

[\(56条消息\) 软件工程期末考试题库（超全）萌萌哒的瑞萌萌的博客-CSDN博客软件工程期末考试题库及答案](#)

[\(56条消息\) 一个神奇的大学科目《软件工程》，知识点总结+测试题，包你不挂科hebtu666-CSDN博客软件工程](#)

[\(56条消息\) 《软件工程导论》学习笔记·山河已无恙-CSDN博客软件工程导论](#)

[\(56条消息\) 软件工程知识点复习总结星星的小孩的博客-CSDN博客软件工程复习](#)

[\(56条消息\) 软件工程概论题库一曲无痕奈何-CSDN博客软件工程题库](#)

[\(56条消息\) 软件工程 实践者的研究方法 中文题答案/1的博客-CSDN博客软件工程实践者的研究方法答案](#)

[\(56条消息\) 《软件工程—实践者的研究方法》读书笔记AlbertHuo的专栏-CSDN博客软件工程实践者的研究方法读书笔记](#)

[\(56条消息\) 软件工程—实践者的研究方法AthlenaA的博客-CSDN博客软件工程实践者的研究方法](#)

[\(56条消息\) 软件工程的经典简答题lockepeak的专栏-CSDN博客软件工程简答题](#)

问答题汇总：

1.简述软件设计的过程

- 软件设计是把许多事物和问题抽象起来，并且抽象它们不同层次和角度，将需求转变为软件陈述的过程，是迭代的过程
- 软件设计，是根据需求规格说明书，对整个设计过程进行计划，然后实施具体的设计过程，即先整体再局部，也是不断迭代和精化的过程
- 需要对生成的设计规格说明书进行评审，启动质量评价的标准，若未通过评审，需重新修改设计，直至评审通过，确定最后定型的过程本身。进入后续阶段，完成软件设计过程

2.衡量模块独立性的两个定性标准是什么？这两个标准的定义分别是什么？在我们的软件设计中，关于模块独立性我们追求的目标是什么？

- 衡量模块独立性的两个定性标准是内聚和耦合
- 耦合是指对一个软件结构内不同模块彼此之间互相互赖（连接）的紧密程度；而内聚则标志一个模块内部各个元素彼此结合的紧密程度
- 我们追求的目标是高内聚低耦合

3.什么是黑盒测试法？有哪些常用的黑盒测试方法？

参考链接：[黑盒测试方法（全）](#)

- 黑盒测试法把程序看成一个黑盒子，完全不考虑程序的内部结构和处理过程，它只检查程序功能是否能按照规格说明书的规定正常使用、程序是否能适当地接受输入数据并产生正确的输出信息
- 黑盒测试方法有等价类划分法、边界值分析法、错误推测法、因果图法....
- 这里补充列举两个老师要求我们重点掌握的等价类划分法和边界值分析法

对于等价类划分法，这里有一个例题：

参考链接：[软件工程期末考试题库（超全）](#)

例：某“调整工资”处理模块接受一个“职称”的变量，根据职称的不同（助教，讲师，副教授，教授）作不同的处理，其中若是助教还必须输入工龄，只有工龄超过两年才能调整工资。请用等价类划分法设计测试用例。

解：

1.划分等价类

输入条件	合理等价类	不合理等价类
职称	1.教授 2.副教授 3.讲师	5.四种职称之外的任意一种
职称兼工龄	4.助教兼工龄大于2年	6.助教兼工龄等于两年 7.助教兼工龄小于两年

2.设计测试用例

输入数据	预期结果	覆盖范围
教授	输入有效，进行调整工资处理	1
副教授	输入有效，进行调整工资处理	2
讲师	输入有效，进行调整工资处理	3
助教 3	输入有效，进行调整工资处理	4
助教 2	输入有效，但不调整工资	6
助教 1	输入有效，但不调整工资	7
工程师	输入无效	5

对于边界值分析法，例题如下：

参考链接：[黑盒测试方法之边界分析法及其示例](#)

例：有一Web系统某查询条件是 1990 年 1 月到 2049 年 12 月，由 6 为数组表示，前 4 位表示年份，后 2 位表示月份

解：

- 从日期的长度考虑，考虑5位的数字字符、7位的数字字符
- 从年份的取值范围角度，考虑年份1989、1990、2049、2050
- 从月份的取值范围角度，考虑月份00、01、12、13

什么是白盒测试法？需要遵循的原则？有哪些常用的白盒测试方法？

参考链接：[软件测试基础知识 - 说一说黑盒与白盒的测试方法 了解->熟悉->掌握->精通](#)

- 白盒测试也称为结构测试或者逻辑驱动测试，是针对被测单元内部是如何工作的测试。它根据程序控制结构设计测试用例，主要用于软件或程序验证
- 需要遵循的原则：

1. 保证一个模块中所有的独立路径至少被测试一次
2. 所有逻辑值均需要测试真（true）和假（false）两种情况
3. 检查程序的内部数据结构，保证其结构的有效性

- 常用的白盒测试方法：

静态测试：不用运行程序的测试，包括**代码检查、静态结构分析、代码质量度量、文档测试**，它可以由人工进行，充分发挥人的逻辑思维优势，也可以借助软件工具（Fxcop）自动进行

动态测试：需要执行代码，通过运行程序找到问题，包括**功能确认与接口测试、覆盖率分析、性能分析、内存分析**等

补充：白盒测试中的逻辑覆盖包括语句覆盖、判定覆盖、条件覆盖、判定/条件覆盖、条件组合覆盖和路径覆盖。六种覆盖标准发现错误的能力呈由弱到强的变化：

- 1.语句覆盖：每条语句至少执行一次。
- 2.判定覆盖：每个判定的每个分支至少执行一次。
- 3.条件覆盖：每个判定的每个条件应取到各种可能的值。
- 4.判定/条件覆盖：同时满足判定覆盖和条件覆盖。
- 5.条件组合覆盖：每个判定中各条件的每一种组合至少出现一次。
- 6.路径覆盖：使程序中每一条可能的路径至少执行一次。

4.什么是耦合?模块的耦合包含哪些类型?

- 耦合是对一个软件结构内**不同模块**之间**相互连接程度的度量**
- 模块的**内聚**包含以下几种类型：偶然内聚或者巧合内聚、逻辑内聚、时间内聚、过程内聚、通信内聚、顺序内聚、功能内聚（不是回答本题，是作为补充）
- 模块的**耦合**包含：内容耦合、公用耦合、控制耦合、标记耦合、数据耦合、外部耦合等

5.什么是编码风格?为什么要强调编码风格?

- 编码风格包含以下内容：程序内部良好的文档（注释）、数据说明、语句构造、输入输出和效率保障
- 由于编码风格对软件的可读性、可维护性、可靠性、可用性很重要，故我们强调编码风格的重要性

6.什么是软件危机?软件危机的表现是什么? 其产生的原因是什么?

- **定义上说**，软件发展的第二阶段末期，由于计算机硬件技术的进步，计算机运行速度、容量、可靠性有显著的提高，生产成本显著下降，这为计算机的广泛应用创造了条件。一些复杂的，大型的软件开发项目提出来了，但是，软件开发技术的进步一直未能满足发展的需要。在软件开发中遇到的问题找不到解决方法，使问题积累起来，形成了尖锐的矛盾，因此导致了软件危机。
- **简单来说**，软件危机是指在计算机软件的开发和维护过程中所遇到的一系列严重问题。

软件危机的典型表现：

1. 对软件开发成本和进度的估计不准确
2. 无法满足用户需求，导致用户很不满意
3. 质量不可靠，经常失效
4. 难以更改、调试和增强
5. 没有合适的文档
6. 软件成本比重上升
7. 软件开发生产率跟不上计算机应用迅速深入的趋势

产生的原因：

(课件版答案)：**客观上**，软件产品开发的复杂程度和难度随软件规模呈指数级别的增长；**主观上**，软件开发人员缺乏工程性的、系统性的方法论；程序员具有编程的能力，但对软件开发这一过程性较强的任务缺乏足够的工程化思维；对软件开发认识的一些误区：如软件神话等；没有将“软件产品研发”和“程序编码”区分清楚

(csdn答案)：从规模和结构、管理、经费、技术、工具几个方面入手，

- (1) 软件的规模越来越大，结构越来越复杂
- (2) 软件开发管理困难而复杂
- (3) 软件开发的经费不断增加
- (4) 软件开发技术落后
- (5) 生产方式落后开发工具落后，生产率提高缓慢

7.软件生存周期包含哪些活动？

软件生存周期的活动包括：**问题的定义、可行性研究、软件需求分析、系统总体设计、详细设计、编码、测试和运行、维护**

补充：






1.流程图画法

组成

流程图一般由由圆角矩形、矩形、菱形、平行四边形、箭头组成。

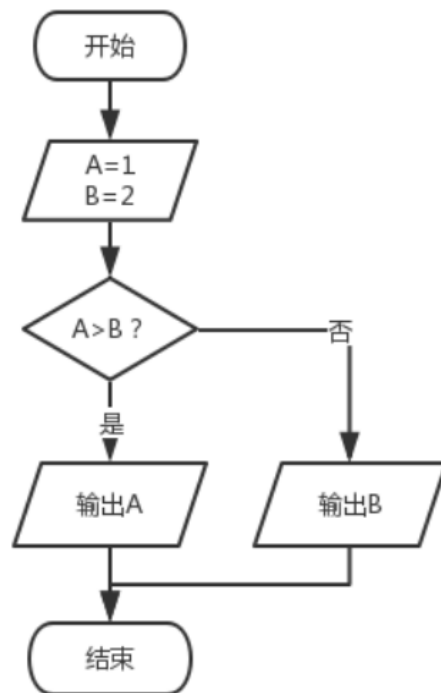
作用

流程图一般都是用圆角矩形来表示“开始”与“结束”，用矩形表示行动方案、普通工作环节，菱形表示判断，平行四边形表示输入输出，箭头表示 workflow 方向。

图形	作用
	“开始”与“结束”
	行动方案、普通工作环节
	判断
	输入输出
	workflow 方向

例子1

比如说我们要定义两个数 $a=1$ ， $b=2$ 然后进行比较输出大的数，那么流程图如下



2.数据流图画法

数据流图（DFD）是结构化系统分析方法的主要表达工具，数据流图，主要是为了说明在一个项目中，数据的处理与流动情况。

一：数据流图的基本成分：



(1) **数据加工**：表示对数据进行的操作, 如“处理选课单”、“产生发票”等，命名时最好使用动宾短语或者主谓词组

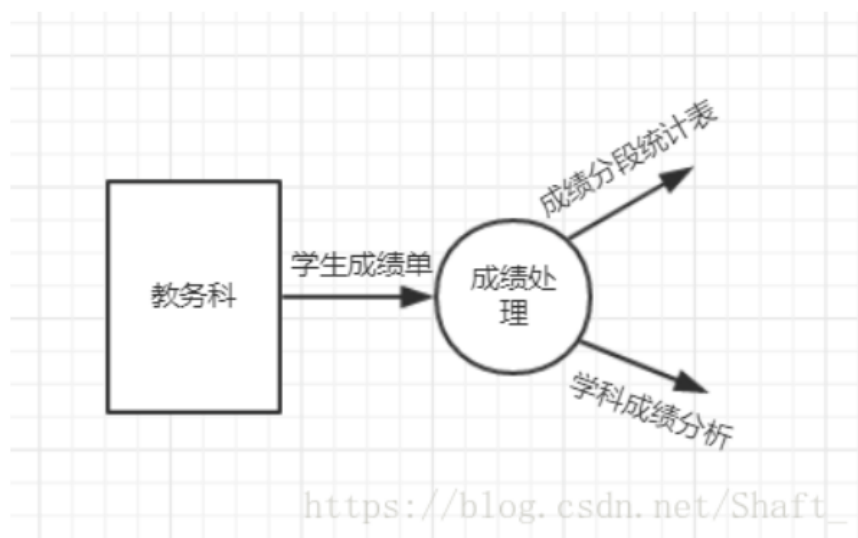
(2) **外部实体**：位于系统之外的信息提供者，数据输入的源点或是数据输出的终点。

(3) **数据流**：数据流可从加工流向加工，也可在加工与数据存储或外部实体之间流动；两个加工之间可有多股数据流。数据流的命名尽量使用简洁易懂的**名词**。**流向数据存储或从数据存储流出的数据流不必命名**

二：数据流图的设计原则：

(1)：父图-子图平衡原则：

子图可以理解为父图中部分环节的细化。例如我们给出父图：



(56条消息) 数据流图案例baidu 38634017的博客-CSDN博客数据流图

参考链接如上

8.假设你要开发一个软件，它的功能是把73624.9385这个数开平方，所得到的结果应该精确到小数点后4位。一旦实现并测试完之后，该产品将被抛弃。你打算选用哪种软件生命周期模型？请说明你做出这样选择的理由。

- 对这个软件的需求很明确，实现开平方功能的算法也很成熟。因此，既无须通过原型模型来分析需求也无须用原型模型来验证设计方案。此外，一旦实现并测试完之后，该产品将被抛弃，因此也无须使用有助于提高软件可维护性的增量模型或螺旋模型来开发该软件。
- 综上所述，为了开发这个简单软件，使用大多数人所熟悉的瀑布模型就可以了

9.什么是调试？什么是测试？二者有何区别？

- 调试（debug）指的是对模块的调试，是程序员交付可运行的代码模块所必须经历的工作
- 软件测试（test）有以下特点：
 1. 测试的目的是为了发现程序中的错误，是为了证明程序有错误，而不是证明程序无错误
 2. 不仅仅是测试程序，还应对开发过程中的所有产品进行测试，包括文档，其目的是为了尽早地、尽可能多地发现并排除软件中潜在的错误

二者区别如下表格所示：

测试	调试
发现错误	找出错误的位置，并排除错误
有计划	被动的
以已知条件开始，使用预先定义的程序，有预知的结果	以不可知的内部条件开始，结果一般不可预见
由独立的测试组完成，在不了解软件设计的条件下完成	由程序作者进行

10.软件测试的目的是什么？

- 测试是一个为了发现程序中的错误为目的程序执行过程
- 一个好的测试用例是能够最大限度地找到至今未发现地错误
- 测试为了证明程序有错，而不是证明程序无错

11.软件测试应该划分为几个阶段？各个阶段应重点测试的内容是什么？

参考链接：[\(56条消息\) 测试过程分为哪些阶段Arcobaleno-CSDN博客测试阶段](#)

软件测试分为四个阶段：单元测试阶段、集成测试阶段、系统测试阶段、验收测试阶段

- 1.单元测试阶段：单元测试是最小单位的测试，也是最初期的测试阶段，一般是以一个函数方法窗口、一个功能模块、都可以看作是一个单元，主要依据的是详细设计文档，主要以白盒测试为主，一般由开发人员完成
- 2.集成测试阶段：集成测试又称组装测试，在单元测试的基础上把软件逐渐组装起来一起继续测试的过程，逐渐组装的过程会出现很多临时版本（迭代测试）集成测试主要是以黑盒为主（当然接口测试也在这个阶段进行）

3. **系统测试阶段**：整个功能全部完成了之后对集成了硬件和软件的完整系统进行**模拟真实的环境模拟**、测试重点主要在于整个系统能否正常运行和整个系统的兼容性测试

4. **验收测试阶段**：由用户参与完成的过程

- alpha阶段：在软件开发过程中由最终用户对软件进行检查，所以是在开发者的场所进行
- beta阶段：在最终用户的实际环境中由最终用户对软件进行检查，所以是在最终用户的场所进行

12. 软件和硬件的区别

1. 软件是设计开发的，而不是传统意义上生产制造的。
2. 软件不会“磨损”
3. 大多数软件根据实际的顾客需求定制的

13. 什么是软件过程？

软件过程是工作产品构建时的一系列**活动、动作和任务的集合**

14. 有哪些通用框架活动？

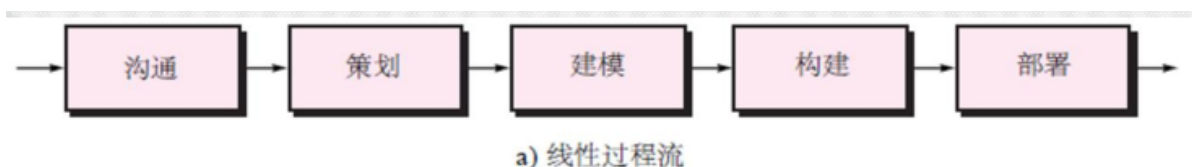
1. **沟通**：目的是理解利益相关者的项目目标，并收集需求以定义软件特性和功能。
2. **策划**：定义和描述了软件工作，包括需要执行的技术任务、可能的风险、资源需求、工作产品和工作进度计划
3. **建模**：利用模型更好地理解软件需求并完成符合这些需求的软件设计
4. **构建**：包括编码和测试以发现编码中的错误
5. **部署**：软件交付到用户，用户对其进行评测并给出反馈意见

15. 什么是过程流？

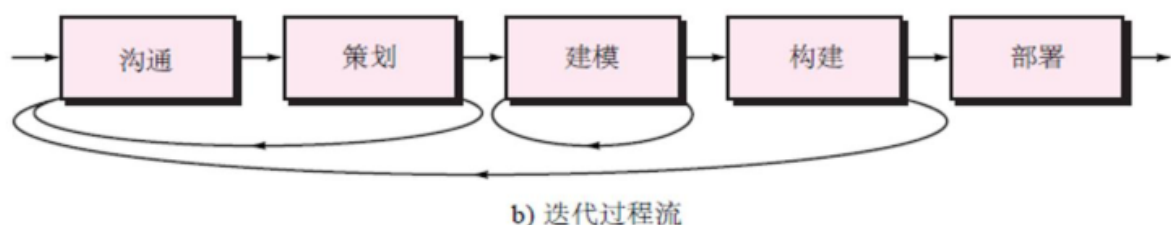
过程流描述在**执行顺序和执行时间**上如何组织框架中的活动、动作和任务

分类：线性、迭代、演化、并行过程流

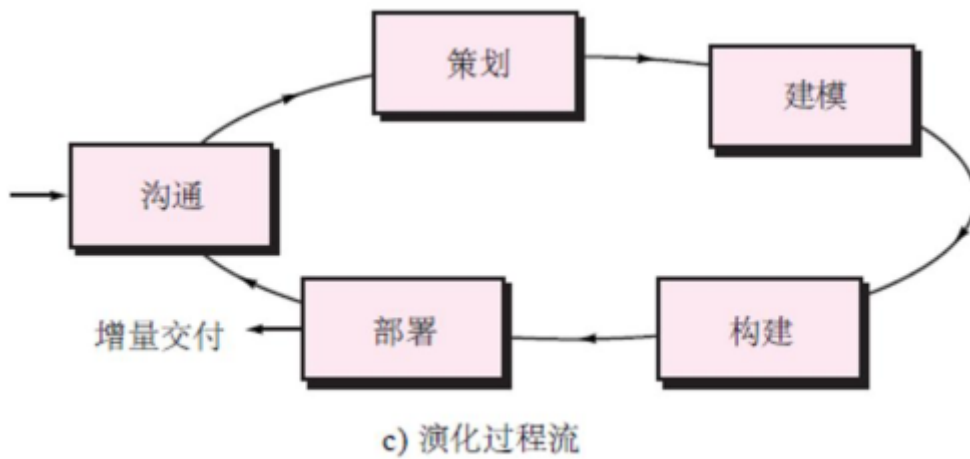
线性过程流：从沟通到部署执行五个框架活动



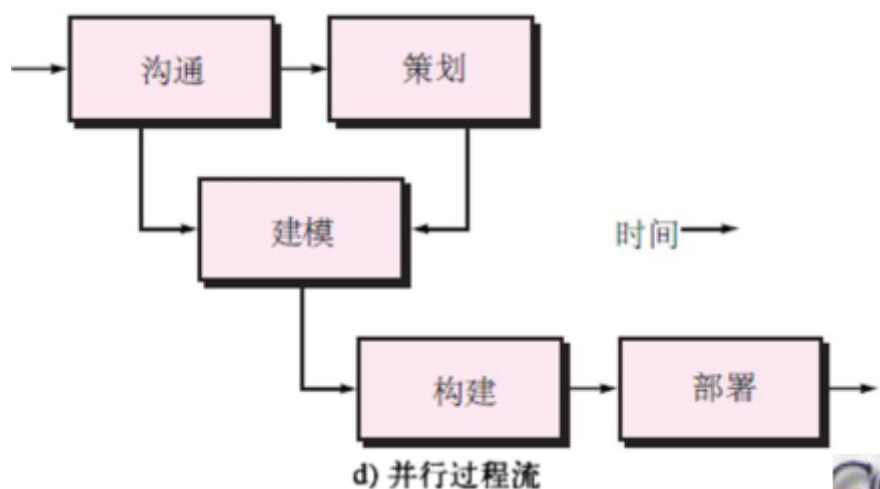
迭代过程流：在执行下一个活动前重复执行之前一个或多个活动



演化过程流：采取循环的方式执行各个活动



并行过程流：将一个或是多个活动与其他活动并行执行



16.什么是过程模式？

过程模式：描述了软件工程工作中遇到的过程相关的问题，明确了问题环境并给出了针对该问题的一种或几种可证明的解决方式

17.什么是过程模型?(重点！！！！)

惯用过程模型是为了改变软件开发的混乱状态，促使软件开发更加有序。

瀑布模型：

又被称为**经典生命周期**，它提出了一个系统的、顺序的软件开发方法。

瀑布模型的**优点：**

- 有利于大型软件开发过程中人员的组织、管理、从而**提高了大型软件项目开发的质量和效率**
- 当需求确定、工作采用线性的方式完成的时候，瀑布模型是一个很有用的过程模型
- 一个有用的过程模型，其中需求是固定的，工作以线性方式完成

缺点：

- 过于理想，缺乏灵活性，很容易产生需求偏差
- 实际的项目很少遵守瀑布模型提出的顺序
- 客户通常很难清楚描述所有的需求
- 客户必须要有耐心，因为只有当项目接近尾声的时候，它们才能得到可执行的程序

综上，故瀑布模型的适用范围是**需求确定，工作能够采用线性的方式完成、小型清晰的项目或长周期的项目、外部环境的不可控因素很少**的软件

V模型:

描述质量保证动作同沟通、建模动作以及早期构建相关的动作之间的联系

V模型强调软件开发的**协作和速度**，将软件实现和验证有机地结合起来，在保证较高的软件质量情况下缩短开发周期

优点：适合**工程量小、人力资源少**并且**开发过程中改动不太大**的项目

缺点：**错误发现时间迟、产生风险的代价高**

增量过程模型

侧重于每一个增量都可以提交一个可以运行的产品

优点：

- 能在**较短的时间内**向用户**提交可完成部分工作的产品**
- 可**规避技术风险**
- **人员分配灵活**：找不到足够的开发人员时，可以先采用增量模型，早期的增量由少量人员实现，如果客户反响较好，则在下一个增量中投入更多人力
- 提高对用户需求的响应：用户看到可操作的早期版本后会提出一些建议和要求，这些我们可以在后续增量中进行调整

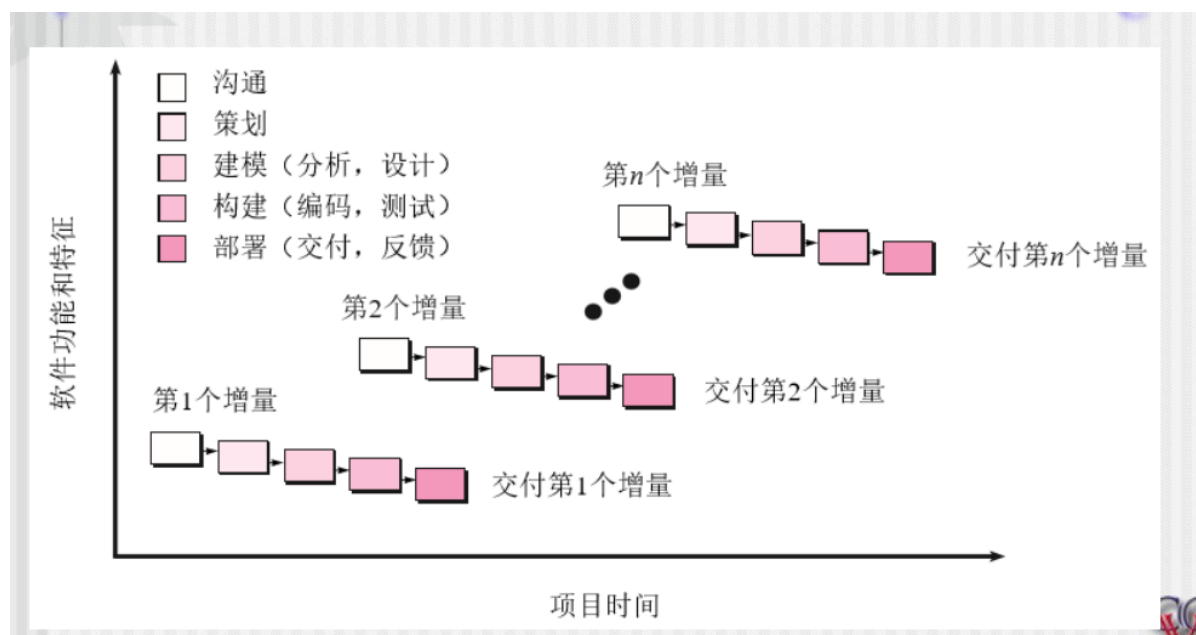
缺点：

- 并行开发构件有可能遇到**不能集成**的风险，软件**必须具备开放式的体系结构**
- 增量模型的**灵活性**使其适应这种变化的能力**大大优于**上面两种模型，但是**也很容易退化为边做边改的模型**，从而使**软件过程的控制失去整体性**

适用范围：

- 进行**已有产品升级或新版本开发**，增量模型是非常适合的
- 对完成**期限要求严格**的作品，可以使用增量模型
- 对**开发领域比较熟悉或者已经有原型系统**，增量模型也是非常适合的
- 项目在既定的商业要求期限之前**不可能找到足够的开发人员**

增量模型示意图：



注：每一个增量都提交一个可以操作的产品，可供用户评估；

第一个增量往往是核心产品....

增量模型应用举例:

应用举例: 开发一个类似于Word的字处理软件

- 增量1: 提供基本的文件管理、编辑和文档生成功能
- 增量2: 提供高级的文档编辑功能
- 增量3: 实现拼写和语法检查功能
- 增量4: 完成高级的页面排版功能

演化过程模型

是**迭代**的过程模型

原型开发: 当**需求很模糊**的时候, 原型开发可以帮助软件开发人员和利益相关者更好地理解究竟需要什么

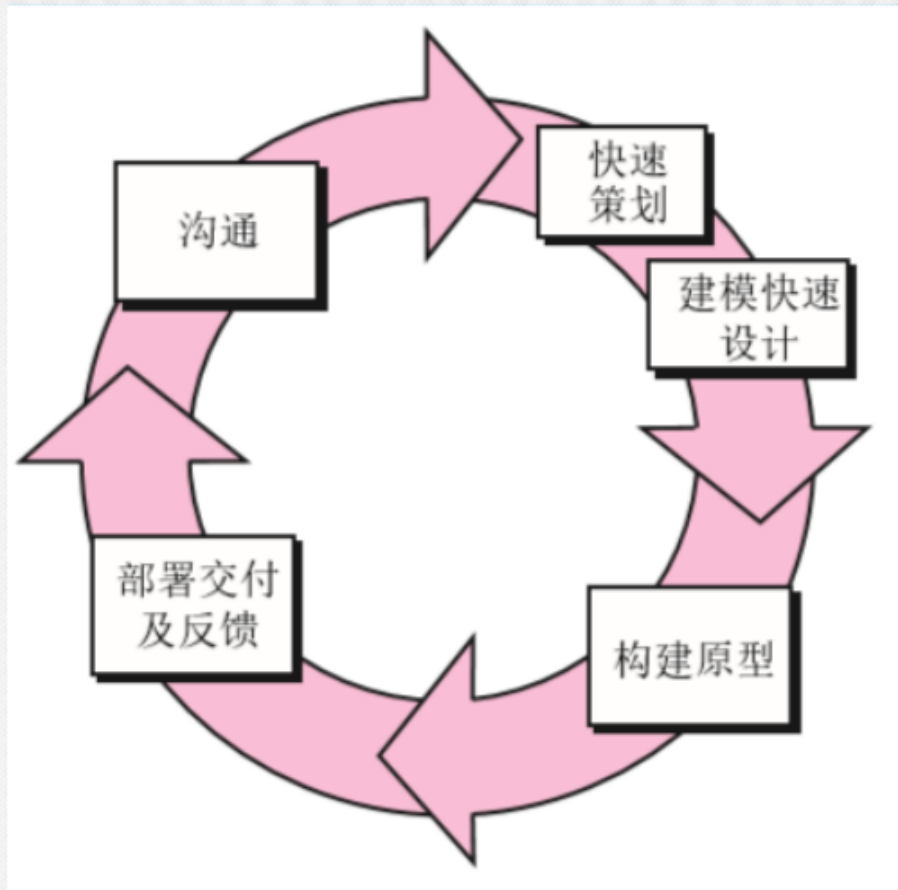
优点:

- 开发者和用户**充分交流**, 采用迭代技术, **逐步澄清模糊需求**, 需求定义比其他模型好得多
- **快速**开发出可以演示的系统, 方便沟通 (PPT)
- 为用户需求的改变提供了**充分的余地**
- 开发**风险低**, 开发**费用低**, 系统**易维护**

缺点:

- **没有考虑软件质量和长期的可维护性**, 系统结构通常较差
- 由于**达不到质量要求**, 产品可能**被抛弃**, 而采用新的模型重新设计

原型开发模型



原型开发模型

应用举例：开发一个教务管理系统

- **第一次迭代：**完成基本的学籍管理、选课和成绩管理功能（6周）
客户反馈（这个在这个模型中非常关键，是这个模型的一大特色）：基本满意，但是对大数据量运行速度慢效率，不需要学生自己维护自己的学籍的功能等
- **第二次迭代：**修改细节，提高成绩统计和报表执行效率（2周）
客户反馈：需要严格的权限控制，报表打印格式不符合要求
- **第三次迭代：**完善打印和权限控制功能（2周）
客户反馈：可以进行正式应用验证

螺旋模型

是一种**风险驱动型**的过程模型生成器，对于**软件集中**的系统，它可以**指导多个利益相关者的协同工作**

优点：

- 结合了**原型的迭代性质**和**瀑布模型的系统性和可控性**特点。
- 强调**风险、阶段质量、提供纠错的机会、使用原型作为风险降低机制**

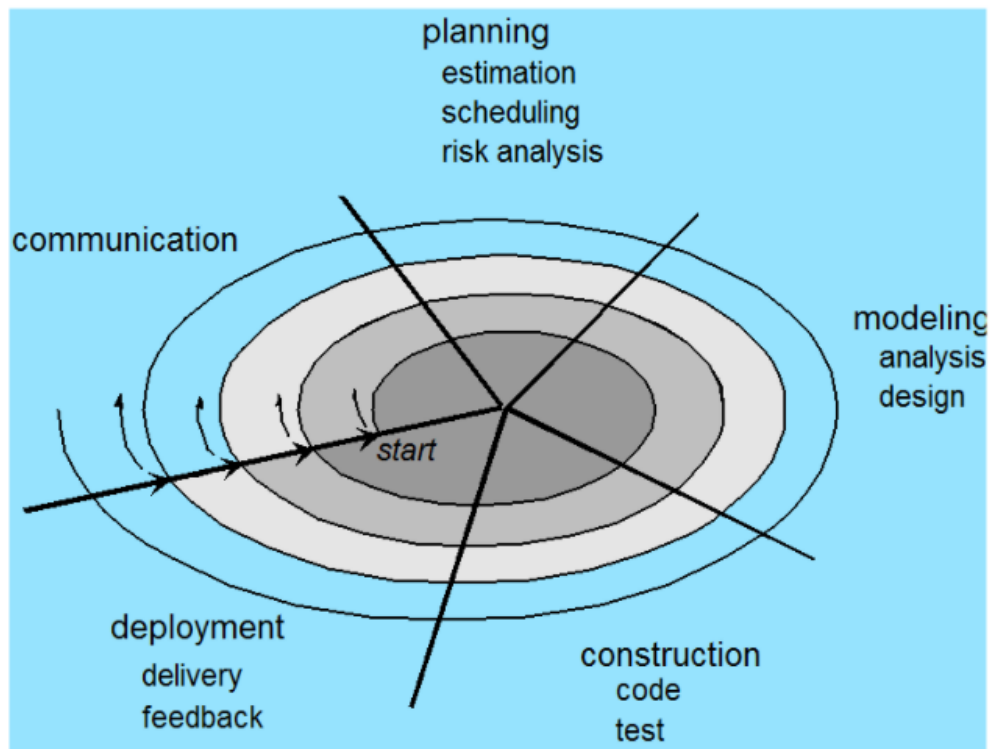
缺点:

- 依赖大量的风险评估专家来保证成功
- 每个阶段都要提出备选方案, 进行风险分析, 研发周期长, 效率低

适用范围: 大型项目

示意图:

Spiral Model



统一过程

是一种“用例驱动、以体系结构为核心、迭代及增量”的软件过程框架, 由UML方法和工具支持。它是一种增量模型, 定义了五个阶段:

1. **起始**: 包括用户沟通和计划活动, 强调定义和细化用例
2. **细化**: 包括用户沟通和建模活动, 重点是创建分析和设计模型
3. **构建**: 细化模型设计, 将设计模型转化为软件构件实现
4. **转换**: 将软件从开发人员传递给最终用户, 并由用户完成beta测试, 然后提供反馈

优点:

1. 任何功能开发后就进入测试过程, 及早进行验证
2. 早期风险识别, 采取预防措施

缺点:

1. 需求必须在开始之前完全弄清楚, 否则可能出现架构错误
2. 必须有严格的过程管理...
3.

18.什么是敏捷过程模型？常见的有哪些？

- 敏捷方法或多或少都遵循敏捷软件开发宣言以及敏捷原则
- **常见的敏捷过程模型**：极限编程、自适应软件开发、动态系统开发方法、Scrum、Crystal、特征驱动开发

19.极限编程？极限编程过程？

极限编程简称XP，是敏捷软件开发使用最广泛的一个方法

包含了**策划、设计、编码和测试**四个框架活动的规则和实践

策划：

1. 开始创造“用户故事”
2. 团队评估每一个故事并分配一个成本（开发周数）
3. 故事被分组到一个可交付增量
4. 承诺在交付日期进行
5. 第一次递增之后，“项目速度”用于帮助估计后续发行版本的发布日期和进度安排

设计：

1. 遵循KIS（保持简洁）原则
2. 鼓励使用CRC卡
3. 设计碰到困按，建议创建"Spike解决方案"
4. 鼓励重构，重构是以不改变代码外部行为而改进其内部结构的方式来修改系统软件的过程

编码：

1. 编码开始之前，建议对故事进行单元测试
2. 鼓励“结对编程”

测试：

1. 所有的单元测试每天都执行
2. "验收测试"由客户规定技术条件，并且着眼于客户可见的、可评审的系统级特征和功能

20.工业级极限编程？

xp的**有机进化** 由XP的最低限要求、以客户为中心、测试驱动精神组成。IXP和XP的主要差别在于其**管理具有更大的包容性，扩大了用户角色，升级了技术实践**

合并了六个新实践：

- 准备评估
- 项目社区
- 项目承租
- 测试驱动管理
- 回顾
- 持续学习

21.需求工程（重点！！！！）

七个任务：起始、获取、细化、协商、规格说明、确认和管理

1. **起始**：项目起始阶段，建立基本的理解，包括对问题、谁需要解决方案、所期望解决方案的性质、与项目利益相关者和开发人员之间达成初步交流合作的结果
2. **获取**：询问利益相关者（客户、用户和其他人）**系统或产品的目标是什么；想要实现什么；系统和产品如何满足业务的需求；最终系统或产品如何用于日常工作**

3. **细化**：在起始和获取阶段获得的信息在细化阶段进行**扩展和提炼**该任务集中于**开发一个精确的需求模型**
4. **协商**：**迭代的方法**给需求**排序**，评估**每项需求**对项目产生的**成本和风险**，表达**内部冲突、删除、组合和修改需求**，以便**各方**均能达到一定的满意度 实现**双赢**
5. **规格说明**：一个规格说明可以是一份写好的文档、一套图形化的模型、一个形式化的数学模型、一组使用场景、一个原型或者上述各项的任意组合
6. **确认**：在这一步对需求工程的工作产品进行质量评估
7. **需求管理**：基于计算机的系统其需求会变更，并且变更的需求贯穿于系统的整个生存期。需求管理是用于帮助项目组在项目进展中标识、控制和跟踪需求以及需求变成的一组获得

21.获取需求（老师画的重点）

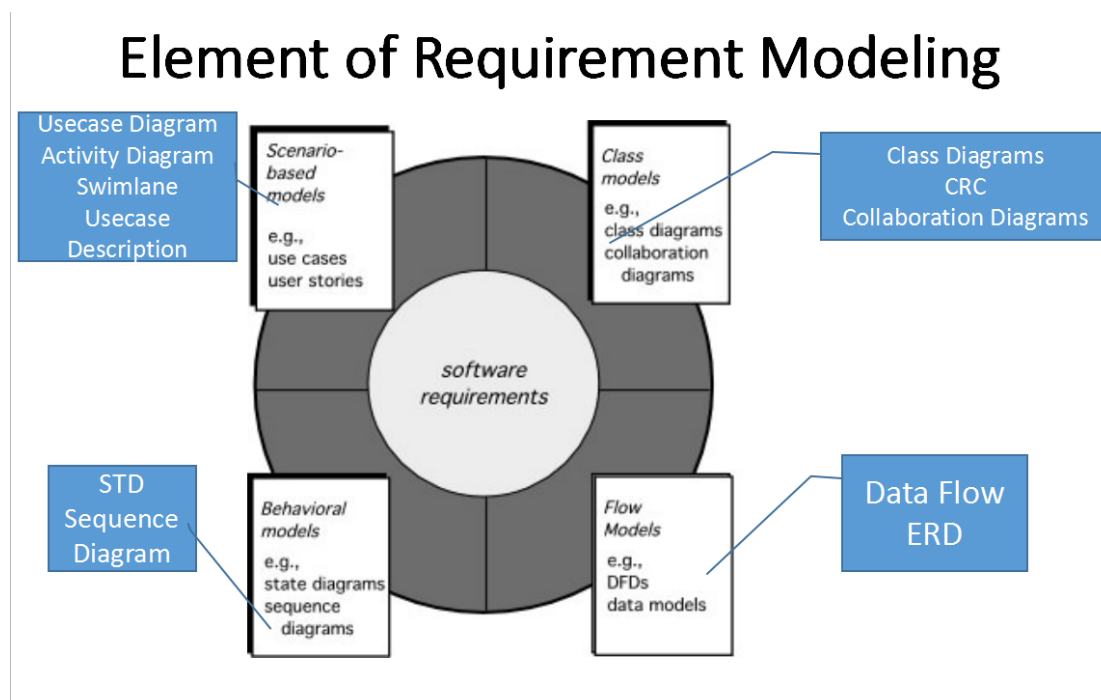
定义：将问题求解、精化、谈判和规格说明等元素结合在一起

利益相关者和开发人员共同完成如下任务：

- ❖ 确认问题
- ❖ 为解决方案的要素提供建议，
- ❖ 商讨不同的方法
- ❖ 描述初步的需求解决方案。

.....

21.需求建模的元素有哪些？



下面是需求模型的元素

在描述需求模型时常用哪些不同的观点？

“我们为什么要构建模型？为什么不直接构建系统本身？答案是我们可以按照如下方式构建模型：突出或强调某些关键的系统特征，同时削弱系统的其他方面。”——Ed Yourdon

如图6-3所示需求模型的每个元素表示源自不同观点的问题。

元素表述用户如何与系统和使用软件时出现的特定活动序列进行交互。基类的元素建模于系统操作的对象，应用在这些对象间影响操作和对（某层级）的操作，以及定义的类型间发生的协作。行为元素描述了外系统，描述了数据对象在流过各种系统功能时是如何转换的。

Scenario-based models

基于场景的模型如：
用例
用户故事

Class models

类模型如：
类图
协作图

模块
information
(Data)

Function

Flow models

行为模型如：
状态图
顺序图

流模型如：
数据流图
数据模型

Behavior

Behavioral models

软件需求

图6-3 需求模型的元素