

第11章 软件项目管理

- ❖ 软件项目管理概述
- ❖ 软件项目规模度量
- ❖ 软件项目估算
- ❖ 项目进度管理
- ❖ 软件配置管理
- ❖ 软件能力成熟度模型

软件项目管理概述

什么是项目？

- ❖ 建造一座大楼、一座工厂或一座水库
- ❖ 举办各种类型的活动，如一次会议、一次晚宴、一次庆典等
- ❖ 新企业、新产品、新工程的开发
- ❖ 进行一个组织的规划、规划实施一项活动
- ❖ 进行一次旅行、解决某个研究课题、开发一套软件



在当今社会中，一切都是项目，一切也将成为项目。

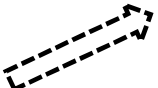
——美国项目管理专业资质认证委员会主席 **Paul Grace**

软件项目管理概述

项目的定义



❖ 项目是一个特殊的将被完成的**有限任务**，它是在一定时间内，满足一系列特定目标的多项相关工作的总称。

❖ 此定义实际包含三层含义： **项目需求**

➤ 项目是一项**有待完成的任务**，且有特定的环境与要求；

 **项目实施与管理**

➤ 在一定的组织机构内，利用**有限资源**（人力、物力、财力等）在规定的时间内完成任务；

➤ 任务要满足一定性能、质量、数量、技术**指标**等要求。

 **项目过程及评估**

软件项目规模度量

- 任何软件项目都需要定量描述才能制定软件开发成本。
- 没有定量的项目将难以展开软件管理和实施过程。

对软件产品的度量分为直接度量和间接度量。

- 直接度量：通过对软件产品的简单属性直接计算而得到结果的过程。简单属性是指代码行数、操作数和运算符个数、接口个数等能直接计数的特征，它反映的是软件产品内部特征。
- 间接度量：通过对软件产品的简单属性、要素的各项特征、准则的经验值间接计算而得到结果的过程。如软件质量评价、软件复杂性测量等。

软件项目规模度量

1. 直接度量——代码行技术

代码行技术是指用程序的代码量来衡量软件的规模。程序的代码量用代码行（**Line of Code, LOC**）表示。

目前，几乎所有软件开发团队都保留代码行数据，同时也把代码行作为宣传、描述软件产品的一个重要数据。

通过代码行度量，还可以得到软件开发生产率、每行代码的平均成本，文档代码的比值、每千代码行的错误率等，这些数据不仅评价系统规模，还能评价软件质量、文档管理等要素。

软件项目规模度量

1. 直接度量——代码行技术

生产率: $P = KL/E$

每行代码的平均成本: $C = S / KL$

文档与代码比: $D = PG / KL$

代码出错率: $R = ER / KL$

KL: 代码行数用KLOC (千行代码)

其 E: 工作量用人月 (PM) 度量

S: 项目总开销

中 PG: 项目文档页数

ER: 项目代码错误数

例: 国外典型软件项目记录如表可计算: P_1 、 C_1 、 D_1 、 EQR_1 的值

项目	工作量(E)	成本(\$)	代码千行	文档页数	错误	人数
Aaa-01	24	168.000	12.1	365	29	3
Ccc-04	62	440.000	27.2	1 224	86	5
Fff-03	43	314.000	20.2	1 050	64	6

软件项目规模度量

1. 直接度量——代码行技术

争议 是否使用代码行数（LOC）做为度量的依据？

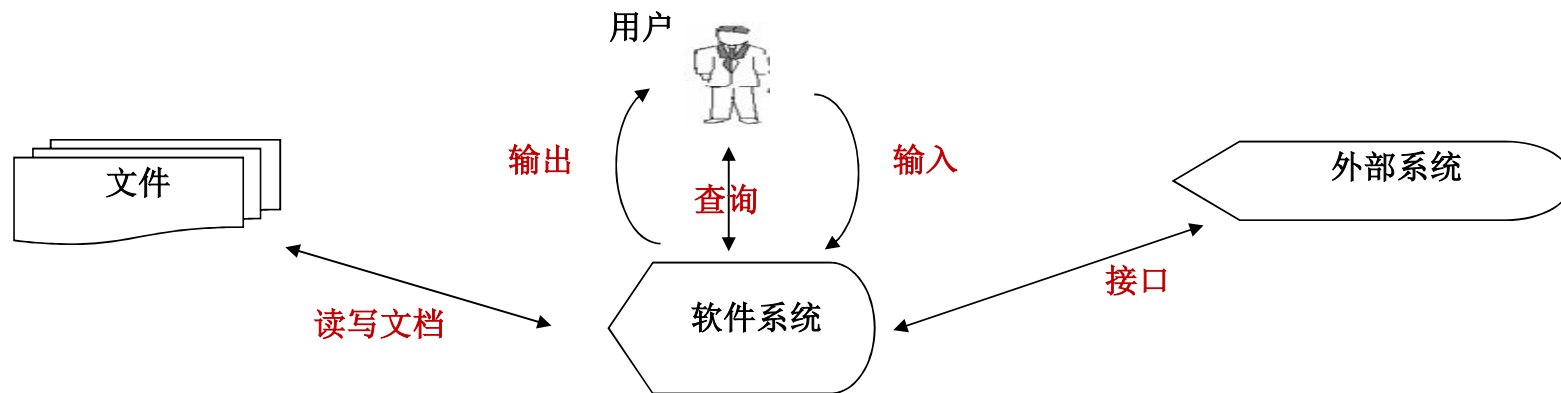


软件项目规模度量

2. 间接度量——面向功能点的度量

鉴于相同功能，不同语言实现存在代码量的差异，且软件需求是面向功能的描述，**Albrecht**提出了面向功能点（**Function Point, FP**）的度量。

面向功能点的度量是基于定义的**5**个信息领域的特征数，以及**14**项技术复杂性因子综合进行的间接度量。下图描述了可用于功能点度量的**5**个信息领域特征。



软件项目规模度量

2. 间接度量——面向功能点的度量

$$FP = \Sigma(\text{总计数值} \times \text{加权因子}) \times (0.65 + 0.01 \times \Sigma F_i)$$

面向功能点的度量是用5个信息域（总计数值）取值：

- ❖ 输入项数： 计算每个用户输入数据数；
- ❖ 输出项数： 计算每个用户输出数据（报表、屏幕、出错信息）；
- ❖ 查询数： 一个查询被定义为一次联机输入，每一个不同查询都计算；
- ❖ 文档数： 计算每个逻辑的主文件数；
- ❖ 外部接口数： 计算可读的接口（包括I/O，外部系统访问、网络等）。

复杂度的加权因子：

	简单	平均	复杂
用户输入数	3	4	6
用户输出数	4	5	7
用户查询数	3	4	6
文件数	7	10	15
外部接口数	5	7	10

软件项目规模度量

2. 间接度量——面向功能点的度量

在计算出功能点之后，可仿照LOC的方式度量软件的生产率、质量和其它属性：

$$\text{生产率} = \text{FP} / \text{E}$$

$$\text{质量} = \text{ER} / \text{FP}$$

$$\text{成本} = \text{S} / \text{FP}$$

$$\text{文档} = \text{ER} / \text{FP}$$

面向功能点的度量的缺点：

- ❖ 对于复杂性技术因素确定的主观因素太多，量化标准把握不统一。
- ❖ 计算所涉及的某些数据不易采集，同时也由于主观因素造成数据确定困难。
- ❖ 对于计算得到的功能点，难以说明其高或低的结果是由哪些因素所导致。

分解、细化，最后定义量化标准

软件项目规模度量

3. 调和不同的度量方法

代码行 (LOC) 和功能点度量之间的关系依赖于实现软件所采用的程序设计语言及设计的质量。

对于相同功能点各种语言的平均代码行数的估算。

语言	LOC/FP（平均值）
Assembly	320
C	128
Pascal	91
Fortran	58
Basic	64
High-Order	105
4GL	15

软件项目估算

对于将要开发的软件系统，常用的估算方法有以下几类：

- ❖ 对比已有项目产生的实际成本，结合当前项目的需求估算项目成本和工作量。
- ❖ 总结已有项目的数据，分析并概括软件项目成本和工作量的经验公式。
- ❖ 按项目中的问题分解。
- ❖ 按过程分解。

软件项目估算

1. 估算模型——代码行、功能点的其它估算模型

(1) 对于代码行和工作量之间的关系，**Boehm**给出了一个基本模型：

$$E = 3.2 * KLOC^{1.05}$$

其中，**E**是工作量，**KLOC**是千代码行。

(2) 如果估算的代码行较大（大于**9KLOC**），**Poty**给出了一个更为准确的模型：

$$E = 5.288 * KLOC^{1.047}$$

(3) **IBM**公司也提供了估算代码行与工作量间关系的模型：

$$E = 5.2 * KLOC^{0.91}$$

$$M = 4.1 * KLOC^{0.36}$$

$$D = 49 * KLOC^{1.01}$$

$$P = 0.54 * E^{0.6}$$

其中，**M**是项目开发时间（月），**D**是文档数量（页），**P**是所需人员（人）。

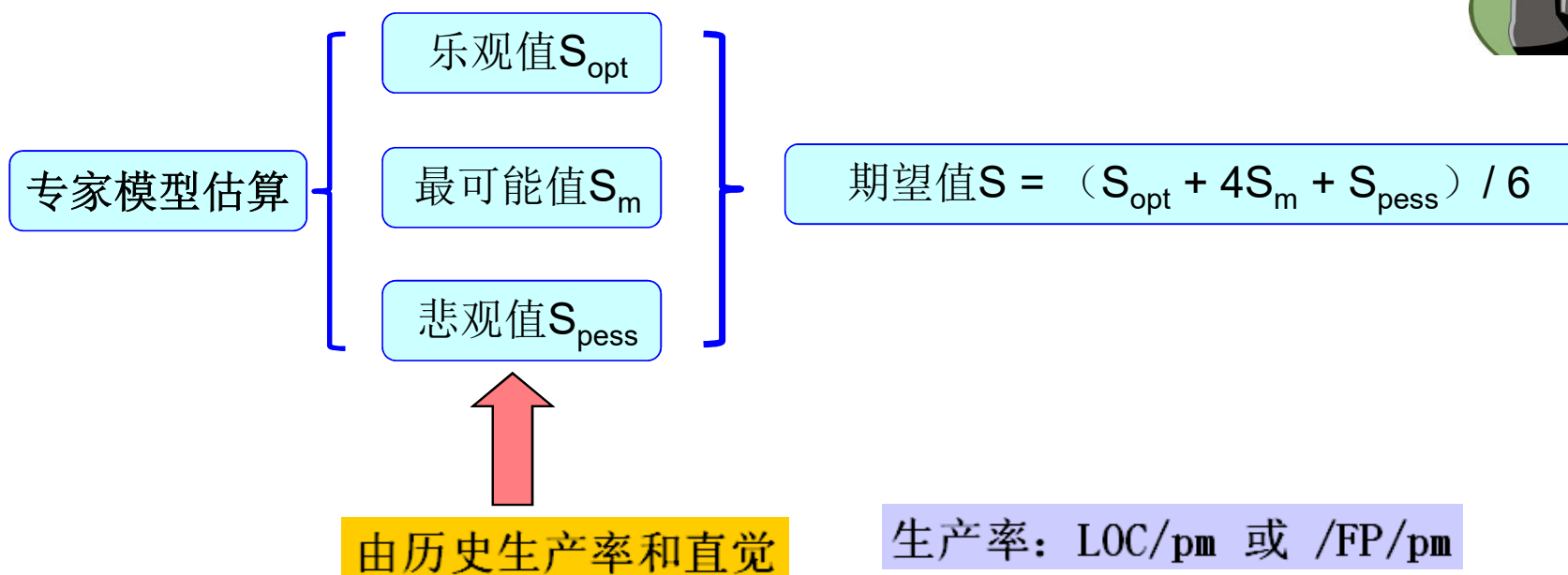
(4) 对于功能点和工作量之间的关系，**Albrecht**和**Gaffney**给出了一个基本模型：

$$E = -13.39 + 0.054FP$$

软件项目估算

2. 估算模型——专家估算模型

专家模型：通过借鉴历史信息，专家提供项目估算所需的信息，或根据以往类似项目的经验，给出相关参数的估算值。



软件项目估算

3. 估算模型——Putnam模型

Putnam软件方程式通过对4000多个软件项目生产率的统计而获得：

$$E = L^3 / (C^3 * T^4)$$

其中： L = 代码行数

T = 软件开发时间

C = 技术状态常数

$$C = \begin{cases} 2000, & \text{比较差的软件开发环境} \\ 8000, & \text{一般的软件开发环境} \\ 11000, & \text{比较好的软件开发环境} \end{cases}$$

软件项目估算

4. 估算模型——COCOMO (Constructive Cost Model) 模型

有三个层次，分别用于开发3个阶段性的COCOMO模型：

- ❖ 基本模型：用于系统初期估算软件开发工作量（及成本）和软件开发所需时间
- ❖ 中级模型：用于估算软件各个子系统开发工作量（及成本）和开发所需时间
- ❖ 详细模型：包含模型2的所有特性，对软件开发过程中每一步骤的影响评估。

项目进度管理

项目进度控制

软件项目进度安排通过把工作量分配给特定软件工程阶段，并规定完成各项任务的起止日期。进度计划将随着项目进程的变化而会有所更改，但作为整个项目开发时间的宏观控制，则不应有太大变化。

软件项目能否按计划时间完成并及时交付合格的产品是项目管理的重点，也是客户关心的重要内容。但如果为了缩短开发时间，则会大大增加项目的工作量。**Putnam**模型已明确说明二者的关系。

项目进度管理

项目进度控制

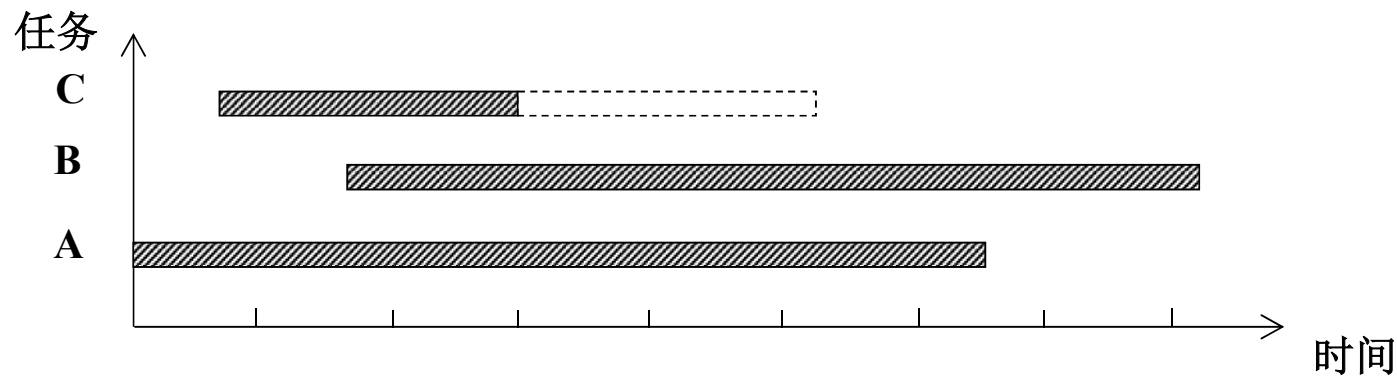
对于软件项目延期的原因，通常有以下几种情形：

- (1) 项目进度本身不合理。
- (2) 团队或小组成员的问题。
- (3) 软件架构的问题。
- (4) 对项目各阶段任务所需的资源投入不足，这源于对各阶段工作量的估算不充分。
- (5) 在项目开发过程中，遇到难以克服的困难。
- (6) 用户需求变更。
- (7) 项目风险管理未做好。

项目进度管理

甘特图

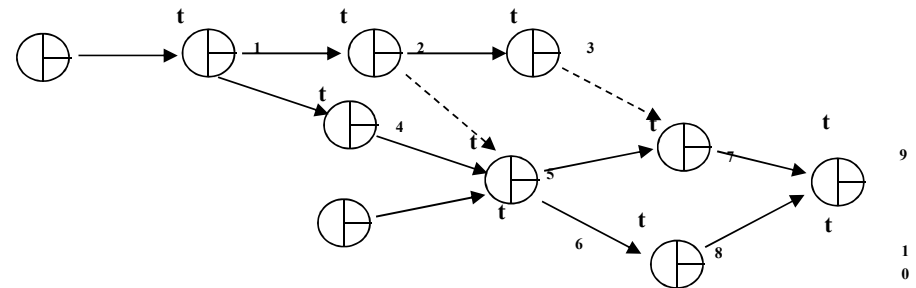
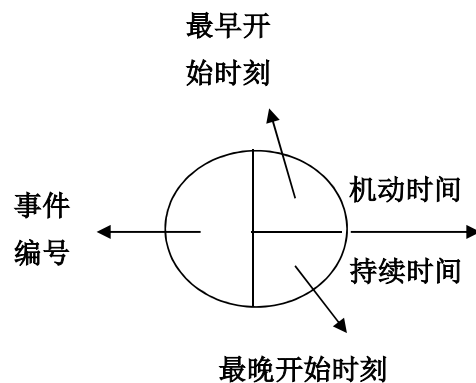
甘特图（**Gantt Chart**）是表示工作进度计划及工作实际进展状况最为简明的图形表示方法。它是历史悠久，使用广泛的进度计划工具之一。



项目进度管理

工程网络图

工程网络图是制定项目计划时又一种常用的图形工具，它不仅能描述项目的起止时间和各项任务的工期，更能现实的描述各任务间彼此的依赖关系。



软件配置管理

- **软件配置管理**（**Software Configuration Management, SCM**）是对软件修改进行**标识**、**组织**和**控制**的技术。**SCM**的目的是通过定义管理软件变化的一组活动来减少由此引起的混乱，提高软件生产率。

SCM定义的变更管理活动主要包括：

- 标识变化；
- 控制变化；
- 监督、记录变化过程；
- 通知与变化相关的所有人员、更新与变化相关的文档。



软件配置管理

软件配置项

- (1) 系统规格说明；
- (2) 软件项目计划，包括软件开发计划、质量保证计划、配置计划和验收确认计划；
- (3) 软件需求规格说明；
- (4) 软件设计规格说明，包括概要设计说明和详细设计说明。
- (5) 程序，包括源代码、目标文件、可执行文件、软件部件库、数据等。
- (6) 软件测试文档，包括测试计划、测试用例、测试脚本和测试报告。
- (7) 维护文档，包括软件维护计划、软件问题报告、变更报告。
- (8) 用户文档，包括用户手册、联机帮助、安装和部署文档等。
- (9) 软件工程和软件质量管理标准与过程。
- (10) 对上述各项内容的按需组合，构成复合软件配置项（针对中小型项目，以及极限编程、敏捷编程等软件开发过程的需要）。



软件能力成熟度模型

CMM

1986年底美国SEI/CMU开始着手研究和制定支持软件开发组织控制、管理和改进软件过程的软件过程成熟度框架，并于1987年开发了“软件过程评估”和“软件成熟度评价”两个模型，于1991年发布软件能力成熟度模型（**Capability Maturity Model for Software, SW-CMM或CMM**），陆续发布了**CMM v1.0（2002年）**和**CMM v2.0（2018年）**。

软件能力成熟度模型

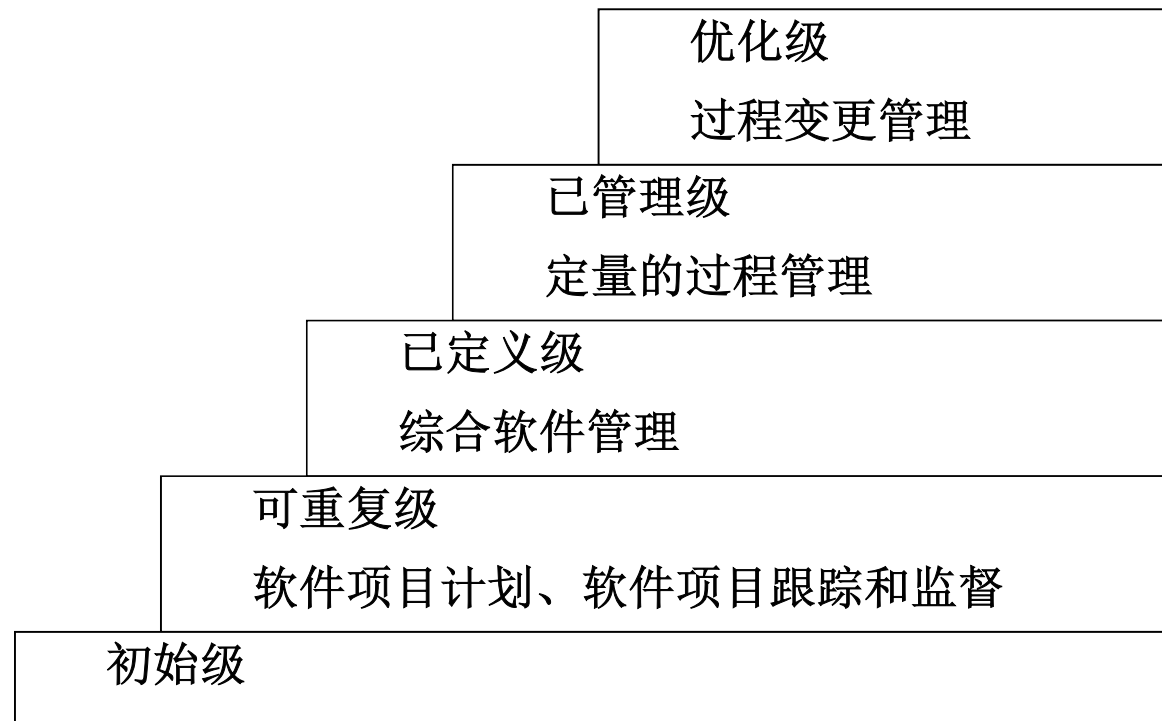
软件能力成熟度等级

- 软件过程成熟度（**Capability Maturity Model for Software, CMM**）是对软件组织在项目定义、组织构建、管理实施、项目度量、过程控制和改善的实践中，对各个开发阶段和管理过程的描述。
- **CMM**把软件过程的改进过程划分为**5**个等级，每个等级都有各自软件过程的基本特征、实践任务和管理目标，每个等级都为过程改进的继续提供基础。当每个等级的过程实施达到该等级的过程目标，对该等级的特征、任务和管理建立一个重要成分并稳定下来，从而也导致在**CMM**框架中软件开发组织的过程能力得到一定程度的提高。



软件能力成熟度模型

软件能力成熟度等级



软件能力成熟度模型

关键过程域

关键过程域（**Key Process Area, KPA**）是描述软件过程的属性集合。它通过定义一组相互关联的软件实践活动和有关的基础设施，达到成熟度等级的目标，同时体现和提高软件过程能力。

等级	成熟度	可视性	过程能力	关键过程域
1	初始级	有限的可视性	一般达不到进度和成本的目标。	
2	可重复级	具有管理可视性	由于基于过去的性能，项目开发计划比较现实可行。	<ul style="list-style-type: none">● 需求管理● 软件项目计划● 软件项目跟踪与监督● 软件子合同管理● 软件质量保证● 软件配置管理
3	已定义级	项目定义软件过程的活动具有可视性	基于已定义的软件过程，组织持续地改善过程能力。	<ul style="list-style-type: none">● 软件机构过程关注点● 软件机构过程定义● 培训计划● 整体化软件管理● 软件产品工程● 组间合作● 同行评审
4	已管理级	定量地控制软件过程	基于对过程和产品的度量，组织持续地改善过程能力。	<ul style="list-style-type: none">● 定量过程管理● 软件质量管理
5	优化级	持续改善软件过程	组织持续地改善过程能力。	<ul style="list-style-type: none">● 过程变更管理● 预防故障● 技术变更管理