**2018级高软上课提问：**

1. **SE和CS有什么区别？**

一个重实践一个偏理论。（话是这么说，实践也是有可能被人家吊锤的hhhhhh）

1. **怎么认识质量？**

* 先验论的观点：质量是可以认知但不可定义的。
* 用户的观点：质量是恰好达到目的。
* 制造业的观点：质量是与规格说明的一致。
* 产品的观点：质量是与产品的内在特征相联系的。
* 基于价值的观点：质量取决于客户愿意支付的金额。

1. **影响软件工程7大关键因素？**

* 交付时间的重要性
* 计算行业经济的变化趋势
* 功能强大的桌面运算
* 互联网络
* 面向对象的技术
* 图形用户界面
* 瀑布模型的不可预知性

1. **软件生命周期是？**

立项，需求分析，设计，编码，测试，交付，维护，退役

1. **软件生命周期各阶段产品？**

可行性研究：可行性报告

需求分析：SRS和用户需求定义文档

概要设计：概要设计规格说明书、数据库或数据结构设计说明书、集成测试计划

详细设计：详细设计规格说明书, 单元测试计划

实现：源程序代码

单元测试：单元测试报告

集成测试：满足概要设计要求的程序、组装测试报告

确认测试：可供用户使用的软件产品(文档，源程序)

1. **单元测试，集成测试，确认测试各自的任务？**

单元测试：对模块进行测试。

集成测试：组装测试应满足概要设计的要求。

确认测试：根据软件需求规格说明书，测试软件系统是否满足用户的需求。

1. **瀑布模型的优缺点？**

优点：软件生命周期模型,使软件开发过程可以在分析、设计、编码、测试和维护的框架下进行。软件开发过程具有系统性、可控性，克服了软件开发的随意性。

缺点：项目开始阶段用户很难精确的提出产品需求，由于技术进步，用户对系统深入的理解，修改需求十分普遍。项目开发晚期才能得到程序的运行版本，这时修改软件需求和开发中的错误代价很大。采用线性模型组织项目开发经常发生开发小组人员“堵塞状态” ，特别是项目的开始和结束。

1. **RUP模型的四个阶段？**

先启、精化、构建、产品化。

1. **RUP模型和瀑布模型的区别？**

RUP是一个弹性模型，如果先启阶段所有用例都被详细描述那就是瀑布模型。RUP模型是一个二维模型，在每个阶段都会涉及到软件生命周期的各个阶段。瀑布模型是一维的线性的。

1. **先启阶段主要工作？**

* 确定项目开发的目标和范围
* 定义主要的需求：用例以及主要的用例场景
* 根据一些主要的用例场景来构建一个基本架构
* 估算开发周期和成本估计潜在的风险
* 主要实践活动－用例建模

1. **构建阶段主要工作？**

* 尽快完成软件产品的开发
* 迭代实现遗留下来的风险较低和比较容易的元素，准备部署
* 在保证开发进度的同时达到足够的软件质量
* 获得一些有用的版本 (alpha, beta等)

1. **拿到项目先做什么？**

项目分解

1. **WBS是啥？**

WBS主要是将一个项目分解成易于管理的几个部分或几个细目，以便确保找出完成项目工作范围所需的所有工作要素，是一种在项目全范围内分解和定义各层次工作包的方法。

1. **简述WBS叶子结点**

可执行，可落地的工作任务，并且有责任人，工作时长，和产品。

1. **叶子结点多长时间合适？**

对于咱们来说四个小时。

1. **WBS两种分解方法（没提）**

基于可交付成果的划分和基于工作过程的划分。

1. **如何选择人员？**

* ability to perform work开展工作的能力
* interest in work工作兴趣
* experience with经验
* Training培训
* ability to communicate with others与其他人交流的能力
* ability to share responsibility共同承担责任的能力
* management skills管理技能

1. **LOC,FP的优缺点？**

LOC优点：

* 用软件代码行数估算软件规模简单易行。

LOC缺点：

* 代码行数的估算依赖于程序设计语言的功能和表达能力；
* 采用代码行估算方法会对设计精巧的软件项目产生不利的影响；
* 在软件项目开发前或开发初期估算它的代码行数十分困难；
* 代码行估算只适用于过程式程序设计语言（C,Pascal），对非过程式的程序设计语言不太适用等等。

FP优点：

* 与程序设计语言无关，它不仅适用于过程式语言，也适用于非过程式的语言；
* 软件项目开发初期就能基本上确定系统的输入、输出等参数，功能点度量能用于软件项目的开发初期。

FP缺点：

* 它涉及到的主观因素比较多， 如各种权函数的取值；
* 信息领域中的某些数据有时不容易采集；
* FP的值没有直观的物理意义。

1. **功能点分类？**

数据类型和人机交互类型

1. **EO和EQ的区别？**

EO是经过输出逻辑处理的数据或信息，EO可能包括对ILF的维护或系统行为的改变。EQ是直接提取，EQ不维护ILF，也不会引起系统行为的改变。

1. **DET和RET的概念**

数据元素类型(Data Element Types DET)是一个用户可识别的、唯一性的、非递归的域。

记录元素类型(Record Element Types RET)是ILF或者EIF中用户能够识别的数据元素小组（集合）。

1. **功能点计算的大致过程**
   1. 识别功能点类型
   2. 识别应用程序的边界范围
   3. 计算数据类型功能点提供的未调整的功能点数量（查表）
   4. 计算人机交互功能所提供的未调整的功能点数量（查表）
   5. 确定调整因子（根据系统特性表进行打分）
   6. 计算调整后的功能点。
2. **什么是风险？**

在给定情况下和特定时间内，那些可能发生的结果与预期结果之间的差异，差异越大，风险越大。

1. **什么是风险成本？**

风险成本＝风险概率×风险后果

1. **什么是风险后果？**

与该事件有关的损失

1. **风险管理四个方面？具体是啥？**

风险标识：列出可能存在的风险

风险预测：建立一个尺度，评价每种风险发生的可能性，以及导致的后果。

风险评估：进一步审查风险，定义一个风险参考水准，发现风险太高，就终止项目

风险管理与监控：辅助项目建立处理风险的策略（风险避免，风险监控，风险管理及监控计划）

1. **制定进度计划常见技术是？**

关键路径法，计划评审技术，图形评审技术

1. **QC具体是什么？包含什么？**

为保证每一件工作产品都满足需求而应用于整个开发周期中的一系列审查、评审和测试。

包含：反馈循环和可度量规约

1. **QA具体是什么？**

QA是管理层的眼睛，用以监视过程是否按计划进行以及产品质量是否符合标准并向管理层汇报

1. **什么是软件配置？**

软件配置是由在软件工程过程中产生的所有信息项构成的，它可以看作该软件的具体形态（软件配置项） 在某一时刻的瞬间影像。

1. **什么是软件配置管理？**

协调软件开发使得混乱减到最小的技术叫做软件配置管理，它是一种标识、组织和控制修改的技术，目的是使错误达到最小并最有效地提高生产效率。

1. **什么是配置审计？**

通过调查研究确定已制定的过程、指令、规格说明、基线及其他特殊要求是否恰当和被遵守、以及实现是否有效。通过正式技术评审或软件配置审计来保证变更的有效性。

1. **敏捷出发点？**

以人为本、尽快交付可使用的产品

1. **敏捷四条宣言？(最后还有一句话)**

* 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

虽然右项也有价值，但我们认为左项具有更大的价值

1. **xp和scrum异同点？**

相同点：都是通过迭代进行开发

不同点：XP计划游戏周期为两周，scrum叫做sprint，周期为一个月，还有个头头。

1. **需求工程出发点？**

捕获用户真实需求，并无歧义的描述

1. **需求分析的四个阶段？**

* 导出需求
* 分析建模
* 规格说明
* 需求确认和校验

1. **什么是校验与确认？**

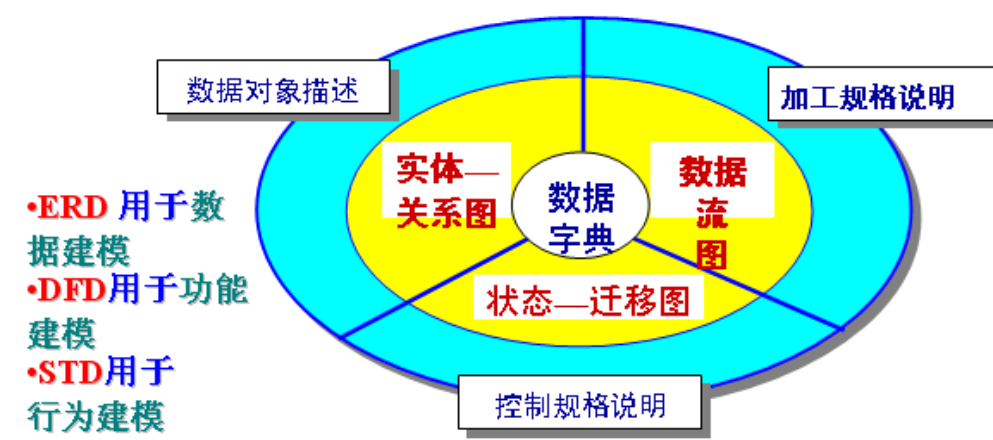
需求确认的目的是用来检查获得的需求定义是否准确的反应了用户的实际需求。

需求校验的目的是用来检查需求规格说明文档和需求定义文档是否一致。

1. **为什么需求分析建模？**
   1. 可以从不同的角度， 不同的抽象级别精确地说明对问题的理解，对目标软件的需求。
   2. 模型可以帮助用户和分析人员排除用户不一致，不合理的部分，挖掘潜在的用户要求。
   3. 模型是分析人员根据初步导出的需求而建立的软件系统结构。
   4. 模型是形成SRS和进行软件设计的基础。
2. **过程论怎么看待程序世界的？**
3. 数据和逻辑是分离的、独立的，各自形成程序世界的一个方面。所谓世界的演变，是在逻辑作用下，数据做改变的一个过程。
4. 过程有明确的开始、结束、输入、输出，每个步骤有着严格的因果关系。
5. 过程是相对稳定的、明确的和预定义的，小过程组合成大过程，大过程还可以组合成更大的过程。

结论：程序世界本质是过程，数据作为过程处理对象，逻辑作为过程的形式定义，世界就是各个过程不断进行的总体。

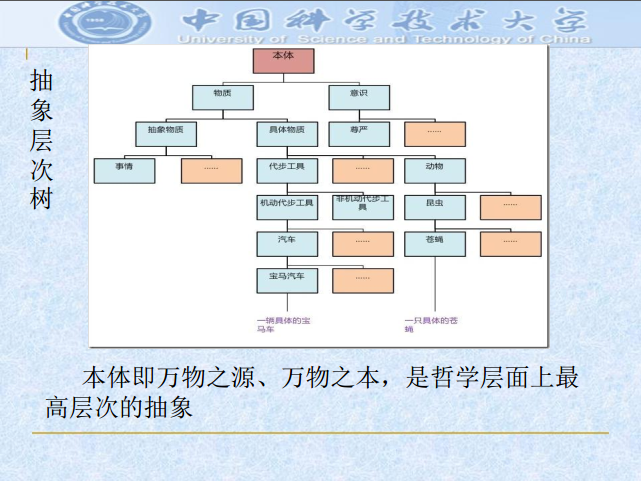
1. **结构化分析模型包含什么？**

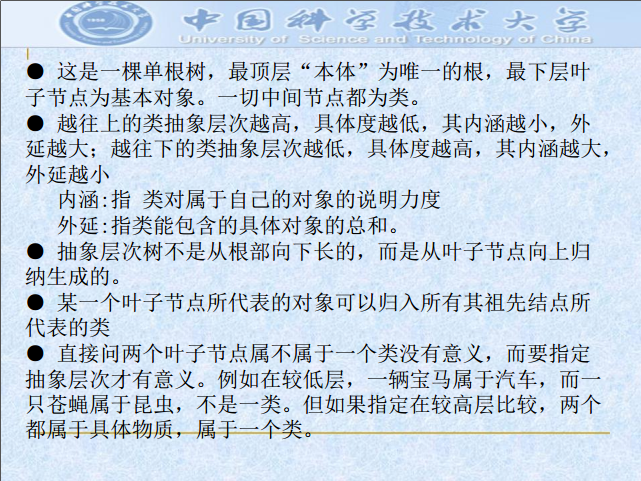


1. **对象论观点是什么？**
2. 数据和逻辑不是分离的，而是相互依存的。
3. 相关的数据和逻辑形成个体，这些个体叫做对象，世界就是由一个个对象组成的。
4. 对象具有相对独立性， 对外提供一定的服务。
5. 所谓世界的演进，是在某个“初始作用力”作用下，对象间相互调用而完成的交互，在没有初始作用力下，对象保持静止。
6. 交互并不是完全预定义的，不一定有严格的因果关系对象间交互是偶然的，对象间联系是暂时的。

结论：世界就是由各色对象组成，然后在初始作用力下，对象间的交互完成了世界的演进。

1. **对象层次树？**





1. **什么是抽象？**

从具体事物抽出、概括出它们共同的方面、本质属性与关系等，而将个别的、非本质的方面、属性与关系舍弃，这种思维过程，称为抽象。

1. **先有对象还是先有类？**

从哲学角度说，先有对象，然后才有类，类和对象是“一般和特殊”这一哲学原理在程序世界中的具体体现。

1. **为什么会有耦合**？

世界上各种对象形成了一张复杂的耦合网，正因为有耦合的存在，世界才能演进。如果没有耦合，世界就变成了“死世界”，无法演进和发展。所以，世界需要耦合。正如马克思主义哲学所说：联系是普遍的、客观的。所以，耦合的存在，有其深刻的哲学意义。

1. **什么是多态？（结合层次树来说）**

多态性（polymorphism）是指同一个操作作用于不同的对象上可以有不同的解释，并产生不同的执行结果。（结合层次树不会hhhhhh）

1. **OO分析时，分析模型包含什么？**

* 需求描述
* 静态模型（对象模型）
* 动态模型（交互次序）
* 功能模型（数据交换）

1. **静态模型包括？**
2. 找出类-对象并进行筛选；
3. 找出对象之间的关联；
4. 划分主题
5. 找出对象的属性；
6. 用继承组织和简化对象类；
7. 迭代并反复提炼模型
8. **动态模型包括**？
9. 编写脚本；
10. 设计用户界面；
11. 画事件跟踪图
12. 画状态图；
13. 审查动态模型。
14. **UML包含哪九张实际的图？**

类图、对象图、用例图、顺序图、协作图、状态图、活动图、构件图、配置图（实施图）。

1. **程序员关注哪些图？**

程序员关注的是系统的配置、装配等问题，因此会侧重于实现视图。

1. **用例图有几种关系？具体是？**

包含关系：用例可以简单地包含其他用例具有的行为，并把它所包含的用例行为做为自身行为的一部分，这被称作包含关系。

扩展关系：扩展关系是从扩展用例到基本用例的关系，它说明为扩展用例定义的行为如何插入到为基本用例定义的行为中。

泛化关系：用例可以被特别列举为一个或多个子用例，这被称做用例泛化。

1. **类与类之间的关系是什么？强弱关系是？**

关联关系、依赖关系、聚合关系、组合关系、泛化（继承）

依赖<关联<聚合<组合

1. **概念设计和技术设计指的是什么？**

概念设计：告诉顾客系统将要做什么

技术设计：告诉程序员系统将要做什么

1. **内聚有哪几种？高内聚需要做到？**

功能性内聚、顺序内聚、通讯内聚、过程内聚、时间内聚、逻辑内聚、偶然（巧合）内聚。

1. **耦合有哪几种？低耦合需要做到？**

内容耦合、公共耦合、控制耦合、标记耦合、数据耦合、非直接耦合

1. **解释一下标记耦合和数据耦合？**

标记耦合：参数传递的是诸如结构类型的数据

数据耦合：参数传递的是一般类型的数据

1. **怎样降低耦合？**

降低模块间接口的复杂度、改变调用模块的方式

1. **作用域和控制域指的是？**

控制域：控制范围，是包括模块本身以及所有下属模块（直接调用模块和间接调用模块）的集合。

作用域：作用范围，它是一个与条件判定相联系的概念。是受该模块内一个判定影响的所有模块的集合。

1. **怎么解决作用域不在控制域之内？**

判定上移和在作用域但不在控制域的模块下移

1. **扇入扇出适中，系统呈椭圆形**
2. **OO设计主要工作是？**

用例实现精化、体系结构设计、构件设计、用户界面设计、数据持久设计、迭代精化

1. **用例精化怎么做？**

(1) 提取边界类、 实体类和控制类；

(2) 构造交互图；

(3) 根据交互图精化类图

1. **类设计原则？**

* SRP 单一职责原则
* OCP 开放封闭原则
* LSP 里氏替换原则
* DIP 依赖倒置原则
* ISP 接口隔离原则

1. **包设计原则？哪些是内聚？哪些是耦合？**

包的内聚性：

* REP 重用发布等价原则
* CCP 共同封闭原则
* ·CRP 共同重用原则

包的耦合性：

* ADP 无环依赖原则
* SDP 稳定依赖原则
* SAP 稳定抽象原则

1. **详细解释DIP？**

高层模块不应该依赖于低层模块，两者都应该依赖于抽象。

抽象不应该依赖于细节，细节应该依赖于抽象。

程序中所有的依赖关系都应该终止于抽象类或者接口。

抢答：

1. 质量的观点？关注哪几种？

2. 概要设计要做哪些设计？

3. 详细设计要做哪些设计？

4. 需求工程主要做什么？关键词是？真实的需求

期中考试考的一道题：

**Wasserman规范**：

* Abstraction抽象
* Analysis and design methods and notations分析、设计方法和符号
* User interface prototyping用户界面原形
* Software architecture软件体系结构
* Software process软件过程
* Reuse复用
* Measurement度量
* Tools and integrated environments工具和集成环境