[企业应用架构阅读笔记2](https://www.cnblogs.com/zlj843767688/p/13096938.html)

时间： 2020-06-11

· 领域模型：使用不同职责的对象来联合解决业务问题，而不是通过事务脚本来处理数据

· 阻抗不匹配：对象模型和关系型数据库之间的不匹配，通常通过对象-关系映射(ORM)解决

· 软件事务的四个特性

1.      原子性：要么全部成功，要么全部回滚

2.      一致性：事务开始和完成时，资源都不应该被破坏

3.      隔离性：事务成功完成之前，其影响不应该被看到

4.      持久性：事务不会因为任何崩溃而丢失更改

· 事务的隔离级别（由高而低）

1.      可串行化：完全隔离，并发执行的结果与以某种顺序依次执行的结果相同

2.      可重复读：允许幻读，更新者向集合中插入了一些元素而读的人只能看到其中一部分

3.      读已提交：允许不可重复读，所有已经提交的数据都可以读

4.      读未提交：允许脏读，允许读未提交的数据

· 会话状态：无状态对象是一种不良设计；用无状态的服务器可以实现有状态的会话；如果有很多会话空闲，可以考虑用数据库存储会话；如果需要频繁访问会话，则应该使用服务器会话

· 分布策略：分布对象的第一定律：不要使用分布对象；分布对象的第二定律：节约使用分布对象

领域逻辑模式分为 事物脚本、领域模型、表模块和服务层四种模式

　　很多设计者喜欢把业务逻辑分成两类：领域逻辑和应用逻辑，前者只与问题领域有关、而后者有时被称为工作流逻辑

1. 事务脚本

　　通过使用SQL语句或者存储过程返回记录集，记录集在系统的各层之间传递，在必要的时候可以通过更新记录集、使用SQL语句或者存储过程的方式更新数据库  
　　事务脚本胜在简单，通常应用在小型的项目和系统中，但业务逻辑越来越复杂，使用这一模式就越来越难以保持良好设计，因为在程序里面充斥了大量的SQL语句和命令，一旦数据结构发生更改或者需要对系统进行修改，可能会出现许多难以发现的问题

2. 领域模型

　　领域模型是合并了行为和数据的领域的对象模型，领域模型创建了一张由互联对象组成的网，其中的每个对象都代表着某个有意义的个体，可能大到一个公司或者小到订单的一行

　　简单领域可以使用活动记录，即简单的单条数据记录和单个对象对应的模式，一个对象对应数据库中的一个表

　　复杂领域模型需要使用数据映射器，它可能使用继承、策略或者其他的设计模式，是一张由互联的细粒度对象组成的复杂网络，我们经常会看到：多个类通过交互来完成很简单的任务

　　在面向对象技术中，通过从一个对象到另一个对象的连续传递可以把行为传递给最有资格处理的对象，它同时消除了很多条件判断行为

领域模型的要点在于隐藏数据库的存在

3. 表模块

　　表模块是处理某一数据库表或视图中所有行的业务逻辑的一个实例

　　表模块通过强类型或弱类型的数据集与对象结合使用，使用主键查询数据，是.Net中使用的很多的一种模式，主要使用主键、半对象化的操作数据---之所以说是半结构化，是因为所用的对象基本上只具有行为，通过传入参数执行特定的操作或者查询记录集，而几乎不承载任何数据

　　在.net中，这种模式因为其容易和UI进行绑定和交互，所以倍受欢迎

4. 服务层

　　通过一个服务层来定义应用程序边界，在服务层中建立一组可用的操作的集合，并在每个操作内部协调应用程序的响应

　　服务层是一种组织业务逻辑的模式，有点类似于业务外观；WEB SERVICE通常担任着服务层的角色

　　服务层可以通过领域外观方法和操作脚本方法实现，领域外观方法中，服务层以领域模型之上的瘦外观集合方式出现，负责实现外观的类中不不包含业务逻辑；而在操作脚本方法中 ，服务层由一组相对复杂的类组成，这些类直接实现应用逻辑，但将领域逻辑委托给封装好的领域对象类

 　　服务层的类的接口是粗粒度的，适合于远程调用。但是，在开始时，我们应该仅设计一个本地调用的服务层，在需要远程调用时，再在服务层上增加一个远程外观。