**第一章**

**1、 软件的概念**

**软件是计算机系统中与硬件相互依存的另一部分，它是包括程序、数据及其相关文档的完整集合。**

**注：程序并不是软件，程序只是软件的组成部分**

**2、软件的特点**

**（1）软件是一种逻辑实体。**

**（2）软件的开发，是人的智力的高度发挥，而不是传统意义上的硬件制造。**

**（3）软件维护与硬件的维修有着本质的差别。**

**（4）软件的开发和运行常常受到计算机系统的限制，对计算机系统有着不同程度的依赖性。**

**（5）软件的开发至今尚未完全摆脱手工艺的开发方式，使软件的开发效率受到很大限制。**

**（6）软件的开发是一个复杂的过程。**

**（7）软件的成本非常高昂。**

**3、软件的分类**

**1). 基于软件功能的划分: 系统软件 应用软件 支撑软件**

**2). 基于软件工作方式的划分: 实时处理软件 分时软件 交互式软件 批处理软件**

**4、软件危机 （简答）**

**20世纪60年代末70年代初，西方工业发达国家经历了一场“软件危机”。这场软件危机表现在：一方面软件十分复杂，价格昂贵，供需差日益增大，另一方面软件开发时又常常受挫，质量差，指定的进度表和完成日期很少能按时实现，研制过程很难管理，即软件的研制往往失去控制。我们称软件开发和维护过程中所中遇到的这一系列严重问题为软件危机。**

**软件危机包含下述两方面的问题：如何开发软件，以满足对软件日益增长的需求；如何维护数量不断膨胀的已有软件。**

**5、软件工程的定义**

**1983年美国《IEEE软件工程标准术语》对软件工程下的定义为：软件工程是开发、运行、维护和修复软件的系统方法，其中“软件”的定义为：计算机程序、方法、规则、相关的文档资料以及在计事机上运行时所必需的数据。**

**6、软件工程方法学**

**通常把在软件生命周期全过程中使用的一整套技术的集合，称为软件工程方法学。软件工程方法学包括三个要素：方法、工具和过程。**

**7、软件生存期**

**一般说来，软件生命期由软件定义、软件开发和软件维护三个时期组成，每个时期又可进一步划分成若干个阶段。**

**1)．软件定义时期 （1）问题定义（2）可行性研究**

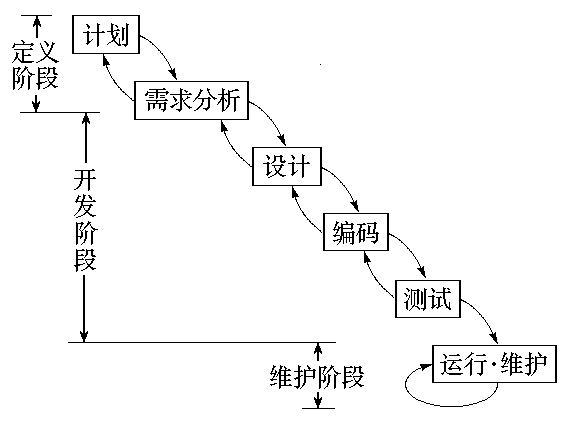
**2)．软件开发时期（1）需求分析（2）总体设计（3）详细设计（4）编码（5）测试**

**3)．软件运行时期 （1）综合测试 （2）运行维护**

**8、软件开发模型**

**1). 瀑布模型：**

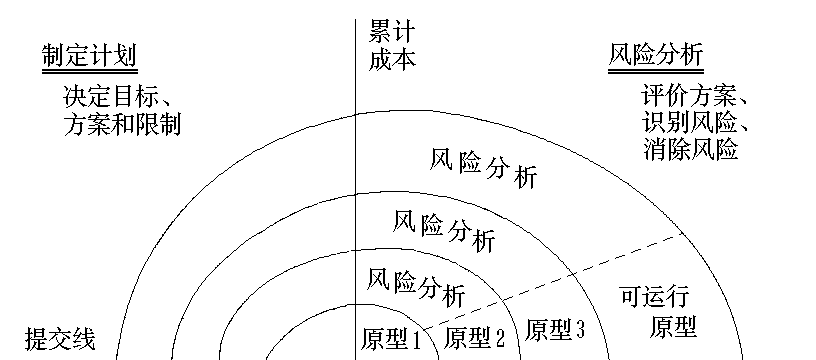
**将软件生存周期的各项活动规定为依照固定顺序连接的若干阶段工作，形如瀑布流水，最终得到软件产品。（缺少风险分析，缺少灵活性）**

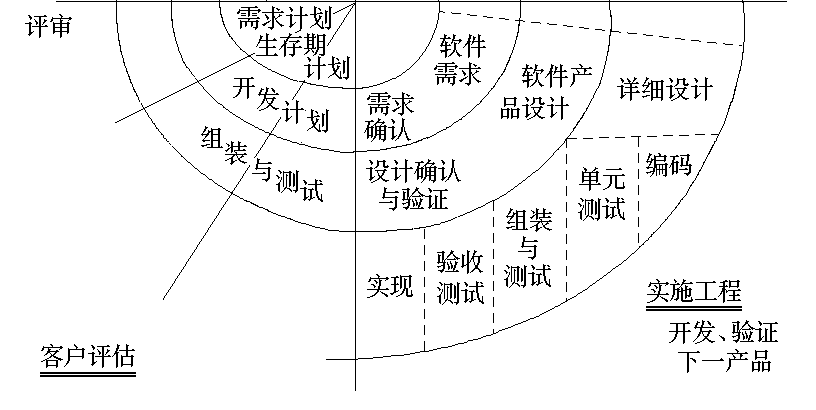
****

**2). 螺旋模型：**

**为了克服瀑布模型的不足，螺旋模型于1988年提出。该模型中加入了风险分析，通常用来指导大型软件项目的开发。**

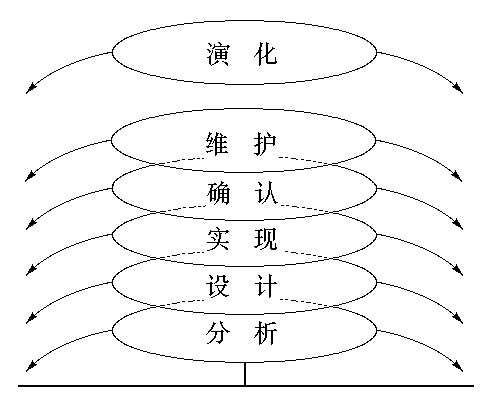
* **螺旋模型沿着螺线旋转，在四个象限上分别表达四个方面的活动，即：**
* **制定计划──确定软件目标，选定实施方案，弄清项目开发的限制**
* **风险分析──分析所选方案，考虑如何识别和消除风险**
* **实施工程──实施软件开发**
* **客户评估──评价开发工作，提出修正建议**

****

****

**3). 喷泉模型：**

**体现了软件创建所固有的迭代和无间隙的特征。**

****

**第二章**

**1. 优秀需求具有的特性**

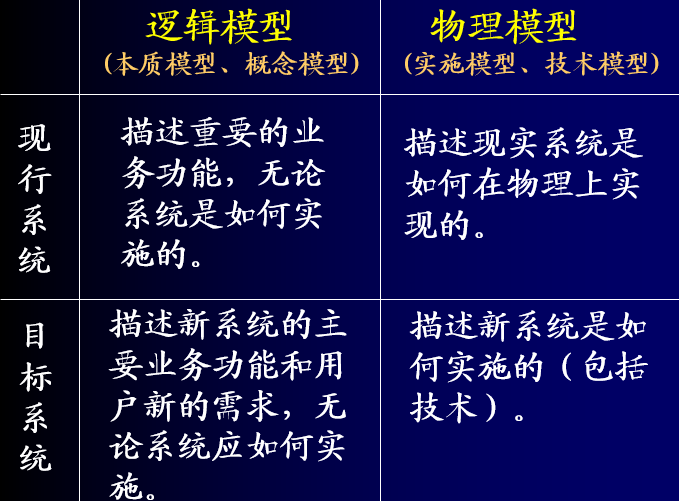
1). 完整性 2. 正确性 3. 可行性 4. 必要性 5. 划分优先级 6. 无二义性 7. 可验证性

**2. 需求分析的任务**

需求分析的主要任务就是要通过软件开发人员与用户的交流和讨论，准确地获取用户对系统的具体要求。

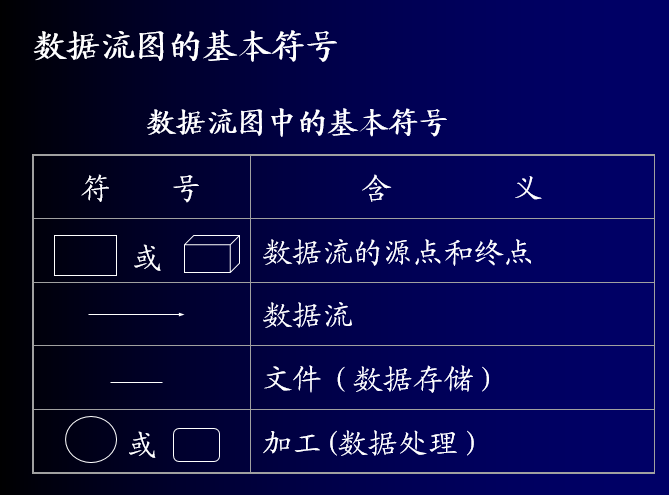
**3. 权利与义务 P17 ~ 20**

**4. 逻辑模型与物理模型 P39**

****

**5. 数据流图 P40 ~ 41**

在需求分析阶段，数据流(也称信息流)是系统分析的基础。**数据流图(DFD，Data Flow Diagram)**是描述软件系统中数据处理过程的一种有力的图形工具。数据流图从**数据传递**和**加工**的角度出发，刻画数据流从输入到输出的**移动和变换过程**。



**6. 数据字典(DD)**

**DD是对所有与系统相关的数据元素的一个有组织的列表,以及精确的、严格的定义，使得用户和系统分析员对于输入、输出、存储成分和中间计算有共同的理解。**

**数据字典是关于数据流图中各种成分详细定义的信息集合，可将其按照说明对象的类型划分为四类条目:**

* + - * + **数据流条目**
        + **数据项条目**
        + **数据文件条目**
        + **数据加工条目**

**7. 加工逻辑的描述 P80**

为了能够直观、明确地表达加工逻辑，经常采用**三种描述方法：**

**结构化语言**

**判定树**

**判定表**

**8. 传统软件工程中的需求分析方法主要包括**： P92

* + - * + **原型化分析方法**
        + **结构化分析方法**

**原型化方法主要用于解决**：建立原型的主要原因是为了解决在产品开发的早期阶段需求不确定的问题用户经理及其他风险承担者在发现在确定和开发产品时原型可以使他们的想象更具体化。

**第三章**

**1. 从工程管理角度来看，软件设计分两步完成：概要设计和详细设计。**

**概要设计的目标是建立软件的体系结构，软件体系结构的三要素是程序构件（模块）的层次结构、构件之间交互的方式，以及数据的结构。**

**2. 软件体系结构的三要素是程序构件（模块）的层次结构、构件之间交互的方式，以及数据的结构。**

**3. 模块具有3个基本属性**

**(1)功能**

**(2)逻辑:描述模块内部怎么做**

**(3)状态：即该模块使用时的环境和条件。**

**4. 总体设计通常由系统设计和结构设计两个阶段组成。**

**1、HIPO图（应用在总体设计阶段）**

**IBM公司发明的HIPO图：**

**层次图 + 输入 / 处理 /输出图**

**(H图) (IPO图)**

**5. 结构图能够描述出软件系统的模块层次结构，清楚地反映出程序中各模块之间的调用关系和联系。**

**6. 模块化可以降低解决问题的复杂度，从而降低软件开发的工作量**

**思考：是不是模块划分得越多越好呢？**

* **增加程序中的模块数可以降低开发每个模块的工作量**
* **同时却增加了设计模块接口的工作量**

**7. 信息隐蔽**

**信息隐蔽是指:模块内部的数据与过程，应该对不需要了解这些数据与过程的模块隐藏起来。只有那些为了完成软件的总体功能而必需在模块间交换的信息，才允许在模块间进行传递。模块之间的信息传递只能通过合法的调用接口来实现。信息隐蔽对提高软件的可读性和可维护性都是非常重要的。软件结构程序设计阶段，需要遵循信息隐蔽原则**

**8. 耦合性和内聚性是判断模块独立性的依据**

**1) 耦合性**

**耦合性是对一个软件结构内部不同模块间联系紧密程度的度量指标。模块间的联系越紧密，耦合性就越高，模块的独立性也就越低。**

**2) 内聚性**

**对一个模块内部各个组成元素之间相互结合的紧密程度的度量指标。**

**耦合内聚分类及其排序**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 偶然内聚 | 逻辑内聚 | 时间内聚 | 过程内聚 | 通讯内聚 | 顺序内聚 | 功能内聚 |

低 内聚 高

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 非直接耦合 | 数据耦合 | 特征耦合 | 控制耦合 | 外部耦合 | 公共耦合 | 内容耦合 |

低 耦合性 高

**常见耦合概念（记顺序）**

**(1) 数据耦合：**

**两个模块之间仅通过模块参数交换信息，且交换的信息全部为简单数据。**

**数据耦合的耦合性最低。通常软件中都包含有数据耦合。**

**(2) 公共耦合：**

**两个或多个模块通过引用公共数据相互联系**

**程序中定义的全局变量，并在多个模块中对全局变量进行了引用。**

**公共耦合的复杂度随着耦合的模块个数的增加而显著增加。在程序设计中，若两个模块间需要交换的数据较多，仅通过参数传递难以实现时，可以考虑采用公共耦合完成，但一定注意尽量降低公共耦合的程度。**

**(3) 控制耦合**

* **模块之间交换的信息中包含有控制信息(尽管有时控制信息是以数据的形式出现的)，则称这种耦合为控制耦合**
* **控制耦合是中等程度的耦合，它会增加程序的复杂性**

**(4) 内容耦合：**

* **一个模块对另一模块中的内容(包括数据和程序段)进行了直接的引用甚至修改，或通过非正常入口进入到另一模块内部，或一个模块具有多个入口，或两个模块共享一部分代码，这种耦合为内容耦合。**
* **内容耦合是所有耦合关系中程度最高的，会使因模块间的联系过于紧密而对后期的开发和维护工作带来很大的麻烦。**

**常见内聚概念（记概念）**

**(1) 偶然内聚：**

**一个模块由多个完成不同任务的语句段组成，各语句段之间的联系十分松散或根本没有任何联系，则称此模块的内聚为偶然内聚。**

**偶然内聚的模块由于组成部分之间没有实质的联系，因此难于理解和修改，会给软件开发带来很大的困扰。偶然内聚是内聚程度最低的一种，在软件设计时应尽量避免。**

**(2) 逻辑内聚**

**若一个模块可实现多个逻辑上相同或相似的一类功能，则称该模块的内聚为逻辑内聚。**

**逻辑内聚模块的组成部分之间有一定的关系，但不同功能混在一起并公用模块中的部分代码，给修改带来了一定的麻烦**

**为了在调用模块时能选择执行其中的某个功能，需要传递相应的控制参数，造成模块间的控制耦合，降低模块的独立性。**

**(3) 时间内聚**

**一个模块包含了需要在同一时间段中执行的多个任务，则称该模块的内聚为时间内聚。**

**(4) 过程内聚：**

**一个模块中的各个部分相关，并且必须按特定的次序执行，则称该模块的内聚为过程内聚。**

**在结构化程序中，通常采用程序流程图作为设计软件和确定模块划分的工具，因此，这样得到的模块往往具有过程内聚的特性**

**(5) 通信内聚:**

**一个模块中的各个部分使用同一个输入数据或产生同一个输出数据，则称该模块的内聚为通信内聚。**

**(6) 顺序内聚:**

**一个模块中的各个部分都与同一个功能密切相关，并且必须按照先后顺序执行(通常前一个部分的输出数据就是后一个部分的输入数据)，则内聚为顺序内聚。**

**(7) 功能内聚：**

**若一个模块中各个组成部分构成一个整体并共同完成一个单一的功能，该内聚为功能内聚。**

**设计目标：**

**高内聚、低耦合**

**（简答）启发式规则（模块设计准则） P66 ~ 80**

**1、改进软件结构，提高模块独立性。（降低模块的耦合性，提高模块的内聚性）**

**2、保持适中的模块规模。**

**3、模块应具有高扇入和适当的扇出；软件结构中的深度和宽度不宜过大。**

**4、模块的作用域应处于其控制域范围之内。**

**5、尽量降低模块的接口复杂度。**

**6、设计单入口、单出口的模块。**

**7、模块功能应该可以预测。**

**9. 系统结构特征可归纳为两种典型形式**

* **变换型结构**
* **事务型结构**

**10.数据流图可分为两种类型：**

* **变换型数据流**
* **事务型数据流**

**11. 软件从一个计算机系统或环境转换到另一个计算机系统或环境的容易程度成为可移植性**

**软件设计中用抽象和分解的目的是：降低复杂度**

**第四章**

**1. 详细设计的任务**

* 1. **确定每个模块的具体算法。**
  2. **确定每个模块的内部数据结构及数据库的物理结构。**
  3. **确定模块接口的具体细节。**
  4. **为每个模块设计一组测试用例。**
  5. **编写文档，参加复审。**

**第七章**

**EG1: 列出UML动态建模机制：**

**1. 消息2. 状态图3. 顺序图4. 协作图5. 活动图**

**消息：在面向对象技术中，对象间的交互是通过对象间消息的传递来完成的。**

**UML定义的消息类型有三种:**

**（1）简单消息(Simple Message)**

**（2）同步消息(Synchronous Message)**

**（3）异步消息(Asynchronous Message)**

**状态图：(State Diagram)用来描述一个特定对象的所有可能状态及其引起状态转移的事件。一个状态图包括一系列的状态以及状态之间的转移。**

**(1) 状态　所有对象都具有状态，状态是对象执行了一系列活动的结果。**

**(2) 转移　状态图中状态之间带箭头的连线被称为转移。**

**顺序图：(Sequence Diagram)用来描述对象之间动态的交互关系，着重体现对象间消息传递的时间顺序。**

**合作图：(Collaboration Diagram)用于描述相互合作的对象间的交互关系和链接关系。**

**活动图：(Activity Diagram) 既可用来描述操作(类的方法)的行为，也可以描述用例和对象内部的工作过程。**

**EG2: 原型化方法主要用于解决什么问题：**

**建立原型的主要原因是为了解决在产品开发的早期阶段需求不确定的问题，用户经理和其他非技术项目风险承担者发现在确定和开发产品时，原型可以使他们的想象更具体化。**

**可能会考：**

**快速原型模型可以有效的适应用户动态需求变化 √**

**在进行总体设计时，应加强模块间的联系 ×**

**需求分析的基本任务是准确定义未来系统目标，确定用户的目标，系统必须做什么 √**

**系统设计包括总体设计与详细设计两个阶段 √**

**第三章P83 变换型结构两部分组成：传入路径、传出路径 ×**

**第三章P83 事务型结构由至少一条接受路径一个事务中心和若干条加工路径组成√**

**结构化分析方法是一种严格预先定义需求的方法，在实施时强调的是分析对象的数据流 √**

**UML（统一建模语言）是软件开发中的一个重要工具，主要应用于基于数据的数据流的工具 ×**

**需求分析规定性能要求 √**

**用于保证软件质量的技术手段：测试 √**

**在白盒法技术测试用例的设计中，什么是最弱的覆盖标准：判定√**

**软件工程中，只根据程序的功能说明而不关注逻辑内部结构成为黑盒，白盒反之。**

**开发大型软件时产生困难的根本原因是： 大系统的复杂性**

**效率是一个性能要求，因此应该在以下那个阶段规定: 需求分析**

**在软件工程中，当前用于保证软件质量的技术手段: 测试**