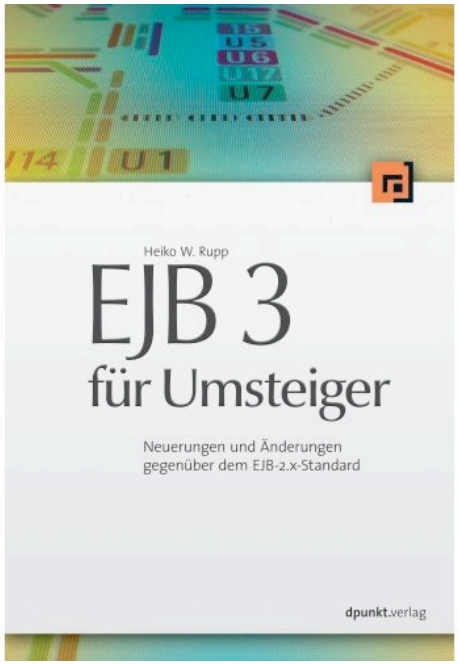
EJB 3 - Ein Blick über den Tellerrand



Heiko W. Rupp
<hwr@pilhuhn.de>

Agenda

- Abriss des Standards
- Blick auf vorhandene Implementierungen
- Erfahrungen aus der Praxis

- Verlosung der 2 Bücher

Agenda

- Abriss des Standards
- Blick auf vorhandene Implementierungen
- Erfahrungen aus der Praxis

- Verlosung der 2 Bücher

Abriss des Standards (1)

- Zurück zum POJO
  - Keine komplexen Interfaces
- Konvention statt Konfiguration
  - Sinnvolle Voreinstellungen
- Metadaten am Code

Abriss des Standards (2)

- Session Beans
  - @Stateful
  - @Stateless
- Message-Driven Beans
  - @MessageDriven
- Entity Beans
  - Neu als JPA

JPA

- Java Persistence API
  - Sehr ähnlich Hibernate / JDO
- Layer über Persistenz-Provider
- Ansprache über EntityManager
- Auch in Java SE nutzbar

Konsequenzen

- Schnellere Entwicklung
  - Weniger Code
  - Kein synchron-halten vieler Artefakte
- Schnelleres und besseres Testen
  - Entity Beans in UnitTests!

Konsequenzen

- Schnellere Entwicklung
  - Weniger Code
  - Kein synchron-halten vieler Artefakte
- Schnelleres und besseres Testen
  - Entity Beans in UnitTests!
  - Session Beans in UnitTests !!

Agenda

- Abriss des Standards
- Blick auf vorhandene Implementierungen
- Erfahrungen aus der Praxis

- Verlosung der 2 Bücher

Implementierungen

- Zu Unterscheiden:
  - EJB 3 vollständig
    - Java EE zertifiziert
    - (Noch) nicht zertifiziert
  - Nur JPA

Implementierungen

- Java EE zertifiziert
  - http://java.sun.com/javaee/overview/compatibility.jsp
- Bea Weblogic 10 Tech Preview
- Kingdee Apusic
- SAP Netweaver Java EE 5 Edition
- Sun Java Enterprise Server Platform Edition 9
- Tmax Soft JEUS 6

Implementierungen (2)

- Server, nicht zertifiziert
  - Red Hat JBoss AS
    - 4.0.5 + EJB 3
    - 5.0.0 Alpha
    - Embedded EJB 3
  - ObjectWeb Easy Beans
  - Sun Glassfish („Referenz“)

Implementierungen (3)

- Server ...
  - Oracle AS
  - Apache Open EJB
    - Basis für Geronimo Server
  - IBM Alpha Feature für WAS 6.1.0.3

JPA-Implementierungen

- JPA-Implementierungen
  - Oracle Toplink Essentials („Referenz“)
  - Hibernate + EntityManager + Annotations
  - Bea Kodo (früher Solarmetric)
  - Apache OpenJPA (OSS Kodo)
  - Apache Cayenne
  - Resin Amber

Agenda

- Abriss des Standards
- Blick auf vorhandene Implementierungen
- Erfahrungen aus der Praxis

- Verlosung der 2 Bücher

Naming

- Dependency Injection
  - Setter-Injection / Field-Injection

```
@EJB
SessionBeanLocal mySB;
```

```
SessionBeanLocal mySB;

@EJB public void setSessionBeanLocal
  (SessionBeanLocal local) {mySB = local;}
```

- Namen im JNDI weiter herstellerspezifisch
  - Teils anders für App. im JAR / EAR

PersistenceContext/-Unit

- PersistenceUnit
  - Sammlung von Entity Beans
- PersistenceContext
  - Wie Hibernate-Session
  - Von EntityManager verwaltet

```
@PersitenceContext
EntityManager em;
```

- Einzig notwendiger Deployment-Deskriptor:
  `persistence.xml`

PersistenceContext

- Szenario:
  - Entity Beans in einem Archiv
  - Nutzende Session Beans in anderem Archiv
- @PeristenceContext braucht Namen aus persistence.xml

```
@PersistenceContext(unitName = „myPU“)
EntityManager em;
```

Finder

- In EJB 2 am Home-Interface
  - „statische Methode“
  - Namenscheck zur Compile-Zeit
- In EJB 3
  - Methode in Session Bean notwendig
  - Deklaration am Entity Bean möglich
    - „Named Query“

Named Querys (1)

- Großer Vorteil:
  - Syntaxüberprüfung zur Deploy-Zeit
  - Bzw. bei Unit-Tests
  - Container kann sie vorkompilieren

Named Querys (2)

- Name muss innerhalb der PU eindeutig sein
  - Voranstellen AbstractSchema

```
@NamedQuery(
    name="Kunde.findByX",
    query="...")
```

  - Nutzung:

```
Query q = em.createNamedQuery
      („Kunde.findByX");
```

Named Querys (3)

- Besser als Konstante

```
@NamedQuery(name=Kunde.FIND_BY_X, ...)
@Entity
public class Kunde
{
  public static final String
  FIND_BY_X="Kunde.FIND_BY_NAME";
```

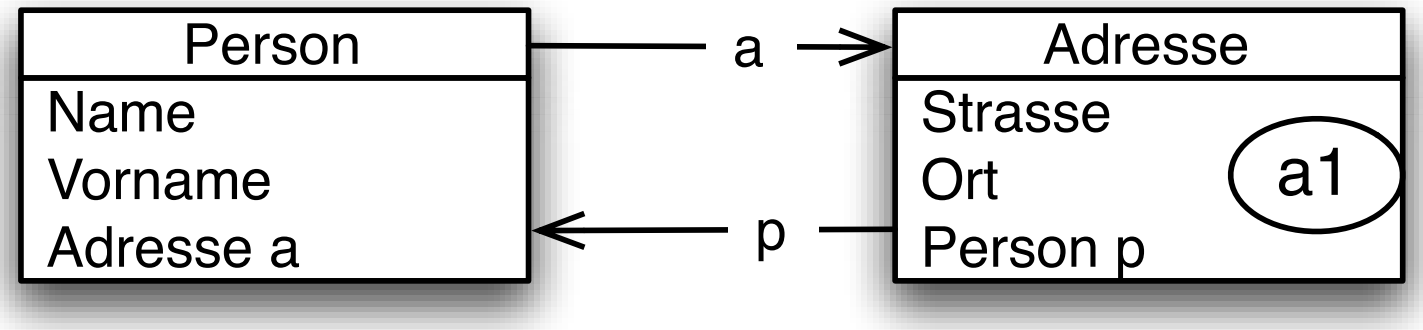
  - Weniger Fehler zur Laufzeit
  - Leichteres Auffinden der Query im Code

Zusammengesetzer PK

- Möglichkeit für zusammengesetzte PK
  - IdClass
  - EmbeddedID
- Beide benötigen extra Klasse für PK
- IdClass ist einfacher für existierende Finder
  - Query-Konversion via Perl-Skript möglich

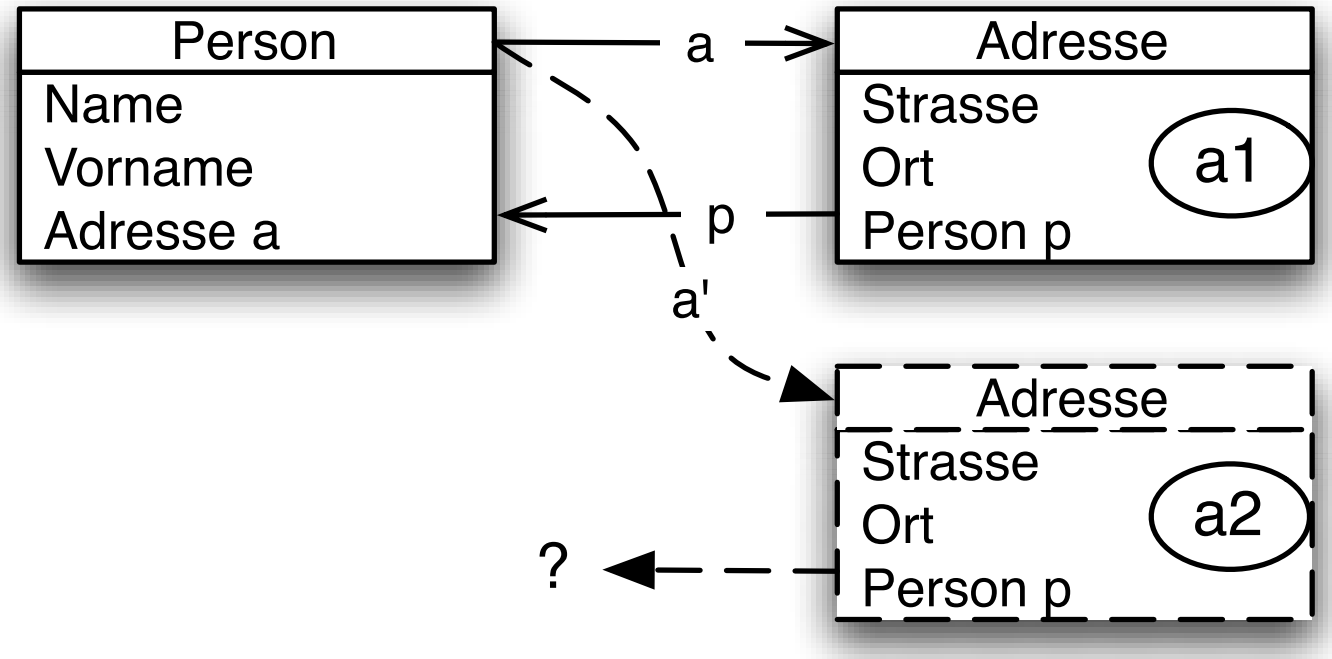
Relationen (1)

- CMP und CMR machen alles automatisch
- JPA Entity Beans sind POJOs
  - Relation durch zwei Felder gekennzeichnet



Relationen (2)

- Umhängen der Relation (neue Adresse)




```
Person.addAdresse(Adresse a) {
    this.a = a;
    a.setPerson(this);
}
```

Relationen (3)

- @OneToMany, @ManyToMany
  - Lazy-Loading
- @OneToOne, @ManyToOne
  - Eager-Loading

- Achtung: equals() und hashCode()

PersistenceContext Teil 2

- 1st-Level-Cache
- Identitätsgarantie
- Tracking von Änderungen
- Update in Batches
- Management des Lebenszyklus
  - Detached Objects

Detached Objects

- Entity Beans können zu Clients gesendet werden
  - VO / DTO müssen nicht mehr sein
  - Re-attach im Entity Manager

PC als Cache

- Cachen der Beans bis insert / update
  - viele Beans: Flush nötig
- Querys schreiben den Cache raus
  - Antipattern:

```
for (int i=0; i< 100000; i++) {
    XY = new XY(i);
    em.perist(i);
    Query q = ...;
    Object o = q.getSingleResult();
    XY.setZ(o);
}
```

PC und Batch-Updates

- Endlich Batch-Update und -Delete
- Laufen am PC vorbei
- Reihenfolge der Operationen ist wichtig

```
public void foo() {
  Query q = ... // Lese Kunden
  List<Kunde> kunden = q.getResultList();
  q = ... // Delete K from Kunde where k.Umsatz = 0
  q.executeUpdate();
  for (Kunde k: kunden) {
     // machwas mit dem Kunde
  }
```

Delegate

- EntityManager.getDelegate()
  - Unterliegender Peristenz-Provider
  - Implementierungsspezifisch
- Nutzung für Features, die JPA nicht bietet
  - findByExample / findByCriteria
  - Ermittlung der unterliegeden Hibernate-Query

Locking von Entity Beans

- EJB 2 CMP
  - Pesimistic Locking
- EJB 3
  - Optimistic Locking
  - Versionsspalte mit @Version
  - OptimisticLockException

Annotationen für EB

- Ort von @Id bestimmt Zugriff durch EM
  - Feld / Getter
  - Muss für die Hierarchie gleich sein
- Tools erwarten eher @Id am Feld

- „Reine Lehre“
  - @Id an Getter/Setter für Tests

Geschäftslogik

- Session Beans
- Message-Driven Beans

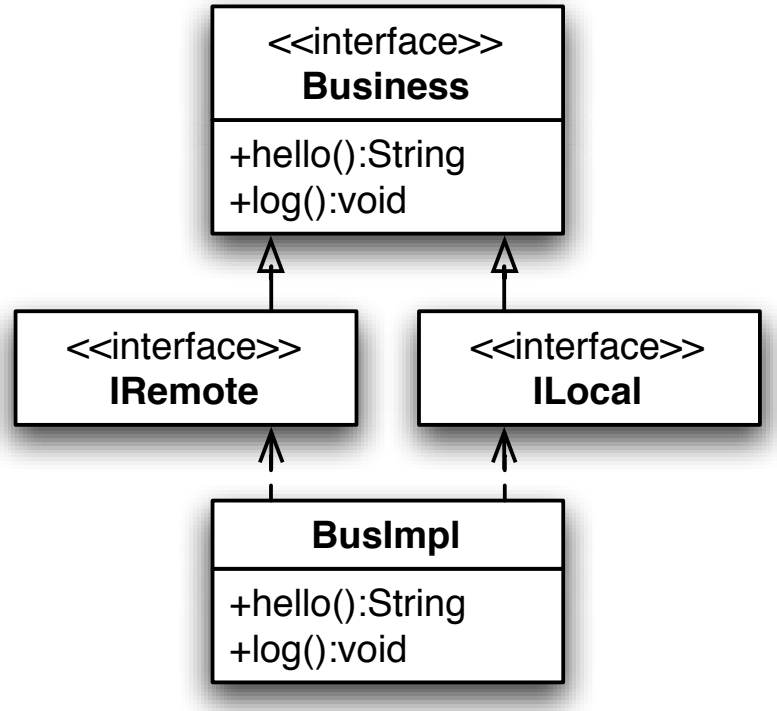
- Dependency Injection möglich
  - @EJB, @Resource, @PeristenceContext
- EJB Context hat lookup() Methode

Session Bean

- @Local als Default
- Kann nicht @Local und @Remote sein
- Muss von Local- und Remote-Interface erben

Session Bean

- @Local als Default
- Kann nicht @Local und @Remote sein
- Muss von Local- und Remote-Interface erben



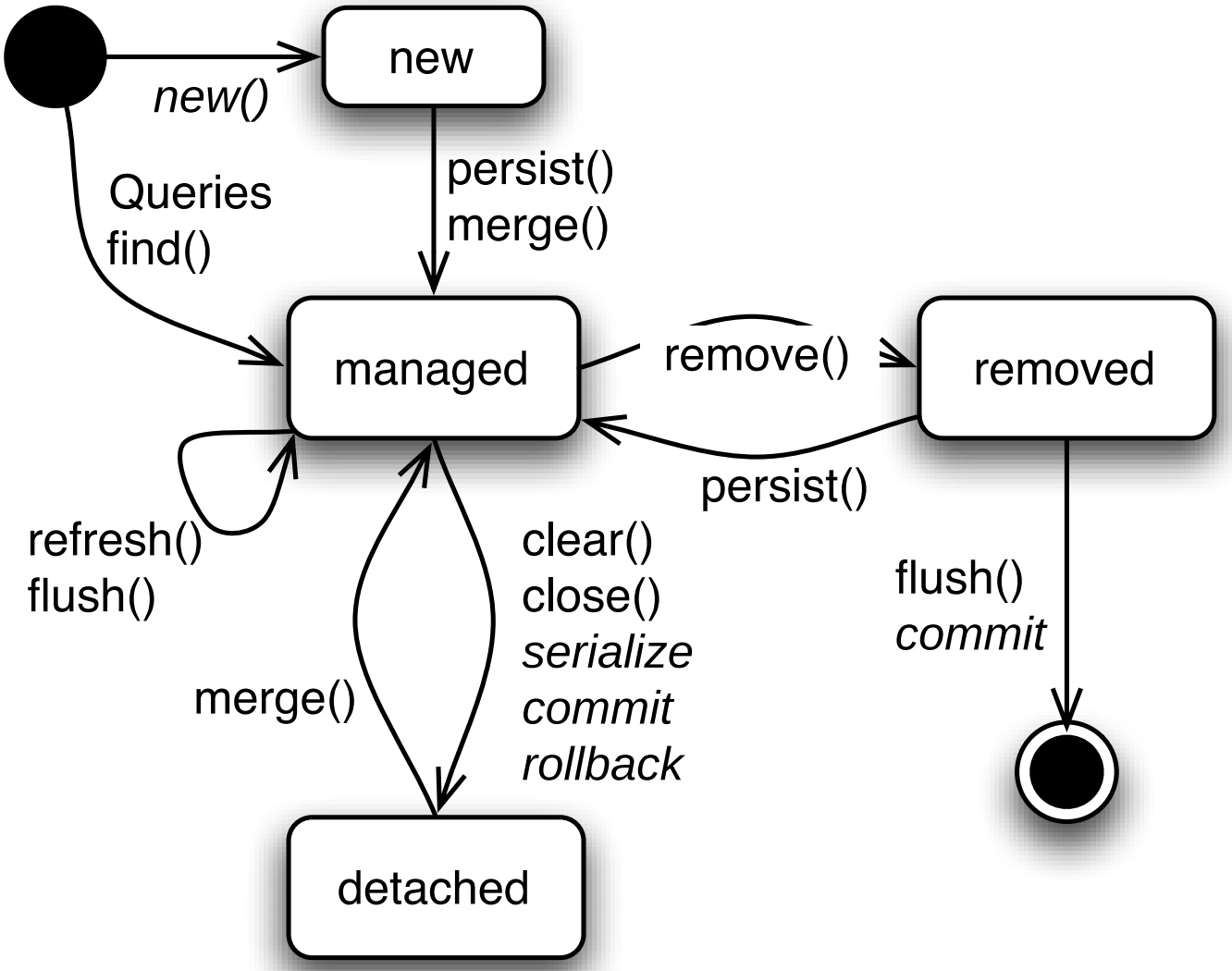
Session Bean Vererbung

- Session Beans können von anderen Session Beans erben
  - Nicht offiziell in der Spezifikation
  - Security / Tx / Interceptoren
    - Haben entsprechende Kommentare
- Beispiel:
  - Session Bean „international“
  - Kind: Session Bean pro Land

Ende ...

- Folien
  - http://bsd.de/e3fu/
- Beispiel-Implementierung
  - Weblog System
- Liste der Implementierungen
  - http://bsd.de/e3fu/referenzen.html

Entity Bean Lebenszyklus



equals() & hashCode()

- Entity Beans müssen diese Methoden implementieren
- Gute Kandidaten?
  - pk?
  - Relationen?
  - Business Keys?