BUPT
TSEG

软件工程 模型与方法
Models & Methods of SE

剩余章节

主要内容

- 软件维护
- 软件项目管理

软件维护

- 软件维护的定义
- 软件维护的分类
- 软件维护的活动
- 软件的逆向和再工程

软件维护的定义和分类

- 在软件已经交付使用之后，为了改正错误或满足新的需要而修改软件的过程，即在软件运行/维护阶段对软件产品所进行的一切改动。
  - 改正在系统运行过程中暴露出来的一些潜在程序错误或设计缺陷，称为改正性维护。
  - 为了适应在软件使用过程中数据环境发生变化或处理环境发生变化而进行的软件修改，称为适应性维护。
  - 为满足用户的其他要求，就需要修改软件并把这些要求纳入到软件之中，称为完善性维护。
  - 为了提高软件的可维护性、可靠性等而事先进行的软件改动，称为预防性维护。

影响软件维护工作量的因素

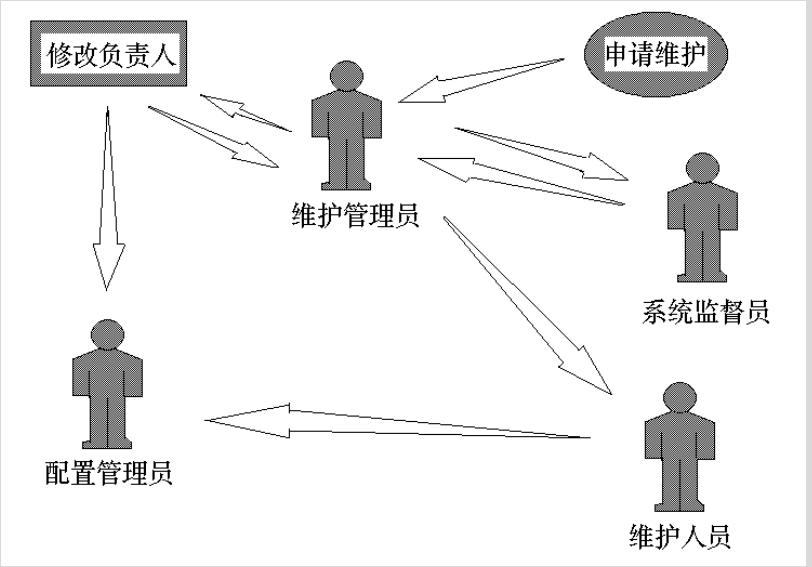
- 系统大小：系统越大，理解掌握起来越困难。系统越大，所执行功能越复杂。因而需要更多的维护工作量。
- 程序设计语言：使用强功能的程序设计语言可以控制程序的规模。语言的功能越强，生成程序的模块化和结构化程度越高，所需的指令数就越少，程序的可读性越好。
- 系统年龄：系统随着不断的修改，结构越来越乱；由于维护人员经常更换，程序又变得越来越难于理解；长期的维护过程中文档在许多地方与程序实现变得不一致。
- 其它：例如，应用的类型、数学模型、任务的难度、开关与标记、IF嵌套深度、索引或下标数等，对维护工作量都有影响。

软件维护中的典型问题

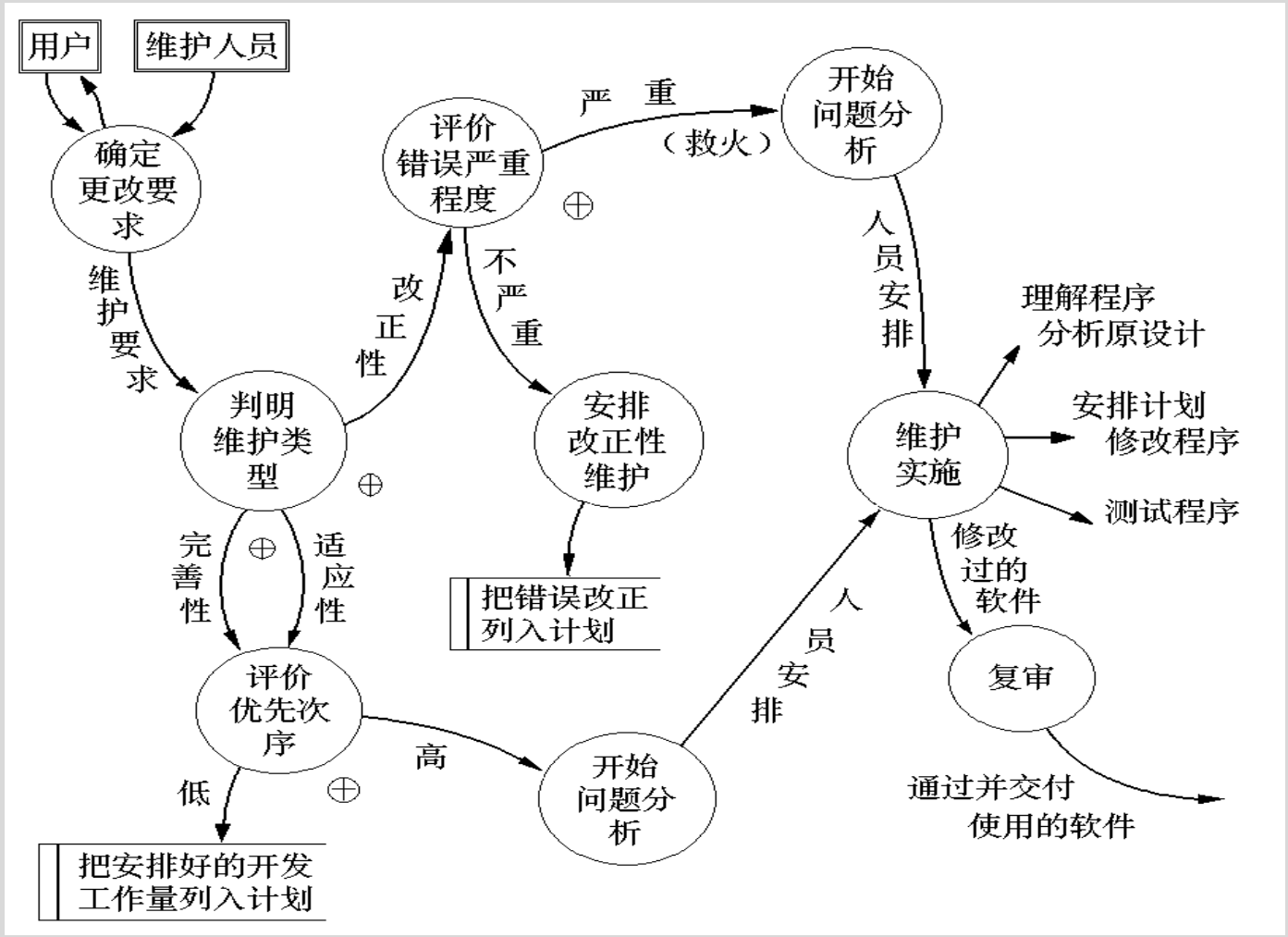
- 首先软件维护人员大多数情况下不是软件开发人员，为此他们会遇到以下问题：
  - 阅读和理解别人写的程序非常困难。
  - 需要维护的软件往往没有合格的文档，或者文档资料明显不足。
  - 不能指望开发人员仔细说明软件。
  - 决大多数软件在设计时没有考虑将来的修改，从而导致了软件的可维护性很差。
  - 软件维护不是一项吸引人的工作，由于以上原因经常导致维护出现困难，从而使软件维护人员产生厌烦和挫折感。

软件维护的活动

- 为了有效地进行软件维护，就必须：
  - 建立维护机构
  - 给出软件维护的工作管理流程
  - 为每一个维护申请规定标准的处理步骤



软件维护的管理工作流程



软件维护的标准化

- 修改请求：一般由用户、程序员或管理人员提出，是软件维护的开始；
- 分类与鉴别：根据修改请求，由维护机构来确认其维护类别，给一个编号，并输入数据库保存；
- 分析：先进行维护的可行性分析，然后进行详细分析；
- 设计：汇总全部信息开始更改，本阶段应更改设计的基线、更改测试计划、修订详细分析结果、核实维护需求；
- 实现：制定程序更改计划并进行软件更改。包括编码、单元测试、集成、风险分析、测试准备审查、更新文档；
- 系统测试：主要进行程序之间的接口测试，以确保加入了修改的软件满足原来的需求，回归测试是确保不要引入新的错误；
- 验收测试：最终的综合测试，由客户、用户和第三方共同进行；
- 交付：此阶段是将新的系统交给用户安装并运行。

软件的逆向工程和再工程

- 术语“逆向工程”来自硬件。成功的逆向工程应当通过考察产品的实际样品，导出该产品的一个或多个设计与制造的规格说明。
- 软件的逆向工程是分析程序，是设计恢复的过程，需要从已存在程序中抽取数据结构、体系结构和程序设计信息。

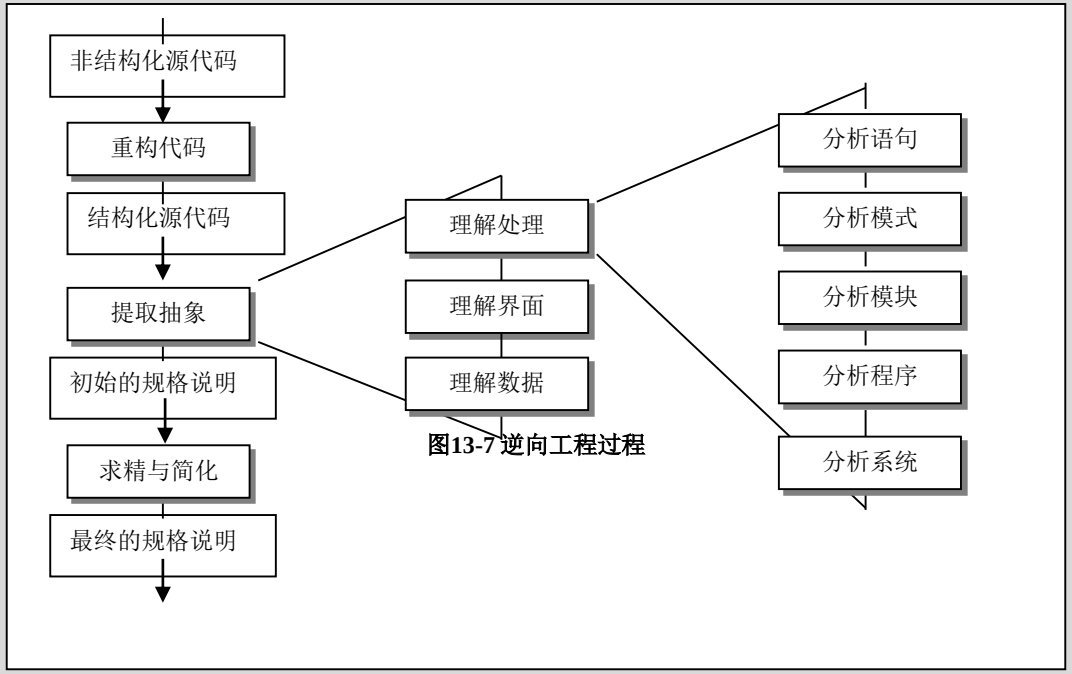
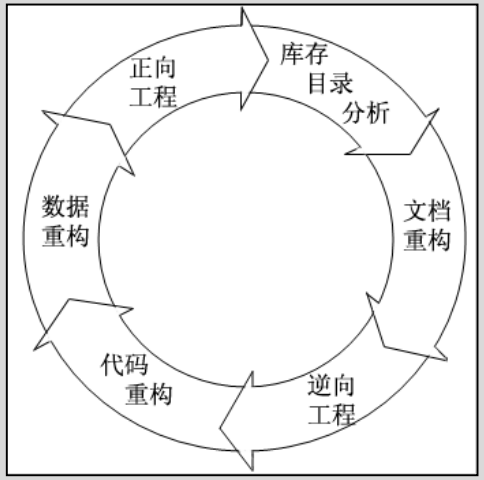


图13-7 逆向工程过程

软件的再工程

- 软件再工程是一类软件工程活动，是一个工程过程，它将逆向工程、重构和正向工程组合起来，将现存系统重新构造为新的形式。
- 再工程的基础是系统理解，包括对运行系统、源代码、设计、分析、文档等的全面理解。
- 在这些理解的基础上，执行重构生成一个设计，它产生与原来程序相同的功能，但具有比原来程序更高的质量。



该模型是一个循环模型，模型组成部分的每一个活动都有可能被重复，且对于任意一个特定的循环来说，过程可以在完成任意一个活动之后终止。在某些情况下这些活动以线性顺序发生，但也有的时候交错发生。

软件项目管理

- 项目和软件项目的定义
- 软件项目管理过程
- 软件项目度量
  - 软件规模度量
  - 软件项目估算
- 软件项目进度安排
- 软件项目的组织结构

项目和软件项目的定义

- 项目，是一项为了创造某一唯一的产品或服务的时限性工作。具有以下特征：
  - 需要由人来完成；
  - 受到有限资源的限制；
  - 需要计划、执行和控制。
- 软件项目是一种成果体现为软件产品的项目，其特有的特征表现为：
  - 软件产品是无形的；
  - 软件产品没有标准的软件过程；
  - 大型软件项目开发常常是“一次性的”。

项目的管理过程

- 项目管理就是为了满足甚至超越项目干系人员对项目的需求和期望的一些活动，并将理论知识、技能、工具和技巧应用到项目的活动中。
- 项目管理包括以下九个知识领域：
  1. 综合管理：将项目管理各种必要要素综合为整体的过程和活动，并在项目管理过程组范围内识别、定义、组合、统一并协调。
  2. 范围管理：界定为了确保成功地完成项目所需要做的工作，也是仅仅被要求做的工作。
  3. 时间管理：阐述确保项目按时完成所需的各项过程。
  4. 成本管理：阐述了确保项目按照规定预算完成需要进行的费用规划、估算、预算的各项过程。
  5. 质量管理：阐述了确保项目达到其既定质量要求所需实施的各项过程。
  6. 人力资源管理：阐述了组织和管理项目团队的各个过程。
  7. 沟通管理：阐述了为确保项目信息及时而恰当地提取、收集、传输、存储和最终处置而需要实施的一系列过程。
  8. 风险管理：阐述了与项目风险管理有关的过程。
  9. 采购管理：阐述了采购或取得产品、服务或成果，以及合同管理所需的各过程。

项目目标及其制约因素

- 项目目标就是在一定时间、预算内完成工作的范围，以使客户满意。
- 实现项目目标要受到四个因素的制约，它们是：
  - 项目范围是为使客户满意必须做的所有工作；
  - 项目成本就是完成项目所需要的费用，它必须在客户为这个项目提供的资金限额以内；
  - 项目进度是安排每项任务的起止时间以及所需的资源等，是为项目描绘的一个过程蓝图。
  - 客户满意度：是指完成的项目质量是否达到预期的效果。

软件项目管理过程

- 启动软件项目：这是软件项目管理的第一个过程，目的是确定软件项目的目标、范围。通常，软件人员和用户是在系统需求工程阶段确定项目的目标和范围的；
- 制定项目计划：项目计划是建立项目行动指南的基准，包括对软件项目的估算、风险分析、进度安排、人员的选择与配备等；
- 项目计划的执行：根据定义的计划由具体的人员实施的各项活动；
- 项目的控制：在项目的执行过程中所必须的监督、跟踪和控制活动，保证按时保质地完成计划的任务；
- 项目结束：在项目执行完毕时进行的总结。

软件项目度量

- 所谓度量，是指根据已明确的规则把数字或符号指定给现实世界中实体的某一属性，以便阐述实体的某种状态。
- 软件度量涉及的范围较广，其度量实体大致划分为三大类：
  - 产品：是指在软件开发过程中产生的各种中间产品、发布的资料和文档等，如规格说明书、设计模型、代码、测试用例等。
  - 过程：是与软件相关的一些活动，如编制规格说明书、详细设计、测试等活动。
  - 资源：是指开发过程中使用的资源，包括人员、团队、软件和硬件、办公地点等。

软件度量的内容

- 软件度量就是为了获取上述实体属性的值。这些实体的属性又划分为内部属性和外部属性。
  - 内部属性：是能够纯粹用实体自身来度量的属性。如产品中设计模块实体的内部属性有：规模、可复用性、耦合度、内聚度等。
  - 外部属性：是指由实体与其相关环境一起共同才能度量的属性。如产品中设计模块实体的外部属性有质量、复杂性、可维护性等。
- 实体属性的度量又可分为直接度量和间接度量：
  - 直接度量：指实体属性的度量不依赖于其他属性的度量。
  - 间接度量：指实体属性的度量与一个或多个其他属性的度量标准有关。

软件项目规模度量

- 其主要目的是为软件项目估算建立基线，是估算软件项目工作量、编制成本预算、策划合理项目进度的基础。
- 其度量对象包括软件产品、软件开发过程和软件资源；
- 需要度量的属性包括：
  - 项目投入的费用、投入的人力、持续的时间；
  - 产生的代码行数、完成的功能点数；
  - 发生的错误数；
  - 软件的生产率、软件质量等。

代码行度量方法

- 代码行指所有的可执行的源代码行数，包括可交付的动作和控制语句、数据定义、数据类型声明等。

项目编码	工作量（人月）	成本（万元）	KLOC	文档页数	错误数	人数
aaa-01	24	16.8	12.1	365	29	3
ccc-04	62	44	27.2	1224	86	5
fff-03	43	31.4	20.2	1050	64	6
...	...	...	...	...	...	...

- 生产率 = KLOC / 工作量（人月数）
- 质量 = 错误数 / KLOC
- 单位成本 = 成本 / KLOC
- 单位文档 = 文档页数 / KLOC

功能点度量

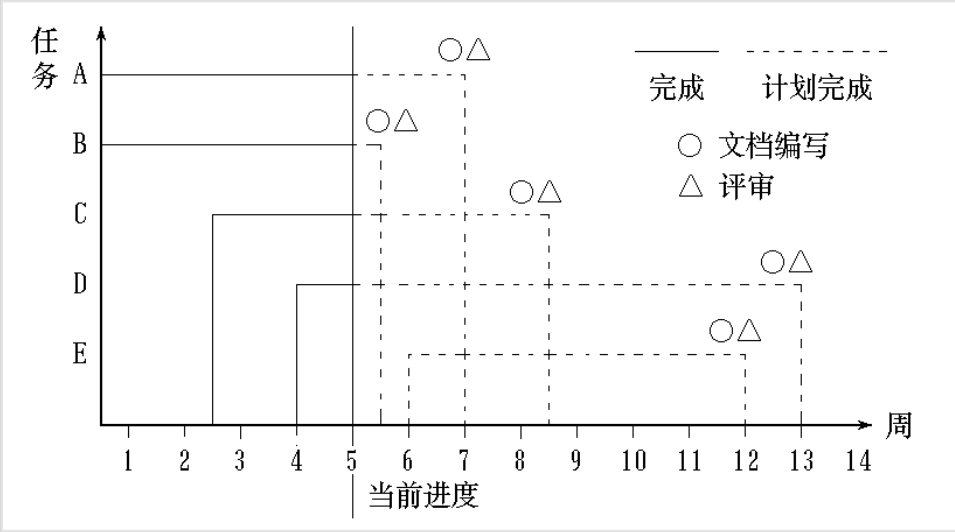
- 功能点度量方法是由IBM公司的工程师（Allan Albrecht）于20世纪70年代提出的，是一种生产率度量法。
- 该方法利用程序的“功能性”和“实用性”，及有关软件数据域的一些计数度量和软件复杂性估计的经验关系式，导出功能点。
- 功能点度量方法需要事先确定五个数据域特征计数：
  - 外部输入数：对每个用户的输入进行计数。
  - 外部输出数：对每个用户得到的输出进行计数。
  - 外部查询数：一个查询被定义为一次联机输入。
  - 内部逻辑文件数：每一个逻辑主文件都应计数。
  - 外部接口文件数：对所有将信息传送到另一个系统中的接口（如磁带、磁盘和可读写光盘上的数据文件）均应计数。
- 计算调整后的功能点：
  - FP = 总计数×（0.65 + 0.01×sum（Fi））， Fi为环境因素

软件项目估算

- 为了制定合理有效的项目计划，就必须事先进行项目估算，确定项目的范围、所需的资源、所能投入的成本以及项目开发所必需的时间。
  - 明确项目范围：包括软件功能、性能、约束、接口和可靠性等；
  - 估算项目资源：包括人力资源、开发环境及可复用的软件构件；
  - 估算成本和工作量：根据软件项目的规模以及以往的经验建立估算项目基线以计算项目的成本和工作量；
    - 基于分解技术的估算模型
    - 基于经验的估算模型
    - COCOMO模型等
  - 确定项目的开发时间：根据上述三项内容及甘特图和PERT技术确定每项任务的关键路径，最终可得到最短、最合理和最长的项目开发时间，从而制定一个合理的项目开发计划。

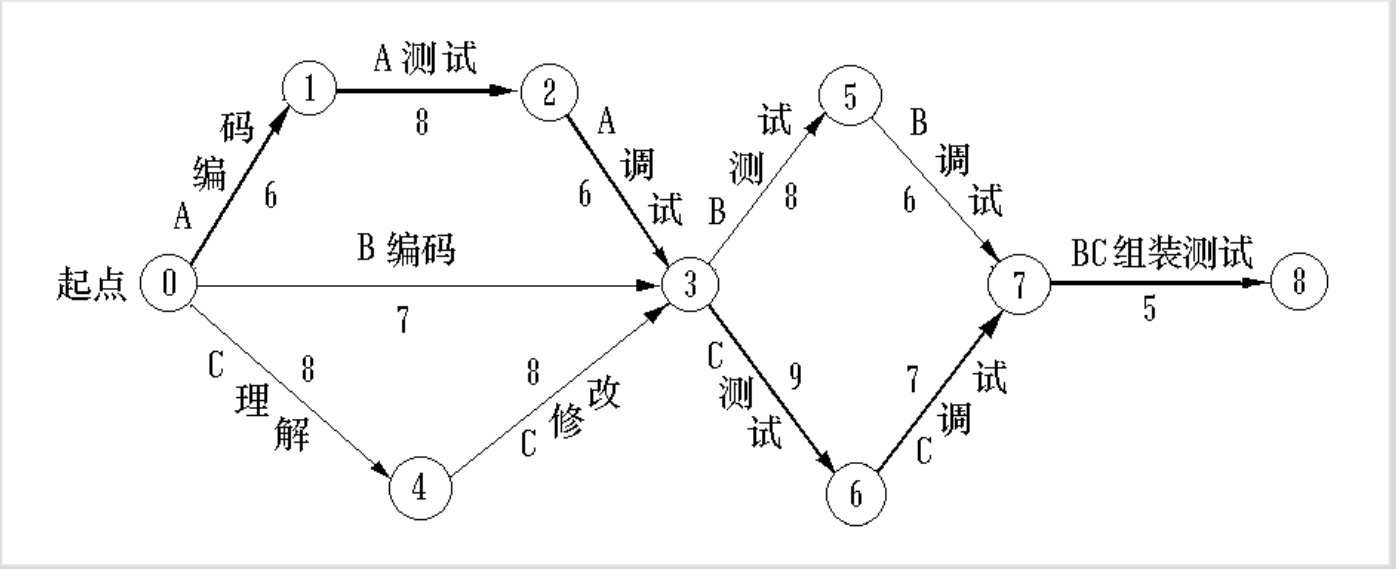
软件项目的进度安排

- 软件项目的进度计划和工作的实际进展情况，需要采用图示的方法描述，特别是表现各项任务之间进度的相互依赖关系。
  - 各任务的计划开始时间，完成时间；
  - 各任务完成的标志（即○文档编写和△评审）；
  - 各任务与参与工作的人数，各个任务与工作量之间的衔接情况；
  - 完成各个任务所需的物理资源和数据资源。



任务完成时间的估计

- 乐观时间ai：顺利的情况下，完成第i项任务的时间。
- 最可能时间mi：正常情况下完成第i项任务的时间。
- 悲观时间bi：最不利的情况下完成第i项任务的时间。
- 由此可算出第i个任务期望完成时间
  - Ti = (ai+4mi+bi)/6



软件项目的组织结构

- 一个大型的软件项目参与人员通常组织成多个开发小组，每个小组有合适数量的参与人员，为了发现开发小组最大的工作效率，必须对项目小组成员进行有效地组织；其原则如下：
  - 尽早落实责任：软件项目要尽早指定专人负责，使他有权有责。
  - 减少接口：一个小组的生产率是和完成任务中存在的沟通途径数目成反比的。
  - 责权均衡：软件经理人员所负的责任应与授予给他的权力对等，不要出现有责无权或者有权无责的不对等情况。

组织结构的模式

- 按课题划分的模式：把软件人员按课题组成小组，成员自始至终参加所承担课题的各项任务。
- 按职能划分的模式：把参加开发项目的软件人员按任务的工作阶段划分成若干专业小组。待开发的软件产品在每个专业小组完成阶段加工以后，沿工序流水线向下传递。
- 矩阵模式：一方面，按工作性质，成立一些专门组，如开发组、业务组、测试组等；另一方面，每一个项目又有它的经理人员负责管理。每个软件人员属于某一个专门组，又参加某一项目的工作。