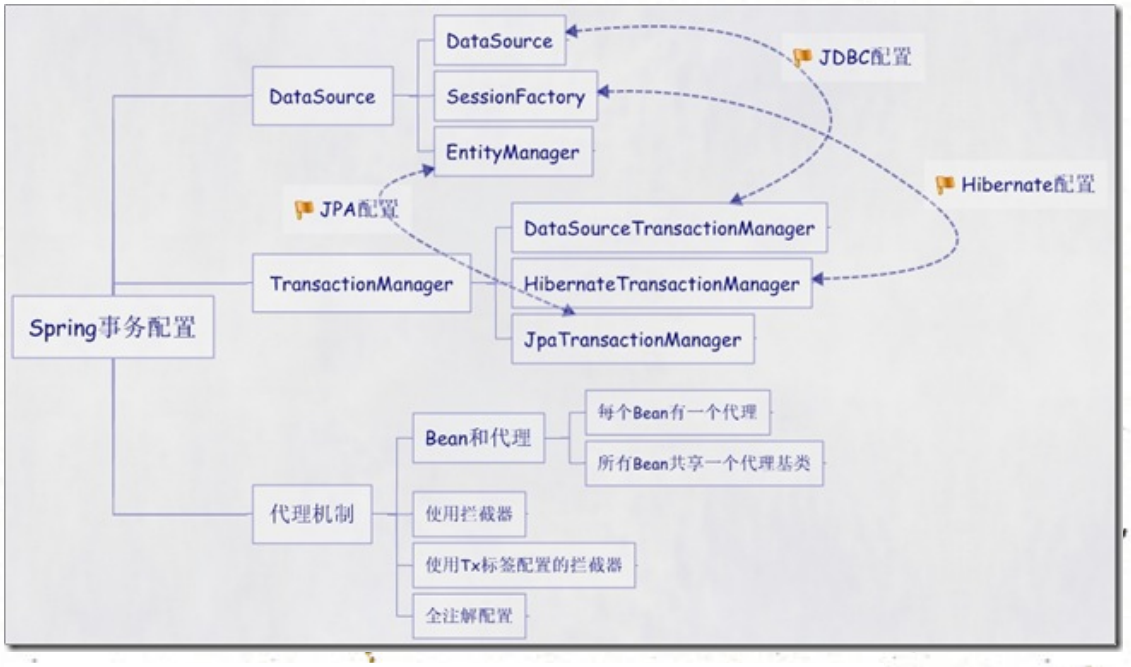
http://www.cnblogs.com/zuiyirenjian/p/4306855.html

Spring配置文件中关于事务配置总是由三个组成部分，分别是DataSource、TransactionManager和代理机制这三部分，无论哪种配置方式，一般变化的只是代理机制这部分。

DataSource、TransactionManager这两部分只是会根据数据访问方式有所变化，比如使用Hibernate进行数据访问时，DataSource实际为SessionFactory，TransactionManager的实现为HibernateTransactionManager。

具体如下图：



根据代理机制的不同，总结了五种Spring事务的配置方式，配置文件如下：

第一种：每个Bean都有一个代理：

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:aop="http://www.springframework.org/schema/aop"
    xmlns:tx="http://www.springframework.org/schema/tx"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
          http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-2.0.xsd
          http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-2.0.xsd">
  <!-- 数据源 -->
  <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
        destroy-method="close">
```

```
    <property name="driverClassName"
        value="com.mysql.jdbc.Driver" />
    <property name="url" value="jdbc:mysql://
        192.168.0.244:3306/test?
useUnicode=true&amp;characterEncoding=UTF-8" />
    <property name="username" value="root" />
    <property name="password" value="root" />
    <!-- 连接池启动时的初始值 -->
    <property name="initialSize" value="10" />
    <!-- 连接池的最大值 -->
    <property name="maxActive" value="10" />
    <!-- 最大空闲值.当经过一个高峰时间后，连接池可以慢慢将已经用不到的连接慢慢释放一
部分，一直减少到maxIdle为止 -->
    <property name="maxIdle" value="20" />
    <!--  最小空闲值.当空闲的连接数少于阀值时，连接池就会预申请去一些连接，以免洪峰来
时来不及申请 -->
    <property name="minIdle" value="10" />
    <property name="defaultAutoCommit" value="true" />
  </bean>
  <!-- 会话工厂 -->
  <bean id="sessionFactory"
        class="org.springframework.orm.hibernate3.LocalSessionFactoryBea
n">
    <property name="dataSource" ref="dataSource" />
    <property name="mappingLocations">
      <list>
        <value>classpath:/com/nms/entity/**/*.hbm.xml</value>
      </list>
    </property>
    <property name="hibernateProperties">
      <props>
        <prop key="hibernate.dialect"> org.hibernate.dialect.MySQL5Dialect
</prop>
        <prop key="hibernate.show_sql">true</prop>
        <prop key="hibernate.format_sql">true</prop>
      </props>
    </property>
  </bean>
  <!-- 定义事务管理器 -->
  <bean id="transactionManager"
        class="org.springframework.orm.hibernate3.HibernateTransactionMa
```

```
nager">
    <property name="sessionFactory" ref="sessionFactory" />
  </bean>
  <!-- 配置服务层 -->
  <bean id="userDaoAgency" class="com.dao.impl.UserDaoImpl">
    <property name="sessionFactory" ref="sessionFactory" />
  </bean>
  <bean id="userDao"
       class="org.springframework.transaction.interceptor.TransactionPr
oxyFactoryBean">
    <!-- 配置事务管理器 -->
    <property name="transactionManager" ref="transactionManager" />
    <property name="target" ref="userDaoAgency" />
    <property name="proxyInterfaces" value="com.dao.UserDao" />
    <!-- 配置事务属性 -->
    <property name="transactionAttributes">
      <props>
        <prop key="*">PROPAGATION_REQUIRED</prop>
      </props>
    </property>
  </bean>
</beans>
```

第二种：所有Bean共享一个代理

```
<?xml version="1.0" encoding="UTF-8"?>
<beans>
  <!-- 数据源 -->
  <bean id="dataSource"
    class="org.apache.commons.dbcp.BasicDataSource" destroy-
method="close">
    <!-- // 配置同上 -->
  </bean>
  <!-- 会话工厂 -->
  <bean id="sessionFactory"
    class="org.springframework.orm.hibernate3.LocalSessionFactoryBean">
    <!-- // 配置同上 -->
  </bean>
  <!-- 定义事务管理器 -->
  <bean id="transactionManager"
    class="org.springframework.orm.hibernate3.HibernateTransactionManage
r">
```

```
    <property name="sessionFactory" ref="sessionFactory" />
  </bean>
   <!-- 定义事务 -->
  <bean id="base"
    class="org.springframework.transaction.interceptor.TransactionProxyF
actoryBean"
    lazy-init="true" abstract="true">
    <!-- 配置事务管理器 -->
    <property name="transactionManager" ref="transactionManager" />
    <!-- 配置事务属性 -->
    <property name="transactionAttributes">
      <props>
        <prop key="*">PROPAGATION_REQUIRED</prop>
      </props>
    </property>
  </bean>
  <!-- 配置服务层 -->
  <bean id="userDao" class="com.dao.impl.UserDaoImpl">
    <property name="sessionFactory" ref="sessionFactory" />
  </bean>
  <!-- 代理对象 -->
  <bean id="userDaoAgency" parent="base">
    <property name="target" ref="userDao" />
  </bean>
</beans>
```

第三种：拦截器：

```
<?xml version="1.0" encoding="UTF-8"?>
<beans>
  <!-- 数据源 -->
  <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
destroy-method="close">
    <!-- // 配置同上 -->
  </bean>
  <!-- 会话工厂 -->
  <bean id="sessionFactory"
class="org.springframework.orm.hibernate3.LocalSessionFactoryBean">
    <!-- // 配置同上 -->
  </bean>
   <!-- 定义事务管理器（声明式的事务） -->
```

```
  <bean id="transactionManager"
class="org.springframework.orm.hibernate3.HibernateTransactionManager">
    <property name="sessionFactory" ref="sessionFactory" />
  </bean>
   <!-- 定义事务 -->
  <bean id="transactionInterceptor"
class="org.springframework.transaction.interceptor.TransactionIntercepto
r">
    <property name="transactionManager" ref="transactionManager" />
    <!-- 配置事务属性 -->
    <property name="transactionAttributes">
      <props>
        <prop key="*">PROPAGATION_REQUIRED</prop>
      </props>
    </property>
  </bean>
  <bean
class="org.springframework.aop.framework.autoproxy.BeanNameAutoProxyCrea
tor">
    <property name="beanNames">
      <list>
        <value>*DaoImpl</value>
      </list>
    </property>
    <property name="interceptorNames">
      <list>
        <value>transactionInterceptor</value>
      </list>
    </property>
  </bean>
  <!-- 配置服务层 -->
  <bean id="userDaoAgency" class="com.dao.impl.UserDaoImpl">
    <property name="sessionFactory" ref="sessionFactory" />
  </bean>
</beans>
```

第四种：使用tx标签配置的拦截器：

```
<?xml version="1.0" encoding="UTF-8"?>
<beans>
  <!-- 数据源 -->
```

```
  <bean id="dataSource">
    <!-- // 配置同上 -->
  </bean>
  <!-- 会话工厂 -->
  <bean id="sessionFactory">
    <!-- // 配置同上 -->
  </bean>
  <context:annotation-config />
  <context:component-scan base-package="com.dao" />
  <!-- 定义事务管理器 -->
  <bean id="transactionManager"
        class="org.springframework.orm.hibernate3.HibernateTransactionMa
nager">
    <property name="sessionFactory" ref="sessionFactory" />
  </bean>
  <!-- 声明式容器事务管理,指定事务管理器为transactionManager, add*方法参与事务 -
->
  <tx:advice id="txAdvice" transaction-manager="transactionManager">
    <tx:attributes>
      <!--REQUIRED表示没有sessionFactory的时候会自动建立，有的时候就不会建立了--
>
      <tx:method name="add*" propagation="REQUIRED" />
    </tx:attributes>
  </tx:advice>
  <!-- 定义切面 -->
  <aop:config>
    <!-- 只对dao层实施事务 -->
    <aop:pointcut id="interceptorPointCuts" expression="execution(*
com.dao.*.*(..))" />
    <!-- Advisor定义，切入点和通知分别为txPointcut、txAdvice -->
    <aop:advisor advice-ref="txAdvice" pointcut-
ref="interceptorPointCuts" />
  </aop:config>
</beans>
```

第五种：注解：

```
<?xml version="1.0" encoding="UTF-8"?>
<beans>
  <!-- 数据源 -->
  <bean id="dataSource">
```

```
    <!-- // 配置同上 -->
  </bean>
  <!-- 会话工厂 -->
  <bean id="sessionFactory">
    <!-- // 配置同上 -->
  </bean>
  <context:annotation-config />
  <!-- 使用注解的包路径 -->
  <context:component-scan base-package="com.dao" />
  <!-- 支持  @Transactional 标记 -->
  <tx:annotation-driven transaction-manager="transactionManager"/>
  <!-- 定义事务管理器 -->
  <bean id="transactionManager"

class="org.springframework.orm.hibernate3.HibernateTransactionManager">
    <property name="sessionFactory" ref="sessionFactory" />
  </bean>
</beans>
```