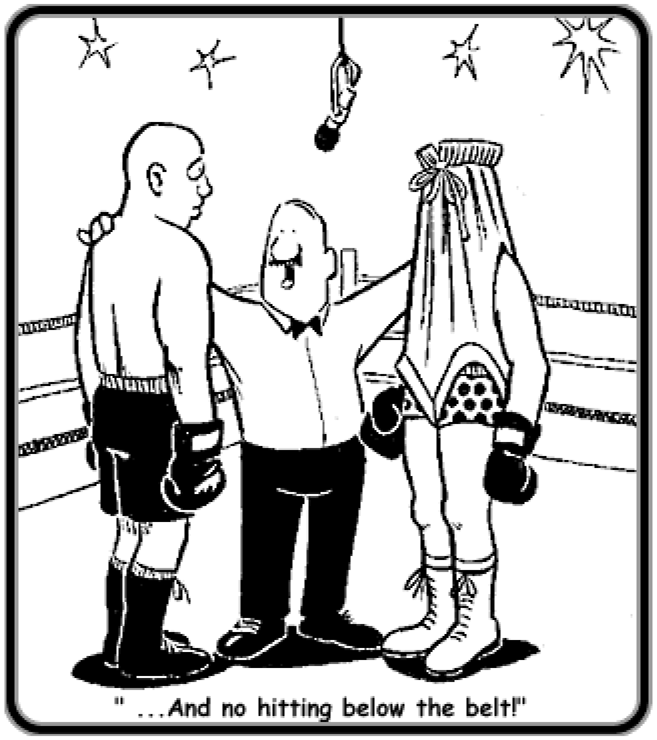
Spring and EJB 3: All-Star Team or Neighbors with Good Fences

Debu Panda
Author: *EJB 3 In Action*

EJB 3 and Spring



Java EE and frameworks: State of Nation

Dot Com Boom (J2EE 1.2/1.3)

- EJB darling of industry!
- Every application must have EJB ☺

Dot Com Bubble burst (J2EE 1.4)

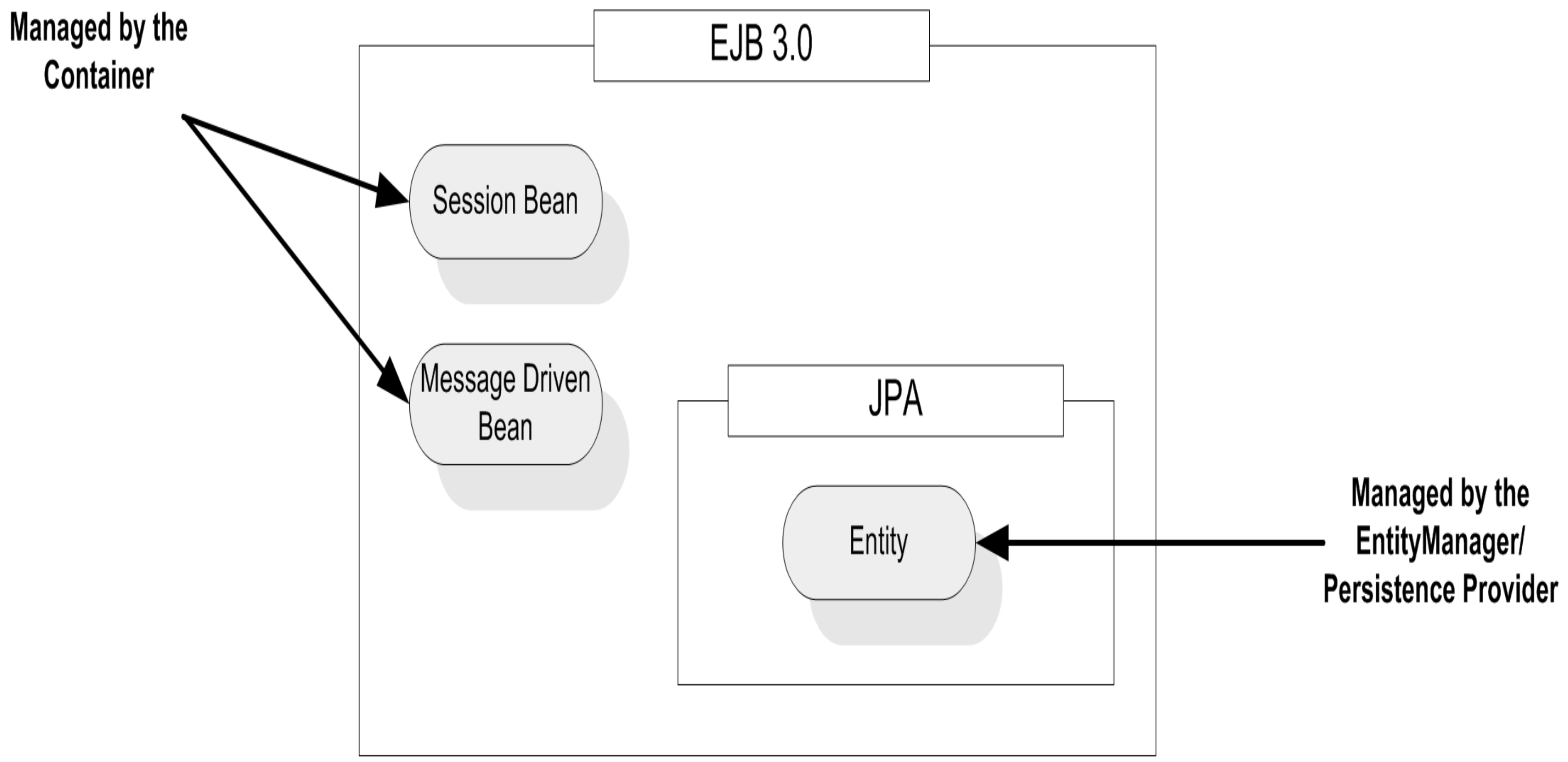
- Hangover of EJB overdose over
- Lightweight frameworks appeared to simplify complexities of enterprise Java!

Now! (Java EE 5)

- Lightweight frameworks rule the world!
- EJB reinvented as POJO!
- Scripting Languages and .Net challenge supremacy of Java
- Can EJB 3 be savior for Java EE!

ORACLE

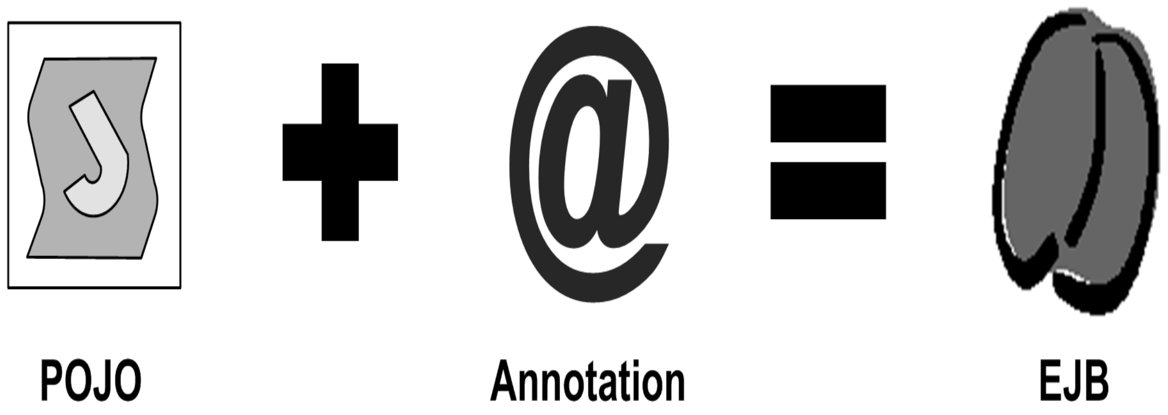
EJB 3 : Produces Two Specs



Simplified Development with EJB 3

- POJO (Plain Old Java Object) Class
  - EJB Class is a plain java class
- POJI (Plain Old Java interface)
  - Regular business interface
  - EJB interface does not have to implement EJB specific API
- No need of home interface
- Annotations for type of EJB and interface
- Deployment descriptor is optional

EJB 3 : Simplifying with annotations



@Stateless

@Stateful

@MessageDriven

EJB 3 Example

```
@Stateless
public class PlaceBidBean implements PlaceBid {
…

public Long addBid(Bid bid) {
                 ..
 }

}
```

Example Spring Bean

```
public class BidServiceBean implements BidService  {
..

public Long addBid(Bid bid){}
}
```

```
<bean id="bidService"
class="actionbazaar.buslogic.BidServiceBean">
    <property name="bidEAO">
      <ref bean="bidEAO"/>
    </property>
    …
  </bean>
```

EJB 3 and Spring Feature Sets

	EJB 3	Spring
Dependency Injection	Can inject anything in the container including EJBs, data sources, JMS resources, JPA resources and web services endpoints. Field and Setter No standard solution for injecting regular POJOs	Can inject almost anything including lists, maps, properties and JNDI resources Constructor and setter
Transaction management	Works right out of the box, but only JTA is supported	Have to configure it to make it work, but supports a number of strategies including JTA, JDBC and Hibernate
Persistence	Tightly integrated through JPA	Framework support for JPA, Hibernate, TopLink, JDBC, iBatis
State management	Robust support through Stateful Session Beans and Extended Persistence Context	Indirect support dependent on web container session management
Web services	Seamless support for JAX-WS 2.0	Poor direct support, best integration available is via configuring XFire for registered beans.
Messaging	Supported out of the box through Message Driven Beans.	Need to add configuration for each message listener. However, JMSTemplate and MessageListenerAdapter add nice abstraction.
Remoting	Integrated support through Session Bean remote interfaces. Supports distributed transactions and security.	Remoting support may be added via configuration. Remote transactions and security are not supported. However protocols other than RMI such as Hessian and Burlap are supported.
AOP	Simple but limited support through interceptors.	Robust support through AspectJ.
Security	Integrated support for declarative and programmatic security through JAAS.	Must add and configure Acegi security. However, support beyond JAAS is possible through Acegi.
Scheduling	Simple scheduling possible through EJB Timer service.	Must add and configure Quartz for scheduling.

EJB 3 with JPA

```
@Stateless
public class PlaceBidBean implements PlaceBid {
@PersistenceContext
      private EntityManager entityManager;

@TransactionAttribute(TransactionAttributeType.REQUIRED)
  public void addBid(Bid bid) {
      entityManager.persist(bid);
   }
}
@Local
public interface PlaceBid {
      void addBid(Bid bid);
}
```

JPA with Spring

- Integrated with JPA to use container-managed EntityManager
- Provides further simplification with JpaTemplate
- Ships TopLink Essentials as default JPA provider

```
public class BidServiceBean implements BidService  {
protected BidEAO bidEAO ;
public void set BidEAO(BidEAO bidEAO) {
    this.bidEAO = bidEAO;
}
public Long addBid(Bid bid){
  return this.bidEAO.saveBid(bid).getBidId();
}
```

```
public class BidSpringEAO extends BasicSpringEAO
implements BidEAO {
    public void persistBid(Bid bid) {
…
  this.getJpaTemplate().saveBid(bid);
}
```

Spring Configuration

```
<bean id="entityManager"
  class="org.springframework.jndi.JndiObjectFactoryBean">
    <property name="jndiName">
      <value>java:comp/env/actionBazaar</value>
  </property>
     ..
</bean>

 <bean id="bidService"
class="actionbazaar.buslogic.BidServiceBean">
    <property name="bidEAO">
      <ref bean="bidEAO"/>
    </property>
</bean>

 <bean id="bidEAO"
    class="actionbazaar.persistence.eao.BidSpringEAO"
    autowire="byType">
     <property name="entityManager" ref="entityManager"/>
</bean>
```

EJB 3 Transactions

- Natively integrated with JTA Transaction Manager
- Bean or Container managed

```
@Stateless
@TransactionManagement(TransactionManagementType.CONTAINER
)
public class PlaceBidBean implements PlaceBid {
..

@TransactionAttribute(TransactionAttributeType.REQUIRED)
  public void addBid(Bid bid) {
       entityManager.persist(bid);
   }
}
```

Transactions with Spring

- Proprietary integration with JTA Transaction Manager of application servers (Oracle, BEA, IBM)
- May use other local transaction managers

```
@Transactional(propagation=Propagation.REQUIRED)
public Long addBid(Bid bid){
..
}
```

```
<tx:annotation-driven/>
<bean id="transactionManager" class=
"org.springframework.transaction.jta.OC4JJtaTransactionManage
r">

</bean>
```

EJB 3 Web service

```
@Stateless
@WebService
public class PlaceBidBean implements PlaceBid {
@PersistenceContext
      private EntityManager entityManager;

@TransactionAttribute(TransactionAttributeType.REQUIRED)
@WebMethod
  public void addBid(Bid bid) {
      entityManager.persist(bid);
   }
}
```

ORACLE

Spring Configuration for a Web service

```
<bean
class="org.springframework.remoting.rmi.RmiServiceExporter">
    <property name="serviceName" value="placeBid"/>
    <property name="service" ref="placeBid"/>
    <property name="serviceInterface" value="PlaceBid"/>
    <property name="registryPort" value="1199"/>
</bean>

<bean id="placeBidService"
class="org.codehaus.xfire.spring.remoting.XFireExporter">
    <property name="serviceFactory"
ref="xfire.serviceFactory"/>
    <property name="xfire" ref="xfire"/>
    <property name="serviceBean" ref="placeBid"/>
    <property name="serviceClass" value="PlaceBid"/>
</bean>
..
```

Spring Configuration for a Web service

```
…

<bean
class="org.springframework.web.servlet.handler.SimpleUrlHand
lerMapping">
    <property name="urlMap">
        <map><entry key="/PlaceBidService">
<ref bean="placeBidService"/></entry>
        </map>
    </property>
</bean>
```

ORACLE

EJB 3 State management

```
@Stateful
public class BidderAccountCreatorBean implements
BidderAccountCreator {

@PersistenceContext(type=PersistenceContextType.EXTENDED)
private EntityManager entityManager;
    public void addLoginInfo(LoginInfo loginInfo) {
        entityManager.persist(loginInfo);
    }



    @Remove
    public void createAccount() {
        entityManager.flush();
    }
}
```

No state management support with Spring

EJB 3 Dependency Injection Examples

```
@EJB AdminService bean;
public void privilegedTask()
{
    bean.adminTask();
}
```

```
@Resource(name="myDB")
public void setDataSource(DataSource myDB) {
  customerDB = myDB;
}
```

```
@WebServiceRef(wsdlLocation=
"http://ejb3inaction.com/webservice/PlaceBidSe
rvice?WSDL")
private PlaceBidService bidservice;
```

Dependency Injection in Spring

```
<bean id="placeBid" class="PlaceBidBean">
   <property name="bidDao" ref="bidDao"/>
   <property name="dataSource">
        <jee:jndi-lookup jndi-name="jdbc/ActionBazaarDB"/>
    </property>
    <property name="concurrencySensitivity" value="1"/>
    <property name="adminEmails">
        <props>
            <prop key="administrator">
                administrator@somecompany.org
            </prop>
            <prop key="support">
                support@somecompany.org
            </prop>
      </props>
    </property>
</bean>
```

AOP in Spring

```
@Aspect
public class AuditAspect {
@Before("execution(public * *(..)) &&
@annotation(Auditable)")
    public void audit(JoinPoint joinPoint) {
         System.out.println("Entering: " + joinPoint);
         System.out.println("  with args: " +
joinPoint.getArgs());
    }
}

public class PlaceBidBean implements PlaceBid {
    ..

    @Auditable
    public void addBid(Bid bid) {
        sessionFactory.getCurrentSession().save(bid);
    }
}
```

Poor man's AOP in EJB 3: Interceptor

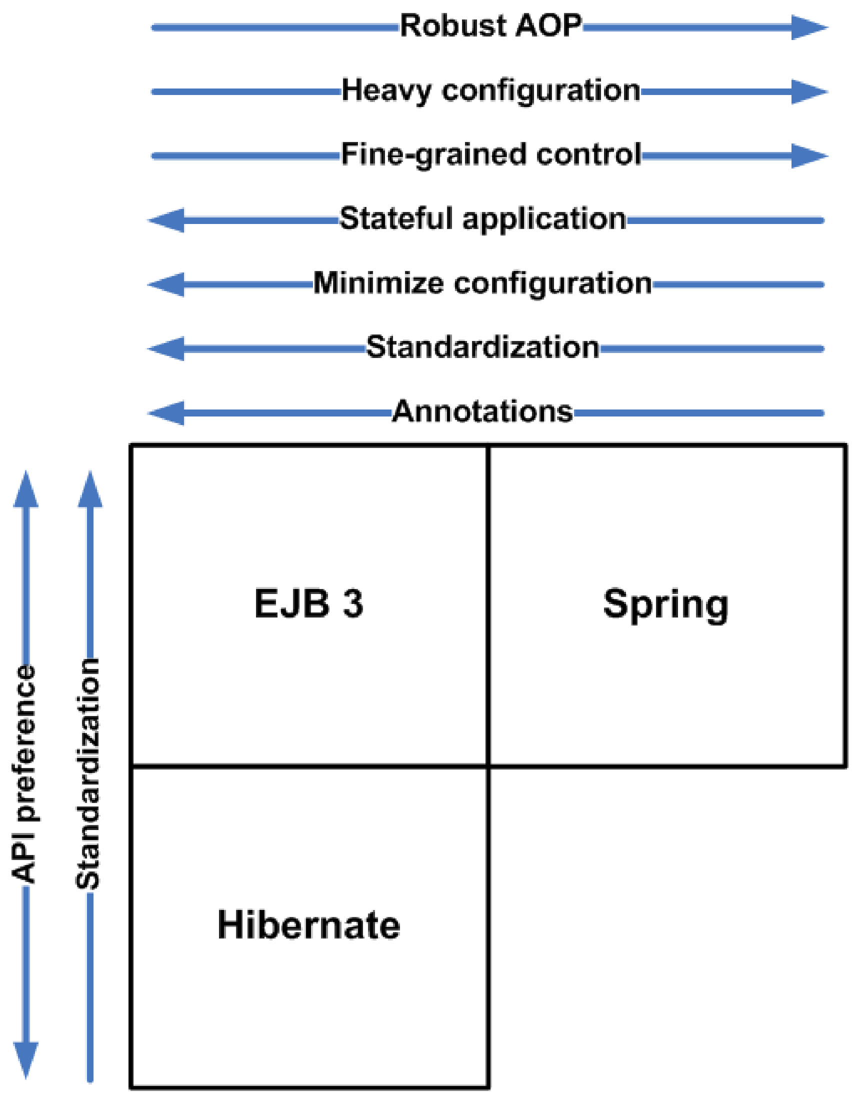
```
public class ProfilingInterceptor {
  @AroundInvoke // mark this method as a bean
   interceptor
    public Object checkPermission(InvocationContext
   ctx) throws Exception {
          System.out.println("*** checkPermission
   interceptor invoked");
         … }
}
@Stateless
@Interceptor({oracle.ejb30.ProfilingInterceptor.class})
public class PlaceBidBean implements PlaceBid {
}
```

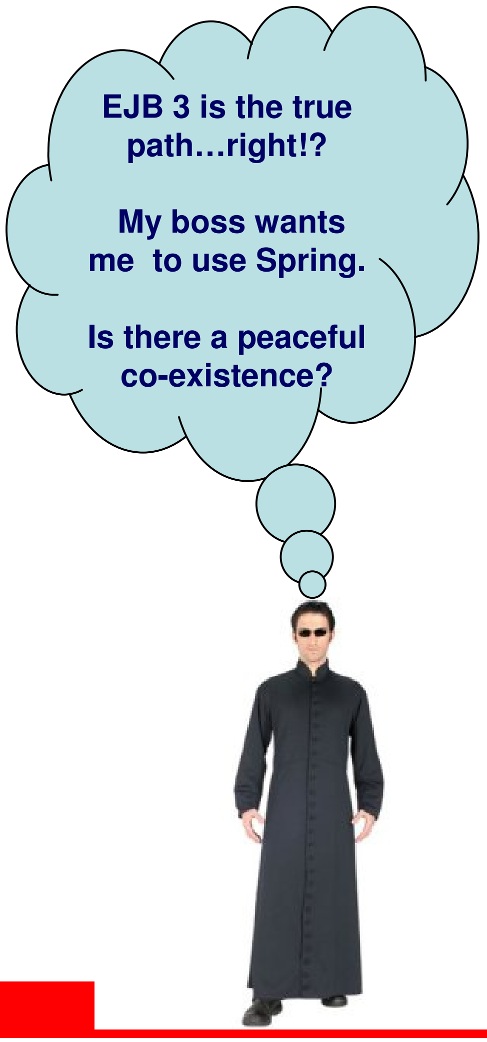
ORACLE

Review : EJB 3 and Spring Feature Sets

	EJB 3	Spring
Dependency Injection	Can inject anything in the container including EJBs, data sources, JMS resources, JPA resources and web services endpoints. Field and Setter No standard solution for injecting regular POJOs	Can inject almost anything including lists, maps, properties and JNDI resources Constructor and setter
Transaction management	Works right out of the box, but only JTA is supported	Have to configure it to make it work, but supports a number of strategies including JTA, JDBC and Hibernate
Persistence	Tightly integrated through JPA	Framework support for JPA, Hibernate, TopLink, JDBC, iBatis
State management	Robust support through Stateful Session Beans and Extended Persistence Context	Indirect support dependent on web container session management
Web services	Seamless support for JAX-WS 2.0	Poor direct support, best integration available is via configuring XFire for registered beans.
Messaging	Supported out of the box through Message Driven Beans.	Need to add configuration for each message listener. However, JMSTemplate and MessageListenerAdapter add nice abstraction.
Remoting	Integrated support through Session Bean remote interfaces. Supports distributed transactions and security.	Remoting support may be added via configuration. Remote transactions and security are not supported. However protocols other than RMI such as Hessian and Burlap are supported.
AOP	Simple but limited support through interceptors.	Robust support through AspectJ.
Security	Integrated support for declarative and programmatic security through JAAS.	Must add and configure Acegi security. However, support beyond JAAS is possible through Acegi.
Scheduling	Simple scheduling possible through EJB Timer service.	Must add and configure Quartz for scheduling.

The Matrix: EJB3, Spring and Hibernate

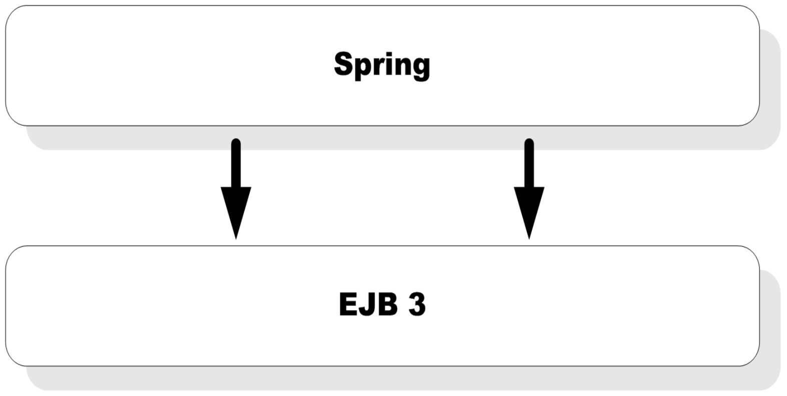


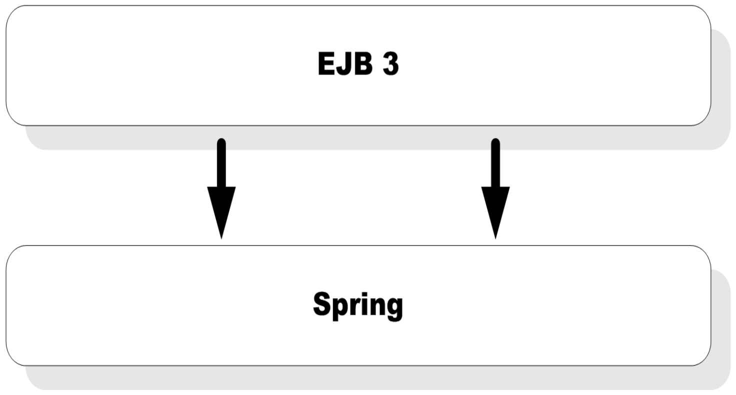

EJB 3 is the true path…right!?

My boss wants me to use Spring.

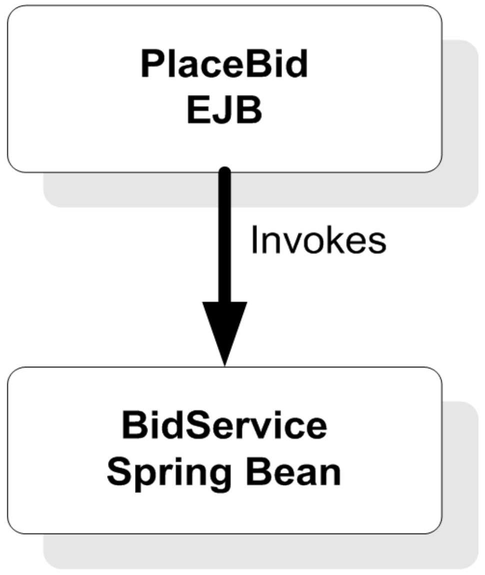
Is there a peaceful co-existence?

Spring and EJB 3 Integration



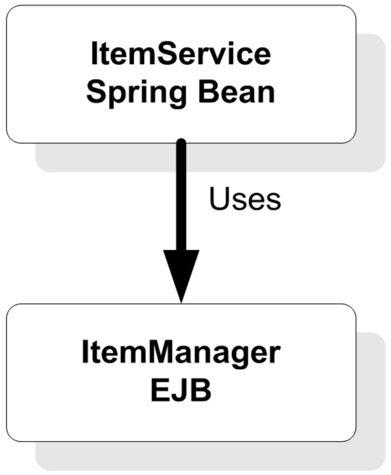


Spring-enabled Session Beans



```
@Stateless
public class PlaceBidBean extends
   AbstractStatelessSessionBean
                implements PlaceBid {

   private BidServiceBean bidService;

   protected void onEjbCreate() {
          bidService =
           (BidServiceBean)
            getBeanFactory()
            .getBean("bidService");
      }

      public Long addBid(Bid bid) {
          return bidService.addBid(bid);
      }
   }
```

Accessing EJB from Spring Beans



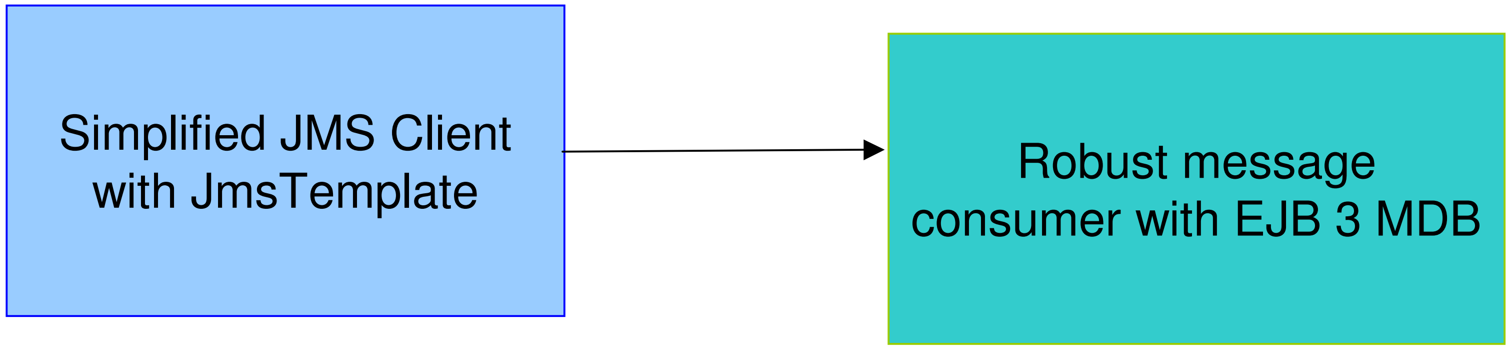
```
public class ItemServiceBean implements
   ItemService {
   // Setter injection of ItemManagerEJB
      public void setItemManager(
                 ItemManager itemManager) {
      this.itemManager = itemManager;
      }

      public Long addItem(Item item) {
      Item item =
          itemManager.addItem(item);
      ..}
   }
```

```
<jee:jndi-lookup id = "itemManager"
     jndi-name = "ejb/ItemManager"
   resource-ref = "true"/>
```

EJB 3 features in Spring

- Spring Pitchfork project provides partial support of EJB 3
- Supports EJB 3 Annotations
  - Annotations such as @Stateless, @Resource, @Interceptors are used

MDB and Spring JmsTemplate


Simplified JMS Client with JmsTemplate
Robust message consumer with EJB 3 MDB

EJB 3 and Spring : The Bottom Line

- **Use EJB 3 if you:**
    - Like annotations and dislike XML
    - Prefer a tightly integrated solution stack that makes sensible default choices for you and minimizes configuration.
    - Your application is very stateful.
    - Standardization is an important consideration.
    - JSF and are considering frameworks such as Oracle ADF, JBoss Seam.
- **Use Spring if you:**
    - Want your application to be portable across platforms not supporting EJB 3
    - Want to build your own solution stack (such as with iBATIS, Quartz or Acegi).
    - Need advanced AOP features.
    - Your application requires a lot of configuration beyond gluing together components and resources.

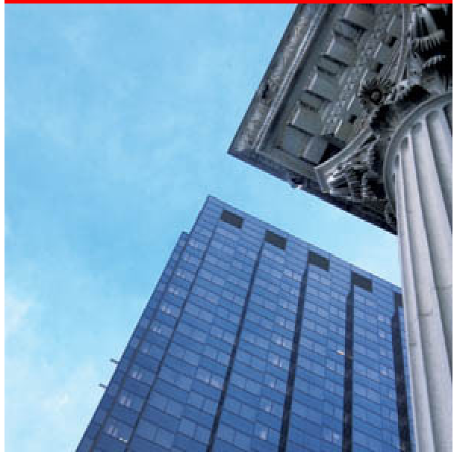
Summary

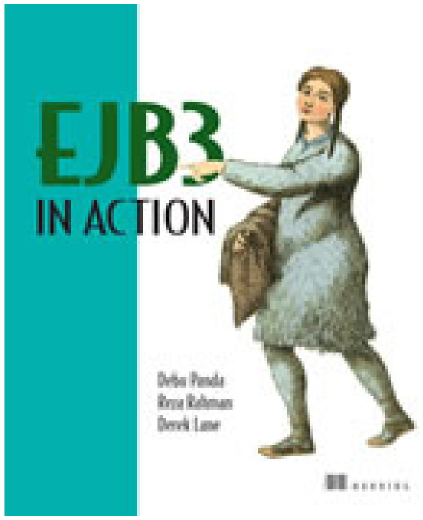
- Both EJB 3 and Spring can be used complimentary technology together

Resources

- Spring and Java EE : Simplicity and Power Combined (JDJ articles http://java.sys-con.com/read/393298.htm and http://**java**.sys-con.com/read/366297.htm)
- Spring Framework (http://springframework.org/)
- OTN Spring Resource Center: http://otn.oracle.com/spring/
- EJB 3 In Action (http://manning.com/panda)
- Spring in Action

Shameless Marketing plug





http://manning.com/panda

http://debupanda.com

ORACLE