第 8 章 软件项目管理

软件项目管理是对软件项目开发全过程的管理，是对整个软件生存期的所有活动进行的管理。任何工程的成败，都与管理的好坏密切相关。软件项目更是如此，由于软件产品所具有的特殊性，软件项目的管理对于保证软件产品的质量具有极为重要的作用，是软件项目开发成功的关键。

随着软件的规模和复杂度的不断增大，开发人员的增加，以及开发时间的增长，这些都增加了软件项目管理的难度，同时也突出了软件项目管理的必要性和重要性。事实证明由管理失败造成的后果要比开发技术错误造成的后果更为严重。很少有软件项目的实施进程能准确地符合预定目标、进度和预算的，这也就足以说明软件管理的重要性。

例如：Windows 2000 的开发是微软公司历史上最艰巨的任务，仅仅是核心部门的成员就有 2500 人，测试用的代码就有 1000 万行，测试中所用到的脚本程序就有 6500 种。类似规模如此之大的软件系统，如果没有科学的、规范的、有效的管理，是不可能成功的。因此软件项目管理是软件工程的重要研究内容之一。

8.1 软件项目管理概述

软件项目管理就是为了使软件项目能够按照预定的成本、进度、质量顺利完成，而对人员（People）、产品（Product）、过程（Process）和项目(Project)进行分析和管理的活动。软件项目管理先于任何技术活动之前开始，并且贯穿于软件的整个生命周期。

软件项目的组织管理，不仅仅需要技术、工程或科研方面的知识，而且需要多方面的综合知识，这些知识涉及到系统工程、管理学、统计学、心理学、社会学、经济学，乃至法律等方面的问题。尤其涉及到社会因素、精神因素、人的因素，这远远不是简单的技术问题。

软件项目的管理，既要学习国外先进的管理经验，也不能完全照搬外国的管理技术，还必须结合我国的实际情况，根据我们的工作条件、人员和社会环境等多种因素全面考虑。

此外，实践是取得管理经验的重要途径，是管理技术的基础。很显然，管理能够取得效率，能够赢得时间，是软件项目能够开发成功的关键。

8.1.1 软件项目管理的特点

软件产品与其他任何产业的产品不同，它是非物质性的产品，是知识密集型的逻辑思维产品。对于这样看不见摸不着的产品，将思想、概念、算法、流程、组织、效率和优化等因素综合在一起，它是难以理解和驾驭的产品。由于软件的这种独特性，使软件项目管理过程更加复杂和难以控制。

由此软件项目管理的主要特点为：

① 软件项目管理涉及的范围广，它涉及到软件开发进度与计划、人员配置与组织、项目跟踪与控制等。

② 综合应用多方面的知识，特别是要涉及到社会的因素、精神的因素、认知的因素，这比技术问题复杂得多。

③ 人员配备情况复杂多变，组织管理难度大。

④ 管理技术的基础是实践，为取得管理技术成果必须反复实践。

8.1.2 软件项目管理的主要活动

为使软件项目开发成功，必须对软件开发项目的工作范围、可能遇到的风险、需要的资源、要实现的任务、经历的里程碑、花费的工作量，以及进度的安排等做到心中有数。而软件项目管理可以提供这些信息。任何技术先进的大型项目的开发，如果没有一套科学的管理方法和严格的组织领导，是不可能取得成功的。即使在管理技术较成熟的发达国家中都尚且如此，在我国管理技术水平不高、资金比较紧缺的情况下，重视大型软件项目开发的管理方法及技术就显得尤为重要。

软件项目管理的对象是软件工程项目，因此软件项目管理涉及的范围将覆盖整个软件工程过程。软件项目管理的主要活动有：

(1) 软件可行性分析

即从技术、经济和社会等方面对软件开发项目进行估算，避免盲目投资，减少损失。

(2) 软件项目的成本估算

在开发前估算软件项目的成本，以减少盲目工作，降低软件项目风险。

软件项目的成本估算，重要的是对项目所需资源的估算。即在软件项目开发前，对软件项目开发所需资源进行估算。一般采用“金字塔”表示软件开发所需的资源，如图 8.1 所示。

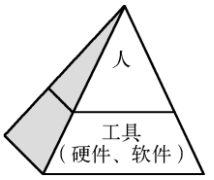

图 8.1 软件开发所需的资源

①人力资源。在考虑各种软件开发资源时，人是最重要的资源。在安排开发活动时必须考虑人员的技术水平、专业、人数，以及在开发过程中各阶段对各种人员的需求，如图 8.2 所示，Putnam−Norden 曲线描述了软件开发各阶段，对各类人员的需求情况。

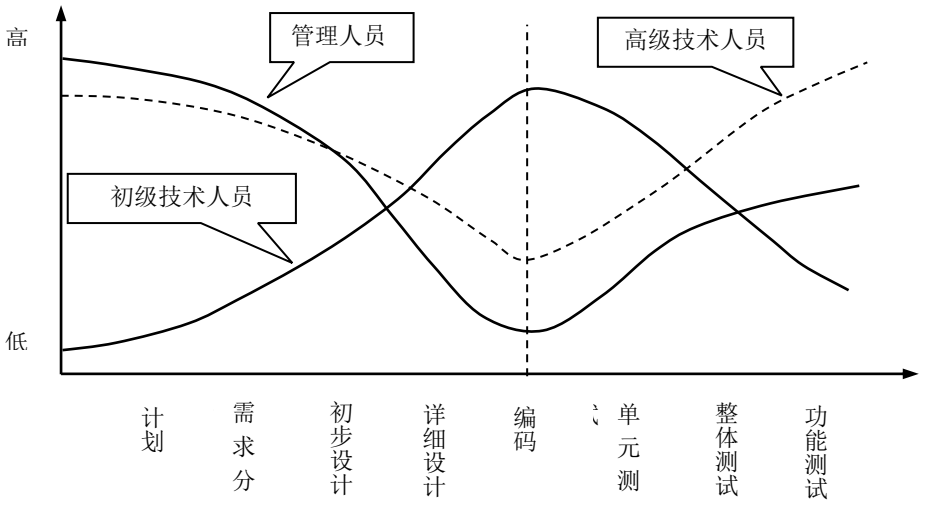

图 8.2 Putnam−Norden 曲线

②硬件资源。在计划软件项目开发时，考虑三种可能的硬件资源，主要包括宿主机（软件开发时使用的计算机及外围设备）、目标机（运行已开发成功的软件的计算机及外围设备）和其他硬件设备（专用软件开发时需要的特殊硬件资源）。

③软件资源。软件在开发期间使用了许多软件工具来帮助软件的开发。因此软件资源实际就是软件工具集，主要软件工具分为业务系统计划工具集、项目管理工具集、支援工具、分析和设计工具、编程工具、组装和测试工具、原型化和模拟工具、维护工具、框架工具等。通常可提供支持软件开发全过程的集成化的软件工程环境。

④软件复用性及软件构件库。为了提高软件的生产率和软件产品的质量，应建立可复用的软件构件库。对于软件的复用，经常被人们忽略，但这却是相当重要的一环。

（3）软件生产率

通过对影响软件生产率的五种因素（人、问题、过程、产品和资源）进行分析，在软件开发时，更好地进行软件资源配置。

（4）软件项目质量管理

软件项目的质量管理也是软件项目开发的重要内容，对于影响软件质量的因素和质量的度量都是质量管理的基本内容。

（5）软件计划

开发软件项目的计划涉及到实施项目的各个环节，带有全局的性质。计划的合理性和准确性往往关系着项目的成败。

（6）软件开发人员管理

软件开发的主体是软件开发人员，对软件开发人员的管理十分重要，它直接关系到如何发挥最大的工作效率和软件项目是否开发成功。

8.2 软件项目计划

软件项目计划是一个软件项目进入系统实施的启动阶段，主要进行的工作包括：确定详细的项目实施范围、定义最终的工作成果、评估实施过程中主要的风险、制定项目实施的时间计划、成本和预算计划、人力资源计划等。

8.2.1 软件项目计划内容

软件项目计划的目标是为项目开发人员提供一个框架，使之能合理地估算软件项目开发所需的资源、经费和开发进度，并有效地控制软件项目开发过程，使开发过程能够按此计划进行。在做计划时，必须就需要的人力、项目持续时间及成本做出估算。

制订软件项目计划的目的在于建立并维护软件项目各项活动的计划，软件项目计划其实就是一个用来协调软件项目中其它所有计划，指导项目组对项目进行执行和监控的文件。一个好的软件项目计划可为项目的成功实施打下坚实的基础。

软件项目计划的主要内容如下：

1．项目范围

对该软件项目进行综合描述，定义软件所要做的工作。包括对以下开发内容的概述：项目目标、主要功能、性能限制、系统接口、项目的特殊要求等。

2．项目资源

包括人员资源、硬件资源、软件资源和其他可能需要的资源。着重强调对项目规模和资源的估算，是因为低质量的项目资源估算将不可避免地造成资源短缺，进度延迟和预算超支。又由于项目资源估算是从软件规模估算中直接衍生出来的，所以低质量的规模估算是造成许多软件项目问题的根本原因。

3．项目进度安排

进度安排是软件计划的重要内容。直接影响到整个项目能否按期完成，因此这一环节是十分重要的。

在制定软件项目进度安排时，主要依据是合同书和项目计划。通常的做法是把复杂的软件项目分解成许多可以准确描述、度量、可独立操作的相对简单的任务，然后安排这些任务的执行顺序，确定每个任务的完成期限、开始时间和结束时间。软件项目进度安排考虑的主要因素有：

- ●项目可以支配的人力及资源
- ●项目的关键路径
- ●生存周期各个阶段工作量的划分
- ●工程进展如何度量
- ●各个阶段任务完成标志
- ●如何自然过渡到下一阶段的任务等

4．制定软件项目计划要注意的问题

由于软件项目有其特殊性，不确定因素多，工作量估计困难，因此，项目初期难于制定一个科学、合理的项目计划，需要在计划执行过程中不断修正和精化。

8.2.2 软件开发进度计划

软件开发进度计划安排是一件困难的任务，进度安排的好坏往往会影响整个软件项目能否按期完成，因此在安排软件开发进度时，既要考虑各个子任务之间的相互联系，尽可能并行地安排任务，又要预见潜在的问题，提供意外事件的处理意见。

对于较大软件项目的进度计划，为了表示各项任务之间的进度的相互依赖关系，可以采用下面介绍的几种工具来描述计划进度，即一般的表格工具、甘特图、PERT 技术与 CPM 方法。

1. 一般的表格工具

一般采用表格描述进度表，它非常简单明了。图 8.3 所示直观地给出了一个需要一年时间开发的软件项目各项子任务的进度安排。

月份 / 任务	1 2 3 4 5 6 7 8 9 10 11 12
需求分析	▲▲▲
总体设计	▲▲▲
详细设计	▲▲
编码	▲▲▲▲
软件测试	▲ ▲ ▲

图 8.3　进度表

2. 甘特图

甘特图（Gantt Chart）是先把任务分解成子任务，再用水平线段描述各个任务的工作阶段，线段的起点和终点分别表示任务的开始和完成时间，线段的长度表示完成任务所需的时间。图 8.4 给出了具有五个任务的甘特图。

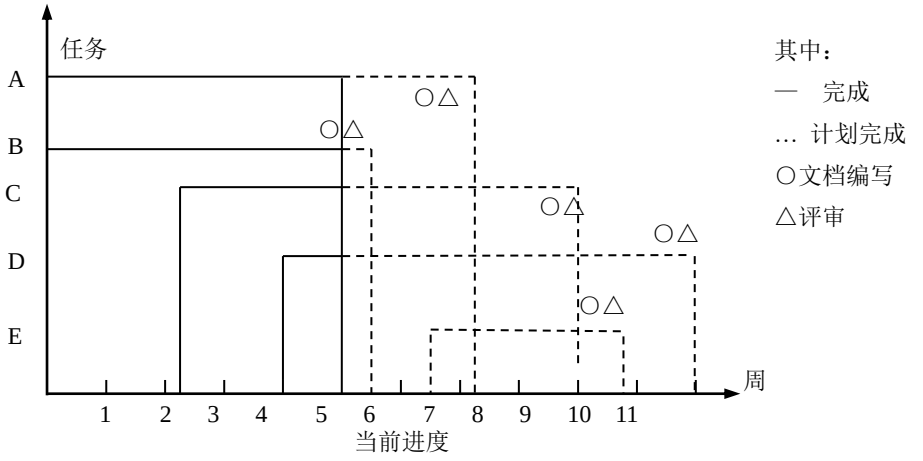

图 8.4 甘特图

甘特图只能表示任务之间的并行和串行关系，它标明了各任务的计划进度和当前进度，能够动态反映软件开发的进展情况，但是它不能够反映多个任务之间的复杂逻辑关系。

3. 时标网状图

时标网状图（timescalar network）也称为改进的 Gantt 图，主要增加了各子任务之间的逻辑依赖关系。如图 8.5 所示，它表示 A～E 共 5 个任务之间在进度上的依赖关系，如 E2 的开始取决于 A3 的完成，虚箭头表示虚任务。

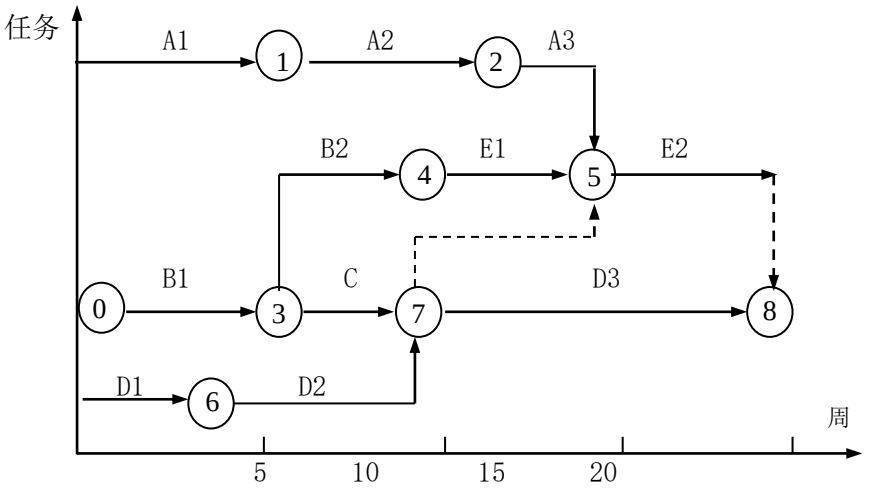

图 8.5 时标网状图

4. PERT 技术和 CPM 方法

PERT（Program Evaluation & Review Technique）计划评审技术或 CMP（Critical Path Method）关键路径法，都是采用网络图来描述项目的进度安排。例如，图 8.6 描述了开发模块 A、B、C 的任务网络图。其中模块 A 是公用模块，模块 B 和模块 C 的测试有助于模块 A 调试的完成。模块 C 利用了现有的模块，但对它要在理解之后做部分修改。最后直到模块 A、B 和 C 做组装测试为止。整个工作步骤如图 8.6 所示，图中各边上所标注的数字为该任务所持续的时间，单位为周。数字结点为任务的起点和终点。

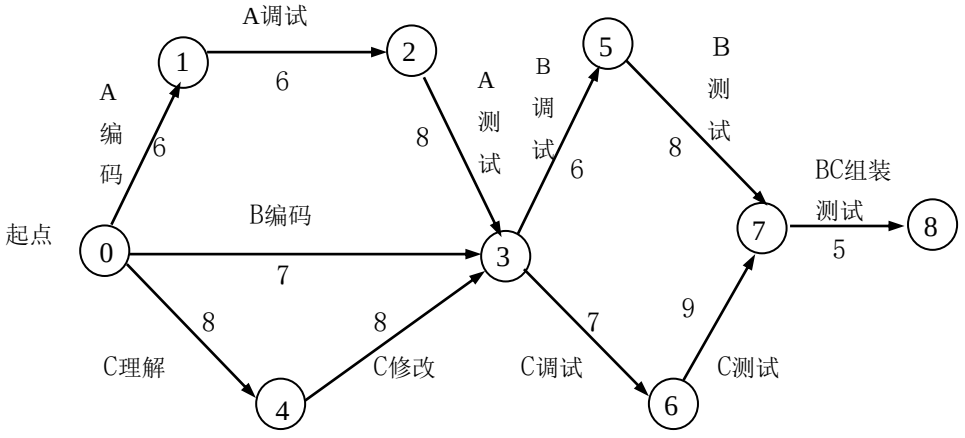

图 8.6 任务网络图

最后，在软件工程项目中必须处理好进度与质量的关系。在软件开发过程中，常常会遇到在追求进度时赶任务，在这种进度的压力下，有可能牺牲软件质量。同时产品质量和生产

率也有着密切的关系。

8.3 软件项目成本估算技术

在计算技术发展的早期，软件成本在基于计算机系统的总成本中占很小的比例，因此在软件成本的估算上，即使出现极大的误差，其影响也非常小。现在的软件在计算机系统中已成为最贵和最有价值的部分，因此要求软件的成本估算不能再出现较大的误差。软件的成本估算是可行性分析的重要依据，也是软件管理的重要内容，直接影响到软件开发的风险。

软件的成本和工作量的估算从来都没有成为一门精确的科学，因为变化和影响的因素太多，人、技术、环境等都会影响软件开发的最终成本和工作量。但是软件的成本估算又非常重要，因此人们不断地从理论和统计等方面归纳总结出一些估算方法和技术，尽可能降低盲目开发软件的风险。

8.3.1 影响成本估算的因素

由于成本估算是软件项目开发管理的重要内容，是可行性分析的重要依据。为了正确地进行成本估算，首先应该充分了解影响成本估算的主要因素，从而更有效地进行成本估算。主要影响因素如下：

（1）开发软件人员的业务水平

软件开发人员的素质、经验、掌握知识水平的不同，在工作中表现出很大的差异，直接影响到软件的质量与成本。

（2）软件产品的规模及复杂度

按 YOURDON 分类法，将软件产品的规模分为如表 8.1 中的几类，每一类都具有不同的特点，在管理时要分别对待。

对于软件产品的规模的度量，一般是以开发时间和产品规模作为主要的分类指标。

表 8.1 软件产品规模分类表

类别	参加人员	研制时间	产品规模（源代码行）
微型	1	1～4 周	0.5k
小型	1	1～6 月	1～2k
中型	2～5	1～2 年	5～20k
大型	5～20	2～3 年	50～100k
超大型	100～1000	4～5 年	1M
极大型	2000～5000	5～10 年	1～10M

- 微型：可不做严格的系统分析和设计，在开发过程中应用软件工程的方法。
- 小型：如数值计算或数据处理问题，程序往往是独立的，与其他程序无接口，应按标准化技术开发。
- 中型：如应用程序及系统程序，存在软件人员之间、软件人员与用户之间的密切联

系、协调配合。应严格按照软件工程方法开发。

- 大型：如编译程序、小型分时系统、应用软件包、实时控制系统等。必须采用统一标准，严格复审，但由于软件规模庞大，开发过程可能出现不可预知的问题。
- 超大型：如远程通信系统、多任务系统、大型操作系统、大型数据库管理系统、军事指挥系统等。子项目间有复杂的接口，若无软件工程方法支持，开发工作不可想象。
- 极大型：如大型军事指挥系统、弹道防御系统等，这类系统极少见，更加复杂。

软件的复杂性即软件解决问题的复杂程度，主要按照应用程序，实用程序和系统程序的顺序由低到高排列。

（3）软件产品的开发所需时间

很明显软件产品开发时间越长成本越高。对确定规模和复杂度的软件存在一个“最佳开发时间”，也就是完成整个项目的最短时间，选取最佳开发时间来计划开发过程，可以取得最佳经济效益。

（4）软件开发技术水平

软件开发的技术水平主要指软件开发方法、工具、语言等，技术水平越高，效率越高。

（5）软件可靠性要求

一般在软件开发过程中可靠性要求越高，成本响应也就越高。因此一般根据软件解决问题的特点，要求合理的可靠性。

8.3.2 成本估算模型

1. 成本估算

软件开发成本主要是指软件开发过程中所花费的工作量及相应的代价，其中主要是人的劳动的消耗，因此，软件产品开发成本的计算方法不同于其他物理产品的成本的计算。

软件产品不存在重复制造过程，它的开发成本是以一次性开发过程所花费的代价来计算的。因此软件成本估算，应以软件计划、需求分析、设计、编码到测试等软件开发全过程所花费的代价为依据。

必须指出的是对于一个大型软件项目，由于其项目本身非常复杂，参与开发的人员众多，而且各子系统的问题难度、要求可能都有区别，因此软件项目的成本估算是一件复杂和困难的事，必须建立相应的估算模型，按照一定的方法、技术来进行估算。

无论采取哪种估算模型，当一个问题过于复杂时，可应将它进一步分解，以降低其问题的难度。可在估算每一个子问题的成本的基础上，再加以综合，最后得到整个软件项目的成本估算量。这是在进行成本估算中常用的“分而治之”的策略。

软件成本估算通常是估算软件的下列指标。

① 源代码行（LOC）估算。源代码行是指机器指令行/非机器语言的执行步，使用它们可以作为度量生产率的基本数据。

② 开发工作量估算。它是估算任何项目开发成本最常用的技术方法。根据项目开发过程，通常使用的度量单位是人月（PM）、人年（PY）或人日（PD）。

③ 软件生产率估算。它是指单位劳动量所能完成的软件数量，度量单位常用：LOC/PM，￥/LOC 或￥/PM。

同时，软件开发时间估算也是很重要的，在软件项目开发前必须进行估算。

软件成本估算模型可分为两大类：理论模型和统计模型。下面将具体介绍一些具有代表性的软件开发成本估算模型，如专家估算模型、IBM 估算模型和 Putnam 估算模型等。需要说明的是，还没有一种软件模型能够适用于所有软件类型和开发环境。

2. 专家估算模型

专家估算模型是由 Rand 公司提出的，又称为 Deiphi 技术，是由多位专家共同进行成本估算，这样就避免了单独一位专家的偏见。

该模型是由多位专家对软件的源代码行数反复进行估算的结果。例如，由 n 位专家进行成本估算，每位专家根据系统规格说明书，开会反复进行讨论，然后每位专家独立地进行估算，给出自己所认为的软件的最小源代码行数、最大源代码行数和最可能的源代码行数。并根据下列公式计算每位专家估算的源代码的期望值 L_i。

$$L_i = \frac{a_i + 4m_i + b_i}{6}, \quad L = \frac{1}{n}\sum_{i=1}^{n} L_i$$

式中，a_i 表示估计的最小行数，b_i 表示估计的最大行数，m_i 表示最可能的行数。之所以进行独立估算，是为了避免专家相互影响或因权威等因素产生的影响，造成估算结果的缺陷。

再根据每位专家所估算的期望值 L_i，计算第 1 次的期望中值 L。在此基础上，再反复召开小组讨论会，每位专家再反复独立地估算源代码行数的期望值 L_i，反复计算期望中值 L。

当前后两次计算的期望中值 L 之差的绝对值小于预先指定的小正数时，则取后一次的期望中值 L 为软件的源代码行数。通常以千行计。

至于预先指定的小正数，可以通过与历史资料进行类比确定，也可以根据软件项目的应用领域及性能要求确定。

推算出该软件的源代码行数后，参考这类软件的历史资料，推算出每行源代码所需成本，将估算的源代码行数，乘以推算出的每行源代码所需成本，就得到该软件的成本估算值。

3.IBM 估算模型

1977 年由 Walston 和 Felix 总结了 IBM 联合系统分部（FSD）负责的 60 个项目的数据。其中各项目的源代码行数 L 为 400～467000 行，开发工作量为 12～11758PM，共使用 29 种不同语言和 66 种计算机。利用最小二乘法拟合，得到如下估算公式（源代码行数 L，以千行计）：

工作量：　E=5.2　（PM）

项目持续时间：　D=4.1　（月）

人员需要量：　S=0.54　（人）

文档数：　DOC=49　（页）

IBM 模型利用已估算的特性（如源代码行数）来估算各种资源的需要量。模型一般是在可收集到足够有效的历史数据的局部环境中推导出来的。在处理过程中一般一条机器指令为一行源代码，源代码不包含程序的注释、作业命令和调试程序。对于非机器指令编写的源程序，

如汇编语言或高级语言程序，应通过：转换系数=机器指令条数/非机器语言执行步数，将其转换为机器指令源代码行数来考虑。

IBM 模型是一个静态单变量模型，但不是一种通用模型，因此应用中应根据实际情况调整模型中的参数。

4. Putnam 估算模型

这是 1978 年由 Putnam 提出的估算模型，该模型是一种动态多变量模型，它用于估算在软件开发的整个生存期中工作量的分布。根据一些大型项目（如 30 人年以上）中工作量的分布情况（如图 8.7 所示）推导出如下估算公式：

$$L = C_K K^{1/3} t_{\mathrm{d}}^{4/3}$$

式中，L 表示源代码行数，K 表示所需人力（PY），t_{d} 表示开发时间，C_K 表示技术水平常数。C_K 值与开发环境有关：对于差的开发环境，C_K =2000～2500；对于正常的开发环境，C_K =8000～10000；对于好的开发环境，C_K =11000～12500。

由上述公式可以得到所需开发工作量的公式：

$$K = L^3 C_K^{-3} t_{\mathrm{d}}^{-4} \quad （人年）$$

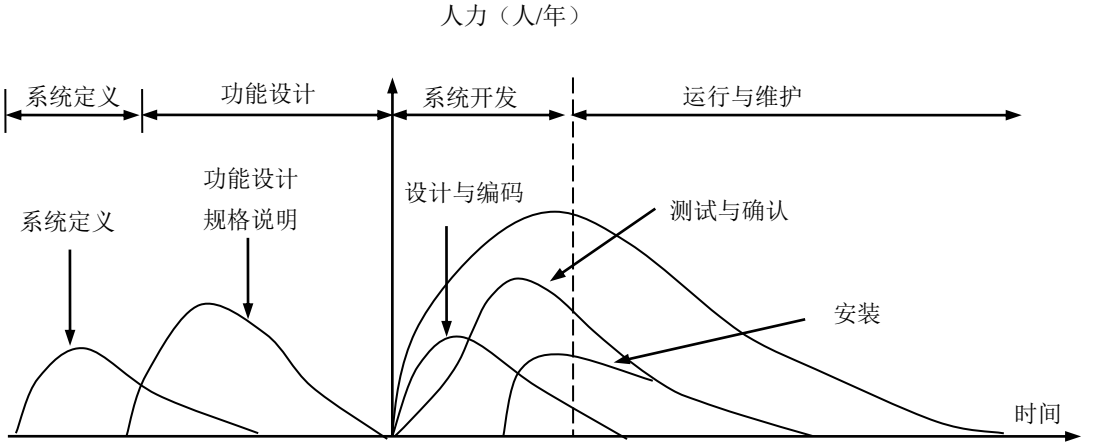

图 8.7 人力使用的分布

5. COCOMO 模型

由 TRW 公司开发的结构型成本模型（Constructive Cost Model，COCOMO 模型）是最精确、最易于使用的成本估算方法之一。它是由 Boehm 提出的结构型成本估算模型。

COCOMO 模型是一种层次模型，按照其详细程度分为以下三级。

①基本的 COCOMO 模型，它是一个静态单变量模型，对整个软件系统进行估算。

②中间的 COCOMO 模型，它是一个静态多变量模型，将整个软件系统分为系统和部件两个层次，系统由部件构成，它把软件开发所需的成本看成程序大小和一系列“成本驱动属性”的函数，用于部件级的估算，更为精确。

③详细的 COCOMO 模型，它将软件系统分为系统、子系统和模块三个层次，除包括中

级模型中所考虑的因素外，还考虑了在需求分析、软件设计等每一阶段的成本驱动属性的影响。

该模型主要对工作量 MM（单位：PM）和进度 TDEP（单位：月）进行估算，模型中考虑到估算量与开发环境有关，将开发项目分为以下三类：

①组织型（Organic）：相对较小、较简单的软件项目。程序规模不是很大（小于 5 万行），开发人员对产品目标理解充分，经验丰富，熟悉开发环境。大多数应用软件及老的操作系统、编译系统属于此种类型。

②嵌入型（Embedded）：此种软件要求在紧密联系的硬件、软件和操作的限制条件下运行，通常与某些硬件设备紧密结合在一起。因此，对接口、数据结构、算法要求较高。如大型复杂的事务处理系统，大型、超大型的操作系统，军事指挥系统，航天控制系统等。

③半独立型（Semidetached）：对项目要求介于上述两者之间，规模复杂度属中等以上，最大可达 30 万行。如大多数事务处理系统、新操作系统、大型数据库系统、生产控制系统等软件属此种类型。

下面分别讨论三级 COCOMO 模型：

●基本的 COCOMO 模型

其估算公式为：$\mathrm{MM} = C_l \times \mathrm{kloc}^a$

式中，MM 是工作量（PM），kloc 是估计的源代码行，C_l 是模型系数，a 是模型指数。C_l 、a 取决于开发项目的模式为组织型、半独立型或嵌入型。

表 8.2 是根据 63 个项目的数据统计结果，按照基本的 COCOMO 模型估算的工作量和开发进度的情况。

表 8.2 基本 COCOMO 模型的工作量和进度公式

总体类型	工作量	进度
组织型	$\mathrm{MM}=10.4(\mathrm{kloc})^{1.05}$	$\mathrm{TDEV}=10.5(\mathrm{MM})^{0.38}$
半独立型	$\mathrm{MM}=3.0(\mathrm{kloc})^{1.12}$	$\mathrm{TDEV}=10.5(\mathrm{MM})^{0.35}$
嵌入型	$\mathrm{MM}=3.6(\mathrm{kloc})^{1.20}$	$\mathrm{TDEV}=10.5(\mathrm{MM})^{0.32}$

●中间的 COCOMO 模型

进一步考虑了 15 种影响软件工作量的因素，可更加合理地估算软件工作量和进度等。将 15 种因素分为产品因素、计算机因素、人员因素和项目工程因素几类。下式描述了中间的 COCOMO 模型。

$$\mathrm{MM} = C_l \times \mathrm{kloc}^a \times \prod_{i=1}^{15} f_i$$

式中，f_i 是成本因素，内容如表 8.3 所示，表中列出了对 15 种影响软件工作量的因素，按等级打分的情况。

表 8.3　15 种影响软件工作量的因素 f_i 的等级分

工作量因素 f_i		非常低	低	正常	高	非常高	超高
产品因素	软件可靠性	0.75	0.88	1.00	1.15	1.40	
	数据库规模		0.94	1.00	1.08	1.16	
	软件复杂度	0.70	0.85	1.00	1.15	1.30	1.65
计算机因素	时间约束			1.00	1.11	1.30	1.66
	存储约束			1.00	1.06	1.21	1.56
	环境变更率		0.87	1.00	1.15	1.30	
	计算机换向时间		0.87	1.00	1.07	1.15	
人员因素	系统分析员能力		1.46	1.00	0.86		
	应用领域实际经验	1.29	1.13	1.00	0.91	0.71	
	程序员能力	1.42	1.17	1.00	0.86	0.82	
	开发人员环境知识	1.21	1.10	1.00	0.90	0.70	
	程序时间语言知识	1.41	1.07	1.00	0.95		
项目工程因素	设计技术	1.24	1.10	1.00	0.91	0.82	
	软件工具	1.24	1.10	1.00	0.91	0.83	
	进度限制约束	1.23	1.08	1.00	1.04	1.10	

•详细的 COCOMO 模型

详细 COCOMO 模型的名义工作量公式和进度公式与中间 COCOMO 模型相同。只是在考虑成本因素 f_i 时，按照开发阶段分别给出各层次更加详细的值。针对每个影响因素，按模块层、子系统层、系统层，有 3 张工作量因素分级表，供不同层次的估算使用。每一张表中工作量因素又按开发各个不同阶段给出。

8.3.3　成本/效益分析

成本/效益分析的目的是，从经济角度评价开发一个新的软件项目是否可行。其第一步是估算待开发系统的开发成本和运行费用（系统的操作费用和维护费用），与系统可能取得的效益（有形的和无形的）进行比较。系统的经济效益则等于因使用新系统而增加的收入，加上使用新系统可以节省的运行费用。以下介绍几种度量效益的模型：

（1）货币的时间价值

成本估算是要对项目投资。因投资先于取得效益，因此要考虑货币的时间价值。通常以利率形式表示。假设，年利率为 i，P 元钱在 n 年后的价值 F 为

$$F = P\,(1+I)^n$$

（2）投资回收期

投资回收期是衡量工程价值的经济指标，是指工程累计经济效益等于最初投资所需要的时间。投资回收期越短，就能越快地获得利润，由此这项工程就越值得投资。

（3）纯收入

工程的纯收入是衡量工程价值的另一项经济指标。将在整个生存周期内新系统的累计经

济效益与投资之差称为纯收入。

（4）投资回收率

投资回收率主要用于衡量投资效益的大小，并且可以用它和年利率比较，衡量工程是否有投资价值。设现在的投资额为

$$P = F_1/(1+j) + F_2/(1+j)^2 + \cdots + F_n/(1+j)^n$$

式中，F_i 是第 i 年年底的效益，i=1,2,3,$\cdots$,n；n 是系统的使用寿命；j 是投资回收率。

8.4 软件项目人力资源管理

现代的软件项目总是以团队的方式进行开发的，而团队中的人就是这个团队最大的财富，也是它最不确定的因素。这要求项目管理者要解决技术和非技术层面两方面的问题。管理者必须仔细规划和安排成员的工作，并时常鼓励他们，以保证项目的正常进行并导向成功。而糟糕的团队管理肯定会导致团队的混乱：成员分工不明确、互相猜疑指责，最后的结果是团队中只有一两位能干的成员在为团队做贡献，而其他人则明显地在“搭便车”。所以，必须对项目团队进行卓有成效的管理。

基于对人们工作动机的研究，专家将团队中的成员分为三类：

- 事业型：这类成员一心扑在工作上，他们可能成为团队中的技术骨干；
- 自我实现型：这类成员的主要动机是个人成功，他们感兴趣的是项目的成功是否能达到个人目标。
- 交际型：这里成员喜欢表达，并善于沟通。这类人员在团队中扮演着相当重要的角色，是成员关系的粘合剂、润滑油。

如何将这些不同个性的人员组织在一起是一项艰巨的工作，这里我们只是做一些简单的讨论。

8.4.1 软件团队建设

团队建设是现代软件开发中的一项关键任务，如何做好团队建设工作是所有项目管理者的首先应该考虑的问题。

所谓开发团队，很多人可能想到的就是技术人员的简单组合，团队利用人员各自的经验以期达到最终目标。然而事实证明，这样的组合也许根本无法完成任务。成功的团队应该具有一种团队精神（Team Spirit），团队的成功与个人的目标是一致的。因此，管理者需要对团队进行建设，而非简单地把人员召集在一起就行。

根据专家的建议，团队建设一般包含四项工作：

1. 团队构成

团队应该在成员的技能、经验和个性中找到一个平衡点。团队管理者应该善用三类人员：事业型的成为技术骨干；自我实现型的推动项目的进展；交际型的帮助内部交流。

可以看到，团队管理者是一个非常重要的角色。他们必须追踪每日的工作，保证项目的有效推进，并向更高级别的管理者汇报工作。

遴选团队负责人的常见方式是任用那个技术能力最强的成员。这意味着，他/她将肩负两项任务：技术领导和项目监管。这似乎是一种自然的、服众的和安全的策略。然而，一个不争的事实是，繁琐的日常管理会使这位成员分心，从而降低他们真正的价值。所以，很多时候将技术领导和项目监管这两项工作由不同的成员来担任将更有利于项目开展。

2. 团队凝聚力

一个好的团队的外在表现为：所有成员对团队的忠诚。忠诚带来凝聚，凝聚带来优势：

- 拥有内部标准
- 成员更紧密地工作在一起
- 成员能了解彼此的工作
- 做到无私编程（Egoless Programming）：程序是团队而非个人财产

团队凝聚力依赖于多项因素：文化性的、个性的等等。团队管理者可以用多种方式提升凝聚力，例如拓展活动、有家人参与的社会活动等。给予成员足够的信任、成员间的相互信任可以使成员有强烈的归属感，这也是提升凝聚力的关键。。

3. 团队交流和沟通

团队成员间的交流是项目开发过程中非常重要并且是必须的环节。成员间没有沟通而各自为战将会为项目带来灾难性的后果。良好的沟通能使成员互相了解各自的开发状态，从而避免不好结果的发生，同时还能提升凝聚力。

可能会影响沟通的因素有：

- 团队规模。团队越大，团队成员间的单向沟通途径会急剧增加（如图 8.8 所示），成员间的充分沟通也就越困难。所以，团队的规模在 3-5 人为宜。如果项目较大，可以采用层级的方式组织人员。

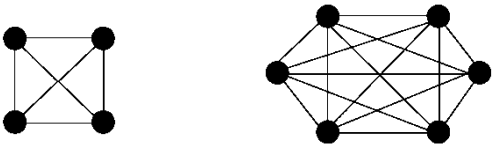

图 8.8 团队成员从 4 名增加到 6 名后的沟通路径数量变化

- 团队结构。团队结果往往采用层级结构，这意味着信息的向下广播。当团队规模较大时，这可能导致信息延迟甚至被误解。实践显示，非正式结构的交流也许比正式层级结构的交流更有效。
- 团队构成。团队中相同/相近个性或同性别的成员越多，冲突就越多，交流也就越困难。因此，管理者应考虑在团队中加入不同个性或性别的成员。团队中的女性成员更倾向于是交际型的，她们的存在有利于团队的建设和发展，正所谓：“男女搭配，干活不累”。
- 工作环境。团队成员的工作环境应该是利于交流的。例如：尽量将成员的办公位置集中在一起、降低办公桌隔断高度、设置公共会议/休息/阅览空间等。

4. 团队组织结构

团队内部人员的组织形式对生产率有影响。一般常有的组织形式有主程序员制、民主制和层次式团队三重方式。如图 8.9 所示。

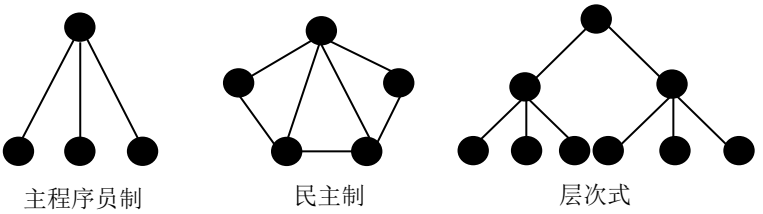

图 8.9 组织形式

8.4.2 团队人员的选留

如何合理地选留团队人员是成功完成软件项目的切实保证。所谓合理选留团队人员，应该包括：按不同阶段适时任用人员，恰当掌握人员标准。

1. 人员配备遵循的原则

人员配备主要遵循三个原则：

① 重质量：软件项目是技术性很强的工作，对于关键性的任务任用少量有能力和经验的人员去完成。

② 重培训：必须花费精力培养所需的技术人员和管理人员。

③ 双阶梯提升：人员的提升应分别按技术职务和管理职务进行。

2. 选择人员的途径

在一个项目中，人员的来源途径一般有三种：

① 候选人自荐。通过他们的简历或 CV 可以粗略了解候选人的背景和经历。

② 面试。通过面试可以获得候选人更直观的信息。

③ 他人推荐。通过曾和候选人共事过的人员推荐，也可以获得相应的信息。

其他任何合适的方式也可以被采用。一个需要把握的原则是：选合适的人放在合适的位置上。

3. 选择/评价人员的条件

软件项目对人的因素越来越重视。在评价和任用软件人员时，必须掌握一定的标准。人员素质的优劣会直接影响到项目的成败。评价的主要条件是：

① 应用领域经验：具备应用领域的基础知识；

② 开发经验：善于分析和综合问题，具有严密的逻辑思维能力；

③ 教育背景：能在一定程度上展示候选人的基础。需要说明的是：在现代的项目中，只选择那些具有计算机专业背景的人员是不够的，特别是在跨专业、行业的项目中；

④ 适应能力：善于听取意见，善于团结协作，有良好的人际关系；

⑤ 沟通能力：具有良好的书面和口头表达能力，掌握多媒体演示工具；
⑥ 工作态度：工作踏实、细致，不靠运气，遵循标准和规范，具有严格的科学作风；
⑦ 个性：有耐心、有毅力、有责任心。

4. 团队成员的去留

任何一个团队的管理者都会面临如何留住那些骨干成员的问题。团队成员会因个人原因、外部原因等各种因素离开团队，特别是骨干成员的离开会对项目开发造成致命的打击。规范化的工作流程、严谨的工作制度也许会在一定程度上降低风险，但管理者应该意识到，在项目开发期间保持团队和人员的稳定是压倒一切的。所以，管理者应当做到：

① 尊重成员的个人选择，不人为设置各种障碍；
② 满足成员合理的个人需求。较低层次的做法是加薪、升职，高一点的是帮助规划个人发展愿景，最高层次是提供个人自我实现的平台；
③ 满足成员的社会需求。保持或提升成员的社会认可度，包括树立良好的社会形象、提升社会地位等；
④ 加大团队建设的力度。营造和谐的团队文化，增强凝聚力，提高成员的忠诚度；
⑤ 要有“剜肉剔骨”的决心。团队中总会有 Black Sheep。如果他们无改善计划，或者在规定期限内无改善进展，为团队注入了负能量，那么无论他们多么重要，也要将其坚决清退。

总之，管理者需要把握的一个原则是：情感留人、待遇留人、事业留人。

8.5 项目风险管理

近年来软件开发技术、工具都有了很大的进步，但是软件项目开发超时、超支、甚至不能满足用户需求而根本没有得到实际使用的情况仍然比比皆是。软件项目开发和管理中一直存在着种种不确定性，严重影响着项目的顺利完成和提交。因此，对软件风险的研究、管理已经成为软件项目管理的重要内容。

8.5.1 软件项目风险管理概述

1. 风险的定义

风险（Risk）是一种潜在的危险。软件项目由于其自身的特点而存在风险，甚至是灾难性的风险。风险管理即是预测、控制和管理项目风险。

20 世纪 80 年代，Boehm 首先对软件开发中的风险进行了论述，并提出软件风险管理的方法。Boehm 认为，软件风险管理指的是“试图以一种可行的原则和实践，规范化地控制影响项目成功的风险”，其目的是“辨识、描述和消除风险因素，以免它们威胁软件的成功运作”。

软件项目风险不仅会影响项目计划的实现，影响项目的进度，增加项目的成本，甚至使软件项目不能实现。因此风险管理决定了软件项目的成败。

所以，软件项目的风险管理是软件项目管理的重要内容。在进行软件项目风险管理时，要辩识风险，评估它们出现的概率及产生的影响，然后建立一个规划来管理风险。

2．软件项目中的风险

风险管理的主要目标是预防风险。软件项目的风险无非体现在以下四个方面：需求、技术、成本和进度。项目开发中常见的风险有如下几类：

(1) 需求风险

引起需求的风险可能有以下几种情况：

① 需求过程中由于客户参与不够，因此需求定义不完整，或者有二义性；

② 需求的继续不断变化，又缺少有效的需求变化管理过程。

(2) 计划编制风险

① 计划、资源和产品定义缺乏有效依据，全凭客户或上层领导口头指令,并且不完全一致；

② 虽然制订相应计划，但计划不现实；

③ 产品规模比估计的要大，但又没有相应地调整产品范围或可用资源；

④ 涉足不熟悉的产品领域，花费在设计和实现上的时间比预期的要多。

(3) 组织和管理风险

① 仅由管理层或市场人员进行技术决策，管理层审查、决策的周期比预期的时间长，导致计划进度缓慢，计划时间延长；

② 低效的项目组结构降低生产率；缺乏必要的规范，导至工作失误与重复工作；

③ 非技术的第三方的工作(预算批准、设备采购批准、法律方面的审查、安全保证等)时间比预期的延长或者预算削减，打乱项目计划。

(4) 人员风险

① 开发人员和管理层之间关系不佳，影响全局；项目组成员之间发生冲突，沟通不畅，导致设计、接口出现错误；

② 某些开发人员不熟悉的软件工具和环境，或项目后期加入新的开发人员，使工作效率降低；

③ 缺乏项目急需的具有特定技能的人，又缺乏激励措施，士气低下，降低了生产能力。

(5) 其他风险

① 设施、工具等不具备的“开发环境风险”；

② 客户对于最终产品不符合用户需求而要求重新设计开发的“客户风险”；

③ 质量低劣的“产品风险”；

④ 设计质量低下，有些必要的功能无法使用现有的代码和库实现的“设计和实现风险”；

⑤ 在执行过程中出现的“过程风险”。

8.5.2 软件项目风险管理过程

风险管理包括：风险识别、风险估算、风险评价、风险监控和管理。

1．风险识别

识别风险是系统化地识别已知的和可预测的风险，在可能时避免这些风险，且当必要时控制这些风险。识别潜在的风险，是进行项目风险管理的基础。风险识别包括确定风险的来源，风险产生的条件，描述其风险特征和确定哪些风险事件有可能影响本项目。风险识别不是一次就可以完成的事，应当在项目开发过程的自始至终定期进行。

根据风险内容，通常可分别对以下三类风险进行提取和分析。

（1）项目风险(Project)。与项目有关的预算、进度、人力、资源、用户需求、项目规模、复杂性等方面的问题，都属于这类风险。

（2）技术风险（Technical)。是指影响开发质量和交付时间的设计、实现、验证、维护、接口等方面的问题。

（3）商业风险(Business)。包括与产品的商业运作有关的市场风险、预算风险、决策风险、销售风险等。

识别风险的方法和工具有多种，这里介绍一种由 Keil.M.等人总结的识别风险的提问单：

① 最终用户对该项目和待构造的系统支持吗？

② 需求已经被软件项目组和用户完全理解吗？

③ 软件开发机构的高层管理者和用户方的管理者已正式承诺支持该项目吗？

④ 用户已充分参加到需求定义中了吗?

⑤ 最终用户的期望实现了吗？

⑥ 项目的需求和工作稳定吗？

⑦ 软件项目组拥有合适的技能吗？

⑧ 项目组对所需开发技术有经验吗？

⑨ 项目组人员数量能够完成此项目吗？

⑩ 所有用户对该项目的重要性和系统需求有共识吗？

通过回答提问单的问题，识别风险。

在进行具体的软件项目风险识别时，可以根据实际情况对风险分类。但简单的分类并不是总行的通的，某些风险根本无法预测。在这里，我们介绍一下美国空军软件项目风险管理手册中指出的如何识别软件风险。这种识别方法要求项目管理者根据项目实际情况来标识影响软件风险因素的风险驱动因子，这些因素包括以下几个方面。

① 性能风险：产品能够满足需求和符合使用目的的不确定程度。

② 成本风险：项目预算能够被维持的不确定的程度。

③ 支持风险：软件易于纠错、适应及增强的不确定的程度。

④ 进度风险：项目进度能够被维持且产品能按时交付的不确定的程度。

每一个风险驱动因子对风险因素的影响均可分为四个影响类别——可忽略的、轻微的、严重的及灾难性的。

2．风险估算

在进行了风险辨识后，就要进行风险估算，风险估算也称为风险评估，一般是从两方面进行估算：

① 从影响风险的因素考虑风险发生的可能性，即风险发生的概率。

② 风险发生所带来的损失的严重程度，评价如果风险一旦发生所产生的后果。

需要强调的是如何评估风险的影响，如果风险真的发生了，它所产生的后果会对三个因素产生影响：风险的性质、范围及时间。风险的性质是指当风险发生时可能产生的问题。风险的范围是指风险的严重性及其整体分布情况。风险的时间是指主要考虑何时能够感到风险及持续多长时间。另外，评估每一个风险，以确定新的情况是否引起风险的概率及影响发生改变。

为了反映风险产生的可能程度和风险产生后果的严重程度，需要建立风险度量的指标体系。如一种简单的风险评估技术是建立如表 8.4 的风险评估表。

从表中可见，按照风险产生后果的严重程度分为灾难性的、严重的、轻微的和可忽略的 4 类。从性能、支持、成本和进度四方面对风险进行评估，表中给出了这 4 方面的评估标准，综合考虑可以确定所产生风险的严重程度。

表 8.4　风险评估表

类别 / 成本		性能	支持	成本	进度
灾难性的	1	无法满足需求而导致任务失败		错误导致成本增加，资金短缺超出预算	
	2	性能严重下降，达不到技术要求	无法响应或无法支持的软件	资金严重短缺，很可能超出预算	无法按期交付完成
严重的	1	无法满足需求而导致系统性能下降，任务能否完成受到质疑		错误导致运行延迟和成本增加	
	2	技术性能有所下降	在软件修改中有所延后	资金不足，可能超支	交付日期可能延后
轻微的	1	不能满足需求而导致次要任务性能下降		对成本和进度都有影响	
	2	技术性能稍微降低	能相应软件支持	有较充足的资金来源	计划进度可完成
可忽略的	1	无法满足需求而导致使用不方便或操作不易		错误对成本和进度影响不大	
	2	技术性能不会减低	易于软件支持	可能低于预算	交付日期可能提前

3．风险评价

风险评价是在风险估算的基础上，对所确定的风险做进一步的确认。定义项目的风险参考水准，进一步验证风险评估结果的准确性，并按照风险发生概率高低和后果严重的程度进行排序。一般可定义成本、性能和进度作为三个典型的参考量。

进行风险评价，通常由下列三元组的形式描述：

(ri, li, xi)　　i=1,2,3,….l

其中：ri,为风险， li,为风险发生的概率， xi 为风险发生后的影响。i 为风险的种类。

图 8.10 描述了受成本超支和进度延迟影响的风险参考水准，给出了受这两个因素影响的参考点，达到参考点，将造成项目终止。

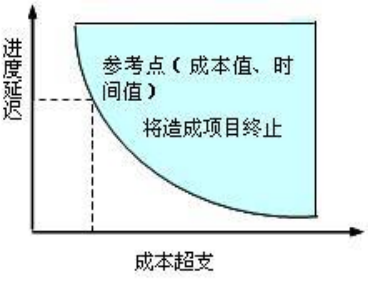

图 8.10　风险参考水准

4．风险监控和管理

一个有效的策略必须考虑风险避免、风险监控和风险管理及意外事件计划这样三个问题。风险的策略管理可以包含在软件项目计划中，或者风险管理步骤也可以组成一个独立的风险缓解、监控和管理计划。

(1) 避免风险

是一种主动避免风险的活动。是在风险发生前分析引起风险的原因，采取措施，避免风险发生。

(2) 风险监控

软件开发是高风险的活动。如果项目采取积极风险管理的方式，就可以避免或降低许多风险，而这些风险如果没有处理好，就可能使项目陷入瘫痪中。因此在软件项目管理中要进行风险跟踪。对已识别的风险在系统开发过程中进行跟踪管理，

风险监控贯穿在软件开发的全过程，是一种项目跟踪活动。主要监控对项目风险产生主要影响的因素，并随时记录项目的执行情况，确定还会有哪些变化，以便及时修正计划。

(3) 风险管理监控计划

制订风险监控计划 RMMP（Risk Management and Monitoring Plan），保证文档的正确性，按监控计划记录、管理风险分析的全过程。

RMMP 计划将所有风险分析工作文档化，并且由项目管理者作为整个项目计划的一部分来使用，RMMP 计划的大纲主要包括：主要风险，风险管理者，项目风险清单，风险缓解的一般策略、特定步骤，监控的因素和方法，意外事件和特殊考虑的风险管理等。这些相应的解决方案和措施，以便在发生风险时能够主动应对。

8.5.3 风险管理的理论和模型

讨论风险管理中，常用的经典的理论和模型。

1．Boehm 模型

对风险管理，Boehm 基本沿袭了传统的项目风险管理理论，指出风险管理由风险评估和风险控制两大部分组成，风险评估又可分为识别、分析、设置优先级 3 个子步骤，风险控制则包括制订管理计划、解决和监督风险 3 步。

Boehm 用以下模型对风险进行定义：

$$RE=P(UO)*L(UO)$$

其中：RE 表示风险或者风险所造成的影响，P(UO)表示令人不满意的结果所发生的概率，L(UO)表示糟糕的结果产生的破坏性程度。

Boehm 理论的核心是基于包括人员短缺、不合理的进度安排和预算、不断的需求变动等的 10 大风险因素列表，该列表是通过对美国几个大型航空或国防系统软件项目的深入调查，编辑整理而成的，因此有一定的普遍性和实用性。

针对每个风险因素，Boehm 都给出了一系列的风险管理策略。且将管理层的注意力有效地集中在高风险、高权重、严重影响项目成功的关键因素上，而不需要考虑众多的低优先级

的细节问题。

该理论存在一些不足，没有清晰明确地说明风险管理模型到底要捕获哪些软件风险的特殊方面，因为列举的风险因素会随着风险管理方法而变动，同时也会互相影响。这就是说风险列表需要改进和扩充，管理步骤也需要优化。

2．CRM 模型

该模型由 SEI（Software Engineering Institution）提出，作为世界上著名的旨在改善软件工程管理实践的组织，提出了持续风险管理管理模型 CRM（Continuous Risk Management）。

CRM 的基本思想是：不断地评估可能造成恶劣后果的因素；决定最迫切需要处理的风险；实现控制风险的策略；评测并确保风险策略实施的有效性。

CRM 模型要求在项目生命期的所有阶段都关注风险识别和管理，它将风险管理划分为 5 个步骤：风险识别、分析、计划、跟踪、控制。框架显示了应用 CRM 的基础活动及其之间的交互关系，强调了这是一个在项目开发过程中反复持续进行的活动序列。每个风险因素一般都需要按顺序经过这些活动，但是对不同风险因素开展的不同活动可以是并发的或者交替的。

3．Leavitt 模型

SEI 和 Boehm 的模型都以风险管理的过程为主体，研究每个步骤所需的参考信息及其操作。而 1964 年由 Aalborg 大学提出的 Leavitt 模型则不同，其基本思想是着重从导致软件开发风险的不同角度出发探讨风险管理。

Leavitt 模型将形成各种系统的组织划分为 4 个组成部分：任务、结构、角色和技术。这 4 个组成部分分别代表了软件开发的各因素。

- 角色：表示所有的项目参与者，例如软件用户、项目经理和设计人员等；
- 结构：表示项目组织和其他制度上的安排；
- 技术：包括开发工具、方法、硬件软件平台；
- 任务：描述了项目的目标和预期结果。

Leavitt 模型的关键思路是：模型的各组成部分是密切相关的，一个组成部分的变化会影响其他的组成部分，即一个系统开发过程中任何 Leavitt 组成成分的修改都会产生一些问题，可能导致风险发生，甚至导致软件修改的失败。

因此，使用 Leavitt 模型从 4 个方面分别识别和分析软件项目的风险是极有条理性和比较全面的。在进行软件项目管理时，可以采用不同的方法对不同的方面进行风险管理。

Leavitt 模型实际上是提出一个框架，可以更加广泛和系统地将软件风险的相关信息组织起来。Leavitt 理论的设计方法和实现研究已经广泛应用于信息系统中，它所考虑的都是软件风险管理中十分重要的环节，而且简单、定义良好、适用于分析风险管理步骤。

8.6 软件质量保证

软件质量反映了软件的本质。软件质量是一个软件企业成功的必要条件，其重要性怎样强调都不过分。而软件产品生产周期长，耗资巨大，如何有效地管理软件产品质量一直是软

件企业面临的挑战。由于软件质量具有难以进行定量度量的属性，这里主要从管理的角度讨论影响软件质量的因素。

我们把影响软件质量的因素分成三组，分别反映用户在使用软件产品时的三种不同倾向或观点。这三种倾向是：产品运行、产品修改和产品转移，如图 8.11 所示。

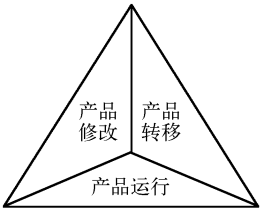

图 8.11 软件质量模型

8.6.1 软件质量因素

软件质量因素及其定义如表 8.5 所示。

表 8.5　软件质量因素及其定义

质量因素		定义
产品运行	正确性	系统满足规格说明和优化目标的程度，即在预定环境下能正确地完成预期功能的程度
	健壮性	在硬件故障、操作错误等意外情况下，系统能作出适当反应的程度
	效率	为完成预定功能，系统需要的计算资源的多少
	完整性	即安全性，对非法使用软件或数据，系统能够控制（禁止）的程度
	可用性	对系统完成预定功能的满意程度
	风险性	能否按照预定成本和进度完成系统看法，并为用户满意的程度
产品修改	可理解性	理解和使用该系统的容易程度
	可修性	诊断和改正运行时发现的错误所需工作量的大小
	灵活性	即适应性，修改或改进正在运行的系统所需工作量的大小
	可测试性	软件易测试的程度
产品转移	可移植性	改变系统的软、硬件环境及配置时，所需工作量的大小
	可重用性	软件在其他系统中可被再次使用的程度（或范围）
	互运行性	把该系统与另一个系统结合起来所需工作量

8.6.2 软件质量保证工作

进入 20 世纪 80 年代，软件质量问题开始被大家重视。具体可以从两个方面来理解软件质量保证工作。

一方面从顾客驱动观点看，注重于复审和校正的方法并保证一致性，其关键是需要一种客观的标准来确定并报告软件开发过程及其成果的质量，一般由“软件质量保证小组”完成。

另一方面，从管理者驱动观点看，注重于确定为了产品质量必须做些什么，并且建立管理和控制机制来确保这些活动能够得到执行。关键步骤如下：

① 以客户对于质量的需求为基础，对项目开发周期的各个阶段，建立质量目标。

② 定义质量度量（Metrics）。以衡量项目活动的结果，协助评价有关的质量目标是否达到。

③ 确定质量活动。对于每一个质量目标，确定那些能够帮助实现该质量目标的活动，并将这些活动集成到软件生命周期模型中去。

④ 执行已经确定的质量活动。

⑤ 评价质量。在项目开发周期的各阶段，利用已经定义好的质量度量来评价有关的质量目标是否达到。

⑥ 若质量目标没有达到，采取修正行动。

8.6.3 软件项目的跟踪与控制

在软件项目实施过程中进行跟踪与控制，是软件项目管理的重要内容，也是保证软件质量的重要措施。可根据具体情况采用不同的方法进行追踪。

软件度量和保证的条件通常包括：适应性、易学性、可靠性、针对性、客观性和经济性。

软件质量度量方法有以下三种：

① 精确度量。使用质量度量评价准则进行详细度量，工作量大，但度量精确度也高。

② 全面度量。可以与简易度量并用，对各个质量设计评价准则进行度量，工作量可以控制在一定的范围内。

③ 简易度量。顾名思义，就是对各个质量评价进行简单的工作量度量。

8.7 软件配置管理

软件产品不是静态的、一成不变的系统，它会随着时间的推移而演化，从而产生多个版本。这些版本包含对错误的修订，以及对不同软硬平台的适应。另一种可能是在开发过程中就存在多个并行的版本。所以，管理者必须对软件进行配置管理。

8.7.1 软件配置管理的基本概念

软件配置管理（Configuration Management，CM）应用于整个软件工程过程，是一种标识、组织和控制变更/修改的技术。开发过程中软件的变更是不可避免的，而变更可能加剧项目开发者之间的混乱。CM 活动的目标就是为了标识变更、控制变更、确保变更正确实现并向其他有关人员报告变更，其目的是使错误率降到最低并最有效地提高生产效率。

软件配置管理常用到 CM 工具。它们被用于保存系统的不同版本、整合系统以及跟踪用户版本（Release Version）。

软件配置管理有时被认为是软件质量管理的一部分。当开发团队将软件交付给质量保证团队后，后者负责检查系统的质量，然后将受检后的系统交付给配置管理团队去控制软件的变更。受控的系统有时称为“基线系统（Baselines）”，是可控演化的起点。

配置管理过程以及相关的文档应当遵循严格的标准，例如 IEEE 828-1983、ISO 9000 或者 CMM。但无论如何，为了确保软件的质量，开发团队必须形成内部的正式 CM 标准。

图 8.12 示意了一个典型的 CM 过程。

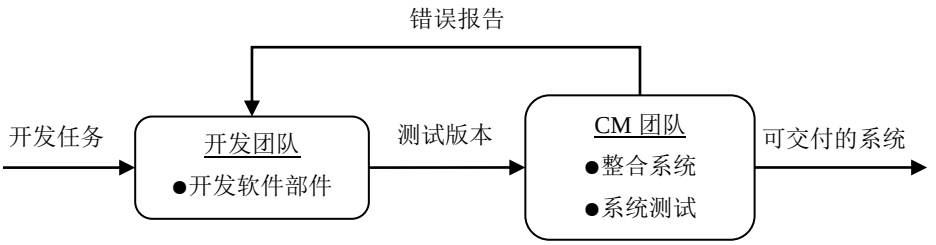

图 8.12 典型的 CM 过程

图示的过程是一个迭代过程：开发团队每次交付给 CM 团队的系统都是基于前一个的修改版本。在增量开发模式中，交付的除了已完成的部件外，可能还会包含一些未完成但已具备基本功能并能用于测试的软件部件；在 CM 团队进行测试的同时，开发团队可以并行地为那些未完成的部件添加功能。这个过程将反复迭直到整个系统完成并被测试。

8.7.2 软件配置管理的活动

软件配置管理的关键活动包括：制订配置计划、配置项标识、变更控制、版本控制、系统整合等。

1. 配置计划（Configuration Management Planning）

配置计划详细描述了配置管理过程中的标准和过程。该计划应当是通用的，由开发机构高层制订的，并能有适应于该机构的每一个项目，其中应当包括：

① 被管理的软件实体的定义和标识实体的正式框架；
② 任命负责实施配置管理过程以及向管理团队提交软件实体的人员；
③ 用于变更控制和版本控制的策略；
④ 对管理过程文档的描述；
⑤ 对管理工具的描述；
⑥ 对管理数据库的定义。

其中，一个重要的部分是任命负责人。除了上面提到的两名负责人外，可能还会确定文档的复检人（Reviewer）。在常见的人员配置结构中，项目经理或者团队负责人将担负起这些责任。

2. 配置项标识（Configuration Item Identification）

一个大型的软件项目会产生相当多的、会经常性和规律性变更的文档。这些文档或相关文档的分组称为“配置项（Configuration Item）”。

标识配置项应当仔细规划。所有受控的配置项都应当采用统一的、唯一的命名方式。这些文档之间的关系也必须用某种方式明确地标识出来。常用的方式是采用树形结构，这种结构能非常清晰地展示项目之间的关系。

与之配套的设施是配置数据库。配置数据库用于保存于配置相关的信息，它的主要功能是帮助解决系统变更的冲突以及提供管理信息。在该数据库中，应该能够查询到诸如系统运行的软硬平台、系统创建了多少个版本、某个版本的错误信息等等。

3. 变更控制（Change Control）

软件一定会因某些原因变更。因此，为了让变更能被平滑处理，在变更之前，应当提交一份变更提案表（Change Request Form，CRF），该表包含如下内容：

(1) 项目名称/编号
(2) 变更提案人/日期
(3) 变更内容
(4) 变更分析人/分析日期
(5) 变更影响的部件
(6) 变更评估
(7) 变更优先级
(8) 变更实现
(9) 预估成本
(10) 提交到 CCB 的日期/CCB 裁决日期
(11) CCB 的裁决
(12) 变更实现人/变更日期
(13) 提交 QA 日期/QA 裁决
(14) 提交 CM 日期
(15) 注释

上面内容中提到的 CCB 是变更控制委员会（Change Control Board）的缩写。CCB 可以根据项目的规模灵活组建。

图 8.13 是变更控制的流程框图。

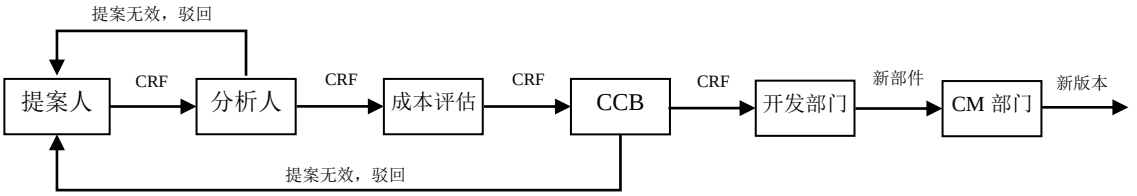

图 8.13 变更控制的流程

提案无效的原因有多种，例如：变更已被处理、变更将导致成本增加等等。

4. 版本控制（Version & Release Control）

版本控制用于标识和跟踪系统开发过程中产生的不同版本。术语 Version 和 Release 都被翻译为“版本”，但二者有所不同：

- 一个系统的 Version 是该系统开发过程中的一个与先前产生的不同实例（Instance）；
- 一个系统的 Release 是交付给用户的一个 Version。基于此，Release 一词也常被翻译成“发行版”或“发行”。

现代的版本控制总是借助于 CASE 工具来实现的。CASE 工具往往包含一个数据库，开

发人员从库中借出（Check Out）系统部件并对其进行编辑，然后再将编辑好的部件归还（Check In）到库中。此时，版本控制系统会创建一个新的版本，该版本被赋予唯一标识。

(1) 版本标识

版本标识的主要技术为版本编号（Version Numbering）。图 8.14 是版本编号的一个示例。

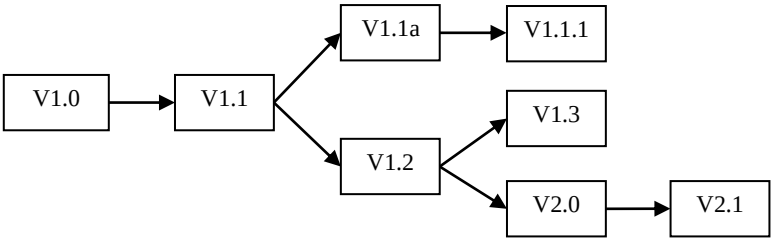

图 8.14 版本编号变迁

(2) 发行版管理

发行版面向的是最终客户。因此，一个系统的发行版绝不仅仅是一套可执行代码，它还应该包括：

- 系统配置文件：用于定义安装配置；
- 数据文件：安装程序用到的数据，包括系统的可执行代码；
- 安装程序：运行此程序可以将系统安装到指定的目标软硬平台上；
- 纸质或电子文档：系统的功能性描述文档，供用户阅读；
- 与发行版相关的其他信息包。

在向用户交付发行版时，不能使现行版本依赖于旧版本。否则，应该交付的是一个升级、修复或者更新版本。

此外，软件产品的发布日期、发行介质等也是需要考虑的因素。发布日期主要基于市场营销方面的考虑。发行介质以前主要是 CD-ROM 或者 DVD，而现在主要以网络下载为主。特别地，在云计算模式下，软件产品的发行方式又将迎来一次革命性的变迁。

5. 系统整合（System Building）

系统整合的任务是将所有系统部件组装成一套可以在特定目标平台上运行的完整系统。因此，系统整合需要如下保障因素：

- 所有系统需要的软件部件；
- 整合系统所需的适合版本；
- 所需的数据文件；
- 适合的整合工具，例如适合版本的编译器。

目前，自动化的 CM 工具被用于系统整合。CM 团队会编写一个整合脚本(Build Script)，其中定义了各系统部件之间的依赖关系，指定了所需的整合工具。系统整合工具凭此整合脚本驱动，将部件组装成最后的完整系统。图 8.15 是系统整合的流程示意图。

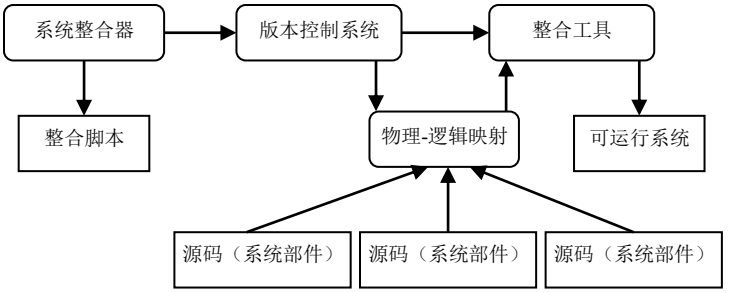

图 8.15 是系统整合流程

图中，物理-逻辑映射部件用于将系统部件源码的物理存储映射成不依赖于物理结构的逻辑结构。在使用编译型语言编写系统部件的情况下，整合工具包括依赖于目标软硬平台（例如操作系统）的编译器和链接器，它们可以将源码转换为依赖于目标平台的可执行代码。目前，越来越多的软件系统（例如 Web 应用）使用非编译型的语言（例如 PHP 这样的解释型脚本语言）开发。这些系统的部件以源码的形式存在，运行时不需要编译和链接，而是依赖于一个适合的解释器或虚拟机，以及运行时支撑平台。

8.8 企业资源规划

企业资源规划（Enterprise Resource Planning，ERP）系统，是指建立在信息技术基础上，以系统化的管理思想，为企业决策层及员工提供决策运行手段的管理平台。ERP 系统集信息技术与先进的管理思想于一身，成为现代企业的先进运行模式，反映时代对企业合理调配资源，最大化地创造社会财富的要求，成为企业在信息时代生存、发展的基石。

8.8.1 资源管理发展过程

计算机技术特别是数据库技术的发展为企业建立管理信息系统，甚至对改变管理思想起着不可估量的作用，管理思想的发展与信息技术的发展相互依赖，相互促进。而实践证明信息技术已在企业的管理层面扮演越来越重要的角色。

信息技术最初在管理上的运用，也是十分简单的，主要是记录一些数据，方便查询和汇总。现在已发展到建立在全球 Internet 基础上的跨国家、跨企业的运行体系，可分为如下几个阶段：

（1）MIS 系统阶段（Management Information System）

企业的信息管理系统主要是记录大量原始数据、支持查询、汇总等方面的工作。

（2）MRP 阶段（Material Require Planning）

企业的信息管理系统对产品构成进行管理，借助计算机的运算能力及系统对客户订单、在库物料、产品构成的管理能力，实现依据客户订单，按照产品结构清单展开并计算物料需求计划、实现减少库存，优化库存的管理目标。

（3）MRP II 阶段（Manufacture Resource Planning）

在 MRP 管理系统的基础上，系统增加了对企业生产中心、加工工时、生产能力等方面的

管理，以实现计算机进行生产排程的功能，同时也将财务的功能囊括进来，在企业中形成以计算机为核心的闭环管理系统，这种管理系统已能动态监察到产、供、销的全部生产过程。

（4）ERP 阶段（Enterprise Resource Planning）

进入 ERP 阶段后，以计算机为核心的企业级的管理系统更为成熟，系统增加了包括财务预测、生产能力、调整资源调度等方面的功能，配合企业实现 JIT（Just In Time，准时制）管理、全面质量管理和生产资源调度管理及辅助决策的功能，成为企业进行生产管理及决策的平台工具。

（5）电子商务时代的 ERP

Internet 技术的成熟为企业信息管理系统增加了与客户或供应商实现信息共享和直接的数据交换的能力，从而强化了企业间的联系，形成共同发展的生存链，体现企业为达到生存竞争的供应链管理思想。ERP 系统相应实现这方面的功能，使决策者及业务部门实现跨企业的联合作战。

由此可见，ERP 的应用的确可以有效地促进现有企业管理的现代化、科学化，适应竞争日益激烈的市场的要求，它的导入，已经成为大势所趋。

8.8.2 ERP 系统的管理思想

从管理思想、软件产品、管理系统三个层次进一步给出 ERP 的定义：

① ERP 是由美国著名的计算机技术咨询和评估集团 Garter Group Inc.提出的一整套企业管理系统体系标准，其实质是在 MRP Ⅱ基础上进一步发展而成的面向供应链（Supply Chain）的管理思想；

② ERP 是综合应用了客户-服务器体系、关系数据库结构、面向对象技术、图形用户界面、第四代语言（4GL）、网络通信等信息产业成果，以 ERP 管理思想为核心的软件产品；

③ ERP 是整合了企业管理理念、业务流程、基础数据、人力物力、计算机软硬件于一体的企业资源管理系统。

从以上定义可以看出，ERP 的核心管理思想就是实现对整个供应链的有效管理，主要体现在以下三个方面：

（1）体现对整个供应链资源进行管理的思想

在知识经济时代仅靠自己企业的资源不可能有效地参与市场竞争，还必须把经营过程中的有关各方，如供应商、制造工厂、分销网络、客户等纳入一个紧密的供应链中，才能有效地安排企业的产、供、销活动，满足企业利用全社会一切市场资源快速高效地进行生产经营的需求，以期进一步提高效率和在市场上获得竞争优势。ERP 系统实现了对整个企业供应链的管理，适应了企业在知识经济时代市场竞争的需要。

（2）体现精益生产、同步工程和敏捷制造的思想

ERP 系统支持对混合型生产方式的管理，其管理思想表现在两个方面：

一是“精益生产（Lean Production）”的思想，它是由美国麻省理工学院（MIT）提出的一种企业经营战略体系。即企业按大批量生产方式组织生产时，把客户、销售代理商、供应商、协作单位纳入生产体系，企业同其销售代理、客户和供应商的这种合作伙伴关系组成了一个

企业的供应链。这即是精益生产的核心思想。

二是“敏捷制造（Agile Manufacturing）”的思想。当市场发生变化，企业遇有特定的市场和产品需求时，企业的基本合作伙伴不一定能满足新产品开发生产的要求，这时，企业会组织一个由特定的供应商和销售渠道组成的短期或一次性供应链，形成“虚拟工厂”，把供应和协作单位看成是企业的一个组成部分，运用“同步工程（SE）”，组织生产，用最短的时间将新产品打入市场，时刻保持产品的高质量、多样化和灵活性。这即是“敏捷制造”的核心思想。

（3）体现事先计划与事中控制的思想

ERP 系统中的计划体系主要包括：主生产计划、物料需求计划、能力计划、采购计划、销售执行计划、利润计划、财务预算和人力资源计划等，而且这些计划功能与价值控制功能已完全集成到整个供应链系统中。

另一方面，ERP 系统通过定义事务处理（Transaction）相关的会计核算科目与核算方式，以便在事务处理发生的同时自动生成会计核算分录，保证了资金流与物流的同步记录和数据的一致性，从而实现了根据财务资金现状，可以追溯资金的来龙去脉，并进一步追溯所发生的相关业务活动，改变了资金信息滞后于物料信息的状况，便于实现事中控制和实时做出决策。此外，计划、事务处理、控制与决策功能都在整个供应链的业务处理流程中实现，要求在每个流程业务处理过程中最大限度地发挥每个人的工作潜能与责任心。流程与流程之间则强调人与人之间的合作精神，以便在有机组织中充分发挥每个人的主观能动性与潜能。实现企业管理从“高耸式”组织结构向“扁平式”组织结构的转变，提高企业对市场动态变化的响应速度和能力。

8.8.3 应用 ERP 与企业的关系

ERP 是借用一种新的管理模式来改造原企业旧的管理模式，是先进的、行之有效的管理思想和方法。目前，ERP 软件在实际的推广应用中，其应用深度和广度都不到位，多数企业的效果不显著，没有引起企业决策者的震动和人们的广泛关注。有必要明确 ERP 的应用对促进企业发展的重要作用。

（1）实施 ERP 是企业管理全方位的变革

ERP 是现代管理理论的集中体现，要规范企业管理及其有关环节，必须使之成为领导者、管理层及员工自觉的行动，使现代管理意识扎根于企业中，成为企业文化的一部分，才能实现企业的持续发展和全方位的变革。ERP 的实施过程要注意以下问题：

①企业管理班子要取得共识，做好管理的基础工作，这是任何好的应用软件和软件供应商都无法提供的，只能靠自己勤勤恳恳地耕耘。把 ERP 的实施称为“第一把手工程”，这说明了企业的决策者在 ERP 实施过程中的特殊作用。ERP 是一个管理系统，牵动全局，没有第一把手的参与和授权，很难调动全局。

② ERP 的投入是一个系统工程，并不能立竿见影，它所贯彻的主要是管理思想，这是企业管理中的一条红线，它长期起作用、创效益，在不断深化中向管理要效益。

此外，实施 ERP 还要因地制宜，因企业而别，具体问题具体分析。首先，要根据企业的具体需求上相应的系统，而不是笼统地都上小型机，或者不顾企业的规模上 WindowsNT，这样长期运作，对企业危害性极大。其次，这种投入不是一劳永逸的，由于技术的发展很快，

随着工作的深入，企业会越来越感到资源的紧缺，因此，每年应有相应的投入，才能保证系统健康地运转。

（2）实施 ERP 可最大限度地发挥企业资源的作用

企业的硬件资源包括厂房、生产线、加工设备、检测设备、运输工具等，而人力、管理、信誉、融资能力、组织结构、员工的劳动热情等则是企业的软件资源。在企业运行发展中，这些资源相互作用，形成企业进行生产活动、完成生产任务、创造社会财富、实现企业价值的基础，反映企业在竞争发展中的地位。

ERP 系统的管理对象便是上述各种资源及生产要素，通过 ERP 的使用，最大程度地发挥这些资源的作用，使企业的生产过程能及时、高质量地完成客户需要的产品，并根据客户需求及生产状况做出调整资源的决策。

企业发展的重要标志是，能否合理调整和运用资源，如果没有 ERP 这样的现代化管理工具，企业资源状况及调整方向不清楚，要做调整安排是相当困难和漫长的，资源的运行难于比较、把握，并做出调整。ERP 系统正是针对这些问题设计的，成功推行的结果必使企业能更好地运用资源。

（3）ERP 的实施需要复合型人才

复合型人才的培养需要有一个过程和一定的时间，但企业领导者常把这样不多的人才当作一般管理者，没有把他们当作是企业来之不易的财富，这与长期忽视管理有关。这些复合型人才在企业中的地位远远不及市场开拓人员和产品开发者，而是“辅助”角色，不是政策倾斜对象，这种因素是造成人才流失的重要原因。另外，当企业上 ERP 时，这些复合型人才起到了先导作用。

总之，条件具备的企业要不失时机地上 ERP 管理系统，不能只搞纯理论研究、再研究，长时间地考察。要首先整理好内部管理基本数据，选定或开发适合自己企业的 ERP 软件，条件成熟了就上。

（4）ERP 的风险及其预防

企业的条件无论多优越，所做的准备无论多充分，实施的风险仍然存在。在 ERP 系统的实施周期中，各种影响因素随时都可能发生变化。如何有效地管理和控制风险是保证 ERP 系统实施成功的重要环节之一。

通常人们在考虑失败的因素时，一般着重于对实施过程中众多因素的分析，而往往忽视项目启动前和实施完成后 ERP 系统潜在的风险。对于 ERP 项目而言，风险存在于项目的全过程，包括项目规划、项目预准备、实施过程和系统运行。归纳起来，ERP 项目的风险主要有以下几方面：

①缺乏规划或规划不合理；

②项目预准备不充分，表现为硬件选型及 ERP 软件选择错误；

③实施过程控制不严格，阶段成果未达标；

④设计流程缺乏有效的控制环节；

⑤实施效果未做评估或评估不合理；

⑥系统安全设计不完善，存在系统被非法入侵的隐患；

⑦　灾难防范措施不当或不完整，容易造成系统崩溃。

8.8.4 ERP 应用成功的标志

ERP 应用是否成功，原则上说，可以从以下几个方面加以衡量：

（1）系统运行集成化

这是 ERP 应用成功在技术解决方案方面最基本的表现。ERP 系统是对企业物流、资金流、信息流进行一体化管理的软件系统，其核心管理思想就是实现对“供应链（Supply Chain）”的管理。软件的应用将跨越多个部门甚至多个企业。为了达到预期设定的应用目标，最基本的要求是系统能够运行起来，实现集成化应用，建立企业决策完善的数据体系和信息共享机制。

（2）业务流程合理化

这是 ERP 成功应用在改善管理效率方面的体现。ERP 应用成功的前提是，必须对企业实施业务流程重组，因此，ERP 应用成功也即意味着企业业务处理流程趋于合理化，并实现了 ERP 应用的以下几个最终目标：

①企业竞争力得到大幅度提升；

②企业面对市场的响应速度大大加快；

③客户满意度显著改善。

（3）绩效监控动态化

ERP 的应用将为企业提供丰富的管理信息。如何用好这些信息并在企业管理和决策过程中真正起到作用，是衡量 ERP 应用成功的另一个标志。在 ERP 系统完全投入实际运行后，企业应根据管理需要，利用 ERP 系统提供的信息资源设计出一套动态监控管理绩效变化的报表体系，以期及时反馈和纠正管理中存在的问题。这项工作一般是在 ERP 系统实施完成后由企业设计完成的。企业如未能利用 ERP 系统提供的信息资源建立起自己的绩效监控系统，将意味着 ERP 系统应用没有完全成功。

（4）管理改善持续化

随着 ERP 系统的应用和企业业务流程的合理化，企业管理水平将会明显提高。为了衡量企业管理水平的改善程度，可以依据管理咨询公司提供的企业管理评价指标体系对企业管理水平进行综合评价。评价过程本身并不是目的，为企业建立一个可以不断进行自我评价和不断改善管理的机制，才是真正目的。这也是 ERP 应用成功的一个经常不被人们重视的标志。

8.8.5 SAP ERP 简介

SAP 是德国的一家著名的生产 ERP 软件的公司，其市场份额位居世界之首。SAP 的产品有面向企业级的 SAP R/3 Enterprise，以及面向中小型公司的 MySAP。

SAP 公司成立于 1972 年，总部位于德国沃尔多夫市，是全球最大的企业管理和协同化电子商务解决方案供应商，全球第三大独立软件供应商。目前，SAP 在 50 多个国家拥有 29000 多员工。在 120 多个国家和地区拥有 18800 多家客户、56000 多个系统安装点，以及 1000 多万名最终用户，世界 500 强中 80％以上的公司都在使用 SAP 的管理解决方案。

SAP 起源于 Systems Application Products in DATA Processing，SAP 既是公司名称，又是其 ERP（Enterprise-wide Resource Planning）软件的名称。

1988 年，SAP 陆续在多家证交所上市，包括法兰克福和纽约证券交易所。SAP 软件为 ERP 提供了成功解决方案，使企业得以高速、有效的良性运行。SAP 为 21 个行业提供融合了各行业“最佳业务实践”的行业解决方案，这些行业包括航空与国防、汽车、金融服务、化工、消费品、工程与建筑、医疗卫生、高等教育、高科技、保险、媒体、石油与天然气、煤矿、医药、公用事业、零售业、电信、电力、钢铁冶金、交通运输及公共设施等。SAP 在每个行业都有行业解决方案图，以充分展示各行业特殊业务处理要求，并将其绘制到 SAP 解决方案和合作伙伴补充方案中，完成包括基于网络的端到端的业务流程。

SAP 的主要优势包括以下几个方面：

①基于 30 年最佳实践的行业经验。SAP 在 30 年来持续领先 ERP 市场份额。事实上，影响 ERP 实施的关键环节包括设计、定制、实施。因此对于业务的深入了解，以及提供咨询、服务支持的能力至关重要。

②完善的产品线，丰富的特性及功能。SAP 产品涵盖 ERP/CRM/PLM/SCM/SRM 等完整的企业价值链。ERP 产品具有完整的自助服务、分析、财务、人力资本管理、运营和企业服务功能。此外，还包括对诸如用户管理、配置管理、集中数据管理和 Web 服务管理等系统管理问题的支持。

③广泛的国内外客户群体。全球有 120 多个国家的超过 20000 家企业正在运行着 64500 多套 SAP 软件，中国的 400 多家来自各行业的企业在使用 SAP 的解决方案，如中国电信、招行、浪潮集团等。

以 SAP 的 R/3 系统为例，它是现代企业管理的技术先驱和代表产品，内容包括：

- FI——应收、应付、总账、合并、投资、基金、现金等；
- CO——利润及成本中心，产品成本、项目会计、获利分析等；
- AM——固定资产、技术资产、投资控制等；
- SD——销售计划、询价报价、定单管理、运输发货、发票等；
- MM——采购、库房管理、库存管理、MRP、供应商评价等；
- PP——工厂数据、生产计划、MRP、能力计划、成本核算等；
- QM——质量计划、质量检测、质量控制、质量文档等；
- PM——维护及检测计划、单据处理、历史数据、报告分析等；
- HR——薪资、差旅、工时、招聘、发展计划、人事成本等；
- PS——项目计划、预算、能力计划、资源管理、结果分析等；
- WF——工作定义、流程管理、电子邮件、信息传送自动化等；
- IS——针对不同行业提供特殊应用。

其基础部分包括：R/3 系统内核、数据库、支持各类平台的接口、ABAP/4 工具语言等。

SAP R/3 软件具备以下功能和主要特点:

①功能性。R/3 以模块化的形式提供了一整套业务措施，其中的模块囊括了全部所需要的业务功能，并把用户与技术性应用软件相联系而形成一个总括的系统，用于公司或企业战略上和运用上的管理。

②集成化。R/3 把逻辑上相关联的部分连接在一起，重复工作和多余数据被完全取消，规程被优化，集成化的业务处理取代了传统的人工操作。

③灵活性：R/3 系统中方便的裁剪方法使之具有灵活的适应性，从而能满足各种用户的需要和特定行业的要求。R/3 还配备有适当的界面来集成用户自己的软件或外来的软件。

④开放性。R/3 的体系结构符合国际公认的标准，使客户得以突破专用硬件平台及专用系统技术的局限。同时，SAP 提供的开放性接口，可以方便地将第三方软件产品有效地集成到 R/3 系统中来。

⑤用户友好。图标与图形符号简化了人机交互时的操作。统一设计的用户界面确保了工作人员能够运用同样熟悉的技术从事不同的工作。

⑥模块化。R/3 的模块结构使用户既可以一个一个地选用新的实用程序，也可以完全转入一个新的组织结构体系。

⑦可靠。作为用户的商业伙伴，SAP 始终不断地为集成化软件的质量设立越来越多的国际标准。

⑧低成本高效益。信息处理是取得竞争优势的要点之一。当竞争加剧时，企业必须更加努力地获取其市场占有量。这就要使用高度集成化的数据处理软件，而 R/3 正是这种软件的优秀典范。

⑨国际化。R/3 支持多种语言，而且是为跨国界操作而设计的。R/3 可以灵活地适应各国的货币及税物要求。

⑩服务：R/3 系统实施过程中，用户将得到 SAP 技术专家的全面支持与服务，包括组织结构方面与技术方面的咨询，项目计划与实施方面的协助，以及各类培训课程。

小结

大量的工程实践证明，软件项目管理是保证软件产品质量，保证软件项目开发成功的关键。现代软件开发特别强调对软件开发全过程的跟踪和控制，因此软件项目管理贯穿了整个软件生命周期。项目管理的实施，可以对软件开发成本进行有效控制，避免或控制项目风险。

本章在介绍项目管理的主要特点和内容的基础上，对软件项目的可行性研究、软件成本估算技术及如何保证软件质量等项目管理的重要问题进行了讨论。特别是对项目的组织及人员管理进行了较详细的讨论，因为人员既是开发过程中最不确定的因素，又是最重要的因素。

软件项目的风险管理是软件项目管理的重要内容。本章还对风险的识别、风险估算、风险评价、风险监控和管理进行了详细的讨论。

习题八

1．为什么要进行软件项目管理？

2．软件项目管理有哪些特点？

3．软件项目管理主要对哪些方面进行管理？

4．项目可行性报告包括哪几部分的内容？

5．一个公司为它的电子商务网站建设发出标书，你的公司准备投标。作为项目负责人，请你估算这个项目的成本。

6．竞标的对手实力很强，你准备采用压价的方式来争取这个项目吗？

7．考察一个实际的项目，看看项目涉及的人员有哪几类？他们各自负责什么样的工作？

8．如果你是培训经理，负责对项目中的程序员进行程序设计语言 C#的培训。那些接受

培训的程序员有刚从学校毕业的学生，也有从其他公司跳槽过来的人员。请根据他们的不同特点制订一套合理的培训方案。

9．请设计一份问卷调查表，其内容主要涉及各类人员在项目进行过程中的心理和生理需求，然后进行抽样调查。

10．分析你的问卷表结果，看看能得出什么样的结论。

11．有这样一个项目，参与人员较多，公司分配给项目组的办公面积也有 1000m^2，还有各种开发设备。假设你是项目负责人，请根据你得出的调查结论，提出一套方案对项目中涉及的硬件设施和人员进行分配和管理。

12．为什么软件项目风险管理是项目管理的重要内容？风险管理有效地保证了软件产品的质量？

13．软件项目可能有哪些类型的风险？如何识别？

14．如果你接手了一项前任项目经理未完成的项目，那么你打算如何对项目进行管理？

15．影响软件质量的因素有哪些？

16．假设你正在负责一项软件开发项目，你的客户对软件的质量非常关心。请写一份质量度量报告来取信你的客户。

17．ERP 系统的管理思想是什么？

18．有人说，一个公司不上 ERP 是等死，上 ERP 是找死。你如何理解这样的抱怨？又打算怎样来反驳？