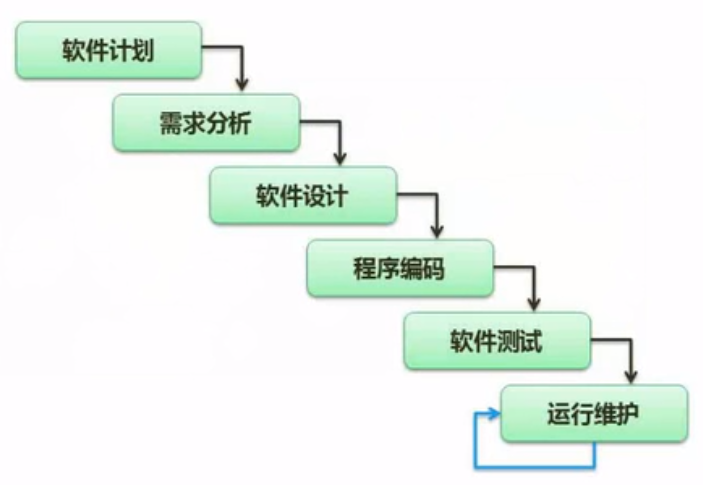
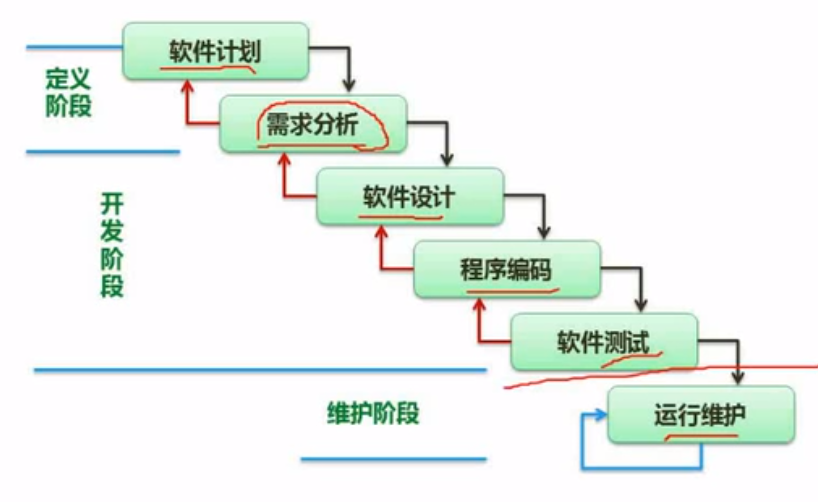
# 软件开发模型

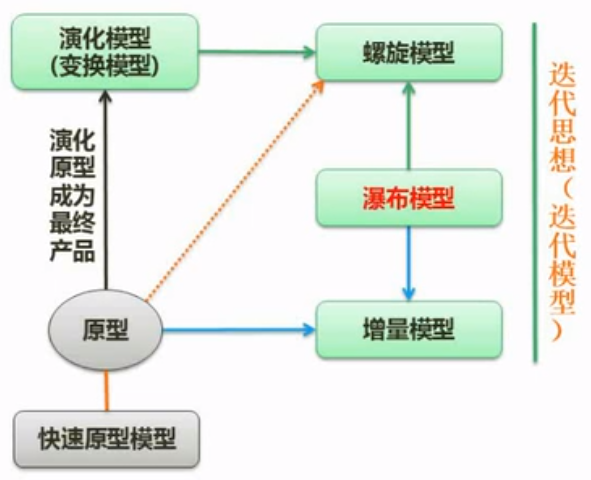
|  |  |
| --- | --- |
| 瀑布模型  演化模型  增量模型  螺旋模型  快速原型模型  喷泉模型  V模型 | 迭代模型/迭代开发方法  快速应用开发  构件组装模型/基于构件的开发方法  统一过程/统一开发方法  敏捷开发方法  模型驱动的开发方法  基于架构的开发方法 |

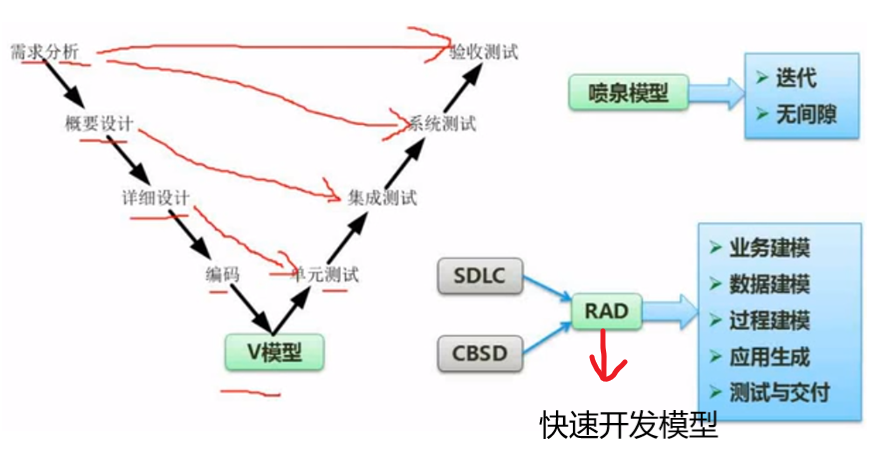
## 瀑布模型（SDLC）



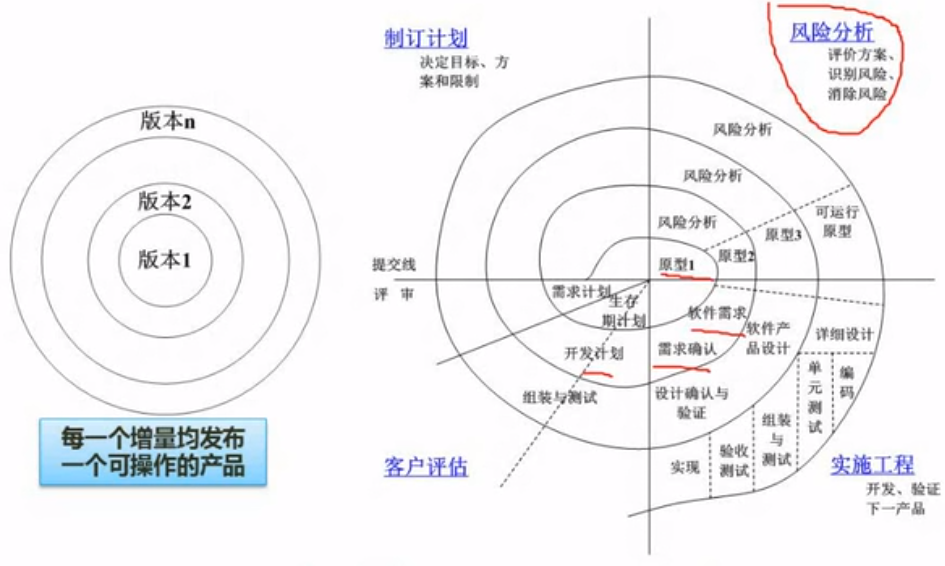


## 其他经典模型

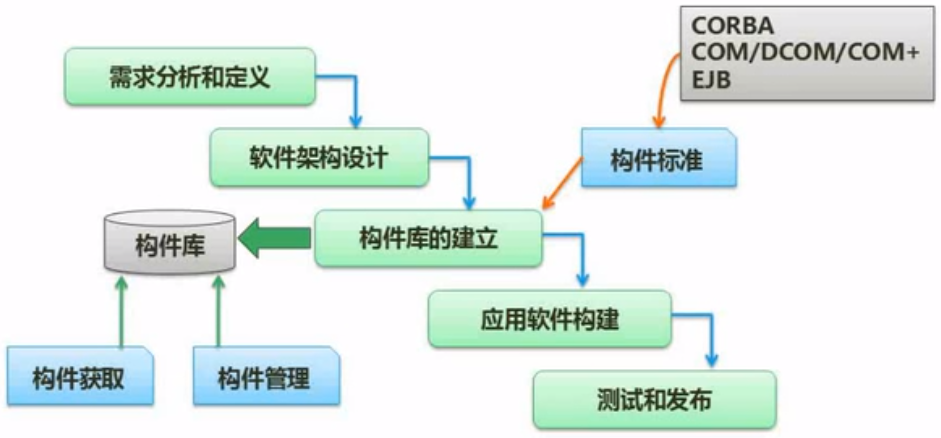




## 增量模型与螺旋模型



## 构建组装模型（CBSD）



### EJB

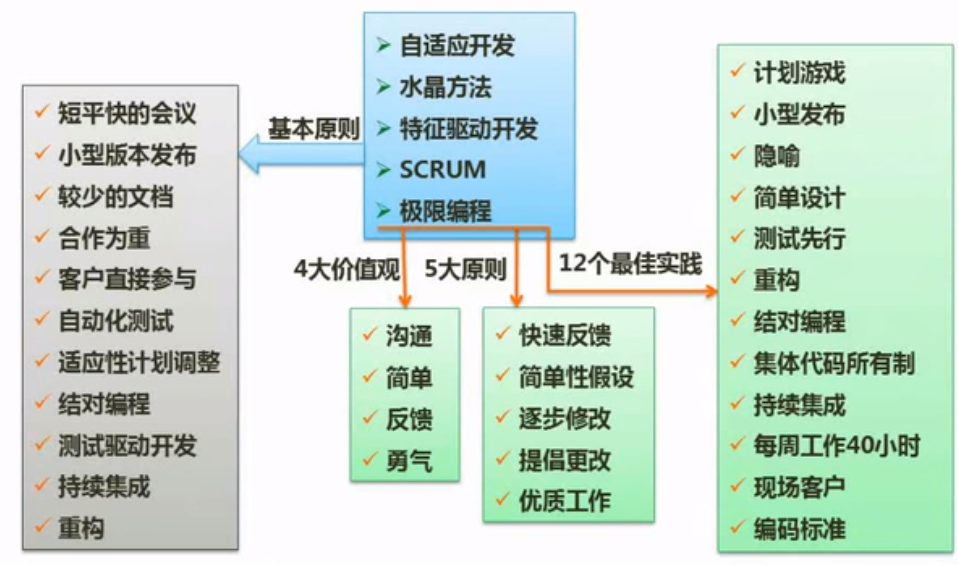
EJB是sun的javaEE服务器端组件模型，设计目标与核心应用是部署分布式应用程序。简单来说就是把已经编写好的程序（即：类）打包放在服务器上执行。凭借java跨平台的优势，用EJB技术部署的分布式系统可以不限于特定的平台。EJB（Enterprise JavaBean）是J2EE的一部分，定义了一个用于开发基于组件的企业多重应用程序的标准。其特点包括网络服务中心支持和核心开发工具（SDK）。在J2EE里，Enterprise Java Beans（EJB）称为Java企业Bean，是Java的核心代码，分别是会话Bean（Session Bean），实体Bean（Entity Bean）和消息驱动Bean（Message Bean）。在EJB3.0推出以后，实体Bean被单独分了出来，形成新的规范JPA。

#### 简介

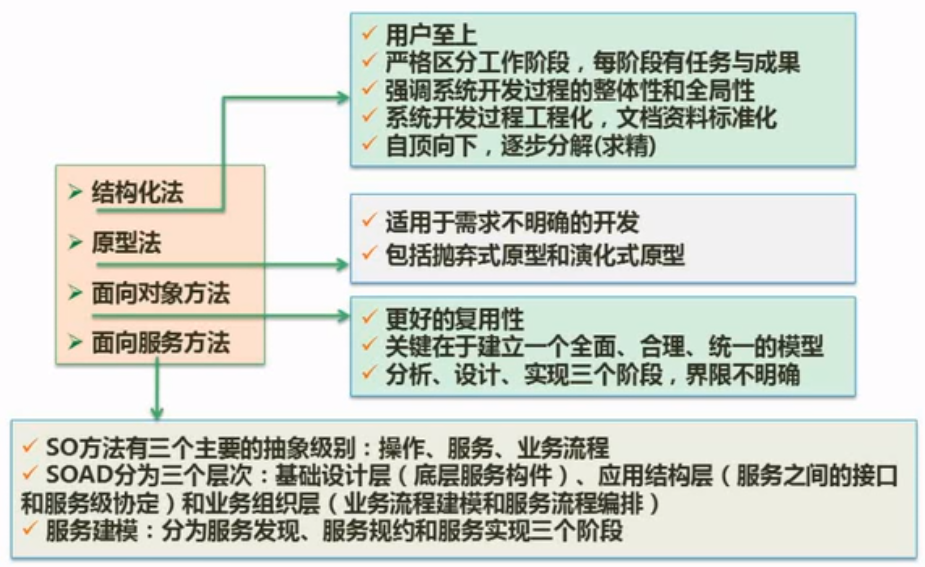
1. SessionBean用于实现业务逻辑，它可以是有状态的，也可以是无状态的。每当客户端请求时，容器就会选择一个SessionBean来为客户端服务。Session Bean可以直接访问数据库，但更多时候，它会通过Entity Bean实现数据访问。
2. EntityBean是域模型对象，用于实现O/R映射，负责将数据库中的表记录映射为内存中的Entity对象，事实上，创建一个EntityBean对象相当于新建一条记录，删除一个EntityBean会同时从数据库中删除对应的记录，修改一个EntityBean时，容器会自动将Entity Bean的状态和数据库同步。
3. MessageDriven Bean是EJB2.0中引入的新的企业Bean，它基于JMS消息，只能接收客户端发送的JMS消息然后处理。MDB实际上是一个异步的无状态SessionBean，客户端调用MDB后无需等待，立刻返回，MDB将异步处理客户端请求。这适合于需要异步处理请求的场合，比如订单处理，这样就能避免客户端长时间的等待一个方法调用直到返回结果。

EJB实际上是SUN的J2EE中的一套规范，并且规定了一系列的API用来实现把EJB概念转换成EJB产品。

## 敏捷开发方法

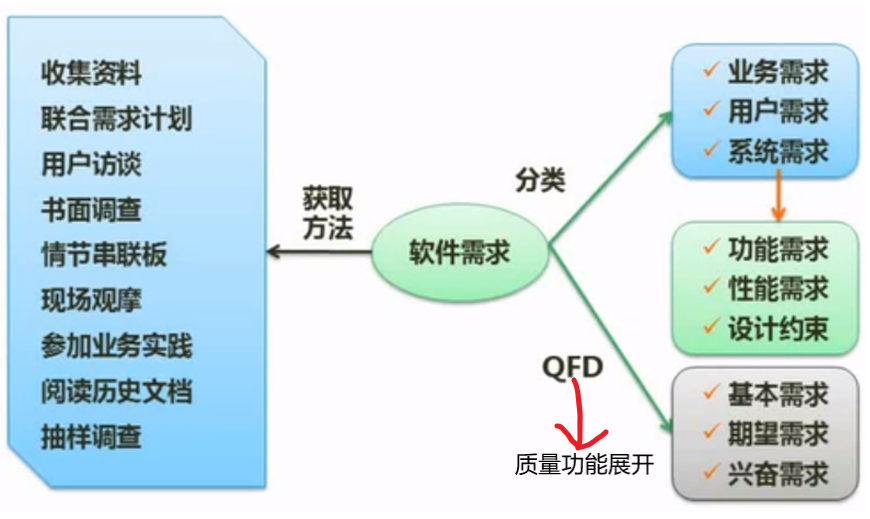


## 信息系统开发方法



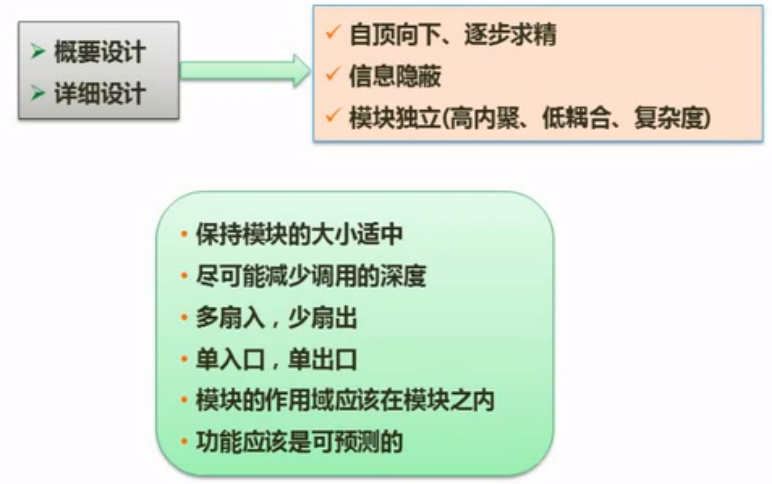
## 需求开发

### 需求分类于需求获取



## 结构化设计

### 基本原则

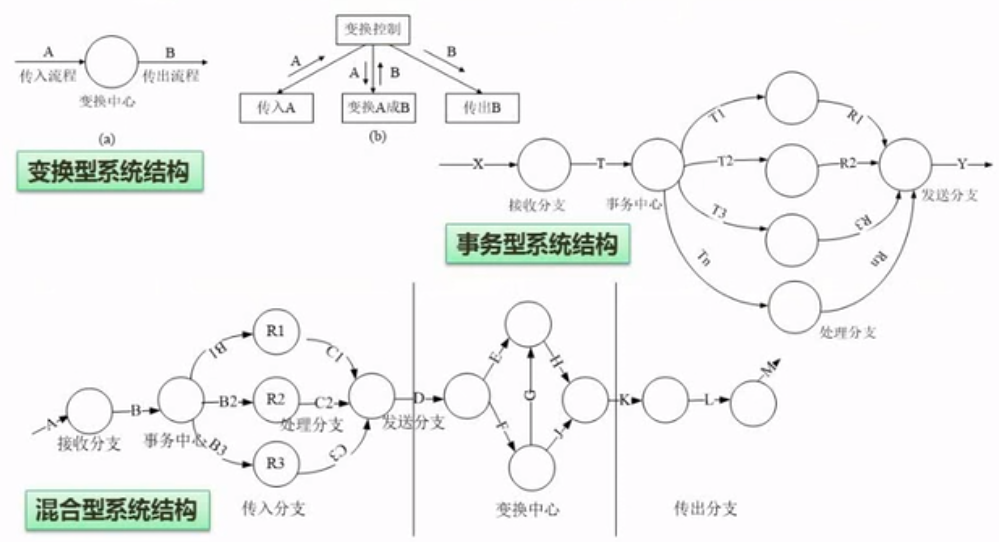


### 内聚与耦合

|  |  |
| --- | --- |
| 内聚类型 | 描述 |
| 功能内聚 | 完成一个单一功能，各个部分协同工作，缺一不可 |
| 顺序内聚 | 处理元素相关，而且必须顺序执行 |
| 通信内聚 | 所有处理元素集中在一个数据结构的区域上 |
| 过程内聚 | 处理元素相关，而且必须按特定的次序执行 |
| 瞬时内聚（时间内聚） | 所包含的任务必须在同一时间间隔内执行 |
| 逻辑内聚 | 完成逻辑上相关的一组任务 |
| 偶然内聚（巧合内聚） | 完成一组没有关系或松散关系的任务 |

|  |  |
| --- | --- |
| 耦合类型 | 描述 |
| 非直接耦合 | 两个模块之间没有直接关系，他们之间的联系完全是通过主模块的控制和调用来实现的 |
| 数据耦合 | 一组模块借助参数表传递简单数据 |
| 标记耦合 | 一组模块通过参数表传递记录信息（数据结构） |
| 控制耦合 | 模块之间传递的信息中包含用于控制模块内部逻辑的信息 |
| 外部耦合 | 一组模块都访问同一全局简单变量，而且不是通过参数表传递该全局变量的信息 |
| 公共耦合 | 多个模块都访问同一个公共数据环境 |
| 内容耦合 | 一个模块直接访问另一个模块的内部数据；一个模块不通过正常入口转到另一个模块的内部；两个模块有一部分程序代码重叠；一个模块有多个入口 |

### 系统结构/模块结构



## 软件测试

### 测试原则与类型

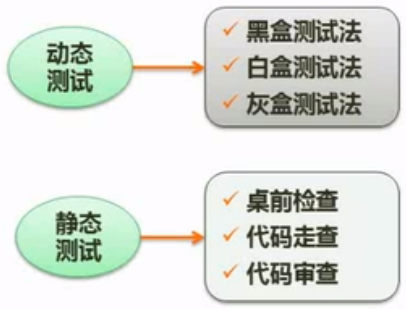
尽早、不断的进行测试

程序员避免测试自己设计的程序

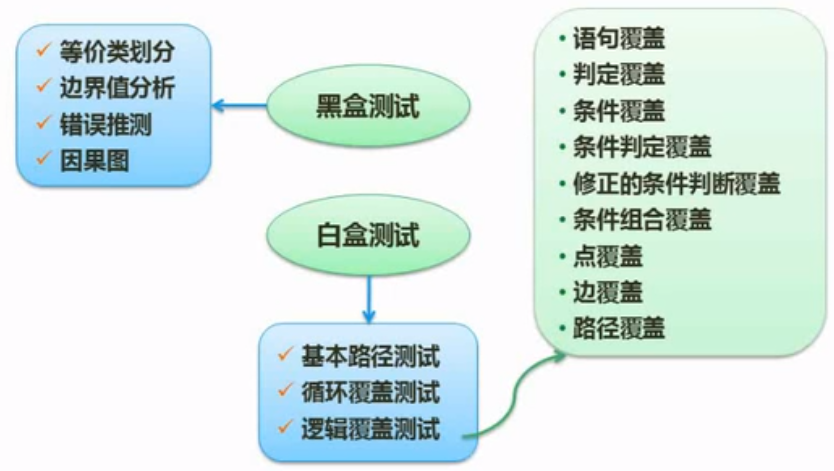
既要选择有效、合理的数据，也要选择无效、不合理的数据

修改后应进行回归测试

尚未发现的错误数量与该程序以发现错误数成正比



### 测试用例设计



### 测试阶段

