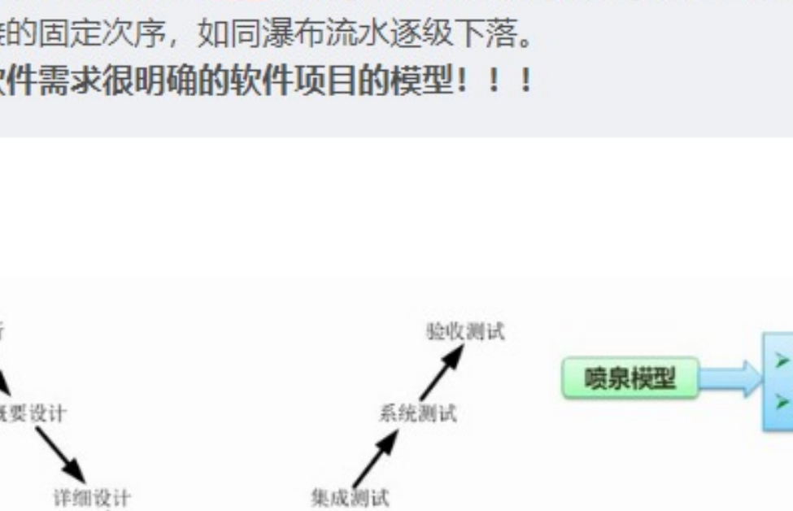
文章目录:

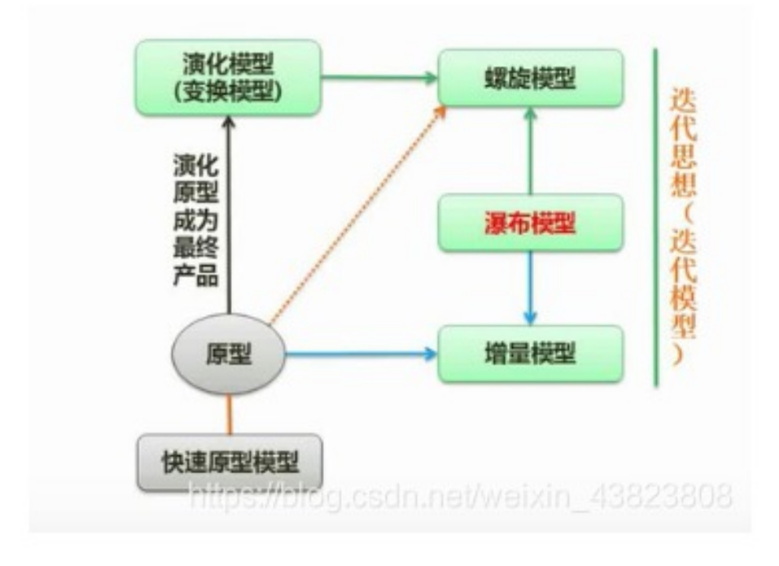
1.软件开发模型



1.1 瀑布模型

> 瀑布模型是将软件生存周期中的各个活动规定为依线性顺序连接的若干阶段的模型，包括需求分析、设计、编码、测试、运行与维护。它规定了由前至后、相互衔接的固定次序，如同瀑布流水逐级下落。
> **瀑布模型以文档作为驱动，适合于软件需求很明确的软件项目的模型！！！**

1.2 V模型



> **V模型是瀑布模型的一个变体，描述了质量保证活动和沟通、建模相关活动以及早期构建相关的活动之间的关系。**

1.3 增量模型

> 增量模型融合了瀑布模型的基本成分和原型实现的迭代特征，它假设可以将需求分段为一系列增量产品，每一增量可以分别开发。**当使用增量模型时，第1个增量往往是核心产品，客户对每个增量的使用和评估都作为下一个增量发布的新特征和功能，这个过程在每一个增量发布后不断重复，直到产生了最终的完善产品。增量模型强调每一个增量均发布一个可操作的产品。**

1.4 演化模型

1.4.1 原型模型

> **原型模型比较适合于用户需求不清、需求经常变化的情况。当系统规模不是很大也不太复杂时，采用原型模型比较好。**

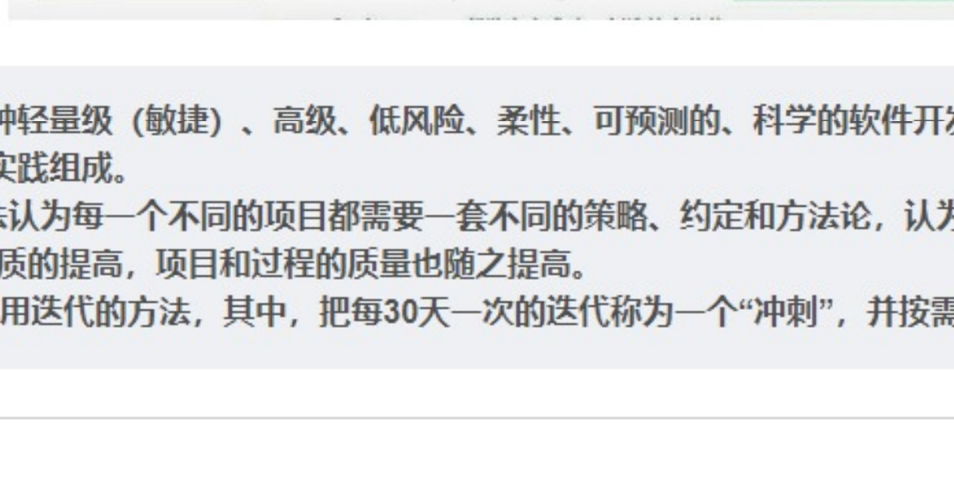
1.4.2 螺旋模型

> 螺旋模型将瀑布模型和原型模型结合起来，加入了两种模型均忽略的风险分析，弥补了这两种模型的不足。
> **螺旋模型强调风险分析，使得开发人员和用户对每个演化层出现的风险有所了解，从而做出应有的反应。同时，该模型特别适合用于庞大、复杂并且具有高风险的系统。（也适用于用户需求不清、需求经常变化的情况）**

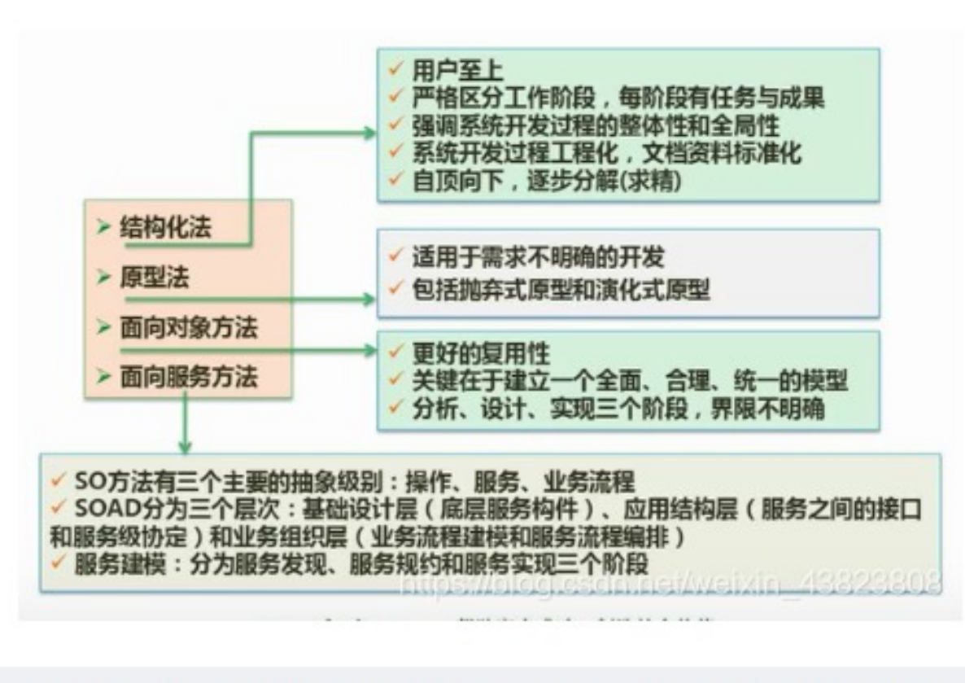
1.5 喷泉模型

> **喷泉模型是一种以用户需求为动力、以对象作为驱动的模型，适合于面向对象的开发方法。喷泉模型使得开发过程中具有迭代性和无间隙性（在开发活动之间不存在明显的边界，允许开发活动交叉，迭代地进行）。**

1.6 构建组装模型

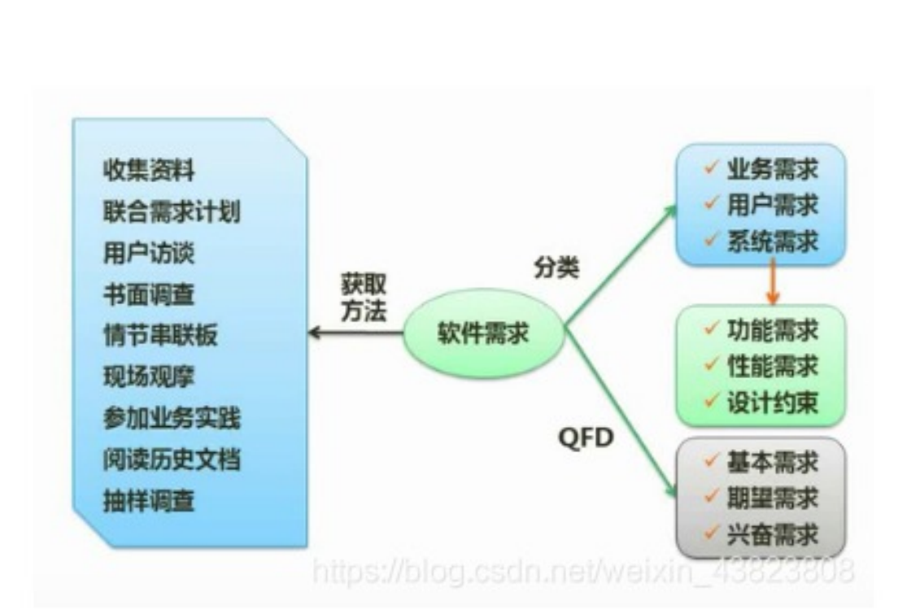


1.7 敏捷开发方法



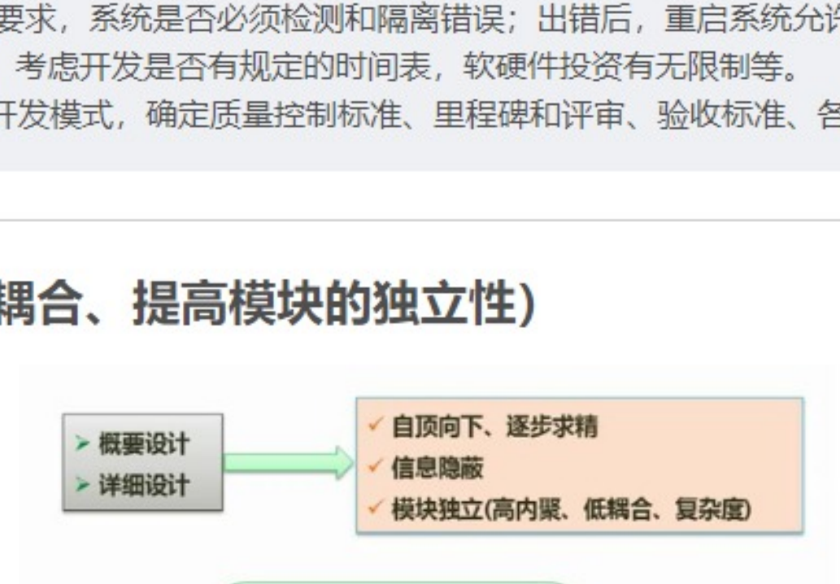
> **①极限编程（XP）：XP是一种轻量级（敏捷）、高级、低风险、柔性、可预测的、科学的软件开发方式。主要就是由上图中的4大价值观、5大原则、12个最佳实践组成。**
> **②水晶法（Crystal）：水晶法认为每一个不同的项目都需要一套不同的策略、约定和方法论，认为人对软件质量有重要的影响，因此随着项目质量和开发人员素质的提高，项目和过程的质量也随之提高。**
> **③并列争球法（Scrum）：使用迭代的方法，其中，把每30天一次的迭代称为一个"冲刺"，并按需求的优先级别来实现产品。**

2.软件开发方法



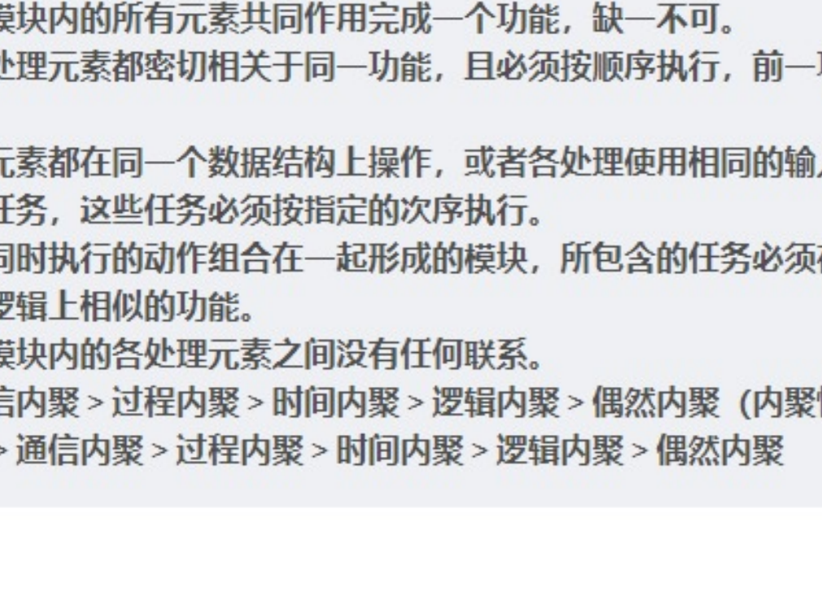
> **①结构化方法：由结构化分析、结构化设计、结构化程序设计构成，它是一种面向数据流的开发方法。总的指导思想是自顶向下、逐层分解，基本原则是功能的分解与抽象。特别适合于数据处理领域的问题，但是不适合解决大规模、特别复杂的项目，且难以适应需求的变化。**
> **②Jackson方法：这是一种面向数据结构的开发方法。以数据结构为驱动，适合于小规模的项目。当输入数据结构与输出数据结构之间没有对应关系时，难以应用此方法。**
> **③原型化方法：比较适合于用户需求不清、业务理论不确定、需求经常变化的情况。当系统规模不是很大也不太复杂时，采用该方法是比较好的。**
> **④面向对象开发方法：包括面向对象分析、面向对象设计和面向对象实现，采用统一建模语言（UML）。**

3.软件需求分析

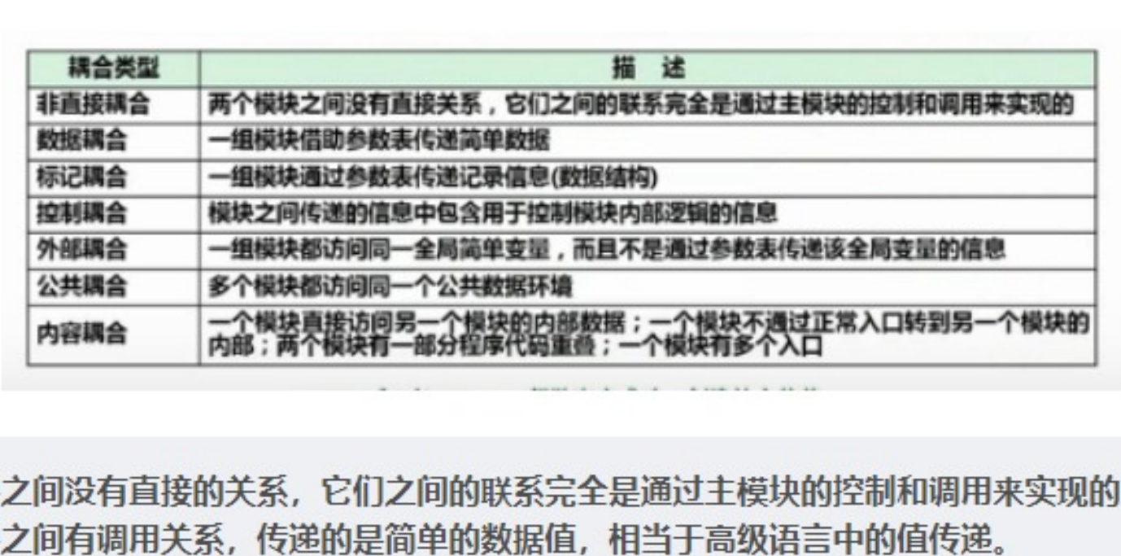


> ①功能需求：考虑系统要做什么，在何时做，在何时如何修改或升级。
> ②性能需求：考虑软件开发的技术性目标。例如：存储容量限制、执行速度、响应时间及吞吐量。
> ③用户或人的因素：考虑用户的类型。
> ④环境需求：考虑未来软件应用的环境，包括硬件和软件。对硬件设备的需求包括：机型、外设、接口、地点、分布、湿度、磁场干扰等。对软件设备的需求包括：操作系统、数据库、网络等。
> ⑤界面需求：考虑来自其他系统的输入，到其他系统的输出，对数据格式的特殊规定，对数据存储介质的规定。
> ⑥文档需求：考虑需要哪些文档，文档针对哪些读者。
> ⑦数据需求：考虑输入输出数据的格式，接收、发送数据的频率，数据的准确性和精度，数据流量，数据需保持的时间。
> ⑧资源使用需求：考虑软件运行时所需要的数据、其他软件、内存空间等资源；软件开发、维护所需的人力、支撑软件、开发设备等。
> ⑨安全保密需求：考虑是否需要对访问系统或系统信息加以控制，隔离用户数据的方法，用户程序如何与其他程序和操作系统隔离以及系统备份要求等。
> ⑩可靠性需求：考虑系统的可靠性要求，系统是否必须检测和隔离错误；出错后，重启系统允许的时间等。
> (11)软件成本消耗与开发进度需求：考虑开发是否有规定的时间表，软硬件投资有无限制等。
> (12)其他非功能需求：如采用某种开发模式，确定质量控制标准、里程碑和评审、验收标准、各种质量要求的优先级等。

4.软件设计（高内聚、低耦合、提高模块的独立性）



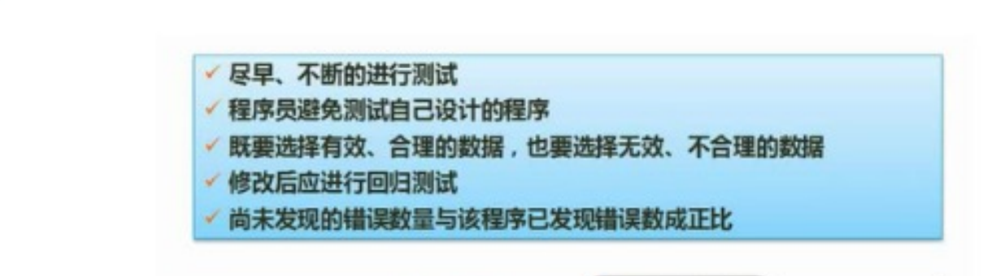
4.1 内聚



内聚类型	描　述
功能内聚	完成一个单一功能，各个部分协同工作，缺一不可
顺序内聚	处理元素相关，而且必须顺序执行
通信内聚	所有处理元素集中在一个数据结构的区域上
过程内聚	处理元素相关，而且必须按特定的次序执行
瞬时内聚(时间内聚)	所包含的任务必须在同一时间间隔内执行
逻辑内聚	完成逻辑上相关的一组任务
偶然内聚(巧合内聚)	完成一组没有关系或松散关系的任务

> ①功能内聚：这是最强的内聚，指模块内的所有元素共同作用完成一个功能，缺一不可。
> ②顺序内聚：指一个模块中的各个处理元素都密切相关于同一功能，且必须按顺序执行，前一功能元素的输出就是下一功能元素的输入。
> ③通信内聚：指模块内的所有处理元素都在同一个数据结构上操作，或者各处理使用相同的输入数据或者产生相同的输出数据。
> ④过程内聚：指一个模块完成多个任务，这些任务必须按指定的次序执行。
> ⑤瞬时内聚（时间内聚）：把需要同时执行的动作组合在一起形成的模块，所包含的任务必须在同一时间间隔内执行。
> ⑥逻辑内聚：指模块内执行若干个逻辑上相似的功能。
> ⑦偶然内聚（巧合内聚）：指一个模块内的各处理元素之间没有任何联系。
> 内聚性：功能内聚 > 顺序内聚 > 通信内聚 > 过程内聚 > 时间内聚 > 逻辑内聚 > 偶然内聚（内聚性越高，模块独立性越强）
> 模块独立性：功能内聚 > 顺序内聚 > 通信内聚 > 过程内聚 > 时间内聚 > 逻辑内聚 > 偶然内聚

4.2 耦合

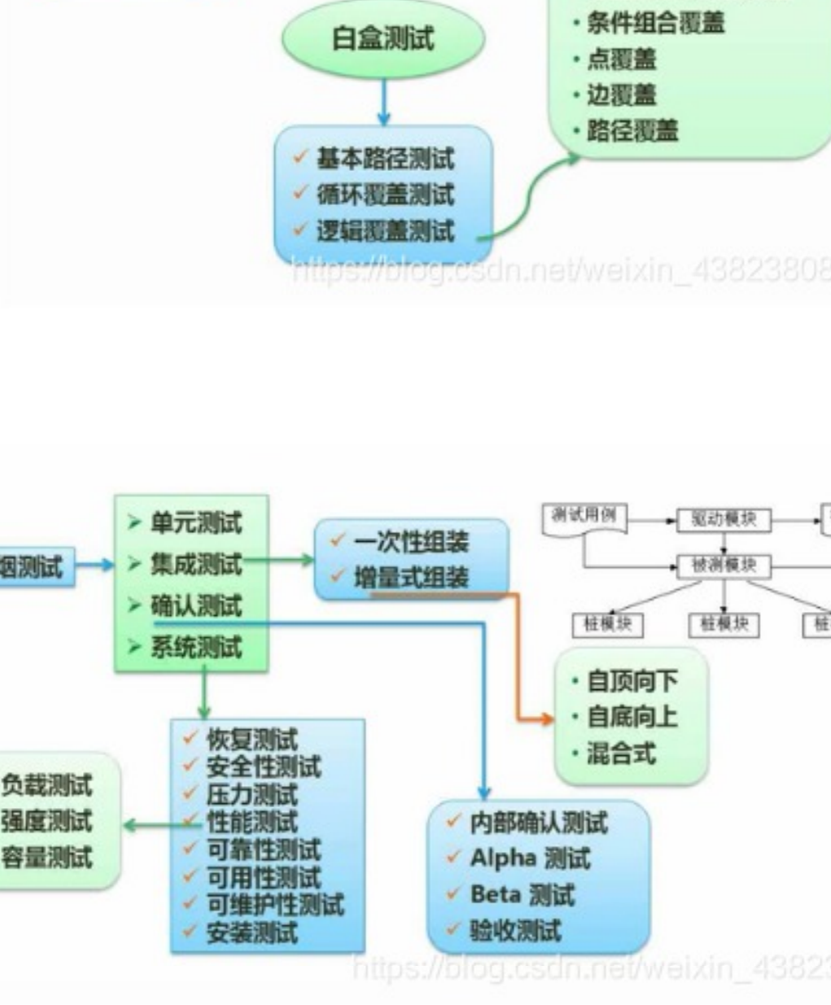


耦合类型	描　述
非直接耦合	两个模块之间没有直接关系，它们之间的联系完全是通过主模块的控制和调用来实现的
数据耦合	一组模块借助参数表传递简单数据
标记耦合	一组模块通过参数表传递记录信息(数据结构)
控制耦合	模块之间传递的信息中包含用于控制模块内部逻辑的信息
外部耦合	一组模块都访问同一全局简单变量，而且不是通过参数表传递该全局变量的信息
公共耦合	多个模块都访问同一个公共数据环境
内容耦合	一个模块直接访问另一个模块的内部数据；一个模块不通过正常入口转到另一个模块的内部；两个模块有一部分程序代码重叠；一个模块有多个入口

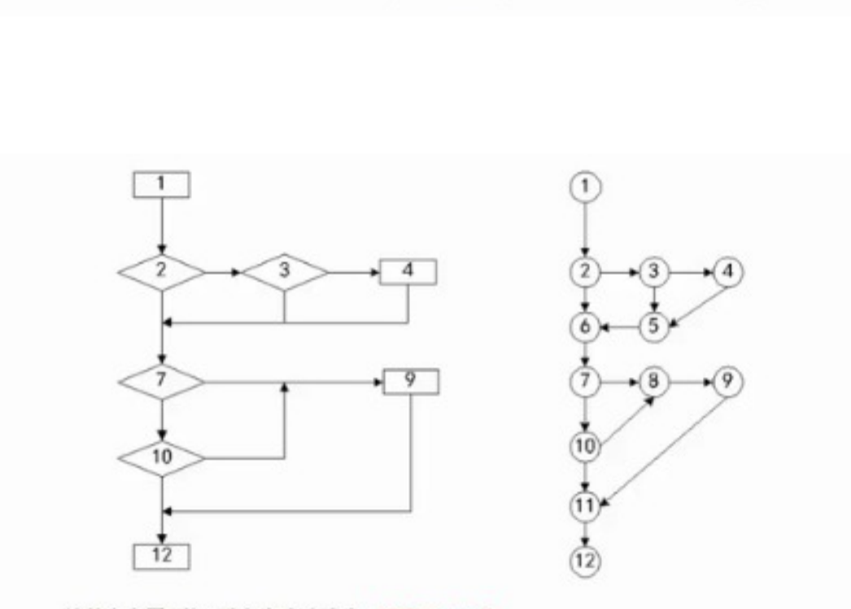
> ①无直接耦合：两个模块之间没有直接的关系，它们之间的联系完全是通过主模块的控制和调用来实现的。
> ②数据耦合：指两个模块之间有调用关系，传递的是简单的数据值，相当于高级语言中的值传递。
> ③标记耦合：指两个模块之间传递的是数据结构。
> ④控制耦合：指一个模块调用另一个模块时，传递的是控制变量，被调用模块通过该控制变量的值有选择地执行模块内的某一功能。
> ⑤外部耦合：模块间通过软件之外的环境联结（如I/O将模块耦合到特定的设备、格式、通信协议上）。
> ⑥公共耦合：指多个模块都访问同一个公共数据环境。
> ⑦内容耦合：当一个模块直接使用另一个模块的内部数据，或通过非正常入口转入另一个模块内部的耦合。
> 耦合性：无直接耦合 < 数据耦合 < 标记耦合 < 控制耦合 < 外部耦合 < 公共耦合 < 内容耦合（耦合性越低，模块独立性越强）
> 模块独立性：无直接耦合 > 数据耦合 > 标记耦合 > 控制耦合 > 外部耦合 > 公共耦合 > 内容耦合

5.软件测试

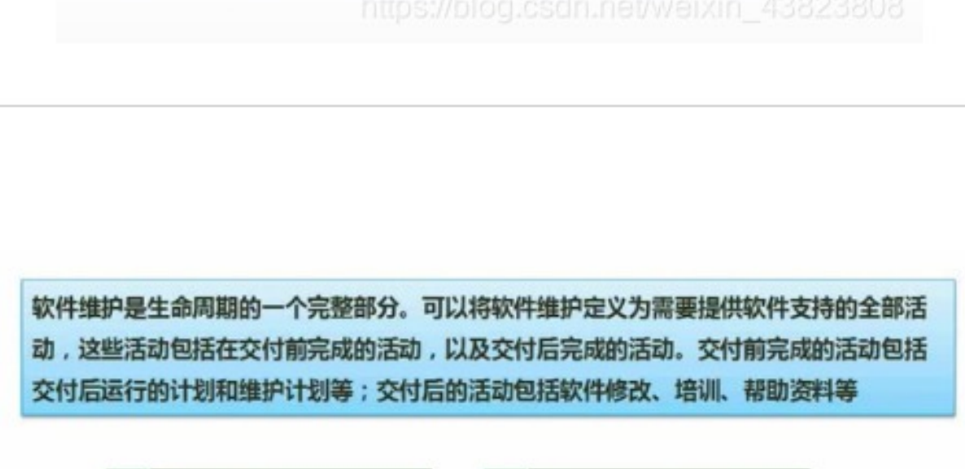
5.1 测试原则与类型



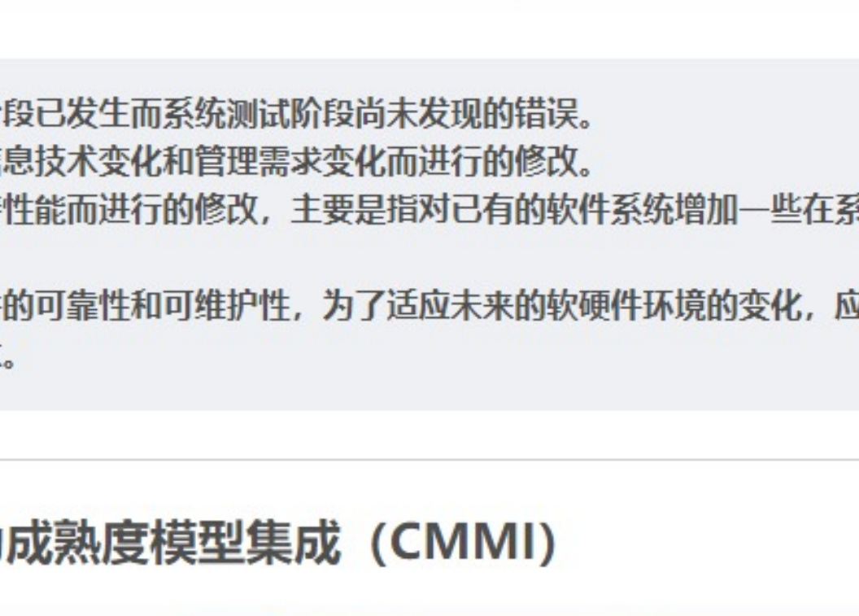
5.2 测试用例设计



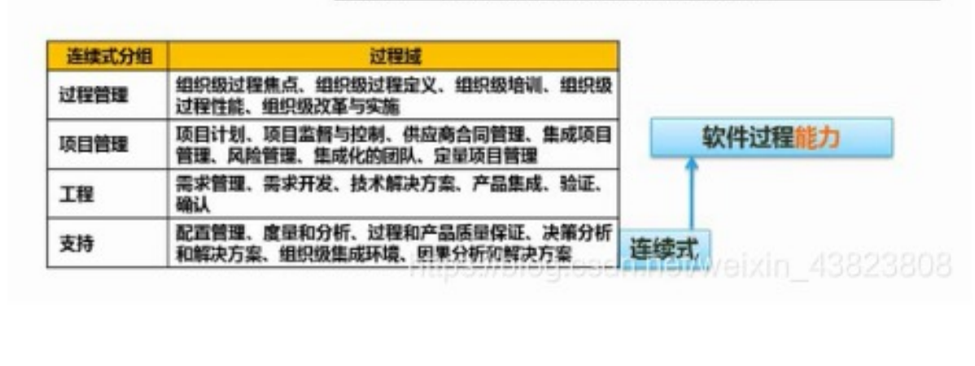
5.3 测试阶段



5.4 McCabe复杂度



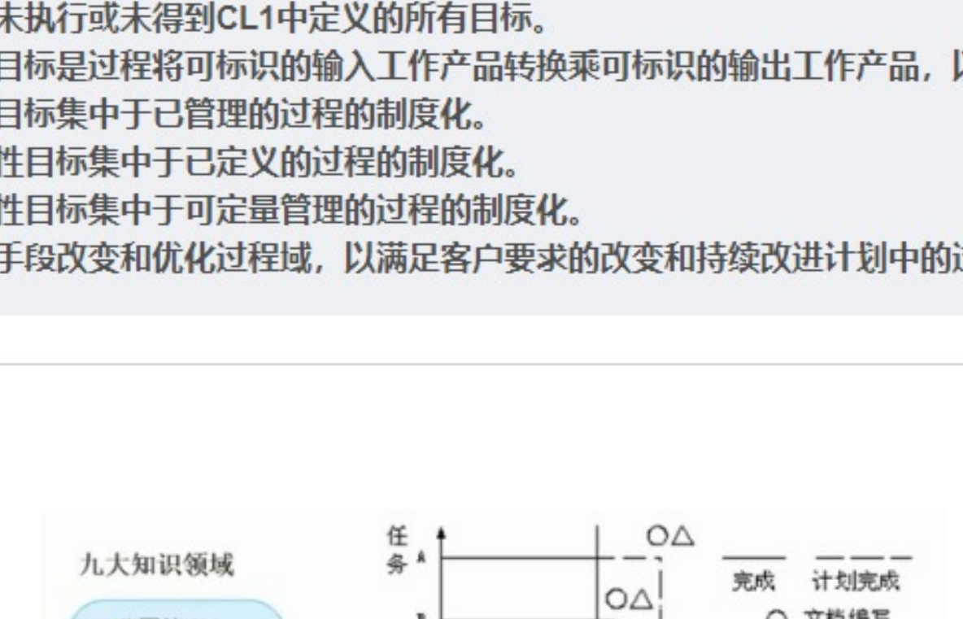
6.软件维护



软件维护是生命周期的一个完整部分。可以将软件维护定义为需要提供软件支持的全部活动，这些活动包括在交付前完成的活动，以及交付后完成的活动。交付前完成的活动包括交付后运行的计划和维护计划等；交付后的活动包括软件修改、培训、帮助资料等

> ①改正性维护：改正在系统开发阶段已发生而系统测试阶段尚未发现的错误。
> ②适应性维护：使应用软件适应信息技术变化和管理需求变化而进行的修改。
> ③完善性维护：为扩充功能和改善性能而进行的修改，主要是指对已有的软件系统增加一些在系统分析和设计阶段中没有规定的功能与性能特征。
> ④预防性维护：为了改进应用软件的可靠性和可维护性，为了适应未来的软硬件环境的变化，应主动增加预防性的新的功能，以使应用系统适应各类变化而不被淘汰。

7.软件过程改进——能力成熟度模型集成（CMMI）



成熟度等级	过程域
已管理级	需求管理、项目计划、配置管理、项目监督与控制、供应商合同管理、度量和分析、过程和产品质量保证
已定义级	需求开发、技术解决方案、产品集成、验证、确认、组织级过程焦点、组织级过程定义、组织级培训、集成项目管理、风险管理、集成化的团队、决策分析和解决方案、组织级集成环境
定量管理级	组织级过程性能、定量项目管理
优化级	组织级改革与实施、因果分析和解决方案

连续式分组	过程域
过程管理	组织级过程焦点、组织级过程定义、组织级培训、组织级过程性能、组织级改革与实施
项目管理	项目计划、项目监督与控制、供应商合同管理、集成项目管理、风险管理、集成化的团队、定量项目管理
工程	需求管理、需求开发、技术解决方案、产品集成、验证、确认
支持	配置管理、度量和分析、过程和产品质量保证、决策分析和解决方案、组织级集成环境、因果分析和解决方案

7.1 阶段式模型

> ①初始的：过程不可预测且缺乏控制。
> ②已管理的：过程为项目服务。
> ③已定义的：过程为组织服务。
> ④定量管理的：过程已度量和控制。
> ⑤优化的：集中于过程改进。

7.2 连续式模型

> ①CL0（未完成的）：过程域未执行或未得到CL1中定义的所有目标。
> ②CL1（已执行的）：其共性目标是过程将可标识的输入工作产品转换乘可标识的输出工作产品，以实现支持过程域的特性目标。
> ③CL2（已管理的）：其共性目标集中于已管理的过程的制度化。
> ④CL3（已定义级的）：其共性目标集中于已定义的过程的制度化。
> ⑤CL4（定量管理的）：其共性目标集中于可定量管理的过程的制度化。
> ⑥CL5（优化的）：使用量化手段改变和优化过程域，以满足客户要求的改变和持续改进计划中的过程域的功效。

8.软件项目管理

8.1 时间管理（Gantt图、PERT图）

> **①Gantt图：能够清晰地描述每个任务从何时开始、到何时结束，任务的进展情况以及各个任务之间的并行性。但是它不能清晰地反映出各任务之间的依赖关系，难以确定整个项目的关键所在，也不能反映计划中有潜力的部分。**
> **②PERT图：不仅给出了每个任务的开始时间、结束时间和完成该任务所需的时间，还给出了人物之间的依赖关系，即哪些任务完成后才能开始另外一些任务，以及如期完成整个工程的关键路径。但是它不能反映任务之间的并行关系。**

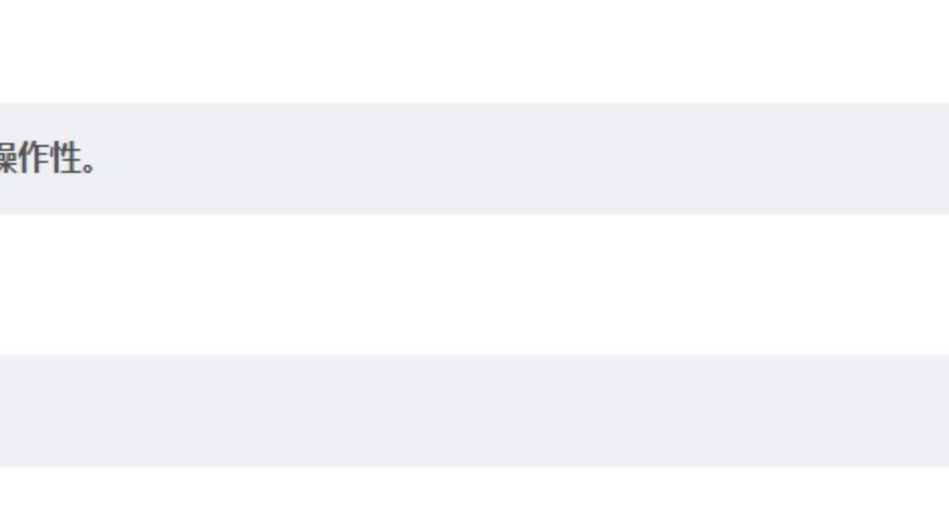
进度安排的常用图形描述方法有Gantt图和PERT图。Gantt图不能清晰地描述______；PERT图可以给出哪些任务完成后才能开始另一些任务。下图所示的PERT图中，事件6的最晚开始时间是______。

（1）A.每个任务从何时开始　B.每个任务到何时结束
C.每个任务的进展情况　D.各任务之间的依赖关系
（2）A.0　B.3　C.10　D.11

时间\事件	1	2	3	4	5	6	7	8	9
最早开始时间	0	2	2	0	4	**4**	9	9	15
最晚开始时间	0	2	9	6	4	**10**	9	11	15

> 由上图分析，以及Gantt图和PERT图的相关概念，可知，第一空选D，第二空选C。

8.2 风险管理



风险是指"损失或伤害的可能性"。

风险曝光度（Risk Exposure）：计算方法是风险出现的概率乘以风险可能造成的损失。
假设正在开发的软件项目可能存在一个未被发现的错误，而这个错误出现的概率是0.5%，给公司造成的损失将是1000000元，那么这个错误的风险曝光度就应为1000000×0.5%=5000元。

9.软件质量保证

9.1 功能性

> ①适合性。②准确性。③互用性。④依从性。⑤安全性。

9.2 可靠性

> ①成熟性。②容错性。③易恢复性。

9.3 易使用性

> ①易理解性。②易学性。③易操作性。

9.4 效率

> ①时间特性。②资源特性。

9.5 可维护性

> ①易分析性。②易改变性。③易测试性。④稳定性。

9.6 可移植性

> ①适应性。②一致性。③易安装性。④易替换性。