**2019-2020学年 第1学期**

**软件学院《Rational统一过程》报告**

|  |  |
| --- | --- |
| **班级** | **软件学院11班** |
| **学号** | **55171106** |
| **姓名** | **杜维康** |

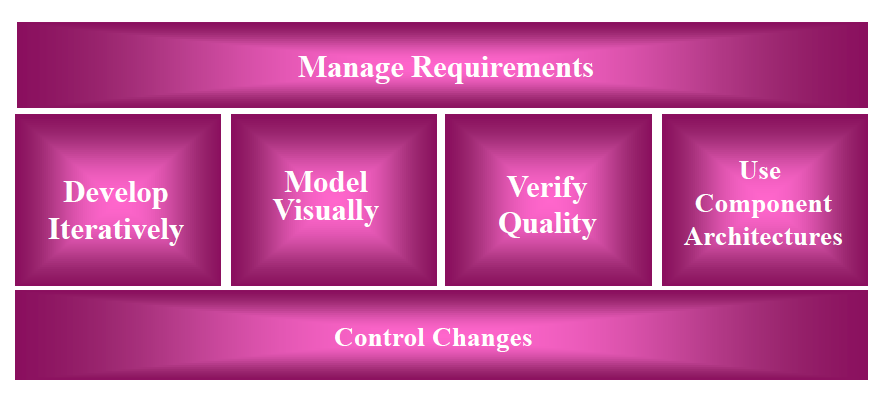
**要求：报告用小四号字体，标清题号。不需要抄写题目。**

**一、简答题**

**1.**（1）软件开发的六大最佳实践：

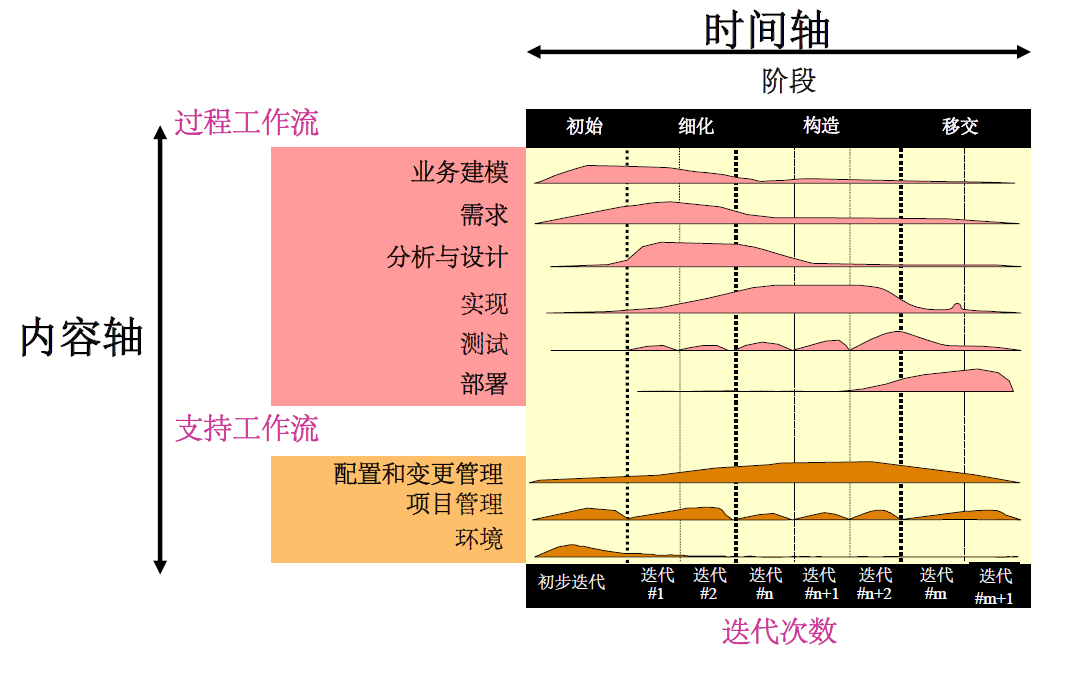
* 迭代式开发（Develop Iteratively）
* 管理需求（Manage Requirements）
* 基于组件的体系结构（Use Component Architectures）
* 可视化建模（Model Visually）
* 验证软件质量（Verify Quality）
* 控制软件变更（Control Changes）

（2）它们之间的关系：



第1题图 六大最佳实践的关系

**2.**RUP的二维过程结构图：



第2题图：RUP二维过程结构

**3.**RUP应用了四种模型元素：

（1）工作人员(worker)：谁来做

* 工作人员(worker)定义了个人或一个工作组的行为和职责。
* 行为用活动(activity)表示，每一个“worker”都与一组“内聚”的活动相联系。
* 内聚是指这些活动最好由一个人来完成。
* 职责的表示通常与某一特定制品(artifact)相联系，这些制品由worker制造、修改和控制。

常见的工作人员有：

* 系统分析员：概述系统功能和界定系统；引导和协调需求，用例建模
* 设计师：定义一个或多个类的职责、操作、属性和它们之间的关系；决定如何调整类以适应实现环境
* 测试设计师：计划、设计、实现和评估测试，包括产生测试计划和测试模型，执行测试规程，评估测试覆盖度、测试结果和测试效率

（2）活动(activity)：怎么做

* 活动(activity)定义了worker执行的工作
* 活动有明确的目的，能够产生有意义的结果。通常是生产制品或更新制品(如模型、类或计划)
* 活动时间可以是几个小时到几天
* 一个活动通常涉及一个worker，影响到一个或几个制品
* 多个活动也可能重复作用于一个制品，如从一个迭代过程到另一个迭代过程时

常见的活动有：

* 计划迭代过程：项目经理完成
* 寻找用例和参与者：系统分析员完成
* 评审设计：设计评审员完成
* 执行性能测试：性能测试人员完成

（3）制品(artifact)：做什么

* 制品(artifact)是由过程生产、修改或使用的有形产品
* 项目在生产出最终软件产品的过程中生产或使用它们
* 根据与活动的关系，制品分为输入制品和输出制品

常见的制品有：

* 模型：如用例模型和设计模型
* 模型元素：如一个用例、类、子系统
* 文档：如一个业务用例、软件架构文档
* 源代码
* 可执行文件

（4）工作流(workflow)：什么时候做，描述能够产出有用成果的有重要意义的活动序列，并表示出worker之间的交互作用

RUP将所有工作人员和活动按涉及的领域进行逻辑分组，划分为九个核心工作流：

* 核心过程工作流：业务建模、需求、分析和设计、实现、测试、实施
* 核心支持工作流：项目管理、配置和变更管理、环境

在RUP中，有一些附加的过程元素：

（1）指南

* 工作指南：包括评审工作指南、用例研讨会工作指南
* 制品指南：包括建模指南、编程指南、用户接口指南

（2）模板：用于制造出相应的制品

（3）工具指南：介绍如何使用特定软件工具来完成每个步骤

（4）概念：一些关键概念如迭代、阶段、制品、风险、性能测试

**4.**

RUP的阶段和里程碑在时间轴上的分布顺序：

初始

细化

构造

移交

时间

生命周期

目标里程碑

生命周期

框架里程碑

最初运作

能力里程碑

产品发布

里程碑

第4题图：RUP的阶段和里程碑在时间轴上的分布顺序

**5.**初始阶段：

* 确定最终产品的构想及其业务用例、并定义项目范围。
* 初始阶段以生命周期目标(LCO)里程碑为结束点。
* 令所有的项目相关人员对生命周期目标达成一致意见。

细化阶段：

* 计划出必须完成的活动和需要的资源；详细说明产品特性并设计架构。
* 细化阶段以生命周期构架(LCA)里程碑为结束点。
* 分析问题领域、建立合理的构架基础、确定项目计划、评价项目最有可能出现的风险因素。

构造阶段：

* 构造整个产品，逐步完善视图、构架和计划，直到产品(完整的构想)已完全准备好交付给用户。
* 构造阶段以最初运作能力(IOC)里程碑为结束点。
* 所有保留下来的构件和应用程序特征将被开发并集成以形成产品，而后所有的特征将被彻底测试。

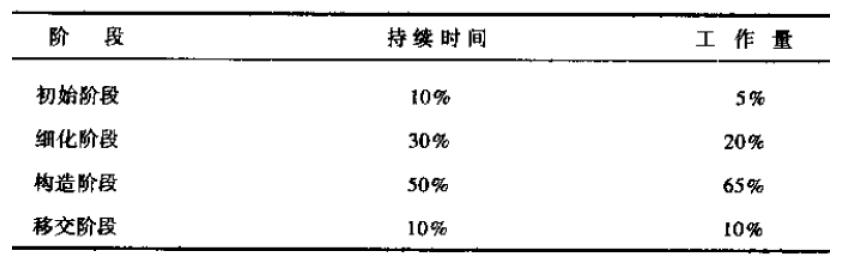
移交阶段：

* 移交产品给用户，包括制造、交付、培训、支持及维护产品，直至用户满意。
* 移交阶段以产品发布版本里程碑为结束点，这也是整个周期的结束点。
* 软件产品移交给用户群，具体包括：
* Beta测试，确认系统与用户的期望是否一致
* 平行操作项目将要替代的遗留系统(选作)
* 转换运行的数据库(选作)
* 培训用户和维护人员
* 产品的首次展示

**6.**RUP的典型项目剖面图：



项目各阶段持续时间和工作量的相对比例：



**7.**框架：框架是一种特殊的软件，它并不能提供完整无缺的解决方案，而是为构建解决方案提供良好的基础。框架是一种半成品。典型地，框架是系统或子系统的半成品；框架中的服务能被最终应用系统直接调用，而框架中的扩展点是供应用开发人员定制的“可变化点”。

构架：构架就是当你去掉任何部分，就无法使其他人理解整个系统和解释它是如何工作的系统描述。

构架要让设计师、程序员、使用者和经理能完成以下任务：

* 理解这个系统是做什么的
* 理解系统是怎样工作的
* 能够完成系统的一部分工作
* 扩展系统
* 对系统部分重用，从而建立另一系统。

构架要对以下四个方面作出决策：

* 软件系统的组织
* 选择构成系统的结构元素和它们之间的接口，以及当这些元素相互协作时所体现出的行为。
* 如何组合这些元素，使它们逐渐成为更大的子系统。
* 构架风格，它将指导系统组织及其元素、它们之间的接口、协作和构成。

对构架概念的解释：

* 软件构架不仅仅注重软件本身的结构和行为，还注重其他特性：使用性、功能性、性能、弹性、重用、可理解性、经济和技术限制及折中方案、美学等。
* 构架是设计的一部分，它决定了如何建立系统。但它不是全部的设计，它只涉及一些重要元素——那些对系统质量有着普遍的深远影响的元素。
* 构架是关于结构和组织的，但是也处理行为。
* 构架不仅仅关心系统内部，还着眼于系统的两个外部语境：操作语境(它的最终用户)和开发语境(开发系统的技术人员)。它不仅包含系统的技术问题，还包含经济和社会问题。
* 构架同时还重视一些“软”问题，如风格和美学。

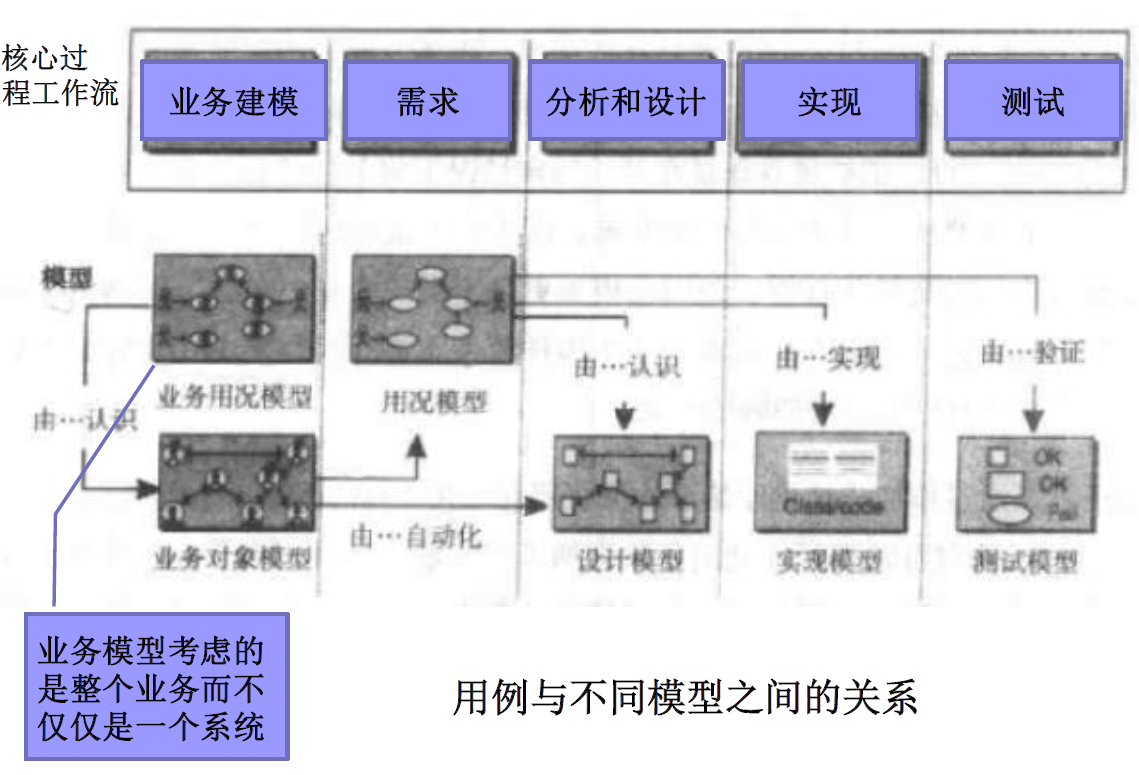
模型和框架视图的区别：

* 模型是系统的完整表达，不同的需求要创建不同的模型来表述
* 构架视图只关注对构架重要的方面
* 不是所有的设计都是构架

**8.**用例模型在不同阶段的作用：

* 用例模型是需求工作流的结果
* 在分析和设计阶段，用例是将需求活动和设计活动连接起来的桥梁，在用例实现中充当基础。通过遍历用例，开发人员会找到对象和类。
* 在实现阶段，设计模型是实现规格说明。用例是设计模型的基础，我们根据设计类实现用例。在设计模型中实现用例可以理解系统的动态性，并确定在哪里来优化系统性能。
* 在测试阶段，用例构成了确定测试用例和测试规程的基础。每一个用例都用来验证系统。
* 在实施阶段，用例包可以用来计划阶段性实施，还可以定义系统变量。
* 用户界面的定义和原型设计也来源于以用例记事板形式出现的用例。
* 在业务建模中也使用用例概念。但是业务模型考虑的是整个业务而不仅仅是一个系统。

用例模型和其它模型的关系：



**二、综合题**