【软件工程】山东大学软件工程复习要点及答案

原创 shadowingszy 发布于2018-12-25 21:23:44 阅读数 1140 ☆ 收藏

找到这篇文章的人一定被软件工程考试搞得头秃吧。
这篇文章将会让你不再那么头秃~
文章内容仅供参考，如有错误，请指正。

Chapter01

SE的定义、目的、方法及作用?

定义：软件工程即用系统科学的工程性方法解决软件开发时遇到的问题，也就是，将系统化的、严格约束的、可量化的方法应用于软件的开发、运行和
目的：规范软件开发流程，推出高质量的软件产品。
方法：面向对象模式、结构化开发模式、基于过程模式、某种定制的模式等。
作用：降低开发成本，达到要求的软件功能、提高软件性能、提高软件的可移植性、降低维护费用、按时完成工作并交付使用。

说明错误、缺陷、失败的含义与联系。

含义：
错误：软件开发过程中人为产生的错误。（比如，代码写错了）
缺陷（故障）：在实现软件功能的时候存在的问题。（比如，代码写错了导致系统无法启动，是静态的）
失败：在运行时，软件违背了它应该有的行为。（比如，使用者发现某个计算功能算不出结果，是动态的）

联系：
人为原因导致错误，错误导致软件缺陷，用户在使用这样的软件时，因为软件缺陷而导致软件失败。
缺陷是开发者角度的名词，是静态的。
失败是用户角度的名词，是动态的。

软件质量应从哪几个方面来衡量？论述之。

一个好的软件要从三方面衡量：
1、产品质量。
2、过程质量。
3、商业质量。

1、产品质量：
衡量标准有两方面：用户和开发者。
用户：足够的功能，上手简单，易用。
开发者：从软件的内部特征来衡量，比如缺陷的数量。

2、过程质量（软件生产过程的质量）：
很多事件都会影响过程的质量，并且最终影响到软件的质量。
量化方法：CMM等。

3、商业质量：
技术价值：各种技术指标。（运行时间，维护费用等）
商业价值：机构对于软件是否与其战略利益相吻合的一种战略评估。
目标：将技术价值和商业价值进行统一。

现代软件工程大致包含的几个阶段及各个阶段文档。

1、需求分析
完成需求规格说明书（SRS）。

2、系统设计
完成系统结构图（SAD）。

3、程序设计
完成模块功能与数据描述文档。

4、软件开发
完成开发记录文档。

5、单元测试
按照程序设计中的要求进行测试，完成单元测试文档。

6、集成测试
按照SAD测试，完成集成测试文档。

7、系统测试
按照SRS进行测试，完成系统测试文档。

8、系统提交
提交系统说明文档。

9、维护
提交维护记录文档。

什么是软件过程?

软件开发活动中各种组织规范方法。

什么是重用、抽象等现代软件工程主要概念?

1、重用※
重复采用以前开发的软件系统中具有的共性部件，用到新的开发项目中去。

2、抽象※
基于某种层次归纳水平的问题描述。它使我们将注意力集中在问题的关键方面而非细节。

3、分析、设计方法和符号描述系统
使用标准来对程序进行描述，利于交流和建模并检查其完整性和一致性，利于重用。

4、用户界面原型化
建立系统的小型版, 通常具有有限的关键功能,以利于用户评价和选择，证明设计或方法的可行性。

5、测度和度量
通用的评价方法和体系，有助于使过程和产品的特定特性更加可见，包括量化描述系统、量化审核系统。

6、工具和集成环境
通过框架比较软件工程环境提供的服务，以决定其好坏。

Chapter02

什么是过程（process）？软件过程的重要性是什么？软件生命周期?

软件开发过程中产生某种期望结果的一系列有序任务，涉及活动、约束和资源。

重要性：
1、通用性：软件过程可以让一系列开发活动保持一致性和结构性，因而具有了通用性。
2、指导性：软件过程使我们可以分析、检查、理解、控制和改善软件开发活动。
3、可以把获得的经验传递给其他人。

软件生命周期：
1、需求分析
2、系统设计

3、程序设计
4、软件开发
5、单元测试
6、集成测试
7、系统测试
8、系统提交
9、维护

瀑布模型及各阶段文档，优缺点?

需求分析——SRS
系统设计——SAD
程序设计——算法和数据描述文档
编码——源程序及注释
单元测试和集成测试——单元测试报告
系统测试——系统测试报告
验收测试——验收测试报告
运行与维护——维护报告

优点：
1、简单性：很容易向用户解释。
2、基础性：是其他更复杂模型的基础（通过加入额外的开发活动和循环）。
3、每个过程都有相应的提交产物，方便进度评估。

缺点：
1、面对需求变动时，该模型无法实际处理重复开发问题。
2、文档转换时有困难。

原型的概念与用途。

概念：一种部分开发的产品，用来让用户和开发者共同研究，提出意见，为最终的产品定型。

特点：
1、原型化设计可以使开发者更容易地提高软件质量。
2、原型化设计可以提供多种解决方案供用户选择。

论述分阶段开发模型的含义, 其基本分类及特点是什么?

系统被设计成部分提交，每次用户只能得到部分功能，而其他部分处于开发过程中。

基本分类及特点：
1、增量式开发：
开始建造的版本是规模小的部分功能的系统，后续版本添加包含新功能的子系统，最后版本是包含所有功能的子系统集合。

2、迭代式开发：
系统开始就提供了整体框架，但是各部分功能都不够完善，后续版本会完善各部分的功能。

3、增量式+迭代式结合开发：
一个新发布的版本可能包含新功能，并对已有功能做了改进。

螺旋模型四个象限的任务及四重循环的含义?

螺旋模型每次迭代有四个任务，依次是（四个象限）：
计划、目标/可选方案、风险评估、 开发与测试。

螺旋模型共有四次迭代，依次是（每个象限的四重循环）：
操作概念、软件需求、软件设计、开发与测试。

什么是UP， RUP，进化式迭代等市场流行的过程模型?

UP模型即统一过程模型，是一种用例驱动的，以基础架构为中心的，迭代式，增量式的软件开发模型。

该模型的四个阶段：
开始阶段、确立阶段、构建阶段和移交阶段。

该模型的六道核心工序：
业务模型工序、需求工序、分析设计工序、实现工序、测试工序和部署工序。

RUP模型是IBM提出的提供支持和包装的UP模型。

迭代开发是统一过程模型（RUP）的关键实践。
开发被组织成一系列固定的短期小项目。
每次迭代都产生经过测试、集成并可执行的局部系统。
每次迭代都具有各自的需求分析、设计、实现和测试。
随着时间和一次次迭代，系统增量式完善。

Chapter03

什么是项目进度？活动？里程碑？

项目进度是对特定项目的软件开发周期的刻画。包括对项目阶段、步骤、活动的分解，对各个离散活动的交互关系的描述，以及对各个活动完成时间及时间的初步估算。

活动是项目的一部分，一般占用项目进度计划的一段时间。

里程碑是特定的时间节点，标志着活动的结束，通常伴随着提交产物。

如何计算软件项目活动图的关键路径？冗余时间？最早和最迟开始时间？

关键路径：从起点到终点花费最长时间的路径。
冗余时间：在不耽误总体进度的前提下，最早开始工作和最晚开始工作时间的差值。

软件项目团队组织的基本结构？

1、主程序员负责制
由一个主程序员主导系统的设计和开发。
优势：交流最小化，迅速做出决定，效率高。
劣势：主观性强，对主程序员要求高。

2、忘我方法
每个成员平等的承担责任，批评和表扬只针对产品，不针对个人。

一般软件项目的团队组织结构是介于以上两个极端之间的。

专家估算法的大致含义？算式估算法的大致含义？

专家估算法：
使用专家的知识和经验，对软件项目的工作量进行评估，是一种经验性的猜测。
一般使用三种方法：
1、类推法：
做出三种估计，乐观的、悲观的、最可能的估计，加权平均。
2、Delphi法：
几组专家预测，取平均值。
3、Wolverton法：
通过计算老问题（O），新问题（N）和容易的（E），适中的（M），困难的（H）这2x3的组合，形成矩阵进行估计。

算式估算法：
$E = (a + bSc) m (X)$
E为工作量，a，b，c都为常数，S是估算的系统规模，X 是一个从X1到Xn的n维度成本因素向量，m是基于该因素的调整因子。

试述COCOMO模型的三个阶段基本工作原理或含义。

COCOMO 模型的关键在于针对项目开发的不同阶段来设置工作量的衡量标准，逐步细化，逐渐准确。: E = bSc m (X)

在阶段一，项目通常构建原型以解决包含用户界面、软件和系统交互、性能和技术成熟性等方面在内的高风险问题。

在阶段二，即早期设计阶段，已经决定将项目开发向前推进，但是设计人员必须研究几种可选的体系结构和操作的概念。

在阶段三，即后体系结构阶段，开发已经开始，而且已经知道了更多的信息。在这个阶段，可以根据功能点或代码行来进行规模估算，而且可以较为轻成本因素。

什么是软件风险？了解主要风险管理活动？有几种降低风险的策略?

软件开发过程中不希望看到的，有负面结果的事件。

风险管理活动：
风险评价：风险识别，风险分析，风险优先级分配
风险控制：风险降低，风险管理计划，风险化解

降低风险的策略：
1、避免风险：改变功能和性能需求，使风险没机会发生。
2、转移风险：通过把风险分配到其他系统中，或者购买保险以便在风险成为事实时弥补经济上的损失。
3、假设风险：用项目资源，接受并控制风险。比如在开发时主动有意识地进行测试。

Chapter04

需求的含义是什么?

需求是对来自用户的关于软件系统的期望行为的综合描述，涉及系统的对象、状态、 约束、功能等。

需求作为一个工程，其确定需求的过程是什么?

1、原始需求获取：客户给出的需求。
2、问题分析：理解需求并通过建模或模型化方式进行描述。
3、规格说明草稿：利用符号描述系统将定义规范化表示。
4、需求核准：开发人员与客户进行核准。
5、形成软件规格说明书，即SRS。

举例说明获取需求时，若有冲突发生时，如何考虑根据优先级进行需求分类。

对需求进行优先级划分：
1、必须要被满足的需求。
2、非常值得做但是不是必须的需求。
3、可选的需求。（可做可不做的需求）

需求文档分为哪两类?

1、需求定义
完整地罗列了客户期望的需求。

2、需求规格说明（SRS）
将需求重述为“要构建的系统将如何运转”的规格说明。

什么是功能性需求和非功能性需求？ 设计约束？过程约束？如何区分?

1、功能需求：描述系统内部功能或系统与外部功能的交互作用，涉及系统输入应对、 实体状态变化、输出结果、设计约束、过程约束等。
2、设计约束：已经做出的设计决策或限制问题解决方案集的设计决策。涵盖物理环境、接口、用户等方面。
3、过程约束：对用于构建系统的技术和资源的限制，涵盖资源、文档等方面。
4、非功能需求：描述软件方案必须具备的某些质量特征，例如系统性能、安全性、响应时间等。

Chapter05

什么是软件体系结构？设计模式？设计公约？设计？

软件体系结构：
一种软件的解决方案，用于将系统分解为单元子系统，以及这些单元如何相互关联，还包括这些单元所有的外部特性。

设计：
将需求中的问题转变成软件解决方案的创造性过程。

设计模式：
一种针对软件模块给出的一般性解决方案，提供较低层次的设计决策。

设计公约：
一系列设计和决策的集合，用于提高系统某方面的设计质量。

软件设计过程模型的几个阶段？

1、建模：尝试可能的分解，根据需求描述的系统的关键特性等确定软件体系结构。
2、分析：分析初步的体系结构，主要关注系统级别的决策，如软件的质量、性能等。
3、文档化：确定各个不同的模型视图。
4、复审：检查文档是否满足了所有需求。
5、最终产出：软件体系结构文档，即SAD。用来和开发团队中其他人员交流系统级别设计决策的有力工具。

论述设计用户界面应考虑的问题。

设计界面要注意解决的要素:
1、隐喻：可识别和学习的基本术语、图像和概念等。
2、思维模型：数据、功能、任务的组织与表示。
3、模型的导航规则：怎样在数据、功能、活动和角色中移动及切换。
4、外观：系统向用户传输信息的外观特征。
5、感觉：向用户提供有吸引力的体验的交互技术。

文化问题：
设计界面时需要考虑使用系统的用户的信仰、价值观、道德规范和传统等因素。
1、使用国际设计/无偏见设计，排除特定的文化参考或偏见。
2、采用定制界面，使不同用户看到不同的界面 。

用户偏好：
为具有不同偏好的人选择备选界面。

模块独立性、耦合与内聚的概念及各个层次划分？

模块的独立性取决于两个部分：内聚和耦合，我们追求的是高内聚低耦合。
内聚是软件内部组成成分的关联程度。
耦合指的是两个软件间的关联程度。

举例说明耦合与内聚的基本分类。以及各个分类的含义与特征

内聚：

1、偶然内聚：模块各部分不相关，只为方便或偶然性原因放入同一模块。比如强行放入一个类中没有任何关系的方法。

2、逻辑内聚：模块中各部分只通过代码的逻辑结构相关联，会共享程序状态和代码结构，但相对于数据、功能和目标的内聚比较弱。 比如因为有相同骤而放在?一起的两个没有关系的计算。

3、时间内聚：部件各部分要求在同一时间完成或被同一任务使用而形成联系。比如初始化模块中需要完成变量赋值、打开某文件等工作。

4、过程内聚：要求必须按照某个确定的顺序执行一系列功能，模块内功能组合在一起只是为了确保这个顺序。其与时间性内聚相比优点在于其功能总
动和针对相关目标，如写数据->检查数据->操作数据这一过程。

5、通讯内聚：各部分访问和操作同一数据集，如来自于同一传感器的所有不相干数据。

6、顺序内聚 ：各部分有输入输出关系，操作同一数据集，并且操作有顺序。

7、功能内聚：理想情况，各部分组成单一功能，且每个处理元素对功能都是必须的，每个元素执行且只执行设计功能，如一个简单的输出程序。

8、信息内聚：在功能内聚的基础上，进行数据抽象化和基于对象的设计。

耦合：

1、内容耦合：A模块实际上修改了B模块，B模块完全依赖于A模块。

2、公共耦合：不同模块可以从公共数据存储区来访问和修改数据。

3、控制耦合：一个模块通过传递参数或返回代码来控制另一个模块的活动。

4、标记/特征耦合：使用一个复杂的数据结构进行模块间传递消息，并且传递的是该数据结构本身。比如将一个数组传递给另一个模块，数组仅用于计
制。

5、数据耦合：模块间传递的是数据值，是最受欢迎的耦合。

6、非耦合：模块相互之间没有信息传递，但是不太现实。

软件过程中复审的概念，设计复审的重要性。

检查文档是否满足所有功能和质量需求。

重要性：
1、可以和用户一起检查软件的概要设计。
2、可以向开发者呈现并明确软件的技术设计。
3、程序员通过复审可以在下一阶段的工程实施前得到本阶段工作的反馈。

Chapter06

什么是设计模式?

设计模式：
一种针对软件模块给出的一般性解决方案，提供较低层次的设计决策。

面向对象设计模式：
简单工厂模式。
工厂方法模式。
抽象工程模式。
观察者模式。
单例模式。
桥梁模式。
责任链模式。
适配器模式。
代理模式。
策略模式。

OO设计的基本原则?

1、单一职责原则
2、重用原则
3、开闭原则
4、里氏替换原则
5、依赖倒转原则
6、接口隔离原则

7、迪米特法则

OO开发有何优势?

1、语言的一致性：
采用相同的语义结构（类、对象、接口、属性、行为）描述问题和解决方案。

2、软件开发过程的一致性：
从需求分析和定义、高层设计、底层设计到编码和测试等，所有的过程都采用相同的语义结构。

OO开发过程有几个步骤?

1、面向对象需求分析。
2、面向对象高层设计。
3、面向对象底层设计。
4、面向对象编程。
5、面向对象测试。

Chapter07

一般性的编程原则应该从哪三个方面考虑?

编程原则应该从控制结构，算法和数据结构三方面来考虑。

控制结构：程序如何传递数据。
算法：程序如何处理数据。
数据结构：程序如何储存数据。

在编写程序内部文档时，除了HCB外，还应添加什么注释信息？注意什么?

需要添加其他程序注释——
（1）解释性注释：本段源代码是在做什么的注释。
（2）分解性注释：通过注释将代码分解成多个段。
（3）版本注释：随着时间进行修改的记录。

注意的问题：
1、分段注释
2、注释和代码要一并更改。
3、注释要有意义。
4、一边写代码一边写注释，不要写完代码回过头来添加注释。

什么是极限编程(XP)? 以及派对编程?

极限编程：
极限编程（XP）是一种轻量级的软件开发方法，属于敏捷开发方法。它将复杂的开发过程分解为一个个相对比较简单的小周期，通过交流、反馈等方法
和客户可以非常清楚开发进度、变化、待解决的问题和潜在的困难等，并根据实际情况及时地调整开发过程。

派对编程：
一种敏捷开发方法，其开发方式是两个程序员共同开发程序，且角色分工明确。一个负责编写程序，另一个负责复审与测试。两人定期交换角色。

Chapter08

有几种主要的缺陷类型?

1、算法缺陷
算法的某些处理步骤或逻辑有问题，以至于软件部件对给定的输入数据无法产生正确的输出。

2、计算和精度缺陷
算法或公式在实现时出现错误，或计算结果的精度达不到要求。

3、文档缺陷
文档和程序不一致。

4、过载缺陷
程序运行时，对数据结构的使用超过了其承载能力。

5、能力缺陷
程序活动到达极限时，其性能会变得不可接受。

6、时序性缺陷
多个同时执行或者一个仔细定义的顺序执行的进程之间协调不当。

7、性能缺陷
系统性能达不到要求。

8、恢复性缺陷
系统运行失败后无法恢复。

9、硬件和系统软件缺陷
提供的硬件或者系统软件并没有按照文档中的操作条件或步骤运作。

10、代码的标准和规程缺陷。
代码没有遵循组织机构的标准和过程。

什么是正交缺陷分类?

被分类的任何一项故障都只属于一个类别，则分类方案是正交的。如果故障属于不止一个类，则失去了度量的意义。

测试的各个阶段及其任务?

1、单元测试：将每个程序构件与系统中的其他构件隔离，对其本身进行测试。

2、集成测试：验证系统构件是否能够按照系统和程序设计规格说明中描述的那样共同工作的过程。

3、功能测试：对系统进行评估，以确定集成的系统是否确实执行了需求规格说明中描述的功能，其结果是一个可运转的系统。

4、性能测试：测试系统的软硬件性能是否符合需求规格说明文档。 其结果是一个确认的系统。

5、验收测试：确定系统是按照用户的期望运转的。

6、安装测试：确保系统在实际环境中按照应有的方式运转。

7、系统测试：功能测试、性能测试、验收测试和安装测试统称为系统测试。

黑盒、白盒的概念?

1、黑盒测试：将测试的对象看作是一个密闭的黑盒，我们的测试就是向闭盒提供输入的数据，并记录产生的输出。测试的目标是确保针对每种输入，
与预期的输出相匹配。黑盒测试参考的文档是系统设计和程序设计阶段的文档。
2、白盒测试：将测试对象看作一个白盒，然后根据测试对象的结构用不同的方式进行测试。

什么是单元测试？ 什么是走查和检查?

将每个程序构件与系统中的其他构件隔离，对其本身进行测试。

走查：不正式的的代码评审。
检查：正式的代码评审，事先准备问题清单，依据清单比对代码和文档的一致性。

黑盒方法各自的分类？测试用例的设计和给出方法（结合补充材料）

黑盒测试方法：
1、等价分类法：将输入域划分为若干等价类。每个测试用例都代表了一类与它等价的
其他例子。
2、边界值分析法：把测试值选在等价类的边界上进行测试。
3、错误猜测法：猜测程序中哪些地方容易出错，并据此设计测试用例。
4、因果图法：适用于被测试程序有很多输入条件，程序的输出又依赖输入条件的各种组 合的情况。

白盒方法的分类，各种覆盖方法等。（课件和补充课件）

白盒测试方法:
1、语句覆盖：程序的每条语句都要执行一次。
2、判定(分支)覆盖：程序的每个分支都要执行一次。
3、条件覆盖：要求判定中的每个条件均按照“真”、“假”两种结果至少执行一次。
4、条件组合覆盖：要求所有条件结果的组合都至少出现一次。

集成测试及其主要方法的分类？（驱动，桩的概念）

驱动程序：调用特定构件并向其传递测试用例的程序，即代替上层模块的调用程序。

桩：用于模拟测试时缺少构件时的活动的程序。可以使测试能够正常的进行下去，即代替下层模块的应答程序。

1、自底向上集成
先测试系统最底层的模块，接着测试调用这些底层模块的模块，直到测试完毕。

2、自顶向下集成
先测试系统最上层的模块，接着测试顶层模块调用的下层模块，直到测试完毕。

3、一次性集成
先测试每一个模块，之后将所有模块一并集成。

4、三明治集成
将系统分成三层，目标层处于中间、目标层上有一层，目标层下有一层。在顶层采用自顶向下的方式集成，在较低层采用自底向上的方式集成，测试集

Chapter09

系统测试的主要步骤及各自含义？（图9.2）

1、功能测试：根据SRS测试系统功能。
2、性能测试：根据SRS测试系统性能。
3、验收测试：根据客户的需求定义，由客户和用户一起测试。
4、安装测试：在用户环境下进行测试。

什么是回归测试?

回归测试是用于新的版本或者改进版本的一种测试，以验证与旧版本相比，软件是否仍然以同样的方式执行同样的功能。

功能测试的含义极其作用?

含义：
测试需求设计（SRS）中提出的功能性需求。

作用：
有很高的故障检测率。

功能测试的基本指导原则?

指导原则：
1、较高的查错概率。

2、 独立的测试团队。
3、了解预期的输出结果。
4、对合法与非法的输入都要进行测试。
5、不能修改系统。
6、达成某些条件后才能停止测试。

性能测试的含义与作用?

含义:
测试非功能性需求。

作用:
确保系统的可靠性、可用性和可维护性。

性能测试的主要分类?

分类:
1、压力测试:短时间内加载极限负荷,验证系统能力
2、容量测试:验证系统处理巨量数据的能力。
3、配置测试:测试各种软硬件配置(最小到最大)。
4、兼容性测试:测试接口等。
5、回归测试:如果这个系统要替代现有系统时需要进行此测试。
6、安全性测试
7、计时测试
8、环境测试
9、质量测试
10、恢复测试
11、维护测试
12、人为因素测试

确认测试概念,确认测试分类? (基准测试和引导测试)

由用户检查软件系统是否满足了他们的需求的测试。

种类:
1、基准测试:
由用户准备典型测试用例,在实际安装后的系统运作并由用户对系统执行情况进行评估。
2、引导测试:
在假设系统已经永久安装的前提下执行系统。它依赖系统的日常工作进行测试,相对基准测试不是非常的正式与结构化。

什么是alpha测试? β测试?

α测试:客户进行实际的测试前,先自己组织团队(或者委托其他团队)测试这个系统。
β测试:客户实际进行测试。

什么是安装测试?

在用户的系统环境中安装软件并进行测试。

shadowingszy
发布了31 篇原创文章 · 获赞 125 · 访问量 3万+