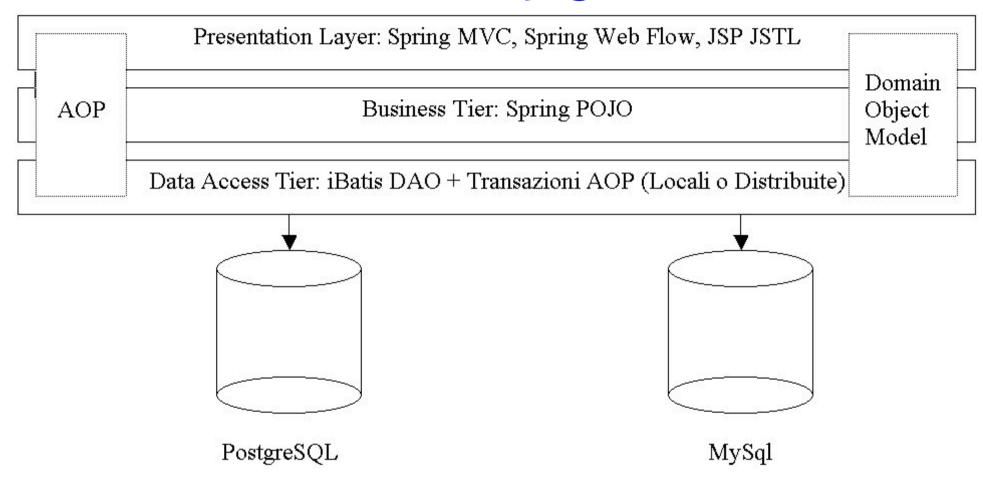
# Spring, iBATIS e Transazioni AOP nel Jug Avis Web

Massimiliano Dessì Frame S.r.l http://www.jugsardegna.org/vqwiki/jsp/Wiki?MassimilianoDessi

# **Architettura Generale Jug Avis Web:**



Interamente Interface Driven Design sviluppato usando TDD (Test Driven Development)



# Soluzioni prese in considerazione per l'accesso a DB

JDBC Hibernate iBatis



# JDBC:

### Pro

- Ben Conosciuta, possibilità controllo fine sul Codice e su SQL

### **Contro**

- -Eccessiva Verbosità
- -Manutenzione con query complesse composte dinamicamente
- -Codice "boilerplate" con try-catch obbligatori
- -La classe DAO "deve" sapere di essere eseguita dentro una transazione e deve essere scritto di conseguenza

# **Hibernate:**

### Pro

- -ORM, ragionamento OO-LIKE, non SQL-LIKE
- -Tools per velocizzare generazione codice e files
- -Adatto con DB modificati spesso
- -Caching

### **Contro**

- -Non SQL-LIKE, no DBA Tuning
- -Necessario per il web l' uso del pattern Open Session in View
- -N+1 Select
- -Esperienza per controllo fine settaggi configurazione

Licenza: GPL



# iBatis:

### Pro

- -Data Mapper, SQL separato dal codice, DBA Tuning
- -Semplicissimo da usare
- -Caching
- -Velocità rispetto ad ORM
- -Ora disponibile tool per generazione codice (Abator)
- -N+1 select evitabile scrivendo opportunamente le query

### **Contro**

Non è ancora "famoso" e conosciuto

Licenza: Apache

# iBatis configurazione con Spring

```
<bean id="jug4avisweb.sqlMapClient"</pre>
       class="org.springframework.orm.ibatis.SqlMapClientFactoryBean">
       property name="configLocation">
           <value>WEB-INF/jug-sqlMapConfig.xml</value>
       </property>
</bean>
<bean id="jug4avisweb.dataSource"</pre>
       class="org.springframework.jndi.JndiObjectFactoryBean">
       property name="jndiName">
          <value>java:comp/env/jdbc/jug4avis</value>
       </property>
</bean>
<bean id="transactionManager"</pre>
       class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
      cproperty name="dataSource">
</bean>
```

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE sqlMapConfig</pre>
    PUBLIC "-//iBATIS.com//DTD SQL Map Config 2.0//EN"
    "http://www.ibatis.com/dtd/sql-map-config-2.dtd">
<sqlMapConfiq>
<settings
cacheModelsEnabled="true"
enhancementEnabled="true"
lazyLoadingEnabled="false"
maxRequests="64"
maxSessions="20"
maxTransactions="8"
useStatementNamespaces="false"
/>
    <sqlMap
resource="org/jugsardegna/avis/web/db/ibatis/CentroAvisDaoPostgres.xml"/>
    <sqlMap
resource="org/jugsardegna/avis/web/db/ibatis/DonorDaoPostgres.xml"/>
</sqlMapConfig>
```

### File di configurazione jug4avisweb-sqlMapConfig.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE sqlMap PUBLIC "-//iBATIS.com//DTD SQL Map 2.0//EN"</pre>
"http://www.ibatis.com/dtd/sql-map-2.dtd">
<sqlMap>
   <cacheModel id="file-cache" type ="LRU" >
       <flushInterval hours="24"/>
       <flushOnExecute statement="insertCentro"/>
       <flushOnExecute statement="deleteCentro"/>
       <flushOnExecute statement="aggiornaCentro"/>
       cproperty name="cache-size" value="1000" />
   </cacheModel>
  <typeAlias type="org.jugsardegna.avis.web.domain.CentroAvis" alias="centroAvis"/</pre>
    <resultMap class="centroAvis" id="resultCentroAvis">
       <result property="id" column="id"/>
       <result property="nome" column="nome"/>
        <result property="fax" column="fax"/>
        <result property="telefono" column="telefono"/>
        <result property="email" column="email"/>
        <result property="sitoWeb" column="sito web"/>
        <result property="localita" column="localita"/>
        <result property="creditoResiduoSms" column="credito residuo sms"/>
        <result property="numeroDonatori" column="numero donatori"/>
    </resultMap>
```

```
<select id="getCentroAvis" resultMap="resultCentroAvis">
    SELECT * FROM centri avis WHERE id= #value#;
  </select>
  <delete id="deleteCentro" parameterClass="long">
         DELETE FROM centri avis WHERE id=#value#;
  </delete>
    <insert id="insertCentro" parameterClass="centroAvis">
         <selectKev kevProperty="id" resultClass="long">
              select nextval('id univoco')
         </selectKey>
         INSERT INTO centri avis (id, nome, localita, telefono, email, fax, sito web,
         credito residuo sms, numero donatori)
         VALUES (#id#, #nome#, #localita#, #telefono#, #email#, #fax#, #sitoWeb#, #creditoResiduoSms#,
         #numeroDonatori#);
    </insert>
    <update id="aggiornaCentro" parameterClass="centroAvis">
         UPDATE centri avis SET nome = #nome#, telefono = #telefono#, email = #email#,
         fax =#fax#, sito web = #sitoWeb#, localita = #localita#, credito residuo sms = #creditoResiduoSms#,
         numero donatori = #numeroDonatori#
         WHERE Id = \#id\#;
    </update>
</sqlMap>
```

File di configurazione CentroAvisDaoPostgres.xml

# iBatis DAO (1)

### **SqlMapCentroAvisDao**

# iBatis DAO (2)



- I dao non sanno minimamente di essere dentro una transazione o meno
- non siamo obbligati a gestire try-catch (naturalmente in una altra parte della applicazione, verranno tracciate e gestite convenientemente in caso vengano lanciate eccezioni)
- Le query sono separate e facili da modificare



# Transazioni (obiettivo)

Vogliamo fare in modo che alcuni metodi vengano eseguiti come unità atomiche di lavoro:

- Senza modiche nel codice in maniera che siano metodi riutilizzabili
- Senza scrivere codice,
   ma dichiarando (Declarative Transaction Demarcation)
   semplicemente quali metodi e
   con quali livelli di scope

Nota: Le transazioni di cui necessitiamo sono su un datasource locale, non distribuito.

Per realizzare i nostri obiettivi usiamo il supporto per utilizzare l'AOP (Aspect Oriented Programming) per mezzo di Spring.

Con Aspect

con classi Java che fungano da Proxy

In questa brevissima presentazione vedremo il secondo caso, di più facile comprensione.

## Nota:

AspectJ verrà usato nel Jug Avis Web con l'uscita della versione 2.0 di Spring nella primavera 2006

# **Brevissima spiegazione sull' AOP:**

- -Separazione delle competenze trasversali
- -Complementare all' 00

es.

Nelle classi, dove è necessario il logging, ad esempio il tracciamento delle eccezioni, viene richiamato un Logger (staticamente o con l' IOC), e viene eseguito il tracciamento

Con la programmazione ad Aspetti scriviamo solo una volta il modulo (Aspect) e nella sua dichiarazione diciamo quando deve essere eseguito.

Se usiamo AspectJ, durante la "compilazione" (weaving), l'aspect weaver (compilatore compone il sistema finale legando i nostri moduli trasversali (aspects).

Naturalmente questo processo produce bytecode standard per la virtual machine.

Il logger sparisce dalle classi dove lo avremmo dovuto mettere per poterlo usare con il solo uso dell' 00.

Possiamo anzi, creare anche un aspect che gestisca il lancio delle eccezioni!

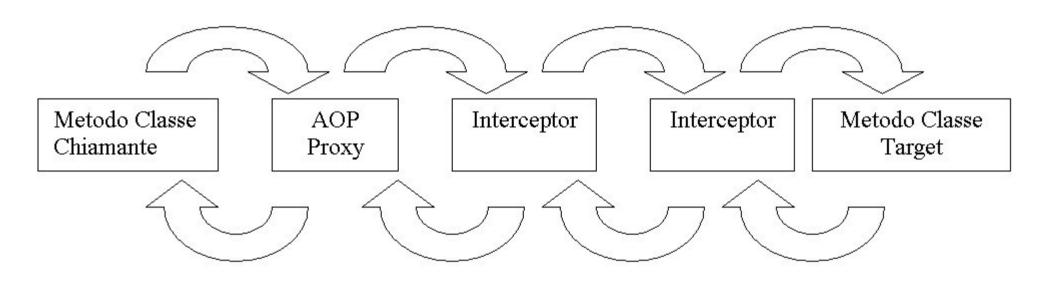
Grazie all' AOP possiamo modularizzare le competenze trasversali che altrimenti con la programmazione ad Oggetti "tradizionale" sarebbero presenti in tutte le classi in cui sono necessarie.

Vediamo invece ora come è possibile "emulare" il funzionamento di AspectJ, con delle classi proxy usando il pattern Proxy e Decorator.

Naturalmente il funzionamento con le classi proxy, è conforme alle interfacce della AOP Alliance, per poter essere interoperabile tra le varie implementazioni AOP.

Il funzionamento con i Proxy di SpringAOP, ruota attorno all' uso di una catena di intercettori tra l'oggetto chiamante e quello invocato. Ogni elemento della catena ha la possibilità di interporre logica prima della invocazione, dopo l'invocazione, o circuitare anche l' invocazione.





Ora che abbiamo spiegato in maniera molto veloce lo scopo dell' AOP, vediamo come possiamo usarlo nel JUG Avis per le Transazioni.

Prendiamo come esempio la classe ManagerCentriAvis, dove, quando un centro deve essere eliminato, deve eliminare nella stessa transazione anche i donatori di quel centro.

Perciò questa classe sarà la classe target.

Come Proxy usiamo un classe predefinita fornita da Spring:

**TransactionProxyFactoryBean** (AOP Proxy di pag. 18) che per fare il suo lavoro ha bisogno dell' iniezione di

- -TransactionManager
- -Target (ManagerCentriAvis)
- -TransactionAttributes (nomi o pattern ai quali applicare la transazionalità)

Nella parte dei TransactionAttributes possiamo definire la propagazione della transazione, il livello di isolamento, il timeout e il flag readOnly.

Nota:

Se avessimo bisogno di dichiarare più manager useremmo una altra tecnica.

```
<bean id="jug4avisweb.managerCentriAvisTarget"</pre>
class="org.jugsardegna.avis.web.service.impl.ManagerCentriAvis">
cproperty name="centroAvisDao"><ref bean="jug4avisweb.centroAvisDao"/>
</bean>
<bean id="jug4avisweb.managerCentriAvis"</pre>
class="org.springframework.transaction.interceptor.TransactionProxyFactoryBean">
       property name="transactionManager">
           <ref bean="transactionManager"/>
       </property>
       property name="target">
           <ref bean="jug4avisweb.managerCentriAvisTarget"/>
       property name="transactionAttributes">
           props>
              prop key="insert*">PROPAGATION_REQUIRED</prop>
              </props>
       </property>
</bean>
```

# **Transaction Attributes:**

- PROPAGATION\_REQUIRED : Supporta la transazione esistente, Se non esiste ne crea una nuova
- PROPAGATION\_SUPPORTS: Supporta la transazione esistente, Se non esiste esegue non transazionalmente
- PROPAGATION\_MANDATORY:Supporta la transazione esistente, Se non esiste lancia una eccezione
- PROPAGATION\_REQUIRES\_NEW:Fa partire sempre una nuova transazione, se ne esiste già una la sospende.
- PROPAGATION\_NOT\_SUPPORTED:Non viene eseguito se esiste una transazione attiva. Eseguito sempre non transaz.e sospende quelle attive.
- PROPAGATION\_NEVER: Eseguito non transazionalmente, viene lanciata na eccezione se esiste una transazione attiva.
- PROPAGATION\_NESTED:Eseguito in una transazione annidata se ne esiste una, altrimenti, viene eseguita come propagation\_required.

# Riferimenti 1

Spring:

http://www.springframework.org

Nel Jug Sardegna (Articoli, Recensioni Libri):

http://www.jugsardegna.org/vqwiki/jsp/Wiki?SpringFramework

# Jug Avis:

http://www.jugsardegna.org/vqwiki/jsp/Wiki?JugAvis

# Jug Avis Web (prima parte):

http://www.jugsardegna.org/vqwiki/jsp/Wiki?action=action\_view\_attachment & attachment=Spring webflowJug16Luglio2005.pdf



IBatis:

http://ibatis.apache.org/

Nel Jug Sardegna:

http://www.jugsardegna.org/vqwiki/jsp/Wiki?IBatisCaseStudy

Abator for iBatis:

http://ibatis.apache.org/abator.html



# AspectJ:

http://www.eclipse.org/aspectj/

Aspect-Oriented Software Development AOSD http://www.aosd.net/

AspectJ in Action

http://www.manning.com/books/laddad

# Grazie per l'attenzione.