**软工知识点速查.md**

**软件工程方法学三要素： 方法，工具，要素**

**14-15卷子**

**一、**

**1.软件系统虽然复杂性高，但却相对稳定不易发生改变 F**

**2.“响应时间”为软件系统的“功能性需求” F P94**

**3.在UML状态图中，必须包含初始状态，但可以不包含终止状态 T**

**4.联络人除了负责本项目组与其他项目组沟通，还可在本项目组中担任其他角色 T P65**

**5.在UML用例图中，为了描述特定用例中与异常处理相关的功能，通常采用包含关系进行处理 F（扩展关系）**

**6.UML是一种可扩展语言，而模板是扩展UML语言表达能力的一种常用机制 T P55**

**7.在系统设计阶段，一个好的子系统分解应具有低耦合性 T P173**

**8.在构件图中，通常采用符号---o表示构件的需求接口 F P173**

**9.在对象设计阶段，代理/授权机制可用于代替实现/继承机制，从而提高系统的效率 T**

**10.在UML图中，公共属性或操作采用符号#进行标注 F（受保护的）**

**二、**

**C D C D C C B C B C**

**1.UML主要起源于James Rambaugh，Ivar Jacobson，Grady Booch三个人的早期工作**

**2.对于UML状态图，关于动作的描述：**

**动作通常是与“事件”相关联的、执行时间很短，执行过程不可打断 √**

**采用do标记进行标识 × （活动采用do）**

**3.对于UML顺序图：**

**通过分析顺序图可以实现对用例图的精化**

**顺序图主要关注多个对象之间的控制流，但也可用于描述对象之间的数据流**

**顺序图的第二列通常对应于控制对象 ×（边界对象）P142**

**顺序图的一种等价描述方式是协作图**

**4.计划内沟通：问题陈述、客户评审、项目浏览、同行评审、现状浏览、集思广益、发布、事后浏览**

**计划外沟通：需求澄清、需求变化、问题求解**

**5.对于UML用例图：**

**一幅用例图可能涉及多个参与者，用例采用自然语言撰写，用例之间可具有继承关系 √**

**参与者位于系统边界之内 ×**

**6.班级—学生不具有继承关系**

**7.对于UML类图，下列哪类人员不关注其内容**

**应用域专家、系统分析师、对象设计师 √**

**客户 ×**

**8.对于框架技术：**

**使用框架可提高软件系统的可重用性和可扩展性**

**框架一般面向特定技术或特定应用领域**

**黑盒框架的扩展需依赖于继承及动态绑定 ×白盒P249**

**中间件框架主要用于分布式环境下的应用程序**

**9.软件开发过程中，一个错误发现的越晚，为改正它所付出的代价就越大**

**10.从实现的角度看，对源代码进行优化属于重构 P296**

**模型转换：模型→模型，优化或简化原始模型**

**重构：源代码的转换，提高可读性和可修改性**

**正向工程：对象模型→源代码模板**

**逆向工程：源代码→模型**

**三、**

**1.内部产品：test manual测验手册，status report状态报告，class model 类模型，source code**

**交付产品：user manual用户手册，specification规格说明**

**2.repository仓库——数据库管理系统、编译器、软件开发环境**

**cs结构——web浏览器系统**

**peer to peer对等——实时聊天系统**

**事件驱动风格：Field系统、Win32 GUI程序**

**面向服务架构风格：金蝶EAS (Enterprise Application Suite)**

**正交架构风格：汽修服务管理系统**

**目前比较好的MVC：老牌的有Struts，Webwork。新兴的MVC框架有Spring MVC，Tapestry，JSF等。还有，如Dinamica，VRaptor等**

**黑板系统风格：黑板架构实现的基本出发点是已经存在一个对公共数据结构进行协同操作的独立程序集合。**

**面向服务SOA架构风格：服务请求者、提供者、注册中心、发布、发现、绑定调用**

**Orthogonal softwara:正交软件架构：组织层、线索的组件**

**3.UML——unified modeling language统一建模语言**

**OMT:object modeling technology 对象建模方法**

**OCL（Object Constraint Language）（考到） 对象约束语言**

**OOAD（Object Oriented Analysis And Design)（考到）面向对象分析和设计**

**FRIEND: First Responder Interactive Emergency Navigational Database**

**ODD：The Object Design Document 对象设计文档**

**CMM：Capability Maturity Model for Software 软件能力成熟度模型**

**LOC：length of code (代码行)**

**COCOMO：constructive cost model 建筑成本模型**

**Application programmer’s（programming） interface (API) 应用程序编程接口**

**ISO ： International Standard Organization 国际标准化组织**

**OSI ： Open System Interconnection （当时写下来觉得没什么用，没想到居然真的考到这个了！） 开放式系统互联**

**OOSE：object-oriented software engineering 面向对象软件工程**

**RAD ：Requirement Analysis Document 需求分析文档**

**CRC；Class Responsibility Cooperation 计算机网络的定义**

**ATAM: Architecture Tradeoff Analysis Method（体系结构权衡分析方法）**

**四、**

**1.实体对象：incident事件、dispatcher调度者、fieldOfficer现场工作人员、emergencyReport紧急情况报告**

**边界对象：acknowledgmentNotice确认通知、dispatcherStation调度基站、reportEmergencyButton报告紧急情况按钮、emergencyReportForm紧急情况报告表格、fieldOfficerStation现场工作基站、IncidentForm事件表格**

**控制对象：reportEmergencyControl报告紧急情况控制、manageEmergencyControl管理紧急情况**

**P137**

**10-11卷子**

**一、**

**1.改进程序设计技术，采用先进的编程思想是解决软件危机的唯一途径 F**

**2.瀑布模型的各阶段之间具有明确的顺序关系，并且前一阶段的输出文档是后一阶段的输入文档。 T**

**3.可行性研究的根本目的不是解决问题，而且确定问题是否值得去解。 T**

**4.代码行技术适用于估算软件成本的方法之一。 T**

**5.需求分析阶段的任务是准确回答“系统要怎么做”的问题。F**

**6.程序设计的三种基本结构是顺序、循环和选择。 T**

**7.面向对象软件工程方法学是以算法为核心，整个开发过程是基于功能分析和功能分解。 F**

**8.用黑盒法测试时，测试用例是根据程序内部逻辑设计的。F**

**9.单元测试中使用的驱动程序用来模拟主程序，替代被测试模块所调用的程序模块。 F**

**10.等价划分、边界值分析和错误推测法等式适用于白盒技术的测试技术。 F**

**FTTTFTFFFF**

**二、**

**1.不属于软件危机的表现形式的：**

**对软件开发成本和进度估计不明确**

**软件不可维护性**

**软件没有适当的文档资料**

**软件产品不能使用 ×**

**2.解决软件危机的途径主要是：**

**使用先进的程序设计思想**

**提高开发人员的编码水平**

**从技术和组织管理两方面采取措施 √**

**矿大组织规模**

**3.软件开发时，一个错误发现的越晚，为改正它所付出的代价就越大**

**4.关于软件生命周期的叙述中：**

**瀑布模型是软件开发过程中的唯一标准化模型 ×**

**软件的生命周期实在生产软件产品时，将一个复杂的生产过程分解成简单的几个阶段√**

**软件的生命周期的划分随开发的软件对象不同可以变化 √**

**总体设计即概要设计 √**

**5.下列属于可行性研究阶段的任务有 成本/效益分析**

**6.不具有继承关系：**

**小汽车——大众牌小汽车 √ 人员——经理√ 教师——教授 √**

**图书馆——期刊阅览室 ×**

**7.从软件开发的整个过程来看，工作量最大的是软件开发的软件测试阶段**

**8.瀑布模型本质上是线性模型（增量模型× 螺旋模型× 各阶段没有明显界限×）**

**9.估算软件规模（成本）的方法是功能点分析技术**

**10.条件覆盖的涵义是判定表达式中的每个条件都要取到各种可能的结果**

**三、**

**1.什么是软件工程？软件生命周期一般包含哪几个阶段？**

**软件工程是指导计算机软件开发和维护的一门工程学科，包括技术和管理两个方面。**

**软件生命周期一般包含三个阶段：**

**软件定义：问题定义，可行性研究，需求分析**

**软件开发：总体设计、详细设计、编码实现、软件测试**

**软件维护：改正性、适应性、完善性、预防性维护**

**软件的生命周期定义为6个阶段**

**1)问题定义与可行性研究/问题是什么及有可行性解吗?/关于系统目标的定义、规模、经济效益分析系统的高层抽象逻辑模型2)需求分析/系统需要做什么?/系统的逻辑模型：数据字典、数据流图、算法描述3)系统设计(3.1)系统概要设计/如何解决这个问题?/可能的解法：系统流程图、系统层次图和结构图(3.2)系统详细设计/如何具体实现系统?/编码规格说明、HIPO图、PDL 4)编码/获得正确的程序/源程序规范、源程序、单元测试结果分析5)测试/符合需求的软件系统/集成和系统测试方案、测试结果分析6)运行与维护/持续地满足用户需要/相关记录**

**3.** **黑盒测试主要关注构件的输入/输出行为，黑盒测试不涉及构件内部结构，也不测试构件的行为或是结构。白盒测试主要关注构件内部结构，白盒测试要确保对象动态模型的每个状态以及对象之间每个交换行为都要被测试到，而这些内容与特定的输入/输出行为不相关联。因此，白盒测试要先于黑盒测试。**

**白盒测试用例设计：逻辑覆盖法、语句覆盖，判定覆盖，条件覆盖，判定-条件覆盖，条件组合覆盖，路径覆盖；**

**黑盒测试用例设计：等价类划分法，边界值分析法，因果图法，错误推断法，功能图法**

**4.改正性维护、适应性维护、完善性维护、预防性维护**

**CMM(软件能力成熟度模型)：初始级(initial)，可重复级(repeatable)，已定义级(defined)，已管理级(managed)，优化级(optimizing)**

**四、**

**1.** **写出三种软件模型及其优缺点，并举个栗子来论述一下**

**瀑布模型 ：传统的生命周期模型**

**优点：阶段间的顺序性和依赖性，推迟实现的观点，保证质量的观点。**

**缺点：不适合需求模糊的系统，开发初始阶段很难弄清系统需求。**

**快速原型模型 ：在获取一组基本的需求定义后，一个目标系统的最初版本-原型，并把它交给用户试用、补充和修改，再进行新的版本开发。反复进行这个过程，直到得出系统的“精确解”，即用户满意为止。**

**优点：逼真的模型，可以使用户迅速作出反馈，循环回溯迭代**

**缺点：一种可能来自用户，他们舍不得将“活生生”的原型废弃不用，要求开发者仅做修改，就交付使用，另一种常常来自开发者，当他们熟悉原型后，明知它有不足，却不愿全部推倒重来，宁可在最终系统中保留一部分不理想的程序。这些偏向如不纠正，都将影响软件开发质量。**

**螺旋模型 ：是瀑布模型和演化模型的结合，并增加了风险分析**

**优点：在项目的所有阶段都考虑各类风险，从而能在风险变成问题之前降低它的危害。**

**缺点：难以是用户相信演化方法是可控的，过多的迭代周期，也会增加开发成本和时间。**

**其他知识点：**

**Agility敏捷**

**Hazards危害**

**Malicious恶意的**

**Vendors供应商**

**Stakeholders利益相关者**

**SaaS：Software-as-a-Service**

**1.OCL：关键字context标明该表达式所适用的实体，关键字后面会是如下关键字inv,pre,post中的一个，分别对应UML版型《invariant》《precondition》《postcondition》，然后就是实际的OCL表达式（例子见图）**

**2.五种图的定义：** **Activity diagrams, which show the activities involved in a process or in data processing .**

**Use case diagrams, which show the interactions between a system and its environment.**

**Sequence diagrams, which show interactions between actors and the system and between system components.**

**Class diagrams, which show the object classes in the system and the associations between these classes.**

**State diagrams, which show how the system reacts to internal and external events.**

**3.软件测试：** **Testing can only show the presence of errors in a program. It cannot demonstrate that there are no remaining faults.**

**Development testing is the responsibility of the software development team. A separate team should be responsible for testing a system before it is released to customers.**

**Development testing includes unit testing, in which you test individual objects and methods component testing in which you test related groups of objects and system testing, in which you test partial or complete systems.**

**When testing software, you should try to ‘break’ the software by using experience and guidelines to choose types of test case that have been effective in discovering defects in other systems.**

**Wherever possible, you should write automated tests. The tests are embedded in a program that can be run every time a change is made to a system.**

**Test-first development is an approach to development where tests are written before the code to be tested.**

**Scenario testing involves inventing a typical usage scenario and using this to derive test cases.**

**Acceptance testing is a user testing process where the aim is to decide if the software is good enough to be deployed and used in its operational environment.**

**4.软件进化：Legacy systems are older software systems, developed using obsolete software and hardware technologies, that remain useful for a business.**

**It is often cheaper and less risky to maintain a legacy system than to develop a replacement system using modern technology.**

**Software re-engineering is concerned with re-structuring and re-documenting software to make it easier to understand and change. （过后维护）**

**Refactoring, making program changes that preserve functionality, is a form of preventative maintenance.（预防性维护）**

**5.（可依赖的系统）** **The use of redundancy（冗余） and diversity in hardware, software processes and software systems is essential to the development of dependable systems.**

**6.** **（可信赖的工程）Reliability metrics include probability of failure on demand (POFOD), rate of occurrence of failure (ROCOF) and availability (AVAIL).**

**Dependable system architectures are system architectures that are designed for fault tolerance.**

**7.** **（Ch16 Component-based software engineering）A component is a software unit whose functionality and dependencies are completely defined by its interfaces**

**8.（项目计划）The COCOMO II costing model is a mature algorithmic cost model that takes project, product, hardware and personnel attributes into account when formulating a cost estimate.**

**9.** **（Ch25 Configuration management）System releases include executable code, data files, configuration files and documentation. Release management involves making decisions on system release dates, preparing all information for distribution and documenting each system release.**

**1.软件架构风格（software architecture style）,又称软件架构惯用模式（software architecture idiomatic paradigm），是描述某一特定应用领域中系统组织方式的惯用模式，作为“可复用的组织模式和习语”，为设计人员的交流提供了公共的术语空间，促进了设计复用与代码复用。**

**使用架构风格的优势**

**它极大地促进了设计的重用性和代码的重用性，并且使得系统的组织结构易被理解。**

**即使没有给出具体实现细节，依然可以通过“客户/服务器”架构风格大致推测出系统的组成结构和工作方式。**

**使用标准的架构风格可较好地支持系统内部的互操作性以及针对特定风格的分析**

**如：“管道/过滤器”风格可用来分析调度、吞吐量、延迟，和死锁等问题。**

**架构风格的基本属性**

**设计元素的词汇表（包括组件、连接件的类型以及数据元素，例如：管道，过滤器，对象，服务等）**

**配置规则：决定元素组合的拓扑约束**

**例如限制某一风格中的组件至多与其它两个组件相连。**

**元素组合的语义解释以及使用某种风格构建的系统的相关分析**

**2.** **面向对象（Object-Oriented）方法是80年代初期提出的一种新型的程序设计方法，它彻底改变了过去数据流、事物流分析方式的缺点，采用直接对问题域进行自然抽象的方法，并逐渐发展成包括面向对象分析、设计、编程、测试、维护等一整套内容的完整体系。**

**该架构风格从现实世界中客观存在的事物出发，强调直接以问题域中的事物为中心来思考问题、认识问题，根据这些事物的本质特征，将其抽象为系统中的对象，并作为系统中的基本构成单位。**

**优点：**

**（1）对象隐藏了其实现细节，所以可以在不影响其它对象的情况下改变对象的实现，不仅使得对象的使用变得简单、方便，而且具有很高的安全性和可靠性。**

**（2）设计者可将一些数据存取操作的问题分解成一些交互的代理程序的集合。**

**缺点**

**当一个对象和其它对象通过过程调用等方式进行交互时，必须知道其它对象的标识。无论何时改变对象的标识，都必须修改所有显式调用它的其它对象，并消除由此带来的一些副作用。**

**16.敏感点影响一个质量属性，权衡点影响多个质量属性**

**Eg. 2010年软件架构师考试题**

**\_\_\_(62)\_\_\_是实现一个特定质量属性的关键特征，该特征为一个或多个软件构件所共有。**

**“改变加密的级别可能会对安全性和性能都产生显著的影响”，这是一个对系统\_\_\_(63)\_\_\_的描述。**

**17.软件质量困境（ Dilemma ）：如果生产了一个存在严重质量问题的软件系统，你将受到损失，因为没有人想去购买。**

**另一方面，如果你花费无限的时间、极大的工作量和高额的资金来开发一个绝对完美的软件，那么完成该软件将花费很长的时间，生产成本是极其高昂的，以至于破产。**

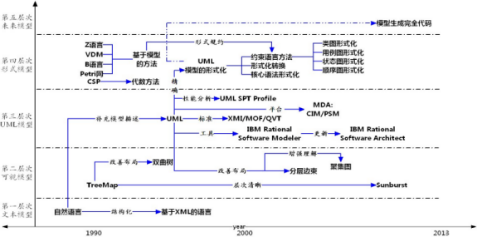
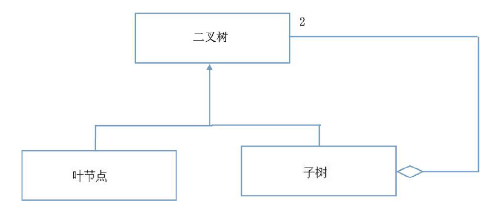
**要么错过了市场机会，要么几乎耗尽所有的资源。**

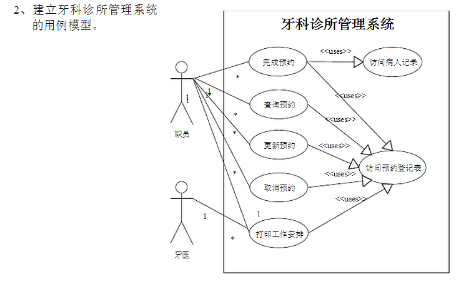
**所以企业界的人们努力达到奇妙的中间状态：一方面，产品要足够好，不会立即被抛弃（比如在评估期）；另一方面，又不是那么完美，不需花费太长时间和太多成本。 [Ven03]**

**18.** **软件质量可以定义为：在一定程度上应用有效的软件过程，创造有用的产品，为生产者和使用者提供明显的价值。架构质量评估：（评估系统体系结构和重要属性）**

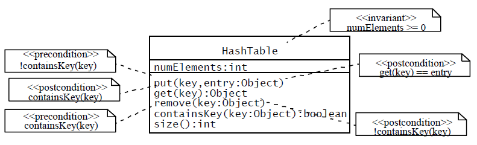
**重要属性：可修改性、可用性、性能、可测试性、易用性、安全性架构质量**

**19. 软件项目管理：**项目经理和项目组织的努力，运用系统理论的方法对项目及其资源进行计划、组织、协调、控制，旨在实现项目的特定目标的管理方法体系**，基本内容**：项目定义、项目计划、项目执行、项目控制、项目结束**关注点：**人员、产品（范围包括：目标和范围、可选的解决方案、技术或管理的约束等 ）**过程：**软件过程框架、**项目：**科学的方法及工具科学的方法及工具，有计划，可控制

管理过程：. 软件项目管理过程软件项目启动（——确定产品的范围）建立项目组织（组织不同类型的项目组成员共同协作完成软件项目）、项目计划（估算、进度安排、风险管理）跟踪与控制（跟踪：其核心在于反映项目变化、提供相关信息的报告，控制：并对项目的未来走向进行预测，再此基础上进行项目的各种调整）

**确定产品范围A work breakdown structure （WBS）**，书写格式：树形结构；目录结构（产品分解：将一个复杂问题分解成若干个小问题过程分解：根据软件开发过程中的活动(分析、设计、编码、测试等)进行分解）**工作包**（**Work Packages）**

WBS结构的叶子节点，即任务分解的最小单位，不可再划分的子任务（不在wbs中的则不在项目中）

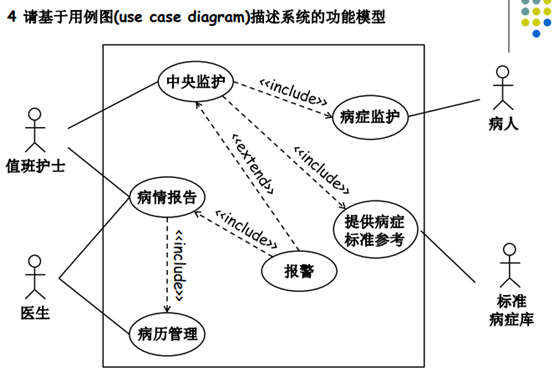
**成本估算的常用方法：**

1）基于已经完成的类似项目进行估算，这是一种常用的也是有效的估算方法2）基于规模的估算：基于LOC；基于FP的估算3）基于经验估算模型的估算： COCOMO模型

**基于规模的估算：**项目计划人员为每个功能或信息域确定一个估算值的范围：乐观值opt，可能值m，悲观值pess；计算各个功能或信息域的规模的期望值S=(Sopt +4Sm + Spess)/6；应用历史生产率数据估算项目的成本

**功能点估算：FP**基于软件信息域的特征(可直接测量)和软件复杂性进行规模估算

**功能点度量方法步骤：**计算信息域特征的值CT；计算复杂度调整值；计算功能点FP

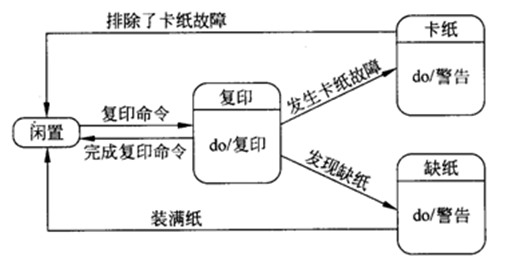
**- 项目进度管理1.**Defining Task Sets2.Sequencing Activities3.Scheduling Steps4.Using Gantt charts (甘特图) for scheduling5.Schedule Tracking

**构建项目任务网络（Project network diagrams）**

**Arrow Diagramming Method (ADM)（AOA）**

**Precedence Diagramming Method (PDM)**

**任务网络的优点:**Show precedence (优先关系) well;Ability to identify critical path （关键路径分析）;Ability to perform “what if” analysis

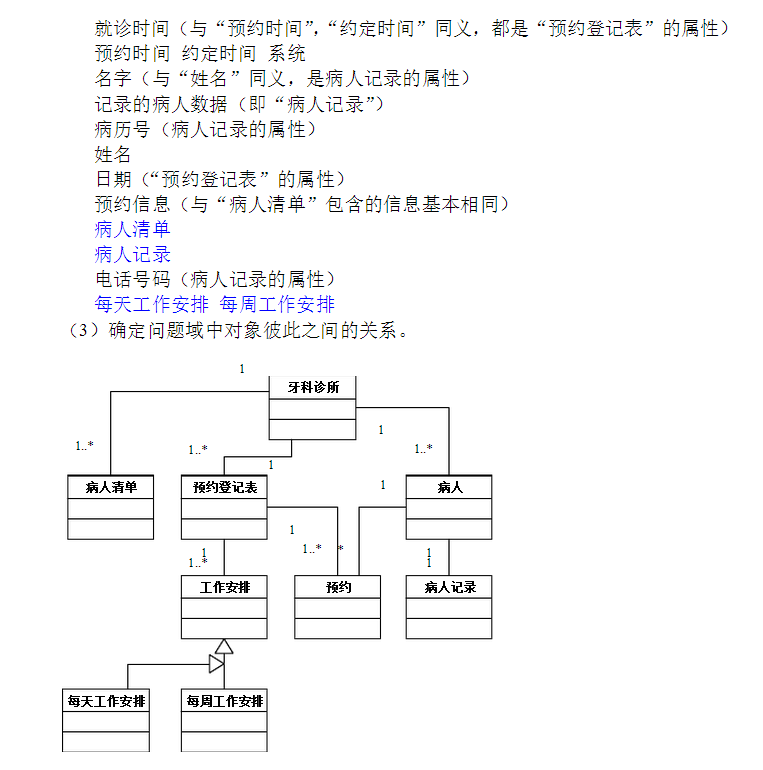
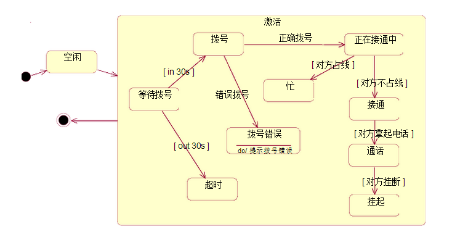
**任务网络的缺点：**Default model assumes resources are unlimited;we need to incorporate Resource Dependencies ourselves when determining the “real” Critical Path;Difficult to follow on large projects

**- 风险管理**（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）（项目、技术、商业风险）

**主动风险管理**的过程：风险识别（已知的和可预测的风险）、风险分析（风险预测：

评价每种风险发生的可能性或概率以及当该风险发生时所导致的后果、风险预测任务：建立一个尺度，以反映风险发生的可能性描述风险的后果、估算风险对项目及产品的影响、标注风险预测的整体精确度以免产生误解 建立风险表：综合考虑风险发生的概率和风险所产生的影响，对风险表排序 风险评估，定义中截线）、风险规划、风险监控

**风险规划(RMMM)**的主要目的就是辅助项目团队制定处理风险的策略（风险管理就是识别评估风险，建立、选择、管理和解决风险的可选方案和组织方法）

**有效的风险规划必须考虑：**风险避免Risk Mitigation）风险监测 Risk Monitoring）风险管理及应急计划（Risk Management Plan）对于每个中截线以上的风险，都应制定RMMM

**Open closed liskov subsititution dependency inversion interface segration**

**软件度量：**度量的作用是为了有效地采用定量的方式来进行管理Measure(noun 测度)：为产品或过程的某些属性的程度、数量、维数、容量或大小提供量化的指示。

Metric(度量)：一个系统、构件或过程具有给定属性的量化测量程度

Indicator（指标）：是一个度量或度量的组合，

**度量对象：产品、过程、项目**

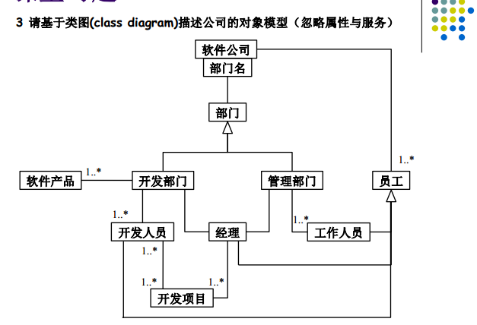
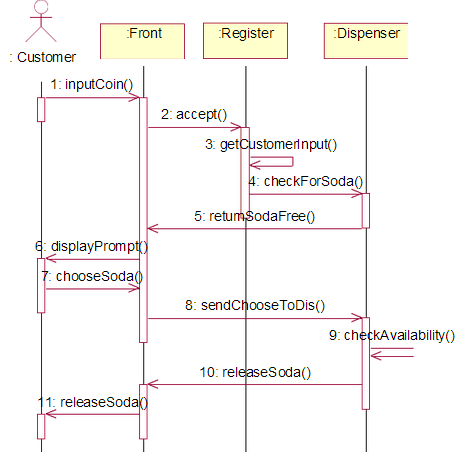
**敏捷方法**

**敏捷开发的基本理念：**强调个体和互动比强调过程和工具更好、可运行的软件、

强调与客户合作、强调响应变化

**敏捷开发在实践**中表现为一种迭代、增 量和持续集成的开发方法。迭代反映了项目的开发节奏，是一个多周 期的开发过程。 增量说明了项目的实际进展，整个项目就 是由很多增量构成的。 持续集成反映了集成增量的过程是持续进 行的、

与软件架构出发点同：都是一个权衡的过程、发目的都是为了提高软 件开发效率、提高软件质量、降低软件成 本，将开发团队的价值最大化。

**特点：**轻详设，种子架构+详细架构，把传统的详设分散到整个过程中，提高效率减少风险、团队设计、简单设计（表达方式的简 单化和现实抽象的简单化）

优秀的**敏捷软件架构的设计过程**一般同 时包含**规划式设计和演进式设**计，具体 体现为初始阶段设计和迭代过程中的设 计

**20.**