**软件工程定义**1是指导计算机软件开发和维护的一门工程学科2采用工程的概念原理技术和方法来开发与维护软件3把经过时间考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来4以经济地开发出高质量的软件并有效的维护它

**软件工程本质特征**1关注于大型程序的构造2中心课题是控制复杂性3软件经常变化4开发软件的效率非常重要5和谐的合作是开发软件的关键6软件必须有效的支持他的用户7软件工程领域中是由具有一种文化背景的人替具由另一种文化背景的人

（**快速原理模型**）**两种基本类型**1.抛弃式原型：最初的原型在完成时并得到用户认可之后，将不会做为交付给用户的最终系统的一部分，而是被抛弃，其目的只是为了收集与验证需求。该类原型是不可执行的2.演化式原型：最初构造的原型具备较高的质量，包含系统的核心功能，然后通过收集需求对其进行不断的改善和精化。该类原型是可执行的，将成为最终系统的一部分 **问题**：1缺乏过程可见性2系统结构通常会很差3需要一些特别的技术，通常与主流技术不兼容 **适用情况：**1适合中小规模的交互系统2可用于大型系统的局部开发（如系统界面），可以和瀑布模型混合使用3生命周期较短的系统

**增量模型：**使用增量模型开发软件时，把软件产品作为一系列的增量构件来设计，编码，集成和测试，每个构件由多个相互作用的模块构成并且能够完成特定的功能 **注意：**1分解成增量构件时应使构件的规模适中2把新构件集成到现有软件中时，所形成的产品必须是可测试的 **优点：**1能够在较短时间内向用户提交可完成一些有用的工作产品2用户由较充裕的时间学习和适应新产品3项目失败的风险低4最重要的系统服务将接受最多的测试

**快速应用开发（RAD）**侧重于短开发周期的增量过程模型，是瀑布模型的高速变体，通过基于构件的构建方法实现快速开发，多个团队并行进行开发 **优点**：1RAD模型强调可复用程序构件的开发，支持多小组并行工作以缩短整体工期 **缺点**：1需要大量的人力资源来创建多个相对独立的RAD团队2如果没有在短时间内为急速完成整个系统做好准备，RAD项目将会失败3如果系统不能被合理的模块化，RAD将会带来很多问题4技术风险很高的情况下不宜采用RAD

**形式化方法模型**1形式化方法模型是一种基于形式化数学变换的软件开发方法2凡是采用严格的数学语言，具有精确的数学语义的方法都称为形式化方法3它可以系统规格说明转换为可执行的程序 **主要特点**1用数学记号表达的详细形式化规格说明2设计实现和单元测试等开发过程由变换开发过程代替3通过一系列变换将形式化规格说明细化为程序**（形式化程序求精）**

**净室方法的观点（20世纪80年代）1**通过第一次正确的书写代码增量，并在测试前验证其正确性来避免成本很高的缺陷消除过程2它的过程模型：在代码增量集成到系统的同时，进行代码增量的统计质量验证3系统的功能随时间增加

**动态视角：RUP阶段模型** 初始 细化 构造 移交 **实践观点：RUP的好的实践**1迭代的开发软件2管理需求3使用基于构件的体系结构4建立可视化模型5不断的验证软件质量（贯穿整个开发）6控制对软件的变更，使每个修改需求可接受和可跟踪

**极限编程**：敏捷过程中最重要的部分，广泛应用于需求模糊且经常改变的场合

**可行性研究任务：**使用最小的代价在尽可能短的时间内确定问题是否能够解决

**技术可行性：**是指决策的技术和决策方案的技术不能突破组织所拥有的或有关人员所掌握的技术资源条件的边界 **经济可行性**：是指可以使用的资源的可能性（资源包括人力资源自然资源和资金条件）

**操作可行性：**是指系统投产后运行或操作条件，对系统可行性的支持

**系统流程图：**描绘物理系统的工具 **数据流图：**描绘系统的逻辑模型 **用途：**1基本目的是利用它作为交流信息的工具2作为分析和设计的工具3可以逻辑模型考虑系统的物理实现

**需求分析任务**：1必须定义软件应完成的功能，要求建立功能模型2必须理解并描述问题的信息域，应该建立数据模型3必须描述作为外部事件结果的软件行为，要求建立行为模型

**需求获取4个方法：1**访谈2简易的应用规格说明技术（面向团队，用户于开发者合作）3面向数据流自顶向下求精4快速模型

**软件设计任务：**从工程管理的角度来看，设计软件分概要设计与详细设计两部分完成

**总体设计的过程：**1划分出组系统的物理元素-程序，文件，数据库，过程和文档等2每个元素还是黑盒子（1确定最佳方案2结构设计3数据库设计4测试计划5文档，审查）

**设计原理：1模块化**。就是把程序划分成独立命名且可独立访问的模块，每个模块完成一个子功能，把这些模块集成起来构成一个整体，可以完成指定功能，满足用户的需求**2抽象。**现实世界中一定事物，状态或过程之间总存在着某些相似的方面（共性）。把这些相似的方面集中和概括起来，暂时忽略它们之间的差异，这就是抽象**3自顶向下，逐步求精。**将软件体系结构自顶向下，对过程细节和数据细节从抽象到具体，逐层细化，直到用编程语言的语句能够实现为止。**4信息隐藏和局部化**。应该这样设计和确定模块，使得一个模块内包含的信息（过程和数据）对于不需要这些信息的模块来说是不能访问的。局部化的概念和信息隐藏的概念是密切相关的，所谓局部化是指把一些密切的软件元素物理的放的彼此靠近。显然局部化有助于实现信息隐藏**5模块独立性。**每个模块完成一个相对独立的子功能，并且与其他模块间的接口简单

**判定表**表示复杂的条件（input）组合与应做动作（output）之间的对应关系。判定表的出现先于软件工程十几年，但不适用于过程设计。**判定表的性质**：提供了一种把各种动作与条件转换为表格形式的表达式。**判定表的好处：**不容易产生误解，还可作为一种表驱动算法的机器直接使用。

**伪代码：**伪代码是一种介于自然语言和形式化语言之间的半形式化语言，是一种用于描述功能模块的算法设计和加工细节的语言，也称为程序设计语言。 **伪代码的语法规则分为外语法和内语法**1外语法应当符合一般程序设计语言常用的语句和语法规则2内语法可以用英语中一些简单的句子短语和通用的数学符号来描述程序应执行的功能

**面向数据结构的设计**是根据问题的数据结构定义一组映射，把问题的数据结构转换为问题解的程序，最终目标是得出对程序处理过程的描述

**软件测试定义**：为了发现程序中的错误而执行程序的过程。**注意：**1只能尽可能差错，不能证明程序中没错2测试员与程序员不应是同一人

**MYERS软件测试的目的：**1测试是程序的执行过程，目的在于发现错误2一个好的测试用例在于能发现至今未发现的错误3一个成功的测试是发现了至今未发现的错误的测试。

**单元测试（白盒）定义：**集中检测软件设计的最小单元模块，目的在于发现各模块内部可能存在的各种差错。**应用白盒测试技术**：单元测试从程序内部结构出发设计测试用例，主要使用白盒测试技术。

**主要手段**：1代码审查2制作测试软件：Driver（驱动）和stub（存根）

**驱动程序：**通常也就是一个主程序，她接受测试数据，把这些数据传送给被测试的模块，并且印出有关的结果。

**存根程序：**代替被测试的模块所调用的模块，也可以称为虚拟子程序。

**集成测试：**往往能发先软件设计中的错误，也可能发现需求说明中的错误。1，非渐增式测试。2渐增式测试

**渐增式测试的几种策略：**1自顶向下。**优点：**在早期即对主要控制及关键的抉择进行检验。**问题：**stub只是对低层模块的模拟，测试时没有重要的数据自下往上流，许多重要的测试须推迟进行。在早期不能充分开产人力。2自底向上。

**白盒测试技术三种方法：**1逻辑覆盖。逻辑覆盖程度由弱到强顺序为语句覆盖，判定覆盖，条件覆盖。2基本路径测试。3.循环测试。

**黑盒测试技术-等价划分：定义：**等价划分技术是用来设计发现错误种类的测试用例，以减少必须设计的测试用例总数。**问题**：如何选择适当的子集，使其发现最多的错误。**解决方法：**把所有可能的输入数据划分为若干等价类，这样就可能得到一种假设，即测试某个等价类的代表值就等价于对这一类其他值的测试。**设计测试方案时的步骤：**1设计一个新的测试方案以尽可能多地覆盖尚未被覆盖的有效等价类，重复这一步骤直到所有有效等价类都被覆盖为止。2. 设计一个新的测试方案，使它覆盖一个而且只覆盖一个尚未被覆盖的无效等价类重复这一步骤直到所有无效等价类都被覆盖为止。

**黑盒测试技术边界值分析**1程序最容易在边界发生错误2通常与等价划分结合进行

**软件的维护**1改正性维护2适应性维护3完善性维护4预防性维护

**多态：**根据为请求提供服务的对象不同可以得到不同的行为。**多态实现：**1方法重载。是java实现多态性的一种方式，是指同一个类中的两个或两个以上的方法可以有同一个名字，只要他们的参数声明不同即可。2方法覆盖。覆盖体现了子类补充或改变父类的能力，通过覆盖使一个方法在不同子类间表现出不同的行为

**面向对象建模（3个模型）**对象模型 动态模型 功能模型

**接口定义：**描述了一个类或构件的一组外部可用的服务（操作）集。

**对象（object）**── 对象是类的实例，其名字下边加下划线，对象的属性值需明确给出。

**用例（use case）**── 也称用况，用于表示系统想要实现的行为，即描述一组动作序列（即场景）。而系统执行这组动作后将产生一个对特定参与者有价值的结果。

**状态图定义：**描述了一个对象或一个交互在生存周期内响应事件所经历的状态序列，单个类或者一组类之间协作的行为都可以用状态机来描述。

**关联关系—聚合。定义：**聚合也称为聚集，是一种特殊的关联。它描述了整体和部分之间的结构关系。**分类：**如果在聚合关系中处于部分方的实例可同时参与多个处于整体方实例的构成，则聚合为**共享聚合。**如果部分类完全隶属于整体类，部分类需要与整体类共存，一旦整体类不存在了，则部分类也会随之消失或失去存在价值，这种聚合称为**复合聚合。**

**泛化关系定义：**泛化的关系就是一般类和特殊类之间的继承关系。在UML中一般类亦称为泛化类，特殊类亦称为特化类。泛化针对类型而不针对实例,因为一个类可以继承另一个类,但个对象不能继承另一个对象。泛化可进一步划分为**普通泛化**和**受限泛化**两类.

**用例与用例图(功能模型) 场景**:是用来描述用户和系统之间交互的顺序的步骤**用例:**是为了达到某一用户目标而组合在一起的一组场景。**用例图:**用来显示在系统(或其它实体)内的用例与系统参与者之间的关系。主要使用场合:需求获取，定义，分析。**扩展关系。**向一个用例中加入一些新的动作后构成了另一个用例，这两个用例之间的关系就是扩展关系，后者通过继承前者的一些行为得来通常把后者称为扩展用例.**使用关系。**当有一大块相似的动作存在于几个用例，又不想重复描述该动作，将重复的部分分离为一个用例，两用例间的关系称为使用关系。

**面向对象设计模型：**从面向对象分析到面向对象设计(OOD)，是一个逐渐扩充模型的过程

**分析**是提取和整理用户需求，并建立问题域精确模型的过程。**设计**则是把分析阶段得到的需求转变成符合成本和质量要求的、抽象的系统实现方案的过程。

在实际的软件开发过程中分析和设计的界限是模糊的。分析和设计活动是一个多次反复迭代的过程。与面向对象分析模型一样，也由主题、类与对象、结构、属性、服务等5个层次组成。**大多数系统的面向对象设计模型，在逻辑上都由4大部分组成**。1问题域子系统2人机交互子系统3任务管理子系统4数据管理子系统

**类构件：**面向对象技术中的“类”，是比较理想的可重用软构件，称之为类构件。

**可重用软构件应具备的特点：**模块独立性强 具有高度可塑性 接口清晰、简明、可靠

**类构件的重用方式：**实例重用 继承重用 多态重用

**设计问题域子系统** 面向对象设计。仅需从实现角度对问题域模型做一些补充或修改，主要是增添、合并或分解类与对象、属性及服务，调整继承关系等等。当问题域子系统过分复杂庞大时，应该把它进一步分解成若干个更小的子系统。

**设计人机交互子系统** 在面向对象设计过程中，则应该对系统的人机交互子系统进行详细设计，以确定人机交互的细节。使用由原型支持的系统化的设计策略，是成功地设计人机交互子系统的关键。

**设计任务管理子系统** 1. 分析并发性。通过面向对象分析建立起来的动态模型，是分析并发性的主要依据。2. 设计任务管理子系统。确定事件驱动型任务。确定时钟驱动型任务。确认优先任务。确定关键任务确定协调任务：尽量减少任务数。确定资源需求：软硬件需求

**设计数据管理子系统** 1选择数据存储管理模式：文件管理系统，关系数据库管理系统，面向对象数据库管理系统。2设计数据管理子系统。（1）设计数据格式（与数据存储管理模式有关）（2）设计相应服务。

**面向对象的设计模式（必考）：**state模式 Decorator模式 Adapter模式 observer模式

**软件项目管理定义：**软件项目管理先于任何技术活动之前开始，并且贯穿于软件的整个生命周期之中。

**COCOMO2给出3个层次的软件开发工作量估算模型**，对软件细节考虑的详尽程度逐级增加。

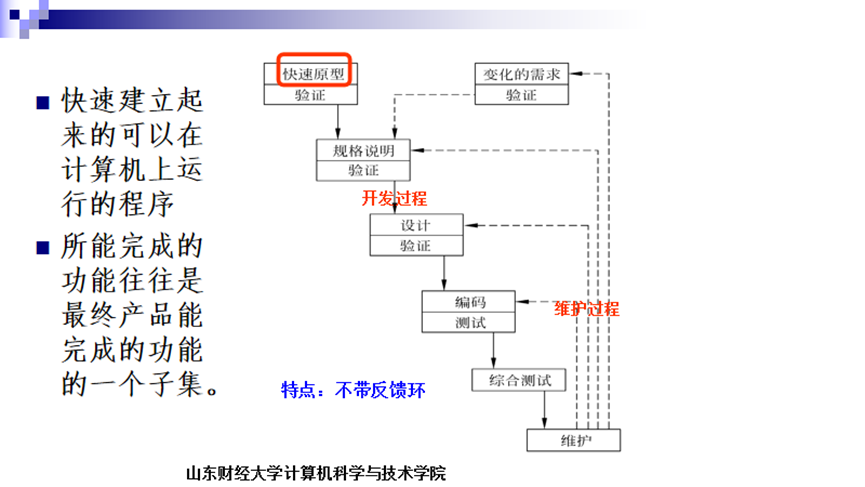
**3个层次的估算模型分别是：** 应用系统组成模型，用于估算构建原型的工作量。早期设计模型，适用于体系结构设计阶段。后体系结构模型，适用于完成体系结构设计之后的软件开发阶段。

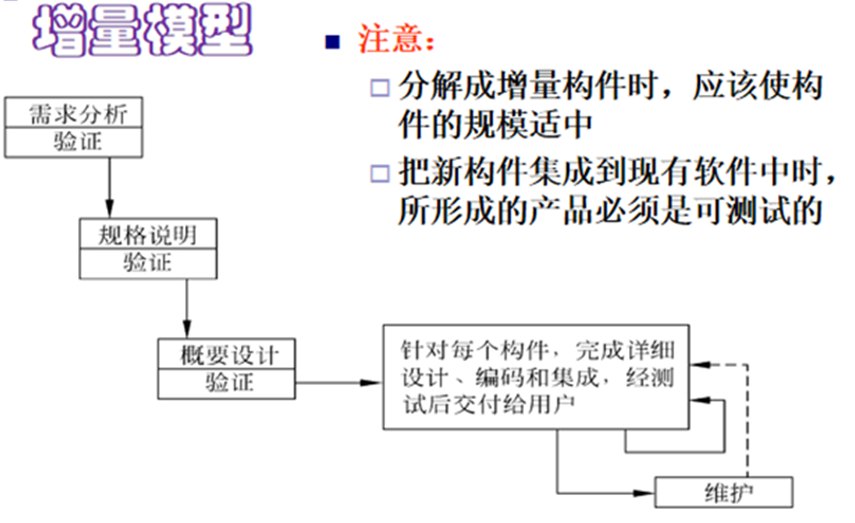
**Gantt图的主要优点：**Gantt图能很形象地描绘任务分解情况，以及每个子任务(作业)的开始和结束时间。具有直观简明和容易掌握、容易绘制的优点。

**软件质量保证措施SQA（三个方面**）1基于非执行的测试（复审或评审），主要用来保证在编码之前各阶段产生的文档的质量；2基于执行的测试（软件测试），需要在程序编写出来之后进行，它是保证软件质量的最后一道防线；3程序正确性证明，使用数学方法严格验证程序是否与对它的说明完全一致。

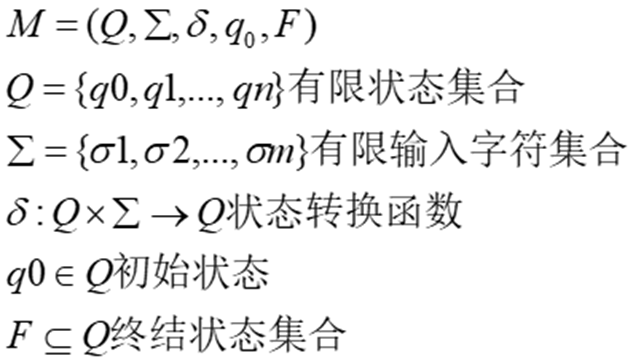
**软件配置管理主要有5项任务：**1. 标识软件配置中的对象。基本对象、聚集对象2. 版本控制。联合使用规程和工具，以管理在软件工程过程中所创建的配置对象的不同版本。3. 变化控制4. 配置审计5. 状态报告

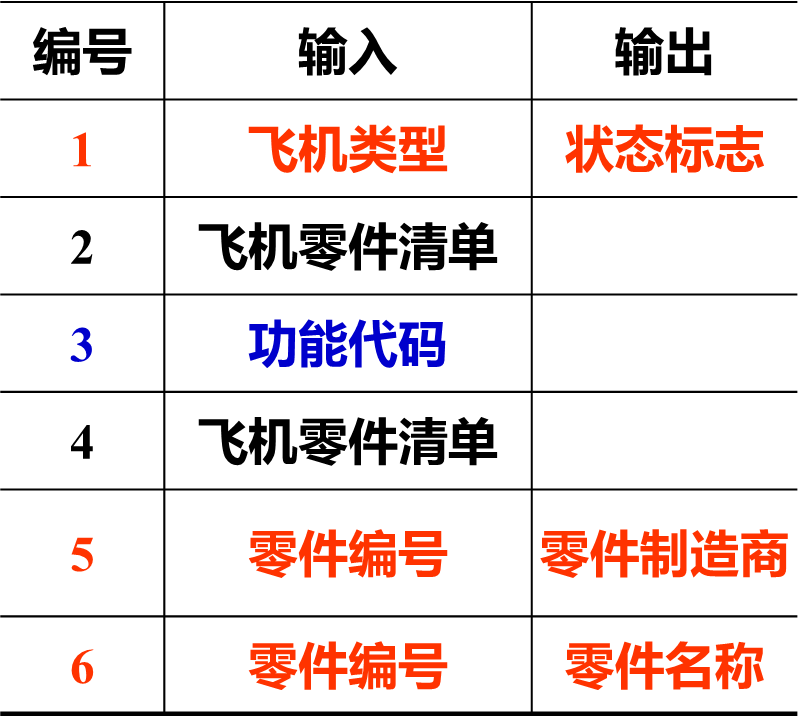
**快速原型模型**





有限状态机表示





接口1,5,6都是数据耦合，传递的都是简单变量

接口3传递一个功能代码，属于控制耦合

当具有逻辑内聚的模块传递功能代码时就传递了控制元素。此时属于控制耦合

图中p、t和u具有公共环境耦合

更新同一个数据库

接口2中，如果s使用或更新p传递给它的零件清单中的所有元素，则属于数据耦合；

如果s只访问该清单的一部分元素，则属于特征耦合

接口4与接口2类似

