
  - nicht gerne große Daten übertragen - nur per value möglich

=> wenige Aufrufe mit mittelkleinen Daten

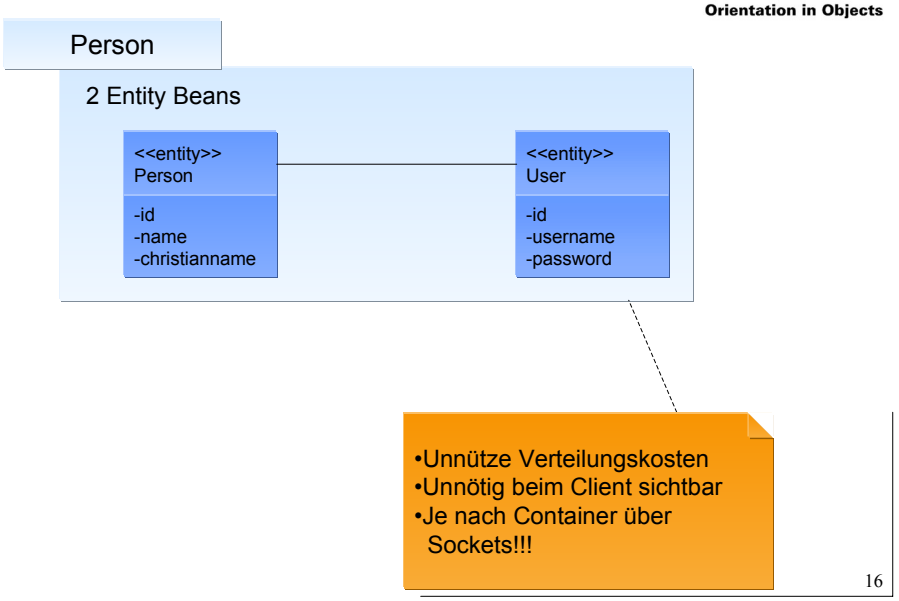
=> Im Prinzip pro Entität getData und setData mit Value Objects

=> Keine Aggregate möglich

## EJB 1.1 - Aggregat mit Remote Entity

Orientation in Objects



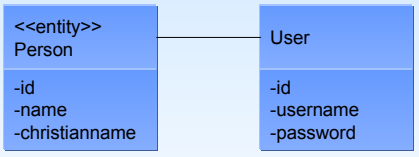

 

## EJB 1.1 - Aggregat mit Dependent Value Class

Orientation in Objects

Person

1 Entity Bean + 1 Dependent Value Class




- Kosten der Serialisierung
- Keine Abfrage auf User möglich

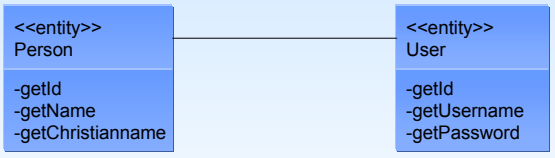
 

## EJB 2.0 - Aggregat mit Local Entity

Orientation in Objects

Person

1 Entity Bean + 1 Local Entity Bean




- Keine Verteilungskosten
- Trotzdem Abfragen möglich

## Remote vs. Locale Interface


- Vorteile Remote Interfaces
  - Schwache Kopplung zwischen Client und Bean
  - Verteilungsunabhängigkeit
  - Isolation - call by value
- Vorteile Locale Interfaces
  - Preiswerter Zugriff
  - Möglichkeit Daten zu teilen - call by reference
  - **CMR**

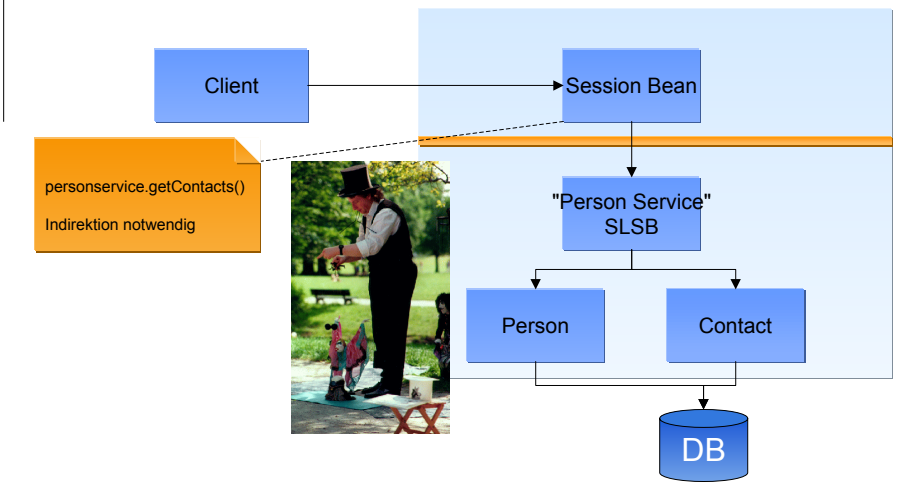
 

## Agenda


- EJB 2.1 Persistenz
  - Container Managed Persistence
  - Local Interfaces
  - Container Managed Relationships
  - EJB-QL
  - Auswirkungen aufs Design
- JDO Einführung
- Vergleich der beiden Modelle
- Architekturvarianten

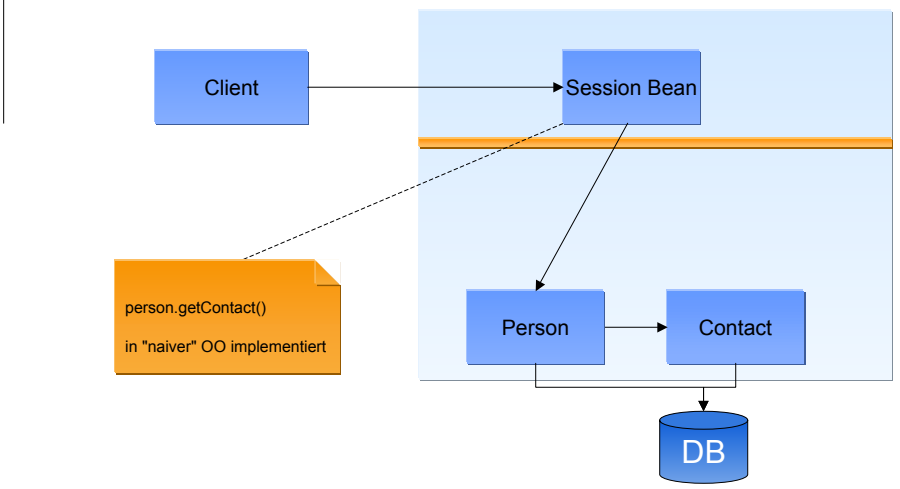
 

## Beziehungen in EJB 1.1



## Beziehungen in EJB 2.0



## Beziehungen - Code

Orientation in Objects

```
public abstract class PersonBean implements EntityBean {

   public abstract long getId();
   public abstract void setId(long Id);
   public abstract String getName();
   public abstract void setName(String name);
   public abstract String getChristianName();
   public abstract void setChristianName(String cname);

   public abstract Collection getContacts();
   public abstract void setContacts(Collection contacts);

}
```

## Beziehungen - Referentielle Integrität

Orientation in Objects

- Der Container ist verantwortlich für die referentielle Integrität
- Dies gilt sowohl für 1-1 als auch für 1-n Relationen.
- Beispiel: Ein setXX-Aufruf bei einer 1-1-Relation kann 4 Objekte beeinflussen.
- Beispiel: Wenn man einer Collection bei einer 1-n-Relation mit add einen Wert hinzufügt, der schon Mitglied einer anderen Relation ist, wird er dort gelöscht.

## Agenda

Orientation in Objects

- EJB 2.1 Persistenz
  - Container Managed Persistence
  - Local Interfaces
  - Container Managed Relationships
  - EJB-QL
  - Auswirkungen aufs Design
- JDO Einführung
- Vergleich der beiden Modelle
- Architekturvarianten

## Abfragesprache EJB QL

Orientation in Objects

- DBMS-Unabhängigkeit durch eigene Abfragesprache
  - wird in ejb-jar.xml deklariert
- Finder werden portabel
- Teilmenge von SQL 92 mit Erweiterungen für die Navigation über Beziehungen

## Beispiele für EJB QL

Orientation in Objects

**Einfache Abfrage**

```
SELECT OBJECT(s)
FROM Seminar AS s
```

**Abfrage mit Parameter**

```
SELECT OBJECT(s)
FROM Seminar AS s
WHERE s.title = ?1
```

**Abfrage über Relationen hinweg:**

```
SELECT OBJECT(e)
FROM Seminar s, IN( s.events) e
WHERE e.place < ?1
```

## EJB-QL Funktionen

Orientation in Objects

- String Funktionen:
  - CONCAT(String, String) returns a String
  - SUBSTRING(String, start, length) returns a String
  - LOCATE(String, String [, start]) returns an int
  - LENGTH(String) returns an int
- Arithmetische Funktionen:
  - ABS(number) returns a number (int, float, or double)
  - SQRT(double) returns a double

## EJB 2.1 Aussichten Persistenz

Orientation in Objects

- ORDER BY

```
SELECT OBJECT(c) FROM Person p, IN(p.contacts) c WHERE
  p.name = 'kieninger' ORDER BY c.town
```

- Aggregate Operators
  - avg, count, max, min, sum

```
SELECT AVG(p.assessment) FROM EVENT e,
  IN(e.participations) p WHERE e.place = 'Mannheim'
```

- null value handling (und anderes bei queries) geklärt

```
SELECT c.phone FROM Person p, IN(p.contacts) c WHERE
  p.user.name = %1 AND c.email IS NOT NULL
```



## Finder vs. Select Methoden

Orientation in Objects

| | Finder methods | Select methods |
|---|---|---|
| **Methode** | find<Method> | select<Method> |
| **Sichtbarkeit** | Auch für Client zur Verfügung gestellt | Nur innerhalb Entity Bean Klasse |
| **Wird aufgerufen an** | Einer Bean Instanz im „pooled State“ | Gerade benutzter Instanz (pooled oder nicht) |
| **Rückgabewert** | EJBObjects oder EJBLocalObject **vom selben Typ wie die Beanklasse** | EJBObject, EJBLocalObject oder ein cmp-Feld |

## EJB QL im Deployment descriptor

Orientation in Objects

```
<query>
   <query-method>
        <method-name>findByBalance</method-name>
        <method-params>
              <method-param>long</method-param>
        </method-params>
   </query-method>
   <ejb-ql>
        SELECT OBJECT(a)
        FROM Account a
        WHERE a.balance = ?1
   </ejb-ql>
</query>
```

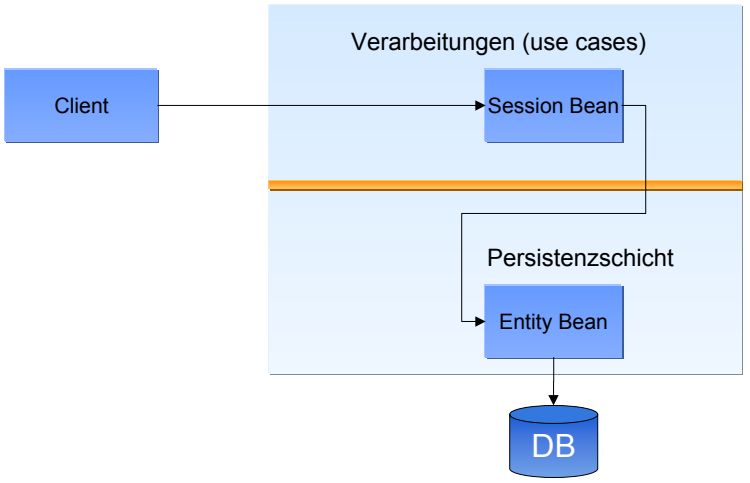


## Agenda

Orientation in Objects

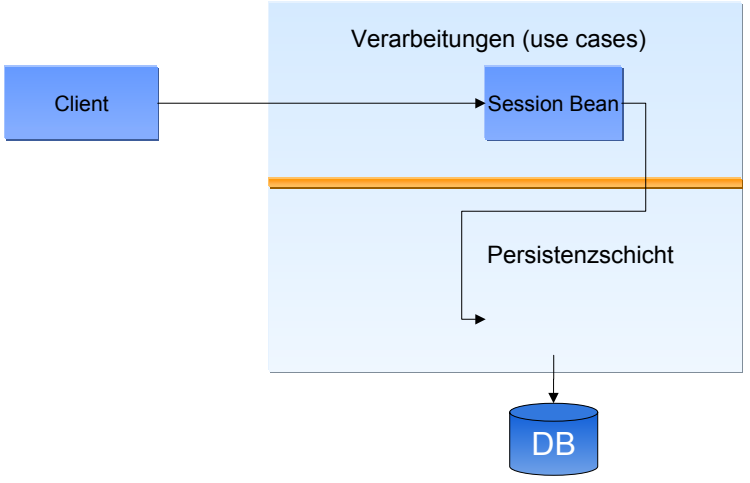
- EJB 2.1 Persistenz
  - Container Managed Persistence
  - Local Interfaces
  - Container Managed Relationships
  - EJB-QL
  - Auswirkungen aufs Design
- JDO Einführung
- Vergleich der beiden Modelle
- Architekturvarianten

## Naive Vorstellung - Facade



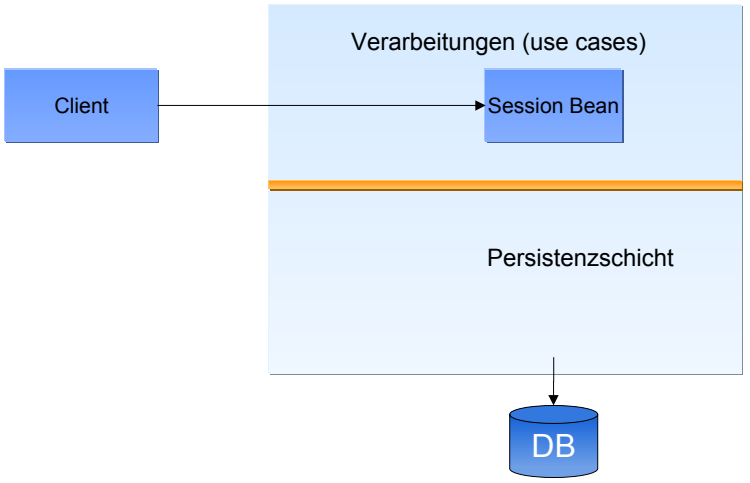

 

## Naive Vorstellung - Facade



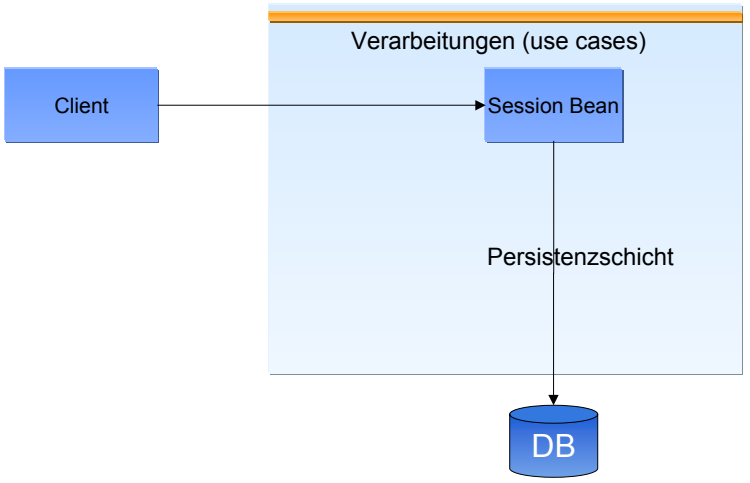

 

## Naive Vorstellung - Facade



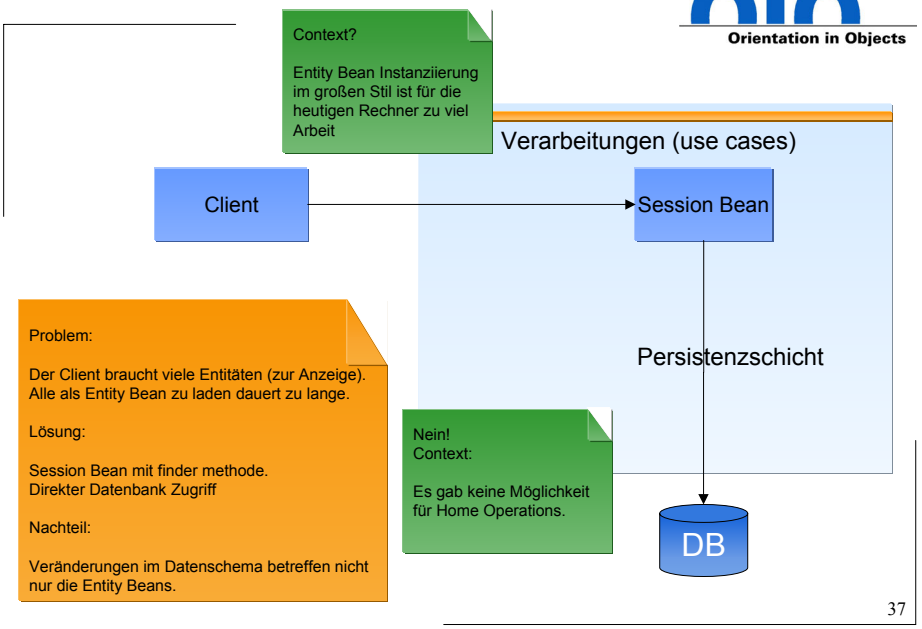

 

## Naive Vorstellung - Facade



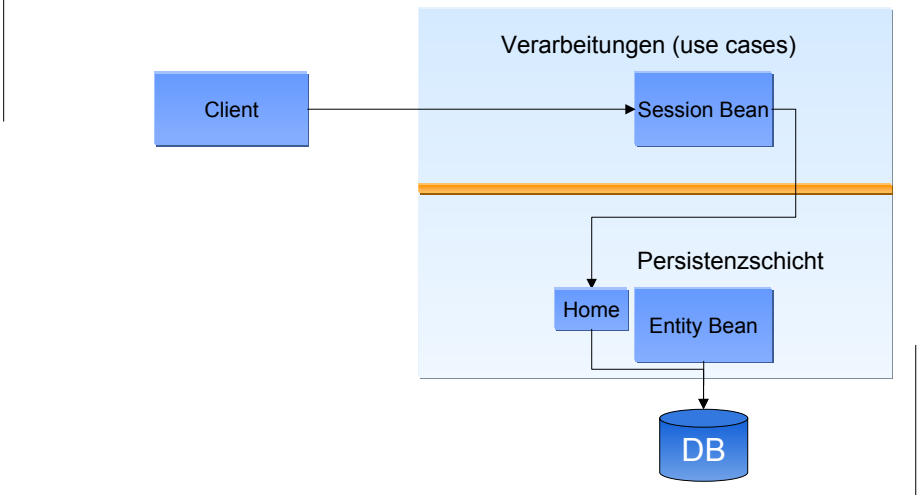

 

## Pattern: Bulk Reader

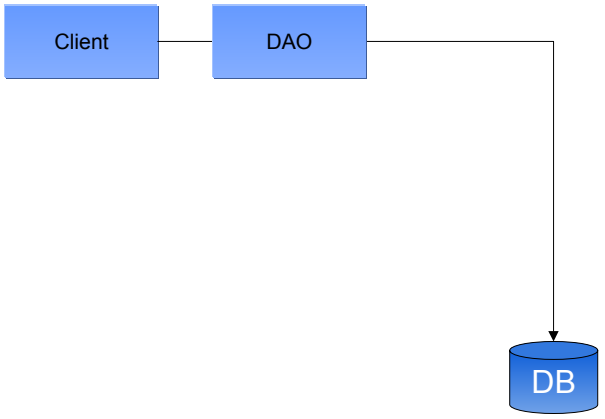





## Bulk Reader mit Home Operation



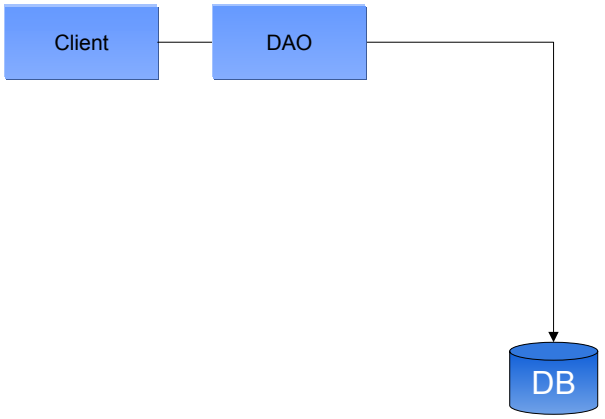
## Noch ein J2EE Pattern: Keine EJBs



## Fast Lane Reader

http://unimut.fsk.uni-heidelberg.de/unimut/schwob?schwob_url=http://www.oio.de



## Ankündigungsdaten J2EE 1.4 / EJB 2.1

Orientation in Objects

- EJB 2.1
  - Expert Group formation 23.10.2001
  - Community review 27.05.2002
  - Proposed Final Draft 22.08.2002
- **J2EE 1.4 - Proposed Final Draft 3** *April 15, 2003*
  - „J2EE 1.4 hätte eigentlich schon in der zweiten Jahreshälfte 2002 erscheinen sollen. Die erste Verschiebung hatte Sun mit ständigen Änderungen am SOAP-Standard durch das World Wide Web Consortium begründet. Der JCP wurde also gewissermaßen zweimal durch das bei der Veröffentlichung von 1.3 gegebene Versprechen ausgebremst, in der kommenden Server-Java-Version XML und Web-Services nativ zu unterstützen“.

http://www.computerwoche.de/index.cfm?pageid=254&artid=45517&type=detail

## Wünsche an EJB 2.2

Orientation in Objects

- Read-Only Entity Beans mit CMP
- Plug-In-Mechanismus für Persistence Manager
- Wege zur Behandlung großer ResultSets
  - Kein Iterator-Support
  - Keine Bulk-Manipulation
- Weitere Datenbankintegration
  - z.B. für Stored Procedures
- Dynamisches EJB-QL

## J2EE 1.3-Server 03.05.2002

Orientation in Objects

- Am 3.5.2002 schon verfügbar
  - Borland, Fujitsu, Macromedia, Pramati, SilverStream, Sun, Sybase, Trifork, BEA, CA, SAS
- Heute zusätzlich verfügbar
  - ATG, IBM, Iona, NEC, Novell, Oracle, Seebeyond, Spiritsoft, TMax, Trifork
- Bald
  - SAP Web Application Server 6.30

http://java.sun.com/j2ee/compatibility.html

## Agenda

Orientation in Objects

- EJB 2.1 Persistenz
  - Container Managed Persistence
  - Local Interfaces
  - Container Managed Relationships
  - EJB-QL
  - Auswirkungen aufs Design
- JDO Einführung
- Vergleich der beiden Modelle
- Architekturvarianten

## Wie kommt es zu JDO?

Orientation in Objects

- JDBC:

```
...
String sql = "";
int personID = generateID();
sql1 = "INSERT INTO PERSON (ID, NAME) VALUES (" + personID + ", '" +
   order.customer.name + "'" + ")";

sql2 = "INSERT INTO CUSTOMER (ID, PERSONID, CUSTOMERNO) VALUES (" +
   customerID + ", " + personID + ", '" + order.customerNo + "'" + ")";
statement.execute(sql);

sql3 = "INSERT INTO ORDER (ID, DOCUMENTID, " + "ORDERNO, SUBJECT,
   CUSTOMERID) VALUES (" + orderID + ", " + documentID + ", '" +
   contract.contractNo + "'" + ", '" + contract.subject + "'" + "," +
   customerID + ")";
...
```

- JDO:

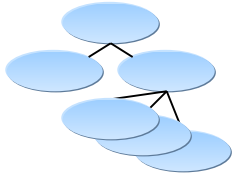
```
pm.makePersistent(order);
```

## Kurz und Knapp - Was ist JDO?

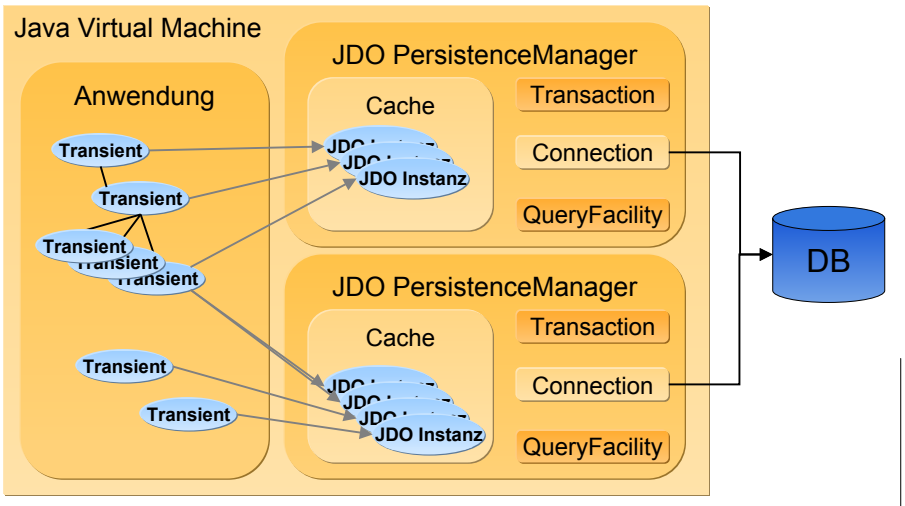
Orientation in Objects

- Standard für transparente Persistenz von Objektmodellen
- Abstrahiert von Datenspeichern (RDBMS, ODBMS, XML, ...)
- Teil von J2SE seit 30.04.2002 durch JSR 12
- Mögliche Integration in Application Server (Managed Environment)
  - Mittels JCA (J2EE Connector Architecture)



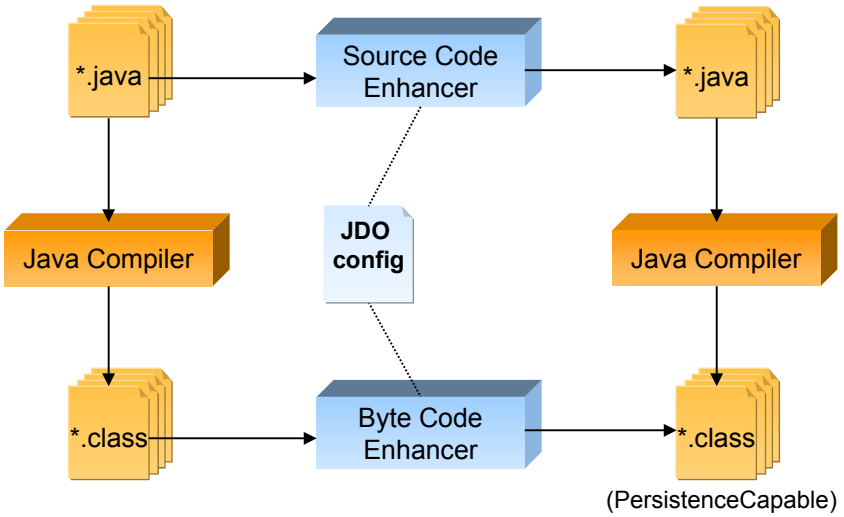
 

## JDO Architektur - non-managed






## Der Weg zum Persistenzglück



## JDO - Deskriptor / Mapping

Orientation in Objects

```
<?xml version="1.0"?>
<jdo>
  <package name="de.oio.vo">
    <class identity-type="datastore" name="Person">
      <field name="name">
       <extension key="type" value="java.lang.String"
         vendor-name="xyz"/>
       <extension key="column" value="NAME"
          vendor-name="xyz"/>
      </field>
    </class>
  </package>
</jdo>
```



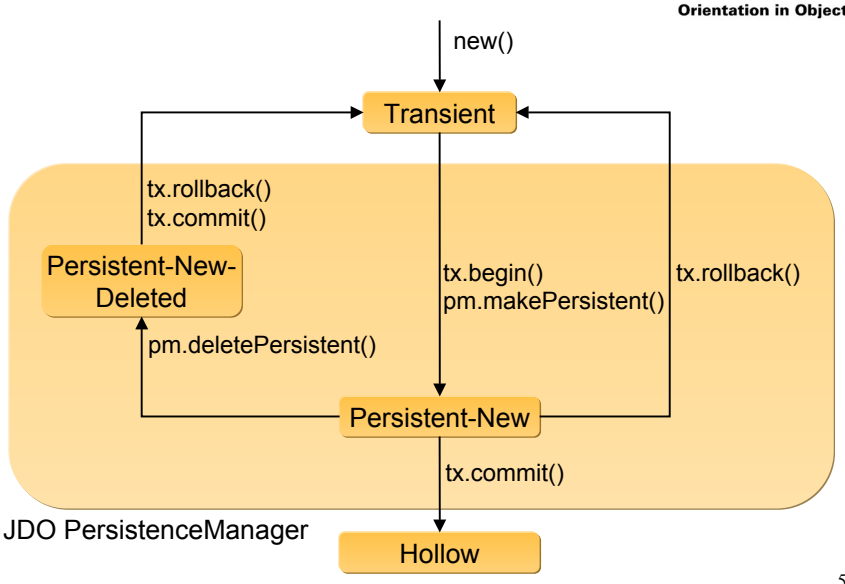
## JDO - HelloWorld

Orientation in Objects

```
...
PersistenceManagerFactory pmf;
pmf = JDOHelper.getPersistenceManagerFactory(...);
PersistenceManager pm = pmf.getPersistenceManager();
Transaction tx = pm.currentTransaction();

tx.begin();
Person p = new Person („Sohn“);
pm.makePersistent( p );
tx.commit();
...
```

## Einfaches Beispiel von Zuständen






## JDOQL - Java Data Objects Query Language

- DBMS-Unabhängigkeit durch eigene Abfragesprache
- Ermöglicht
  - Filter
  - Parameter
  - Variablen
  - Queries über bestehende Ergebnismengen
  - Ausdrücke

## JDOQL - Beispiel mit Parameter

Orientation in Objects

```
...
Query query = pm.newQuery(Person.class);
query.setFilter( "name == pname" );
query.declareParameters( "String pname" );
Collection result = (Collection) query.execute("Sohn");
Iterator it = result.iterator();
if ( it.hasNext() ) { person = (Person) it.next(); }
...
```

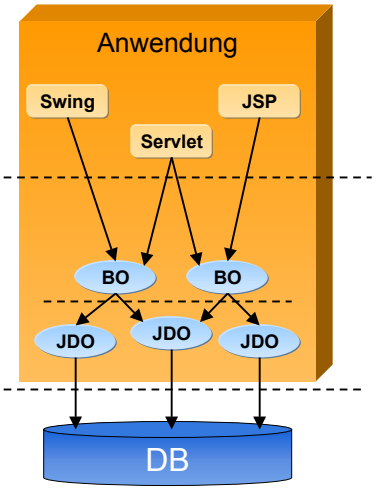
## Besondere Ausdrücke

Orientation in Objects

- Collections
  - `query.setFilter("projects.contains(proj) && " + "proj.budget > 10000000");`
  - `query.setFilter("projects.isEmpty(proj)");`
- startsWith() / endsWith()
  - `query.setFilter("name.startsWith(\"Dirk\")");`
  - `query.setFilter("name.endsWith(\"Sohn\")");`
- Extended JDOQL
  - Herstellerspezifisch

## Architektur (non-managed)

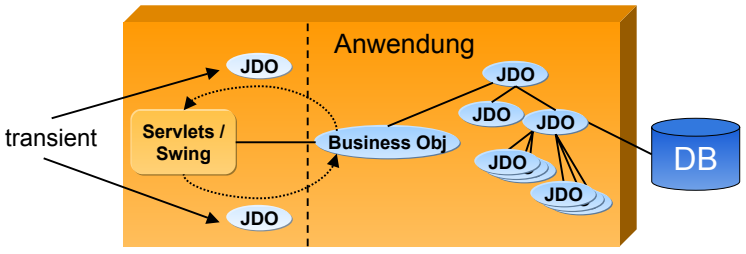



- N--Tier-Architektur ist gängig
- Abbildung der Logik erfolgt durch Business Objekte
- Persistenz erfolgt durch JDO
- Wie werden die Schichten getrennt?

## JDO braucht keine Value Objects? (I)

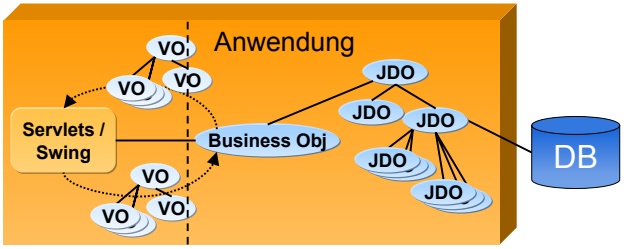
- Mit makeTransient() wird einzelnes Objekt Transient
- Ein „enhanced“ Objekt kann in „non-enhanced“ Objekt serialisiert werden
- Verlieren ihre Objektidentität
- Nur das einzelne Objekt wird Transient, nicht der Graph



## JDO braucht keine Value Objects? (II)

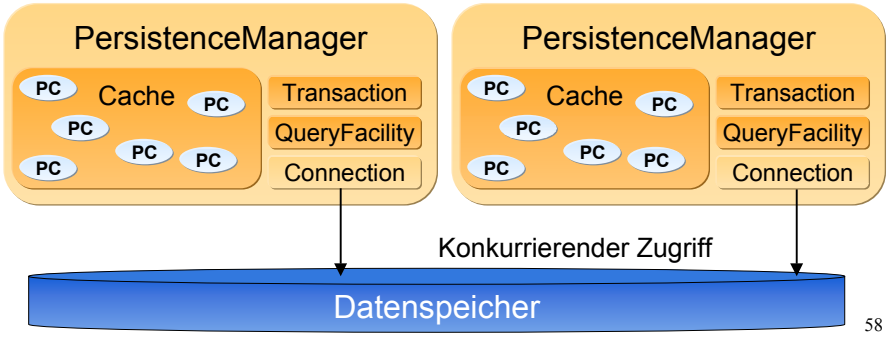
- Value Objects (VO) bzw. Data Transfer Objects (DTO)
- Datenkontainer ohne Logik
- Eventuell Value Object Assembler der Value Objects behandelt
- Entkopplung vom Objektmodell



## Concurrency - Konkurrierender Zugriff

- Es kann mehrere PMs im Speicher geben
- Objekte können mehrfach im Speicher auftauchen (ObjectId)
- JDO verwendet Concurrency Management von Impl.
  - ACID wird gewährleistet
  - Optimistic Locking ist optional



## Implementierungen

Orientation in Objects

- Anbieter (RDMS)
  - Kodo JDO (Solarmetrik)
  - JDO Genie (Hemisphere Technologies)
  - Jrelay (Object Industries)
  - IntelliBO (SignSoft)
- Datenbanken (ODBMS)
  - enJin (Versant)
  - FastObjects (Poet)
- Open Source
  - TriactiveJDO - TJDO (Sourceforge)

## Agenda

Orientation in Objects

- EJB 2.1 Persistenz
  - Container Managed Persistence
  - Local Interfaces
  - Container Managed Relationships
  - EJB-QL
  - Auswirkungen aufs Design
- JDO Einführung
- Vergleich der beiden Modelle
- Architekturvarianten

## JDO Probleme

Orientation in Objects

- Query-Interface
  - benutzt Strings zur Steuerung anstelle von z.B. Java-API
  - Aufwendiges Parsing nötig, viele Fehler nur zur Laufzeit erkennbar
- Enhancer
  - Kritik: Bytecode-Veränderungen sind böse
  - Antwort:
    - **Reflection als Alternative wäre langsam**
    - **Veränderung des Objektbaums sind auch nicht schön**
    - **Bytecode-Kompatibilität vorgeschrieben.**
  - Fazit: Gute Lösung

## JDO Probleme cont.

Orientation in Objects

- Sicherheitsmodell
  - Jeder braucht den Persistence Manager
  - Dieser steuert allerdings :
    - **Cache leeren**
    - **Transaktionen**
    - **Instanzen transient machen**
  - Es sollte nicht alles jedem zur Verfügung stehen.
- Keine gute Unterstützung von bidirektionalen Relationen

## Probleme von Entity Beans

Orientation in Objects

- Ohne local interfaces keine Unterstützung für Relationen
  - stimmt, entweder verteilte Komponenten oder CMR
- CMR bringt Performanceprobleme
  - in keinem Fall alle Relations (ER) oder Assoziationen (OO) auf CMR mappen
- Ein EJB-Modell ist weder streng relational noch objektorientiert
  - kein naives Mapping von OO- oder ER-Modellen auf Entity Beans

## Probleme von Entity Beans cont.

Orientation in Objects

- Graphen werden nicht automatisch persistent
- EJBs haben eine riesige Lernkurve.
  - Korrekt
- Semantische Unterschiede zwischen lokalen und remoten Aufrufen sind kritische Falle
  - defensive Copies

## Beide QL unreif!

Orientation in Objects

**EJB-QL**

- Keine dynamischen Abfragen
- Aggregatfunktionen
- Arbeitet mit java.util.Collection oder java.util.Set
- Keine subqueries

**JDOQL**

- Dynamische Abfragen
- Keine Aggregat-Funktionen in JDOQL
- Spezielle Collection für Lazy Loading und DB-Cursor
- Vorauswahl vorhanden

## Agenda

Orientation in Objects

- EJB 2.1 Persistenz
  - Container Managed Persistence
  - Local Interfaces
  - Container Managed Relationships
  - EJB-QL
  - Auswirkungen aufs Design
- JDO Einführung
- Vergleich der beiden Modelle
- Architekturvarianten

## Fowler on POJO

Orientation in Objects

- "I've come to the conclusion that people forget about regular Java objects because they haven't got a fancy name - so while preparing for a talk Rebecca Parsons, Josh Mackenzie and I gave them one: POJO (Plain Old Java Object). A POJO domain model is easier to put together, quick to build, can run and test outside of an EJB container, and isn't dependent on EJB (maybe that's why EJB vendors don't encourage you to use them.)"

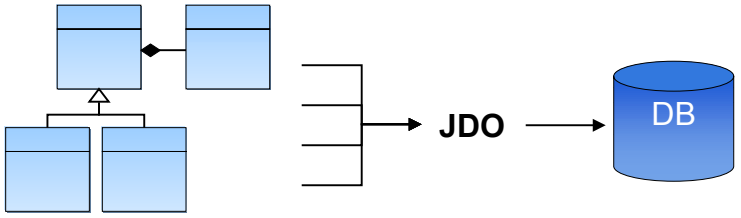
Martin Fowler, http://www.martinfowler.com/isa/domainModel.html

## POJO - Plain Old Java Objects

Orientation in Objects

**Objektorientiertes Modell (POJO)**



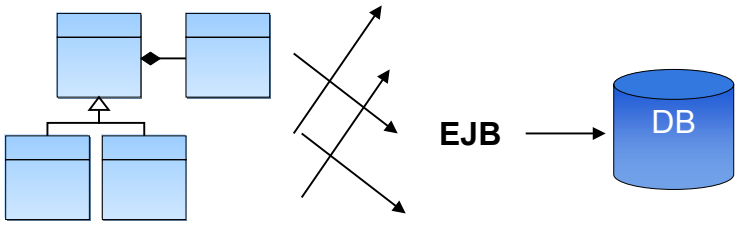

„Nette kleine Grüne-Wiese-Anwendung“

## POJO mit EJBs

Orientation in Objects

**Objektorientiertes Modell (POJO)**



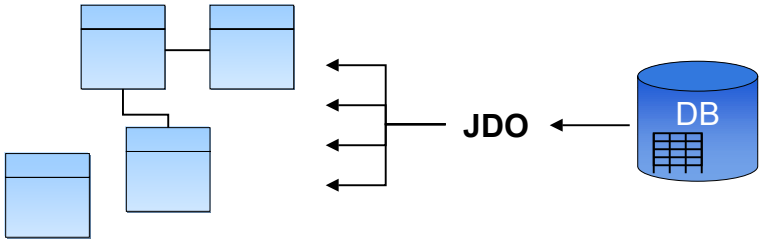

„Böse verteilte transaktionale Grüne-Wiese-Anwendung“



## Reverse Engineering

Orientation in Objects

**Objektbasierter Datenzugriff**

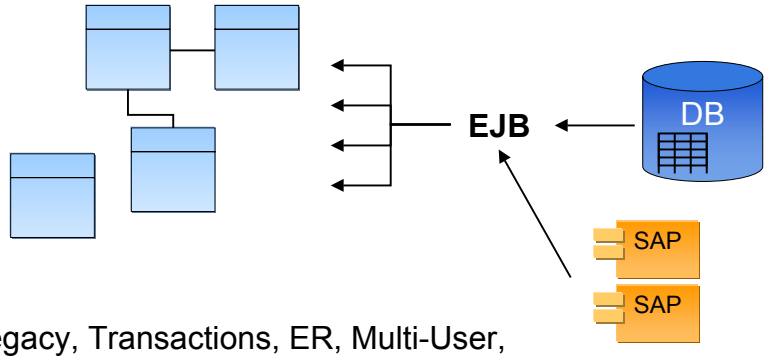



„Schnelles Reverse Engeneering eines ER-Modells“

## Reverse Engineering mit Services

Orientation in Objects

**Objektbasierter Datenzugriff**

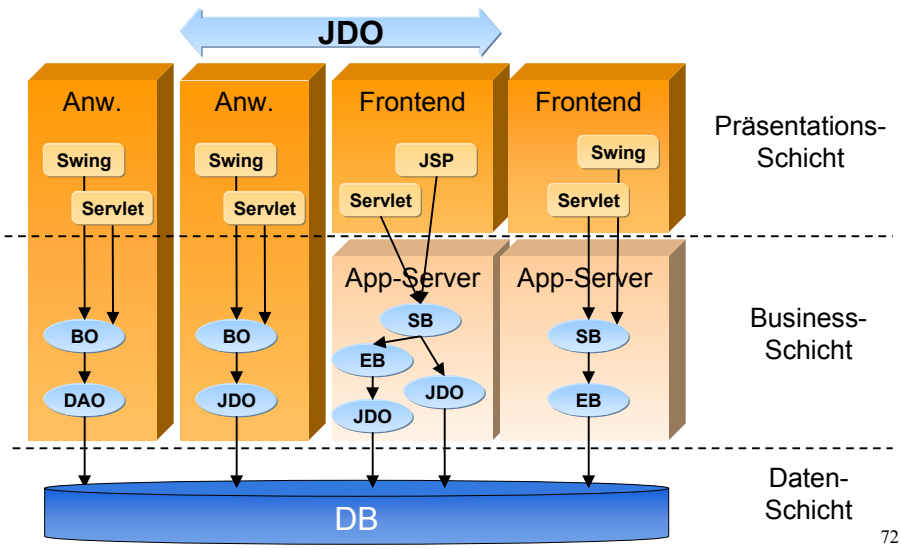



„Legacy, Transactions, ER, Multi-User,
Method based Security, distributed components“

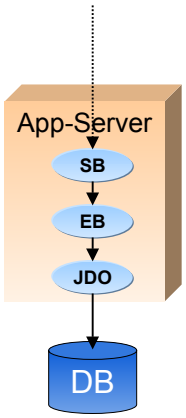


## JDO Architekturen

Orientation in Objects



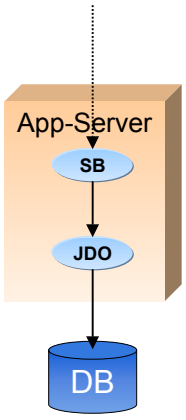
## JDO in Verbindung mit Entity Bean (BMP)




- Wenn CMP schlecht
- Für remotes JDO statt nur Session Facade
- Vorteile von JDO gehen verloren
  - Beziehungen zwischen Objekten
  - Vererbung
  - ...
- Oder für CMP durch Appserver-Hersteller

## JDO als Persistenz hinter SessionFassade




- Gut für JDO hinter EJB-Fassade
- JDO Vorteile bleiben nutzbar
- Features von Container nutzbar durch JCA
  - Externes Transactionmanagement
  - Externes Connectionmanagement
  - Security
- Leichtgewichtiges JDO hinter SessionBeans

## Web Konfusion - sind diese Aussagen richtig?

Orientation in Objects

- EJB stützt sich auf relationale Datenbanken, JDO auf ODBMS!
- Mit JDO kann ich eine Klasse auf zwei Tabellen mappen, mit EJB nicht!
- JDO kann keine Web-Services, EJB schon!
- Das default-mapping einer Bean-Instanz auf eine Tabellenzeile geht nicht weit genug bei CMP!

Orientation in Objects

# Vielen Dank für Ihre Aufmerksamkeit

**Dirk M. Sohn**
<sohn@oio.de>

**Tobias Kieninger**
<kieninger@oio.de>

**Orientation in Objects GmbH**
Weinheimer Str. 68
68309 Mannheim
http://www.oio.de
info@oio.de