软工II简答题补充

变更控制的过程

1. 提请变化请求
2. 评估变化
3. 批准或拒绝变化
4. 完成变更修改工作
5. 验证

简述软件生命周期的含义

软件生命周期是软件的产生直到报废或停止使用的生命周期，简要描述了该系统的开发活动简史。

建立需求规格说明的必要性

1. 方便交流：软件开发过程中，**子任务与人员**之间存在错综复杂的关系，存在大量的沟通和交流，所以要编写软件开发中要编写不同类型的文档，每种文档都是针对项目中需要广泛交流的内容。因为**软件需求**需要进行**广泛交流**，所以要把需求文档化。
2. 跟踪和度量：需求规格说明是在**软件产品的角度以系统级需求列表的方式**描述软件系统解决方案，书写需求规格说明，可以建立管理控制的基线，方便任务分配，制定工作计划，进行跟踪和度量。
3. 过程管理：在实验中，需求规格的重要性不只体现在结果上，还包括中间过程，在书写需求规格过程中，才真正把问题域的问题和分析模型的成果转化为系统级需求，方便小组成员真正明确需求，个人认为在这个阶段包含一部分的需求在发现和完整化。

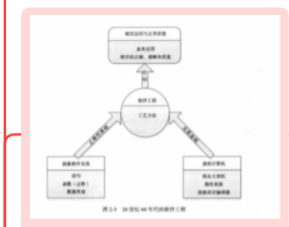
软件设计的核心层次及其主要思想

1. 高层设计：基于反映软件高层抽象的构件设计，描述系统的高层结构、关注点和设计决策。
    1. **部件**承载了系统主要的**计算与状态**
    2. **连接件**承载部件之间的**交互**
    3. 部件与连接件都是抽象的类型定义（就像类定义），它们的实例（就像类的对象实例）组织构成软件系统的整体结构，**配置**将它们的实例连接起来
2. 中层设计：更加关注组成构件的模块的设计、导入/导出、过程之间调用关系或者类之间的协作，模块划分**隐藏**一些程序片段（数据结构+算法）的细节，暴露接口于外界
3. 低层设计：深入模块和类的内部，关注具体的数据结构、算法、类型、语句和控制结构等。

软件工程的发展

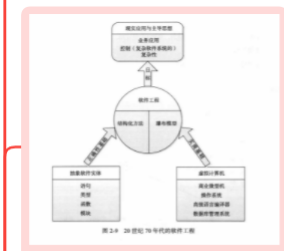
50

- 科学计算
- 以机器为中心进行编程
- 像生产硬件一样生产软件



60

- 业务应用（批量数据处理和事务计算）
- 软件不同于硬件
- 用软件工艺的方式生产软件



70

- 结构化方法
- 瀑布模型
- 强调规则和纪律

80

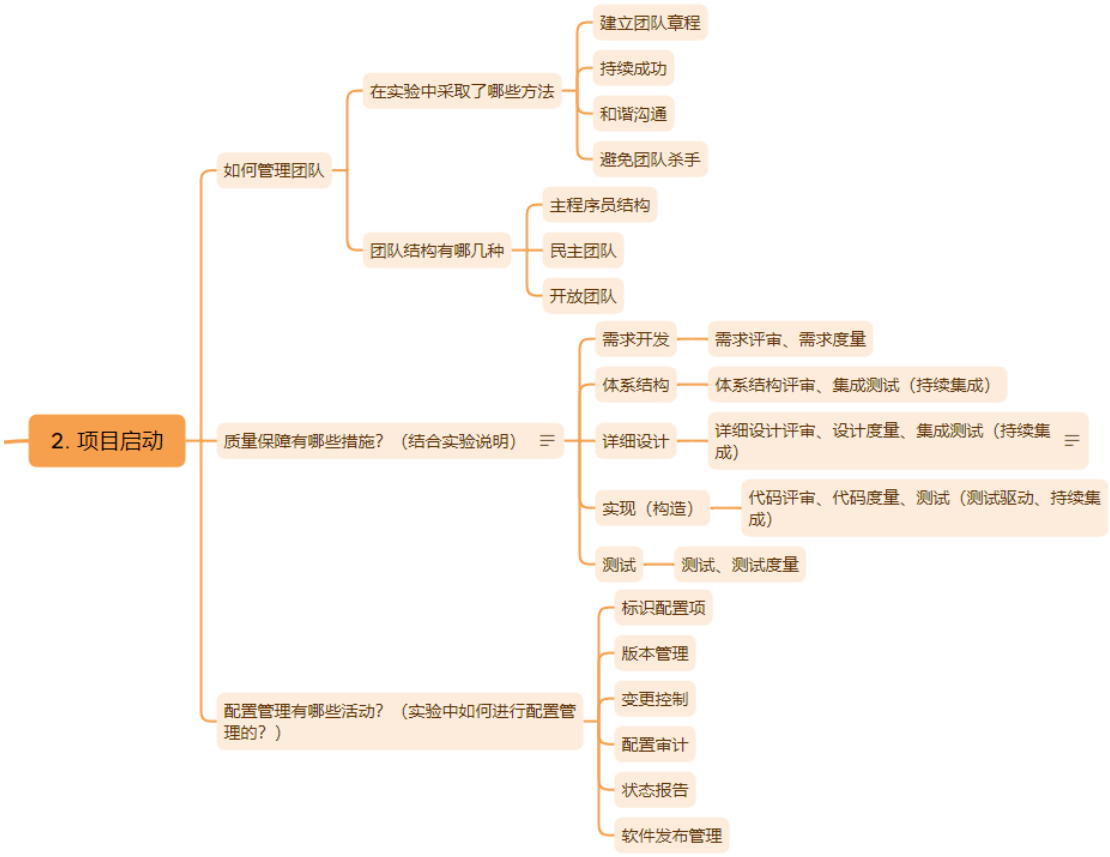
- 追求生产力最大化
- 现代结构化方法/面向对象编程广泛应用
- 重视过程的作用

90

- 企业为中心的大规模软件系统开发
- 追求快速开发、可变更性和用户价值
- WEB应用的出现

- 大规模WEB应用
- 大量面向大众的WEB产品
- 追求快速开发、可变更性、用户价值和创新

项目启动



瀑布模型

虽然从现在的角度来看，瀑布模型不能算是一个好的软件过程模型，但是考虑到它所出现的时代背景，与构建－修复模型相比，它还是体现出了明显的优势：为软件开发活动定义了清晰的阶段划分（包括了输入/输出、主要工作及其关注点），这让开发者能够以关注点分离的方式更好地进行那些复杂的软件项目的开发活动。

瀑布模型的局限性有以下几点：

- 对文档的过高期望。瀑布模型是文档驱动的，要求每个阶段都产生完备和可靠的文档，这一方面会耗费很大的工作量和成本，另一方面不切实际，因为实际开发中需求经常变化，无法建立完备可靠的文档。
- 对开发活动的线性顺序假设。瀑布模型将开发活动区分为不同阶段，要求一个阶段的工作经过验证后才能进入后续阶段，这也是不切实际的。因为在实际开发中，常常需要进行一定的后续工作才能验证当前的工作是否正确、可靠（例如，只有完成体系结构原型关键代码才能验证体系结构设计的有效性）。
- 客户、用户参与不够。瀑布模型将需求限制为一个阶段，也就将客户、用户的项目参与限制在了一个时间段。但过去的实践，尤其是 20 世纪 90 年代以来的实践表明，这种用户参与方式是远远不够的，会导致项目的失败。成功的项目开发需要客户、用户从始至终的参与。
- 里程碑粒度过粗。瀑布模型要求一个阶段完成之后才进行验证，即在每一个阶段设置一个里程碑，这是远远不够的。因为现在的软件系统比较复杂，常常需要数月才能结束一个阶段，这种情况下的里程碑粒度过粗，基本丧失了“早发现缺陷早修复”这一瀑布模型最有意义的思想。尤其是只有在所有开发完成之后（常常持续数年），客户和用户才能看到软件产品的方式具有极高的风险。

增量迭代模型

3. 特点

增量迭代模型的优点是：

- 迭代式开发更加符合软件开发的实践情况，具有更好的适用性；
- 并行开发可以帮助缩短软件产品的开发时间；
- 渐进交付可以加强用户反馈，降低开发风险。

增量迭代模型的缺点是：

- 由于各个构件是逐渐并入已有的软件体系结构中的，所以加入构件必须不破坏已构造好的系统部分，这需要软件具备开放式的体系结构。
- 增量迭代模型需要一个完备、清晰的项目前景和范围以进行并行开发规划，但是在一些不稳定的领域，不确定性太多或者需求变化非常频繁，很难在项目开始就确定前景和范围。

因为能够很好地适用于大规模软件系统的开发，所以增量迭代模型在实践中有着广泛的应用，尤其是比较成熟和稳定的领域。

演化模型

3. 特点

演化模型的优点是：

- 使用了迭代式开发，具有更好的适用性，尤其是其演化式迭代安排能够适用于那些需求变更比较频繁或不确定性较多的软件系统的开发；
- 并行开发可以帮助缩短软件产品的开发时间；
- 渐进交付可以加强用户反馈，降低开发风险。

演化模型的缺点是：

- 无法在项目早期阶段确定项目范围，所以项目的整体计划、进度调度，尤其是商务协商事宜无法准确把握；
- 后续迭代的开发活动是在前导迭代基础上进行修改和扩展的，这容易让后续迭代忽略分析与设计工作，蜕变为构建－修复方式。

在实践中，不稳定领域的大规模软件系统开发适合使用演化模型进行组织。

增量迭代模型与演化模型的异同

3. 特点

演化模型的优点是：

- 使用了迭代式开发，具有更好的适用性，尤其是其演化式迭代安排能够适用于那些需求变更比较频繁或不确定性较多的软件系统的开发；
- 并行开发可以帮助缩短软件产品的开发时间；
- 渐进交付可以加强用户反馈，降低开发风险。

演化模型的缺点是：

- 无法在项目早期阶段确定项目范围，所以项目的整体计划、进度调度，尤其是商务协商事宜无法准确把握；
- 后续迭代的开发活动是在前导迭代基础上进行修改和扩展的，这容易让后续迭代忽略分析与设计工作，蜕变为构建－修复方式。

在实践中，不稳定领域的大规模软件系统开发适合使用演化模型进行组织。