**一、PO :(persistant object )，持久对象**

可以看成是与数据库中的表相映射的java对象。使用Hibernate来生成PO是不错的选择。  
  
**二、VO :(value object) ，值对象**

通常用于业务层之间的数据传递，和PO一样也是仅仅包含数据而已。但应是抽象出的业务对象,可以和表对应,也可以不,这根据业务的需要.  
  
PO只能用在数据层，VO用在商业逻辑层和表示层。各层操作属于该层自己的数据对象，这样就可以降低各层之间的耦合，便于以后系统的维护和扩展。  
  
**三、DAO :(Data Access Objects)  ，数据访问对象接口**DAO是Data Access Object数据访问接口，数据访问：顾名思义就是与数据库打交道。夹在业务逻辑与数据库资源中间。  
  
J2EE开发人员使用数据访问对象（DAO）设计模式把底层的数据访问逻辑和高层的商务逻辑分开.实现DAO模式能够更加专注于编写数据访问代码.  
  
DAO模式是标准的J2EE设计模式之一.开发人员使用这个模式把底层的数据访问操作和上层的商务逻辑分开.一个典型的DAO实现有下列几个组件：  
  1. 一个DAO工厂类；  
  2. 一个DAO接口；  
  3. 一个实现DAO接口的具体类；  
  4. 数据传递对象（有些时候叫做值对象）.  
  具体的DAO类包含了从特定的数据源访问数据的逻辑。  
  
**四、BO :(Business Object)，业务对象层**表示应用程序领域内“事物”的所有实体类。这些实体类驻留在服务器上，并利用服务类来协助完成它们的职责。  
  
**五、POJO :(Plain Old Java Objects)，简单的Java对象**实际就是普通JavaBeans,使用POJO名称是为了避免和EJB混淆起来, 而且简称比较直接.  
其中有一些属性及其getter、setter方法的类,有时可以作为value object或dto(Data Transform Object)来使用.当然,如果你有一个简单的运算属性也是可以的,但不允许有业务方法,也不能携带有connection之类的方法。