2021 年 6 月

往年卷与非标准答案

pp

2015

1. 软件架构从哪来？列举五个可能的来源
    1. 功能性需求
    2. 非功能性需求
    3. 架构重要需求
    4. 质量需求
    5. 涉众
    6. 开发组织
    7. 技术环境
    8. 商业因素
2. **软件产品线的体系结构与简单产品的体系结构有什么区别
    1. 产品线的目的是实现高重用性、高可修改性；之所以如此有效，是因为可以通过重用，充分利用产品的共性，从而产生生产的经济性
    2. 在产品线架构中有一组明确允许发生的变化，但是对于常规架构来说，只要满足了单个系统的行为和质量目标，几乎任何实例都是可以的
3. ****如何对质量属性场景建模？以“刺激-响应”的形式，给这两个质量属性图形化建模：可用性、性能
    1. 刺激源
    2. 刺激
    3. 环境
    4. 制品
    5. 响应

    6. 响应度量

4. 描述架构模式和战术之间的关系，列举四个战术名称并描述他们的用途

    1. 战术比模式简单，它们使用单一的结构或机制来处理单一的架构要求
    2. 模式把多个设计决定组合在一起
    3. 都是架构师的主要工具
    4. 战术是模式设计和创造的基石
    5. 大多数模式由几个不同的战术组成

5. **简要描述在软件架构过程中涉及的一般活动

    1. 创造系统的商业案例
    2. 理解需求
    3. 创造和选择架构
    4. 与包括开发者在内的涉众沟通架构
    5. 分析或评估架构
        1. 总的方法
        2. 质量特定技术
    6. 实现架构
    7. 确保架构符合要求

6. ****连线，并为每种样式列出 4 个视图

    1. 它如何被构造成一组样式单元？模块样式
        1. 分解视图、使用视图、概括视图、分层视图、方面视图、数据模型视图、逻辑视图
    2. 它如何被构造成一组有运行时行为和交互的元素？组件-连接器样式
        1. 管道-过滤器视图、客户机-服务端视图、点对点视图、面向服务架构视图、发布-订阅视图、共享数据视图、多层视图
    3. 它如何与它环境中的非软件结构关联？分配样式

    1. 部署视图、安装视图、工作安排视图、开发视图、物理视图

7. **解释代理人架构模式的上下文、优点和限制

    1. 定义了一个叫做代理人的运行时组件，它在多个客户机和服务器之间进行通。包括客户机、服务器、代理人元素，可能还有客户机端代理和服务器端代理。连接关系将客户机（与可选的客户机端代理）和服务器（与可选的服务器端代理）与代理人相关联，只允许客户机连接到代理人、服务器连接到代理人（或分别通过客户机端代理和服务器端代理）

    2. 降低客户机和服务器间的耦合；提高服务的位置透明性；提高组件的可变性、扩展性、伸缩性和可重用性

    3. 代理人在客户机和服务器之间增加了一层间接寻址，导致延迟，可能是性能瓶颈；代理人可能是一个单一的失败点；代理人增加了前期的复杂度；代理人可能是安全攻击的目标；代理人可能难于测试

8. ****为什么软件架构要用不同的视图来文档化？举例 4 个视图并给出名称和目的

    1. 视图是一组系统元素及其关系的表示，这些元素不一定是全部系统元素，而是特定类型的元素。视图让我们将系统实体划分成感兴趣和易于管理的系统表示。不同的视图支持不同的目标和用户，并凸显出不同系统元素和关系。不同视图在不同程度上展现不同的质量属性

    2. 视图

        1. 逻辑视图：描述架构中重要的元素及其之间的关系

        2. 进程视图：描述架构的并发和通信元素

        3. 物理视图：描述主要过程和元素是如何被映射到应用程序硬件

        4. 开发视图：捕获软件组件的内部组织

        5. 架构用例：捕获架构的需求，与多个特定视图关联

9. 简要描述 **SOA 的基本原则，讨论 SOA 对质量属性的影响，比如互操作性、可伸缩性和安全性

    1. 原则

        1. 服务解耦：服务之间的关系最小化，只是相互知道接口

    2. 服务契约：服务按照描述文档所定义的服务契约行事
    3. 服务封装：除了服务契约所描述内容，服务将对外部隐藏实现逻辑
    4. 服务重用：将逻辑分布在不同的服务中，以提高服务的重用性
    5. 服务组合：一组服务可以协调工作，组合起来形成定制组合业务需求
    6. 服务自治：服务对所封装的逻辑具有控制权
    7. 服务无状态：服务将一个活动所需保存的资讯最小化
2. 对质量属性
    1. 良好的互操作性，符合开放标准
    2. 服务动态识别、注册、调用，可伸缩性高
    3. 模组化，可重用性高
    4. 服务自身高内聚、服务间松耦合，最小化开发维护中的相互影响，可修改性高
    5. 单个服务的规模变小，可维护性高
    6. 使用消息机制及异步机制，可靠性高
    7. 分布式系统，可扩展性、可用性高
    8. 各独立服务演化不可控，安全性不高
    9. 难以测试验证，可测试性不高
10. ***描述 ATAM 过程每个阶段产生的输出
    1. 合作与准备：评估团队领导和主要项目决策者
        1. 谁：涉众的初步名单
        2. 逻辑：何时？何地？如何?
        3. 评估报告何时交付给何人
        4. 评估报告包含何种信息
    2. 评估 1：评估团队和项目决策者

    1. 架构的简短展示
    2. 业务目标的表达（驱动因素）
    3. 作为场景实现的特定质量属性需求的优先级列表
    4. 效用树
    5. 风险和非风险
    6. 敏感点和权衡点
3. 评估 2：评估团队、项目决策者和架构涉众
    1. 涉众社区的优先级场景列表
    2. 风险主题和受到威胁的业务驱动因素
4. 后续行动：评估团队和主要涉众
    1. 最终评估报告

11. 为什么软件产品线和模型驱动架构有高可重用性？比较并讨论他们的共同点和不同点
12. **分布式缓存更新

2017

1. **简要描述软件架构过程中的一般活动，以及每个活动的主要输入和输出
2. **软件产品线的体系结构与单个产品的体系结构有什么区别
3. **在软件设计中应用的一般设计策略是什么？给出每个策略的软件架构的简明工作示例
    1. 分解
        1. 将质量属性需求分解并分配给元素，使满足给定的约束和安排，实现系统的质量和业务目标
    2. 抽象
        1. 凸显问题
    3. 逐步地：分治

    1. 逐步地设计多个 ASR

4. 生成和测试

    1. 生成当前的设计，看作一个假设

    2. 测试当前假设中的错误

    3. 在下一个设计假设中修正当前的错误，保留正确的东西

5. 迭代：递增地改进

    1. 不断地重复生成和测试来进行迭代

6. 可重用元素

    1. 提取设计中可以重用的元素

4. ****如何对质量属性场景建模？以“刺激-响应”的形式，给这两个质量属性图形化建模：可用性、可修改性

5. ***描述 ATAM 过程每个阶段产生的输出

6. ****连线，并为每种类别的样式列出 4 个视图

7. **什么是 ASR？ 列出四个提取和识别 ASR 的来源和方法

    1. Architecture Significant Requirement，架构重要需求，是一种在架构上有深刻影响的需求，如果没有这种需求，架构可能会有很大的不同。质量属性需求越困难、越重要，就越有可能对架构产生重大影响，因此成为 ASR

    2. 从需求文档收集：如果需求影响了关键架构设计决定的制定，根据定义就是 ASR

    3. 采访涉众：质量属性研讨会，包括一个架构驱动因素列表和一组 QA 场景，这些场景由涉众（作为一个优先组）确定

    4. 理解业务目标

    5. 从质量属性效用树捕获 ASR

8. 请说出至少三个面向对象的原则，并解释他们是如何在***策略模式中应用的

    1. 单一职责原则：策略模式中，每一个具体策略都为了同一个单一的职责

    2. 开闭原则：策略模式中增加新的策略不需要修改原有的代码

3. 里氏代换原则：策略模式中引用父类的地方都可以透明使用子类对象
4. 依赖倒转原则：策略模式中高层模块和细节都依赖于抽象
5. 接口隔离原则
6. 合成复用原则：策略模式使用聚合来复用而不是继承
7. 最小知识原则

9. **典型的软件架构文档包中应该包括什么？简要描述各个组件及其用途
    1. 文档路线图：描述文档中有哪些信息以及在哪里可以找到这些信息
    2. 视图如何被文档化：描述视图的文档结构
        1. 主要展示：显示视图的元素及其关系，通常图形化
        2. 元素目录：详细描述元素及其属性、关系及其属性、元素接口和行为
        3. 上下文图：展示系统和其部分如何与环境关联
        4. 可变性指南：描述该视图如何应对未来架构的任何变化点
        5. 缘由：为什么这个视图反映了设计，提供一个令人信服的论据以说明它是健全的
    3. 系统概览：概要地描述系统
    4. 在视图间映射：描述每种视图的相似和映射
    5. 缘由：描述最终选择的视图的原因
    6. 目录：索引、词汇表、缩略词表
10. ****描述 4+1 视图
11. 软件设计的三个变化维度，每个维度的变化点。不同的绑定时间如何影响可修改性和可测试性

2018

1. 软件架构的关注点有哪些？涉众有哪些
    1. 关注点：需求、质量属性、特性请求、担忧、风险、复杂度（脑力难对付、管理难对付）、适合于环境（与环境一致和谐、与商业策略和用户目的一致）

    1. https://easandbox.wpcomstaging.com/2019/07/15/architecture-concerns/
    2. http://www.bredemeyer.com/pdf_files/ArchitectureDefinition.PDF
2. 涉众：架构师、需求工程师、开发人员、测试人员、集成人员、维护人员、需要互操作的其他系统的设计人员、质量属性专家、经理、产品线经理、质量确保团队、客户、终端用户、产品线应用程序构建者、分析师、基础设施支持人员

2. 软件需求、质量属性、ASR 的区别和联系
    1. 联系：软件需求包括质量属性和架构重要需求；质量属性需求越困难、越重要，就越可能对架构产生重大影响，因此可能是 ASR；ASR 的实现可能会影响质量属性
    2. 区别：软件需求是总体的概念，质量属性是其中一部分内容，是整个系统的期望特性，是在功能需求之上的，强调系统需要达到的质量，ASR 是在架构上重要的需求，强调这个需求对于架构有重大影响；质量属性侧重于描述需求要求的系统质量，ASR 侧重于描述需求可能影响系统架构
3. 组件-连接器样式的本质是什么？以 MVC 模式举例
    1. 组件-连接器样式回答了“如何被构造称一组具有运行时行为和交互的元素”这个问题，显示一些运行时存在的元素，“连接”关系指明了哪些连接器连接到哪些组件，将连接器的端点连接到组件的端点
    2. MVC 模式将系统功能拆分成三个组件：模型、视图和在模型和视图之间中介的控制器，“通知关系”连接了模型、视图和控制器的实例，通知相关状态改变的元素
4. ****如何对质量属性场景建模？以“刺激-响应”的形式，给这两个质量属性图形化建模：可用性、可修改性
5. 风险、敏感点、权衡点是什么？各举一个例子
    1. 风险：可能对所需质量属性产生负面影响的架构决定；增加备份数据库导致性能下降；用户的简单密码是安全性的风险
    2. 敏感点：对于特定质量属性敏感的架构决定；可靠性对于增加备份数据库敏感；用户的简单认证降低安全性

3. 权衡点：影响多个质量属性的架构决定；增加备份数据库让可靠性提升，让性能下降

6. ****连线，并为每种样式列出 4 个视图

7. 分层模式和多层模式的区别
    1. 分层模式是一种模块视图；多层模式是一种分配视图
    2. 分层强调的是单向的层与层之间的关系，每层是一个整体；多层的层是分组，将组件按照类型、环境等进行分类得到的分组，包含很多组件，允许相互访问
    3. 分层强调的是代码的组织形式；多层强调的是代码的执行位置
    4. 分层的每个层通常在同一机器运行；多层的每个层可能在不同的机器上运行

8. 描述 ADD 过程
    1. 确保有充足的需求信息
    2. 选择一个系统元素进行分解
    3. 为选中元素识别 ASR
    4. 选择一个符合 ASR 的设计理念
        1. 识别设计关注点
        2. 为从属关注点列出可选的模式、新战术
        3. 从列表中选择模式、战术
        4. 决定模式、战术与 ASR 之间的关系
        5. 捕捉初步架构视图
        6. 评估和解决不一致
    5. 实例化架构元素并分配职责
    6. 为实例化的元素定义接口
    7. 验证和细化需求，并使他们成为实例化元素的约束
    8. 重复直到所有 ASR 都被满足

9. ****为什么软件架构需要用不同的视图描述？举出四种视图的例子，列出名称和目的
10. **软件产品线架构如何实现可变性？描述可变性机制的工作方式
11. 设计一个飞行模拟软件，要求能模拟多种飞机的特性。为了在将来支持更多飞机种类，要求使用***策略模式。画出架构图和类图
12. ？

2019

1. ****如何对质量属性场景建模？以“刺激-响应”的形式，给这两个质量属性图形化建模：互操作性、可修改性
2. **什么是 ASR？列出四个提取和识别 ASR 的来源和方法
3. ****描述 4+1 视图并画图
4. **在软件设计中应用的一般设计策略是什么？给出每个策略的软件架构的简明工作示例
5. ****连线，并为每种类别的样式列出 4 个视图
6. **典型的软件架构文档包中应该包括什么？简要描述各个组件及其用途
7. ***描述在 ATAM 的每一个过程中，有哪些涉众，职责是什么
8. **软件产品线架构如何实现可变性？描述可变性机制的工作方式，和变化点
9. **解释代理人架构模式的上下文、优点和限制
10. 微服务和 **SOA 的区别、相同点
    1. 区别
        1. SOA 采用企业服务总线通信，微服务采用轻量级通信机制
        2. SOA 的每个服务职责较重，微服务的每个服务职责单一
        3. SOA 强调中央管理，微服务去中心化
        4. 微服务的服务间耦合比 SOA 更小
    2. 相同点

    1. 都采取了分布式组件的形式
    2. 各个组件可以相互服务而无需知道实现细节
    3. 服务自身高内聚、服务间松耦合
    4. 微服务是 SOA 的一种

11. 设计一个卖票系统，不同的角色有不同的打折方案，用***策略模式设计并画图，说明策略模式的使用场景
12. **分布式缓存更新