2015A

> 一、敏捷方法的特征有哪些？哪些关于敏捷特征的说法施加于敏捷方法之上是不合适的？为什么？

敏捷方法最主要体现是：小周期迭代式持续交付。遵循如下价值观：↵

1 小周期迭代 2 快速响应变更 3 价值交付 4 自动化 （也是 scrum 的特征)↵

- **敏捷软件开发宣言的 4 个简单的价值观：** ↵
  - **个体和交互 胜过 过程和工具**↵
  - **可以工作的软件 胜过 面面俱到的文档** ↵
  - **客户合作 胜过 合同谈判** ↵
  - **响应变化 胜过 遵循计划** ↵

关于敏捷方法的一些特征表述可能带着一项的误导，例如：↵

轻量级方法：这是对以 XP 为代表的一类方法的误导，事实上，这类方法对工程规范有着极为严格的要求；↵

拥抱变更、变更驱动：仅仅是口号，对待变更，所有软件工程方法都是限制和管理的态度。↵

TDD 可以提供更高的开发质量：并没有足够的证据支持(在开发功能代码之前,先编写单元测试用例代码)。↵

> 五、如果对质量的追求是无止境的，在不考虑资源和成本的前提之下，有哪些可能有效的 策略?

2015 年答案↵

✓ → 重视测试，并且将测试过程文档化并且稳定化；↵

✓ → 重视小组评审，同样定义评审过程，并且使得评审过程的 performance 稳定化↵

✓ → 重视个人评审，提升评审者技能；↵

✓ → 重视设计↵

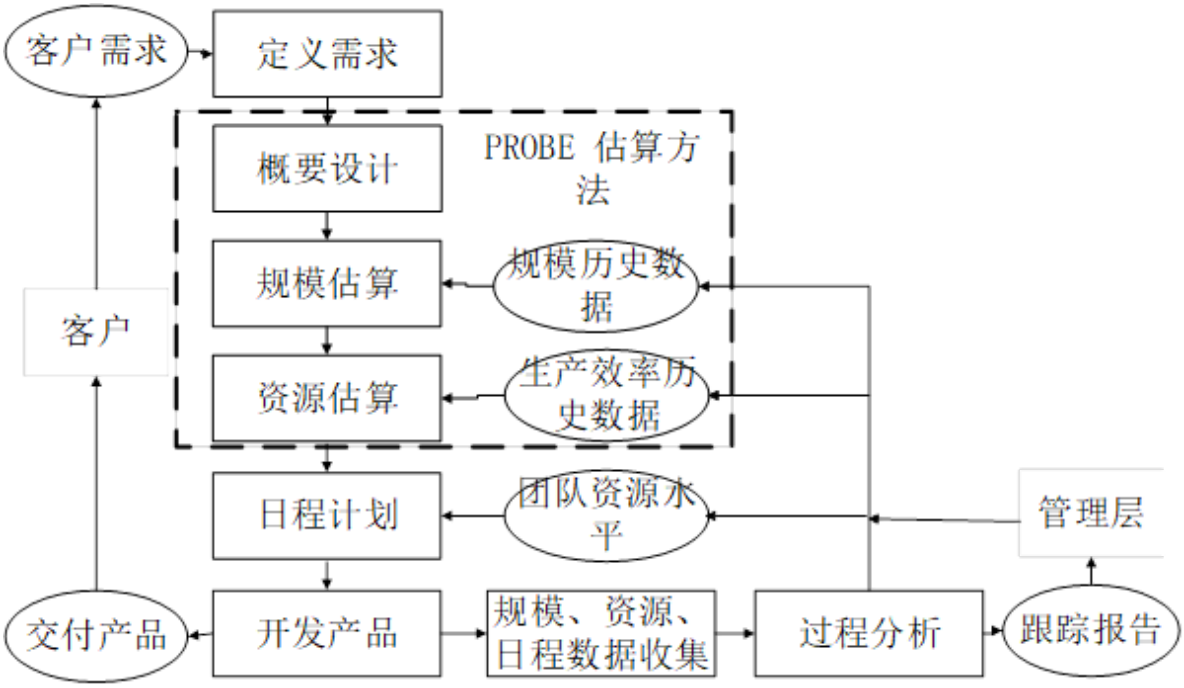
✓ → 开展设计验证↵

2013答案

- → 测试+评审+设计：测试，小组评审，个人评审，设计，设计验证↵
- → 质量策略↵
  - → 首先确保基本没有缺陷，然后再考察其他的质量目标。↵
  - → 用缺陷管理来代替质量管理↵
  - → 高质量的产品也就意味着要求组成软件产品的各个组件基本无缺陷↵
  - → 各个组件的高质量是通过高质量评审来实现的↵

> 六、如何开发一份让人无法拒绝的日程计划？请描述其基本步骤和一些注意事项。

2014年答案：这种日程计划的关键是必须用正推的方式来制订项目计划。一个典型的项目计划框架如下



在这个过程中，除了概要设计和资源估算之外，其他环节尽可能自动化完成。充分参考历史数据进行估算。

> 九、请描述CMMI模型的5个等级的特征，并且解释为何CMMI模型不应该是敏捷方法的对 立面

1. *Initial* 原始级别，开发相对混乱，依赖个人英雄主义，没有过程概念，救火文化盛行；
2. *Managed* 已经管理级别，项目小组级体现着项目管理的特征，有项目计划和跟踪，需求管理、配置管理等等；
3. *Defined*，已经定义级别，在公司层有标准流程和相应的裁剪规范，每个项目小组可以据此定义自己的过程，使得优秀的做法可以在公司层共享；
4. *Quantitatively Managed*, 定量管理，构建预测模型，已统计过程控制的手段来管理过程项目；
5. *Optimizing* 持续优化，继续应用统计方法识别过程偏差，找到问题根源并消除，避免未来继续发生类似问题。

（5分）

*CMMI*模型本身并不是开发模型，而是一个过程改进的模型，刻画了软件组织从不成熟到成熟的路线图，简单说，*CMMI*模型不是一种具体的开发方法。而大部分所谓的敏捷方法都是开发方法，因此，两者是完全不同性质的事物，将这两者对立是不合适的。

> 十、请结合A/FR、PQI、Review Rate、DRL、Yield尽可能具体描述一个软件项目应该如何从多方面来确保开发的高质量

这些指标既是开发过程中质量管理的一些参考指标，同时也体现在计划安排中应该注意的质量元素。具体如下：

1. 在项目计划过程中应该安排确保高质量开发结果的活动，例如，按照*A/FR*、*PQI*等指标的要求，安排对各类产物（文档和代码）的个人评审和小组评审；
2. 这些评审活动应该满足一定的要求，特别体现在时间方面。例如，评审时间应该多于测试时间的两倍以上（*A/FR*）；评审时间应该是相应开放时间的*50%*以上（*PQI*）；评审速度要求（*Review Rate*）等
3. 充分借鉴质量指标所体现的开发质量状况，尽早制订相应的质量补救措施。例如，*PQI*所体现的缺陷密度、所有上述指标的参考值等等。一旦超标，往往意味着质量方面有偏差，应当及时补救。
4. 利用*yield*等指标，构建质量预测模型，更加积极（*Proactive*）地判定和控制开发质量；
5. 依据*PQI*和*Yield*指标所体现的信息，通过过程改进来提升开发质量。

2015B

> 四、请结合SCRUM这种敏捷方法论述敏捷方法应该具备的特征？同时解释为何常见的若干种描述敏捷方法对立面的方法的特征（例如，严格、重型、计划驱动等等）并不合适

1. 小周期迭代↵
2. 快速响应变更↵
3. 价值交付↵
4. 自动化↵

严格：所有优秀的工程方法和实践都是严格的。↵

重型：如上，此外，轻量级和重型其实并没有刻画具体方法，何为重型，并没有严格定义；而且，对于变更这件事情，几乎所有方法都是限制，因此，很难说敏捷方法是轻量级方法。↵

计划驱动：所有正式的项目都是计划驱动的，否则计划的作用无法体现↵

> 九、请解释A/FR， PQI的计算方式，并且解释这两个指标有什么用途？

- A/FR = PSP 质检成本/PSP 失效成本；
- 用途
  - 理论上，A/FR 的值越大，往往意味着越高的质量。
  - 过高的 A/FR 往往意味着做了过多的评审，反而会导致开发效率的下降。作为指南，
  - 在 PSP 中 A/FR 的期望值就是 2.0
- PQI 为 5 个数据乘积
  - 设计质量：设计的时间应该大于编码的时间
  - 设计评审质量：设计评审的时间应该大于设计时间的 50%
  - 代码评审质量：代码评审时间应该大于编码时间的 50%
  - 代码质量：代码的编译缺陷密度应当小于 10 个/千行
  - 程序质量：代码单元测试缺陷密度应当小于 5 个/千行
- 用途
  - 判断模块开发质量
  - 规划质量活动计划
  - 过程改进

> 十、请分别描述PDCA模型和IDEAL模型的主要步骤?

plan do check action↵

●→八个步骤↵

1.→分析现状，找出问题； ↵

2.→分析影响质量的原因； ↵

3.→找出措施； ↵

4.→拟定措施计划； ↵

●→为什么要制定这个措施？ ↵

●→达到什么目标？ ↵

●→在何处执行？ ↵

●→由谁负责完成？ ↵

●→什么时间完成？ ↵

●→怎样执行？ ↵

5.→执行措施，执行计划； ↵

6.→检查效果，发现问题； ↵

7.→总结经验，纳入标准； ↵

8.→遗留问题转入下期 PDCA 循环。 ↵

图 1.20 戴明的 PDCA 循环示意图

●→IDEAL 模型解决了软件组织在各种质量改进环境下的需要。它包括了软件过程改进周期中的五个阶段，IDEAL 是代表这五个阶段的单词的首字母。 ↵

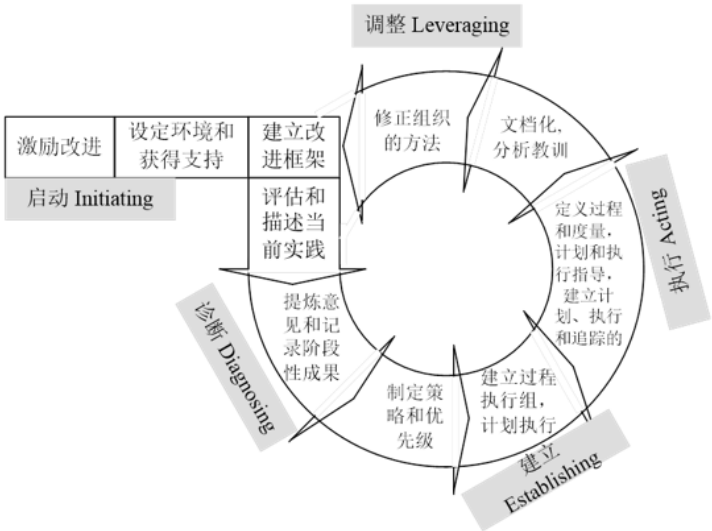
■→I: Initiating 开始 ↵

■→D: Diagnosing 诊断、评价↵

■→E: Establishing 建立↵

■→A: Acting 执行↵

■→L: Leveraging 调整↵



↵

↵

↵

↵

↵

↵

↵

↵

↵

↵

↵

十一、请描述一下度量和分析活动在一个软件项目当中的作用，以及应该如何正确开展？（本题满分10分）

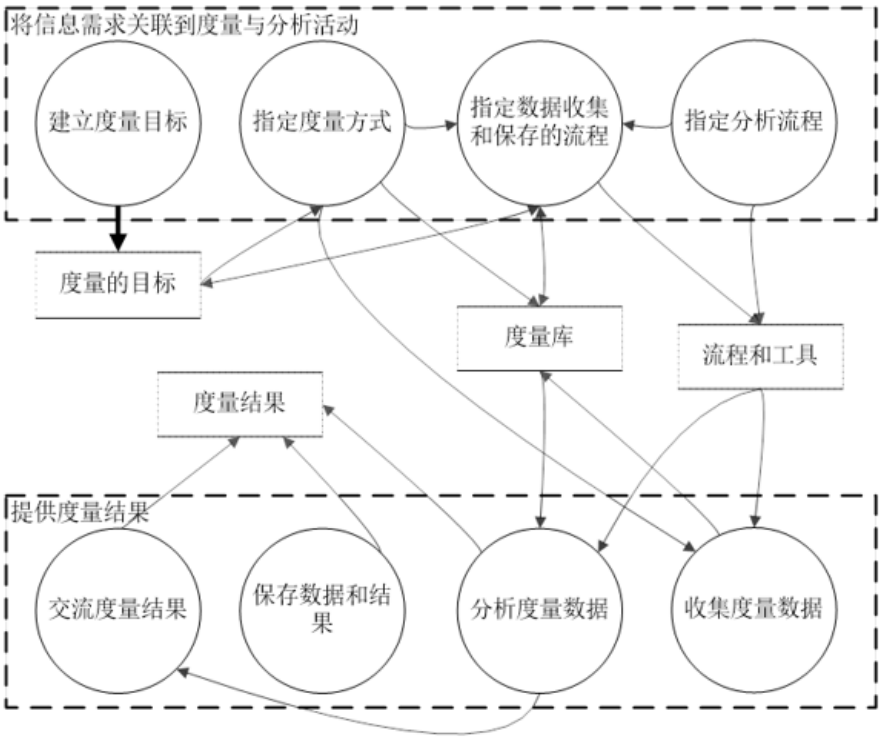
●→作为项目管理支持类的活动，度量和分析活动可以支持如下的项目管理活动

■→客观的估计与计划↵

■→根据建立的计划和目标，跟踪实际进展↵

■→识别与解决过程改进相关议题↵

■→提供将度量结果纳入未来其他过程的基础↵



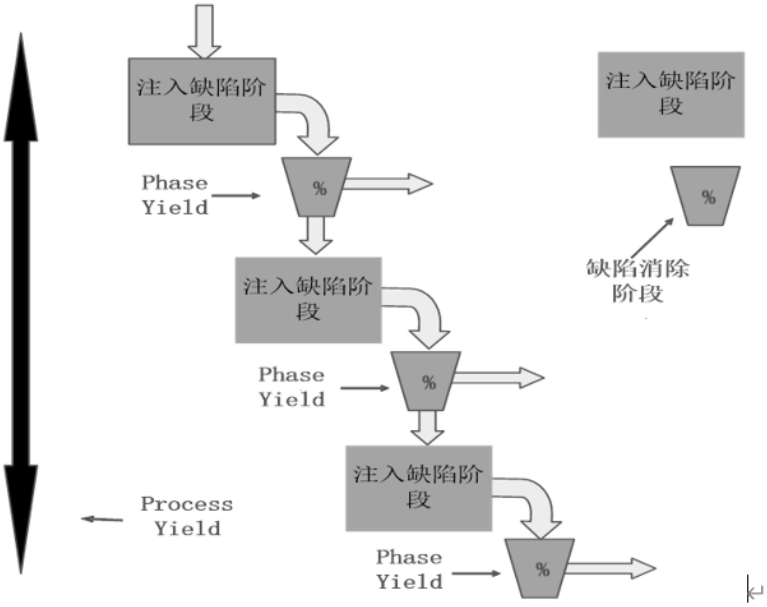
或者可以解释GQM

2013

> 1. 请结合GQM思想解释PSP过程的基本度量元有哪几个

- GQM
  - 概念层（目标）：目标是为某个特定的对象而定义的。这里的对象是指软件产品、软件过程以及相关的资源等。
  - 操作层（问题）：基于一定的刻画上述目标是否达成或者目标达成的进展情况的模型，使用一系列的问题来定义所研究的对象，然后得出评价或评估特定目标达成进展情况。所选择的问题应当尽量体现质量相关的话题。
  - 量化层（度量）：试图以量化的方式回答上述操作层识别出来的问题。
- PSP基本度量项
  - 度量时间：所属阶段、开始时间、结束时间、中断时间、净时间
  - 度量缺陷：发现日期、注入阶段、消除阶段、消除时间、关联缺陷
  - 度量规模：
    - 选择的规模度量方式必须反映开发成本
    - 必须精确定义
    - 必须能有自动化方法统计
    - 有助于早期规划
  - 日程（TSP）

> 3. 请给予Yield度量指标构建缺陷预测模型，并列举该模型的可能改进方案



由上图，只需要知道注入阶段缺陷注入水平（速度或者密度）以及消除阶段的缺陷消除水平（速度或者能力）就可以基于 yield 构建一个基本的缺陷预测模型。

可能的改进是假设注入水平和消除水平都符合正态分布，因此，可以用蒙特卡罗方法模拟结果。

> 6. 请解释配置管理中配置项和产品基线的概念，并设计一个流程对单元测试后 已经纳入配置库的代码，修改集成测试中的若干错误后，该如何控制变更。

- 配置项是在配置管理当中作为单独实体进行管理和控制的工作产品集合
- 基线是一个或多个配置项及相关的标识符的代表，是一组经正式审查同意的规格或工作产品集合，是未来开发工作或交付的基础，而且只能经由严格的变更控制程序才能改变。
- 如何控制变更
  - 跟踪变更请求：1）启动变更请求处理程序，将变更情况保存在变更请求数据库中；2）分析变更建议和所需进行的修改将对工作产品、进度、日程等造成的影响；3）如果变更请求影响到其他基线，则与相关的干系人一起审查这些变更请求，并取得他们的同意；4）跟踪变更申请直到结项
  - 控制配置项变更：1）确认这些修订已得授权；2）更新配置项；3）将旧基线归档保存，并获取新基线；4）执行审查，确保该变更没有对基线造成意料外的影响；5）上当记录配置项的变更内容和变更理由

> 8. 如何制定一份让人无法拒绝的计划，请描述基本步骤和一些注意事项

- 日程计划
  - 制定任务计划和日程计划；前者描述项目所有的任务清单，任务之间的先后顺序、以及每个任务所需时间资源，后者描述了各个任务在日程上的安排，哪天开始哪天结束；制定资源计划，结合任务计划即可定义日程计划
- 质量计划
  - ➢ 项目的质量计划中应当确定需要开展的质量保证活动。
  - ➢ 典型的质量保证活动包括个人评审、团队评审、单元测试、集成测试、系统测试以及验收测试等。
  - ➢ 在质量计划中需要解决的关键的问题是该开展哪些活动，以及这些活动开展的程度，如时间、人数和目标分别是什么。
- 风险计划

  目的是在风险发生前，识别出潜在的问题，以消除潜在问题的负面影响
  - 风险识别
    - ◇ 识别与成本、进度及绩效相关的风险
    - ◇ 审查可能影响项目的环境因素
    - ◇ 审查工作分解结构的所有组件，作为风险识别的一部分，以协助确保所有的工作投入均已考虑
    - ◇ 审查项目计划的所有组件，作为风险识别的一部分，以确保项目在各方面均已考虑
    - ◇ 记录风险的内容、条件及可能的结果
    - ◇ 识别每一风险相关的干系人
    - ◇ 利用已定义的风险参数，评估已识别的风险
    - ◇ 依照定义的风险类别，将风险分类并分组
    - ◇ 排列降低风险的优先级
  - 风险应对

    识别风险之后，就应当制定相应的风险管理策略，以应对各类风险
  - 典型策略：风险转嫁，风险解决，风险缓解

> 9. 请分别介绍解释Deming、Crosby以及Juran等三位质量管理大师主要贡献以 及他们的工作对软件过程和项目管理的借鉴意义。

Deming：

1） 提出质量改进的思想，并被二战后的日本工业界所采用；

2） 提出PDCA循环，被称为“戴明环”（Plan、Do、Check、Action），为最基本的质量和管理工具；

3） 提出十四点原则（树立改进产品和服务的坚定目标；采用新的思维方法；停止依赖检验的办法获得质量；不再凭价格标签进货；坚持不懈地提高产品质量和生产率；岗位培训制度化；管理者的作用应突出强调；排除畏难情绪；打破部门和人员之间的障碍；不再给操作人员提空洞的口号；取消对操作人员规定的工作定额和指标；不再采用按年度对人员工件进行评估；创建积极的自我提高计划制度；让每个员工都投入到提高产品质量的活动中去）。

Crosby：

1） 提出了“零缺陷”的概念，即第一次就把事情做对。而要做到这一点，就需要把工作放在预防上而不是质量检验上；

2） 提出了质量管理的绝对性：

- 质量就是符合要求，而不是“完美”
- 质量来自于预防，而不是检验
- 质量的标准是“零缺陷”，而不是可接受质量水平
- 质量的衡量标准是“不符合要求的代价”

3） 提出质量改进的基本要素（6C-变革管理的六个阶段）：

- 领悟（Comprehension）——理解质量真谛
- 承诺（commitment）——制定质量策略的决心
- 能力（capability）——教育与培训
- 沟通（communication）——成功的经验文档化、制度化
- 改正（correction）——预防与提高绩效
- 坚持（continuance）——强调质量管理成为一种工作方式

4） 发展质量成熟度的度量。

Juran：

1） 其主编的《质量控制手册》为当今世界质量控制科学的“圣经”，奠定了“全面质量管理TQM”的理论基础（Total Quality Management）；

2） 提出了适用性质量（质量是一种适用性，即产品在使用期间能满足使用者的要求），质量不仅要满足明确的需要，也要满足潜在的需求。这一思想使质量管理范围从生产过程中的控制进一步扩大到产品开发和工艺设计阶段，即挖掘用户潜在需求；

3） 提出了质量三步曲（质量计划->质量控制->质量改进）；

4） 提出了Juran质量螺旋；

5） 提出了80/20原则，认为有80%的质量问题是由于领导责任引起的。从而将人力与质量管理结合起来。

Shewhart：（补充）

1） 最早将统计控制的思想引入质量管理，为质量改进奠基人；

2） 提出PDS模型（计划-执行-检查 Plan-Do-See），后被戴明进一步发展为PDCA。

Humphrey：（补充）

1）采用Crosby的成熟度度量，提出了软件能力成熟度模型（CMM），对于软件过程管理与改进具有建设性作用。

2014

> 1. 过程度量项定义时应当注意的方面，据此评价PSP基本度量元的合理之处

规模度量的方式，最终↵

必须反映开发的成本↵

必须精确↵

必须能用自动化的方法来统计↵

必须有助于早期规划↵

↵

过程度量在过程管理和改进中起着极为重要的作用↵

PSP 的基本度量项：时间，缺陷，规模，日志（TSP）↵

- ■→ 度量时间：所属阶段、开始时间、结束时间、中断时间、净时间↵
- ■→ 度量缺陷：发现日期、注入阶段、消除阶段、消除时间、关联缺陷↵
- ■→ 度量规模：↵

多个度量值的组合，可以回答大部分软件开发的问题↵

什么时候完成，质量如何，成本多少。这三个问题，是软件项目管理的三个主要问题。

> 2. 解释客户需求与产品需求的差别，并设计一个流程来完成需求开发工作

客户需求描述的是客户的期望。往往表现为，客户在实际工作中碰到了一些具体问题，希望通过某个东西来帮忙解决这些问题。客户的这种解决问题的愿望，往往就表述为客户需求。比如，客户希望有一种快速进行数据计算的工具帮助他/她完成繁琐的计算工作。这就是一个客户需求。↵

客户需求可能很简单，也可能很复杂；可能很清晰，也可能很模糊。这就需要开发团队与客户一起进行交流、协商，从而弄清客户的真正意图。↵

产品需求描述的是开发团队所提供的解决方案。即针对上述的客户需求，开发团队设计出一个可以帮助客户解决工作当中碰到的问题的方案。↵

一个流程：↵

> 4. 如何对某个软件产品组件进行质量评价，可以选择哪些度量元，这些度量元如何辅助判断产品组件的质量?

1）·为什么要实施度量和分析活动？↵

- → 在软件项目管理决策的过程中，基于客观的数据很重要，这种客观决策可以显著消除错误决策的风险。而这些客观数据的获得，必须依照一定的流程以正确的方式获得和使用。度量和分析活动就定义了上述客观数据的获取与使用方式↵
- → 度量和分析可以支持如下项目管理活动：客观的估计与计划；根据建立的计划和目标，跟踪实际进展；识别与解决过程改进相关议题；提供将度量结果纳入未来其他过程的基础↵

↵

- → 度量时间：所属阶段、开始时间、结束时间、中断时间、净时间↵
- → 度量缺陷：发现日期、注入阶段、消除阶段、消除时间、关联缺陷↵
- → 度量规模：↵

> 6 为何说将规范方法、计划驱动方法等特征作为敏捷方法的对立面有很大的误导性质？如何通过多种维度改进这种对各类开发过程的理解？

●→敏捷与规范，软件开发中看似对立的两个属性，实际上相得益彰。↵

●→计划驱动的开发人员必须敏捷，敏捷开发人员必须规范。成功的关键在于找到两者的平衡点。↵

●→这个平衡点随项目所处的环境以及所涉及的风险而变化。仅凭一腔热情径直地采用极端方法的开发人员，必须学会如何根据实际情况恰当地平衡敏捷与规范。↵

> 7 举例说明验证和确认的区别与联系

●→验证(Verification)和确认(Validation)都是为了提升最终产品的质量而采取的措施。

●→验证和确认的目的不同。↵

- → 验证是目的是确保选定的工作产品与事先指定给该工作产品的需求一致。↵
- → 确认的目标则是确保开发完成的产品或者产品组件在即将要使用该产品或者产品组件的环境中工作正确。↵

> 8 应对风险的典型策略有哪些？举例说明

识别风险之后，就应当制定相应的风险管理策略，以应对各类风险↵

典型策略：风险转嫁，风险解决，风险缓解↵

> 9 区分质量管理和缺陷管理的联系和差异，解释为何在软件开发中将质量和生产效率两者进行妥协不合适

●→1964 年，Crosby 提出了“零缺陷”的概念，即第一次就把事情做对。↵

■→1）质量管理的绝对性↵

◆→（1）质量就是符合要求，而不是“完美”。↵

◆→（2）质量来自于预防，而不是检验。 ↵

◆→（3）质量的标准是“零缺陷”，而不是可接受质量水平。↵

◆→（4）质量的衡量标准是“不符合要求的代价”↵

Devops

> 摩尔定律和反摩尔定律分别是什么？对软件的发展有什么意义

1. 摩尔定律：集成电路每18个月集成度会成倍增加，计算机的产品性能会提升一倍。

   在摩尔定律主导下的科技公司主要有一下的特点：

   1. 科技公司必须在较短的时间内研发出下一代的产品，以提高产品的性品的性能；
   2. 研发的产品必须着眼于下一代的产品框架，其容量性能是常常是以10倍左右来设计。
2. 反摩尔定律：在摩尔定律作用下，例如产品性能增加n倍，卖出同样多的产品，收入将是产品性能增加倍数的1/n.

   1. 这个定律刺激着科技公司必须投入大量的研发费用跟上时代的潮流
   2. 新技术的变让大公司要时刻注意周遭技术的变革，跟上时代的方法直接投资甚至收购出现新技术的创业公司来保证自身对行业的掌控力。

> 软件架构的演化过程？微服务对devops的意义

1. 演化：C/S架构 -> B/S架构 -> 服务端MVC架构 -> 客户端MVC架构 -> 服务端-N层架构(展示层、业务层、持久层、数据库) -> SOA -> 微服务架构
2. 微服务

   1. DevOps是微服务实施的充分必要条件。
   2. 微服务是DevOps实施的充分但不必要条件
   3. | | | |
      | --- | --- | --- |
      | **微服务** | • **简单**：单个服务简单，只关注于一个业务功能。
• **团队独立性**：每个微服务可以由不同的团队独立开发。
• **松耦合**：微服务是松散耦合的。
• **平台无关性**：微服务可以用不同的语言/工具开发。
• **通信协议轻量级**：使用REST或者RPC进行服务间通信 | • 运维成本过高
• 分布式系统的复杂性
• 异步，消息与**并行**方式使得系统的开发门槛增加
• 分布式系统的复杂性**也会让**系统的测试变得复杂 |

> 构建一个基本的devops pipeline需要哪些软件和工具

4. 下列工具在DevOps Pipeline的作用

   1. Docker：容器将一个软件封装在一个完整的文件系统中，该文件系统包含它运行所需的一切：代码、运行时、系统工具、系统库——任何你可以安装在服务器上的东西。这保证了它总是以相同的方式运行，而不管它在什么环境中运行
   2. Git：分布式版本控制系统，强调速度、数据完整性和对分布式非线性工作流的支持
   3. Jenkins：持续集成工具，配置流水线等
   4. SonarQube：静态代码检查工具，用来静态的检查软件

> 软件项目管理和软件过程管理在管理对象，实现目标和参考模型三个角度的区别

10.3. 软件项目管理与软件过程管理

1. 软件项目管理(产品生产管理，关注各个目标(成本、质量等)有没有实现，着眼于**一个特定的项目**)：包括估算、计划、跟踪、风险管理、范围管理、人员管理、沟通管理等。
    1. 三要素：目标、状态、纠偏
    2. 三目标：成本、质量、工期
    3. 是应用方法、工具、技术和人员能力来完成软件项目，实现软件目标的过程。
2. 软件过程管理(关注研发流程和生产效率)
    1. 管理对象：软件过程
    2. 关注研发的流程，关注**流程的输出效率和质量**，不关注某个特定的项目
    3. 目的是为了让软件过程在开发效率、质量等方面有更好的性能绩效。
    4. 包括CMM/CMMI和SPICE模型

> 什么是迭代式开发？有什么特点和优势？

迭代式开发也被称作迭代增量式开发或迭代进化式开发，是一种与传统的瀑布式开发相反的软件开发过程，它弥补了传统开发方式中的一些弱点，具有更高的成功率和生产率。

优点

1.降低风险

2.得到早期用户反馈

3.持续的测试和集成

4.使用变更

5.提高复用性

开发特征

1.在进行大规模的投资之前就解决了关键的风险分析。

2.使得早期的用户反馈在初始迭代中就能出现。

3.对各个目标里程碑提供了短期的焦点（阶段性的中心）。

4.对过程的测量是通过对实现的评定（而不仅仅是文档）来进行的。

5.可以对局部的实现进行部署。