# 第一章

1、根据自己的经验，谈谈对软件危机的看法

软件成本日益增长

开发进度难以控制

软件质量差

软件维护困难

2、理解软件体系结构的定义

体系结构是以组件、组件之间的关系、组件与环境之间的关系为内容的某一系统的基本组织结构以及指导上述内容设计与演化的原理

1. 影响构架的因素主要包括哪些

系统涉众

开发组织

构架师的素质和经验

技术环境

1. 构架主要研究哪些内容
   1. 对系统组成元素的描述
   2. 这些元素相互之间的交互
   3. 系统组成的模式
   4. 模式的约束

5、构架的描述

* + 非正式的
  + 因人而异
  + 针对特定系统的
  + 通常由框图和线条以及相关的解释所组成

6、构架由什么来决定

最终决定因素是质量属性

7、三种主要的结构类型

* + 模块结构
  + 组件－连接器结构
  + 分配结构

8、软件构架的作用

* 涉众之间的交流
* 系统设计的前期决策
* 可传递的系统级

# 第二章

1、管道过滤器风格的组件及属于哪种结构风格、优缺点

* 属于经典的数据流体系结构风格
* 优点

系统的总体输入输出是每个过滤器的简单组合，易于理解

支持重用

易于维护和修改

系统的一些特性容易分析，比如吞吐量、死锁

支持并发

* 缺点

经常退化为批处理系统

交互能力差、管道之间同步困难

2、面向对象模型的组成元素

封装 继承 多态

3、什么是基于事件的模式

对象之间通过方法调用进行交互，但是调用是隐式的，通过事件机制进行调用触发

* + 对象可以发布（广播）事件
  + 对象也可以通过事件注册将某个方法和事件进行绑定

一旦该事件触发，所有与该事件绑定的方法都将被调用

4、分层模式的优缺点

**优点**

* 支持逐层抽象的方式进行设计，从而可以将一个系统的设计划分为多个相对简单的层次
* 容易修改，每层的改变至多影响上下两层
* 支持重用，只要保持接口不变，每层的具体实现的替换是透明的
* 支持可移植性

**缺点**

* 性能上的额外支出
* 并非每个系统都可以进行分层设计
* 跨层使用可能会使损坏整个模型

1. 仓库模式中，数据交换的两种方式
   1. 共享的数据放在中心数据库中，所有组件可以通过访问数据库进行交互
   2. 每个组件维护自己的数据库，通过显示的传递数据与其他组件进行交互
2. 模块结构包括哪些风格及各风格的组成元素

***分解风格—模块***

***使用风格—模块***

***泛化（类）风格—模块***

***分层风格—层***

1. 组件－连接器视图类型包括哪些风格，重点是c/s风格服务器的任务及主要组成元素，P2P风格例子，作用

共享数据

服务器/客户端（C/S）

P2P

通信进程

管道和过滤器

发布－订阅

* C/S体系风格中，服务器的主要任务是：
  + 数据库安全性的要求
  + 数据库访问并发性的控制
  + 数据库的备份与恢复
* 三层C/S风格的主要组成元素：
  + 客户端
  + 应用服务器
  + 数据库服务器
* **中间件是最重要的构件**

例子Bittorrent, eMule

作用：

* P2P风格展示了一副以合作的领域划分应用的系统视图

P2P实体既是服务器，有时客户端

为系统部署在一个分布式的平台上提供了灵活性

* P2P计算

分布式的计算应用

在一个计算机网络中，高效的利用CPU和磁盘资源

负载分布

1. 分配视图风格包括哪些风格，

部署风格 实施风格 工作分配风格

实施及分配风格作用及组成元素

实施风格作用

* 在开发、编译中管理和维护与软件元素相对应的文件
* 开发者使用实现风格来
  + 找到相应的文件进行更新，测试，编译
  + 找到文件过去的版本
* 对一个特定的系统，指定版本的差别
* 对需要特殊关注的元素进行高亮显示

实施风格组成元素：

软件元素：一个模块

环境元素：一个配置项，比如：文件或目录

分配风格作用：

* 管理团队资源的分配
* 向新员工解释一个项目的结构
* 对项目成本和开发周期的详细估计

组成元素：

软件元素：一个模块

环境元素：一个组织单元，比如：一个人，一个队伍，一个部门，一个子开发商等

# 第三章

1、常用的质量属性有哪些?功能性是不是质量属性？

1. **易用性**
2. **可修改性**
3. **性能**
4. **安全性**
5. **可测试性**
6. **可用性**

不是

2、质量属性的达成，必须在哪些阶段进行考虑？

设计 实现 部署

3、质量属性的类型包括哪三个方面？

* 系统的质量属性
* 商业属性
* 与构架本身相关的一些质量属性

4、可用性场景衡量指标如何计算？故障与错误区别？

* **α = 平均正常工作时间 / (平均正常工作时间+ 平均修复时间)**
* 故障 VS. 错误
  + 系统的用户可以观察到故障，但是观察不到错误
  + 系统的错误可能引起一个故障，也可能被屏蔽

5、安全性场景包括哪六个方面？

* 不可否认性－交易不能被交易的任何一方所否认
* 保密性－未经授权不能访问数据或服务
* 完整性－按计划来提交数据或服务的属性
* 保证－交易的各方应该是他们声称的人
* 可用性－系统可用于合法的用途（没DOS攻击）
* 审核－系统在内部跟踪活动，跟踪级别足以对活动进行重构

6、易用性一般分为哪五个方面？

* 学习系统的特性
* 有效地使用系统
* 将错误的影响降到最低
* 使系统适应用户的需要
* 提高自信和满意度