2008

基于 CMMI 的软件工程教程

张万军 储善忠 袁宝兰 王红女

胡希明

杭州电子科技大学

2008-8-20

内容简介

本书是在作者多次讲授实用软件工程课程的讲义基础之上，以配合学生实训项目的进展，对讲授的内容章节适当进行了调整。并没有分成工程过程及项目管理过程、支持过程三大部分讲解，而是穿插起来讲解。

本书的内容以 CMMI1.2 版本相关过程管理思路为基础，重点提炼出 CMMI 中各过程域（简称 PA）的精髓，结合当前国内企业实际开发需求及 CMMI 推行情况，对 CMMI 及软件工程相关理论、思想、实践进行简化，编写出满足于软件技术专业或软件工程专业本、专科生适用的软件工程教材，也可以作为工程型软件技术专业学生项目实践类课程的参考书，对于中小型企业 CMMI 体系的推广及评估也具有一定的参考作用。同时，通过对每个过程实践提供详细的实训指导，提高学生的实际动手能力，增强对软件开发过程的规范化认识。以 CMMI 中的工程过程、项目管理、支撑过程、过程管理四大领域中的相关 PA 为知识点，以国内企业实际使用的模式来编写，考虑到学生的实际接受能力，每个领域中均提供了简化后并能充分体现 CMMI 精髓的模板及表单。

在本书中，模拟了一个软件企业，该企业有一个规模为 10 人左右的研发部门，以此部门要开发一个新产品类软件研发为场景进行实训，书中讲解内容包含了企业中常见的三类项目：新产品研发类、合同定制类和产品升级类。在本书后继版本中会增加软件外包类、技术服务类等项目的相关内容。对于需要实训的章节，在本书配套的《基于 CMMI 的软件工程实训指导》里提供相对应的实训指导内容，以方便学生利用书中的内容进行软件项目的实训。

作为书的配套资料，编写了《基于 CMMI 的软件工程实训指导》，并附带光盘，供在校学生做为教材使用。在《基于 CMMI 的软件工程实训指导》中，重点是解释各种模板或表单的填写方法及内容，以方便学习过程中师在此基础之上，选择其他项目的实训。为了降低学生在实训项目开发过程中编程技术的门槛，保证实训围绕软件过程管理开展。在《基于 CMMI 的软件工程实训指导》提供了基于微软平台和基于 Java 平台的实训框架，使得学生在此框架

基础之上完成所选项目的功能模块的编写即可，而不必再把重要精力放到系统怎么实现的技术上。

采用本书的学生要求学习过 UML 系统分析与设计、至少一门高级程序语言、熟悉.NET 平台或 Java 相关开发平台或工具。

专科（本书中统称第一类学员）建议讲解：第 1 章 软件工程基础、第 2 章 案例机构设置及岗位职责、第 3 章 立项管理、第 5 章 项目初步计划、第 6 章 需求开发及管理、第 7 章 项目估算及项目详细计划、第 8 章 软件配置管理、第 10 章 项目跟踪及控制、第 11 章 系统设计、第 12 章 软件测试简介、第 13 章 系统实现与测试过程、第 14 章 制定测试方案及编写测试用例、第 17 章 项目总结共十三章的内容；根据学生的实际情况，可以补充讲解一下配置管理工具的使用、Project 基本操作等知识点。建议学时为：4+3，即每周 4 节讲授，6 节上机，以保证小型实践项目的完成。

本科或开设过软件工程课程的专科（本书中统称第二类学员）建议在第一类学员的基础之上增加：第 4 章 项目评审管理、第 9 章 风险管理、第 15 章 系统测试、第 16 章 客户验收、第 18 章 产品及过程质量保证共五章内容。建议学时为：4+3 或 2+3，由老师根据学生原来的基础来确定，只是增加了这五章相关内容的实践环节。

软件工程硕士或高年级软件工程本科生（本书中统称第三类学员），建议讲解 21 章全部内容，以对整个机构一级的软件开发及管理有整体了解，同时增加附录部分的讲解，基本上满足 CMMI3 级的相关要求（未包含“SAM——供应商合同管理”和“OT——机构培训”两个过程域的内容）。开课学时，可以根据学生的基础来调整，根据对企业里人员培训的经验来看，一个小型项目导入 CMMI3 级，从立项到项目完成，一般需要 3 个月左右时间，其中讲授时间一般为每周 4-6 节课，其他时间为项目组成员讨论及应用；并且在讲解的时候，先讲解机构级的相关内容，即 20 章 软件开发过程管理和 21 章 ，然后再针对具体项目的应用情况进行讲解。

在本书的过程编写中，得到了胡希明老师的大力支持及精心指导，在此深表感谢。

目　录

第1章 软件工程基础[1]

内容提要：

- 软件工程基本原理
- 质量管理体系 **ISO9001**
- 项目管理知识体系 **PMBOK**
- 软件能力成熟度模型集成 **CMMI**
- 软件过程管理标准化国内动态

在学习软件工程及软件过程管理之前，我们可以看到，机械工业以至一般的加工业，都已经有了上百年的历史，产品的生产流程以及工厂、车间、工种等等的机构设置和角色分工都有了成熟的模式。但是，软件企业及其软件产品的生产，历史不长，加之软件本身的智力劳动的特性，软件作为产品的生产流程及其相应的管理活动，还远远没有一个成熟的模式。

近 10 年来，在国家各级主管部门的政策倡导和支持下，中国软件企业的决策者也从各自的成长历程中认识到了加强和改进内部管理特别是技术管理的重要性，纷纷投入大量的人力、物力和财力，学习、采用和实施一系列的学科、标准和模型，例如软件工程、ISO9001、PMBOK 以及 CMM、CMMI 等等。

1.1 软件工程基本原理

为了改进软件企业的管理，为了“更快、更好、更便宜”的开发软件产品，既要有技术措施（方法和工具），又要有必要的组织管理措施。从学科发展角度出发，人们很自然的想到

[1] 本章的内容在胡希明老师培训讲义基础之上编写，根据 CMMI-Dev V1.2 进行了修改。

了软件工程。因为软件工程正是从管理和技术两方面研究如何采用工程的概念、原理、技术和技术方法并加以综合，指导开发人员更好地开发和维护计算机软件的一门新的学科。

自从 1968 年在联邦德国召开的一次国际会议上正式提出并采用“软件工程”这个术语以来，研究软件工程的专家学者们陆续提出了 100 多条关于软件工程的准则或“信条”。著名的软件工程专家波汉姆（Boehm）综合这些学者们的意见，并总结了多年开发软件的经验，于 1983 年在一篇论文中提出了软件工程的 7 条基本原理。他认为这 7 条原理是确保软件产品质量和开发效率的原理的最小集合。人们虽然不能用数学方法严格证明它们是一个完备的集合，但是，事实证明在此之前已经提出的 100 多条软件工程原理都可以由这 7 条原理的适当组合所蕴含或派生所得到。

下面给出这 7 条基本原理的简要内容。

1. 按照软件生命周期的阶段划分制定计划，严格依据计划进行管理。

在软件开发与维护的整个生命周期中，需要完成许多性质各异的工作，应该把软件生命周期划分成若干个阶段，并相应地制定出切实可行的计划，然后严格按照计划对软件的开发与维护工作进行管理。共有 6 类计划，包括项目概要计划、里程碑计划、项目控制计划、产品控制计划、验证计划和运行维护计划。

不同层次的管理人员都必须严格按照计划各尽其职地管理软件开发与维护工作，绝不能受客户或上级人员的影响而擅自背离或随意修改预定计划。

2. 坚持进行阶段评审

软件质量保证工作不能等到编码阶段结束之后再进行，因为大部分缺陷是在编码之前造成的（统计结果显示：设计阶段注入的缺陷占缺陷总数的 63%，而编码阶段注入的缺陷仅占 37%），并且缺陷发现与改正越晚，所需付出的代价就越高。因此，在每个阶段都应进行严格的评审，以便尽早发现在软件开发过程中所犯的错误。

3. 实行严格的产品控制

在软件开发过程中不应随意改变需求，因为改变一项需求往往需要付出较高的代价。但是，在软件开发过程中改变需求又是难免的，由于外部环境的变化，相应地改变用户需求是

一项客观需要，显然不能硬性禁止客户提出需求变更请求，而只能依靠科学的控制技术来顺应这种要求。也就是说，当改变需求时，为了保持软件各个配置项的一致性，必须实行严格的产品控制，其中主要是实行基准配置管理，定义基线，管理和控制基线。基准配置管理也称为变更控制。一切有关修改软件的建议，特别是涉及到对基准配置的修改建议，都必须按照严格的规程进行评审，获得批准以后才能实施修改。绝对不能谁想修改软件（包括尚在开发过程中的软件），就随意进行修改。

4. 采用现代程序设计技术

从提出软件工程的概念开始，人们一直把主要精力用于研究各种新的程序设计技术。例如 20 世纪 60 年代末提出的结构程序设计技术以及后来发展的面向对象的分析技术和编程技术，等等。实践表明，采用先进的技术既可提高软件开发的效率，又可提高软件维护的效率。

5. 结果应能清楚地审查

软件产品不同于一般的物理产品，它是看不见摸不着的逻辑产品。软件开发人员的工作可视性差，难以准确度量，从而使得软件产品的开发过程比一般产品的开发过程更难于评价和管理。为了提高软件开发过程的可视性，更好地进行管理，应该根据软件开发项目的目标及完成期限，规定开发机构的责任和产品标准，从而使得所得到的结果能够清楚地加以审查。

6. 开发小组的人员应该少而精

这条基本原理的含义是，软件开发小组的组成人员的素质应该好，而人数则不宜过多。开发小组人员的素质和数量是影响软件产品质量和开发效率的重要因素。素质高的人员的开发效率比素质低的人员的开发效率可能高几倍至几十倍，而且素质高的人员所开发的软件中的缺陷明显少于素质低的人员所开发的软件中的缺陷。此外，随着开发小组人员数目的增加，因为交流情况、讨论问题而造成的开销也急剧增加。因此，组成少而精的开发小组是软件工程的一条基本原理。

7. 承认不断改进软件工程实践的必要性

遵循上述 6 条基本原理，就能够实现软件的工程化生产，但是，仅有上述 6 条原理并不能保证软件开发与维护的过程能赶上时代前进的步伐，跟上技术的不断进步。因此，应把承

认不断改进软件工程实践的必要性作为软件工程的第 7 条基本原理。按照这条原理，不仅要积极主动地采纳新的软件技术，而且要注意不断总结经验。例如收集进度和资源耗费数据，收集缺陷类型和问题报告数据等等。这些数据不仅可以用来评价新的软件技术的效果，而且可以用来指明必须着重开发的软件工具和应该优先研究的技术。

以上 7 条只是基本原理，对每一个软件企业而言，如何根据这几条原理管理和改进软件产品的开发和维护过程，问题还是不少，主要是可操作性差，缺少评价标准，以及缺少相互之间的可比性。于是，人们又只好求助于其他与产品质量管理、项目管理相关的标准体系，或者是新出现的并已证明有效的专门关于软件过程改进和管理的评价模型。

从当前及今后一个时期看，一个软件企业在技术、产品管理方面可采用的标准体系或模型，基本上有三个，它们之间的关系如下图示：

图表 1-1 企业技术、产品体系模型图

【注：CMM——Capability Maturity Model，能力成熟度模型；CMMI——Capability Maturity Model Integration，能力成熟度模型集成；PM——Project management，项目管理；PMBOK——（A Guide to the Project Management Body Of Knowledge——项目管理知识体系指南），由 PMI(美国项目管理协会)颁布。】

从上图可以看出：三者不存在互相包含的关系，但有很强的关联性；三者不存在互相替代的关系，但侧重点各有不同；PM 和 ISO9001 并不专门针对软件企业，但可用于软件企业特别是包括软件产品、集成工程和服务的软件企业；CMM、CMMI 专用于软件企业或软件项目，或系统集成企业或系统集成项目。

1.2 质量管理体系 ISO9001

本节简要介绍质量管理体系 ISO9001:2000。ISO9001 规定了企业质量管理体系的基本要求，它是通用的，适用于所有行业或经济领域，不论其提供何种类别的产品。ISO9001 本身并不规定产品质量的要求。

1、质量管理原则

为促进质量目标的实现，ISO9001 标准明确规定了以下八项质量管理原则：

- 以顾客为中心。
- 高层管理者推动。
- 全员参与。
- 采用过程方法。
- 系统的管理。
- 持续改进。
- 基于事实的决策。
- 互利的供方关系。

2、建立和实施质量管理体系的步骤

建立和实施质量管理体系，一般应按下列步骤进行：

- 确定顾客的需求和期望；
- 建立企业的质量方针和质量目标；
- 确定实现质量目标所必需的过程和职责；
- 对每个过程实现质量目标的有效性确定测量方法；
- 通过测量，确定每个过程的现行有效性；
- 确定防止不合格项并消除产生原因的措施；
- 寻找提高过程有效性和效率的机会；
- 确定并优先考虑那些能提供最佳结果的改进；

- 为实施已确定的改进，对战略、过程和资源进行策划；
- 实施改进计划；
- 监控改进效果；
- 对照预期效果，评价实际结果；
- 评审改进活动，确定必要的纠正、跟踪措施。

3、过程方法

任何“得到输入并将其转化为输出”的序列活动均可视为过程。

为使组织有效运行，必须识别和管理许多内部相互联系的过程。通常，一个过程的输出将直接形成下一个过程的输入。系统识别和管理组织内所使用的过程，特别是这些过程之间的相互作用，称为“过程方法”。ISO9001 标准鼓励采用过程方法建立和实施质量管理体系。

4、实例介绍

此处给出某著名软件企业采用 ISO9001 标准，建立和实施质量管理体系的概况，供参考。

- 过程识别

整个质量管理体系由四个大过程以及大过程所包含的若干个子过程构成，分别定义如下：

体系管理过程：对应于 ISO9001：2000 版标准条款 4 和 5，主要活动包括整个质量管理体系所包含的过程及子过程的识别和划分、过程之间关系的确定以及质量管理体系文件的编写、管理和控制；还包括确定管理承诺、质量方针、质量目标、职责划分以及为了质量管理体系的实施、保持和持续改进而进行的质量策划和管理评审。

资源管理过程：对应于 ISO9001：2000 版标准条款 6（资源管理与提供），为了质量管理体系的实施、保持和持续改进，公司应保证在人员编制、员工培训、基础设施、工作环境等方面提供必要、合理和充分的资源。

产品实现过程：是核心业务过程，对应于 ISO9001：2000 版标准条款 7（产品实现过程），包括产品策划子过程、与顾客相关的子过程、设计开发子过程、采购子过程以及生产和服务提供子过程。

监测、分析和改进过程：对应于 ISO9001：2000 版标准条款 8（测量、分析和改进）。主

要活动包括顾客满意、内部审核、过程监视、产品监视等过程的监视与测量活动，以及不合格品控制、数据分析、持续改进和纠正、预防措施。

上列 4 个基本过程以及相关子过程的相互关系，如图 1-2 所示（某公司 ISO9001 体系中过程关系图示例）。

■ 过程关系

图 1-2 描述了某公司 ISO9001：2000 版质量管理体系的整体过程关系，该公司主要是把 ISO9001 质量管理体系分成了四大块，分别为：体系管理过程，资源管理过程，产品实现过程，监测、分析与改造过程。这四大块形成一个循环，使得公司质量管理体系有效运转，并且为过程的持续改进提供保证。每块包含的内容及他们之间详细的关系如图 1-2 所示（见下页）。

■ 质量体系文件的分层结构

质量体系文件分为四个层次：

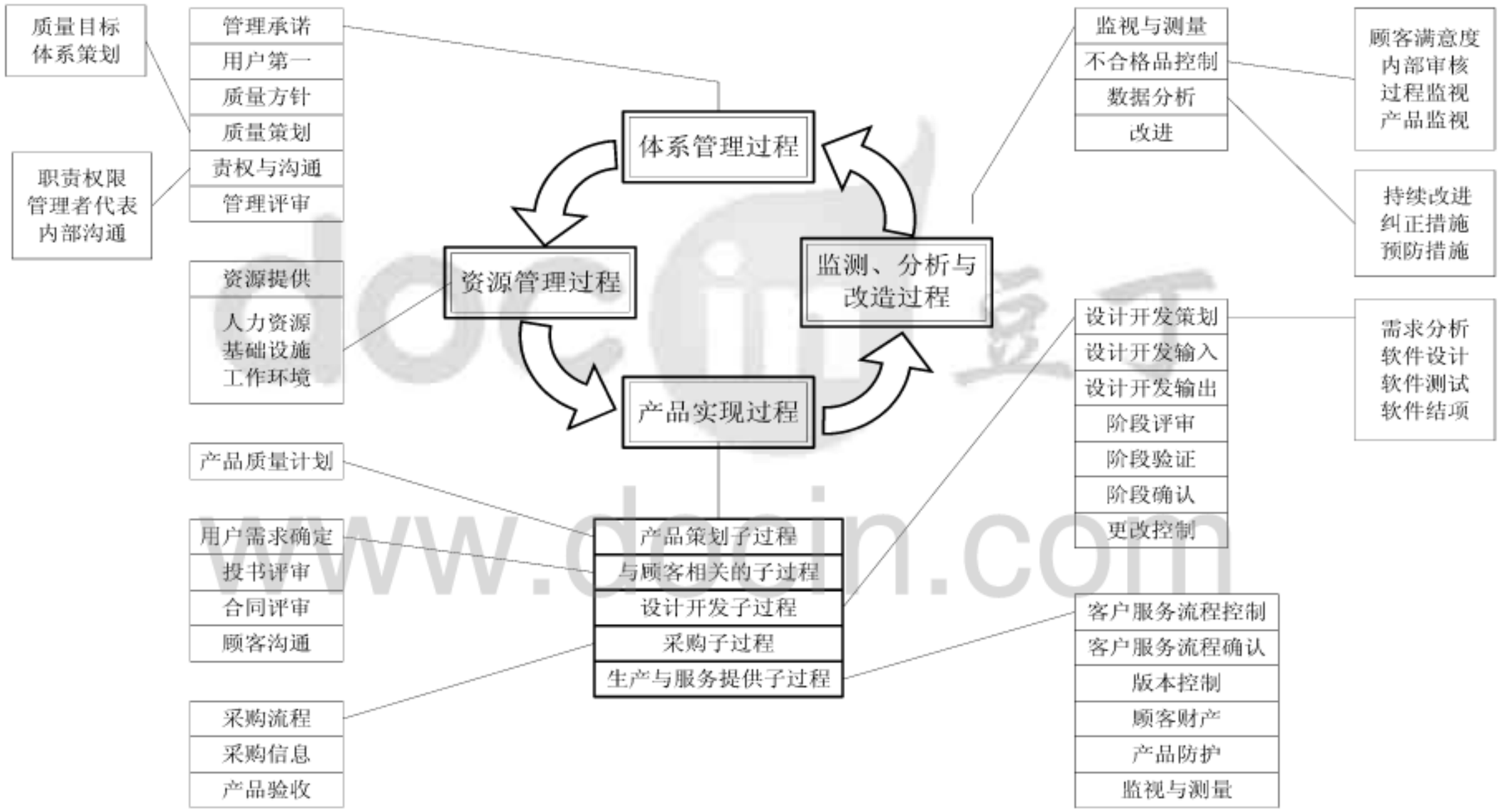
质量手册：质量体系文件中的纲领性文件。阐明公司质量方针、质量目标和质量策略；描述影响和参与质量活动的部门、岗位职责、权限和相互关系，同时概要描述了质量体系的主体文件即程序文件（规程）。

程序文件：质量手册的支持性文件，具体描述质量活动各个过程、子过程以及各阶段中所采取的措施和必需遵循的流程。

规定/规范：结合公司的具体情况而颁布的各类技术规范、工作规定及其配套考核细则。

表单模板：包括质量记录模板、文档模板等。

图表 1-2 过程关系图



1.3 项目管理知识体系 PMBOK

1. 项目基本属性

项目，是在限定时间内、利用有限的资源、完成有一定质量要求的目标而进行的一系列有序活动的一次性组合。充分认识项目属性，有利于做好项目管理。项目基本属性 8 条，包括：

- 整体性，是一系列活动的有序组合。
- 唯一性，每个项目均是具体的、特殊的，没有二个完全相同的项目。
- 一次性，目标一旦完成，项目即告结束。
- 目标性，一个项目有确定的成果性目标。
- 多约束性，在多种约束条件下完成项目的成果性目标，约束包括时间、资源、质量以及其他非技术性约束。
- 依赖性，项目活动的进行涉及多个方面的因素，有对内部各级各部门的依赖，有对用户条件的依赖，有对标准的依赖和对各类变更的依赖等等。
- 冲突性，项目内部会有多种冲突，需要沟通、协调和培训。
- 周期性，项目不同，但都有其基本的生命周期属性，都会经历大体相同的阶段。

2. 项目参数

用于刻画一个项目的主要参数有：范围、进度、资源、成本和质量。

3. 项目生命周期

项目生命周期划分成四个阶段：定义；策划；实施；收尾。项目生命周期与软件生命周期阶段划分的对应关系，如下表：

项目生命周期	软件生命周期
项目定义	立项管理、需求开发及管理
项目计划	项目计划
项目实施	系统设计、编码、测试
项目收尾	发布、提交、运行维护、技术支持和产品退役

4. 项目管理基本过程

项目管理基本过程共五个：启动过程、策划过程、执行过程、控制过程和结束过程。

5. 项目管理基本职能

项目管理基本职能有九个，分别为：项目整体管理、项目范围管理、项目时间管理、项目成本管理、项目质量管理、项目人力资源管理、项目沟通管理、项目风险管理、项目采购管理，这九个领域中分别包含的内容，如图 1-3 所示（见下页）。本书讲解的内容中，涉及到了除项目采购管理之外的八个领域。

6. 项目管理成熟度模型

CMM（见下一节）发布后，有人又根据 PMBOK 和 CMM，进一步提出了一套项目管理成熟度模型（Project Management Maturity Model）,简称 PMMM。PMMM V5.0(2002 年 10 月)标准文本与 CMM 非常相似，也分成 5 级（分别是：初始级、可重复级、已定义级、受管理级、优化级），也有关键过程域的概念，每个级别包含的要点及具体内容如表 1-4 所示（见下页）。

图表 1-3 项目管理职能图

图表 1-4 项目管理成熟度等级表

成熟度等级	关键过程域	
Initial Process 初始级	Project Definition	项目定义
Repeatable Process 可重复级	Project Establishment	项目立项
	Requirement Management	需求管理
	Risk Management	风险管理
	Project Planning	项目计划
	Project Monitoring and Control	项目监控

成熟度等级	关键过程域	
	Management of Suppliers and External Parties	供方和外部合作方管理
	Project Quality Control	项目质量控制
	Configuration Definition and Control	配置定义与控制
Defined Process 已定义级	Organizational Focus	机构聚焦
	Project Management Process Definition	项目管理过程定义
	Project Training	项目培训
	Integrated Management	综合管理
	Lifecycle Control	生命周期控制
	Inter-team Coordination	组间协调
	Quality Assurance	质量保证
Managed Process 受管理级	Project Metrics	项目度量
	Organizational Quality Management	机构质量管理
Optimization 优化级	Proactive Problem Management	意外问题管理
	Technology Management	技术管理
	Continuous Process Improvement	持续过程改进

1.4 软件能力成熟度模型集成 CMMI

一、什么叫 **CMMI**?

软件能力成熟度模型集成的英文全名是 Capability Maturity Model Integration，缩写为 CMMI。1984 年，美国国防部希望将国防部的软件委派给其他软件公司开发,由于没有办法客观评价软件公司的开发能力,因此委托卡内基—梅隆大学软件工程学院（Carnegie Mellon University Software Engineering Institute，简称 CMU/SEI）进行研究希望能够建立一套工程制度，用来评估和改善软件公司的开发过程和能力，并协助软件开发人员持续改进流程的成熟度及软件质量，从而提升软件开发项目及公司的管理能力，最终达到软件开发功能正确、缩短开发进度、降低开发成本、确保软件质量的目标。

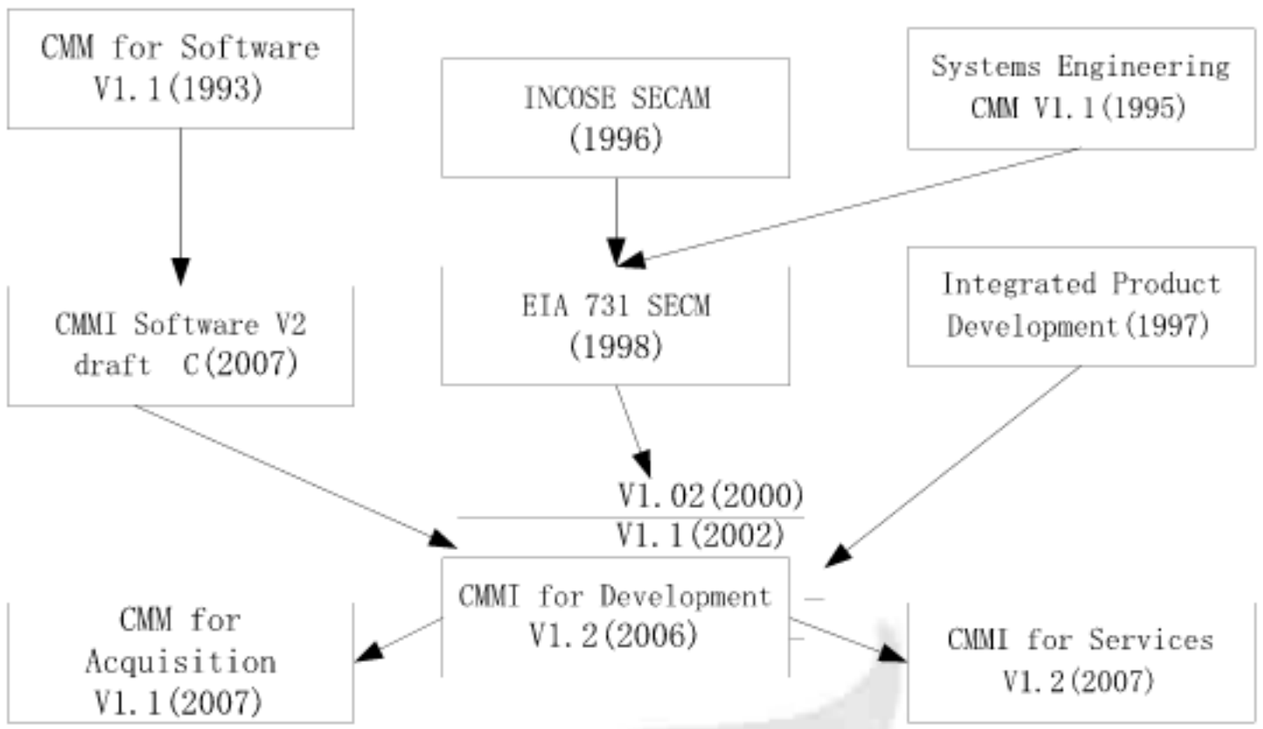
由此，SEI 在 1987 年提出了关于软件的《过程成熟度模型框架和成熟度问卷简要描述》，并在美国国防项目承包商范围内开始试行 CMM 等级评估。软件能力成熟度模型（Software Capability Maturity Model ，简称 SW-CMM） V1.0 发表之后，美国国防部合同审查委员会提出，发包单位可以在招投标程序中规定“投标方要接受基于 CMM 的评估”的条款，发包单

位将把评估结果作为选择承包方的重要因素之一。注意，接受并进行 CMM 评估只是有了参加美国军方项目投标的资格，CMM 评估决非像国内有些媒体上讲的那样："CMM 是进入美国市场以至国际市场的通行证"。但是，CMM 评估对软件过程改进确实有明显的促进作用，这使 SEI 看到了 CMM 评估的巨大商业前景，因此从 1990 年以后，（完整的 SW-CMM V1.1 版本于 1993 年发布），SEI 把基于 CMM 的评估作为商业行为推向市场。

在 CMM1.0 推出之后，很多单位都先后在不同的应用领域发展了自己的 CMM 系列，其中包括系统工程能力成熟度模型（Systems Engineering Capability Maturity Model，简称 SE-CMM）、集成的产品开发能力成熟度模型（Integrated Product Development Capability Maturity Model，简称 IPD-CMM）、人力资源管理能力成熟度模型（People Capability Maturity Model，简称 P-CMM）等应用模型。

这些不同的模型在自己的应用领域内确实发挥了很重要的作用，但是由于架构和内容的限制，他们之间并不能通用。于是 SEI 于 2000 年 12 月公布了能力成熟度模型集成（Capability Maturity Model Integration，简称 CMMI），主要整合了 SW-CMM2.0 版、系统工程能力模型（Systems Engineering Capability Model，科称 SECM）、IPD-CMM0.98 版。在随后的发展过程中，本着不断改进的原则，CMMI 产品团队不断评估变更请求并进行相应的变更，逐渐发展到目前的 CMMI1.2 版本。CMMI 的发展历程，如图 1-5 所示。

图表 1-5CMMI 发展历史图



现在使用的最新版本是 2006 年发布的 CMMI for Development V1.2 版，在 CMMI for Development 系列中包含两个模型，分别简称为 CMMI-Dev、CMMI-Dev+IPPD。由于 CMMI 是可扩充的集合，2007 年发布 CMMI for Services V1.2、CMMI for Acquisition V1.2，今后可能还会有新的学科模型出现。本书以 CMMI-Dev 为主来讲解，其包含了产品或服务的开发和维护活动，CMMI-Dev+IPPD 模型还包含集成团队开发和维护活动的应用。与 V1.1 相比，在 CMMI-Dev V1.2 版本中，降低了模型的复杂度及规模，同时扩展了模型的覆盖面，增加了对硬件的扩展。去掉了高级实践及共用特征；去掉了供方来源（SS）过程域；把集成化供方管理（ISM）合并到供方协议管理（SAM）；对 IPPD 相关资料进行统一整理及简化；把机构集成环境（OEI）相关内容整理进机构过程定义（OPD）；把集成化团队（IT）合并相关内容整理进集成化项目管理（IPM）；再就是，在 V1.2 中两种表示法使用同一个标准文本进行发布，共计 573 页。

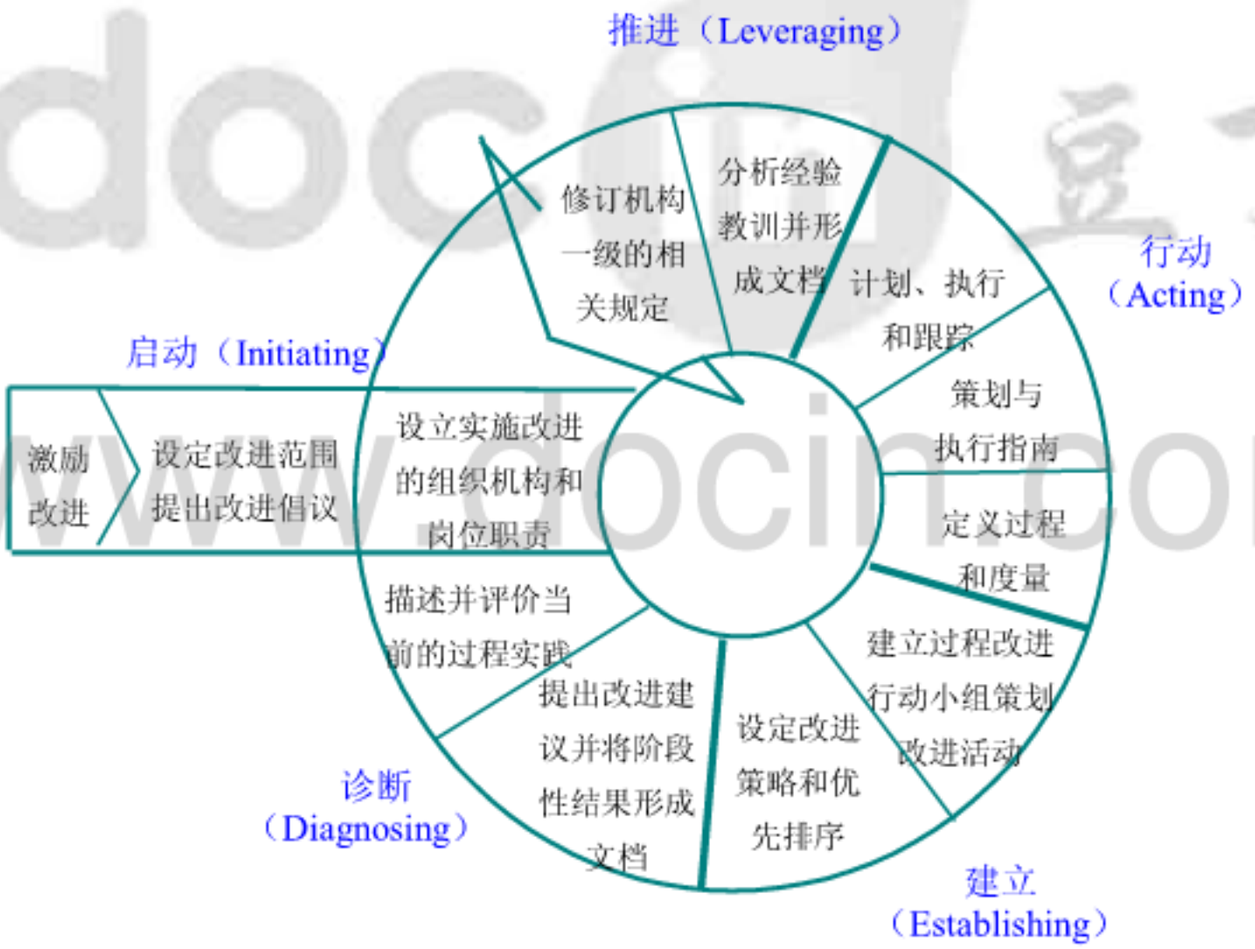
【注：V1.1 与 V1.2 模型的详细变化，请参见官方发布的 CMMI V1.2 升级培训讲义。】

二、CMMI 和过程改进

软件过程改进是一个持续的、全员参与的过程。CMMI 实施或软件过程改进(Software Process Improvement)采用的方法称 IDEAL 模式，分五步：启动(Initiating)、诊断(Diagnosing)、建立(Establishing)、行动(Acting)和推进(Leveraging)，如图 1-6 所示。在企业进行软件过程改

进时，通常诊断这一步做的不够到位，从而影响了整个过程改进的效果。诊断主要是描述并评价当前企业开发的过程，也就是识别现有开发过程，并且对现有过程中存在的问题进行发现；然后是提出改进建议并将阶段性的成果形成文档，最后是对这些问题改进的过程及方法设定策略，并根据公司实际情况进行优先级排序。比如：在公司产品开发过程中，可能会出现研发部门与工程部门或技术支持部门之间相互扯皮，那么就有可能是产品发布流程及支持维护流程出问题。如果该问题通过诊断，发现影响到了士气或团结，那么其优先级就可以设定为高，在某一阶段的过程改进时，重点解决这方面的问题。通过如此方法，逐步理顺开发过程中存在的问题，改进开发过程，提高产品质量及客户满意度，降低整体运营成本。

图表 1-6 软件过程改进的 IDEAL 模型图



三、CMMI 结构框架

CMMI 模型中，最基本的概念是“过程域”（Process Area），每个过程域分别表示了整个过程改进活动中应侧重关注或改进的某个方面的问题。模型的全部描述就是按过程域作为基

本构件而展开的，针对每个过程域分别规定了应达到什么目标（Goals）以及为了达到这些目标应该做些什么“实践”（Practices），但模型并不规定这些实践由谁做、如何做等等。在 V1.2 版本中，共计 22 个过程域，比 V1.1 版本有了不小的简化。下表按英文字母排序给出全部过程域清单，至于过程域的分类和分级则在后面再说明：

图表 1-7CMMI 过程域清单表

英文全称	简称	中文名称
Causal Analysis and Resolution	CAR	因果分析与解决方案
Configuration Management	CM	配置管理
Decision Analysis and Resolution	DAR	决策分析与解决方案
Integrated Project Management+IPPD	IPM+IPPD	集成化项目管理
Measurement and Analysis	MA	度量分析
Organizational Innovation and Deployment	OID	机构改进与部署
Organizational process Definition+IPPD	OPD+IPPD	机构过程定义
Organizational process Focus	OPF	机构过程聚焦
Organizational process Performance	OPP	机构过程性能
Organization Training	OT	机构培训
Production Integration	PI	产品集成
Project Monitoring and Control	PMC	项目监督与控制
Project Planning	PP	项目计划
Process and product quality Assurance	PPQA	过程和产品质量保证
Quantitative project Management	QPM	项目定量管理
Requirements Development	RD	需求开发
Requirements Management	REQM	需求管理
Risk Management	RSKM	风险管理
Supplier Agreement Management	SAM	供方协议管理
Technical Solution	TS	技术解决方案
Validation	VAL	确认
Verification	VER	验证

上表所列 22 个过程域，是按英文字母顺序排列的，很难看出它们相互之间的关系，其中“+IPPD”表示该过确良程域包含专门针对 IPPD 的目标及实践，如果选择 CMMI-DEV+IPPD 模型的时候，这些目标及实践是必须实现的。其实，如果我们从机构和项目组、项目管理、

过程管理三个方面加以考察，则可以将上列 22 个过程域分成如下四大类：

图表 1-8CMMI 过程域分类表

过程管理类	项目管理类	工程类	支持类
机构过程聚焦（OPF） 机构过程定义（OPD+IPPD） 机构培训（OT） 机构过程性能（OPP） 机构改进和部署（OID）	项目计划（PP） 项目监督与控制（PMC） 供方协议管理（SAM） 风险管理（RSKM） 集成化项目管理（IPM+IPPD） 项目定量管理（QPM）	需求开发（RD） 需求管理（REQM） 技术解决方案（TS） 产品集成（PI） 验证（VER） 确认（VAL）	度量分析（MA） 过程和产品质量保证（PPQA） 配置管理（CM） 成果分析与解决方案（CAR） 决策分析与解决方案（DAR）

另外，22 个过程域并非各自完全独立，而是互有联系，表 1-10 给出了过程域之间的主要关系。

图表 1-9 过程域之间的主要关系表

过程域	相关过程域
需求管理（REQM）	需求开发（RD）、技术解决方案（TS）、项目计划（PP）、配置管理（CM）、项目监督与控制（PMC）、风险管理（RSKM）
项目计划（PP）	需求开发（RD）、需求管理（REQM）、风险管理（RSKM）、技术解决方案（TS）
项目监督与控制（PMC）	项目计划（PP）、度量分析（MA）
供方协议管理（SAM）	项目监督与控制（PMC）、需求开发（RD）、需求管理（REQM）、技术解决方案（TS）
度量分析(MA)	项目计划（PP）、项目监督与控制（PMC）、配置管理（CM）、需求开发（RD）、需求管理（REQM）、机构过程定义（OPD）、项目定量管理（QPM）
过程和产品质量保证（PPQA）	项目计划（PP）、验证（VER）
配置管理（CM）	项目计划（PP）、项目监督与控制（PMC）
需求开发（RD）	需求管理（REQM）、技术解决方案（TS）、产品集成（PI）、验证（VER）、确认（VAL）、风险管理（RSKM）、配置管理（CM）
技术解决方案（TS）	需求开发（RD）、验证（VER）、决策分析和解决方案（DAR）、需求管理（REQM）、机构改进和部署（OID）
产品集成（PI）	需求开发（RD）、技术解决方案（TS）、验证（VER）、确认（VAL）、风险管理（RSKM）、决策分析和解决方案（DAR）、配置管理（CM）、供方协议管理（SAM）
验证（VER）	需求开发（RD）、确认（VAL）、需求管理（REQM）
确认（VAL）	需求开发（RD）、技术解决方案（TS）、验证（VER）
机构过程聚焦	机构过程定义（OPD）

过程域	相关过程域
（OPF）	
机构过程定义（OPD）+IPPD	机构过程聚焦（OPF）
机构培训（OT）	机构过程定义（OPD）、项目计划（PP）、决策分析和解决方案（DAR）
集成化项目管理（IPM）+IPPD	项目计划（PP）、项目监督与控制（PMC）、验证（VER）、机构过程定义（OPD）+IPPD、度量分析（MA）
风险管理（RSKM）	项目计划（PP）、项目监督与控制（PMC）、决策分析和解决方案（DAR）
决策分析和解决方案（DAR）	项目计划（PP）、集成化项目管理(IPM)、风险管理（RSKM）
机构过程性能（OPP）	项目定量管理（QPM）、度量分析（MA）
项目定量管理（QPM）	项目监督与控制（PMC）、度量分析（MA）、机构过程性能（OPP）、机构过程定义（OPD）、集成化项目管理（IPM）、原因分析和解决方案（CAR）、机构改进和部署（OID）
机构改进和部署（OID）	机构过程聚焦（OPF）、机构过程定义（OPD）、机构过程性能（OPP）、度量分析(MA)、集成化项目管理（IPM）、决策分析和解决方案（DAR）
原因分析和解决方案（CAR）	定量项目管理（QPM）、度量分析（MA）、机构改进和部署（OID）

四、CMMI 的阶梯表示

在阶梯式表示法中，CMMI 所包含的 22 个过程域，在所有的共性目标中，只有 GG2 和 GG3 适用，其中 GG2 有 10 个 GP，在 2 级时必须达到；GG3 有 2 个 GP，在 3 级时必须达到。

按照成熟度（Maturity）的概念分成 4 个组，如下表所示：

图表 1-10 按成熟度等级划分的过程域清单

成熟度等级	过程域缩写	过程域名称	目标数	实践数
2 级 受管理级	REQM	需求管理	1+1+1【注 1】	5+10+2【注 1】
	PP	项目计划	3+1+1	14+10+2
	PMC	项目监督与控制	2+1+1	10+10+2
	SAM	供方协议管理	2+1+1	8+10+2
	MA	度量分析	2+1+1	8+10+2
	PPQA	过程和产品质量保证	2+1+1	4+10+2
	CM	配置管理	3+1+1	7+10+2
3 级 已定义级	RD	需求开发	3+1 【注 2】	10+12 【注 2】
	TS	技术解决方案	3+1	8+12
	PI	产品集成	3+1	9+12

成熟度等级	过程域缩写	过程域名称	目标数	实践数
	VER	验证	3+1	8+12
	VAL	确认	2+1	5+12
	OPF	机构过程聚焦	2+1	9+12
	OPD+IPPD	机构过程定义	1+1+1【注 3】	6+12+3【注 3】
	OT	机构培训	2+1	7+12
	IPM+IPPD	集成化项目管理	2+1+1【注 3】	9+12+5【注 3】
	RSKM	风险管理	3+1	7+12
	DAR	决策分析与解决方案	1+1	6+12
4 级 定量管理级	OPP	机构过程性能	1+1	5+12
	QPM	项目定量管理	2+1	8+12
5 级 持续优化级	OID	机构改进与部署	2+1	7+12
	CAR	因果分析与解决方案	2+1	5+12

【注 **1**：每一个等级均包含有若干个过程域，每个过程域均规定了应达到的若干个目标，目标分为二类（特定目标和共性目标），为了达到每个目标又规定了若干个应该执行的活动（即实践），实践也相应的分为特定实践和共性实践。（详细说明见下一章），此处“x+y+z”是本书编者临时用的约定，不是 CMMI 标准中的用法，含义如下：

- x，表示该过程域应达到的特定目标（SG）数，或者为应执行的特定实践（SP）数；
- y，表示该过程域应达到的共性目标（GG）数，或者为应执行的共性实践（GP）数；
- 按 CMMI 标准规定，一个机构如果希望达到成熟度等级 2，则应通过过程改进，达到 2 级所包含 7 个过程域中每个过程域的 x+y 个目标；
- z，达到 2 级后，如果希望达到 3 级，则除了达到 3 级所包含每个过程域的 x+y 个目标之外，还要回到 2 级，在已经达到的 2 级每个过程域的 x+y 个目标之外再达到附加的 z 个共性目标以及为此而执行的多个附加共性实践。】

【注 **2**：此处 x、y 含义同上，而 z 只有 2 级才有，其他 3、4、5 级均不存在 z 的问题（后面还会有更细的说明）。】

【注 **3**：此处 x、y 含义同上，而 z 表示需要进行 IPPD 领域评估时才有的特定目标（SG）数，或者为应用的特定实践（SP）数。】

成熟度等级为机构的过程改进提供了一种阶梯式的上升顺序。按照这个顺序实施过程改

进，不需要同时处理可能涉及的所有过程，而是把过程改进的注意力集中于当前本机构最需要改进一组过程域上。每个成熟度等级为提升到更高一级奠定基础。每个级别的基本特征，描述如下：

级别 **1**——初始级

初始级的基本特征，包括：

- 机构项目组实际执行的过程是特定的（ad hoc）和无规则的。
- 机构一般不可能提供支持过程的稳定环境。
- 项目的成功往往取决于个人的能力和拼搏精神。离开了具备同样能力和经验的人，就无法保证在下一个项目中也能获得同样的成功。
- 机构在这种特定且无规则的环境中常常也能生产出可以使用的产品，但是伴随这种“成功”的往往是项目超过预算、拖延进度以至匆忙交付（或发布）从而大大地增加了产品交付后必需承担的维护成本。

级别 **2**——受管理级

一个机构，通过过程改进，针对图 1-10 所列 2 级所含的 7 个过程域，有效地执行了每个过程域标准规定的 x+y 个实践，达到了每个过程域标准规定的 x+y 个目标（不包括附加的 z 个目标），就认为该机构的整体开发能力达到成熟度 2 级。 达到成熟度 2 级的机构的基本特征，包括：

- 分派给项目组的项目需求得到管理；
- 项目的规模、工作量、成本、进度作了估计，并制定了项目开发计划，按计划进行项目开发；
- 在开发全过程中，按计划对项目进行监督和控制；
- 过程和产品相对于计划和标准的符合性得到客观评价，纠正不符合项；
- 产品配置项及其变更得到管理；
- 定义了过程和产品的基本度量，进行测量，对测量数据进行分析；
- 供方协议得到管理。

第 2 级与第 1 级之间的一个重要区别在于过程受到管理的程度。在第 2 级成熟度等级上，项目中的具体过程均受到严格控制，项目的成本、进度和质量目标能够得到实现。由第 2 级成熟度反映出来的过程规范有助于确保现行的实践不至于由于受到多重压力而被偏废。这些实践如果在其他类似的项目上使用，可望得到相同的结果。在第 1 级成熟度等级上，项目中的具体过程由项目开发者个人控制，机构无法或不能完全控制项目的过程，项目的成本、进度和质量目标等难以得到实现。

级别 **3**——已定义级

通过过程改进，一个机构的整体开发能力达到了成熟度等级 3，应满足如下 3 个条件：

- 达到了等级 2 所包含的每个过程域规定的 x+y 个目标；
- 达到了等级 3 中选定的学科所包含的每个过程域的 x+y 个目标；
- 达到了等级 2 所包含的每个过程域规定的 z 个附加目标。

成熟度等级 3 的基本特征，如下：

- 制定和维护机构标准过程集 OSSP(Organization's Set of Standard Processes)；
- 建立和维护机构过程资产 OPA(Organizational Process Assets)；

说明：OPA 包含如下内容：OSSP；生命周期描述；裁剪指南及准则；机构度量数据库（Repository）；机构过程资产库（Organization's Process Asset Library）(过程数据库和文档库)；机构过程性能基线（Organization's Process Performance Baselines）及其计算模型描述。

- 项目组一致地遵循机构裁剪指南，对 OSSP 进行裁剪，形成项目定义过程 PDP（Project Defined Processes），按项目定义过程进行项目开发；
- 达到等级 2 和等级 3 所包含的每个过程域的目标；
- 过程制度化的程度应达到“已定义级”。

成熟度等级 3 与等级 2 的重要区别，包括：

- 在等级 2，项目组所用的过程（包括过程描述、规程、方法、标准等）可能很不相同；但在等级 3，每个项目组的过程（即项目定义过程）都是一致地从同一个机构标准过程集经过裁剪而得到的，即便有区别也是裁剪指南所允许的。

- 在等级 3，过程的描述更详细、执行更严格，并且在执行和管理过程时更加强调对过程活动相互联系的深入理解，以及对过程、工作产品及其服务的更加详细的度量。

级别 4－定量管理级

通过过程改进，机构整体开发能力达到成熟度等级 4，应满足如下条件：

- 达到等级 2、3 和 4 所含每个过程域的特定目标；
- 达到等级 2、3 所含每个过程域的共性目标；
- 识别对过程性能和项目定量目标产生显著影响的过程或子过程，并采用统计学方法或其他定量技术定量地控制这些过程。

成熟度等级 3 与 4 的主要区别在于二者的可预测性：

- 在等级 4，过程性能受到统计控制并可以定量地预测目标；
- 在等级 3，只能定性地预测过程性能。

级别 5——持续优化级

通过过程改进，机构整体开发能力达到成熟度等级 5，应满足如下条件：

- 达到等级 2、3、4 和 5 所含过程域的全部特定目标；
- 达到 2、3 级所含过程域的共性目标；
- 根据对造成过程性能偏差的共同原因的定量理解，持续改进过程性能。

成熟度等级 5 所特别关注的过程性能的持续改进，可以是渐进式的也可以是突破性的改进。机构应根据商业目标及其变化，设定过程改进的定量目标。改进对象包括已定义过程和机构标准过程集 OSSP，但必须是已达到定量管理级的过程。通过持续改进，着重解决（或消除）引起过程性能偏差的共同原因、缺陷根源以及其他问题。应该根据对达到过程改进定量目标的贡献、对机构现行过程的影响以及所需代价，选择、评价和部署过程改进（例如采取适当措施，移动或提高某项性能的均值，减小均方差等等），并根据过程改进定量目标评价过程改进的效果。

定量管理的过程（即 4 级）与持续优化的过程（即 5 级）二者的主要区别在于：

- 持续优化过程，通过不断改进以解决引起过程偏差的共同原因；

- 定量管理过程，则侧重于消除引起过程偏差的特殊原因，并提供统计学意义上的预测结果。这个预测结果可能对达到机构的过程改进目标意义不大。

1.5 软件过程管理标准化国内动态

为加快我国软件能力模型标准的制定，信息产业部科技司于 2000 年 9 月主持成立了软件体系评估标准特别工作组（信息产业部电子标准化所为组长单位），同时提出了“依据我国软件政策，利用国际先进经验，结合我国国情，制定出有助于指导和促进我国软件企业发展的评估模型标准”的原则。

在过去几年研究工作的基础上，工作组进一步深入研究了 CMM、CMMI、ISO/IEC TR 15504、ISO 9000-3 以及其他有关的资料和文件，结合国情，确定了以 CMMI 作为主要参考文件来制定标准。在对标准草案进行会议审查和上网广泛征求意见的同时，组织了标准试点，最终形成了（已于 2001 年 4 月正式颁布）：《中华人民共和国电子行业标准（SJ/T 11235-2001）——软件能力成熟度模型》，《中华人民共和国电子行业标准（SJ/T 11234-2001）——软件过程能力评估模型》。

【注：前者以 CMMI-SW 标准的阶梯式表示法文本为依据，后者则以 CMMI-SW 标准的连续式表示法文本为依据。】

第2章 案例机构设置及岗位职责

内容提要：

- 案例介绍及机构设置
- 岗位角色职责

作者体会：在单位里，没有永远主管也没有永远的下属；好的主管应当希望自己的下属都能超越自己，变成自己的主管；“严以律己，宽以待人”是做为主管的前提条件；“兢兢业业工作，认认真真做人”是做好下属的基础。这些体会，提供给项目实训分组及角色分配时借鉴使用。

2.1 案例介绍及机构设置[2]

我们常见的软件工程教材及一些软件行业标准通常只规定了该做什么，并没有规定如何做，为了方便读者学习本书内容，本书模拟一个软件企业，该企业采用 CMMI 模型改进软件过程管理，该企业为 IT 企业提供管理信息系统产品的开发、安装、实施及维护。在此基础之上，如果该企业要开发一个新软件产品，应当怎么来开展工作？都是需要执行哪些活动？

在企业实际开发过程中，采用 CMMI 标准，进行公司软件开发过程的改进时，必须结合公司的实际情况，结合公司的历史、现状和将来的发展，来开展。正如质量管理体系具有明显的个性一样，软件过程管理也应具有个性。

从 CMMI 最初提出的动机就可以看出，它是从给定软件需求开始的，对软件项目组而言这是对的，但对公司管理而言就不够了。因此，我们国内企业在进行开发管理时，在 CMMI 的过程域（Process Area，简称 PA）之外，一般会增加一个 PA，即产品或合同项目的立项管

[2] 本节内容参照胡希明老师的培训讲义编写，根据学生授课的一些经验进行了调整。

理。

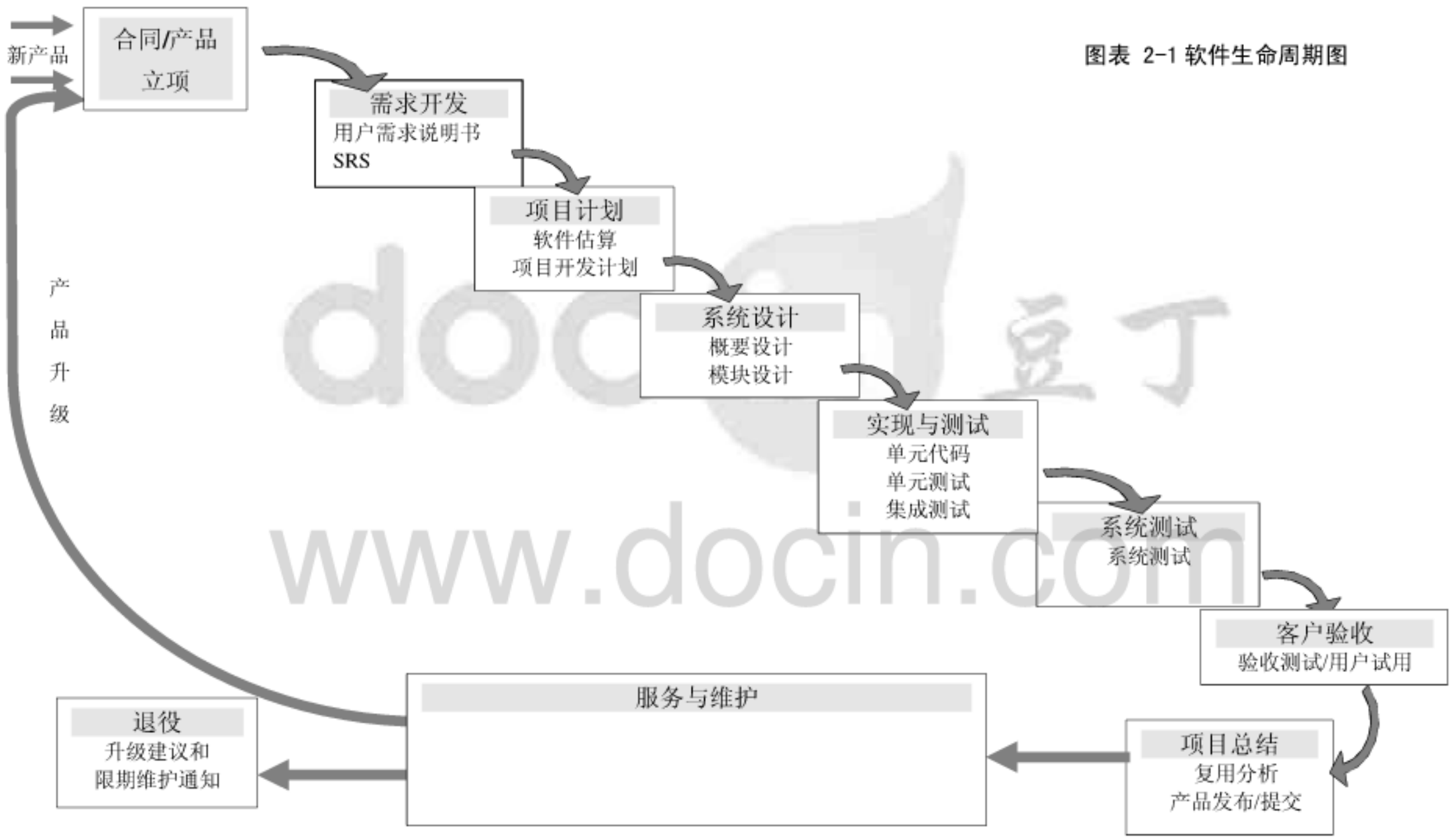
再有，CMMI 来认为产品开发完成后，提交验收就可结束。因此，产品发布或出售后的维护，CMMI 只是在软件产品工程过程域中通过一个实践加以规定。但对公司来说，远非如此，产品发布后还有许多事情必须做，技术交接、工程安装、日常维护、技术支持等等。这些活动对公司来说是至关重要的。因此，在案例企业的过程管理中，将产品服务与维护（包括工程安装、日常维护、技术支持）作为一个重要过程与 CMMI 标准中的其它过程域并列。

当然，其它具体情况还很多。总之，CMMI 标准要学习、要采用，但必须结合公司实际。通过努力，逐步形成一套带有明显企业特色、又符合 CMMI 标准但不受 CMMI 局限的软件过程管理模式。下面就是我们建议的一些具体做法。

① 确定软件生命周期

根据当前国内大部分行业管理信息系统应用软件开发及系统集成的特征，参照 CMMI 模型及其他标准的要求，在本案例中使用的软件产品生命周期模型基本是瀑布模型（或者称之为改进型瀑布模型），比较适用于以自身研发产品为核心的系统集成服务提供类的公司使用。并将整个生命周期划分为 10 个阶段，分别为：合同/产品立项、需求开发、项目计划、系统设计、实现与测试、系统测试、客户验收、项目总结、服务与维护、退役，在本书中没有讲到的有服务与维护、退役两阶段的内容。具体各个阶段主要活动或工作产品，见图表 2-1 软件生命周期图。每个阶段的工作内容和工作产品，更详细可以参见图表 2-2 软件生命周期阶段工作内容和工作产品列表。

图表 2-1 软件生命周期图



图表 2-2 软件生命周期阶段工作内容和常见工作产品列表

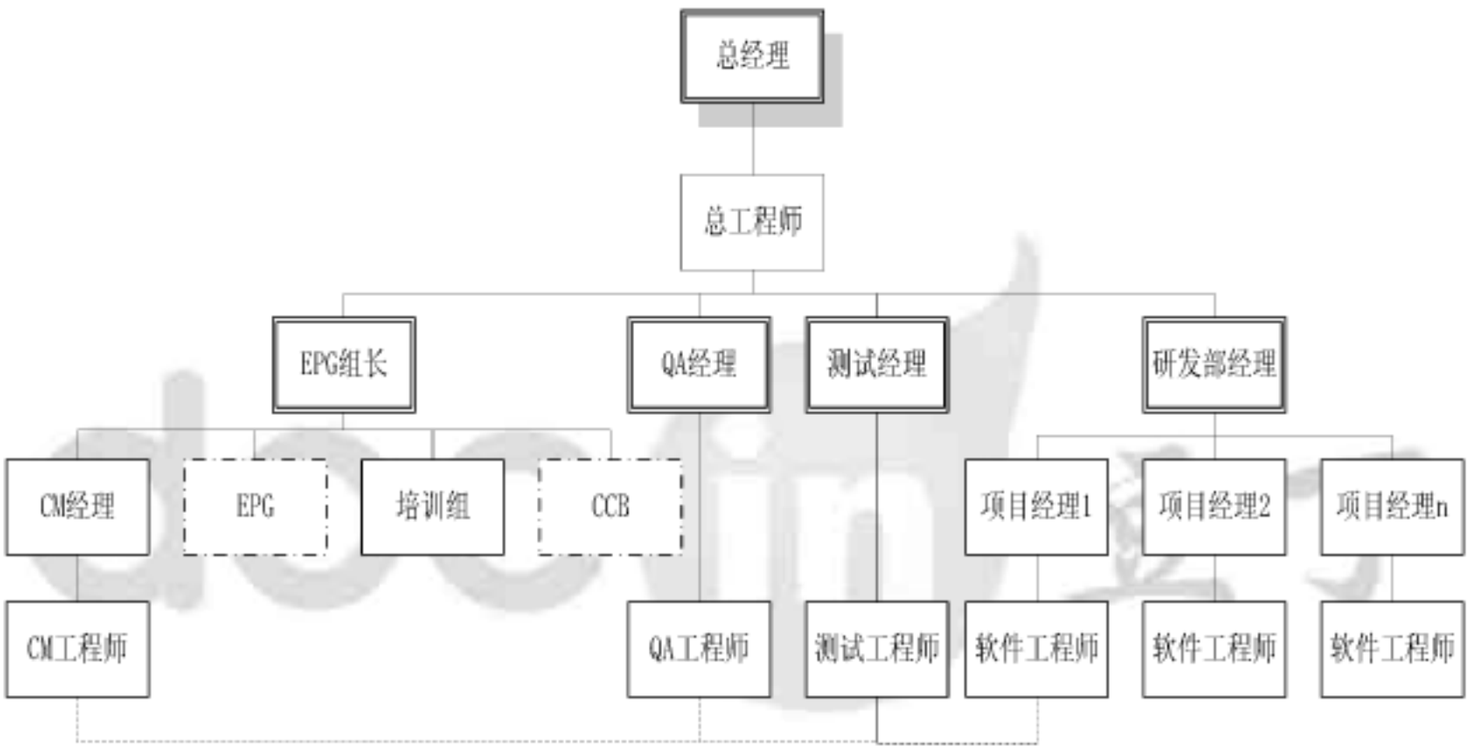
阶段		工作内容		工作产品	
立项		1.可行性研究/合同评审、签订	2.立项评审	1. 立项可行性分析报告 3.立项报告 5.立项通知书 6. 项目任务书	2.用户需求说明书（初稿） 4.需求和项目计划阶段工作计划
需求		1.编制并完善《用户需求说明书》 2.软件需求规格说明书编写	3.工作产品评审 4.需求跟踪及管理	1.用户需求说明书 3.用户需求跟踪矩阵	2.软件需求规格说明书 4. 需求变更申请表
计划		1.项目范围分析、工作分解 3.编制进度表 5.编写配置管理计划 7.计划评审、批准	2.估计规模、工作量等 4.评估项目风险 6.编写《项目开发计划》	1.项目开发计划 含：质量保证计划、CM 计划、风险管理计划和培训计划	2.评审记录
设计		1.概要设计 3.数据库设计	2. 模块计 工作产品评审	1.《概要设计说明书》 3.《数据库设计说明书》	2.《模块设计》 4.评审记录
实现与测试		1.编码 2.编制各类用户手册	2.单元测试	1.单元代码 3.单元缺陷管理列表	2.单元测试用例列表 4.单元测试报告
测试	集成测试	1.《集成测试计划》编制、评审 3.集成测试	2.《集成测试用例》编制、评审 4.《集成测试报告》编制、确认	1.集成测试计划 3.缺陷管理列表 5.评审记录	2.《集成测试用例》 4.《集成测试报告》
	系统测试	1.《系统测试计划》编制、评审 3.系统测试	2.《系统测试用例》编制、评审 4.《系统测试报告》编制和确认	1.系统测试计划 3.测试记录、缺陷记录	2.《系统测试用例》 4.《系统测试报告》
项目总结		1.代码复用总结 3.《项目总结报告》编制和评审 5.项目总结会议	2.各类手册评审和批准 4.产品/项目归档 6.项目结项/产品发布	1.产品及各类手册 3.产品基线建立和审计	2. 项目总结报告 4.评审记录

② 组织结构

在企业里，根据公司的商业目标，结合公司当前组织机构设置的实际情况，来设置相应的研发部门的组织机构，除此之外，在公司里根据实际需要一般还会设有市场部、财务部、

工程部、技术支持部、客户服务部等部门。在本书的案例，我们设置一个与软件过程管理直接相关的机构框架，具体如图 2-3 所示，其中 CCB 可以设置为项目级也可以设置为公司级，本书模拟的研发机构为 10 左右，设置为公司级。对于规模比较大的公司，可以按产品级来划分设置或设置为项目级。

图表 2-3 案例公司研发相关组织结构图



③ 制定机构商业目标

企业采用 CMMI 标准实施软件过程改进，必须服务于企业的商业目标。因此，企业高层管理者应该先提出机构商业目标。进一步，商业目标一方面要分解到各个部门，作为部门绩效考核的依据；另一方面则用于制定过程改进的目的和要求，用于定义过程，用于定义过程度量和质量度量等等。

④ 过程划分

按照 CMMI 标准的要求，结合国内一些公司的实际情况，我们将整个软件生命周期（从立项到发布，直到技术支持）划分为 24 个过程（如表 2-4 所示），本书中重点讲解了这 24 个过程中的 17 个，会在后继出版的书籍详细讲解另外 7 个过程。在企业进行过程改进时，可以根据公司商业目标及过程改进目标有重点的分批次执行。

图表 2-4 本书遵循的 CMMI 级别过程划分表

序号	CMMI 级别	过程名称	说明
1	二级	立项管理	标准中没有，自己增加过程
2		需求开发及管理	二级中涉及部分内容，即需求管理
3		项目计划	
4		项目跟踪及控制	
5		产品及过程质量保证	
6		配置管理	
7		采购管理	本书未涉及相关内容
8		项目评审管理	包含技术评审及管理评审
9		客户验收	标准中没有，自己增加过程
10		产品升级与维护	标准中没有，自己增加过程，本书未涉及相关内容
11	三级	软件开发过程管理	
12		需求开发及管理	全部内容
13		系统设计	
14		系统实现与测试	
15		系统测试	
16		培训管理	本书未涉及相关内容
17		项目总结	标准中没有，自己增加过程
18		风险管理	
19		决策分析与解决方案	
20		度量分析	
21	四级	过程定量管理	本书中涉及部分内容
22		项目定量管理	本书中未涉及相关内容
23	五级	缺陷预防	本书中未涉及相关内容
24		技术更新	本书中未涉及相关内容
25		过程改进	本书中未涉及相关内容

2.2 岗位角色职责

本书中根据上一节设置的研发机构，针对 CMMI 实际要求及开发过程管理、公司级过程管理的要求，对该研发机构中设置的角色职责划分如表 2-5 所示。这也是本书中讲解各类软件开发过程活动时角色对照表，实训时需要根据实训指导手册进行适当简化。

图表 2-5 常设岗位角色表

常设角色		职责简述
过程管理角色	工程过程组（EPG）	由相关业务部门的部门经理、质量保证经理、配置管理经理、技术专家组成，有一位组长。 EPG 职责： 1. 制定适合于本机构的过程规范； 2. 在机构范围内推广该规范（如培训、考核），评估机构过程能力等。 EGP 组长职责： 1. 制定过程改进计划并跟踪执行； 2. 向总经理提交 EPG 过程改进活动的报告（如进展报告、工作周报等）； 3. 向总经理汇报过程改经工作的问题，争取总经理的协助。
	质量保证小组（QAG）	由质量保证经理(QA 经理)和质量保证工程师组成。 质量保证经理职责： 1. 质量保证经理为每个项目指定一名质量保证工程师；对质量保证工程师提交的项目组内无法解决的不符合问题进行协调； 2. 监督规范的实施，确保所有项目以及相关部门准照规范开展工作； 3. 分析机构内共性的质量问题，给出质量改进建议和措施，协组 EPG 完善规范。 4. 对过程改进项目执行质量保证相关活动。
	配置管理小组（CMG）	由配置管理经理(CM 经理)和配置管理员组成。 配置管理经理职责： 1. 维护机构级配置管理库及过程资产；为每个项目指定一名配置管理员； 2. 依据文档化的规程，协助配置管理员制订 CM 计划，并审核 CM 计划；审计各阶段的配置管理活动报告； 3. 根据项目需要选择合理的配置管理工具，报 EPG 批准纳入过程资产库，定期组织培训； 4. 根据配置管理员提交的配置管理活动报告，定期进行度量、分析，形成分析结果，给出改进措施，实现配置管理过程持续改进； 5. 组织协调配置管理员与软件工程师或技术服务部门之间的工作交流与问题处理；
项目管理	总工程师	1. 是机构内所有项目的主管，对立项管理和结项管理有最终决策权； 2. 对 QA 经理提交的无法解决的不符合问题进行协调。 3. 审查所有的对机构外部的个人和组所作的软件项目承诺； 4. 组织协调跨部门或与客户的工作交流与问题处理；

常设角色		职责简述
角色	研发部经理	1. 监督项目经理的工作，审批项目经理的各种申请； 2. 参加评审会并审阅评审报告； 3. 负责监督软件过程规范的实施； 4. 参与软件、硬件、技术服务等软件相关阶段的工作产品、使用技术、工具的评审和审批，并给予必要的支持；
	项目经理（PM）	1. 向研发部经理或总工程师汇报工作； 2. 对项目进行规划、对进度实施监控、进行风险管理和需求管理； 3. 监督项目成员的工作，审批项目成员的各种申请及子计划； 4. 制定编码与单元测试、系统集成的阶段性计划 5. 参加评审会并审阅评审报告； 6. 配合质量保证工程师不合格问题的解决及跟踪，支持其工作； 7. 负责项目的度量工作。
工程过程角色	项目组成员	1. 项目组内除项目经理外的其他所有人员，包括以下人员：需求开发人员、系统设计人员、开发人员、测试人员。
	需求开发人员	1. 调查、分析并定义需求，撰写相应的需求文档，尽最大努力使需求文档能够正确无误地反映用户的真实意愿。
	系统设计人员	1. 根据需求文档设计软件系统的体系结构、用户界面、数据库、模块等，并撰写相应的设计文档。
	开发工程师	1. 根据系统设计文档，编写软件系统的代码； 2. 随时测试和检查自己的代码，及时消除代码中的缺陷。
	测试经理	1. 依据文档化的规程，为每一个软件项目制定测试计划，并按得到批准的计划开展活动； 2. 组织编写测试用例； 3. 根据项目需要选择合理的测试工具，报 EPG 批准纳入机构资产库，并定期组织培训；
	测试工程师	1. 从事集成测试、系统测试，负责参与项目开发各个过程工作产品的可测试性的审查和验证，及时发现、记录缺陷并验证缺陷等关闭活动； 2. 为项目编写集成测试及系统测试用例，并执行软件测试过程； 3. 项目测试结束后，编写测试报告提交测试经理；

常设角色		职责简述
支撑过程角色	配置管理员（CM工程师）	1. 依据文档化的规程，为每一个软件项目制定配置管理计划，从机构资产库中选择合理的配置工具，并按得到批准的计划开展活动； 2. 根据软件项目 CM 计划，建立配置库系统，识别将置于配置管理之下的所有软件工作产品； 3. 依据文档化的规程，对基线更改进行控制，定期形成更改请求摘要与状态报告，提交项目经理； 4. 依据文档化的规程，由软件基线库生成产品，并控制其发布； 5. 依据文档化的规程，记录配置项/单元的状态； 6. 编写标准的报告，记录 CM 活动和产品基线的内容，定期整理配置数据，形成 CM 活动报告提交项目经理及配置管理经理；
	培训组	1. 根据机构发展战略，总结出将来可能要有培训需求； 2. 定期或不定期地从项目组获得培训需求，从以上两种方式收集培训需求、确定培训计划，并实施该计划，撰写《培训评估报告》； 3. 维护培训资料库。
	产品维护人员	1. 为客户提供与产品相关的服务（如技术咨询），快速响应客户的要求，给客户一个满意的解答。 2. 纠错性维护：及时解决用户遇到的技术故障和消除产品中的缺陷； 3. 完善性维护：在资源允许的情况下，不断改善产品功能与质量。
	质量保证工程师（QA 工程师）	1. 根据《项目计划》制定《质量保证计划》； 2. 遵循已制定的计划、标准和规程，按照经过评审的《质量保证计划》，从第三方的角度周期性的监控软件开发任务的执行； 3. 通过《QA 审计报告》给项目经理和开发人员提供已识别出的质量问题并跟踪问题的解决过程，与项目组协商不符合问题的解决措施，给出质量改进的建议； 4. 向 QA 经理汇报项目组内不能解决的不符合问题； 5. 撰写并向 QA 经理和项目经理发布《QA 周报》，提供反映产品和过程质量的信息和数据； 6. 协助收集项目度量数据； 7. 参与项目相关评审活动。

图表 2-6 临时设立角色表

临时角色	职责说明
立项小组	由产品创作者（构思者）、业务专家、技术专家、市场人员组成；应有一位主席或组长； 1. 开展立项调查、产品构思、可行性分析等活动，全面考虑公司战略、效率、成本等各方面因素，撰写《立项报告》或《可行性分析报告》；

临时角色	职责说明
	2. 申请立项，并在立项评审会议上答辩。
立项决策委员会	由机构领导、各级经理、市场人员、技术专家、财务人员等组成，应有一位主席或组长； 1. 对立项活动进行评审，委员会投票决定是否同意立项。
评审小组	由机构领导、项目组成员，项目组以外的技术专家等组成；应有一位主席或组长； 1. 对工作成果进行正式技术评审，尽早地发现工作成果中的缺陷； 2. 对项目过程进行管理评审，给出决策建议。
配置控制委员会（CCB）	由一组负责评估和审批配置项的变更的人员（CM）组成，本案例中是由总工程师、市场部产品经理、研发部长、配置管理工程师、配置管理经理、项目经理等人组成； 1. 对配置管理各项活动拥有决策权； 2. 审批配置管理计划； 3. 对递交进来的所有变更请求进行审查、分析，从而决定如何处理这些变更请求，审批变更请求； 4. 基线建立的审批；产品发布的审批；

第3章 立项管理

内容提要：

- 立项管理简述
- 立项管理流程
- 立项管理活动
- 立项管理要点

立项管理（Project Establishment）在 PMBOK 体系中可以划入项目的启动过程。是结合国内软件开发企业实际管理的需要，我们提出的一个过程，需要在软件确定开发之前执行，CMMI 中并没有相关的过程域或实践对应。

从管理的角度来看，立项管理是属于决策的范畴，在公司运作过程中，不只有产品研发的立项，也有业务的立项。比如：市场人员得知某单位要上一个信息系统（在客户关系管理中叫销售机会或销售线索），那么此销售线索是否值得进一步跟踪和做工作，以及如何进一步做工作，使我们拿下这个业务的可能性向现实性转化，签下单子？这个问题的正确决策，并不简单，因为其既有合同能否执行或公司是否有能力执行的问题，也有在合同签订之前的花费问题。如果小看了这个问题的重要性，可能会给市场营销工作带来了许多问题。例如，空忙了一阵子，当了别人的陪衬，最终别人拿走了单子，这还是轻的，何况也不能要求每个单子都我们拿，每次出击都成功；再如，客户什么要求都答应、什么条件都接受，单子签下了，最终是亏本生意。当然，这在公司里属于市场营销管理的内容，本章所讲述主要还是公司有新产品研发意向，或者产品升级意向时怎么来进行立项。对于合同类项目，只要市场部门签订了单子（合同）之后，对于研发部门即认为立项成功，必须进行软件研发。

3.1 立项管理简述

我们通过规范企业里的研发立项过程，主要是想达到如下目的：

- 降低项目投入的风险，避免研发项目的随意性；
- 加强项目的目标、成本和进度考核，提高项目成功比例；
- 杜绝不实际的研发项目展开，避免公司资源浪费。

对于销售立项则不是如此，通过销售立项的审批，是想达到如下目的：

- 不放过可以赢利并且能做或可以做的项目，拒绝亏本并且不值得亏本或者暂时做不了或不值得做的项目；
- 比较清楚地了解用户需求和客户信用状况，减少合同签订后的变更；
- 确定一个比较合理的报价。

为了做出正确的立项决策，我们在有了研发意向或销售意向的时候，应当拿出一个比较全面的项目陈述，比如：《立项报告》、《立项可行性分析报告》等。在其中尽可能全面的考虑方方面面的因素，权衡得失。例如：

- 这个产品研发到底有多大把握？【当然不能要求 100%把握，如能这样的话，就成神仙了。】
- 这个项目可以做吗？技术积累够吗？现有人力调度得过来吗？【资源永远是紧张的，问题是调度。】
- 项目需求明确吗？合理吗？
- 研发人员说至少要半年，用户或公司希望要求 2 个月，怎么办？抽调人员，别的项目用户会怎么反应？
- 这个产品值得研发吗？成本要 100 万，而市场销售可能只有 80 万，最后亏本，谁买单？
- 好，以上这些问题都有答案了，或者多是“庸人自扰，杞人忧天”，那么下一步工作由谁负责，哪些人去做？如何做？

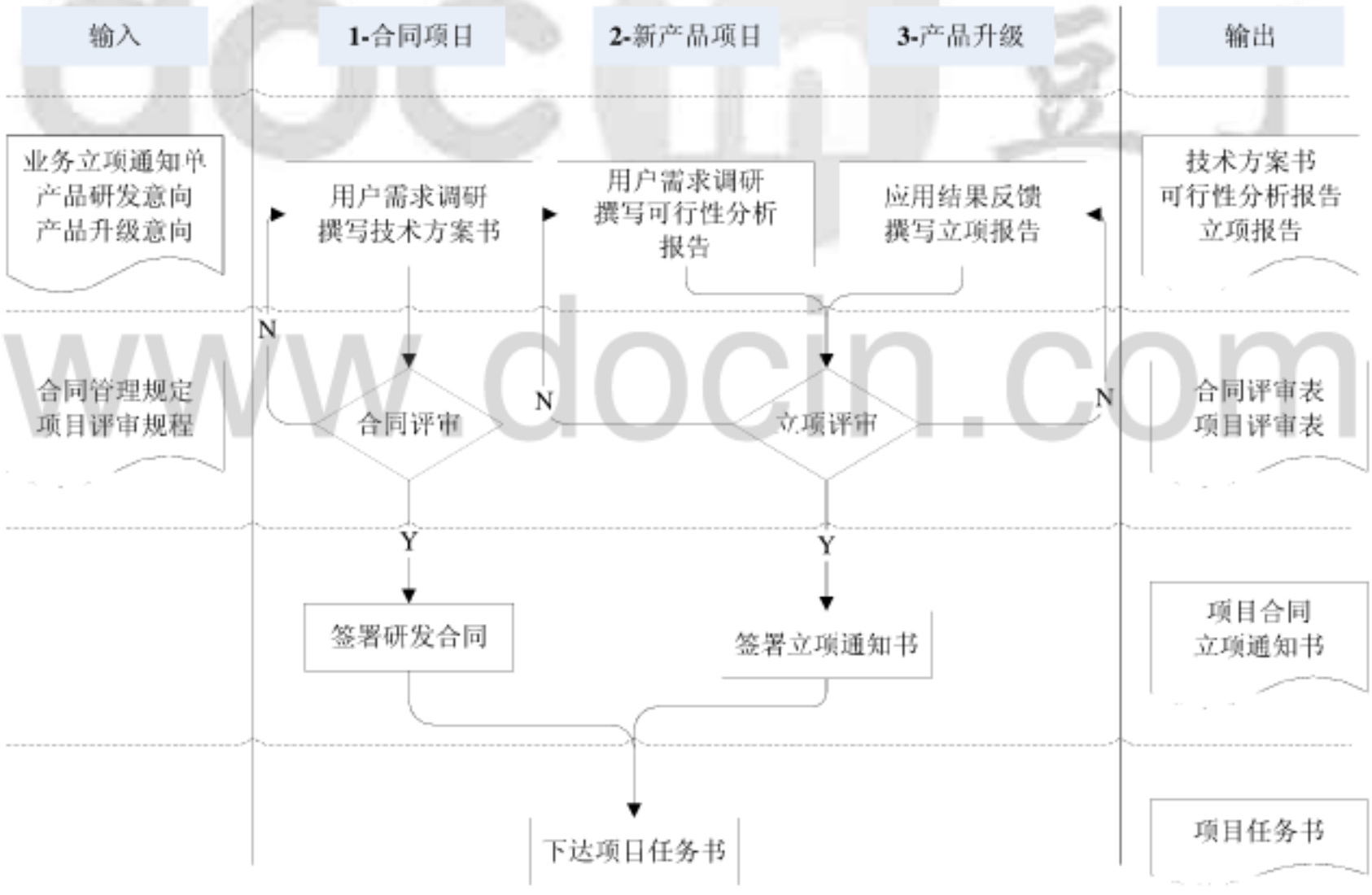
在具体操作过程中，对于合同类研发项目、新产品研发项目均需要进行立项管理，对产品升级类项目可以根据产品升级的程度来确定是否要进行立项，建议项目周期比较短或投入比较少的项目可以不进行立项，而是直接给项目小组下达任务书。一般公司会要求，项目立项过程必须有正式的文档提交，并通过正式评审。

【说明：评审的流程请参见第 4 章项目评审管理相关内容。】

3.2 立项管理流程

对于不同类型的研发项目，相应的立项流程也有所区别，在本书中我们共给出了三类项目的立项流程，具体如图 3-1 所示。

图表 3-1 立项管理流程图



【说明：合同评审和立项评审若通不过，则会有两种可能，一种为重复前一步工作，再次评审；一种为终止销售立项（即该业务不再跟踪及签订合同）或者研发立项。】

3.3 立项管理活动

1、谁提出项目立项?

合同类项目

市场部门若有定制开发类的业务，根据公司确定的市场管理相关规定提出需要技术部门给予配合的申请（一般为《销售立项通知单》，在该单据里会写明将要跟踪的业务基本情况，需要什么样的技术人员配合等内容）。然后，由总工程师从研发部门指定专门的技术人员，配合业务人员做技术方案。注意：在此时选择技术人员时，应当考虑到该人员的综合能力，因为如果该业务合同一旦签订，此人就是该项目的项目经理。

新产品研发类项目

在两种情况下可以提出新产品研发，一是公司对某一产品有研发意向；二是市场部门与技术部门通过讨论，要开发某一新产品。此时，则由总工程师确定立项方式，并指定专人或亲自负责，做立项的前的准备工作。注意，此人通常会为项目立项之后的项目经理，所以在指定人选时需要考虑综合能力。主要是完成立项的可行性分析，并且根据项目的类型确定是否进行技术预研，对于需要用到新技术、新工具及新平台的产品研发，应当进行技术预研。

【说明：研发意向一般是公司对市场分析的基础之上，形成要开拓某一市场而研发产品的意愿。作为立项提出的依据，可以是会议记录，也可以是相关市场分析报告。】

产品升级类项目

根据市场及用户的反馈，由研发部经理或总工程师确定是否进行升级研发，由自己或指定专人负责，做立项准备工作。

【说明：市场及用户反馈一般来源于公司对产品已有用户做的使用情况调查、对本公司产品及同类产品进行的市场调研分析、公司售后服务部门从客户处得到已有产品的使用报告或问题（故障）报告等。】

2、怎样提出项目立项？

合同类项目

由总工程师指定的技术人员参与业务的商务谈判，并撰写技术方案书，然后按公司合同

评审管理流程，提出评审。

新产品研发类项目

由总工程师指定的负责人为主来撰写《立项可行性分析报告》或《立项报告》，该立项负责人通过协调技术部门、市场部门、财务部门等共同来完成资料的编写。根据项目的规模来确定，对于投资比较大或者风险比较大的项目，建议编写《立项可行性分析报告》。具体什么叫“投资比较大或风险比较大”，各公司有各公司的定义方法，一般由公司的业务领域、管理制度来确定。

【建议：学习过程中的实训类项目，第二、第三类学员实训时编写《立项可行性分析报告》和《立项报告》，第一类学员实训时只编写《立项报告》。】

产品升级类项目

由研发部经理或总工程师指定的负责人为主来撰写《立项报告》；若升级规模比较小或研发周期比较短，也可以由总工程师直接下达《项目任务书》。具体什么叫“规模比较小或研发周期比较短”，各公司有各公司的定义方法，一般由公司的业务领域、管理制度来确定。

【说明：《立项报告》中至少应包含的内容为：项目范围及目标、验收标准、技术规划、计划进度、成本预算、风险预计等。】

3、谁进行立项评审？

合同类项目

根据合同管理里的《合同评审办法》进行评审，作为公司管理的一个重要环节，合同管理各公司会有比较明确的规定，在签订合同之前，会对合同的技术内容、可能的风险、违约责任、己方履约能力、收款条款等进行评审。

其他项目

立项负责人在完成《立项可行性分析报告》或/和《立项报告》后，向总工程师或研发部经理提出评审要求。

总工程师或研发部经理确定参加评审的人员，评审两天前立项负责人把相关资料交到评审人员手中。具体是评审前两天，还是几天，可以根据项目的规模、项目的重要度、项目整

体周期来确定。然后召开评审会议，具体评审会议怎么召开？怎么组织？公司里一般有管理评审的相关规定，可以参见第 4 章 项目评审管理的相关内容。

4、怎么进行评审?

合同类项目

按照各公司自己确定的《合同评审办法》执行，本书不做详细的讲解。但只要合同评审通过，对于研发来说，就认为立项通过，即可进入一下环节（总工程师根据与客户签订的合同填写并签发《项目任务书》）。

其他类项目

1. 参加评审的人员收到相关立项评审资料后（《立项报告》和/或《立项可行性分析报告》），仔细阅读并填写《预审问题清单》，在评审前交给立项负责人。
2. 立项负责人根据《预审问题清单》中提出的问题进行修改或准备答辩资料。
3. 举行评审会议，具体评审会议召开的规定请参见第 4 章 项目评审管理相关内容。

【说明：《预审问题清单》是为了缩短评审会议、提出评审效率来制订的，目的是让参加评审的人员在评审会议召开之前对要评审的内容进行仔细阅读，并把发现的问题或疑问填写进《预审问题清单》中，由答辩人对这些问题在会议召开前进行修改或准备答辩资料。】

5、谁签发?

- 如果立项评审通过则需要签发《立项通知书》，若未通过，则在《项目评审表》中写明处理方式，一般分为不接受和变更两种情况，具体请参见项目评审表的内容。
- 对评审中出现的问题，根据《项目评审表》中的内容进行跟踪修订情况，其中对每个问题的修改情况，必须由验证人进行跟踪，并把修改及验证所花的工作量记录进该表。
- 立项申请人编制《立项通知书》，报总工程师批准，由总工程师根据项目规模和费用的大小上报公司总裁或自己审批。具体谁来审批，各公司根据项目类型、规模来确定，比如某公司给出如下规定：
  - 项目估算投入≥20 万的由公司总裁审核批准；

- ◆ 项目估算投入＜20 万的由总工程师审核批准；

■ 总工程师根据立项通知书填写并签发《项目任务书》，合同类项目根据与客户签订的合同填写并签发《项目任务书》。

【说明：《项目任务书》中包含项目经理、项目组成员、QA 人员（质量保证工程师）、CM 人员（配置管理员）及 CCB（配置控制委员会）人员，具体内容请见该章对应的实训指导。】

3.4 立项管理要点

在立项过程中，我们应当对可能遇到的情况做充分的预计，对项目的工作量、成本和工期进行相关科学的估算，切忌使用“拍脑袋”的方式进行估算。对于在项目开展过程中可能遇到的风险也应当识别，并给出一定的应对方案。

根据项目类别的不同，对工作量、成本、工期的估算方法当然要有区别，但是，以下几点对所有的项目都应该得到重视：

- 应该估计整个生命周期每个阶段、各项活动、各类相关人员必须为该项目投入的工作量和成本；
- 工作量和成本估计，如果涉及多个部门，应由相关部门分别估计然后由总工程师汇总，不能一个人说了算，也不能全由研发人员或市场人员说了算，更不能随便估算就交差。
- 既要计入直接成本和工作量，也要考虑间接成本和工作量。
- 为项目开发的需要而购买的软硬设备，或者项目交付后项目成果可以进一步作为公司的新产品或新版本，或者已经有了初步成果而通过新项目则可以节省产品开发所需的投入等。在这些情况下，应妥善分割成本和工作量，全部成本和工作量完全分摊在一个项目头上，显然不合理。比如：在项目研发期间需要购买一台服务服，但该服务器在此项目结束之后还可以继续使用，那么把成本全部算到该项目上显然不合理。
- 工作量到成本的折算，可用上一个年度整个部门全年人均开支，也可用上一个季度

的人均开支，也可用各类人员的人均开支。

- 业务费、项目奖金以及其他必须的特殊开支，应计入（工作量之外的）成本。
- 应考虑风险储备以及项目相关各方沟通协调所可能必需的工作量和成本。
- 要为必然会有的需求变更引起的工作量增加留有余地。

由于立项是一个比较重要的决策活动，所以在立项期间的风险识别及控制风险是非常重要的。比如：当前一些房地产商，由于看到一时的市场比较好，没有考虑到自身的资金能力、国家政策调整等可能的风险，导致拍到土地之后，后续开发成了问题，量严重的情况是该土地被回收。试想想，该项目立项时管理者是否考虑到很有可能发生的风险了吗？对于软件产品研发，也有类似的案例。那么在立项时，对于风险可以按照以下方针来处理：

- 应从思想深处树立风险意识，“风险与机会同时存在，收益和风险始终相伴”。项目风险管理与项目本身的工作同样重要；风险管理是任何项目所必然包含的一项工作内容。
- 不能为躲避风险而错失良机，也不能因漠视风险而使机会变成损失；应该敢于直面风险，“审时度势，权衡取舍，敢于进取，有备无患”。
- 任何项目，确定立项之前，必须进行风险识别，特别是客户信用评估，针对首要风险制定应对措施，估计风险储备并将其计入项目成本。
- 在识别的基础上，分析风险，估计风险属性（发生的可能性和后果的严重性），按行业的具体规定评价风险并排序。
- 针对主要风险（即排在前面的若干项风险，例如前 5 项，前 10 项风险）进行风险策划，确定应对策略，制定应对措施（以至风险管理计划）。
- 明确风险跟踪责任。

【说明：关于风险管理的详细内容，请参见第 9 章 风险管理。此处是重点强调在立项阶段，我们应当进行哪些风险的管理活动。在立项通过之后，此处识别的风险做为项目开发过程中风险管理的一部分进行跟踪与控制。】

第4章 项目评审管理

内容提要：

- CMMI 对应实践
- 项目评审简述
- 评审管理活动

在研发过程中为了保证研发过程及研发产品的质量，都会有大量的评审活动，在我们进行项目评审时，有以下几点原则需要遵循：

- 在项目开发计划中确定各阶段需要进行评审的工作产品及要举行的评审活动（比如里程碑评审、阶段评审等），评审按计划进行，并且有文档化的评审记录。
- 评审之前，项目经理组织人员准备评审相关资料，并提供待评审的工作产品的评审标准；同时发送通知邀请相关评审人员参加，正式评审之前还需要进行预审。
- 评审活动需要有总工程师/研发部经理和有同类产品开发经验的人参与。
- 关注受评审的工作产品，识别并解决工作产品中的缺陷，在项目组成员和评审员之间就工作产品的内容达成一致意见。

在有的企业里根据被评审的对象及目的把评审分成两类，管理评审和技术评审。两类评审目标对象分别如下：

- 管理评审是在项目组内部实施的对管理类工作产品（如：《项目计划》）、项目约定、里程碑的评审，从而保证项目制定出切实可行的计划，避免做出超出项目能力范围之外的约定或承诺，同时把控上一里程碑阶段项目完成的状态，决定是否可以进入下一个里程碑阶段。
- 技术评审，有时叫同行评审，是指项目组成员邀请同行技术专家对工程过程技术类工作产品的评审，尽早地发现工作产品中的问题和缺陷，并帮助项目组成员及时消

除问题和缺陷，从而有效地提高产品的质量。

为了使大家更好的理解项目中评审的过程及重要性，在本书中对管理评审及技术评审进行了简化，统一到项目评审管理中。在开发过程中，所有评审类的活动均使用该章所讲述的内容进行操作。

4.1 CMMI 对应实践

在 CMMIV1.2 版本中，直接与评审相关的实践，在通用实践中有一个；在验证（Verification，简称 VER）过程域中有一个特定目标，即“执行同行评审”（Perform Peer Reviews）；在项目跟踪与控制（Project Monitoring and Control，简称 PMC）过程域中有两个特定实践。分别描述如下：

通用实践（**General Practice**，简称 **GP**）：

GP 2.10 Review Status with Higher Level Management（高级管理者评审状态）。重点关注，高层管理者应参与评审过程活动、状态和结果，并解决争议问题。目的是：为高层管理者提供对过程的必要的可视性。此处所说的高层管理者包括机构内部那些比直接负责管理该过程的管理者层次更高的人员，如总工程师，特别是那些制定机构方针和过程改进方向的管理者，但不包括那些负责对该过程进行日常监督和控制的人，比如项目经理。不同层次的管理者对过程信息有不同的需要。此类评审有助于高层管理者对过程策划和实施情况作出正确的判断或决策。高层评审的频率可以定期也可以事件驱动。

VER（验证）过程域：

SG2 Perform Peer Reviews（执行同行评审），目的是对选定的工作产品进行同行评审。

同行评审是以同行的角度识别工作产品中存在的缺陷及需要变更的地方，参与的人员必须是对此类工作比较熟悉的同行。同行评审是主要是适用于由项目组开发的工作产品，也适用于支撑小组开发的一些文档、跟踪记录等产品。他是一种重要且行之有效的验证方法，公司应当建立一套行之为效的同行评审管理体系。

SP2.1 Prepare for Peer Reviews（同行评审准备），同行评审的准备工作，通常包括：识

别受邀参与每一工作产品审查的人员、识别必要参与的主要审查人员、准备及更新同行审查需使用的数据，例如：检查表、审查准则及同行审查进度等。

SP2.2 Conduct Peer Reviews（执行同行评审），主要是针对所选定的工作产品进行同行评审，并由同行评审的结果识别问题。执行同行评审目的之一，即是能及早发现并去除缺失，并且应当随着工作产品的开发逐步进行。同行评审重点应为被审查的工作产品，而非工作产品的制作人员。评审过程中发现的问题，应与工作产品的主要制作人员沟通，以便修正。可以针对需求、设计、测试、实现活动的关键工作产品及特定的计划工作产品来执行同行评审。

SP2.3 Analyze Peer Review Data（分析同行评审的数据），分析同行评审的准备、执行及结果数据。典型的数据通常包括产品名称、产品规模大小、评审成员、评审类型、每一评审人员的准备时间、评审会议时间、缺陷数、缺陷类型及发生处等。其它可能搜集的工作产品信息，例如：开发阶段、所检查的操作模式及被评估的需求。

PMC（项目跟踪与控制）过程域：

SP1.6 Conduct Progress Reviews（执行进展评审）。定期审查项目进展、性能和遇到的问题，主要是为了：定期与项目相关各方沟通分配给他们的工作任务和工作产品的状态；对控制项目而收集和分析的度量结果进行评审；确定并记录重大问题和与计划的偏差；记录变更请求以及在任何工作产品和过程中发现的问题；记录评审结果；跟踪变更请求和问题报告直至关闭。

SP1.7 Conduct Milestone Reviews（执行里程碑评审）。在项目里程碑处评审项目成果及其完成量，通常为正式评审。主要是为了：与项目相关各方一起，执行里程碑评审；评审项目承诺、计划、状态和风险；确定和记录重大问题及其影响；跟踪采取的纠正措施直至关闭。

从评审的内容上来看，我们可以把上面 GP2.10、PMC 的 SP1.6、SP1.7 划分到管理评审中去，因为他们均不侧重于项目中的技术文档。工程过程域里，但无论是需求、分析、设计、编码还是测试方案（用例），大都要求进行同行评审，我们可以把这类评审（对应于 VER 的

SP2.1、SP2.2、SP2.3）划分到技术评审中。

4.2 项目评审管理简述

项目评审就是从管理或技术的角度审查已开展的项目活动及产生的工作产品，从中找到不符合项目整体目标或期望之处；或者对即将开展的项目活动计划进行审查，通过认可计划，为项目开展提供承诺。在公司里执行项目评审的目的主要有以下三点：

- 为项目在研发过程中各阶段需要进行的评审活动（包括同行评审和管理评审）提供实际的评审操作流程，规范各阶段的评审工作。
- 规范项目中评审计划的执行方式和方法，提高项目评审的效率。
- 方便项目组成员和评审员之间就工作产品的内容达成一致意见。

【说明：在开发过程中，对于需要进行哪些评审，应当在《项目计划》里进行明确，也可以写成专门的评审计划，比如《技术评审计划》——明确所有与技术相关的工作产品评审的时机、参与人员、评审通过的准则等等。】

为了项目开发过程中更好的执行项目评审，按评审要求的严格程度划分了三个类别，分别为：正式评审、非正式评审、审核。另外，也可以按被评审的工作产品来分类，分为技术评审与管理评审两类。在本书中采用第一种分类方式，具体解释如下：

- 正式评审—软件需求、项目计划、项目验收、项目里程碑、项目立项等均需经过正式评审。
- 非正式评审—适用于除需求和验收以外的工作产品评审；项目经理根据项目的类型，选择工作产品的评审方式。
- 审核—不太重要的工作产品或人力资源不足的小型项目。

在进行项目计划时，需要明确项目中各类活动及工作产品的评审计划，这时就需要确定每类活动或工作产品评审的类别，在制定评审计划时，可以考虑如下几个影响因素及要求：

- 根据项目级别项目经理可以确定需要进行的评审活动。
- 根据《机构标准软件过程》来确定需要进行的评审活动。《机构标准软件过程》一般

是对公司开发出来的机构标准过程集合的高层次描述。其中描述了机构标准过程的组成，以及过程体系结构、文档体系结构的说明。对所有即将使用 OSSP 的工作人员熟悉和掌握整个 OSSP 结构、内容及如何使用该过程的指导书，是了解整个 OSSP 的过程体系结构的窗口。

- 考虑项目的类别。本书中我们把项目分为：新产品研发类项目、产品升级类项目、合同类项目三类。
- 立项评审之外，项目中其他评审均是一样的。可以规定新产品研发类项目、产品升级类项目立项必须通过正式评审；合同类项目合同签订后即认为立项评审通过[3]。

4.3 评审管理活动

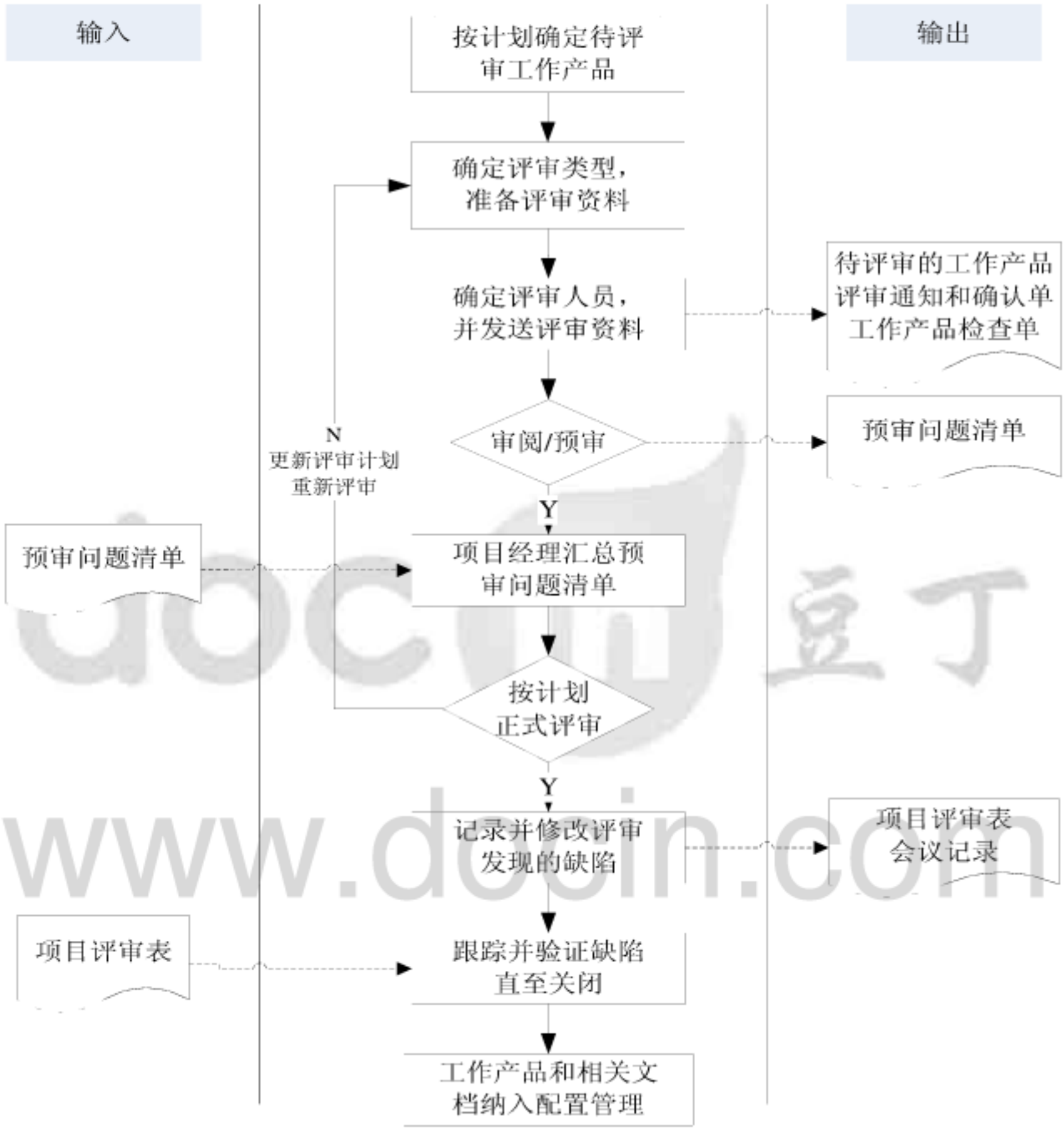
4.3.1 项目评审流程

项目里各类评审应当按计划进行，在评审管理流程中，最重要的一环是制订评审计划。以后的评审工作，均应当在该计划下来执行。为了保证评审的效果，在每次评审之前必须提前准备好待评审工作产品相关的资料，交由参加评审的人员提前阅读，以发现问题。对于正式评审，还需要评审人员在阅读评审资料时给出《预审问题清单》，以减少在开评审会议的用时。具体的操作流程参见图 4-1，对评审流程的操作步骤讲解会在后继段落中给出。

在公司里实际开展评审活动的时候，往往效果不理想。比如：需求分析的评审，好象大家的开评审会议的时候也没有提出什么大的问题，然后评审就通过了，该签字的也签字了。但是项目组在开发过程中，发现需求实际上存在很大的问题。那么，出现了这种情况，我们只能说是评审的效果或目的没有达到。为什么会如此呢？作者在从事几年的主管软件研发和 CMMI 咨询中发现，无非是如下几种原因导致的：一是，在选定评审小组成员的时候，组员没有达到技术评审要求的“同行专家”的水平，他自己对要开发的系统的需求或所处的领域都不清楚，他哪能提出什么问题呢？只能最后在评审结果上签字了事。二是，参加评审的人

[3] 此处的项目类型及对评审的规定只是示例企业里的内部规定，不同企业根据组织形式及产品类别的不同，对评审的要求也不同，读者在学习及应用过程中需要注意。

图表 4-1 项目评审流程图



员没有花一定的精力提前去了解要评审的资料或工作产品，也就是一个责任心的问题。虽然大部分企业对评审都有要求，要求参评人员提前对待评审工作产品仔细研究并提出问题。但由于，参加评审并不是他们的主要工作，他们有自身的工作需要完成，参加评审也不算他们的工作量，所以就出现敷衍的情况，效果也就可想而知了。三是，评审的组织比较仓促或比

较乱，评审会议组织者自身对要讨论的主题掌握不清。参加评审的人员没有时间去仔细阅读待评审的工作产品，或者是评审会上，针对一些无关紧要的问题进行花大量的时间讨论，由于用时太长，不得不了了之结束评审，这些都是评审组织导致的。作者曾遇到过这样的一件事，某公司研发部在进行需求分析评审时，针对文档格式讨论了一个多小时。这家公司的开发过程本身就不规范，没有统一的需求分析模板，在需求分析人员编写软件需求规格说明书的时候，采用了其他公司的一个模板，其他参与评审的人员认为不适合。而此时评审主持人员也没有及时打断讨论，导致原计划三个小时的需求分析评审会，实际用于软件需求规格说明书的评审只有一个小时，其评审效果也就可想而知。

4.3.2 编制项目评审计划

从流程图（图 4-1，见上页）上可以看出，编制项目评审计划是项目评审管理的第一步。无论是什么类型的项目，研发立项通过之后，在项目计划初步时就需要确定项目评审计划，并且将项目评审计划做为项目开发计划的一部分进行文档化（编写进项目开发计划）。根据裁剪的软件生命周期、软件开发过程，描述需要评审的工作产品、评审方式、评审时间、评审人员组成。但是，在项目开发过程中，每个里程碑[4]处必须进行正式评审。项目中所有的评审结果记录在《项目评审表》中，并对结果进行相应处理。每阶段或事件驱动的比较评审计划和实施情况，并根据实际情况调整评审计划。

【说明：裁剪的软件生命周期，一般是根据公司的 OSSP 及裁剪指南来确定，有的公司里要求在专门的项目过程定义（PDP）文档中明确。在剪裁指南中，一般给定项目的特征值，根据这些特征值进行裁剪。比如项目工期、计划最高投入工作量、项目类型等。详细内容参见本书第 5 章 项目计划初步和第 20 章 软件工发过程管理中的相关内容】

对于在开发过程中需要评审的内容，可以参照下表（图表 4-2，见下页）来定义，在实际使用过程中，需要根据对《机构标准软件过程》的裁剪情况来确定对哪些工作产品进行评审。

[4] 里程碑——在项目中的含义为完成阶段性工作的标志，不同类型的项目里程碑不同。

图表 4-2 项目评审建议表

项目类	工作产品列表	评审方法	成果	责任人
新产品研发类项目	项目开发计划	正式	项目评审表	项目评审小组
	用户需求说明书			
	软件需求规格说明书			
	概要设计说明书			
	系统测试用例	非正式		
	集成测试用例			
	数据库设计			
	详细设计说明书			
客户定制或合同类开发项目	项目开发计划	正式	项目评审表	项目评审小组
	用户需求说明书			
	软件需求规格说明书			
	概要设计说明书	非正式		
	系统测试用例			
	集成测试用例（可选）			
	数据库设计（可选）			
	详细设计说明书（可选）			
产品升级类项目	项目开发计划（开发和管理计划）	非正式	项目评审表	项目组成员，相关人员
	用户需求说明书(可选)			
	需求规格说明书	正式		项目评审小组
	概要设计说明书（可选）	非正式		项目组成员
	系统测试用例（可选）			项目经理
	详细设计说明书（可选）			项目经理
	概要设计说明书（可选）	非正式		项目组成员
	系统测试用例			项目经理
所有项目	里程碑评审	正式	项目评审表	项目经理
	用户手册	非正式		文档人员
	项目总结报告	正式		项目经理

【说明：此处的产品列表因项目类型的不同，会有所是不同，通过项目定义过程（Project Defined Process，简称为 PDP）来确定】

在制订评审计划是，需要明确每次评审计划参加的人员，千万不要在评审时临时确定人员。评审人员的确定请遵循如下三点准则，准备每个阶段或每类工作产品的评审人员组成，可以在表 4-3 的建议基础之上进行裁剪、调整确定。

- 为处于项目不同开发阶段的工作产品确定参与评审的成员名单和候选人员名单；
- 质量保证工程师参与正式评审，有选择的参加非正式评审，审核过程不用参加；
- 评审人员[5]在项目计划里明确。

图表 4-3 主要工作产品项目技术评审人员组成建议表

评审阶段	评审对象	建议评审小组组成
立项	《立项报告》	总工程师，研发部经理，项目经理，同行高级经理，销售人员，其他人员
项目计划过程	《项目开发计划书》	总工程师，研发部经理，项目经理，同行项目经理、系统设计人员（系分人员），测试人员，质量保证工程师，销售人员，其他人员
	《测试计划书》	项目组成员，质量保证工程师，测试人员，其他人员
	《质量保证计划》	项目经理，QA 经理，质量保证工程师
	《配置管理计划》	项目经理，质量保证工程师，CM
需求分析	《用户需求说明书》	系统设计人员（系分人员），项目经理，系统测试人员，质量保证工程师，用户代表，业务专家
	《软件需求规格说明书》	系统设计人员（系分人员），项目经理，系统测试人员，质量保证工程师，用户代表，业务专家
设计过程	《概要设计说明书》	系统设计人员（系分人员），程序员，项目经理，系统测试人员
	《详细设计说明书》	系统设计人员（系分人员），程序员，系统测试人员
	《测试用例》	测试工程师，程序员（单元测试）或系统测试人员，质量保证工程师
项目验收（内部）	项目成果	总工程师，研发部经理，项目经理，同行高级经理，其他人员

【说明：此表格中的人员可能会因公司的不同而不同。】

对于管理评审，比如里程碑处的评审等，参加的人员除了项目组、同行专家等技术类人员之外，还需要增加公司具有管理决策能力的人员参加，必要时可以请总经理参加。

[5] 在一些企业里，可能把计划参与评审的人员写入干系人计划中去。

4.3.3 正式评审

正式评审的目的是对须形成基线的配置项进行评审，发现和标识产品缺陷。由仲裁者确定评审结果，缺陷的修改工作被正式验证，缺陷数据被系统的收集并存储在配置库中，参加评审人员在评审结束后签字确认。

一、评审前确认和通知

1. 项目经理确认待评审的工作产品是否已具备评审条件，可根据评审对象的规模确定评审分一个或几个阶段进行，或者根据评审对象、内容的深入分层次进行多次评审。
2. 项目经理填写《评审通知和确认单》[6]，在评审会议前 2-3 天把评审相关资料提交给参加评审人员及质量保证工程师，并与参加评审人员协调，明确评审人员在评审会议中的角色，确定具体评审时间。
3. 项目经理在《评审通知和确认单》的会议进程安排一栏中确定本次评审会议所需时间及具体安排。

二、预审阶段

1. 评审参加人员明确了解他们在评审会议中的角色，在收到评审资料后对待评审工作产品的内容进行详细预审，发现存在的缺陷和问题并分类整理，填写《预审问题清单》。
2. 评审参加人员在评审会议前 1 个工作日将《预审问题清单》反馈给项目经理。
3. 项目经理把《预审问题清单》反馈给作者，并提交给质量保证工程师。
4. 作者根据《预审问题清单》对需要评审的工作产品进行修改，或准备评审答辩资料。
5. 质量保证工程师检查评审组成员是否已经有充分的准备，并收集评审员的评审工作量。

三、正式召开评审会议

1. 会议时间控制在 2-3 小时，人员低于 5 人；主持人宣布注意事项；作者花 5-10 分钟

[6] 《评审通知和确认单》目的是确认参加评审的人员；主要内容为明确被评审的内容，参加人员及时间。不一定局限于特定格式，达到以上目的即可。本章不对此表单提供模板及填写指导说明。

介绍项目背景及本次评审工作产品的主要内容。

2. 每个评审参加人员花一定的时间指出问题，并和作者确定问题和定义问题的严重程度。
3. 主持人控制整个会议的进程；当出现难以确定问题时，由仲裁者确定处理方式。
4. 记录员详细记录各个缺陷的情况，仲裁者将指派作者和评审参加人员在会后处理评审会议中未能解决的问题。
5. 主持人宣布评审结果，评审参加人员通过讨论，就评审结果达成一致意见；记录员形成《项目评审表》，评审人员签字。
6. 项目经理在批准人一栏中签字批准；如果评审结论“需要次要修改”，则确定验证人，并确定作者完成修改的时间。

四、评审结果追踪

1. 《项目评审表》作为作者修改的参考；完成问题修改后提交给项目经理。
2. 项目经理把工作产品、《项目评审表》提交给验证人；验证人进行验证并签字。
3. 项目经理把验证签字后的《项目评审表》递交给质量保证工程师。
4. 质量保证工程师检查作者是否完成修改任务，并且修改后的工作产品得到验证人的检查、确认后，质量保证工程师在项目评审表中签字确认。
5. 评审中产生的相关文档由项目经理统一提交给配置管理员，由其统一纳入配置管理，放进配置管理库。

五、过程审计

质量保证工程师在正式评审结束后，根据《QA 阶段审计报告》中的对评审过程是否符合机构制定的规范进行审计，形成 QA 阶段审计报告，发现评审中产生的问题，持续改进评审流程。

六、数据度量

在每次评审完成后，质量保证工程师在《项目度量数据库》中的《产品评审度量》记录评审的数据，内容包括：评审工作产品名称、工作产品规模、评审次数、评审人员数、评审

时间、评审发现的问题。

【说明：此处均有质量保证人员的工作，具体什么含义会在第 18 章 产品及过程质量保证章节中讲解，当前只了解几个相关表格的填写内容即可，对于不理解之处，各小组的质量保证人员可以教师在实训课上沟通。】

4.3.4 非正式评审

非正式评审：由非作者本人的个人或小组对产品执行详细的检查。目的是审查工作产品是否有错误、是否违反开发标准以及是否存在其他问题。它标识了产品和规格与标准的差异或在检查后提供了建议的方法。参与者(不一定包括软件质量保证工程师)包括作者(不属于检查者)，熟悉被检查技术内容的人员(一个或多个)。

工作步骤

1. 作者完成工作产品，申请进行非正式评审。
2. 实施非正式评审，评审过程由项目经理决定，由项目经理自己或指定资深组员（统称为审查人）对作者提交的工作产品进行审查。
3. 审查人对工作产品提出问题并分类整理，填写《项目评审表》。然后，就检查出的问题向作者提问，作者回答问题，双方要对每个问题达成共识（避免误解）。并为这些问题定义解决方案。
4. 审查人详细记录每一个已达成共识的问题，记录问题的位置，简短描述问题并对其进行分类。
5. 确定结论：项目经理给出评审结论和意见，总结整理《项目评审表》。
6. 作者根据《项目评审表》中提出的问题对工作产品进行修正。
7. 同时，项目经理将《项目评审表》交给质量保证工程师，由质量保证工程师跟踪问题是否已关闭，签署意见并反馈给项目经理。
8. 最后，项目经理把非正式评审中产生的记录统一递交给项目的配置管理员进行配置管理。
9. 在每次评审完成后，质量保证工程师根据《项目评审表》在《项目度量数据库》中

记录评审的数据，内容包括：评审工作产品名称、工作产品规模、评审次数、评审人员数、评审时间、评审发现的问题。

4.3.5 审核

审核目的，由个人对工作产品进行检查，并确定检查结果。其中审核者直接由项目经理指定，一般为各个小组负责人，比如，测试人员提交的东西，由测试组长负责审核。如果项目规模在 15 以下，我们建议审阅者就是项目经理，如果超过 15 人，可以根据实际情况确定，可以为各小组组长。但是批准者均为项目经理。

工作步骤

1. 作者完成工作产品，提交给审核者。
2. 审核者审阅工作产品，发现问题后以口头或书面反馈。
3. 作者修改问题，并把修改后工作产品提交给审阅者验证。
4. 验证通过后，文档首页和修订页中说明审核人员和批准人员（项目经理）的名字。

【注：在正式评审中，审核人员和批准人员一般不同，而在非正式评审和审批中可以为同一人。】

4.3.6 里程碑评审

里程碑评审对项目阶段的进展状况、度量数据和发生的重大问题进行分析审查，总结前阶段工作、完善改进项目中出现的问题、确定项目发展方向和将来工作安排，以保证项目能够按照预定的计划顺利地实施。并使公司领导宏观掌控项目脉博，在项目开发过程中是最重要的一个管理评审，所以在项目计划里确定的里程碑点必须进行正式评审。评审的具体过程，请参见第 10 章 项目跟踪及控制的 10.3 项目跟踪方式。

第5章 项目初步计划

内容提要：

- CMMI 对应实践
- 项目计划简述
- 项目计划流程
- 项目初步计划活动

项目计划作为项目管理的重要组成部分，其目的是建立和维护项目（开发）计划。其主要原则是先做概要/初步计划，然后再对要执行的活动时细化，形成详细计划。在企业实际操作过程中，不可能一次性把项目的估算及详细计划、详细进度安排确定下来。特别是在需求分析还没有完成之前，在对需求还没有明确的情况下，更不可能把系统设计、编码、测试及项目的一些活动计划确定下来。但我们又不能在没有计划或工作安排的情况下进行需求的收集、需求的分析，所以我们把软件开发过程的项目计划分成了项目初步计划及项目详细计划两部分来执行，以保证其在软件开发过程中的可用性。

5.1 CMM 对应实践

CMMI 中有两个过程域与项目计划相关，分别是项目计划（Project Planning，简称 PP）过程域；集成项目管理（Integrated Project Management，简称 IPM），其中共有 23 个实践与此相对应。

PP 的目的是，建立并维护定义项目活动的计划。在此，我们应当把“计划”理解为动词，应当关注的是怎么策划开发计划的过程，通过该过程，形成了相关计划文档（名词）。在后继的开发过程中，应当对该计划的执行情况及维护过程进行关注。项目计划的依据是：项目工

作说明（《项目任务书》、《立项报告》等）和需求文档（《用户需求列表》、《用户需求说明书》等）。在项目计划过程中，要完成：估算工作产品和任务的属性、确定资源需求、协商承诺、编制进度表，以及识别和分析项目风险。由此形成的项目计划书（含各类专项计划及进度表）为执行项目任务和控制项目活动提供了基础，随着需求和承诺的变更、不准确估计、纠正措施以及过程变更，项目计划也应随之修订。其相关的实践描述如下：

SG 1 Establish Estimates（建立估算），目的是建立和维护项目计划有关的各项参数的估计值。

SP 1.1 Estimate the Scope of the Project（估算项目范围），可以通过编制顶层工作分解结构（WBS），用以估计项目范围。一般可能会形成的工作产品有：任务描述、工作包描述、WBS 图。可以通过如下几步完成该实践，一是基于产品结构开发 WBS；二是以足够小的粒度确定工作包，以便明确描述任务、职责和进度；三是识别应从项目组外部获取的工作产品或工作产品组件；四是识别可复用的工作产品。

【说明：WBS 工作分解结构（Work Breakdown Structure，简称 WBS），是面向项目最终可交付物并且以项目开发活动阶段划分为导向的一种用于项目范围分解的树状层次表示图（可用框图表示，也可用锯齿状纵向列表表示），它应包括项目所涉及的全部工作产品和活动，是项目计划和管理的基础性文档。】

SP 1.2 Establish Estimates of Work Product and Task Attributes（对工作产品及任务属性进行估算），在此过程中通常可能会形成如下工作产品：技术方案（开发策略和整体架构，如分布式或 C/S 架构等），任务和工作产品的规模和复杂度描述，项目估算模型（项目估算表），项目属性估算表。通过如下几步完成该实践：一是确定项目的开发策略和整体架构；二是采用合适的方法，确定将被用于估计资源需求的工作产品和任务的属性（如规模、复杂度等）；三是估计工作产品和任务的属性（值）；四是根据需要，估计项目所需要的人力、设备、材料和方法。

SP 1.3 Define Project Lifecycle（定义项目生命周期），定义项目生命周期及其阶段划分和里程碑设置，以便按阶段分配工作量和安排进度，并在里程碑处评价计划执行情况，必要时重新调整工作量和成本分布。会形成项目生命周期阶段划分文档，做为项目开发计划书的一部分。

SP 1.4 Determine Estimates of Effort and Cost（定义工作量及成本的估算），基于某种采用的估算原理或方法，估计项目工作产品和任务的工作量和成本。一般会形成如下工作产品：估算原理或方法的描述文档，项目工作量估计值，项目成本估计值。可以通过如下几步完成该实践：一是选择估算模型、收集历史数据，以便将工作产品和任务的属性（规模等）转换成工作量和成本；二是估计工作量和成本时，应计入支持性基础设施（如关键计算机资源）的工作量和成本；三是使用模型和/或历史数据，估计工作量和成本。

SG 2 Develop a Project Plan（开发一个项目计划），目的是建立和维护项目计划，并以此作为项目管理的基础。

SP 2.1 Establish the Budget and Schedule（建立和维护进度及预算），一般会形成如下工作产品：项目进度表，项目进度依赖关系文档，项目预算表。可以通过如下几步完成该实践：一是确定主要里程碑；二是确定进度假设；三是识别约束条件；四是识别任务之间的依赖关系；五是定义预算和进度；六是建立应采取纠正性措施的判别准则。

SP 2.2 Identify Project Risks（识别项目风险），一般会形成如下工作产品：已识别的风险列表、风险影响和发生概率、风险严重性排序。可能通过如下几步来完成该实践：一是识别风险；二是将识别出的风险写成文档；三是评审风险文档的完备性和正确性，与项目相关各方达成一致意见；四是根据需要修订风险文档。

【说明：关于风险的详细介绍，CMMI 中专门有一个过程域，本书中也有专门的一章进行讲解，请参见第 9 章 风险管理。】

SP 2.3 Plan for Data Management（对数据管理进行计划），数据可能包含多种内容，比如报告、手册、记录、图表、绘图、规格说明、文件等；也有可能以多种方式存在，比如打印的文档、图片、电子格式、多媒体等。数据有可能会被发布或分发给用户及其他组，也有可能不需要分发。以此过程中，一般会形成如下工作产品：数据管理计划，被管理主要数据列表，数据内容及格式描述，给数据提供者或获取者的需求列表，保密性需求，安全性需求，安全控制规程，数据检索及发布的管理制度，项目数据收集进度表，项目数据收集列表。可以通过如下几步完成该实践：一是建立需求和过程以保证数据的

保密性和安全性；二是建立数据存档机制，以及对存档数据访问的机制；三是确定要识别、收集和发布的项目数据。

SP 2.4 Plan for Project Resources（对项目资源进行计划），对开展项目必须的资源进行计划。一般会形成如下工作产品：WBS 工作包，WBS 任务字典，基于项目规模和范围的人员需求，关键设备和/或工具清单，过程/工作流程定义和框图，事务性管理需求清单。可以通过如下几步完成该实践：一是确定开发过程中的需求；二是确定人员配备需求；三是确定设备、工具和组件需求。

SP 2.5 Plan for Needed Knowledge and Skills（对所需知识及能力进行计划），计划执行项目所需的知识和技能。一般会形成如下工作产品：技能需求清单，人员聘用计划，技能和培训数据库。可以通过如下几步完成该实践：一是确定执行项目所需要的知识和技能；二是评估现有并可用的知识和技能；三是选择提供所需知识和技能的手段或方法（如内部培训、外部培训、聘用新员工等等）；四是将选用的手段/手法纳入项目计划

SP 2.6 Plan Stakeholder Involvement（计划干系人参与），计划已确定的项目干系人应介入的活动。一般会编制项目干系人参与计划。

SP 2.7 Establish the Project Plan（建立项目计划），建立和维护总体项目计划内容，形成总体的项目计划文档，比如：软件项目开发计划书或项目开发计划书。

SG3 Obtain Commitment to the Plan（获得对计划的承诺），建立和维护对项目计划的承诺，为了保持有效性，需要得到那些对该计划执行或支持相关人的承诺。

SP 3.1 Review Plans That Affect the Project（评审相关项目计划），评审影响项目的所有计划，理解项目承诺，形成项目计划的评审记录。

SP 3.2 Reconcile Work and Resource Levels（根据可用资源调整工作计划），调整项目计划，反映估计资源和可用资源的实际情况。一般会形成如下工作产品：被修订的方法和对应的估算参数，经重新协商的预算，已修订的进度表，已修订的需求，经重新协商的项目各相关方的约定。

SP 3.3 Obtain Plan Commitment（获得计划承诺）获取项目相关各方对计划中有关项目实

施和支持过程中的职责所作出的承诺。一般会形成如下工作产品：承诺请求记录文档，承诺记录文档。可以通过如下几步完成该实践：一是识别必需的支持，与项目干系人协商约定；二是记录机构一级的承诺，包括正式的和临时的承诺，确保有必要的签字；三是需要时，高级管理部门审查内部承诺；四是必要时，高级管理部门审查外部承诺；五是确定项目组内部各个部分之间、项目组与其他项目组之间、项目组与机构内部相关部门之间对相互接口的承诺，以便于监督。

IPM 的目的是，依据一个集成的、已定义的过程（此过程是从机构标准过程集中裁剪而得到），建立并管理项目及相关干系人的参与。其主要包含的内容有：在项目开始时，通过对机构标准过程集进行裁剪来建立本项目的定义过程（PDP）；用项目定义过程来管理项目；根据机构工作环境标准建立本项目的工作环境；使用并丰富机构过程资产；在产品开发过程中，使相关干系人所关心的事均被识别、考虑并适当的处理；确保相关干系人以协同的适时的方式执行他们的工作，一是侧重于产品和产品组件的需求、计划、目标、问题和风险，二是履行他们的承诺，三是识别、跟踪、解决相关问题。其相关的实践描述如下：

SG1 Use the Project's Defined Process（使用项目已定义过程），项目必须依照从机构标准过程集中裁剪得到的项目定义过程（**PDP**）来执行。

SP1.1 Establish the Project's Defined Process（建立项目定义过程），在项目初期建立项目定义过程，并在整个生命周期中维护该过程。一般是根据如下因素来确定 PDP，客户需求、产品及产品组件需求、承诺、机构过程需要及目标、机构标准过程集及裁剪指南、操作环境、商务环境等。最终产生一个项目定义过程文档。可以通过如下几步完成该实践：一是从可使用的机构过程资产中选择本项目的生命周期；二是从机构标准过程集中选择最适合本项目需要的标准过程；三是根据裁剪指南，对标准过程及其他机构过程资产进行裁剪，得到本项目的定义过程；四是适当使用机构过程资产中的其他已有历史资料，比如：培训资料、模板、示例文档、估算模型等；五是文档化项目已定义过程；六是对项目已定义过程进行同行评审；七是根据需要修订项目定义过程。

SP1.2 Use Organizational Process Assets for Planning Project Activities（使用机构过程资产

计划项目活动），使用机构过程资产和度量数据库来估算和计划项目活动。一般会形成，项目估算记录和项目计划书两个文档。可以通过如下几步完成该实践：一是使用项目定义过程的任务和工作产品做为项目估算和计划项目活动的基础；二是在估算项目参数时，使用机构度量数据库，比如类似项目的历史数据等。

SP1.3 Establish the Project's Work Environment（建立项目的工作环境），依据机构标准工作环境来建立和维护项目工作环境。一般会形成如下工作产品：项目的设备及工具清单，项目工作环境的安装、操作和维护手册，用户调查及结果，使用、执行和维护记录，项目工作环境支持服务。可以通过如下几步完成该实践：一是计划、设计和安装项目工作环境；二是对项目工作环境提供持续维护和操作支持；三是维护项目工作环境组件的工作能力；四是定期审查工作环境满足项目需要的能力及相互之间支持的协调性，如果发现问题，采取适当的措施。

SP1.4 Integrate Plans（集成计划），集成项目计划和其他影响项目定义过程描述的计划。通过该实践形成了集成后的项目计划，可能是一份文档，也有可能是多份文档。可以通过如下几点来完成该实践：一是把其他影响该项目的计划与项目计划集成，主要为，质量保证计划、配置管理计划、风险管理计划、文档计划等；二是把对项目度量指标的定义及度量活动集成到项目计划中；三是识别和分析产品及项目接口风险；四是按顺序确定重大开发因素及项目风险的进度安排；五是合并项目定义过程中执行同行评审的计划；六是在项目培训计划里合并执行项目定义过程的培训需要；七是对于批准 WBS 中任务描述的开始及终止，建立客观的准入、准出标准；八是保证项目计划与干系人计划保持适当的一致性；九是确定如何解决相关干系人之间的冲突。

SP1.5 Manage the Project Using the Integrated Plans（使用集成过的计划管理项目），使用项目计划及其他影响项目和项目定义过程的计划来管理项目。一般会产生如下工作产品：在执行项目定义过程时产生的工作产品，收集的实际度量数据、进度记录和报告，修订的需求、计划和承诺。可以通过如下几步完成该实践：一是应用机构过程资产库来实现项目定义过程，比如使用机构过程资产库里学得的经验来管理项目；二是使用项目定义

过程、项目计划及其他影响项目的计划来跟踪和控制项目活动及工作产品；三是获取和分析选择的度量指标来管理项目和支持机构需要；四是定期审查，并将项目进度与机构、客户及最终使用者当前和期望的需要、目标及需求保持一致。

SP1.6 Contribute to the Organizational Process Assets（为机构过程资产贡献），把工作产品、度量和文档化的经验贡献为机构过程资产。此实践是需要在项目开发过程中进行数据收集，然后在项目总结时，把相关数据存放到机构过程资产库，所以是与第 17 章 项目总结直接对应。一般会形成如下工作产品：机构过程资产的改进建议，从项目中收集的实际过程和产品度量数据，过程描述、计划、培训模式、检查列表和经验等之类的文档，项目中与裁剪和实现机构标准过程集相关的过程资料。

SG2 Coordinate and Collaborate with Relevant Stakeholders（与相关干系人协调和合作）。

SP2.1 Manage Stakeholder Involvement（根据项目集成和定义过程来管理干系人的参与）。一般会产生如下工作产品：协作活动的进度安排及议程，文档化的问题（比如用户需求、产品及产品组件需求、产品架构、产品设计等的问题），解决相关干系人问题的建议。

SP2.2 Manage Dependencies（管理相互依赖关系），与相关的干系人共同识别、协商与追踪重要的依存关系。

SP2.3 Resolve Coordination Issues（解决协调问题），与相关的干系人协调解决问题。通过如下几步完成该实践：一是识别和文档化问题；二是与相关干系人交流沟通问题；三是与相关干系人解决问题；四是对于与相关干系人不能解决的问题，提交给适当级别的管理者；五是跟踪问题直至关闭；六是就问题的状态及解决方案与相关干系人沟通。

5.2 项目计划简述

项目计划的目的是为项目的实施制定一套合理、可行的项目（开发）执行计划。项目计划活动的主要内容包括：分解项目需求，标识项目全部工作产品和活动，编制 WBS；估算工作产品和活动的规模、工作量、成本和所需资源；识别并制定项目资料管理计划；制定工作进度表；识别和分析项目风险，编制风险管理计划；协商相关约定；编写项目开发计划，经

评审、批准并以此作为项目跟踪监督的依据。

公司里执行项目计划活动时，一般会要求遵循一定的指导原则，在此我们给出如下建议供读者参考。项目经理负责，裁剪《机构标准软件过程（OSSP)》得到项目定义过程（PDP)，在此基础上组织项目策划、编制项目开发计划。 项目组在项目计划活动过程中，应该遵循以下几个原则：

1. 以经过评审确认后的《用户需求说明书》和《软件需求规格说明书》中的系统需求作为制定软件项目开发计划的基础。
2. 与其他相关组协商应由他们介入本项目组活动的计划，商定的介入活动纳入本项目计划，并有文字协议或记录。
3. 与部门外部（用户）以及其他相关组的项目约定，应经研发部经理或总工程师审批。
4. 按相关规程规定，估算项目软件的规模、工作量和成本，估算规模、工作量和成本时采用的假设条件和估算结果应经过评审和确认，以便作为机构过程资产最终存入过程数据库。
5. 估计产品运行所必需的关键计算机资源，估计项目开发所必需的设备和工具，形成文档，纳入项目开发计划。
6. 识别、评估软件风险，制定以首要风险应对措施为主要内容的管理计划，纳入项目开发计划。
7. 编制项目软件配置管理计划和质量保证计划，纳入项目整体开发计划书（可能是一份文档也有可能是多份文档)。
8. 项目开发计划书（含各类专项计划）经过评审、确认和批准后纳入基线，用于项目跟踪监督，随后发生的项目开发计划变更应得到控制和管理。

作为项目计划活动的第一个阶段，项目初步计划的目的是：根据项目任务书或项目合同为下一阶段可能进行的工作进行大体规划，定义项目开发计划初步的内容，以方便指导项目开发计划定稿之前的工作（主要是技术方案验证或方案预研，需求获取/收集及需求分析等相关的工作)。一般是在拿到《项目任务书》之后，以项目经理为主，配置管理员、质量保证工

程师及项目其他组员配合。项目初步计划主要分为如下几项内容：

1. 根据初步需求，确定项目的目标和工作范围；
2. 组建项目团队，选择适当人员执行项目任务并明确其工作职责；
3. 定义项目的软件过程和生命周期；
4. 识别项目工作产品；
5. 进行初步 WBS 分解；
6. 编制项目开发计划书初稿。

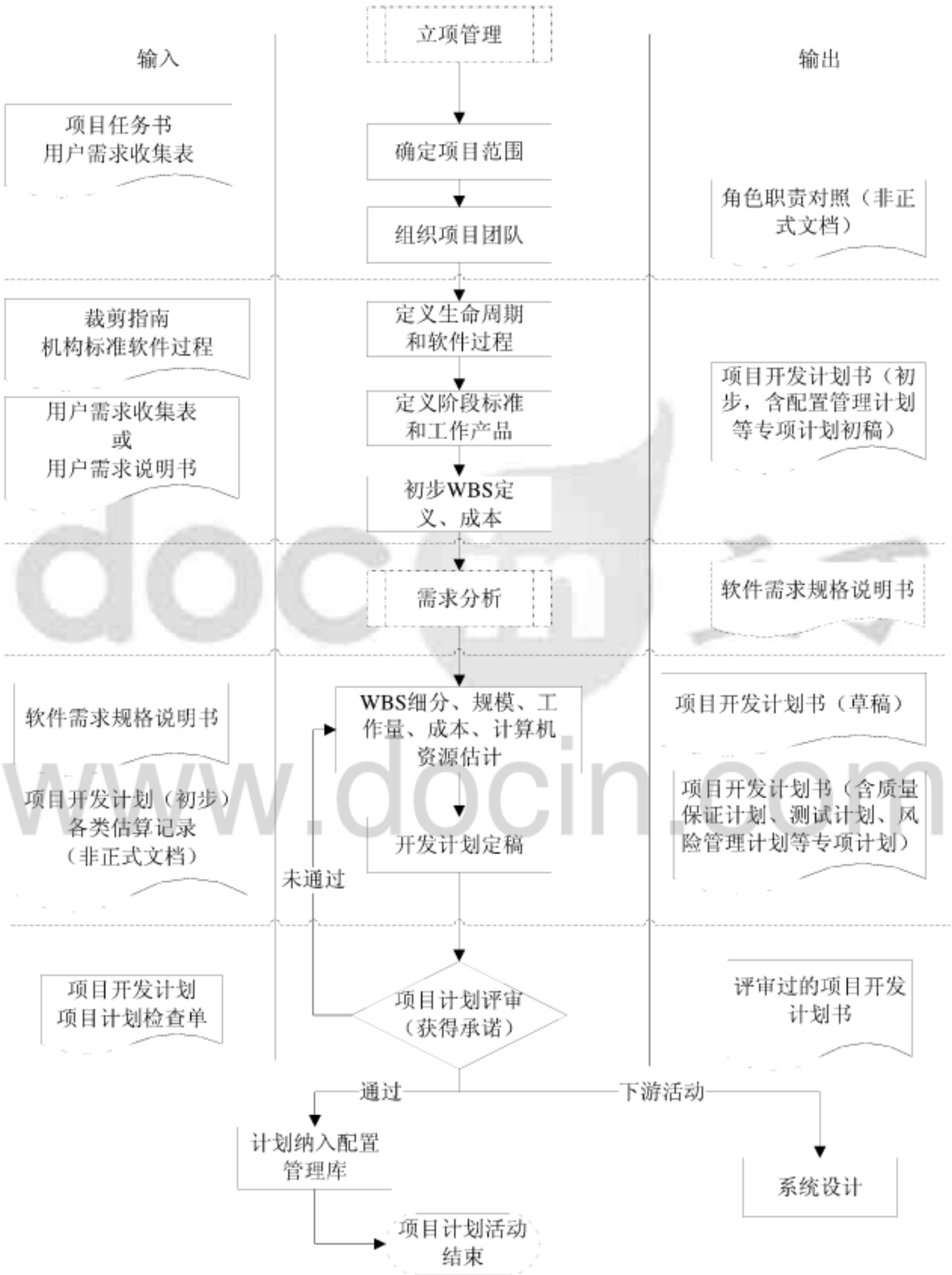
5.3 项目计划流程

就整个项目计划活动而言，通过此过程之后，应当达到如下目标：确定项目生命周期阶段划分以及里程碑设置，标识全部工作产品和活动；确定项目资料清单及其管理计划；估算工作产品和活动的规模、工作量、成本以及各类资源；编制项目配置管理计划（详细的讲解参见第 8 章 软件配置管理）；识别、评估软件风险，制定风险管理计划（详细的讲解参见第 9 章 风险管理）；编制项目进度表；编制过程与产品质量保证计划（详细的讲解参见第 18 章 产品及过程质量保证）；协商并确定组内和/或组间以及与外部（用户）的约定，取得承诺；制定项目计划，并经评审、确认、批准，纳入基线，得到管理和控制。整个项目计划过程可以按照图 5-1（见下页）所示的流程来执行。

在此我们需要讲解一下项目计划活动的最初依据，也就是根据什么来制定项目计划？一般有两种，一是机构内部的项目/产品立项审批文件或项目工作说明[7]（Statement Of Work ，简称 SOW）；二是经过评审、确认或审批的需求文档，包括项目（软件）需求和/或需求规格说明书。项目计划活动开始时，应根据项目工作说明和需求文档，确定项目范围，定义最终产品。

[7] SOW 是一份关于项目范围、目标以及项目成本、进度和资源等约束的文档，它不是设计文档或需求文档，也不是一份完整的法律文书，其作用是用以确定项目工作范围和定义最终产品，其内容一般包括下列要点：项目工作范围、技术目标、应遵循的标准和规范、成本、进度目标及其约束、项目组和其他机构之间的关系、资源（包括人员组成）限制和目标、以及对软件开发和维护的约束和目标。通常，项目任务书、立项报告等就可以作为 SOW（除非项目合同甲方正式提供 SOW）。

图表 5-1 项目计划活动流程图



5.4 项目初步计划活动

项目计划是一个过程，在项目刚立项通过之后，由于需求获取及分析的不够充分，很难把项目计划一次性编制完成。但是一些必须的项目活动及工作需要开展，这就是项目初步计划要解决的问题。在进行项目初步计划时，可以按如下顺序来执行计划活动。

1. 确定项目范围；
2. 组建项目团队；
3. 定义软件过程和生命周期；
4. 制定 WBS 初稿；
5. 识别项目工作产品；
6. 编制其他专项计划初稿；
7. 形成项目开发计划书初稿；

下面详细描述项目初步计划中每个活动的步骤：

第一步，确定项目范围

1. 项目经理根据立项相关文档或项目合同负责确定本项目的目标和工作范围。
2. 在开始制定软件开发计划之前，必须首先知道客户/或产品的核心需求。如果核心需求还没有定义，则需要在该阶段确定需求获取的计划，以对客户需求进行确定、分析和文档化，然后再次明细项目计划内容。

第二步，组建项目团队

1. 项目经理根据项目任务书来组建项目开发团队，也可以根据项目实际情况向研发部经理/公司总工程师提出人员配备申请。
2. 由研发部经理/总工程师和项目经理与适当的组/人进行接触，并与他们协商参该项目的相关事宜，落实究竟让谁参与该项目。在进行项目估计时，还要对所需的其它资源和支持继续进行沟通和协商。
3. 项目团队组建完成后，需要明确地识别项目所需角色，说明每个项目组成员的工作

职责，写入项目开发计划。

第三步，定义生命周期和软件过程

1. 按照《机构标准软件过程》和相关剪裁指南，根据项目特征选择项目类型，对项目特征进行量化选择项目软件过程元素，定义项目软件过程元素的活动，形成本项目的软件过程。
2. 由负责协调软件过程活动的组或个人，如：项目质量保证工程师，QA 经理等(具体人员在项目任务书中确定)评审项目软件过程的剪裁是否合理和适用，并由研发部经理批准。
3. 对于特殊过程的项目，参照《机构标准软件过程》进行更改，并经 EPG 批准后，该过程作为过程财富纳入过程文档库进行管理。若全部遵照机构标准软件过程中的定义，则不需要执行该步骤。
4. 根据《机构标准软件过程》和相关剪裁指南，为软件项目选用项目的唯一、切合实际的软件开发生命周期。如果项目有需要，可根据剪裁原则对标准生命周期模型进行修改，并建立文档。
5. 由负责协调软件过程活动的组或个人，如：项目质量保证工程师，QA 经理等(具体人员在项目任务书中确定)评审项目生命周期模型的剪裁是否合理和适用，并由 EPG 批准。
6. 对于特殊项目，如果其生命周期模型和标准周期模型有很大出入，必须由 EPG 批准；如果合适该生命周期模型将作为过程财富纳入过程数据库进行管理。若全部机构标准软件过程中的定义，则不需要执行该步骤。

【说明：关于里程碑的一些建议及要求：

1. 在项目计划初步阶段，需要确定里程碑划分，同时各个里程碑举行的活动也需要明确。
2. 对于跨度小于两个月或工作量小于 3 人月的项目，可以不划分里程碑。
3. 里程碑划分时，需要为项目的进度留出一定的 Buffer（即缓冲时间），以便进行进度调节。
4. 第二、三类学员实训项目必须划分里程碑，第一类学员实训项目是否划分里程碑授课教师根据学

生的实际开发熟悉程度来划分。】

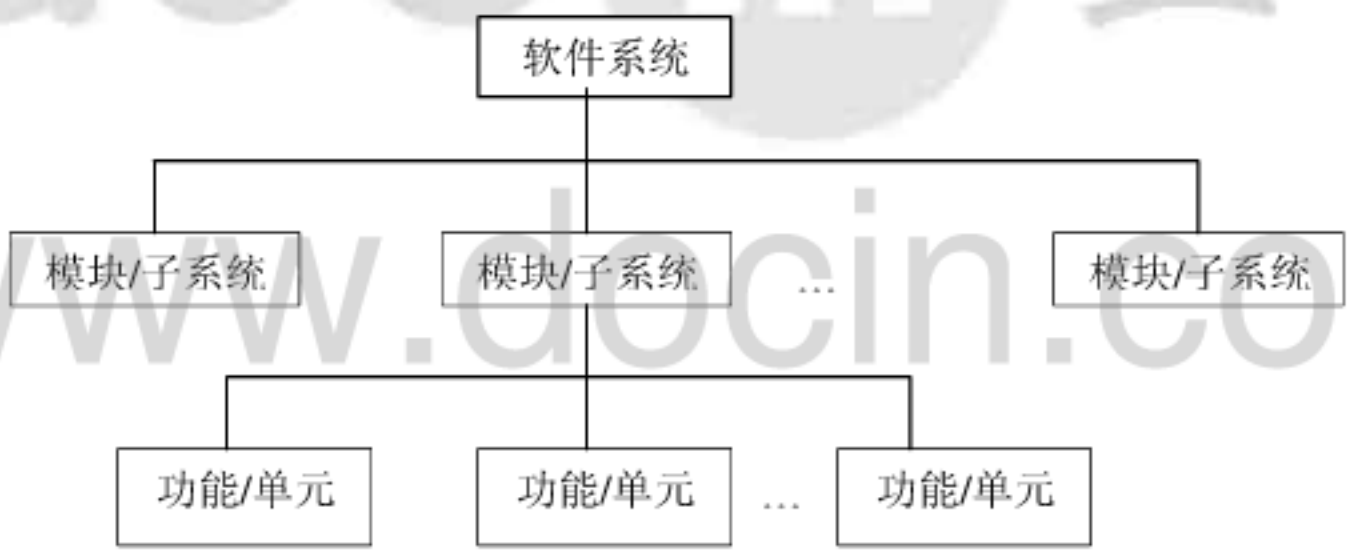
第五步，制订 **WBS** 初稿

制订初步 WBS 分解，在项目的早期阶段开始制定 WBS，并随着工作的展开而逐步细化，定义出易于管理的 WBS 的最底层元素，形成项目开发计划初稿。建议采用项目管理工具（如 MS Project）形成 WBS 工作分解图。

确定 WBS 的分级，一般 4-5 级，在项目的早期阶段一般只需要制订到第 1、2 级的 WBS。在后续的项目各阶段需要逐步细化 WBS，如设计阶段、编码阶段等。 在开发过程中，如何创建一个项目的 WBS，可以有许多方法。下面介绍两种常用的方法：

- 名词型方法，面向产品（最终交付物）结构，按子系统、功能模块进行划分，层层分解，分解到由若干个相对独立的单元为止。当估算产品本身的规模和工作量时，常用此法编制 WBS。如图 5-2 所示。

图表 5-2 面向产品结构的 WBS 示例



- 动词型方法，面向过程活动，按完成最终交付物而必须执行的活动进行分解，层层分解，分解到每一个（由若干个任务组成的）活动均可以作为相对独立的工作包进行定义时为止。当估算整个项目工作量或编制进度表时，常用此法编制 WBS。如图 5-3（见下页）所示。

图表 5-3 面向项目活动和任务的 WBS 示例

不论采用何种方法，分解到最后的模块单元或基本活动（统称工作包），均应遵循如下约定：

- 工作产品和活动分解的细度以可管理、可验证、可分配并相对独立为原则。
- 一个单元工作或一项活动在 WBS 中只能出现一次。
- 一个单元项的工作内容是下一层各个单元项工作的总和。
- 图中连线或表中嵌套深度只表示工作或活动间的内在联系，不表示先后顺序　关系。

- 单元项工作内容尽量采用动词，例如，“编写 SRS”，而不是“SRS 的编写”。
- 如果有分承包方或介入其他相关组的活动，也应在 WBS 中得到体现，但相关的技术细节则应包括在分承包方的项目开发计划中。

一个好的 WBS，每一个单元或工作包必须满足以下 6 个条件：

- 状态是可以计量的；
- 就绪、结束条件可以明确定义；
- 有应交付的成果；
- 便于规模和工作量的估计；
- 完成单元任务的工期不宜太长（如，不超过一个星期）；
- 单元任务的安排可以相对独立。

第五步，识别工作产品

1. 按照定义的项目开发生命周期和技术方法识别软件工作产品，确保过程和产品对应关系清晰、完整；软件工作产品包括：本项目以及相关组产生的工作产品。工作产品的范围取决于项目，取决于由机构、机构的管理人员和客户之间达成的协议。
2. 确定各工作产品的作者，提交时间和验收准则。
3. 确定生命周期各阶段的入口标准、工作任务、出口标准，根据立项报告、项目合同等文档确定各个阶段准确的时间范围。

第六步，编制专项计划

1. 配置管理计划（初稿）——项目经理负责或指定配置管理员编写；
2. 质量保证计划（初稿）——质量保证工程师根据项目进展编写；
3. 风险管理计划（初稿）——项目经理按风险管理相关规定编写；
4. 度量分析计划（初稿）——项目经理按度量分析相关规定编写。

以上三个专项计划在本书中均会有专门的章节讲解，除此之外，常见的专项计划还可能会包含：《干系人计划》、《数据管理计划》（对于数据保密性、安全性要求高的项目，或者项目规模超过 100 人月的项目，必须编写该计划，明确项目中产生所有数据的管理计划及管理

活动）等。

第七步，编制开发计划初稿

在立项报告通过或者项目合同签订后一周内（具体时间要求根据公司及项目规模的不同而不同，由各自公司 EPG 在项目计划规程里规定），项目经理负责，根据项目开发计划流程定义各项项目开发计划的内容，并按照《项目开发计划》模板，编制项目开发计划初稿，同时把其他专项计划集成到整个项目大的计划中。在项目早期开发计划初稿阶段，WBS 的内容包括第 1、第 2 级，计划的具体细化过程在需求规格说明书评审通过后进一步完成。

第6章 需求开发及管理

内容提要：

- **CMMI** 对应实践
- 需求开发及管理简述
- 需求开发及管理流程
- 需求获取
- 需求分析
- 需求评审
- 需求管理

在 IEEE 软件工程标准词汇表(1997 年)中定义软件需求为：

1. 用户解决问题或达到目标所需的条件或能力。
2. 系统或系统部件要满足合同、标准、规范或其它正式规定文档所需具有的条件或能力。
3. 一种反映上面(1)或(2)所描述的条件或权能的文档说明。

通俗的讲，“需求”就是用户的需要，它包括用户要解决的问题、达到的目标、以及实现这些目标所需要的条件，它是一个程序或系统开发工作的说明，表现形式一般为文档形式。

从国内软件企业的实际情况看，交给项目组作为应完成任务的最初描述的需求（给定需求），其来源大体有三个：

- 直接来自机构外部（即合同甲方），此时，给定需求就是以委托方式或合作方式提交的任务书；或者是通过项目投标最终以合同形式确定的用户需求，其载体就是投标

书、合同及其技术附件以及合同签订后编写的软件需求文档，当然还应包括项目开始后几乎难以避免的变更需求。也可以间接的来自外部机构，此时，公司内部不同部门或公司外部的委托方或合作方承接的整个集成项目分配给项目组的软件开发任务就是给定需求。

- 来自部门内部的新产品开发，此时给定需求及其载体就是新产品可行性研究报告、软件需求文档（初稿）以及相应的软件立项审批表。
- 来自已发布产品或已提交项目的最终用户，此时，不同的最终用户或同一个最终用户在不同时间、不可预料地提出的产品的每一个缺陷报告或个性化修改要求都是给定需求，其载体可能是用户的传真或电子邮件，也可能是维护部门的电话记录，还可能是市场人员、工程人员的间接反映。

需求开发及管理（Requirement Development and Management，RDM）的目的是在获得正确的用户需求基础上，经过分析和定义，最终生成项目的《用户需求说明书》和《软件需求规格说明书》。同时借助需求管理寻求客户与开发方之间对需求的共同理解，控制需求的变更，维护需求与后续工作产品之间的一致性。

6.1 CMMI 对应实践

在 CMMI 中，总共有两个过程域与本章讲解的内容直接相关，分别为：需求管理（Requirements Management，简称 REQM），需求开发（Requirements Development，简称 RD）。在这两个过程域中共有 15 个特定实践，由此可见 CMMI 模型对系统需求、软件需求是非常重视的，由于对需求要进行评审及跟踪，所以 VAL 与 VER 两个过程域也与本章相关，针对这两个过程域的描述放到测试相关章节里。

REQM 的目的是管理项目产品和产品组件的需求，并识别需求与项目计划和工作产品之间的不一致项。需求包括：技术性需求、非技术性需求、功能需求、非功能需求，来自客户的需求、来自项目组内部的需求等。项目组应采取适当的步骤，确保协商一致的需求得到管理，以支持项目计划活动的执行。当项目组从被认可的需求提供者接受需求时，在将需求纳

入项目计划之前，应与需求提供者一起对需求进行评审，以解决争议问题或者防止误解。需求提供者与接受者之间一旦取得一致，项目组就应从项目参加者那里取得对于需求的承诺。项目组应该管理需求变更（当出现变更时），并且识别在项目计划、工作产品和需求之间发生的不一致项。作为需求管理的一个部分，需求变更及其理由应填入记录文档，并且在最初需求与产品和产品组件的所有需求之间维持双向可追溯性。其相关的实践描述如下：

SG 1 Manage Requirements（管理需求），对需求进行管理，并识别需求与项目计划和工作产品之间的不一致项。

在整个项目开发过程中，项目组应当通过下列活动维护经过确认的最新需求：管理所有的需求变更；维护需求与项目计划和工作产品之间的对应关系；识别需求与项目计划和工作产品之间的不一致项；采取纠正措施。

SP 1.1 Obtain an Understanding of Requirements（获得对需求的理解），随着项目的进展和需求的衍生，各项活动或学科（需求、设计、开发、测试等）会不断地收到新的需求。为了避免需求漂移（creep）(需求不断的增加或变化)，应建立准则，用以指明接受需求的正规渠道或需求的合法来源。接受需求的活动应该包括与需求提供者一起进行的需求分析活动，以确保对需求的含义取得一致的理解。分析和沟通的结果是形成取得一致理解的需求集合。一般会产生如下工作产品：合格需求提供者判别准则，需求验收和鉴定标准，根据准则进行分析后所得的各类结果，达成一致理解的需求集。可以通过如下几步完成该实践：一是制定合格需求提供者判别准则；二是制定需求验收的客观标准，缺少验收标准通常将导致不充分的验证、高代价的返工以至用户拒收；三是分析需求，确保满足所制定的准则；四是与需求提供者达成对需求的共同理解，以便项目参与者能够对需求做出承诺。

SP 1.2 Obtain Commitment to Requirements（获得对需求的承诺），从项目参与者处取得对需求的承诺。一般会产生如下工作产品：需求影响评估结果，对需求及其变更承诺的记录文档。可以通过如下几步活动完成该实践：一是评估各项需求（包括或特别是变更需求）对已有承诺的影响，当需求发生变更或提出新需求时，应评价它们对项目参与者的

影响；二是协商并记录承诺，在项目参与者对需求或需求变更作出承诺之前，应协商已有承诺的变更。

SP 1.3 Manage Requirements Changes（管理需求变更），在项目开发期间，需求发生变更时，应对需求变更进行管理。在项目开发期间，需求会由于各种原因而发生变化。当原来的需要发生变化并且工作还需要继续进行时，产生了附加的需求，因此必需对现有的需求做出相应的变更。此时，最重要的事情就是有效的和高效的管理这些新增需求和变更。为了有效的分析变更的影响，应该知道每项需求的最初形态，并记录其变更原因。项目经理还要跟踪需求发散性的合理度量，以便判断是否需要采取新的控制措施或修改已有的控制措施。一般会形成如下工作产品：需求状态表，需求数据库，需求决策数据库等。可以通过如下几个活动完成该实践：一是汇总交给项目组的或者项目组产生的全部需求及其变更；二是维护需求变更的历史及变更理由，维护变更的历史数据有助于跟踪需求的发散性；三是从相关的项目干系人的角度出发评价需求变更的影响；四是使需求和需求变更数据可供项目组使用。

SP 1.4 Maintain Bidirectional Traceability of Requirements（维护对需求的双向跟踪），维护需求与项目计划和工作产品之间的双向可追溯性。其意图在于维护每个产品分解层次的需求的双向可追溯性。如果需求得到很好的管理，就可以建立起从原始需求到它的较低层次的需求的可跟踪性，以及从较低层次的需求回到较高层次的最初需求的可回溯性。这种双向可追溯性有助于确定是否所有原始需求都完全得到处理，是否所有的低层需求都可以回溯到有效的来源。需求的可追溯性还可以用于维持需求与其他事项的关系，例如，中间产品或最终工作产品、设计文档的变更、测试计划、以及工作任务等。可追溯性应该包括横向可追溯性和纵向可追溯性，例如，横跨接口两边的需求。在评估需求变更对项目计划、活动以及工作产品的影响时，尤其需要可追溯性。一般会形成如下工作产品：需求追溯性矩阵，需求跟踪系统。可以通过如下几步来完成该实践：一是维护需求的可追溯性，以确保将低层（派生）需求的源需求记入文档；二是维护每个需求与它的各个派生需求的可追溯性，维护每个需求与功能分配、目标、人员和过程的可追溯性；

三是维护功能（模块）之间和跨接口二边的功能之间的横向可溯性；四是生成需求可追溯矩阵。

SP 1.5 Identify Inconsistencies Between Project Work and Requirements（识别项目工作与需求之间的不一致项），识别项目计划和工作产品与需求之间的不一致项，目的是发现需求与项目计划和工作产品之间的不一致项，并且启动纠正措施。一般会形成如下工作产品：以文档形式记录不一致项（包括来源、条件和理由），纠正措施。可以通过如下几步来完成该实践：一是评审项目计划、活动和工作产品与需求及其变更的一致性；二是识别不一致项的来源和理由；三是识别因需求基线的变更而导致的项目计划、活动和工作产品的变更；四是启动纠正措施。

RD 的目的是产生并分析用户、产品及产品组件需求。需求作为设计的基础，在开发过程中，需要收集软件需求、客户或其他利益同担者（stakeholder，或译成干系人）的需要，对所有可能的需求及需要进行分析，并对分析的结果进行评审，最终针对需求达成一致。本过程中描述了三类需求，分别是客户需求、产品需求、产品组件需求，整体来说，这些需求侧重于相关干系人（含客户）的需要，包括那些与各个产品生命周期（比如，接受测试准则）及产品属性（比如，安全性、可靠性、可维护性）有关的需要。需求还侧重于由于解决方案设计的选择而导致的约束（比如，与第三方外购商业软件集成）。

SG 1 Develop Customer Requirements（开发客户需求），收集干系人（包括客户、最终用户、供应商、构造人员、测试人员、设备制造商、后勤支持人员等）的需要、期望、约束、接口，并把他们转化为客户需求。

SP 1.1 Elicit Needs（引出客户需要），引出在产品生命周期内所有阶段干系人的需要、期望、约束和接口。使用的方法及技术通常有：技术演示，临时项目审查，问卷调查、访谈和最终用户的操作场景，操作演练和最终用户的工作分析，原型和模型，头脑风暴，市场调查，Beta 测试，从文件、标准及规范中提取，对现在产品、环境及工作流程模式的调查，使用案例（用例），商业案例分析，对遗留产品反工程，客户满意度调查等。但是有一些需求来源是客户没有办法提供的，主要有：商业政策、标准、业务环境需求（比

如实验室、测试和其他设施、信息技术和基础设施）、技术、遗留产品及产品组件（可重用的产品组件）等。

SP 1.2 Develop the Customer Requirements（开发客户需求），把干系人的需要、期望、约束和接口转化为客户需求。一般会产生如下工作产品：客户/用户需求说明书，验证时客户约束，确认时客户约束。可以通过如下两步完成该实践：一是把干系人的需要、期望、约束和接口转化为文档化的客户需求；二是定义验证和确认时的约束。

SG 2 Develop Product Requirements（开发产品需求），对客户需求细化及描述以开发为产品及产品组件需求。

SP 2.1 Establish Product and Product Component Requirements（建立产品和产品组件需求），基于客户需求来建立和维护产品及产品组件需求。客户需求可能是使用客户的术语来表达，而不是技术性的描述。而产品需求要使用技术术语来表达这些需求，以方便设计决策时使用。一般会形成如下工作产品：衍生需求，产品需求，产品组件需求。可以通过如下几步完成该实践：一是为满足产品及产品组件设计使用技术术语来开发需求；二是根据设计决策的结果派生出需求；三是建立和维护需求变更管理与需求分配之间的关系。

SP 2.2 Allocate Product Component Requirements（分配产品组件需求），为每个产品组件分配需求。已定义解决方案的产品组件需求包括产品性能分配，设计约束，以及需求的满足功能等。一般会形成如下工作产品：需求分配表，临时需求分析，设计约束，派生需求，派生需求之间的关系。可以通过如下几步完成该实践：一是给功能分配需求；二是给产品组件分配需求；三是给产品组件分配设计约束；四是文档化已分配需求之间的关系。

SP 2.3 Identify Interface Requirements（识别接口需求），识别功能（或对象）之间的接口，功能（或对象）之间的接口可能会决定在 TS 过程域中选择不同的解决方案。在产品架构定义时，需要识别产品或产品组件之间的接口。可以通过如下两步完成该实践：一是识别产品外部及产品内部接口（比如，在功能划分或对象之间）；二是开发已识别接口的需求。

SG 3 Analyze and Validate Requirements（分析并确认需求），针对用户预期的环境下分析和验证需求。

SP 3.1 Establish Operational Concepts and Scenarios（建立并维护系统操作概念及场景），一般会产生如下工作产品：操作概念描述，产品或产品组件安装、操作、维护和支持概念性描述，部署概念，用例，时间表场景，新需求。可以通过如下几步完成该实践：一是适当的开发包括功能、性能、维护、支持和部署在内的操作概念及场景；二是定义产品或产品组件将要操作的环境，包括边界和约束；三是对操作概念及场景审查以精练和发现需求；四是当选择产品或产品组件时，开发详细的操作概念，定义产品、最终用户和环境之间的交互，以满足操作、维护、支持和部署的需要。

SP 3.2 Establish a Definition of Required Functionality（建立并维护功能需求的定义）。一般会产生如下工作产品：功能架构图，活动图和用例，使用服务或方法标识的面向对象分析。可以通过如下几步完成该实践：一是分析及量化最终用户的功能要求；二是分析需求以识别逻辑或功能划分（比如，子功能）；三是基于已确定的标准（比如，相似功能、性能或藕合性）对需求分组，使需求分析更容易、更便于聚焦；四是在产品组件开发时，要考虑到时间要求敏感的功能；五是给划分的功能、对象、人员或支持元素分配客户需求，以支持解决方案的整合；六是给功能或子功能分配功能和性能需求。

SP 3.3 Analyze Requirements（分析需求），分析需求以保证他们的必要性和充分性。一般会产生如下工作产品：需求缺陷报告，为解决缺陷提议的需求变更，关键需求，技术性能度量。可以通过如下几步完成该实践：一是分析干系人的需要、期望、约束和外部接口，以消除冲突并且把他们根据相关主题组合在一起；二是分析需求以决定他们是否满足更高级别需求的目标（比如，商业目标等都可以称为更高级别需求）；三是分析需求以保证他们的完成性、可行性、可实现性和可验证性；四是识别与成本、进度、功能、风险或性能有重大影响的关键需求；五是识别需要在开发过程中跟踪的技术性能度量指标；六是分析操作概念及场景以精炼客户需要、约束和接口，并发现新需求。

SP 3.4 Analyze Requirements to Achieve Balance（分析需求以达到平衡），分析需求在干系

人需要和约束之间进行平衡。一般会形成需求相关的风险评估报告，可以通过如下几步完成该实践：一是使用证明模型、模拟和原型来分析平衡利益相关者的需要和约束；二是对需求和功能架构进行风险评估；三是研究产品生命周期概念对需求影响的风险。

SP 3.5 Validate Requirements（确认需求），确认需求以保证终产品将会在用户环境中按照预期运行。可以通过如下几步完成该实践：一是分析需求找到最终产品不能在用户环境按照预期适当的运行的风险；二是通过开发产品表示（比如原型、模拟、建模、场景、情节串连图板）及从相关干系人获得反馈来探索需求的充分性和完整性；三是当设计趋于完成时，在需求确认环境的上下文中对其进行评估，以识别确认问题及发现未阐明的需要及客户需求。

6.2 需求开发及管理简述

需求开发及管理的目的是，在用户和将处理用户需求的软件项目之间建立对用户需求的共同理解，并且在开发过程中保持需求的一致性、有效性、稳定性。

需求开发及管理主要包括的内容为：将分配给软件的系统需求文档化；进行软件需求分析及评审并建立软件需求基线；管理和控制需求基线及需求变更，保持软件需求和项目计划、工作产品及过程活动的一致性。

在公司里，对需求开发及管理会制定一定的准则，所有项目均需要遵守该准则。通常的准则有如下几点：

- 在需求获取时，通过整理获以的需求，从而明确需求来源、确认需求类型（功能需求或非功能需求）、明确需求的优先级，形成《用户需求说明书》，并得到用户的确认。
- 在每个软件项目的需求分析阶段，对给定的《用户需求说明书》进行分析以文档化的形式编制《软件需求规格说明书》（Software Requirement Specification，简称 SRS）。
- SRS 中每一项需求的描述，都必须确保是正确的、最新的、完备的、必要的、可验证的、可追踪的和可测试的。

- SRS 必须通过相关组的评审，相关组包括：开发组、测试组、质量保证组、配置管理组、文档组及个人。只有相关组审批、确认后的 SRS，才能作为项目开发计划以及后续项目工程和管理活动的基础。
- 经过评审确认的 SRS 任何变更，都应得到控制和管理，一旦需求发生变更，项目计划、过程活动、工作产品等要随之变更与需求变更保持一致，并重新提交相关组和个人复审。

6.3 需求开发及管理流程

需求开发及管理活动主要分为四个阶段，分别是：

1. 准备阶段——在项目初步计划书里明确需求收集及分析的进度安排及人员安排。
2. 需求收集阶段——立项阶段用户需求收集不充分或有不明确之处，继续进行用户需求收集，并转化为产品需求。
3. 需求分析阶段——对用户需求列表或/和用户需求说明书中的需求进行分析，给出详细的软件需求规格说明书。
4. 需求管理——评审通过的软件需求规格说明书，纳入基线，严格执行需求变更管理，对需求跟踪矩阵进行管理，要保证需求的双向跟踪。

整体工作流程如图 6-1（见下页）所示，注意需求获取活动一般在立项之前就开始，在立项之后还需要进行部分获取工作。

6.4 需求获取

在进行软件需求获取之前，有必要了解一下系统工程与需求工程之间的区别及关联，具体来说有以下几个方面[8]：

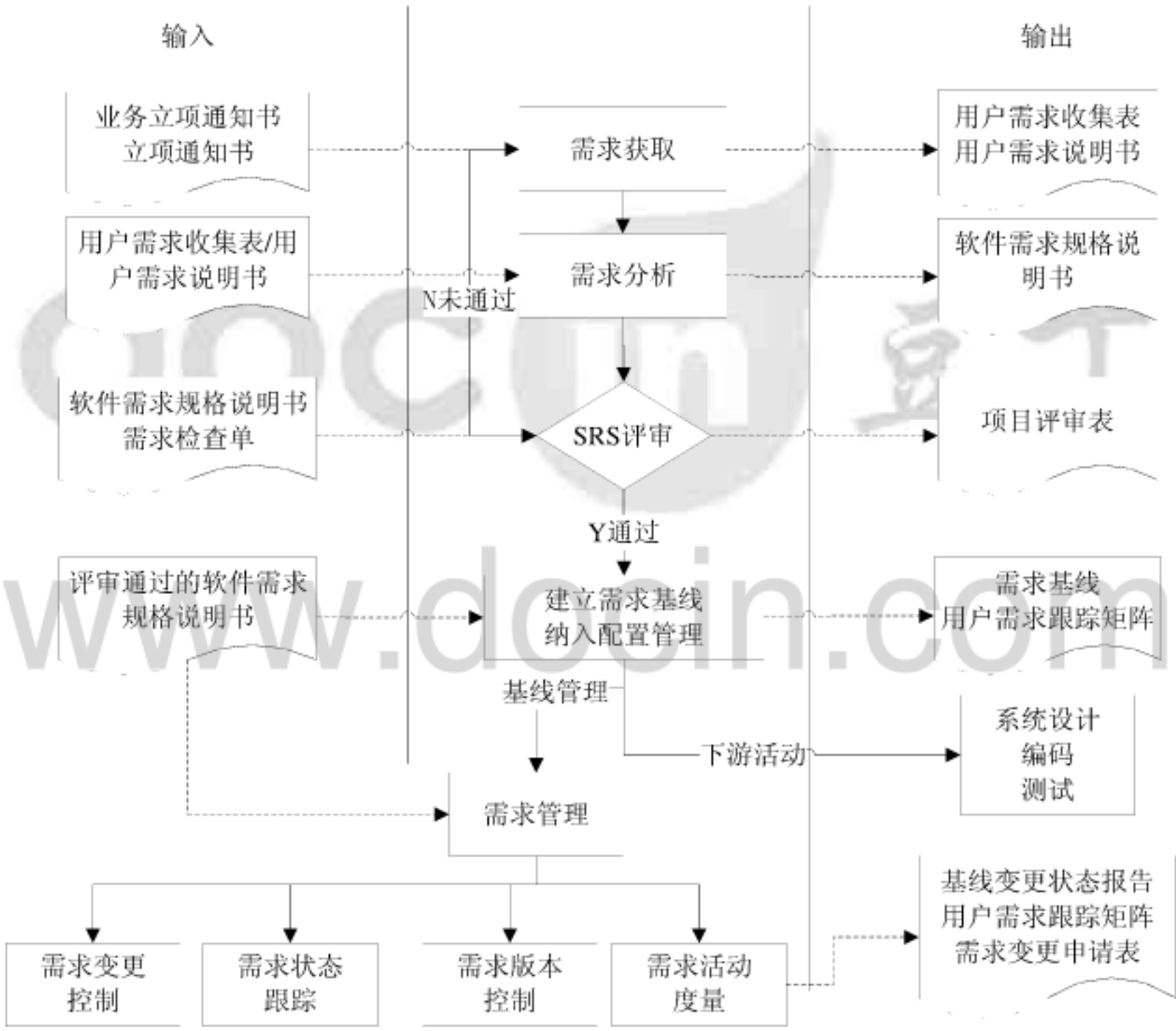
1. 为了了解软件所处的外部“系统”，必须识别硬件、软件、人员、数据库、流程和其

[8] 此部分内容通过对《软件工程——实践者的研究方法》（第 6 版）中相关内容整理编写。

他系统要素的角色，对有效的需求进行提取、分析、说明、建模、确认和管理，这些是系统工程的基础。

2. 切忌“只见树木，不见森林”，这里的“森林”是系统，而树木是实现系统所需的技术要素（包括软件）。如果在理解系统之前就匆忙构造技术要素，毫无疑问将犯错误并让你的客户失望。在关注树木之前，必须先了解森林。

图表 6-1 需求开发及管理流程图



3. 系统工程师通过和客户、未来用户以及其他共利益者一起工作以理解系统需求。

4. 需求工程是帮助软件工程师更好地理解将要解决的问题。

5. 需求工程首先定义将要解决的问题范围和性质；然后是引导、帮助客户定义需要什么；接下来是精练需求，精确定义和修改基本需求。

6. 通过用例来表示所收集的用户需求，放进用户需求说明书中。
7. 在用户收集精练之后，需要对用户需求进行确认，可以对照需求确认检查单来进行。
8. 形成需求跟踪矩阵，开展需求管理，有助于项目组在项目进展中标识、控制和跟踪需求以及变更需求。

需求常见的来源有如下几类：

1. 访问并与有潜力的用户探讨；
2. 把对目前的或竞争产品的描述写成文档；
3. 用户招标文件；
4. 对当前系统的问题报告和增强要求；
5. 市场调查和用户问卷调查；
6. 观察正在工作的用户；
7. 用户任务的内容分析——开发具体的情节或活动顺序，确定用户利用系统需要完成的任务，由此可以获得用户用于处理任务的必要功能需求。

6.4.1 需求获取活动

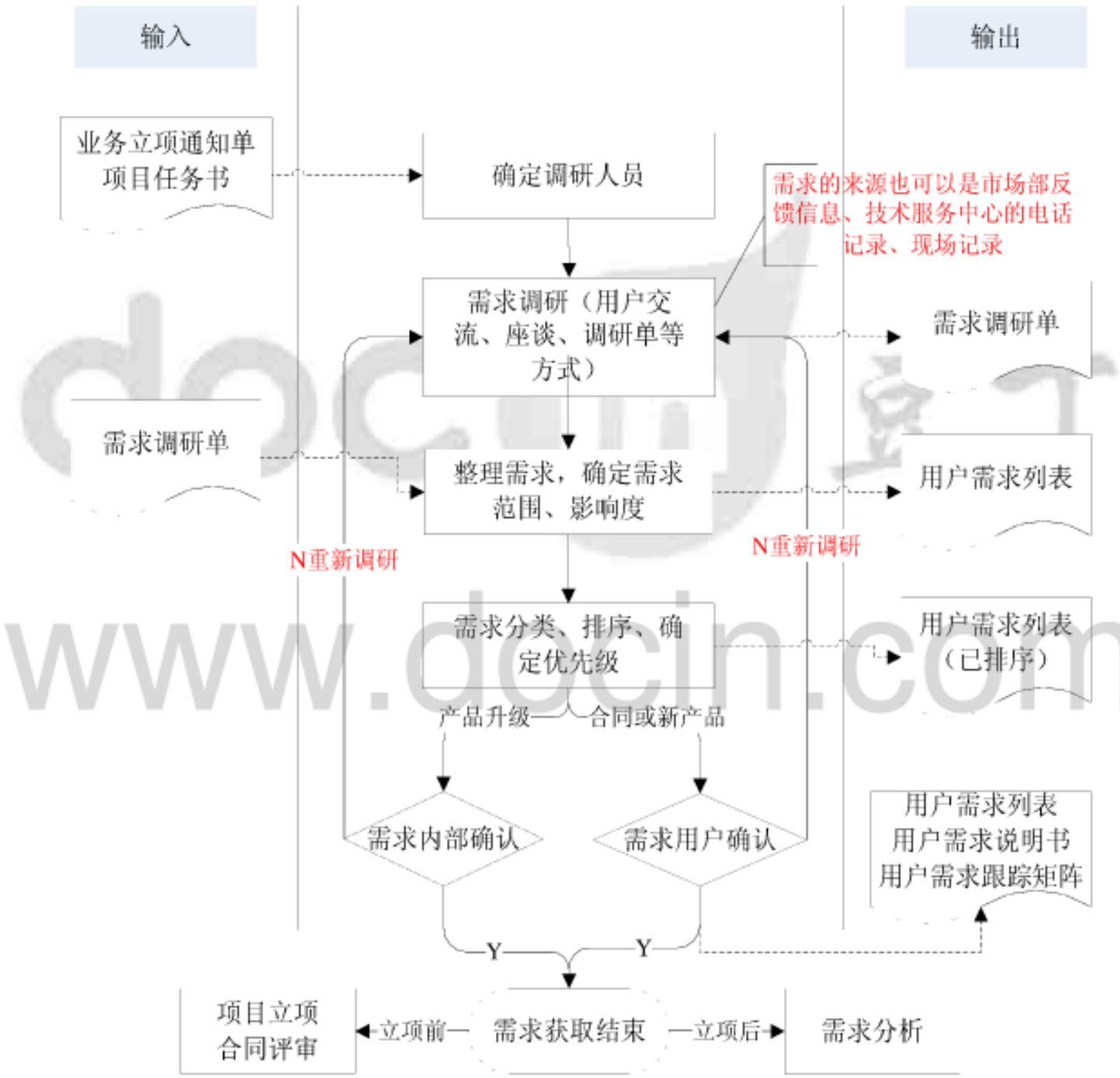
需求获取的流程如图 6-2（见下页）所示。不同类型的项目，在需求获取时会有一定的差别，采用的方法也不一样，具体描述如下：

- 合同类项目
  1. 采取与用户会谈、现场调查、等方法详细记录用户给定需求，形成《需求调研单》，然后进行需求风险识别、分类、筛选、优先排序，整理形成《用户需求列表》，并在此基础之上形成《用户需求说明书》；
  2. 研发部经理/项目经理组织会议讨论确认需求项，需要与用户方(/产品研发提出部门)就《用户需求说明书》的需求项达成一致意见，由用户或总工程师签字确认。
- 新产品研发项目
  1. 对市场上已存在的同类产品/或超前产品进行调研，或从行业标准、规则中提取需求

信息 ，形成《需求调研单》，然后进行需求风险识别、分类、筛选、优先排序，整理形成《用户需求列表》 ，并在此基础之上形成《用户需求说明书》；

2. 研发部经理/项目经理组织会议讨论确认需求项，需要与用户方就《用户需求说明书》的需求项达成一致意见，由用户或总工程师签字确认。

图表 6-2 需求获取流程图



【说明：若是立项后进行需求获取，则需要形成《用户需求说明书》；立项前的需求获取至少需要形成《用户需求列表》，然后由系分人员整理成《用户需求说明书》。在确认过的《用户需求说明书》的基础之上形成需求跟踪矩阵，一般由项目经理或其指定组员编写《用户需求跟踪矩阵》作为需求跟踪管理的基础。为了降

低需求开发及管理过程实训的复杂度，在实训时不需要编写《用户需求说明书》】

■ 产品升级类项目

1. 分析从技术支持部或实施服务人员处反馈的用户新需求、系统缺陷，进行需求分类、评估、识别优先级，并考虑系统升级问题，形成《用户需求列表》，并在此基础之上形成《用户需求说明书》。
2. 研发部经理/项目经理组织、讨论确定《用户需求说明书》，由研发部经理审核签字。

6.4.2 基于用例的需求获取

在使用用例技术来进行需求获取、精练及分析时，建议采用如下的步骤来操作，以方便更好的提练出来软件需求：

1. 定义项目的视图和范围（用户需求列表）；
2. 确定用户类；
3. 在每个用户类中确定适当的代表；
4. 确定需求决策者和他们的决策过程；
5. 选择你所用的需求获取技术；
6. 运用需求获取技术对作为系统一部分的用例进行开发并设置优先级；
7. 从用户那里收集质量属性和其他非功能需求；
8. 详细拟订用例使其融合到必要的功能需求中；
9. 评审用例的描述和功能需求；
10. 如果有必要，开发分析模型用以澄清需求获取的参与者对需求的理解；
11. 开发并评估用户界面原型以帮助想象还未理解的需求；
12. 从用户中开发概念测试用例；
13. 用测试用例来论证用例、功能需求、分析模型和原型；
14. 在继续进行设计和构造系统每一部分之前，重复 6—13 步骤。

在确定了参与者之后，需要对用例进行开发，用例开发的好坏对大家理解需求至关重要，

在检查一个用例时，可以通过回答如下问题来看该用例是否得到很好的描述。

1. 谁是主要参与者，次要参与者？
2. 参与者的目标是什么？
3. 做事开始前有什么前提条件？
4. 参与者完成的主要工作或功能是什么？
5. 按照故事/业务场景所描述的还可能需要考虑什么异常？
6. 参与者的交互中有什么可能的变化？
7. 参与者将获得、产生或改变哪些信息？
8. 参与者必须通知系统外部环境的改变吗？
9. 参与者希望从系统获取什么信息？
10. 参与者希望得知意料之外的变更吗？

用例可以安排 UML 的相关约定去描述，怎么开发用例也是《系统分析与设计》相关课程的范畴，本书不做过多的讲解。但是，在些需要提醒一下读者，在对用例进行开发过程中，我们应当避免用例陷阱：

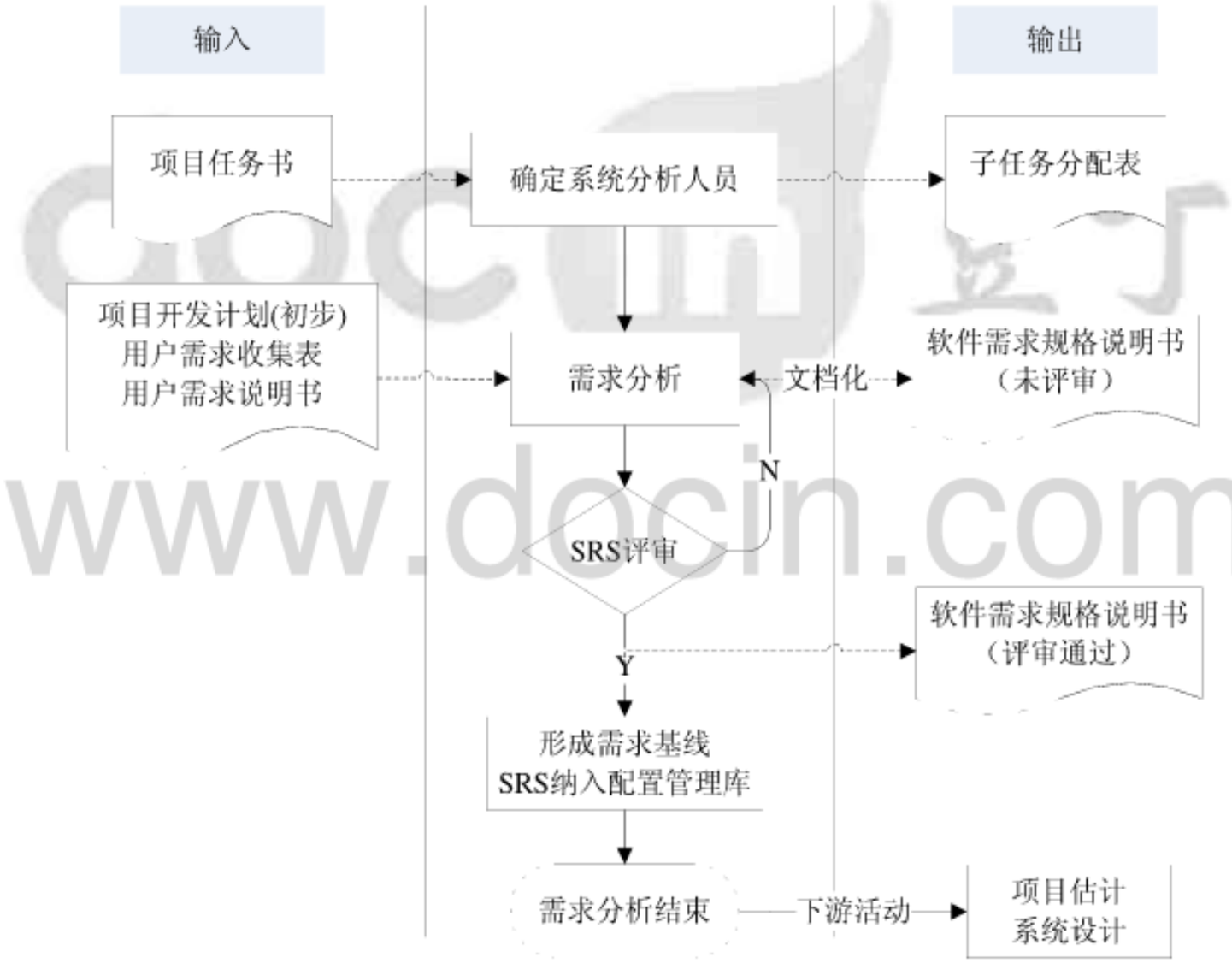
1. 太多的用例，注意合适的抽象级别。
2. 用例冗余，使用“包含”关系，将公共部分分离出来写到一个单独的用例中。
3. 用例中的用户界面设计，用例的重点就是用户使用系统做什么，而不是关心屏幕上是怎么显示的。
4. 用例中包含数据定义，比如数据类型、长度、格式和合法值等，这些应当放到数据字典里。
5. 试图把每一个需求与一个用例相联系，这是不可能的，需要使用规格说明书编写非功能需求、外部接口需求以及一些不能由用例得到的功能需求。

6.5 需求分析

需求分析是提炼、分析和仔细审查已收集到的用户需求。目的在于开发出高质量和具体

的需求，以便做出实用的项目估算并进行设计、构造和测试。在进行需求分析的过程中，需要从不同的视角检验需求，增加查明错误、消除不一致性、发现遗漏的机率。并且，创建分析模型之后，要不断改进，并分析评估其清晰性、完整性和一致性。根据需求分析使用的方法不同一般包含的元素也不同，比如：基于用例（用户场景）表现系统时，主要元素有用例文本、用例图、活动图、泳道图；基于信息流来表现系统时，主要元素有数据流图、控制流图和处理说明。具体需求分析活动的流程如图 6-3 所示，其中，SRS 是软件需求规格说明书的简称。

图表 6-3 需求分析流程图



在进行需求分析时，常用的方法有以下三种：

1. 面向结构分析法

结合获取的《用户需求说明书》，可采用数据流图等分析模型，把系统功能需求、非功能需求按事件流、数据流分析方式，逐层细化到系统操作及操作数据的存储方式，如数据的输

入、输出，并考虑外部接口。

结合数据流图和数据字典,详细说明系统功能间的输入、输出、系统活动及约束条件 ,编制需求规格说明书。

2. 面向对象分析法

使用 UML 辅助类图或其他分析方式来分析已获取的系统需求、用例模型、类图、数据字典等。结合图形化分析模型、类图、顺序图、关联图等进行说明，准确描述用户及系统的交互活动 ，编制需求规格说明书。

3. 快速原型分析法

分析已获取的用户需求，增量、迭代地明确用户工作流程、约束条件等，设定需求的优先级排序，在风险较小的基础上分析、设计和实现系统构架结构或用户界面架构。结合立项时所选用生命周期，迭代进行分析活动 ，编制需求规格说明书。

6.6 需求评审

无论是《用户需求说明书》还是《软件需求规格说明书》在定稿之后均需要进行评审，并且在评审通过之后纳入配置管理，具体执行的步骤为：

1. 《用户需求说明书》或《软件需求规格说明书》完成之后，由项目经理提出评审要求。评审之前项目经理准备评审相关资料，在总工程师或研发部经理的协助下，确定参加评审的人员，并把相关资料及评审通知一并发给参与评审的人员。
2. 按计划进行评审，指定仲裁者、主持人、记录员。通过评审对需求中存在的问题进行讨论，主要针对各评审员提交的《预审问题清单》中的问题，达成一致意见。
3. 评审结束后，记录员整理会议内容，形成《会议记录》。
4. 项目经理或者指定人员整理《项目评审表》汇总评审发现的缺陷及其修改意见，并提交相关人员签字确认。
5. 《用户需求说明书》通过评审之后，由项目经理或者指定人员编写《用户需求跟踪矩阵》供后继项目开发过程中对需求跟踪时使用;《软件需求规格说明书》评审通过

后，项目经理或者指定人员更新《用户需求跟踪矩阵》，填写相应内容。

6. 建立需求基线，把《软件需求规格说明书》、项目评审表及相关文档纳入配置管理。

6.7 需求管理

1. 需求变更控制

建立项目级的 CCB 管理需求变更；软件需求基线的变更应受控制；保持项目计划及其他项目产品与需求一致；变更产生的影响应通知相关人员；由于需求引起的所有变更过程都应受到追踪直至关闭。

变更步骤：

- 变更提出者填写《需求变更申请表》说明变更内容、变更影响范围；
- 提交 CCB 审批，必要时召开变更评审会议；对于不影响需求基线的小范围的变更，可以由项目经理审批；
- 变更完成后，由验证人和质量保证工程师跟踪验证变更引起相关内容的调整是否正确无误；
- 《需求变更申请表》纳入配置管理。

2. 需求版本控制

为评审通过的软件需求文档确定版本号；每一个经过变更软件需求文档应在修订页中描述其修正版本的历史情况，包括已变更的内容，变更日期，变更人姓名和变更后的版本号；项目经理和质量保证工程师负责检查软件需求文档版本的一致性。

3. 需求跟踪矩阵

在每次需求变更之后，必须更新需求跟踪矩阵；在每个阶段工作完成之后，亦需要更新需求跟踪矩阵。

第7章 项目估算及详细计划

内容提要：

- 软件估算简介
- 常用的估算方法
- 项目详细计划

软件项目估算是一个专门的学科，在软件开发过程中，估算不足是造成软件项目失控最普遍的原因之一。因此，本章中首先对常用的估算技术及方法进行简单介绍，然后讲解把软件估算技术怎么应用到制订项目详细计划中去。

在开始讲解本章内容之前，送各位读者一句话，希望在以后项目开发过程中能有所帮助，“向进度落后的项目增加人手，只会使进度更加落后。“——摘自《人月神话》。

7.1 软件估算简介

随着软件系统规模的不断扩大和复杂程度的日益加大，从 20 世纪 60 年代末期开始，出现了以大量软件项目进度延期、预算超支和质量缺陷为典型特征的软件危机，至今仍频繁发生。根据 Standish 组织在 1995 年公布的 CHAOS 报告显示，在来自 350 个机构的 8000 个项目中，只有 16.2%是“成功的(succeeded)”，即能在给定的预算和限期内完成；31.1%是“失败的(failed)”，即未能完成或者取消；其余 52.7%被称为“被质疑的(challenged)”，虽然完成但平均预算超支 89%。2004 年，该组织的统计项目数累计达到 50000 多个，结果显示，成功项目的比例提升到 29%，而被质疑的项目比例仍有 53%。虽然有些研究认为，CHAOS 报告中关于预算超支 89%的数据被夸大了，实际情况应该平均在 30%~40%。但有一点却能够取得共识：人们经常对软件成本估算不足，它与需求不稳定并列，是造成软件项目失控最普遍的两个原因。

那么什么是软件估算呢？软件估算是指根据软件的开发内容、开发工具、开发人员等因素对需求调研、程序设计、编码、测试等整个开发过程所花费的时间及工作量做的预测。软件估算已成为软件工程经济学（Software Engineering Economics）的重要组成部分。

估算不足与估算过多对企业都会产生影响，当估算过多的时候，肯定会使企业的整体成本增加；那么估算不足的时候，产生的问题更严重，可用下图来表示：

图表 7-1 估算不足结果图

一个良好的软件项目计划的建立，必须估算准备开发的软件项目的任务大小（即规模）、资源情况、投入成本、限制因素等，保证对这些内容进行充分的估算。最后，根据估算，才能制定出合理的项目开发计划。为了保证估算的准确性，我们可以考虑如下几点：

- 将估算拖延到项目的最后阶段，虽然是越往后估算，与实际值差距就越小，但是这在实际软件开发过程中是不可能的；
- 基于已完成的类似的项目进行估算，这需要项目组所在的机构资产库里有类似的项

目数据；

- 使用简单的“分解技术”来进行项目成本及工作量的估算，采用自顶至下或自下至上的方法对整个项目进行分解，之后再进行估算；
- 使用一个或多个估算模型或方法进行软件成本及工作量的估算，综合应用多种估算方法，这是在软件开发过程中比较行之有效的操作方法。

在开发过程中，软件估算包含的内容有：软件工作产品的规模估算；软件项目的工作量估算；软件项目的成本估算；软件项目的进度估算；项目所需要的人员、计算机、工具、设备等资源估算。

在估算过程中，通常影响估算准确性的因素有：适当地估算待建造产品的规格的程度；把规模估算转换成人的工作量、时间及成本的能力；项目计划反映软件项目组能力的程度；产品需求的稳定性及支持软件工程的工作环境。

7.2 常用的估算方法

在进行软件估算时，无论使用什么样的估算方法，通常会对估算数据进行统计学上的分析，然后得出更合理的估算结果。使用的最简单也是最有效的公式就是三点统计方法，具体是：产生一个三点或期望值估算，建立关于规模的乐观值、可能值、悲观值，然后采用如下公式进行计算：

$$EV = (S_{opt} + 4S_m + S_{pess})/6$$

S_{opt} 乐观值；

S_m 可能值；

S_{pess} 悲观值；

在实际项目估算操作过程中，无论采用任何估算技术，不管有多高明，都必须与其他方法交叉使用，即使这样，直觉和经验也是必不可少的。

7.2.1 面向规模的估算（LOC 法）

在 LOC（代码行数来估算项目规模）法中的代码行数（Line Of Code）是指所有的可执行的源代码行数，包括可交付的工作控制语言语句、数据定义、数据类型声明、等价声明、输入/输出格式声明等。注释及机器自动生成的代码不能包括在代码行计算中。

LOC 估算法的缺点：

1. LOC 依赖于程序设计语言；
2. 对设计得很好，但较小的程序会产生不利的估算；
3. 不适合非过程语言；
4. 在估算时需要一些可能难以得到的信息。

在公司里，单元编码行的价值和人月均编码行数可以体现一个软件生产机构的生产能力。公司可以根据对历史项目的审计来核算公司的单行编码价值。因此，LOC 可以做为衡量工作量的一个指标。

7.2.2 类比法

适合一些与历史项目在应用领域、环境和复杂度相似的项目，通过新项目与历史项目的比较得到规模估计。类比法估算结果的精确度取决于历史项目数据的完整性和准确度。因此，用好类比法的前提条件之一是公司要建立起较好的项目后评价与分析机制（即至少要做好项目级的度量分析，请参见第 19 章的相关内容），对历史项目的数据分析是可信赖的。

应用前提：

1. 对以前项目规模和工作量的度量是准确的（这一点是国内大部分软件企业的软肋，做的大都不到位）；
2. 至少有一个以前的项目类型与新项目类似；
3. 新项目的开发周期、使用的开发方法、开发工具与以前项目类似，开发人员的技能和经验也不能差别太大。

基本步骤：

1. 整理出项目功能列表；
2. 从机构的历史项目库中，找到类似的历史项目数据；
3. 标识出每个功能列表与历史项目的相同点和不同点，特别注意历史项目做得不够的的地方，估算出新项目中每个功能的规模；
4. 计算出整个项目的规模。

优点：

由于有类似的项目做为参考，估算结果较准确。

缺点：

要依赖于历史经验；必须要有类似的项目可供参考。

注意事项：

既然使用了该方法，那么历史项目中就有可重用的代码。对于可重用代码也需要进行规格及工作量的估算，具体操作方法为：由开发人员或系分人员详细考查已存在的代码，估算出新项目可重用的代码中需要重新设计代码百分比、需要新编码或修改的代码百分比、重新测试的代码百分比，由此可计算出重用这些代码等价代码行数：

$$\text{等价代码行} = \frac{\text{重新设计\%} + \text{重新编码\%} + \text{重新测试\%}}{3} \times \text{已有代码行}$$

7.2.3 面向功能的估算（FP 法）

功能点估算方法（Function Point，简称 FP）是 1975 年，由 IBM 的工程师 Allan Albrecht 首先提出。1979 年，IBM 正式向外界公布该方法；1984 年，国际功能点用户组成立。目前，功能点法已成为具有广泛影响的软件测量方法。FP 估算法是在需求分析阶段基于系统功能的一种规模估算方法，即功能点估算法。

步骤：

1. 通过研究系统需求，确定外部输入、外部输出、外部查询、内部逻辑文件和接口文件的数量。

2. 将这些数据进行加权乘，外部输入——4；外部输出——5；外部查询——4；内部逻辑文件——10；外部接口文件——10。

3. 估算者根据对复杂度的判断，总数可用 25%、0 或－25%来调整。

优点：

对项目早期的规模估计很有帮助，能保持与需求变化的同步。

缺点：

加权调整需要依赖个人经验。

- 功能点方法被广泛的认可在信息系统、数据库密集型、4GL（第四代语言）应用系统开发的规模测量中应用。
- 功能点与软件成本具有明显的成本估算关系。
- 功能点转化为代码行：

C ——128LOC/FP

C++——53 LOC/FP

VB——29 LOC/FP

Java——46 LOC/FP

VC++——34 LOC/FP

PL/SQL——13 LOC/FP

在转化为代码行数之后，就可以使用 IBM 模型，对工作量、人员数量、项目周期等进行估算。IBM 模型具体计算公式如下：

E=5.2×L×0.91，L 代码行数（以 KLOC 计），E 工作量（以人月计）

S=0.54×E×0.6，S 是人员需要量（以人计）

D=4.1×L×0.36，D 是项目持续时间（以月计）

DOC=49×L×1.01，DOC 文档数量（以页计）

7.2.4 面向用例(UCP)的估算

用例点方法（Use Case Point，简称 UCP）是由 Rational 公司（当前为 IBM 公司的下属公司）的 Gustav Karner 于 1993 年提出，在对用例的分析基础之上进行加权调整得出。

步骤：

1. 对每个用例、角色分别赋值加权乘积求和；
2. 计算未调整用例点（Unadjusted Use Case Point UUCP）；
3. 考虑环境因子和技术因子对 UUCP 调整以得到调整用例点（Adjusted Use Case Point AUCP）；
4. 再根据 AUCP 和规模——工作量转换因子得到工作量。

优点：

1. 比较适用于面向对象的软件项目。
2. 经过调整可用于估算测试工作量。

缺点：加权调整需要经验。

UCP 计算的详细步骤：

- 确定和计算技术复杂度因素值（Technical Complexity Factor，简称 TCF） = 0.6 +（0.01×Total Factor），参见表 7-2；
- 确定和计算环境复杂度因素值（Environmental Complexity Factor，简称 ECF） = 1.4 + (-0.03×Total Factor)，见表 7-3；
- 将所有用例中的角色进行汇总分类，计算未平衡角色因素值（Unajusted Actor Weight，简称 UAW），见表 7-4；
- 将所有用例分类，计算未平衡用例因素值（Unadjusted Use Case Weight，简称 UUCW），见表 7-5；
- 计算未平衡用例点数（Unadjusted Use Case Point，简称 UUCP），UUCP = UAW + UUCW

- 确定本机构的工作效率因素（Productivity Factor），用来描述每一个用例点需要多少人时来完成，业界通常采用的数据是 15-30 之间，如果开发组没有历史数据，则可以采用 20。即每个 UCP 点，需要 20 人时来完成。
- 计算用例点数（用此来表明项目的工作量，跟踪时用实际工作量与此处估算的工作量进行对比分析即可），UCP = TCP × ECF × UUCP×PF

其中各个因素及数值的计算如以下图表所示：

图表 7-2UCP 技术复杂度因素（TCF）表

技术因子	描述	权重	复杂度	计算后因素值
T1	系统分步式程度	2	0	0
T2	系统性能目标要求度	1	0	0
T3	终端用户使用效率要求度	1	0	0
T4	内部处理流程复杂度	1	0	0
T5	系统或代码复用程度	1	0	0
T6	易于安装要求度	0.5	0	0
T7	系统易于使用程度	0.5	0	0
T8	移植性	2	0	0
T9	系统易于扩展或升级要求度	1	0	0
T10	并发性要求度	1	0	0
T11	特殊安全功能特性要求度	1	0	0
T12	为第三方系统提供直接系统访问	1	0	0
T13	是否需要特殊的用户培训设施	1	0	0

图表 7-3UCP 环境复杂度因素(ECF)表

环境因子	描述	权重	影响度	计算后因素值
E1	UML 精通程度	1.5	0	0
E2	系统应用经验	0.5	0	0
E3	面向对象经验	1	0	0
E4	系统分析员能力	0.5	0	0
E5	团队士气	1	0	0
E6	需求稳定度	2	0	0

环境因子	描述	权重	影响度	计算后因素值
E7	兼职人员比例高低	-1	0	0
E8	编程语言精通程度	2	0	0

图表 7-4UCP 未平衡角色因素（UAW）表

角色类型	描述	权重	角色数量	计算结果
Simple	角色通过定义好的 API 接口与另一系统交互	1	0	0
Average	角色通过某种协议（比如 TCP/IP 等）与另一系统交互	2	0	0
Complex	系统的最终用户	3	0	0

【说明：1、角色不单纯的指使用系统的最终用户，也可以指本系统以外的其他系统。2、在 UCP 估算方法中将使用系统的最终用户定义为复杂(Complex)角色。】

图表 7-5UCP 未平衡用例因素（UUCW）表

用例类型	描述	权重	用例数量	计算结果
Simple	满足以下任何一种情况的用例均算作简单用例： 1.用例中只含有一个简单用户界面或接口并且仅使用到一个单一数据库实体的用例。 2.用例操作步骤在 3 步以内的用例。 3.它的实现涉及到 5 个类以内的用例。	5	0	0
Average	满足以下任何一种情况的用例均算作普通用例： 1.用例中含有多于一个用户界面或接口并且仅使用到 2 个以上数据库实体的用例。 2.用例操作步骤在 4 到 7 步以内的用例。 3.它的实现涉及到 5 到 10 个类的用例。	10	0	0
Complex	满足以下任何一种情况的用例均算作复杂用例： 1.用例中含有多于一个复杂用户界面(或接口)或处理流程并且仅使用到 3 个以上数据库实体的用例。 2.用例操作步骤在 7 步以上的用例。 3.它的实现涉及到 10 个类以上的用例。	15	0	0

说明：以上表格中的权重（Weight）均是设定值，不要改动，只需要填写深颜色处的数字即可。具体填写方

式请参见实训指导中有关 UCP 估算指导的章节。

7.2.5 基于过程的估算

1. 自顶向下法（Top-Down）

首先对整个系统进行总工作量估算（根据立项报告或合同等），再将过程分解为相对较小的活动或任务，再估算每个任务的规模及完成任务所需的工作量，总工作量逐步分解到各组成部分的工作量，并考虑开发软件所需资源、人员、质量保证、系统安装等的工作量。

优点：

估算工作量小，速度快。

缺点：

对项目中的特殊困难估计不足，估算出来的工作量盲目性大，有时会遗漏被开发软件的某些部分。

比如：接到一个周期限为六个月的项目，项目经理可能做如下估算：

一个月：需求分析；

一个月：系统设计；

两个月：编码；

两个月：联调，测试，改错；

再根据这个估算对每个阶段进一步估算和规划。

2. 自底向上法（Bottom-Up）

该方法是按组件或子功能划分，先对每个组件的工作量估算，然后总计得到整个项目的规模和工作量。

优点：

估算各个部分准确性高；能提高参与人的责任心；

缺点：

缺少各个子任务之间相互联系所需的工作量，还缺少许多与软件开发有关的系统级工作

量（配置管理、质量管理、项目管理等）。所以往往估算值偏低，必须用其他方法进行检验和校正。

7.2.6 Delphi 法详解

Delphi 估算法（Expert Judgment）是一种专家估算技术，在没有历史数据的情况下，这种方式适用于评定过去与将来新技术与特定程序之间的差别，但专家“专”的程度及对项目理解的程度是工作中的难点，但是这种方式对决定其他模型的输入时特别有用。

步骤：

1. 协调人向各专家提供项目规格和估算表格；
2. 协调人召集小组会，各专家讨论与规模相关的因素；
3. 各专家匿名填写估算表格；
4. 协调人整理出一个估计总结，并返回专家；
5. 协调人召集小组会议，讨论差异较大的估算；
6. 专家复查估算，并提交另一个匿名估算；
7. 重复以上步骤，直到达到一个最低和最高估算的一致。

优点：

不需要历史数据；非常适合新的较为特别的项目估算。

缺点：

主观；专家的判断有时并不准确；专家自身技术水平不够时会带来误判。

需要估算的内容描述须含义明确，无二义性。规定统一的评价方法，如三点法，确定完成任务的三种可能估算值：乐观估算，最好情况下的估算值 O；悲观估算，最糟情况下可能的估算值 P；正常估算，一般情况下任务完成估算值 M；最后估算值 E，使用公式：E =（ O + 4M + P）/ 6 得出。结合三点法的 DELPHI 法，称为 Wideband-Delphi。

在使用 Delphi 法进行估算进，参与估算的群体中每人独立做出估算。估算的组织者要忠实于群体的回答，在任何情况下不表露自己的倾向。

图表 7-6 及图表 7-7 是关于 Delphi 用于软件规格估算和软件工作量估算的两个示例供大家参考。更详细的内容请参见本书光盘“模板——第 7 章项目估算及详细计划——《Delphi 估算表》”。

图表 7-6 示例 1：软件规模估算的 Delphi 汇总表

项目名称及版本		项目经理	
项目类型		质量保证工程师	
项目组成员			
估算汇总人员		估算日期	

估算内容												
序号	估算内容	估算单位	组员 1	组员 2	组员 3	组员 4	组员 5	估算结果				备注
								乐观值	悲观值	平均值	估算结果	
1		页数	1.2	3	2.1	2.3	2.1	1.2	3	2.17	2.14	
2		页数	1.2	2.1	2.1	2.3	2.1	1.2	2.3	2.1	1.98	
3		代码行	1.2	2.1	2.1	2.3	2.1	1.2	2.3	2.1	1.98	
4		页数	1.2	2.1	2.1	2.3	2.1	1.2	2.3	2.1	1.98	
5		代码行	1.2	2.1	2.1	2.3	2.1	1.2	2.3	2.1	1.98	
6		代码行	1.2	2.1	2	2.3	0.8	0.8	2.3	1.77	1.69	
7		页数	1.2	2.1	2.1	2.3	2	1.2	2.3	2.07	1.96	
8		页数	1.2	2.1	2.1	2.3	2.1	1.2	2.3	2.1	1.98	
11	项	合计:	10.6	18.7	17.9	20	18.4				17.23	

图表 7-7 示例 2：工作量估算的 Delphi 汇总表格

项目名称及版本号				项目经理					
项目类型				质量保证工程师					
项目组成员									
估算汇总人员	估算单位		人时	估算日期					
估算内容									
序号	估算内容	组员 1	组员 2	组员 3	组员 4	组员 5	估算结果				备注
							乐观值	悲观值	平均值	估算结果	
1	前期策划									0	
2	CM 活动									0	
3	QA 活动									0	
4	测试工作量									0	
5	项目跟踪									0	
6	项目例会									0	
7	项目评审									0	
11	项	3.4	7.3	7.5	8.5	9.3				7.26	

【说明：以上表中，项目类型为“新产品研发类”、“合同定制类”、“产品升级类”三者之一。】

7.3 项目详细计划

在软件需求规格说明书通过评审之后，进行项目计划的细化工作，通过项目计划细化形成详细的项目开发计划书。

对于中小型的项目，在 SRS 通过评审一周内，项目经理需要完成项目计划细化工作，对于大型项目，时间需要延长的，需要由总工程师批准。

项目详细计划完成的主要内容为：

- 估计软件工作产品规模及项目的工作量；
- 根据工作量估计项目成本；
- 估计项目关键计算机资源及项目风险；
- 编写详细项目计划及相关的专项计划；
- 形成定稿的项目开发计划书并进行评审。

在具体进行项目详细计划时，一般可按以下步骤进行项目计划细化：

- 确定估计策略；
- 软件工作产品规模估计；
- WBS 细化；
- 项目工作量估计；
- 项目成本估计；
- 关键计算机资源估计；
- 项目风险估计。

一、项目计划细化—确定估计策略

根据项目的类型、分配的软件需求、软件生命周期和风险估计状况等，结合以往项目的历史数据，制定估计策略。

在项目计划时，根据立项估计数据范围和需求分析的结果，再次估计项目的进度，工作量和成本范围。

识别出现成的可重用在本项目的工作产品（包括需求，设计代码，测试计划和用例等），估计修改和使用这些重用部分的工作量和规模。

在此过程中，会形成如下数据：本项目估计数据、实际数据和修改后的项目数据及相关修改信息；用来作为估计参考的其他项目实际数据；项目间的差异和差异解释。这些数据均需收集，并在项目完成后形成总结文档，作为评价本项目估计正确与否的依据，汇总到机构过程资产库中，为将来的项目估计提供实际数据参考。

二、项目计划细化—规模估计

对已识别的项目工作产品及其活动做出规模估计，包括项目中的子系统数、需求数、代码行数、新产生的文档和重用已有的文档的页数等。

在适当的时候（比如：系统设计完成后），再次估计软件规模，包括新增加的软件部分和重用已有的软件部分（模块数，代码行数/页面数/界面数）。

可使用代码行估算（代码行数包括新增的和修改的代码，以及注释，不包括空格）或功能点分析法来估计软件规模，在估计过程中结合 DELPHI 方法估计软件规模。如果选用其他机构过程资产库里没有的估算方法，其估算过程必须经过 EPG 的批准；经批准的估算过程纳入机构的过程资产库。对于新研发类项目建议使用 UCP 法，对于产品升级类项目建议使用代码行估算。

记录项目规模估计所选择的开发语言、估计的工作分解对象、选定的估计方法、影响估算的风险、可重用的软件部分、参加估计人员、开发方法、所定的假设与推理条件等背景资料，估计过程中形成的数据及最终形成的估计结果，形成项目开发计划书中相应部分。

当项目开发计划发生重大偏离、有必要调整和项目任务细分时，采用原估算方法依据要求对软件规模进行重新估算，并修改项目开发计划书中相应部分或单独形成文档，以便为进行中和将来的项目提供参考依据。

三、项目计划细化—**WBS** 细化

在项目细化阶段，WBS 一般细化到 3—5 级，根据不同项目由项目经理确定细分的级别，并且划分到每个任务完成时间不超过 3 人天。

第 3 级——过程，把每个阶段划分为几个过程。

第 4 级——单元，把每个过程划分为单元（子系统、模块）任务。

第 5 级——任务，一般为低层的详细任务。

建议详细计划的分解粒度为一个人 3 天内可以完成的任务。根据项目大小，由项目经理确定划分的级别。

在小项目及需求明确且稳定的项目中，要在项目开始阶段制订完整的 WBS。分解的粒度以满足管理和估计的需要为准。

较大的项目，WBS 必须随着对要做的工作的深入了解而不断完善。一般的途径是在项目的尽早阶段定义出高层形式的 1—3 级的 WBS，然后在做详细计划时建立低层 4—5 级或更高级成分。

项目的 WBS 中还应该定义出管理和支持活动，如项目管理、配置管理、质量保证。所要遵循的指导原则与上述活动相同。大多数管理和支持活动是在项目的各阶段中持续执行的：它们随项目或阶段的启动而开始，随项目或阶段的结束而终止。只需把这些连续的工作细分到适当的程度以便于估计和监控即可。

四、项目计划细化—工作量估计

如果合适，以其他类似项目的历史数据作为估计的参考数值；在同一个项目中，工作量的单位必须一致，一般可采用“人日”、“人时”等单位。

根据工作分解结构图（WBS），采用 DELPHI 方法或其他方法对技术活动（包括开发过程中的需求、设计、编码、测试、支持活动、项目管理）进行估计。

估计时，记录工作量估计所选择的开发语言、估计的工作分解对象、执行任务人员、开发方法、所定的假设与推理等背景资料，估计过程中形成的数据及最终形成的估计结果，形成项目开发计划书中相应部分。

当项目开发计划发生重大的偏离、有必要调整和项目任务细分时，按照步骤 5 的要求对工作量进行重新估计，并修改项目开发计划书中相应部分或单独形成文档，以便用于进行中和将来的项目提供参考依据。

五、项目计划细化—成本估计

可以根据以往经验，并以其他类似项目的历史成本数据作为估计的参考数值。项目经理根据本项目所需资源、工作量和工作环境要求等估计项目成本范围。

内容主要包括：所需的成本内容，金额，到位时间等。

成本内容一般包括：直接的员工工资，管理费，差旅费，软硬件费用、资料费、里程碑活动经费等。将估算结果写入项目开发计划书中相应部分。

六、项目计划细化—关键计算机资源估计

可以根据以往经验，软件需求和其它可用信息识别关键计算机资源。关键计算机资源包括：开发环境、集成测试环境和用户环境中所用的计算机资源。

估计项目组分析项目开发、测试及用户使用产品所需要的软硬件资源。

估算关键计算机资源时，需要考虑工作产品的规模、软件的运行负载、通信量、内存容量等相关联因素，并预留一定得扩展范围。将估算结果，写入项目开发计划书中相应部分。

七、项目计划细化—风险估计

根据风险管理章节（第 9 章）所讲知识进行风险估计，并把首要风险列表、风险规避措施、缓解方案及相关负责人，跟踪周期等写入开发计划中。具体内容及方法会在风险管理章节中讲解。

八、项目计划定稿

根据以上的估计，形成项目进度表，建议采用 MS Project 编制，便于软件生命周期全过程的跟踪和更新维护。项目经理在制定项目开发计划时，需要识别本项目与相关组和个人进行协调沟通的内容，在项目开发计划中注明。各个专项计划完成，定稿或者并入开发计划。

项目组及相关组和个人参与项目开发计划的非正式评审，在充分讨论、分析的基础上，各方达成对计划进度、任务分配、项目规模估计等的一致意见，项目开发计划书定稿。

在项目详细计划时，除了形成主的项目开发计划之外，一般还会形成以下专项计划，以便更好的对整个项目进行管理：

- 根据配置管理章节（参见本书第 8 章）所讲内容形成配置管理计划；

- 根据软件产品及过程质量保证章节（参见本书第 19 章）所讲内容形成质量保证计划；
- 根据风险管理章节（参见本书第 9 章）所讲内容形成风险管理计划；
- 根据编码与测试相关章节（参见本书第 13、14 章）所讲内容形成测试计划（在项目后继阶段——详细设计完成之后——编写）；
- 根据项目团队的组成，以及项目要求，形成培训计划；
- 根据度量分析章节（参见本书第 19 章）所讲内容形成度量分析计划。

在项目开发计划及相对应的专项计划定稿之后，需要对这些计划一起进行正式的评审，做为项目后继工作开展的基础。在评审时一般按如下原则进行：

- 通过项目开发计划评审，使得对项目开发计划及专项计划达成一致的内部承诺和外部承诺。
- 建议开发计划进行正式评审，对于大型项目或合同类项目应当请主要负责人、市场部经理、客户代表参加。
- 达成承诺并得到批准后的项目开发计划书是项目跟踪和监督的基础，由项目经理提交给配置管理员，将其纳入配置管理库，形成计划基线；若使用 MS Project 编制项目进度表，则在项目进度表上设置比较基准，以方便计划数据与实际数据的比较分析。
- 项目经理在项目进展过程中对项目开发计划的内容进行跟踪并及时更新。

第8章 软件配置管理

内容提要：

- **CMMI** 对应实践
- 配置管理基本概念
- 配置管理活动
- 产品发布流程
- 配置管理工具介绍

软件配置管理（SCM——Software Configuration Management）是 ISO9001 和 CMM Level2 中的重要组成元素。它在软件产品开发的生命周期中，提供了结构化的、有序化的、产品化的管理软件工程的方法，是软件开发和维护的基础。软件配置的概念源于美国空军，为了规范设备的设计与制造，美国空军 1962 年制定并发布了第一个配置管理的标准“ AFSCM375-1，CM During the Development & Acquisition Phases”。

SCM 是指通过技术及行政手段对软件产品及其开发过程和生命周期进行控制、规范的一系列措施和过程，它通过控制、记录、追踪对软件的修改和每个修改生成的软件组成部件来实现对软件产品的管理。SCM 可以协调软件开发使得混乱减到最小，是一种标识、组织和控制修改的技术，目的是使错误达到最小并最有效地提高生产效率。SCM 使软件产品变为受控的和可预见的，它控制这样几个问题：

- 谁做的变更？ WHO
- 软件有什么变更？ WHAT
- 什么时间做的变更？ WHEN
- 为何要变更？ WHY

通过实施 SCM，可以达到可重用过程制度化，包括：满足机构的政策方针、计划和过程

描述文档化、分配适当资源（包括资金，人员和工具）、确定责任和权限、培训相关人员、通过不同级别的管理方法和纠正活动检测状态。

置于 SCM 之下的工作产品包括发送给用户的软件产品（如软件需求文档，软件代码），用于内部使用的软件工作产品（比如：项目过程描述），和用于创建工作产品的工具等（比如：操作系统、数据库、开发工具）。

SCM 还用于建立和维护软件工作产品基线。基线是由配置项及相关实体组成的，包括组成软件产品的相关版本、设计、代码、用户文档等。它是软件生命周期中各开发阶段末尾的特定点，即里程碑。通过正式的技术评审而得到的软件配置的正式文本才能成为基线，它的作用是使各个阶段工作的划分更加明确化，使本来连续的工作在这些点上断开，以便于检验和肯定阶段成果。基线是配置项继续发展的一个固定基础。

实施软件配置管理不论是对软件开发者、测试者、项目经理、质量保证工程师，还是客户都将会获得很多好处：有助于规范团队各个角色的行为，同时又为各个角色之间的任务传递和交流提供无缝的接合；能帮助项目经理更好地了解项目的进度、开发人员的负荷、工作效率和产品质量状况、交付日期等信息。

8.1 CMMI 对应实践

在 CMMI 中有一个专门的过程域与本章相关，即配置管理（Configuration Management，简称 CM）。其目的是通过配置标识、配置控制、配置状态报告和配置审计等活动，建立和维护工作产品的完整性。应纳入配置管理的工作产品包括：提交给客户的产品，指定的内部工作产品，获得的产品、工具，以及被用于构建和描述这些工作产品的其他项。工作产品的配置可以划分为不同等级的粒度，有时可以将一个配置项进一步分解成配置部件和配置单元。基线为配置项的持续变更提供稳定的基础。借助于配置管理系统的配置控制、变更管理和配置审计功能，使基线变更和工作产品发布得到监督和控制。本过程域不仅应用于项目的配置管理，而且应用于机构工作产品的配置管理，比如机构标准过程集、机构过程资产和复用库。共有 7 个实践与之相对应，分别为：

SG 1 Establish Baselines（建立基线），建立已识别工作产品的基线。

SP1.1 Identify Configuration Items（识别配置项），标识将要置于配置管理之下的配置项、组件和相关的工作产品。项目开发阶段（从软件需求评审开始到产品发布为止）各项工程和管理活动所生成的、应纳入配置管理的全部工作产品均应作为配置项。会产生一个配置项列表，可以通过如下几步完成该实践：一是根据一个文档化的准则，选择配置项以及组成配置项的工作产品；二是给配置项分配唯一标识；三是确定每个配置项的重要特征；四是确定何时把配置项置于配置管理之下；五是标识每个配置项的拥有者/负责人。

SP1.2 Establish a Configuration Management System（建立配置管理系统），建立和维护配置管理和变更管理系统，控制工作产品的完整性。一般会产生如下工作产品：配置管理系统及受控的工作产品，配置管理系统访问控制规程，变更请求数据库。可以通过如下几步完成该实践：一是建立某种机制来管理配置管理的多种控制层次；二是存储和检索配置管理系统中的配置项；三是在配置管理系统的控制层次之间共享和传输配置项；四是存储和恢复配置项的存档版本；五是存储、更新和检索配置管理记录；六是从配置管理系统创建配置状态报告；七是保留配置管理系统的内容；八是必要时修订配置管理结构。

SP1.3 Create or Release Baselines（建立或发布基线），创建或者发布基线，供内部使用或提交给客户。一般会形成基线和基线描述两种工作产品，可以通过如下几步完成该实践：一是在创建或者发布配置项的基线之前，获取来自配置控制委员会（CCB）的授权；二是创建或者发布来自于配置管理系统中配置项的基线；三是记录包含于基线中的配置项集合；四是使得最新基线集合可以方便使用。

SG 2 Track and Control Changes（跟踪并控制变更），跟踪和控制配置管理下工作产品的变更。

SP2.1 Track Change Requests（跟踪变更申请），变更申请不只是关于新的或变更的工作产品，还包括工作产品中的错误及缺陷。对变更申请应进行分析，以确定变更会对工作产品、相关工作产品、预算和进度产生的影响，一般是通过变更申请表/单来完成该工作。

可以通过如下几步完成该实践：一是将变更请求录入变更请求数据库；二是分析变更的影响以及变更请求中提出的处理建议；三是评审将在下一个基线中解决的变更请求，受变更影响的项目组成员应参加评审，并取得他们的认可；四是跟踪变更请求的状态直到关闭。

SP2.2 Control Configuration Items（控制配置项），主要是控制配置项的变更，一般会形成配置项的修订历史和基线的存档两种工作产品。可以通过如下几步完成该实践：一是在产品整个生命周期内控制配置项的变更；二是在被变更的配置项进入配置管理系统之前获得合适的授权；三是以适当的方式将受变更的配置项检入和检出配置管理系统，确保配置项的正确性和完整性；四是执行评审以确保变更没有对基线产生预想不到的影响（例如，确保变更没有危害系统的安全性）；五是在必要时，记录对配置项的变更和变更的理由。

SG 3 Establish Integrity（建立完整性），建立和维护基线的完整性。

SP3.1 Establish Configuration Management Records（建立配置管理记录），建立和维护描述配置项的记录。一般会形成如下工作产品：配置项的修订历史记录，变更日志，变更请求文字说明，配置项的状态，基线间的差别。可以通过如下几步完成该实践：一是尽量详细地记录配置管理活动，以便了解每个配置项的内容、状态以及先前版本；二是确保项目相关各方能访问并知道配置项的配置状态；三是确定基线的最近版本；四是确定组成特定基线的配置项的版本；五是描述相邻基线之间的差别；六是必要时修订每个配置项的状态和历史（比如变更和其他纠正措施）。

SP3.2 Perform Configuration Audits（实施配置审计），执行配置审计以维护配置基线的完整性。一般会形成如下工作产品：配置审计结果报告，纠正措施。可以通过如下几步完成该实践：一是评估基线的完整性；二是确保配置管理记录正确地识别配置项；三是评审配置管理系统中配置项的结构和完整性；四是确保配置管理系统中配置项的完备性和正确性；五是确保与配置管理标准和规程的符合性；六是跟踪执行源自审计的纠正措施直到关闭。

8.2 配置管理基本概念

在 20 世纪 60 年代末 70 年代初提出，当时加利福利亚大学圣巴巴拉分校的 Leon Presser 教授在承担美国海军的航空发动机研制合同期间，撰写了一篇名为“Change and Configuration Control”的论文，提出控制变更和配置的概念，这篇论文同时也是他在管理该项目（这个过程进行过近一千四百万次修改）的一个经验总结。

Leon Presser 在 1975 年成立了一家名为 SoftTool 的公司，开发了配置管理工具：Change and Configuration Control（CCC），这是最早的配置管理工具之一。国外已经有 30 多年历史的软件配置管理，在国内的发展却是在 21 世纪这几年的事，在上世纪九十年代，国内的软件企业很少会重视软件的配置管理。

配置管理包含版本控制、工作空间管理、并行开发控制、过程管理、权限管理、变更管理等内容。软件配置管理是在贯穿整个软件生命周期中建立和维护项目产品的完整性，基本目标是：

- 软件配置管理的各项工作是有计划进行的；
- 被选择的项目产品得到识别，控制并且可以被相关人员获取；
- 已识别出的项目产品的更改得到控制；
- 使相关组和个人及时了解软件基准的状态和内容。

为了更好的理解配置管理，如下几个基本概念需要了解：

一、配置库

决定配置库的结构是配置管理活动的重要基础。一般常用的是两种组织形式：按配置项类型分类建库和按任务建库。

按配置项的类型分类建库的方式经常为一些咨询服务公司所推荐，它适用于通用的应用软件开发机构。这样的机构一般产品的继承性较强，工具比较统一，对并行开发有一定的需求。使用这样的库结构有利于对配置项的统一管理和控制，同时也能提高编译和发布的效率。但由于这样的库结构并不是面向和各个开发团队的开发任务的，所以可能会造成开发人员的

工作目录结构过于复杂，带来一些不必要的麻烦。

而按任务建立相应的配置库则适用于专业软件的研发机构。在这样的机构内，使用的开发工具种类繁多，开发模式以线性发展为主，所以就没有必要把配置项严格的分类存储，人为增加目录的复杂性。因此，特别是对于研发性的软件机构来说，还是采用这种设置策略比较灵活。

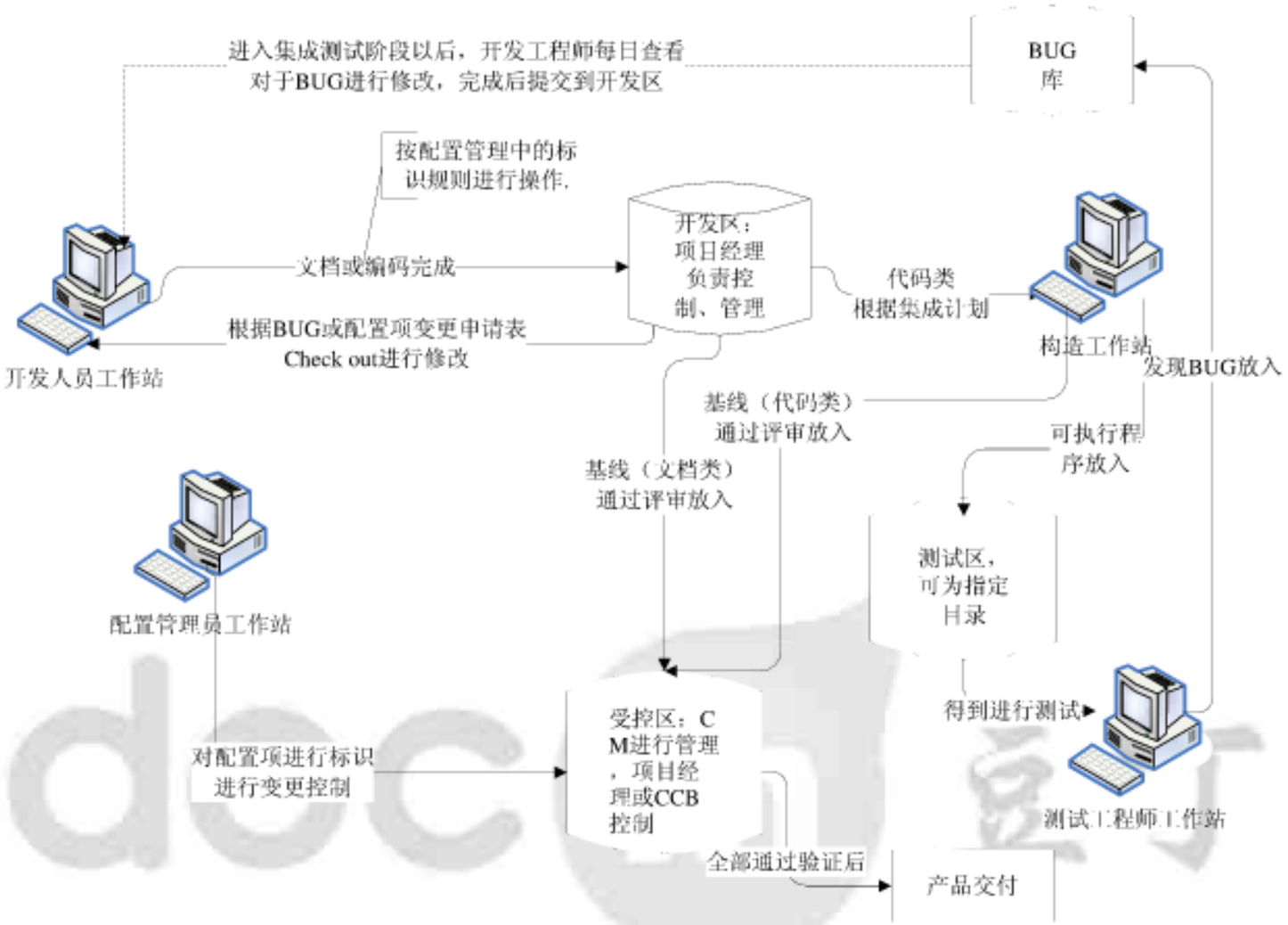
配置库的日常工作是一些事务性的工作，主要保证配置库的安全性，包括：对配置库的定期备份、清除无用的文件和版本、检测并改进配置库的性能等。

在项目开发过程中，配置库可分开发区、受控区和测试区三个区域，其各自存放的内容及存取的规定为：

- 开发区：开发区存放项目组所遵循的过程标准、参考资料、所有未经批准的配置项、已经批准但未纳入基线的配置项，此区域中的配置项由项目经理负责和控制，项目总结结束后删除。
- 受控区：受控区存放基线。此区域的配置项由项目经理或 CCB 评审批准后，由配置管理员从开发区更新而来，此区属配置管理员所有。
- 测试区：该区仅为临时区，不作详细规定，测试通过后需删除该区。测试内容也可由配置管理员从受控区获取（get latest）到指定的路径进行测试。

在图 8-1（见下页）中，给出了某公司研发部门在开发过程中对这三个区的使用控制流程图，供各位读者参考。此公司使用的配置管理工具为 Visual SourceSafe，Bug 管理使用的是 Test Director，在构造工作站上编写了用于自动构造的脚本，每天晚上可以完成自动构造。各位读者在使用时，可以根据自身使用的配置管理工具和开发工具进行调整，以最大限度的方便项目开发使用。

图表 8-1 配置管理库使用建议流程图



软件工程师按如下原则使用配置库：

- 只能访问开发区。
- 在添加配置项后，按公司版本的约定打标识，给定一个初始版本；
- 签入/签出不需要更新标识；
- 当工作产品完成之后，签入后，按公司版本约定打标识；
- 如果需要再修改，则签出；
- 修改完成后签入，三级或四级版本号加一，按打上面版本约定打标识；
- 依次类推，直到该配置项完全定稿。

【注意：文档类工作产品必须保持与封面上版本的一致性】

配置管理员按如下原则使用配置库：

- 拥有配置库的全部权限，建立配置库并分配操作权限；
- 把评审通过的配置项根据评审后确定的版本，打上版本标识；

- 根据审计过的版本控制表生成基线，从开发区把配置项移到受控区；之后，锁定该版本的工作产品；
- 负责配置库的日常维护及备份；
- 发布时定期或事件驱动从配置库生成配置状态报告。

测试工程师按如下原则使用配置库：

- 测试工程除了对测试区域及公共区域有权限外，其他区域均无操作权限；
- 当一个系统/变更测试通过之后，通知配置管理员，由配置管理员根据测试结果对相关配置项打标识。

二、基线

基线是软件文档或源码(或其它产出物)的一个稳定版本，它是进一步开发的基础，只有经过授权后才能变更。建立一个初始基线后，以后每次对其进行的变更都将记录为一个差值，直到建成下一个基线。

基线，由一个或若干个通过（正式）评审并得到确认的配置项组成，是项目进入下一个生命周期阶段的出发点（或基准）。项目组在完成配置项的标识之后，应按照机构标准、结合项目具体情况，定义项目基线，说明每个基线的配置项组成。注意，对某一个特定的基线而言，其配置项组成是随着项目进展而逐步增加的，并且组成基线的任何一个配置项的变更都会引起基线的变更。因此，跟踪、控制基线变更、确保基线的完整性是配置管理的一大要点。

- 建立基线的原因：
  - ✧ 重现性，及时返回并重新生成软件系统给定发布版本的能力，早些时候重新生成开发环境的能力。
  - ✧ 可追踪性，建立项目工作产品之间的前后继承关系，确保设计满足要求、代码满足设计及用正确的代码编译系统。
  - ✧ 报告，来源于基线之间内容的比较，有助于调试并生成发布说明。
- 建立基线有如下优点：
  - ✧ 为开发工作提供了一个定点和快照。

- ✧ 新项目可以从基线提供的定点建立，作为一个单独分支，新项目将与随后对原始项目所进行的变更进行隔离。
- ✧ 各开发人员可以将建有基线的工作产品作为他在隔离的私有工作区中进行更新的基础。
- ✧ 当认为更新不稳定或不可信时，基线为团队提供一种取消变更的方法。
- ✧ 重新建立基于某个特定发布版本的配置，可以重现已报告的错误。

■ 常用基线建立的时机：

- ✧ 需求基线（SRS_BL），在需求分析阶段结束后，《用户需求说明书》、《软件需求规格说明书》经过了评审。
- ✧ 计划基线（PLN_BL），详细计划经过评审。
- ✧ 设计基线（DESIN_BL），在概要设计和详细设计阶段结束后，设计阶段工作产品经过了评审。
- ✧ 实现基线（CODE_BL）：代码和集成测试计划、用例、报告等工作产品经过了评审 。
- ✧ 测试基线（TEST_BL）：系统测试计划、用例、报告等工作产品经过了评审。
- ✧ 发布基线（RELEASE_BL）：通过软件系统验收测试与正式的配置审核，产生了作为最终产品交付用户的配置项的集合。

三、工作空间

工作空间为开发人员提供独立的工作空间。工作空间是被设计用来防止用户之间的相互干扰。它提供了在配置管理下的能在可调对象上持续的工作空间。工作空间是通过版本状态模型来获得的。这就意味着属性“状态”是和构件的版本相联系的。依靠那种状态（例如状态“忙”或“冻结”），构件或者被认为是一个私有的工作区或者被认为是一个公有的库。“忙”构件是可调的并且不能被其他人所使用，像“冻结”就是一个对公共使用来说能获得的但不可调的例子。构件被提交给公共库的同时使得它们在被适当的用户证明后，对公共用途来说是可获得的。在效力上，工作区提供工作的独立性且建立在一个全局的、长期的不可调对象

的库和一个为可调对象且私有的短期的库之间的区别。

在企业里，一般对每个人的工作空间可以建立如下约定：

- 开发人员在项目结束后在本地机器删除所有项目资料；
- 严格按照开发环境的描述安装相关软件，搭建自己的工作平台；
- 及时备份半成品，在开始修改配置项之后检查当前配置项状态/版本号；
- 不随意安装未经过批准的软件。

四、变更控制

对于大型的软件开发项目，无控制的变更将迅速导致混乱，使整个项目无法顺利进行下去而失败。变更控制就是通过结合人为的规程和自动化工具，以提供一个变化控制的机制。本文所涉及的变更控制的对象主要指配置库中的各基线配置项。变更管理的一般流程是：

- 由开发人员或系统分析人员提出变更需求；
- 由 CCB（变更控制委员会）或项目经理审核并决定是否批准；
- 配置管理员根据 CCB 或项目经理 的决定开放相应的权限，并形成记录备案；
- 变更申请人员执行相应的变更。

在这里，将要涉及的变更控制分为两类：一类是基线的变更控制，另一类是软件版本的变更控制。

基线的变更控制：

基线的变更是指在一个软件版本的开发周期内对基线配置项的变更，主要包括基线的应用和更新等活动。基线变更所涉及的操作主要包括基线标签的定义和标签的使用。基线标签属于严格受控的配置项，它的命名必须严格按照相关的命名规范来进行。基线在建立时，按照角色职责的分工,须经 CCB 同意并以正式的将该基线的标识和作用范围通知系统集成员，由后者负责执行；基线一旦划定，由该基线控制的各配置项的历史版本均处于锁定或严格受控状态，任何对基线位置的变更请求都必须按变更控制流程，提交 CCB 批准，然后由系统集成员执行。

软件版本的变更控制：

软件版本的命名规范应事先制定，并按照开发计划予以发布使用。在软件版本的演进过程中既需要从以前的版本中继承，又需要相对的独立性。所以在对于一个子版本（例如某特定用户的定制版本）就需要对一系列配置项从统一的开发起始基线所确定的版本上建立新的分支，然后在此分支上开发新的版本。因此在这样的变更控制流程中，受控的对象还应包括特定的分支类型，以及工作视图的选取规则，同时配置管理员将在这一过程中担负更多的操作职责。

8.3 配置管理活动

对于软件配置管理，一般软件企业会制订一定的机构方针，这些方针是所有软件开发类项目进行配置管理时必须遵循的。以下并于配置管理机构方针是本书讲解的重点：

- 项目组软件配置管理应有专人负责（称配置管理员）；一般中小型项目，由项目经理或指定专人担任配置管理员，负责项目配置管理。大规模项目，则应建立配置管理小组（CM 组），在项目经理领导或授权下负责项目配置管理。
- 配置管理贯穿软件生命周期全过程，但分两个阶段：从需求到产品发布的开发阶段，配置管理由项目经理或指定专人负责；发布后进入产品维护阶段，由负责该产品技术支持部门指定的配置管理员负责。
- 在整个开发阶段，各类工作产品（配置项）及其变更是项目配置管理的重点；而开发环境、测试环境和运行环境的描述文档则只作为配置项纳入配置管理，受到控制。
- 在产品维护阶段，配置管理的重点则包括变更控制、版本控制和基线管理。
- 项目启动后就应开始配置管理活动，包括：定义、标识配置项，定义基线，建立配置库和基线库，确定访问权限，控制配置库/基线库的检出（Check out）和检入（Check in）。
- 在项目计划阶段，应编写配置管理计划（CM 计划），与项目开发计划一起提交评审；在产品发布后进入维护阶段，也应编写 CM 计划。
- 按评审确认的 CM 计划建立基线、审计配置库和基线，及时报告配置状态。

- 每一个产品的所有配置项的变更均应得到管理和控制。
- （每一个项目组的）软件产品最终集成（产品发布基线，或产品发布后的产品维护阶段定期生成的基线），由项目配置管理员负责实施，技术支持的配置管理员负责监督。

在软件开发过程中，配置管理活动一般包含内容有：配置管理计划制定及跟踪；版本管理；确定配置项标识规则；变更管理；发布管理；工作空间管理；报告配置状态；配置审计。

在执行配置管理活动的时候，一般可以按如图 8-2（见下页）所示流程进行。隐含在其中，比较重要的一块内容是识别配置项。配置项一般可以按如下三类进行划分：

- 代码类配置项（源代码、可执行代码以及相关的数据文件）的划分由项目组结合项目具体情况确定（例如，一个单元或模块的代码作为一个配置项），代码类配置项的命名必须结合软件产品的特征，而版本编号则应符合机构统一规定。
- 开发环境、测试环境和运行环境描述，单独成文，并作为单独的配置项进行管理。
- 文档类配置项，比如：需求类文档、计划类文档、设计类文档、测试用例/方案文档、测试报告、用户手岫等。实际执行时项目组应遵照机构标准并结合项目具体情况加以适当裁剪。

但是如下几点可以做为配置项识别时的参考标准：

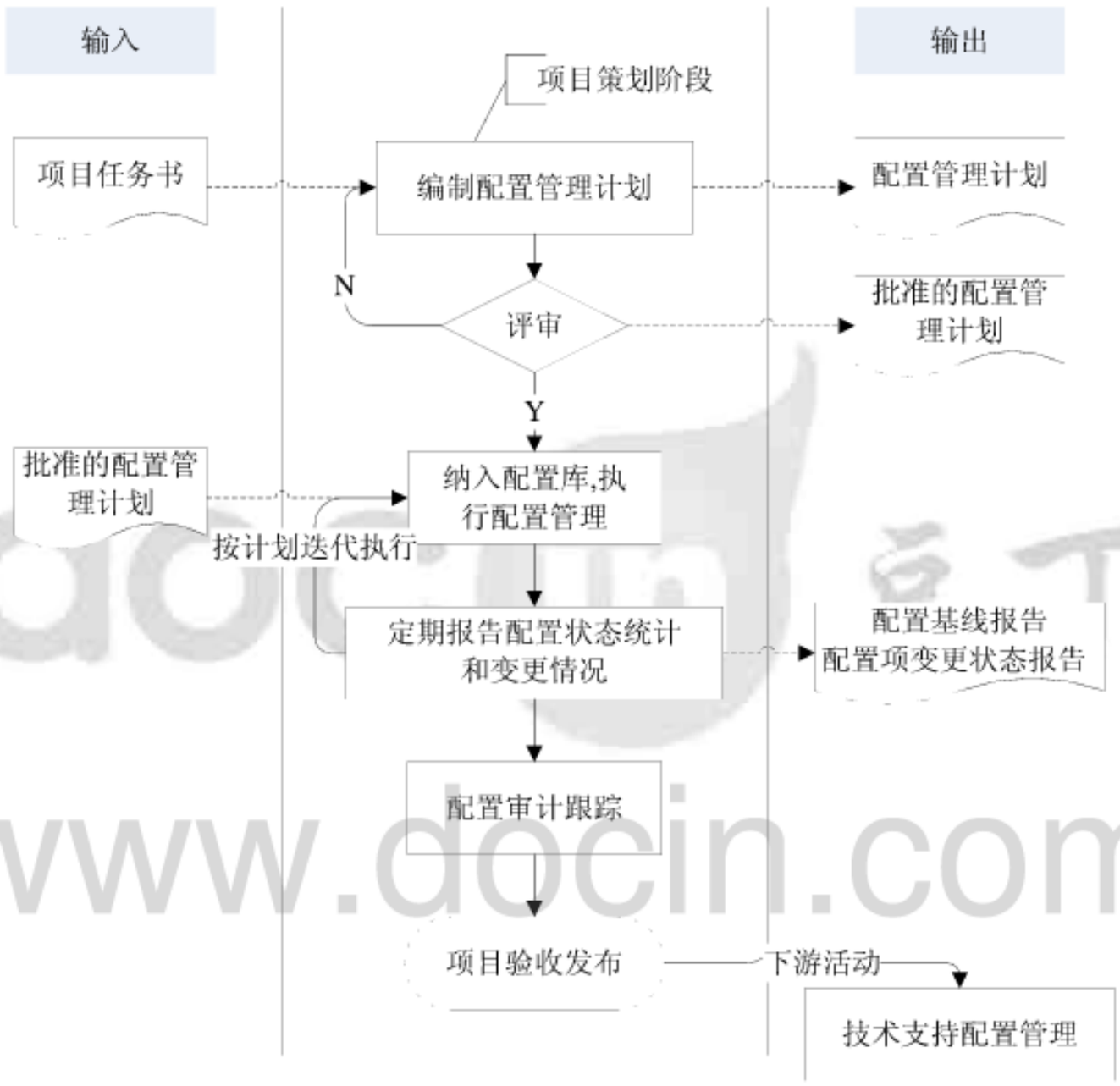
- 可能被两个或两个以上组使用的工作产品；
- 无论是因为错误还是因为需求变更而导致变更的工作产品；
- 工作产品相关依赖，其中一个变更会导致另外一个变更；
- 对项目非常重要的工作产品（环境类文档应当属于这一类）。

在标识配置项的同时，应给每一个配置项确定一个唯一的标识符（即名称），并且为了更好地管理配置项的变更，除了标识符外还应标示版本号。因此，机构应制定配置项标识命名规则，以便统一管理，规则应包含如下内容：

- 文档类配置项命名规则，
- 文档版本编号规则，

- 代码类配置项命名规则，
- 单元（模块）源代码和执行码版本编号规则。

图表 8-2 配置管理活动流程图



8.3.1 编制配置管理计划

配置管理计划的形成时间：项目计划初步阶段形成草稿，项目计划细化阶段定稿，并与项目开发计划一起评审。首先根据《机构标准软件过程》中的列表，识别本项目中的配置项，并作为配置管理计划的一部分。在项目经理的指导下，配置管理员完成配置管理计划。

【注意：此处各配置项纳入配置库的时间应当保持与项目开发计划中此配置项形成时间一致、名称一致。】

定义项目的基线，一般分为需求基线（SRS_BL），在软件需求规格说明书批准时建立；计划基线（PLN_BL），在项目计划批准时建立；设计基线（DESIN_BL）在概要设计、详细设计和数据库设计批准时建立；编码基线（CODE_BL），在单元测试通过时为集成测试建立；测试基线（TEST_BL），在集成测试时通过为系统测试建立；产品基线（RELEASE_BL），在系统测试通过为产品发布时建立。在项目开发过程中，我们建议必须包含需求基线和产品基线，其他基线可以由项目经理根据实际情况裁剪。需要明确各个基线包含的配置项，确定各个基线的生产时间。

配置管理计划文档化：在项目计划初步阶段，必须明确与需求相关的配置项及基线生成时间，配置库结构及权限。确定职责和所需资源；确定软件项目配置项；确定基线条数、基线包含配置项、建立时间、审计人；确定要执行的活动及活动的进度安排。明确配置库目录及存取权限/方式；确定系统的开发环境、测试环境、运行环境；配置库的备份方式。

另外，配置管理计划与项目计划书一起提交评审，具体过程第 4 章 项目评审管理讲解的内容执行。并作为配置管理工作的基础，在评审通过后变为受控项。若配置管理计划需要变更，则按“变更控制”执行。

为了更好的标识配置项的版本，在《配置管理计划》中可以包含版本号的定义规则。我们可以按如下方式来定义版本号：

软件版本号可定义为：分四节，<X>.<Y>.<Z>.[<R>]；

- <X>一位整数，代表主发布版本号，一般从 1 开始编号；
- <Y>一位整数，代表次版本号，一般从 1 开始编号；
- <Z>两位整数，一般从 0 开始编号，代表次发布版本号；
- <R>三位整数（可选），一般从 0 开始依次递增，代表备选发布版本号，如 2.1.5.001 表示 2.1.5 版的第 1 号补丁。

以上各版本号不足时补零，整个系统版本号在发布时另外按该规则标识与开发过程中配置项版本号无关。

版本号的前两节是在项目立项时确定，在整个项目过程中不能修改，后两节在项目中根

据进度进行增长；若配置项在第一次评审通过之后，则<Z>处应当打 1，评审之前打 0 。

其他工作产品（比如各类文档等）版本号定义：分为三节：V<X>.<Y>.<Z>；

- <X>一位整数，代表主版本号，一般从 1 开始编号；
- <Y>一位整数，代表次版本号，一般从 1 开始编号；
- <Z>两位整数，一般从 0 开始编号，代表修改次数；

举例说明怎么来使用该规则为一个：比如编写一个《软件需求规格说明书》，在第一次形成该文档时版本编号为 V0.1.0，每次修改第三位加 1，在第 10 次修改时，变成 V0.2.0，以此类推，直到定稿提交评审（可能为正式评审也有可能为非正式评审）时，版本改为 V1.0.0，评审通过之后批准的版本改为 V1.0.1。以后再修改该文档，第三位以照前面方式加 1 进行标识。若在修改之后，纳入到新的基线，则主版本号改为 V2.0.0，其他次要修改，则不需要更改主版本号。

8.3.2 配置管理审计

配置管理的审计活动一般分为两类，一是对基线的审计，二是对配置库的审计。基线审计是为了检查基线的正确性及一致性；配置库审计是为了保证配置库的完整性、可用性。

基线审计一般按下面步骤进行：

- 项目经理在基线生成之前填写《基线计划及跟踪表》；
- 由指定专人（一般为质量保证工程师、机构级的 CMG 组长或资深工程师，在配置管理计划中明确）根据基线计划及跟踪表对配置库进行审计；
- 审计出的问题修改之后，由 CCB 批准后，配置管理员生成基线，并打基线标识。

配置库的审计一般由配置管理员完成，配置库审计的时机及内容如下：

- 在里程碑处或基线生成之后进行；
- 由配置管理员或项目经理指定负责人对配置库进行审计，填写配置审计报告；
- 主要内容包括：配置库结构是否正确，是否能正常签入签出；基线库的建立手续是否齐全；配置项版本历史信息是否正确；

- 质量保证工程师根据相关规程对配置管理过程进行审计，填写《QA 阶段审计报告》中的“QA 配置管理过程审计报告”，在确保配置管理活动按照要求开展。

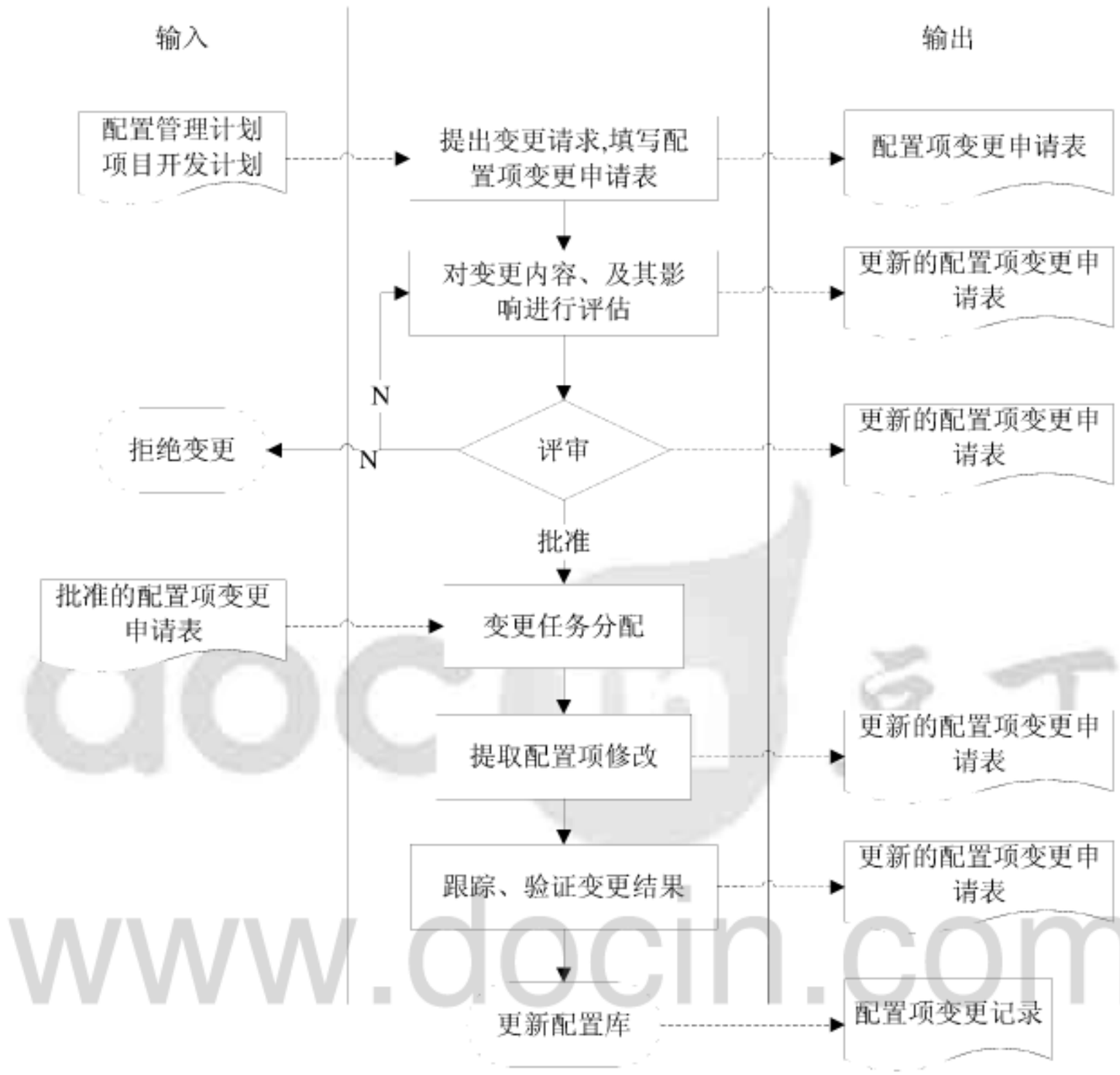
8.3.3 变更控制简述

变更控制做为配置管理的主要内容之一，在操作过程中有严格的控制流程，以保证配置项的一致性、有效性。一般变更控制的内容为：

- 确定变更批准人的责任范围和权限；
- 建立变更控制流程，实施变更控制；
- 对配置项变更进行管理；
- 对基线变更进行管理；
- 设立两个变更授权机构：CCB、项目经理；
- CCB 成员为项目级的，可因项目的不同而有所不同，由总工程师在《项目任务书》中定义。

变更控制活动遵循的流程为：提出变更申请（填写《配置项变更申请表》）——评估变更影响范围（更新《配置项变更申请表》）——评审变更申请（更新《配置项变更申请表》——执行变更任务（更新《配置项变更申请表》——验证变更结果（更新《配置项变更申请表》）——更新配置库，其中根据配置项的类型及变更影响的范围来确定是由项目经理审批变更，还是有 CCB 来评审变更。具体操作流程如图 8-3（见下页）所示，详细的操作步骤及方法在后续章节中讲解。

图表 8-3 变更控制流程图



【说明：本流程图描述了纳入基线库后的配置项变更控制，对于普通配置项的变更，由项目经理自行批准，可以不填写变更控制表，不适用于本流程。】

在项目计划阶段就需要针对变更控制制定相关的计划，比如：所需资源、控制流程等。变更控制需要做的工作：

- 在项目立项时，根据项目规模和特点，确定变更授权机构及其职责，并纳入立项报告及计划阶段的配置管理计划；
- 确定变更等级：变更等级一般由项目经理判断，并在配置管理计划中描述各自控制的变更，建议若是影响需求基线和产品基线的变更以及严重影响项目进度、成本、

产品质量的重大变更提交 CCB 控制；其他变更（如：文字编辑、格式调整）由项目经理控制。

在进行变更控制时，一般会有如下要求：

- 对评审定稿配置项（简称受控项，代码类指通过集成测试之后）和基线（有重大缺陷）的所有变更在实施前均要通过变更授权机构的评审和批准；变更过程必须记录在《配置项变更申请表》并关闭掉。
- 变更控制流程适用于开发过程中所有配置项变更，非以上提到的项可不填写《配置项变更申请表》，但需在修订页中说明。
- 对于受控项，不论是项目经理还是 CCB 控制变更，其提请变更的流程相同，配置管理员只负责受控项标识更新和配置项变更状态报告的填写，不参与其它活动；所有《配置项变更申请表》由项目经理负责提交配置管理员纳入受控库；配置管理员提交配置项变更记录给相关受影响人员。

8.3.4 变更控制活动

在项目开发过程中，遇到需要变更的工作产品（比如：需求变更、设计变更等等）时，一般可以按如下流程来操作。

1. 变更申请人填写配置项变更申请表说明问题来源或修改原因；变更对其他配置项的影响，估计变更对项目造成的影响等。对于代码类变更，可以记录在 BUG 管理工具里，而不填写专门的配置项变更申请表，但在项目经理分配 BUG 时，必须分析变更所需花费的工时、工作量、成本及变更带来的风险，并填写在 BUG 管理工具中。如果代码类变更，对里程碑有影响，则必须填写配置项变更申请表。
2. 项目经理收到变更申请后，评估变更带来的影响、分析变更所需花费的工时、工作量、成本及变更带来的风险等，并将评估结果应写入“审批意见”栏；然后提交变更授权机构（比如，CCB），若是不需要通过 CCB 的变更申请，则项目经理签署意见之后，即可执行变更。

3. 变更控制人判断变更的大小采取合适的评审方式：签字或评审。若采取签字方式，变更控制人在变更控制栏填写审核意见，若采取评审方式，遵照评审规程执行；然后顺次执行以下步骤；
4. 如果变更被拒绝申请，项目经理通知变更申请人，由项目经理提交配置管理员入库，变更结束；
5. 如果变更被批准，项目经理负责通知受影响的人员更改相关配置项，并指定项目组成员实施变更。
6. 修改人根据被批准的配置项变更申请表，根据标识规则从开发区里 Check out 配置项实施变更；修改完后 Check in 并进行标识，在配置项变更申请表中进行变更描述，必要时可用附件。
7. 文档类对象，由验证人验证修改结果并更新配置项变更申请表的状态（已更改），由配置管理员更新配置项变更状态报告，并在开发区处更新配置项标识；基线变更，由项目经理填写版本控制表，审计人员审计通过并 CCB 签字批准，交配置管理员生成基线；
8. 变更实施且被质量保证工程师验证签字后，由项目经理抄送相关人员（包括研发部经理、测试人员、文档人员、配置管理员、质量保证工程师等）并将配置项变更表交给配置管理员纳入 CM 库，同进更新《配置项计划表》中配置项状态，填写《配置项计划表中》配置项变更记录。

变更产生的所有相关文档都纳入配置管理范畴；配置项变更申请表可以是电子表格或纸质文档，形式不限；将受控项变更处理结果汇总在《配置项计划表》的配置项变更记录中。

8.3.5 产品构造

产品构造一般应在集成测试、系统测试前，及产品交付客户前进行；对于一些小的项目，根据项目具体情况，也可考虑只构造一次，即产品交付前。

产品构造还需遵守如下原则：

- 在构造产品之前，需要制定集成计划。
- CCB 审定软件受控区构造的产品的生成。
- 不论为内部或外部使用，有软件受控区构造的产品仅仅由软件受控区中的配置项和单元组成。

项目经理根据整个项目进展情况，制定每次构造的集成计划。在集成之前一天（具体时间长短不同公司或项目有不同的要求及规定）把集成计划提交给相关人员，相关开发人员必须把集成计划中列出的配置项按时提交并标识；若不能按时提交，则及时上报项目经理。一般产品构造的步骤为：

1. 构造人员在本地机器或者其他目标计算机上为产品建立一个目录。若目录原来存在，则需要把目录清空。
2. 配置管理员将软件产品需要的配置项从配置管理库上的开发区中复制到这个路径下，然后对软件产品 Build。
3. 配置管理员把集成的结果填写在集成计划中，然后提交给项目经理。
4. 测试工程师从指定的位置得到构造后的产品进行测试，并把测试出的问题记录到 BUG 管理工具中；若测试通过，则通知配置管理员，由配置管理员根据集成计划中的配置项列表，按照标识规则改变配置项的标识；
5. 如果软件产品需要修改，则从开发区把配置项按标识规则打上标识后，检出(check out)到目标计算机上，在相关人员修改好后检入(check in)，并按标识规则打上标识；重复以上步骤，直至无错误；
6. 若为产品发布构造，则需要把提交给客户的软件产品应该拷贝到光盘、硬盘等介质上。

8.3.6 配置管理的管理活动

一般 CM 管理的活动有两种：

1. 跟踪配置管理活动：

- ✧ 项目经理根据项目实际规模来确定配置库的备份策略，包括确定配置库备份的频率及备份方式、路径等，并对这些策略以文档化的方式写进配置管理计划；配置管理员根据配置管理计划，对配置库进行备份，并对备份操作形成记录。
- ✧ 配置管理员在工作周报中汇报每周配置管理的工作情况，提交 CMG 组长、项目经理及相关组或个人。
- ✧ 配置管理员定期或事件驱动，负责配置管理状况（基线跟踪表、配置审计报告、配置管理问题清单、配置项变更记录、产品发布清单等）相关报告的编写，并报告给 CMG 组长、质量保证工程师、总工程师/研发部经理、项目经理及相关组或个人。

2. 验证配置管理活动：

项目的质量保证工程师负责依据软件质量保证过程和项目的质量保证计划验证配置管理活动的执行符合配置管理计划和本过程。

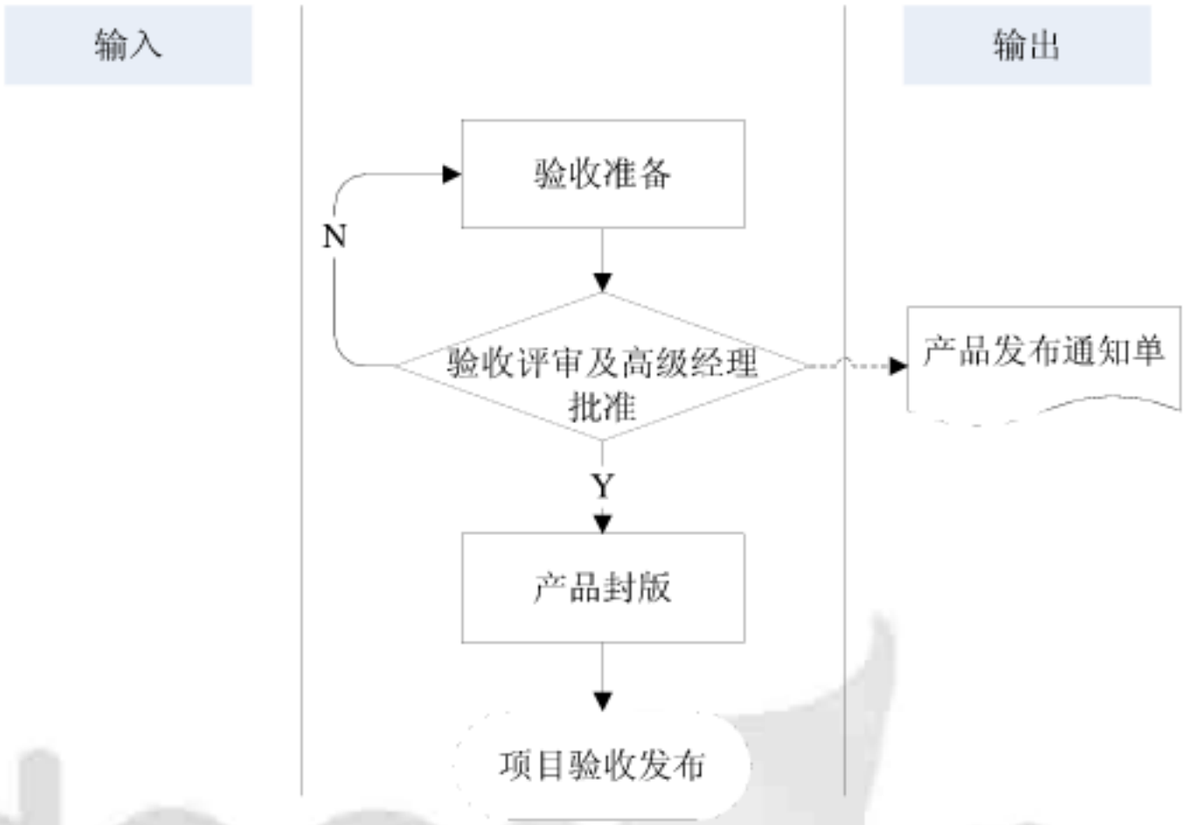
8.4 产品发布流程

在开发过程中发布分为如下几类：

- ■ 产品发布，产品的对外发布，整个项目结项；
- ■ 产品基线发布，产品对内发布，之后可以安装试点或进行 Beta 测试/用户测试（研发部经理/总工程师负责）；
- ■ 其他基线发布，计划基线、需求基线、设计基线、编码基线、测试基线等（项目经理负责）。

其中，产品发布的流程如下图所示：

图表 8-4 产品发布流程图



除了产品发布、产品基线发布外，其他基线发布的步骤遵循：

- 先由项目经理确认受影响的相关人员（如：项目组成员、测试人员、配置管理员、质量保证工程师）；
- 配置管理员将最新的基线报告、配置项变更报告（记录）、版本控制表定期或事件驱动发布给受影响的组和个人；
- 由项目经理确认受影响的组和个人都收到最新的基线报告、配置项变更报告（记录）、版本控制表；
- 基线发布。

产品基线的发布遵循的步骤为：

- 先由项目经理提出产品基线发布申请，由总工程师/研发部经理确认受影响的相关人员（如：项目组成员、测试人员、配置管理员、质量保证工程师、相关业务部门）；
- 配置管理员将最新的基线报告、配置项变更报告（记录）、版本控制表定期或事件驱动发布给受影响的组和个人；
- 由项目经理确认受影响的组和个人都收到最新的基线报告、配置项变更报告（记录）、版本控制表；

■ 举行产品基线发布评审，由总工程师/研发部经理主持，评审通过之后，基线发布。

相比较起来，产品发布是相当严格的活动，在产品发布前一般需要准备：发布必须得到批准；产品基线发布前已完成了验收评审；所有发布的配置项是置于配置控制下；创建了产品发布清单。

一般产品发布的具体步骤如下：

1. 发布前准备：项目经理负责将版本控制表、产品发布申请表内容填写完整，并检查以下内容：
   - ✧ 软件产品是否测试通过；
   - ✧ 配置管理数据库是否经过审计、审计发现的问题是否得到解决；
   - ✧ 检查项目评审表验收结论，是否通过验收评审；
2. 产品发布申请：将版本控制表、产品发布申请表及产品发布通知单提交研发部经理或总工程师审核签字；
3. 产品封版：
   - ✧ 责任人：配置管理员；
   - ✧ 封版内容：配置管理数据库受控区的内容。
   - ✧ 封版的实现：锁定配置管理库（LOCK），备份配置管理库、产品基线；
   - ✧ 封版标识：将封版内容刻成光盘并唯一标识（建议标识方法：部门名+系统名，如：SI-ZYTK 表示软件研发部一卡通产品）；
   - ✧ 封版媒介：光盘、硬盘等介质上；
   - ✧ 约束：封版后的产品将不得随意改动，如需改动，必须遵照变更控制流程执行；
   - ✧ 产品版本升级见标识规则；
   - ✧ 存放：封版后的光盘一式二份，配置管理员提交本部门、总工程师办公室各一份；各部门指定专人统一管理，并将相关信息记录在 CM 产品发布清单。
4. 产品发布：
   - ✧ 内部发布：由项目经理填写产品发布通知单以书面形式在所属部门发布产品；

配置管理员填写产品发布清单；

- ✧ 外部发布：各产品部配置管理员将母盘的安装目录、用户文档目录下的内容刻成光盘提交给用户并填写产品发布清单。

8.5 配置管理工具介绍

当前市面上配置工具种类繁多，还有一些开源的工具软件，下面介绍比较常见的几种配置管理工具软件。

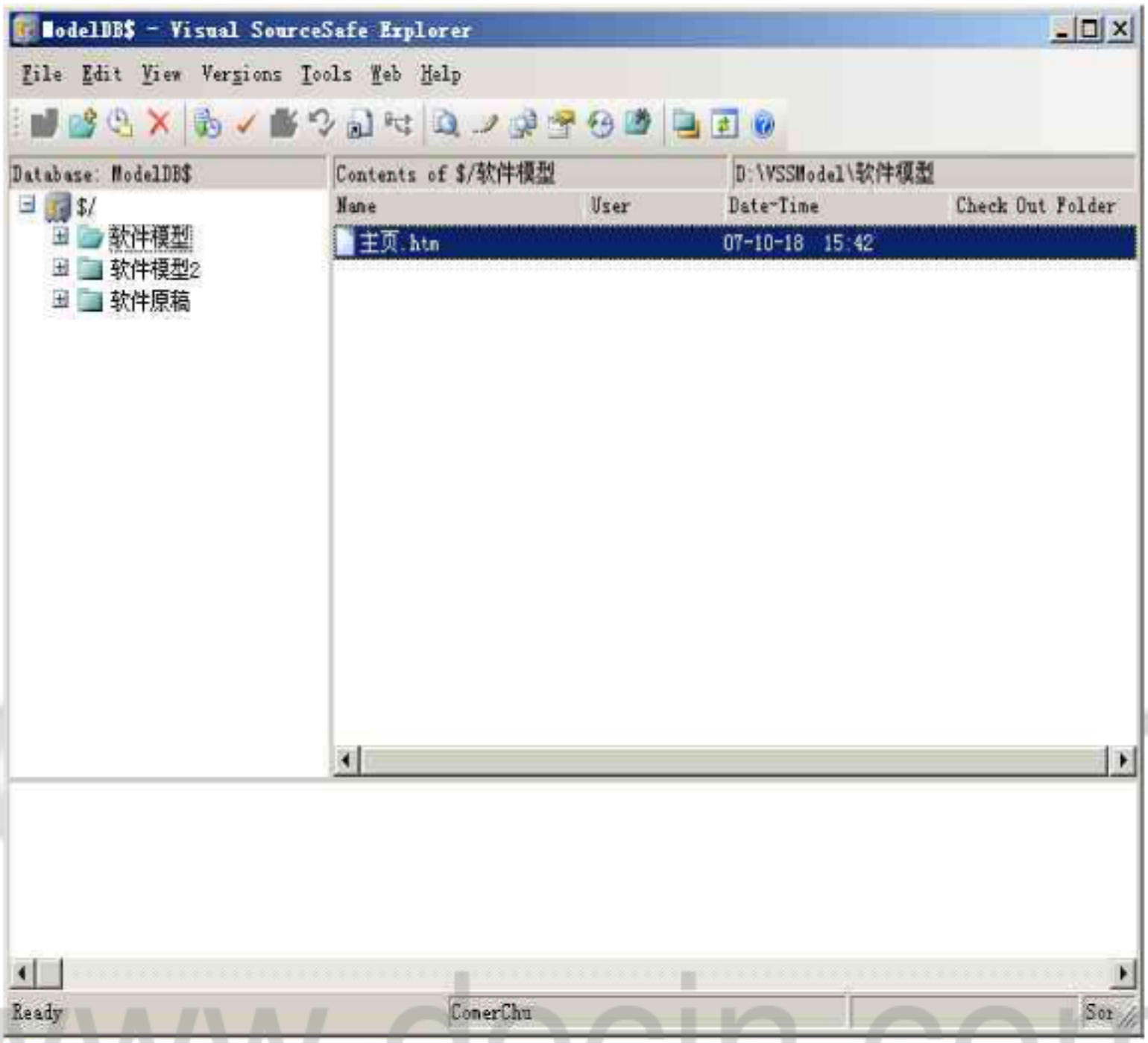
8.5.1 Visual SourceSafe

Visual SourceSafe，即 VSS，是微软公司为 Visual Studio 配套开发的一个小型的配置管理工具，准确来说，它仅能够称得上是一个小型的版本控制软件。VSS 的优点在于其与 Visual Studio 实现了无缝集成，使用简单。提供了历史版本记录、修改控制、文件比较、日志等基本功能。

缺点：只支持 Windows 平台，不支持并行开发，VSS 6.0 版本通过 Check out - Modify - Check in 的管理方式，一个时间只允许一个人修改代码（VSS 2005 支持两种模式，可以进行简单的并行开发），而且速度慢、伸缩性差，不支持异地开发（VSS 2005 可以支持通过互联网配置管理了）。甚至于微软本身也不采用其作为配置管理工具，而是使用一个名为 SLM 的内部工具。图 8-5 为该软件的一个界面截图。

当前不少原来使用 Visual Studio 开发环境的公司逐步往 VSTS+TFS（详细说明请参见实训指导）上迁移，不但更好的满足代码配置管理的需求，更能完成整个项目的全生命周期管理。

图表 8-5Visual SourceSafe2005 的界面示例



8.5.2 CVS

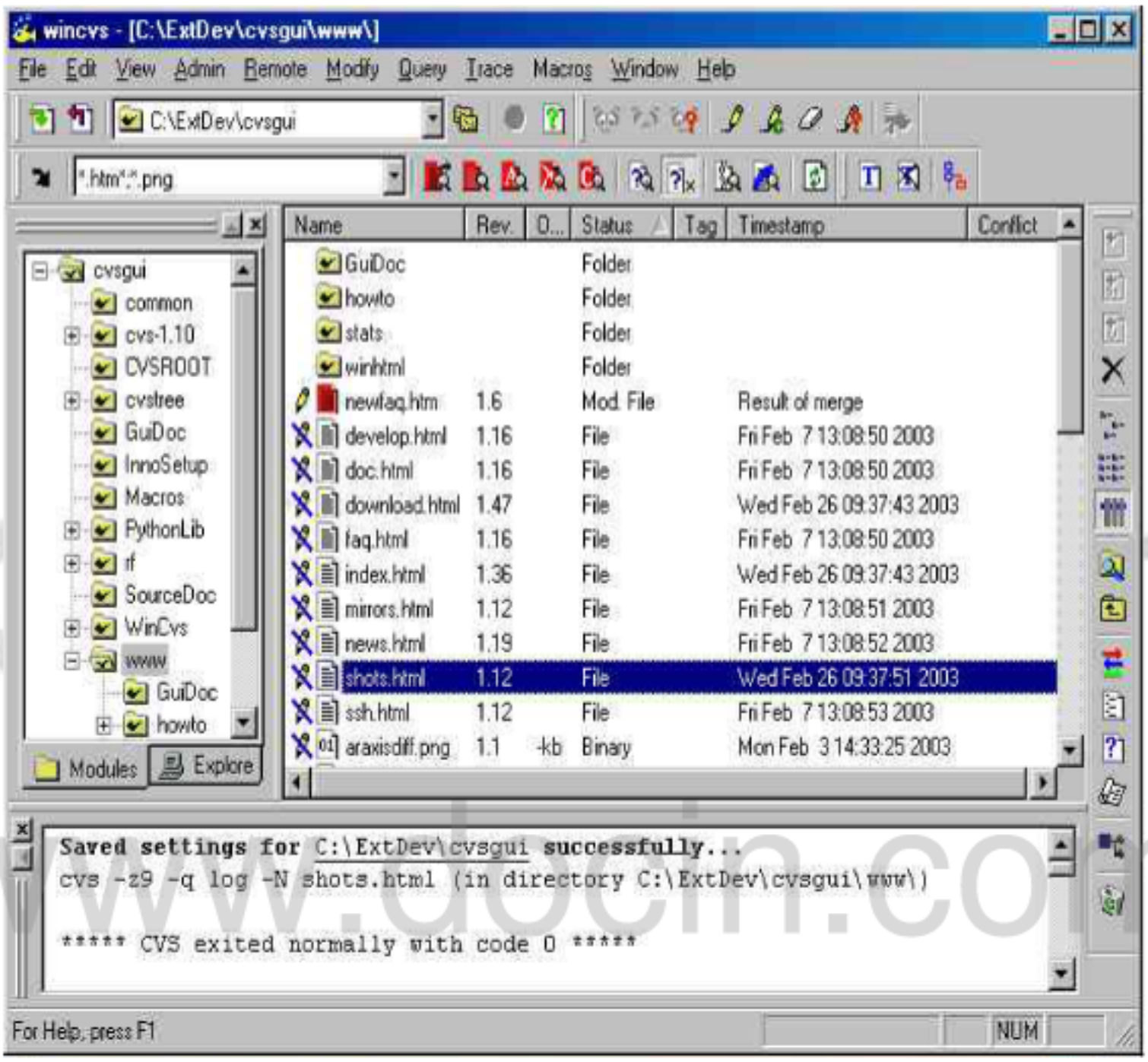
CVS 是 Concurrent Versions System（并行版本系统）的简称。它是著名的开放源代码的配置管理工具。

CVS 是一个基于 Web 的协同软件开发环境，为软件开发而设计的。其中包括：代码版本控制（CVS）、邮件列表、事件和漏洞跟踪、可定制的项目页面、基于网站的、安全的环境、公共或者个人项目、简化的项目和邮件列表管理。

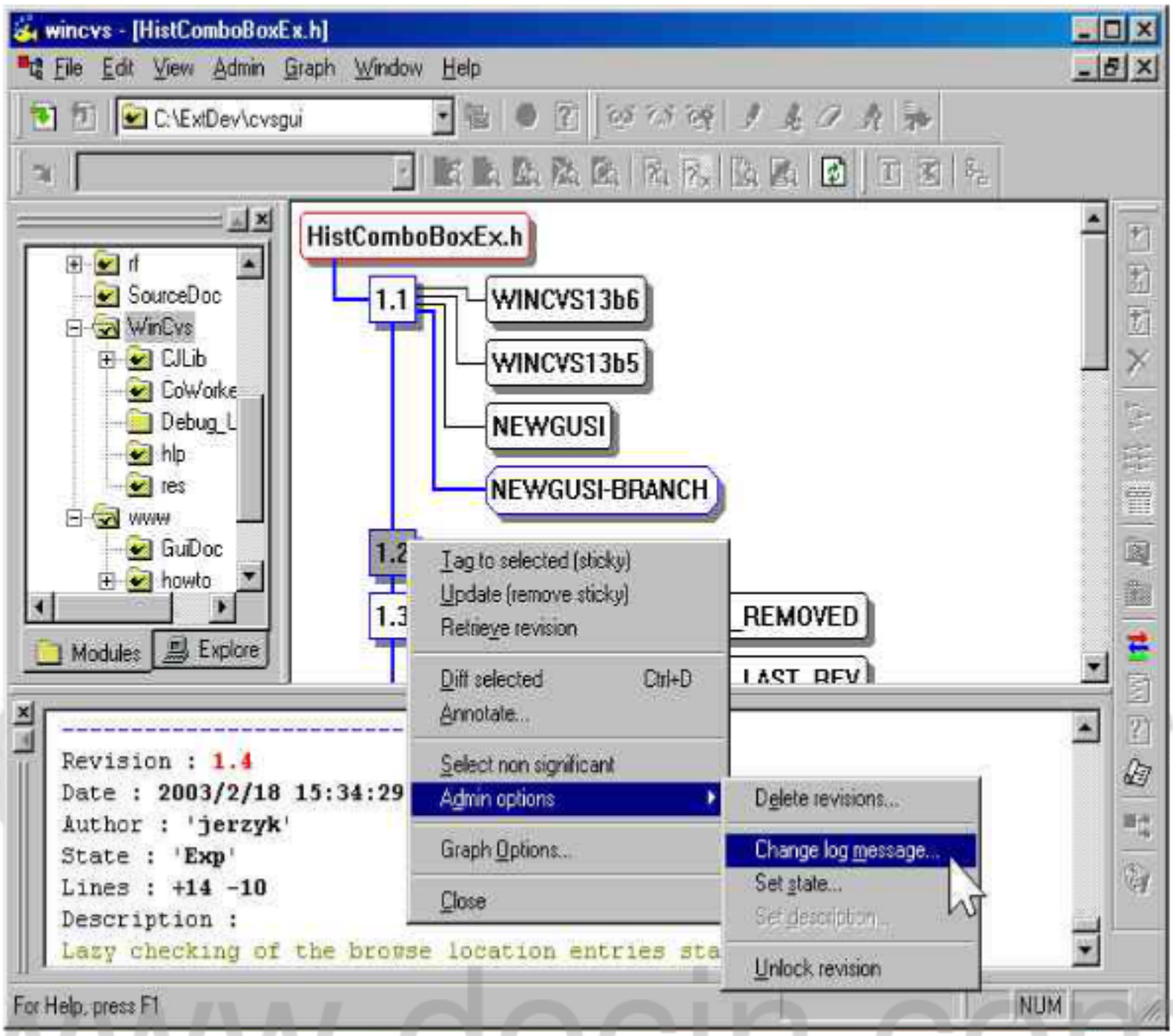
CVS 的官方网站是 http://cvshome.org/。官方提供的是 CVS 服务器和命令行程序，并不提供交互式的客户端软件。许多软件机构根据 CVS 的借口开发了各种各样的客户端软件，最

常见的当推 Windows 环境下的免费软件 WinCVS。其官方网站是 http://www.wincvs.org/，免费提供 WinCVS、MacCVS 和 gCVS 三个版本下载

图表 8-6WinCVS 主界面示例



图表 8-7WinCVS 图形化界面示例



8.5.3 Rational Clear Case

Rational Clear Case 是 IBM 公司的配置管理平台，它能自动追踪每一个文件和目录的变更情况，通过分支和归并功能支持并行开发。在软件开发环境中，Clear Case 可以对每一种对象类型（包括源代码、二进制文件、目录内容、可执行文件、文档、测试包、编译器、库文件等）实现版本控制。因而，Clear Case 提供的能力远远超出资源控制，并且可以帮助团队，在开发软件时为他们所处理的每一种信息类型建立一个安全可靠的版本历史记录。其中包括提供版本控制、工作区管理、Build 管理及流程管理几个部分。

Rational Clear Case 具有如下功能：

- 能提供版本控制、工作区管理、Build 管理及流程管理。

- 提供分布式、跨区域的并行开发模式 可以和 Visual Studio, PowerBuilder, Oracle Developer2000 集成。
- 提供离线模式，让用户可以在家工作，然后合并到开发流程中。
- 提供深入的 build 内核。
- 对执行文件和目录进行自动图形化合并，文件间的差异明显显示。
- 完整控制程序源代码、二进制码、执行码、测试项目、文档以及用户自定义的对象。
- 支持多平台，适合各种开发环境。

图表 8-8Rational ClearCase Explorer 界面示例

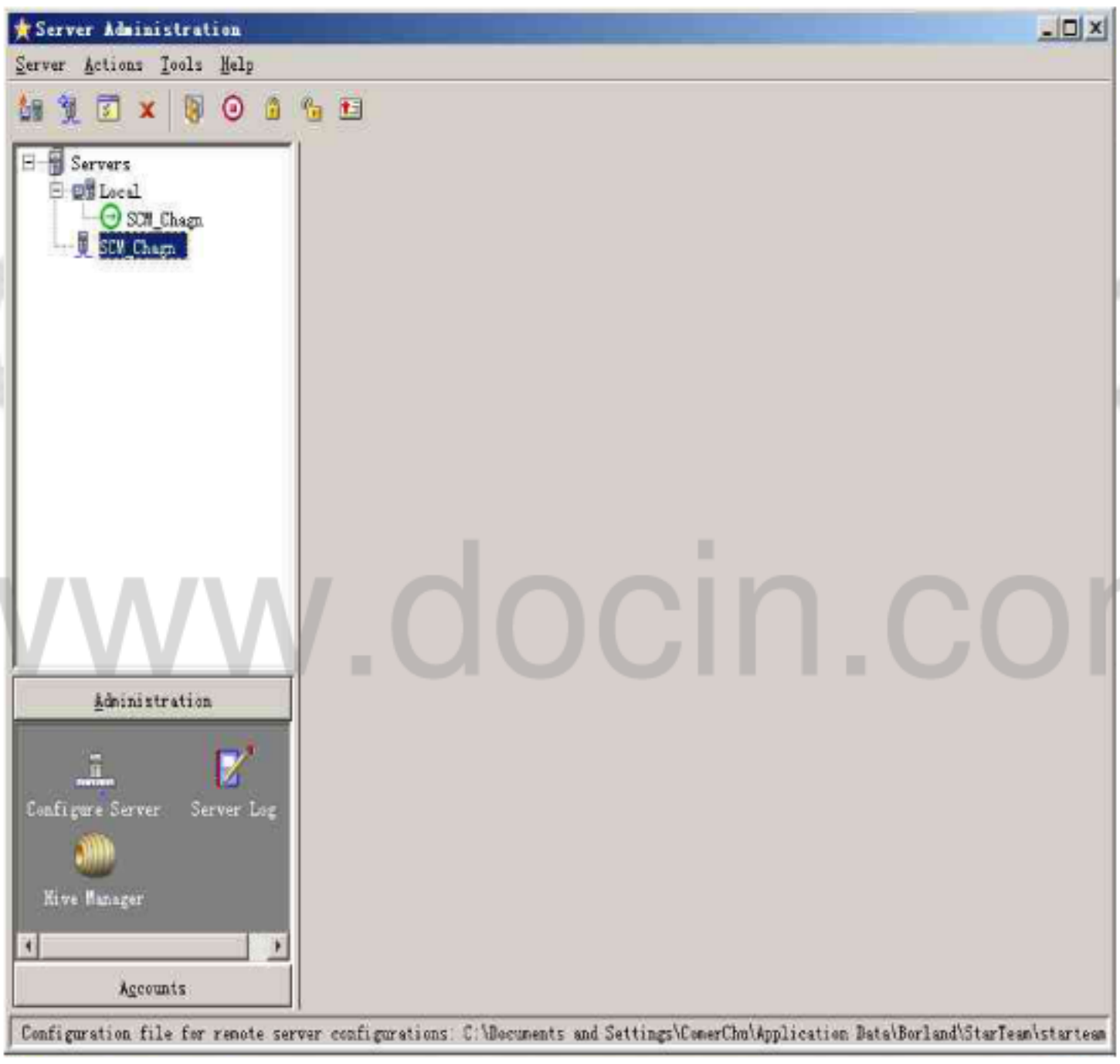
8.5.4 Star Team

Star Team 是 Borland（宝兰）公司的产品，是企业级解决方案，具备强大的综合项目管

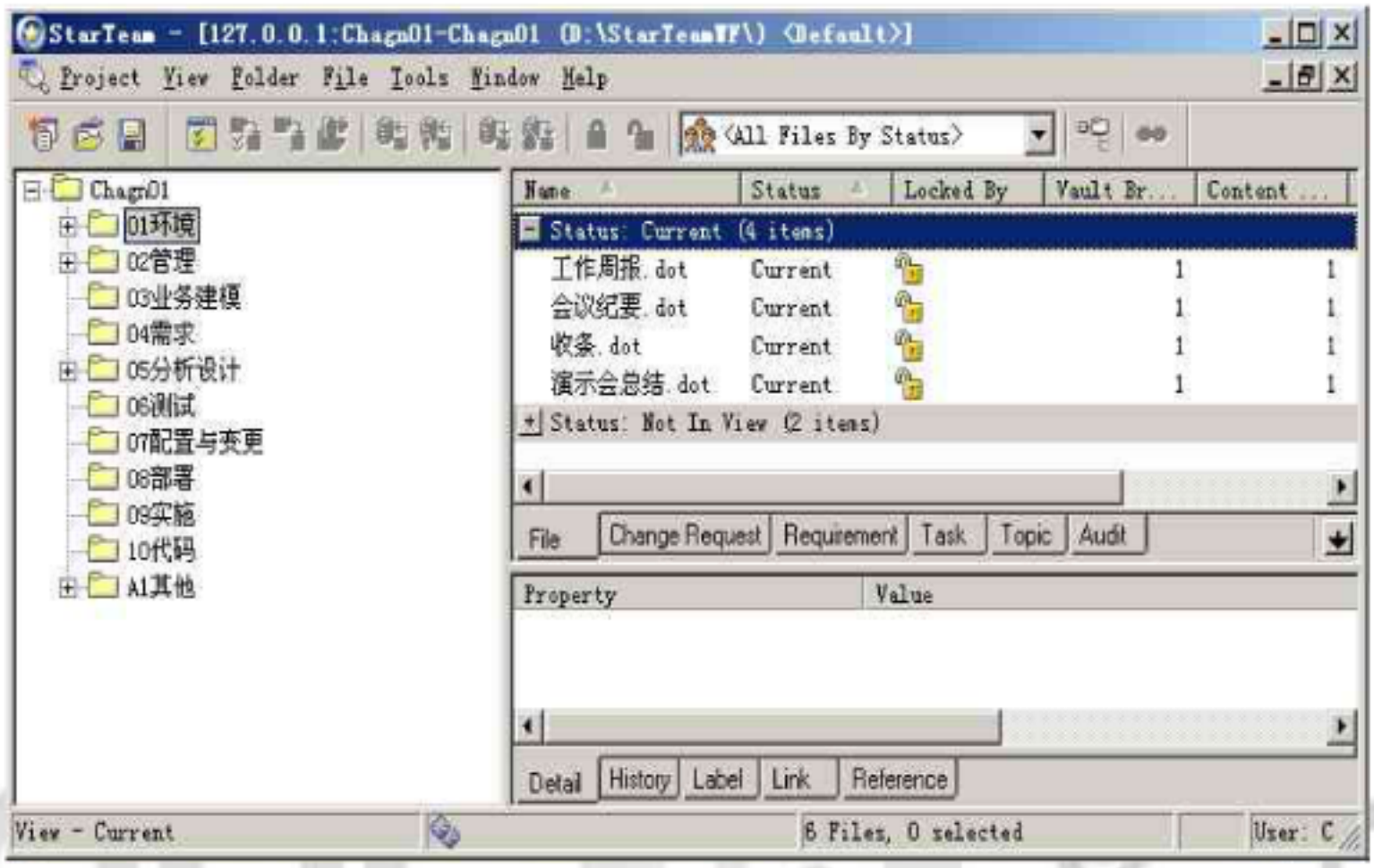
理能力，能够提供一个高度集成的环境，实现控制文件版本、管理需求和变更、追踪缺陷和过程化的讨论、对项目管理所需任务进行管理，从而提高了开发过程中对软件资产和软件问题的跟踪和管理能力。与其它同类产品相比，Star Team 还具备可定制工作流功能、良好的可扩展性、支持多种客户端接入方式等优势。

Star Team 的配置内容存放在数据库中的，可以支持 SQL Server 数据库、Oracle、DB2 等数据库。

图表 8-9Star Team Server 2005 界面示例



图表 8-10Star Team 2005 客户端界面示例



第9章 风险管理

内容提要：

- 风险基础知识
- **CMMI** 对应实践
- 风险管理概述
- 风险管理流程
- 风险跟踪

风险，作为一门科学，始于 16 世纪文艺复兴时代，风险(Risk)一词就源于古代意大利语“risicare”（敢于）。法国数学家、近代概率论的奠基人 B.Pascal 和瑞士数学家、变分学的创始人 D.Bernoulli 都曾经对风险的定义做出过重要贡献。健康的风险意识，规范的风险管理制度，会在很大程度上促进每个项目的成功。但是项目的风险管理能力需要有一个逐步提高的过程，主动的、自上而下的推动，有助于缩短这一过程。本章通过介绍风险及风险管理的基本知识，给出在软件开发项目中风险管理的流程及方法，为软件开发项目的成功提供一定的保障。

9.1 风险基础知识

风险是项目执行全过程中可能发生、一旦发生就会影响目标的实现并进而造成损失的事件或问题。其具有以下两个明显的特征：

- 不确性，事件可能发生也可能不发生，（必然发生的事件应列入项目的约束条件）；
- 损失，事件一旦发生，就会造成（成本、进度和质量等方面的）损失甚至出现严重

的恶性后果。

对于风险，还应认识到：风险与机会共存；收益和风险相伴。审时度势，权衡取舍；敢于进取，有备无患。所以作为一个成熟的项目管理者不能：

- 将风险管理看作是项目工作以外的额外活动。
- 将风险管理看作是本身职责范围以外、由他人负责的活动。

一个成功的项目管理者，应将注意力集中在项目成功的关键因素上；类似地，一个公司要成功地管理风险，应将注意力集中在不断提高风险管理能力的关键因素上。构成风险管理能力的 4 大关键因素：

- 人（People），要点：
  - ✧ 风险管理涉及所有层次的人员，各有各的职责；
  - ✧ 学习、培训和经验有助于提高个人风险管理能力；
  - ✧ 机构应采取足够的激励措施，克服风险管理的障碍；
  - ✧ 了解个人的风险偏好（Preference），用以预测个人行为。
- 过程（Process），要点：
  - ✧ 风险管理过程可划分为二个子过程、5 个过程元素：风险评估（含风险识别、风险分析）；风险控制（含风险策划、风险跟踪、风险应对）；
  - ✧ 过程可以伸缩，以便适应不同类型和不同规模的项目；
  - ✧ 过程应有必要的灵活性；
  - ✧ 过程执行应讲究成本效益。
- 基础设施（Infrastructure），要点：
  - ✧ 决策者的价值观通过机构方针影响或约束人们的行为；
  - ✧ 客户和部门管理者通过需求为项目设定预期目标；
  - ✧ 运用资源应对风险；
  - ✧ 分析风险管理的成本和收益。
- 实施（Implementation），要点：

- ✧ 制定一个主动的、高质量的风险管理计划；
- ✧ 选择符合项目特性的方法和工具。

风险管理能力需要有一个逐步提高的过程。有人提出了“风险管理路线图[9]（Risk Management Map）”的概念，揭示了逐步提高风险管理能力的途径。风险管理路线图分为如下五步：

- ■ 路线图第一步：业务分析
  - ✧ 信息获取
  - ✧ 限制条件
  - ✧ 分析工具
  - ✧ 报告汇总
- ■ 路线图第二步：风险识别
  - ✧ 风险分析基础
  - ✧ 潜在限制条件
  - ✧ 风险识别工具
  - ✧ 风险识别报告
- ■ 路线图第三步：风险评估和衡量
  - ✧ 风险评估对象
  - ✧ 风险评估障碍
  - ✧ 风险评估工具
  - ✧ 风险评估报告
- ■ 路线图第四步：风险规划和应对
  - ✧ 应对的风险事项
  - ✧ 风险应对的障碍
  - ✧ 风险应对方法

[9] 详见《整合进行时——企业全面风险管理路线图》，华小宁、梁文昭，陈昊编著，复旦大学出版社。

- ✧ 风险应对策略

■ 路线图第五步：风险管理体系的全面实施

- ✧ 风险管理体系的基本要素
- ✧ 设计/实施风险管理能力
- ✧ 企业全面风险管理体系的启动和监控

9.2 CMMI 对应实践

在 CMMI 中有一个专门的过程域对应于风险管理，即风险管理（Risk Management，简称 RSKM）过程域，其目的是，在问题发生之前识别潜在的问题，以便策划风险处理活动，在项目或产品生命周期全过程中一旦需要就可启动风险处理活动以缓解对目标实现的不利影响。风险管理是一个连续的、预测未来的过程，是事务和技术管理过程的一个重要部分。风险管理应该解决那些可能危及关键目标实现的问题。应采用持续的风险管理方法，以便有效地预先采取措施缓解对项目会产生严重影响的风险。

尽管技术问题在项目早期或所有阶段都是主要关注的问题，但风险管理必须考虑来自内部和外部的引起成本、进度和技术风险的原因。尽早主动发现风险非常重要，因为风险发现越早，变更或纠正的成本越低，对项目的损害越小。

在 CMMI 中风险管理分为三个部分：定义风险管理策略；识别和分析风险；处理风险，包括必要时实施风险缓解计划。在风险管理过程域中，共有三个特定目标需要实现，对子对应的有 7 个实践，具体为：

SG1 Prepare for Risk Management（准备风险管理），为实施风险管理做好准备工作，通常是形成一个风险管理计划文档，为整个风险管理提供一个战略性的文件。

SP1.1 Determine Risk Sources and Categories（确定风险来源和类别），通常是通过风险源清单和风险类别清单来完成该工作。常见的风险源包括：未确定的需求，不可预知的投入——估算不到位，不切实际的设计，不可得到的技术，不现实的进度估算和安排，人员或技术不足，成本或资金问题，子承包商能力的不确定或不足，买主能力不确定或不

足，与现实的或潜在的客户或客户代表沟通不充分，打断操作的持续性。

SP1.2 Define Risk Parameters（定义风险参数），目的是定义用于分析和分类风险参数，以及用于控制风险管理活动的参数。一般来说，我们需要定义如下三类内容。一是定义评价和量化风险概率和严重程度的一致准则；二是定义每个风险类别的阈值；三是定义阈值可扩展的边界。

SP1.3 Establish a Risk Management Strategy（建立风险管理策略），目的是建立和维护用于风险管理的策略，此策略一般是在公司级的文档里或项目的风险管理计划文档里明确提出。

SG2 Identify and Analyze Risks（识别和分析风险），识别和分析风险，以便决定其相关的重要性。

SP2.1 Identify Risks（识别风险），识别并记录风险。一般需要完成如下几个活动，一是识别成本、进度和性能在产品生命周期各个阶段的风险；二是审查可能影响项目的环境因素；三是审查工作分解结构（WBS）的所有元素，以确保考虑到工作量的所有方面；四是审查项目计划的所有元素，以确保考虑到项目的所有方面；五是将风险的上下文环境、条件以及潜在的后果形成文档；六是识别与每个风险相关的干系人。

SP2.2 Evaluate, Categorize, and Prioritize Risks（对风险进行评审、分类和排序），目的是使用已定义的风险类别和参数，评价和分类每个已识别的风险，并对其进行排序。一般需要完成如下几个活动，一是使用已定义的风险参数，评价已识别的风险；二是按照已定义的风险类别，对风险进行分类和分组；三是确定每个风险的排序，以便于缓解风险影响。

SG3 Mitigate Risks（缓解风险），目的是必要时处理和缓解风险，以减少对目标实现的负面影响。

SP3.1 Develop Risk Mitigation Plans（开发风险缓解计划），目的是按照风险管理策略，开发重要风险的风险缓解计划。一般需要完成如下几个活动：一是定义当风险不可接受时启动风险缓解计划或应急计划的风险等级和阈值；二是确定负责处理每个风险的责任人

或组；三是确定执行每个风险的风险缓解计划的成本——收益比；四是开发项目的整体风险缓解计划，用以权衡单个风险缓解计划和应急计划的实施；五是开发关键风险的应急计划。

SP3.2 Implement Risk Mitigation Plans（实施风险缓解计划），目的是定期监督每个风险的状态，必要时实施风险缓解计划。一般需要完成如下几个活动：一是定期监督风险状态；二是提供跟踪风险处理措施直到关闭的方法；三是当被监控风险超出已定义的阈值时，调用所选的风险处理选项；四是建立每个风险处理活动的性能周期和进度，其中包含开始时间和预期结束时间；五是对每个计划提供持续的资源承诺，以实现风险处理活动的成功执行；六是收集有关风险处理活动的性能度量。

【说明：阀值是专业技术人员根据经验或者是以往项目的参考数据制定的，用来衡量项目实际进展与计划所产生的偏差率，以此监控和指导项目的开发。例如在监控周期中的进度或工作量的偏差率超出制定的阀值时，就给此阶段红色报警（体现在偏差率上）。每个资源模型定义一个或多个阈值。阈值是资源的具有缺省值的具名属性，您可以在定制阶段修改它。通常，为阈值指定的值代表与性能有关的实体的一个重要参考标准，如果超过或未达到该值，管理员可能需要了解其情况。然而有些阈值用作参考值，以限制资源模型的作用域。】

9.3 风险管理概述

在公司里无论是软件产品的开发，还是公司日常运营管理，均应当重视风险的管理，建立行之有效的风险管理及内控体系，有助于在市场经济中取得长远的发展。通常风险管理的目的有如下两点。

- 规范公司风险管理过程，有效地进行项目风险的识别、制定管理策略并进行跟踪控制工作，确保项目顺利完成。
- 主要内容包括：风险识别及分析；制定风险应对策略；风险跟踪及控制；作为项目计划的一部分或者单独编写风险管理计划，并经评审和控制。

应该承认，确实存在许多反对（或不）进行风险管理的理由，似是而非，简单的否定并非易事也于事无补。其实，所有风险中最大的风险可能就是由于传统文化的原因人们对风险

管理缺乏应有的重视。只要看一下以下几种日常生活中普遍存在的现象（风险管理的障碍），就不难理解了：

- 不要成为有消极想法的人。
- 不要做拆台的人。
- 不要只提出问题，除非你有办法解决它。
- 不要说可能有问题，除非你能证明它。
- 不要指出问题，除非你愿意负责解决它。

此外，不妨罗列几条好像就是发生在我们身边的反对进行风险管理的理由：

- 顾客还没有成熟到坦然面对风险的地步。因此，向用户作出的有关进度、成本、报价以及质量等等的承诺只能投其所好，否则用户就不会给我们机会，或者别的竞争者抢走了机会。
- “成功管理”是最好的管理方法。上一次就是这样做的，也没有发生风险，并且成功了，没有必要搞什么风险管理。
- 是有几个项目出了问题，有损失，但公司还在向前发展，大不了赔点钱、余款收不回来，至多诉诸法律，等等。每个项目都要认真考虑风险，搞得什么事都不敢做。
- 如果风险不发生（这是完全可能的），绩效就是我的，如果不幸真的发生了，损失则是公司的。用户有什么要求都答应，合同签下来才是关键。
- 原来就这么做的，如果要控制风险，可能就会暴露已经存在的问题，决策者的面子往哪里摆，还不如照老样子做下去，说不定下半年就会发生奇迹。
- 风险管理承认不确定性的存在，而不确定性会成为低效率的理由。不承认不确定性，承诺一个确定的目标，反而会激发员工的效率。
- 每个部门都有存在的理由，每个项目都有可能成功，因此，一个部门也不能少，每个项目都要做。【结果是：在低收益的项目上浪费了公司资源，付出了高收益项目的机会成本。】

上列这些现象，有技术方面的原因，也有过程方面的原因，还有机制或制度方面的原因，

当然还有文化方面以至个人性格方面的深层次原因。但是，不管是什么原因，应该看到：技术或过程方面的改进可能只需要几个星期，机制或制度的变化则需要几个月，而文化的变化则需要一年、二年甚至更长的时间。“焦头烂额座上客”的典故古代早就有了，距今已有数百、上千年了，至今仍然有很强的针对性而值得人们的思索。可见改变一种文化该是多么困难的事情。因此，要提高风险管理能力，首先就应从决策层开始，高度重视风险意识的培养，注重风险价值观的建设，特别要注意以下几点：

- 主动进行风险管理；
- 明确风险责任；
- 不因风险而责难普通员工；
- 与相关人员交流风险，使风险应对责任人了解风险；
- 从意外结果中吸取教训。

同时，为了确保企业战略目标和年度计划的实现，应该制定并不断完善风险管理机构方针，用以约束全公司各个层次人员的风险行为。例如，应明文规定：

- 公司、部门一级的年度计划，必需包含风险管理计划；每个项目的开发计划必需包括风险管理计划。
- 在制定年度计划或项目开发计划的同时，必需进行风险识别、风险分析以及风险策划，针对首要风险明确风险责任，确定风险应对策略，制定风险应对行动计划。
- 公司和部门均应建立风险跟踪制度，跟踪风险和风险管理计划，定期（或事件驱动）验证风险管理活动相对于风险管理计划的符合性。
- 在每周或每月的例会上应报告风险。
- 在每月或每季的里程碑会议上应评审风险。
- 每个项目组、每个部门以至全公司，应重视经验积累和共享，建立并维护风险数据库。
- 各级管理者不得以风险识别的结果评价员工个人的表现。

在软件开发项目中，对于风队管理除了遵循以上列举的公司级风险管理的方针外，还需

要制定专门针对项目级风险管理的方针，具体如下：

- 每个项目指定一个风险管理人员（一般为项目经理），并制定风险管理计划（可以为项目开发计划的一部分）。
- 项目风险管理人员职责在风险管理计划中被明确分配。
- 在整个生命周期中按计划执行风险管理活动。
- 建立相应的配置管理库存储相关的风险记录。
- QA 经理、项目经理、质量保证工程师定期审计风险管理活动。

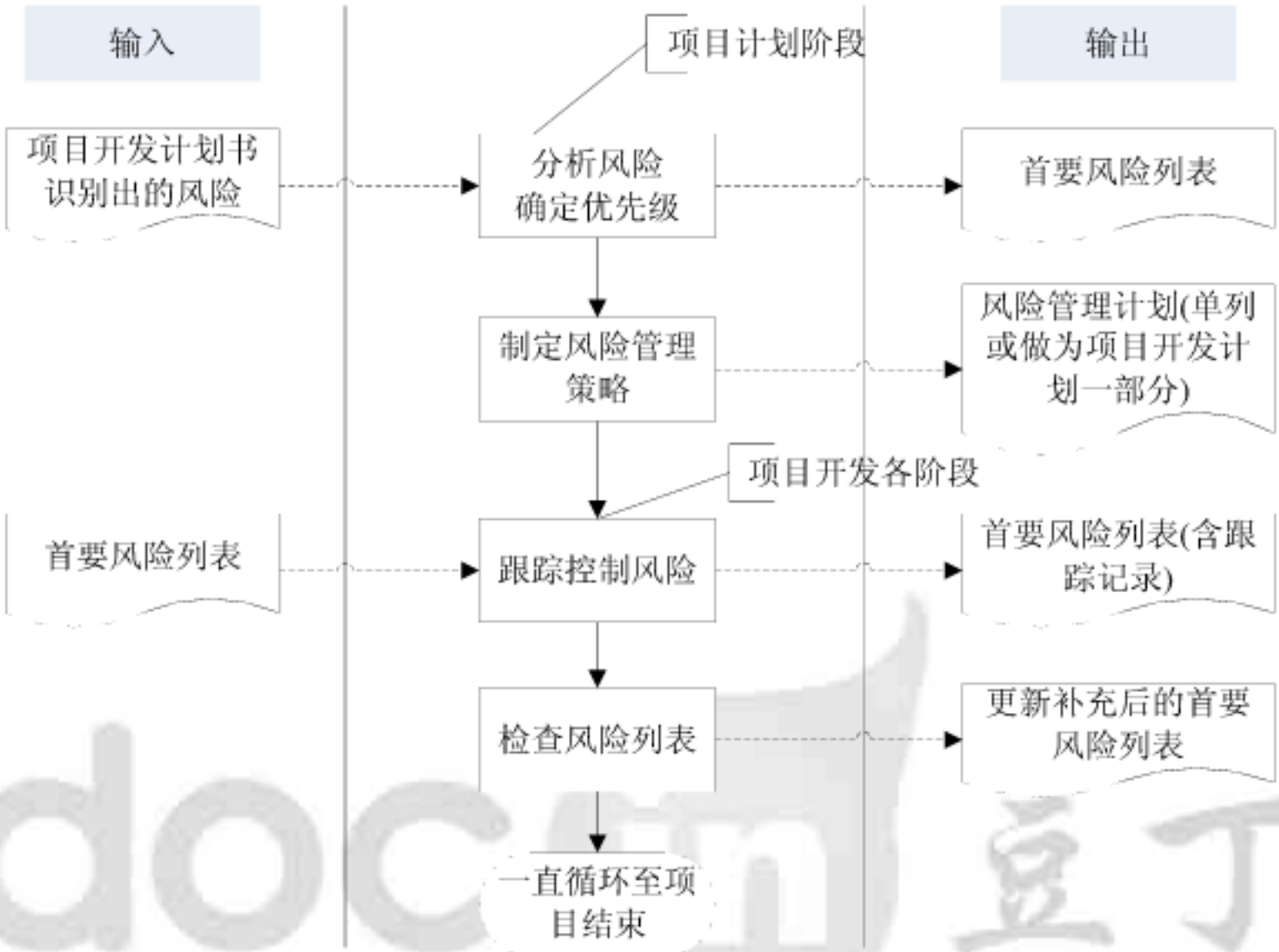
9.4 风险管理流程

9.4.1 风险管理流程图

在项目风险管理流程中，主要分为：一是识别及分析风险，得到主要的风险列表；二是制定风险管理策略，形成风险管理计划；三是对风险进行跟踪控制，并在此过程中再识别分析可能出现的新风险。具体流程如图 9-1（见下页）所示。

风险管理贯穿软件开发的整个生命周期，是项目经理日常工作的一个重要组成部分，在项目组中要形成了“识别——分析——跟踪——关闭”的循环，每周都要对风险进行进行专门的识别、分析及跟踪。

图表 9-1 风险管理活动流程图



9.4.2 识别风险

风险识别是门学科，具有比较强的专业性及管理能力要求，在项目开发过程中，一般可以从下面几个方面进行识别：

- 项目组在项目经理指导下展开风险研讨，必要时吸纳项目相关人员（包括市场人员）参加，对项目开发计划的工作分解结构（WBS）中所有工作要素中可能存在的风险进行识别。
- 对风险的识别可以参照机构提供的风险数据库，对库中罗列的风险项，逐一研讨其在本项目中显含或隐含的可能性。
- 对项目风险的识别，还可以参考其他项目或当前项目的早期阶段中识别出来的或发生过的风险。
- 项目经理负责将识别出来的风险项记录在项目计划的风险估计的风险清单中。

在公司管理或软件研发项目过程中，常见的风险类型有如下几类，具体的分类情况表 9-2 中给出了一些建议供读者参考。

- 需求不明确，或需求分析缺陷，致使最终产品不符合（客户或市场）需要，导致项目目标偏离或在中途作重大变动，延误产品发布时间；
- 对项目工作量估计不足，造成工作不能按计划执行，甚至放弃计划性；
- 客户要求急，开发时间短，加班加点，放松了对过程的控制，导致缺陷不能及时发现，产品性能未达到要求，给后面的产品实施和维护工作带来了较大的压力；
- 测试手段和时间不足，不能充分覆盖所有需求项，导致交付的产品有较多缺陷不能发现。

图表 9-2 风险分类表

产品工程		开发环境		项目约束
1.需求 稳定性 完整性 清晰 有效性 可行性 先例 规模 2.设计 功能 难点 接口 性能 可测试性 硬件约束 非开发软件	3. 编码和单元测试 可行性 单元测试 编码/实现 4. 集成和测试 环境 产品 系统 5. 工程特点 可维护性 可靠性 安全性 保密性 人的因素 特殊性	1.开发过程 正规性 适合性 过程控制 熟悉程度 产品控制 2.开发系统 容量 适合性 可用性 熟悉程度 可靠性 系统支持 交付能力	3.管理过程 计划 项目组织 管理经验 项目接口 4.管理方法 监控措施 人员管理 质量保证 配置管理 5.工作环境 质量态势 合作 士气	1.资源 进度 人员 预算 设备 2.合同 合同类型 约束 依赖关系 3.项目接口 用户 联合承包方 子承包方 主承包方 协同管理 供货商策略

9.4.3 分析风险

项目经理负责组织项目组成员对识别的风险项进行分析，评价风险发生的概率和影响度。必要时，可以邀请相关同行参与风险分析活动。

项目经理组织项目组成员，结合项目管理经验和当前项目实际情况，确定各个风险项的影响估算情况。风险影响是反映风险严重性的一个重要指标，可以按这样的公式计算：风险影响 = 风险发生概率×影响度。

■ 风险发生概率（P）：是指风险发生的可能性。其量化评价方法可按下表描述打分：

定性表示	定量表示（P）	说明
很大	5	风险发生的可能性>80%
较大	4	风险发生的可能性 60%至 80%
一般	3	风险发生的可能性 40%至 60%
不大	2	风险发生的可能性 20%至 40%
很小	1	风险发生的可能性<20%

■ 影响度（C）：是指当风险说明中所预料的结果发生时可能会对项目产生的影响，一旦风险发生而造成的损失，包括成本、进度等多种损失。其量化评价方法可按下表描述进行打分：

定性表示	定量表示（P）	说明
很严重	5	进度延误>30%，或成本超支>30%
严重	4	进度延误 20%~30%，或成本超支 20%~30%
一般	3	进度延误<20%，或成本超支<20%
不太严重	2	进度延误<10%，或成本超支<10%
不严重	1	进度延误<5%，或成本超支<5%

根据以上两个指标的打分情况，我们就可以按下表计算出风险影响量化的值，从而可以对风险进行优先级排序，其中表内的阴影部分表示风险影响比较大，应优先关注或处理。

风险影响		风险可能性				
		很大 5	较大 4	一般 3	不大 2	很小 1
风险后果	很严重 5	25	20	15	10	5
	严重 4	20	16	12	8	4
	一般 3	15	12	9	6	3
	不太严重 2	10	8	6	4	2

风险影响		风险可能性				
	不严重 1	5	4	3	2	1

软件项目常见的前 5 项风险清单如下表所示：

风险	风险陈述	风险背景
资源不足，进度延误	过分乐观的进度，有限的成本，导致进度拖后、成本超支。	人员配备不到位。没有时间进行必需的培训。无法达到进度要求的效率。将加班作为克服进度不够的标准选择。不充分的需求分析导致对产品功能需求的片面理解。
需求不定	不明确的用户需求导致项目软件需求不完整、不确定。	需求文档没有恰当地描述系统构成。接口文档未经确认或批准。需求细节来自现有代码。部分需求（如验收标准）不清楚。因客户方面人员变动引起需求变更或漂移。
人员流失	项目骨干人员中途流失造成项目中断或失败。	长期出差或长时间加班造成骨干人员有厌烦心理。激励不足缺乏激情。个人发展前景不明缺乏信心。团队内部沟通不畅心情不好。开发环境不好工作难度过大。外界吸引再某出路。
项目中途夭折	项目立项时对风险估计不足造成项目半途而废。	用户原因中途中止合同。产品失去市场前景中途停止。竞争对手先行推出或产品功能不及竞争对手，只好停止。骨干流失、技术方案失误造成项目长期拖延。决策失误中途停止项目。
效率低下	长时间的生产效率低下造成项目最终失败。	开发环境不好，影响工作效率。过程管理不严格，造成大量返工。员工培训不够，个人能力欠缺。职责不清、分工不明，造成时间浪费。员工缺少工作激情，影响进度。

9.4.4 制定风险应对策略

通常，应对风险的策略有多种，包括：

- 接纳风险（Acceptance），可以承担风险后果，并为项目计划留出必要的风险储备。
- 回避风险（Avoidance），不参加某些项目的投标，放弃某些项目，放弃项目的某些功能，放弃（项目的）某些目标，等等。

【注意：回避风险的最大风险是丧失机会，付出机会成本。】

- 防范风险（Protection），采取适当措施，减小风险发生可能性和/或风险后果。
- 缓减风险（Reduction），在接纳风险或防范风险的应对策略下，及时采取适当措施，减缓风险发生、防止风险进一步恶化、化解风险、减小风险后果，等等。

- 风险研究（Reserch），收集更多的信息，进一步研究后再定。
- 风险储备（Reserves），为项目进度、成本等留有充分的余地。
- 风险转移（Transfer），例如，外包、外购等。

实际选用何种策略，应视具体情况而定，常用的取舍准则，包括：

- 风险赔率：$\text{风险赔率} = \dfrac{\text{风险应对之前的风险影响} - \text{风险应对之后的风险影响}}{\text{实施风险应对计划的成本}}$
- 分散风险（即不要把全部鸡蛋放在同一个篮子里），意指项目不应过多地依赖于一个供应商、一个客户、一种方法、一个工具以及一个人，等等。

那么，在软件开发过程中的风险也应该引起足够重视，并制定相应的对策。在项目开发时，一般制定风险对策的步骤及包含的内容为：

- 确定各个风险的责任人。
- 确定风险处理方式：
  - ✧ 规避：制定风险规避策略时，项目经理要向总工程师或研发部经理报告，并获得批准。如有必要，还需通知客户以达成一致。
  - ✧ 转移：制定风险转移策略时，项目经理要向总工程师或研发部经理报告，并获得批准，如有必要，还需通知客户以达成一致。
  - ✧ 对某些高优先级的风险不能进行规避和转移，必须承认风险发生的可能性时，可采取接受风险策略来处理风险。
  - ✧ 减缓（降低）：减缓措施主要用在风险发生时。建议制定及时的、正面主动的、具体的应急方案解决问题，以便减少它对项目的影响。
- 制定风险管理计划，纳入项目开发计划或单列，进行评审。

9.5 风险跟踪

9.5.1 风险跟踪

风险跟踪的目的包括：监视风险情景的事件和条件；跟踪风险转化指标及时提供预警报告；为触发机制提供通知及时启动风险行动计划；收集风险应对活动的结果；定期报告风险度量；提供风险状态的可视性。

在开发过程中，风险跟踪是一个日常性的工作，一般采用定期或事件驱动的方式来进行，项目组内所有组员均应当关注项目中的风险。在执行时，对风险的跟踪可按如下步骤进行：

- 质量保证工程师协助项目经理按照项目开发计划，定期或事件驱动地以询问责任人的方式，对风险清单中每个风险项进行跟踪，在风险管理表中记录跟踪状态。
- 项目经理须定期对“已识别”状态的风险重新进行评估，以确定其概率、影响度和优先级是否发生了变化，必要时需对首要风险管理表进行及时的变更。
- 项目经理在项目执行的各个阶段，需要再次对风险进行识别，确定新风险项的概率、影响度、优先级并制定应对策略，必要时对风险管理计划和风险列表及检查表进行及时的变更，以确保风险管理的动态性和完整性。
- 项目经理对照计划定期通报风险的情况，在定期的会议上通告相关人员目前的主要风险以及它们的状态，与计划进行对照回顾风险状态，加强项目组内部交流。

9.5.2 风险应对

随着项目的进展，风险监控活动开始进行。项目管理者监控某些因素，这些因素可以提供风险是否正在变高或变低的指示。例如，频繁的人员流动被标注为一个项目风险，基于以往的历史和管理经验，人员流动的概率为 70%，而影响被预测对于项目成本及进度有严重的影响。应该监控下列因素：

- 项目组成员对项目压力的一般态度。

- 项目组的凝聚力。
- 项目组成员彼此之间的关系。
- 与报酬和利益相关的潜在问题。
- 在公司内及公司外工作的可能性。

除了监控上述因素之外，项目管理者还应该监控风险缓解步骤的效力。例如：上例中，风险缓解步骤要求定义“文档的标准，并建立相应的机制，以确保文档能被及时建立”。如果有关键的人物离开了项目组，这是保证工作连续性的机制。项目管理者应该仔细地监控这些文档，以保证文档内容正确，当新员工加入该项目时，能为他们提供必要的信息。

风险管理及意外事件计划假设缓解工作已经失败，风险变成了现实。继续前面的例子，假定项目正在进行中，有一些人宣布将要离开。如果按照缓解策略行事，则有后备人员可用，因为信息已经文档化，有关知识已经在项目组中广泛进行了交流。此外，项目管理者还可以暂时重新将资源调整到那些需要人的地方去，并调整项目进度，从而使新加入的成员能够“赶上进度”。同时，要求那些要离开的人员停止工作，进入“知识交接模式”。

风险管理计划中的步骤将导致额外的项目开销。因此，风险管理的部分任务是评估何时由风险管理计划中的步骤所产生的效益低于实现它们所花费的成本。本质上是讲，项目计划者执行一个典型的成本－效益分析来估算项目开销变化情况。

对于一个大型项目，可能会标识出 30－40 种风险。如果为每种风险定义三至七个风险管理步骤，则风险管理本身就可能变成一个“项目”。经验表明：整个软件风险的 80%（即可能导致项目失败的 80%潜在的因素）能够由仅仅 20%的已知风险来说明。早期风险分析步骤中所实现的工作能够帮助计划者确定哪些风险在所说的 20%中。

第10章 项目跟踪及控制

内容提要：

- **CMMI** 对应实践
- 项目跟踪及控制简述
- 项目跟踪及控制活动
- 收集项目度量数据
- 处理项目偏离

项目跟踪、监测和控制是项目管理的重要活动，贯穿项目生命周期的全过程。从 CMMI 的角度来说，项目监控，有两个特定目标：

- 通过跟踪、监测，及时了解项目计划的实际执行情况（包括工作量、成本、进度、缺陷、承诺以及风险等），评价项目状态，为项目组长以及各级管理者提供项目当前真实情况的可视性，并用以判断项目是否沿着计划所期望的轨道健康地取得了进展。
- 如果项目状态偏离了期望的轨道，例如工作量或进度的偏离超过了允许的门限值，则应采取纠正措施，改进过程性能，使项目的规模、工作量、进度、成本、缺陷以及风险得到有效控制，必要时修正项目计划，最终将项目调整到计划所期望的轨道上。

项目跟踪的内容，包括活动跟踪、缺陷跟踪、变更跟踪和争议问题（Issues）跟踪。项目生命周期各个阶段均包括若干项活动，一项活动又由若干个任务组成，这些应当在工作分解结构（WBS）里明确。

10.1 CMMI 对应实践

在 CMMI 中有一个专门的过程域（PA）对应于项目监督与控制（Project Monitoring and Control，简称 PMC），其目的是提供对项目进度的理解，以便当项目性能显著偏离计划时采取适当的纠正措施。通过定期评审和里程碑评审二种方式，监测项目实际性能（计划参数、风险、承诺、资料、相关方参与情况等），管理纠正措施。

在 CMMI 标准中，项目跟踪与控制分为两个部分：按照项目计划监控项目；管理纠正措施直到关闭。在项目监督与控制过程域中，共有两个特定目标需要实现，对子对应的有 10 个实践，具体为：

SG1 Monitor Project Against Plan（按照项目计划监控项目），按照项目计划来监控项目实际的进度及性能。通常是通过对个人周报、项目周报、里程碑报告/阶段进度报告等的评审或数据收集分析来执行。

SP1.1 Monitor Project Planning Parameters（监控项目计划的要素），在项目开发过程中，需要按照项目计划来监控与之相关要素的实际值。通常是形成项目性能记录及项目偏差记录等文档，参照项目计划，可以通过监控项目进度、项目成本、消耗的工作量、工作产品及任务的属性、使用的资源、项目组成员的知识和技能等方面来完成。

SP1.2 Monitor Commitments（监控承诺），按照项目计划的规定监控承诺的实现情况。对于尚未满足的承诺，特别是不满足就会造成严重风险的承诺必须进行重点监控。

SP1.3 Monitor Project Risks（监控项目风险），按照项目计划的规定监控风险。形成项目风险监控记录，针对项目当前状态和发生的事件，定期评审风险管理计划；当出现新情况时，评审并修改风险管理计划；向项目相关各方通报风险状态。

SP1.4 Monitor Data Management(监控数据管理)，按照项目计划监控项目数据的管理，项目中形成的各类记录及文档、工作产品等都属于此范围。

SP1.5 Monitor Stakeholder Involvement（监控干系人的参与），按照项目计划监控项目相关各方的参与情况。

SP1.6 Conduct Progress Reviews（执行进度评审），参见 4.1 节。

SP1.7 Conduct Milestone Reviews（执行里程碑评审），参见 4.1 节。

SG2 Manage Corrective Action to Closure（管理纠正措施直到关闭），当项目性能或者结果明显偏离计划时，采取纠正措施，并对这些纠正措施进行管理，直到关闭。

SP2.1 Analyze Issues（分析问题），收集和分析问题，并决定解决问题的纠正措施，形成需要纠正的问题清单，并附上纠正措施。需要通过两个步骤来达到，一是收集问题，二是分析问题以决定是否有必要采取纠正措施。

SP2.2 Take Corrective Action（采取纠正措施），针对问题采取纠正措施。一般是通过如下几种方式来达到：决定并记录解决已认定问题的纠正措施；与项目相关各方一起评审拟采取的纠正措施并达成协议；协商内部和外部承诺的变更。

SP2.3 Manage Corrective Action（管理纠正措施），对采取的纠正措施进行管理，跟踪直至关闭，并且把结果形成记录。

10.2 项目跟踪及控制简述

本书讲的项目跟踪及控制是项目管理的最基本的监控，更严格的定量化监控则作为 CMMI4 级的要求（过程性能定量管理、软件质量定量管理），至于缺陷原因分析和缺陷预防更是 CMMI5 级的要求。风险监控以及偏离范围的确定应属于 CMMI3 级的范围，详见风险管理相关章节。在软件开发过程中，对于整个项目进展及其他参数进行跟踪与控制，一般可以遵循以下方针：

- 项目组应按照项目定义过程开展项目软件开发活动。
- 项目经理负责，依据经评审、批准的项目开发计划书进行项目跟踪和监督。
- 项目组全体成员必须按时如实填写个人工作周报/工作日志；项目经理汇总形成项目组周报，跟踪过程度量数据，了解并掌握项目状态及存在的主要问题。
- 项目经理主持召开项目例会，会上交流、讨论项目状态，进度偏离的处理、质量保证工程师发现的不符合项处理等，并形成会议记录。

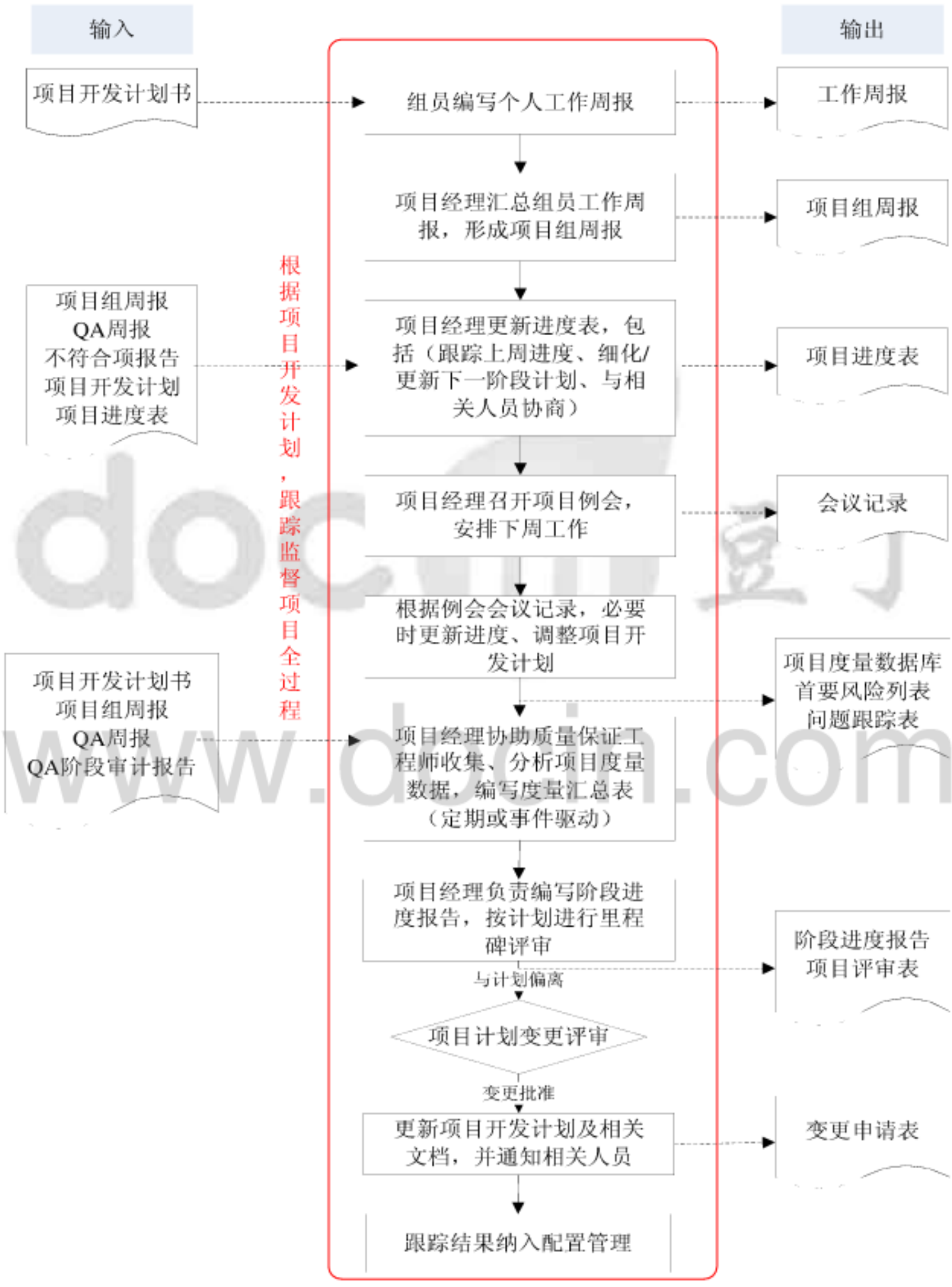
- 定期或事件驱动形成阶段进度报告，汇总并分析包括规模、工作量、进度、风险、成本等在内的度量数据，根据分析结果评价项目当前状态。
- 总工程师负责里程碑处评审，当项目的实际偏离项目开发计划过大时，应采取措施改进过程，必要时修改和调整项目开发计划，注意项目开发计划调整或相关约定的修改应及时与相关组和个人协商确认；如果涉及外部用户或其他部门的相关组，则应请总工程师协助解决。

项目跟踪及控制目的是通过跟踪，及时了解项目开发计划的实际执行情况，评价项目状态，采取监督措施，改进过程效能，使项目的规模、工作量、进度、成本、关键计算机资源以及风险得到有效控制，必要时调整或比变更项目开发计划。项目跟踪的主要内容为：规模跟踪；工作量跟踪；进度跟踪；争议问题跟踪；成本跟踪；风险跟踪；关键计算机资源跟踪。

在实际执行项目跟踪与控制活动时，需要将各项跟踪结果文档化，并进行分析、汇总，达到项目过程可控的目的。项目跟踪与控制的整体流动流程如图 10-1（见下页）所示，在此过程中各类人员的职责如下：

- 项目经理：负责项目跟踪监测和控制；汇总项目组成员的个人工作周报，编写或确认项目周报；主持项目例会和日常评审活动；编制、提交项目状态报告；根据需要及时采取纠正措施包括调整项目计划。
- 项目组员（包括配置管理）：每周填写个人工作周报、参加项目例会；完成项目组长指派的其他监控任务。
- 质量保证工程师：验证各项监控活动与规范、规程的符合性，提交审核报告
- 研发部经理：主持里程碑评审；批准涉及里程碑计划变更的项目计划变更，确认涉及发布计划变更的项目计划变更；解决跟踪过程中项目经理不能解决的争议问题。
- 项目相关各方（干系人）：提交必要的工作周报；参与承诺监测活动；参加里程碑评审；承诺项目计划变更引起的职责分工的改变。
- 总工程师：参加里程碑评审，签署评审结论；批准涉及发布计划变更的项目计划变更；解决跟踪监控过程中项目经理和研发部经理不能解决的争议问题。

图表 10-1 项目跟踪流程图



10.3 项目跟踪活动

由图10-1可见，在项目跟踪时通过多种方式来进行，原则上是从底到上的方式进行跟踪，即先是个人对工作完成情况进行跟踪，然后再项目组整体进行跟踪。常用的方式主要有：《个人工作周报》、《项目组周报》、召开项目例会、举行里程碑评审。然后根据这些跟踪方式收集的数据，再对项目进度进行更新，有时还可能需要调整项目计划。具体讲解如下：

一、个人工作周报

项目组成员（包括测试人员、配置管理员）每周五（不同公司此时间可能规定不同）及时按《个人工作周报》模板总结本周的活动结果，编写个人工作周报提交项目经理和质量保证工程师。个人工作周报的主要内容有：

- 每天及时填写的工作日志，及工作量汇总（可自动生成）；本周工作小结（描述本周工作任务的完成情况，并记录各项工作任务的规模、工作量）；
- 问题反馈和建议；
- 下周个人工作安排；

质量保证工程师利用《QA 周报》每周一报告本项目上周的质量保证工作情况；配置管理员利用《个人工作周报》每周五下午报告本项目本周配置管理工作情况。

二、项目组周报

项目组周报由项目经理每周五（有些公司可能规定为周一）根据项目组成员、配置管理员提交的个人工作周报，汇总本周项目任务完成情况；项目经理在项目过程中跟踪结果与计划相比较；对这些内容分析形成周报。项目组周报主要内容有：

- 本周工作小结，汇总项目组各成员提交的个人工作周报 ，收集本周所有任务的度量数据（包括任务的估计规模和实际规模、估计工作量和本周工作量及剩余工作量、完成状态等；并用简要文字总结本周的实际工作情况及工作成果。
- 建议与问题反馈，汇总项目组成员提交的建议与问题，并总结以前“已识别”问题的解决情况和建议的落实情况。

- 下周工作计划，根据《项目开发计划》中 WBS 列举的工作任务及对本周工作完成情况分析，安排项目组下周工作任务。
- 变更，汇总在本周任务执行过程中发生的所有变更请求记录及其相应的变更控制表。
- 跟踪，跟踪风险　每周更新《首要风险列表》；跟踪关键计算机资源；跟踪成本（即工作量），完善项目度量数据库；跟踪项目组发现的问题，更新《问题跟踪表》。

跟踪步骤一般建议为：跟踪规模——跟踪工作量——跟踪风险——跟踪进度——跟踪关键计算机资源——跟踪成本——跟踪问题 ，将跟踪结果与估计相比较，进行分析，将结果写入项目组周报。

三、项目例会

项目经理每周一安排时间（通常为刚开始上班的时间）进行项目组内部交流讨论，例会的时间可以根据实际情况而定。对于大型项目，周会的召开需要形成一个层次结构，比如先召开项目领导小组的周会，再召开各个小组的周会。再就是，周会的形成并不一定非要面对面坐在一起召开，可以采用多种灵活的方式来进行。但是，无论通过何种形式召开周会，均需要达到以下几个目的：

- 对项目中存在的争议问题（包括技术上、管理上）讨论，形成处理结果；
- 通报项目的总体进度，以及项目跟踪的结果，如风险、成本、进度等，对跟踪发现的问题达成一致的处理意见；
- 讨论并确定下一阶段的工作安排及下周的工作任务；
- 涉及项目开发计划的调整或者相关的变更请求，需要在例会中讨论达成一致意见；
- 讨论解决质量保证工程师发现的不符合项。

项目例会可以按如下方式进行召开，各公司会有各公司的规定，此处给出的是某公司对项目例会召开的规定。

- 项目经理负责在每周一召开项目例会 （如遇特殊情况不能按期召开例会，则需在例会之前通知相关人员并说明理由；项目经理可以决定将项目例会延期或者通过电子邮件等其他方式进行沟通，保证项目组成员每周就项目存在问题、进展情况及风险

有进行讨论，达成一致意见 ）；

- 项目经理在会前指定会议记录人，并将本周有变化的《首要风险列表》、或《问题跟踪表》等相关资料交给记录人，保证例会讨论的内容能被详细、准确记录 ；

在项目例会上需要完成以下内容 ：

- 项目组成员各自分别讲述上周的工作任务完成情况、未完成的工作及其原因、工作中发现的问题以及解决方法等；
- 项目经理根据汇总的项目组周报内容，总结上周的工作完成情况，让项目组员及时了解项目状态；
- 讨论识别出项目组内和相关组间的争议问题和潜在问题；
- 讨论质量保证工程师例行检查发现的不符合项，确定解决方法，指定专人解决；
- 例会上讨论并确定项目经理根据项目组周报内容更新后的项目开发计划，主要是下一阶段和或近几周的工作进度表；
- 项目例会结束后，项目经理或指定专人形成会议记录、更新项目进度表（有变动时）、更新各类跟踪表（包括《首要风险列表》、《问题跟踪表》等）。

问题跟踪应采集的度量数据则包括在生命周期不同阶段发现的问题检测阶段、问题个数、问题类型、问题注入阶段、问题严重程度分类以及问题状态（已提交、关闭）等。

争议问题跟踪则包括争议问题状态以及因争议问题引起的活动的工作量。此处顺便说明一下什么叫争议问题。项目组成员按项目计划进行工作时，随时都可能遇到许多开发计划之外的问题，例如，对标准的理解、对规程的理解、对各类项目文档的理解以及在工作过程中遇到的各种困难或疑问，用户或其他项目相关组也可能提出许多计划外的问题，这些问题统称争议问题（Issues）。项目组长应及时处理争议问题，否则会影响项目进度。记住，没有解决的争议问题会给项目带来风险。

四、里程碑评审

在项目进行到重要的阶段或里程碑阶段，项目经理需要对项目情况进行总结，形成《阶段进度报告》。一般在里程碑评审之前完成，在评审时用于评审当前的项目状态，与项目开发

计划进行比较，及时发现、解决项目过程中存在的问题。

总工程师、研发部经理等高层领导通过里程碑评审确认项目前期阶段的工作成果，并对下一阶段项目安排和活动内容达成一致意见，从而更好的进行项目过程控制。工作步骤有：

- 项目经理及质量保证工程师在重要阶段和里程碑处跟踪项目进度，在《项目度量数据库》形成《项目参数图表分析》分析进度情况，主要包括：项目工作量按阶段分布、项目工作量按类别分布、工作量偏差趋势分析、项目进展盈余分析、项目进度成本偏差趋势分析、项目进度成本性能指标趋势分析等。递交给研发部经理和总工程师；
- 配置管理员利用《基线计划及跟踪表》跟踪报告项目的里程碑状态，递交给项目经理，研发部经理和总工程师；
- 在里程碑评审之前，项目经理需要负责完成《阶段进度报告》，并且组织正式评审，以便确定当前项目状态并对项目下一阶段达成共识、得到承诺，里程碑评审后，形成《项目评审表》，具体评审过程参见第 4 章 项目评审管理。

《阶段进度报告》完成后，发送给总工程师、研发经理、质量保证工程师等相关组和个人，并交配置管理员纳入配置管理。《阶段进度报告》包含的主要内容有：

- 报告时间及所处的阶段名称；
- 项目进度（本阶段主要活动说明、实际与计划比较分析结果、进度性能指数）；
- 工作中遇到的问题及策略（说明项目过程中遇到的问题，及采取的解决措施）数据来源项目组周报；
- 本阶段完成的工作产品（说明本阶段完成的工作产品清单）；
- 风险管理状态、质量保证状态、配置管理状态、需求管理状态等；
- 下阶段工作安排；
- 特殊问题。

另外，对于项目过程出现意外情况，需按项目停止申请原则进行：里程碑评审时，发现项目出现意外情况，必须暂停或终止时，由项目经理填写《项目停止申请表》，说明问题，提

交相关人员签字后，项目方可暂停或停止研发，直至问题解决，重新编制项目开发计划进行。

在里程碑阶段形成的阶段进度报告提交之后，需要执行里程碑评审，相当于项目进展过程中的一种阶段性总结活动。所谓里程碑（milestone），实际上就是项目进展过程中的若干个时间点，这些时间点是在项目计划阶段定义的，并且得到项目相关各方的同意和承诺，在这些时间点上按计划规定进行一次较全面的评审活动，即里程碑评审。较全面的含义包括二个方面：

- 参加评审的人员较全面，除了项目组成员和质量保证工程师、测试工程师之外，重要的还要有高层管理者、用户代表以及其他项目相关各方的代表参加；
- 评审的内容较全面。

里程碑评审，应根据项目组提交的项目状态报告，逐项评价项目状态，并提出改进建议和可采取或应采取的纠正措施，例如：

- 是否按计划完成了工作产品？
- 规模变化是否正常？
- 实际工作量与计划工作量比较，偏离是否正常？
- 实际进度与计划进度比较，偏离是否正常？
- 缺陷分布、缺陷密度、缺陷修复率是否正常？
- 变更是否得到有效控制？
- 争议问题是否得到及时处理？
- 承诺是否兑现？
- 风险管理有哪些变化？
- 培训是否有效？

【注意：里程碑和基线（baseline）是二个不同范畴的概念。基线是一项或若干项配置项的组合，为了配置管理中有效控制配置项的完整性，在项目进展过程中可以设置若干个基线，并分别定义每个基线由哪些配置项组成。如需求基线，由需求文档、需求规格说明书组成；而策划基线，则除了前二个配置项外还增加了项目计划等配置项。由于实际上，基线设置往往与里程碑相匹配，所以二个概念容易混淆。】

【说明：进度性能指数用来衡量项目能否准时交付用的，是进行里程碑评审的主要目的，通过项目分段进行控制，最终希望控制项目的交付具体时间，这也是项目跟踪的最终目的，可以如下定义：进度偏离值（百分比）=（现计划结束日期-初始计划结束日期）/现计划项目周期。其中：现计划项目周期=（现计划结束日期-实际起始日期）+1；单位：天。】

五、更新进度表

项目经理根据周报中汇总的上周各项工作任务的完成情况，更新 WBS 中的项目进度，标注各个任务完成百分比；可以使用项目管理的工具来完成此工作。对于未完成的工作，检查估计剩余工作量和估计完成时间。

- 若估计完成时间超出进度表中浮动范围，但不涉及里程碑点或发布时间点，项目经理需要与项目组员进行沟通，分析任务的难点、商量的具体完成时间，若完成时间不能改变则根据情况做相应的任务调整；
- 若进度延迟涉及里程碑点或产品发布点，经与组员协调确认不能按期完成，则项目经理应与研发经理协商，获得解决方法；

进行项目工作量偏差分析，工作量性能指数（SPI）=（实际工作量总和+估计剩余工作量）/（计划工作量总和）

- 如果 SPI > 1.0，则表明到当前报告时间为止，工作量估计偏低，超过 1.2，需要重新进行项目规模（工作量）估算，调整项目开发进度。
- 如果 SPI < 1.0，则表明到当前报告时间为止，工作量估计偏高，低于 0.8，需要重新进行项目规模（工作量）估算，调整项目开发进度。

问题优先级识别，项目经理识别并分析本周发现的问题的严重程度，得出初步的解决策略，在项目例会中进行讨论，填写《问题跟踪表》。

10.4 收集项目度量数据

项目经理、项目质量保证工程师负责在项目过程中收集和分析度量数据，主要的度量数据有：规模估计值和实际值；工作量估计值和实际值；进度估计值和实际值；风险的估计值

和实际值；关键计算机资源的估计值和实际值；成本的估计值和实际值；偏差分析结果。

度量数据收集有如下步骤：

1. 项目经理每周需要收集的项目度量数据反映在项目组周报中，收集方式和度量数据内容详见“项目组周报”；
2. 质量保证工程师每周跟踪检查项目组周报、《首要风险列表》等的填写内容是否符合规定的要求、相关度量数据是否完成，并收集相关数据填写到《项目度量数据库》中的相应表格；
3. 质量保证工程师每周从项目组周报、QA 周报、《QA 不符合项报告》、《项目评审表》、测试问题报告中跟踪项目中发现的缺陷及其解决情况，在《度量汇总表》中填写《缺陷度量数据汇总表》，并通过《工作量分析图》分析各阶段的缺陷情况；
4. 质量保证工程师定期或事件驱动对收集的度量数据进行分析，将表格提交项目组相关成员。

10.5 处理项目偏离

项目例会和里程碑评审结束后，项目经理将项目实际实施情况和计划进行比较，采取措施调整更新项目进度，必要时进行项目开发计划变更。

1、计划更新（修订）

- 项目经理根据项目组周报、QA 周报和项目例会讨论结果，将项目实际实施情况和计划进行比较，采取措施调整项目进度。
- 估算值与实际值偏离较大时，需要对项目的活动重新进行估算，更新估算结果。
- 项目计划更新的内容包括：不涉及本里程碑进度的任务调整和细化，下阶段的任务细化，人员调整等。
- 项目计划的更新频率可以是每周或由项目经理根据事件驱动决定。
- 项目计划的更新由项目经理在项目开发库中实时进行，并定期纳入受控库进行管理。
- 项目计划的更新没必要生成新版本。

2、计划变更

在涉及到里程碑进度调整时，必须执行计划变更。导致计划产生的变更分两类：配置变更和项目变更。配置变更，如对需求、设计、代码等的变更。项目变更有：

- 生命周期模型改变。
- 规模、工作量、成本预算或进度的变更，估算值与实际值偏离较大时，需要对项目的活动重新进行估算，更新估算结果。
- 度量极限数据小于预定数据。

计划变更内容（ 项目软件开发计划文档，补充对下列内容的重新估算）：

- 成本，规模，关键计算机资源，风险评估，成员人数分配等；
- 新计划的交付日期，工作任务，在项目各个阶段中计划的工作量；
- 项目质量保证计划、CM 计划、测试计划、需求跟踪表等相关文档；
- 其它需要被更新以维护各个工作产品间一致性的工作产品；
- 版本修订控制；

项目需要为变更后形成基线的 WBS 保存比较基准。计划变更权限如下：

- 项目变更由研发经理/CCB 评审和批准计划变更。
- 软件过程变更由 EPG 评审和批准。
- 由配置变更引起的计划变更需经过项目经理/CCB 评审并且批准。

计划变更必须按照配置管理过程——变更控制实行，通知所有相关组别，包括项目经理、研发部经理、总工程师和质量保证工程师。项目经理、总工程师要保证客户或客户代表得到通知，并得到客户的批准。变更后的项目开发计划书获得项目组内部，项目组外部和机构外部的承诺。批准的变更和更新的项目开发计划（含更新后的项目进度表）应该通知给所有受影响的组或个人。变更后的项目开发计划由项目经理统一提交给配置管理员进行配置管理。项目的开发和跟踪活动以变更/更新后的计划为基础。

第11章 系统设计

内容提要：

- **CMMI** 对应实践
- 系统设计简述
- 关于设计模式
- 概要设计活动
- 详细设计活动
- 设计方法简介

系统设计是把软件需求规格说明转达化为软件系统的最重要的技术开发环节。通常，软件设计的技术难度要比编程、测试高。所以程序员、测试员称为“员”，而设计师尊称为“师”。在企业里，设计师的地位和收入通常高于程序员和测试员。

设计的好坏决定了软件系统的优劣。我们可以断言“差的设计必定产生差的软件系统”，但是不能保证“好的设计必定产生好的软件系统”。因为在设计之前有需求开发工作，在设计之后还有编程、测试和维护工作，无论哪个环节出了差错，都会把好事搞砸了。[10]

11.1 CMMI 对应实践

系统设计在 CMMI 中是放到技术解决方案（Technical Solution，简称 TS）过程域的，其中共有 6 个实践与此相对应。就整个 TS 来说，其目的为：针对需求进行设计解决方案，并实现之。这些解决方案、设计和实现应当单独或适当的组合的一起，并且是围绕着产品、产

[10] 《IT 企业研发管理——问题、方法和工具》，林锐 刘兴文 徐继哲 唐勇编著，电子工业出版社 P161

品组件和产品相关的生命周期来进行的。本章所讲解的内容主要是与解决方案的设计、选择有关内容，与之相关的 6 个实践为：

SG1 Select Product Component Solutions（选择产品组件解决方案），目的是从候选的解决方案中选择产品或产品组件的解决方案。

为了能从成本、进度和技术性能的角度出发选择适合于当前软件的解决方案，必须对候选方案进行分析。在选择方案时需要确定一定的准则，一般涉及成本、效益和风险等。

SP1.1 Develop Alternative Solutions and Selection Criteria（开发候选方案和制定选择准则），需要从以下几个方面进行考虑：成本（开发、制造、采购、维护、支持等的功能），性能，产品组件的复杂程序和有关的生命周期过程，产品操作的稳定性和使用条件、操作模式、操作环境和产品生命周期过程中的各种变化，产品的扩展和进化，技术限制，风险，需求和技术的发展变化，最终用户和操作人员的能力及限制，现有产品的特点（可能会被选择做为本项目的解决方案构成部分或全部）。形成的工作产品一般为：候选方案评价准则、候选方案、新技术评估报告、对市场现有产品的评价结果等。

SP1.2 Select Product Component Solutions（选择产品组件方案），选择最能满足已建立的标准的产品组件候选方案。在开发过程中，渐进开发技术相关资料为技术解决方案及开发详细设计，并实现设计。维护选择理由的纪录对后续的决策十分重要。这种纪录可使后续的干系人免于返工，也可在某些适用的应用环境下，提供对技术应用的深入见解。

SG2 Develop the Design（开发设计），目的是开发产品或产品组件的设计。

产品或产品组件的设计，必须提出适当的内容，这不仅是为了实现，也是为了产品生命周期阶段，如修正、重新采购、维护、维持及安装。设计文件提供相关的干系人，以方便对设计的相互了解提供参考，并在产品的开发与后续的生命周期阶段设计上的改变。完整的设计描述，记录于技术相关数据（文档）中。

SP2.1 Design the Product or Product Component（设计产品或产品组件），产品设计包含两阶段，在执行上可能相互重叠：概要设计与详细设计。概要设计建立产品功能与架构，包含产品组成区块、产品组件界定、系统状态与模式、主要的内部接口，以及外部产品

接口。详细设计完整的定义产品组件的结构与功能。主要由以下几个工作组成：第一，建立并维护设计准则，以评估设计；第二，界定、开发或获取适合于产品的设计方法，选择适合的方法并在一定的工具支持下可以对设计提供很大的帮助，一般常用的技术和方法包含，原型法、结构化设计方法、面向对象设计方法、E-R 模型、设计复用、设计模式等；第三，确保设计遵循所应用的设计标准与准则；第四，确保设计遵循已分配的需求；第五，设计文档化。

SP2.2 Establish a Technical Data Package（建立技术数据包），目的是建立并维护技术数据包。完备的技术数据包为开发者提供了开发产品或产品组件的综合性描述，还提供有关产品类型的以下信息：产品架构描述、分配的需求、产品组件的描述、产品相关生命周期过程描述、关键产品特性、必需的物理特征和约束、接口需求、用于确保实现需求的验证准则等。

SP2.3 Design Interfaces Using Criteria（使用准则设计接口），目的是使用已建立的准则来设计产品组件接口。一般形成的工作产品有：接口设计规格说明、接口控制文档等。主要由以下几个工作组成：第一，定义接口准则，这些准则一般是机构过程资产的一部分；第二，识别与其他产品组件相关的接口；第三，识别与外部相关的接口；第四，识别产品组件与产品相关生命周期过程之间的接口；第五，应用准则来设计接口候选方案；第六，记录已选取的接口设计与理由。

SP2.4 Perform Make, Buy, or Reuse Analyses（执行自制、购买或再用之分析），目的是根据已建立的准则，评估产品组件是要开发、购买或再用。技术状况是对开发或得采购产品组件作出选择的重要理由。在开发工作很复杂时，可能以采购现有组件为佳，而拥有先进工具及允足人员情况下则支持自己开发。毕竟，有时购买现有的组件，可能不够完备或不能完全满足系统的需要。一旦作为采购现有组件（或外包开发）的决定，就要在供应商协议中进行落实。

11.2 系统设计简述

系统设计的目的是把用户需求（即《软件需求规格说明书》）从实现的角度进行翻译，能够被开发人员（程序员）顺利地实现。一般分为概要设计和详细设计两部分。从另外一个角度来看，系统设计分为用户功能设计及软件技术架构设计两块内容。在本书中，软件技术架构设计我们放到技术预研时来提供，讲述的重点是用户功能（或需求实现）的设计。

概要设计的目的，一是分析与设计具有预定功能的软件系统体系结构（即模块结构），确定子系统、功能模块的功能及其间的内、接口，确定数据结构；二是设计整个系统使用的技术架构。

详细设计的目的是在给定的技术架构下，设计系统所有模块的主要接口与属性、数据结构和算法，指导模块编程。

从管理的角度来说，对于系统设计应当遵循以下的方针：

- 在每个软件项目的设计阶段，对根据《软件需求规格说明书》设计系统的整体架构并形成概要设计说明书。
- 对概要设计说明书中的每一项功能，详细分解，确定具体算法及数据结构（或数据库结构），形成详细设计说明书，为模块编程提供基础。
- 概要设计说明书和详细设计说明书必须通过相关组的审查，相关组包括：开发组、测试组、质量保证组、配置组、文档组及个人。只有相关组审批、确认后的概要设计说明书和详细设计说明书，才能作为项目开发计划以及后续项目工程和管理活动的基础。
- 经过评审确认的概要设计说明书和详细设计说明书任何变更，都应得到控制和管理，一旦需求发生变更，项目计划、过程活动、工作产品等要随之变更与需求变更保持一致，填写“配置项变更控制表”，并重新提交相关组和个人复审。

11.3 关于设计模式[11]

设计模式（Design pattern）是一套被反复使用、多数人知晓的、经过分类编目的、代码设计经验的总结。使用设计模式是为了可重用代码、让代码更容易被他人理解、保证代码可靠性。毫无疑问，设计模式于己于他人于系统都是多赢的，设计模式使代码编制真正工程化，设计模式是软件工程的基石，如同大厦的一块块砖石一样。

1995 年，由 Erich Gamma、Richard Helm、Ralph Johnson 和 John M. Vlissides 著作的《Design Patterns: Elements of Reusable Object-Oriented Software》，该书的中译本已于 2000 年出版。该书总结的设计模式有：

- Creational Patterns（构造型模式）
  - ✧ Abstract Factory（抽象工厂），提供一个创建一系列相关或相互依赖对象的接口，而无需指定它们具体的类。适用于：一个系统要独立于它的产品的创建、组合和表示时；一个系统要由多个产品系列中的一个来配置时；当要强调一系列相关的产品对象的设计以便进行联合使用时；当提供一个产品类库，而只想显示它们的接口而不是实现时。
  - ✧ Builder（建造器），将一个复杂对象的构建与它的表示分离，使得同样的构建过程可以创建不同的表示。适用于：当创建复杂对象的算法应该独立于该对象的组成部分以及它们的装配方式时；当构造过程必须允许被构造的对象有不同的表示时。
  - ✧ Factory Method（工厂方法），定义一个用于创建对象的接口，让子类决定将哪一个类实例化。Factory Method 使一个类的实例化延迟到其子类。适用于：当一个类不知道它所必须创建的对象的类的时候；当一个类希望由它的子类来指定它所创建的对象的时候；当类将创建对象的职责委托给多个帮助子类中的某一个，并且你希望将哪一个帮助子类是代理者这一信息局部化的时候。

[11] 本节内容根据互联网资料整理编辑而成。

- ✧ Prototype（原型），用原型实例指定创建对象的种类，并且通过拷贝这个原型来创建新的对象。适用于：当要实例化的类是在运行时刻指定时，例如，通过动态装载；为了避免创建一个与产品类层次平行的工厂类层次时；当一个类的实例只能有几个不同状态组合中的一种时。
- ✧ Singleton（单例），保证一个类仅有一个实例，并提供一个访问它的全局访问点。适用于：当类只能有一个实例而且客户可以从一个众所周知的访问点访问它时；这个唯一实例应该是通过子类化可扩展的，并且客户应该无需更改代码就能使用一个扩展的实例时。

■ Structural Patterns（结构型模式）

- ✧ Adapter（适配器），将一个类的接口转换成客户希望的另外一个接口。该模式使得原本由于接口不兼容而不能一起工作的那些类可以一起工作。适用于：使用一个已经存在的类，而它的接口不符合的需求；创建一个可以复用的类，该类可以与其他不相关的类或不可预见的类（即那些接口可能不一定兼容的类）协同工作；使用一些已经存在的子类，但是不可能对每一个都进行子类化以匹配它们的接口。
- ✧ Bridge（桥梁），将抽象部分与它的实现部分分离，使它们都可以独立地变化。适用于：不希望在抽象和它的实现部分之间有一个固定的绑定关系，例如这种情况可能是因为，在程序运行时刻实现部分应可以被选择或者切换；类的抽象以及它的实现都应该可以通过生成子类的方法加以扩充，这时 Bridge 模式使你可以对不同的抽象接口和实现部分进行组合，并分别对它们进行扩充；对一个抽象的实现部分的修改应对客户不产生影响，即客户的代码不必重新编译；想对客户完全隐藏抽象的实现部分。
- ✧ Composite（合成），将对象组合成树形结构以表示“部分-整体”的层次结构。它使得客户对单个对象和复合对象的使用具有一致性。适用于：表示对象的部分-整体层次结构；用户忽略组合对象与单个对象的不同，用户将统一地使用组合

结构中的所有对象。

✧ Decorato r（装饰），动态地给一个对象添加一些额外的职责。就扩展功能而言，它比生成子类方式更为灵活。适用于：在不影响其他对象的情况下，以动态、透明的方式给单个对象添加职责；处理那些可以撤消的职责；当不能采用生成子类的方法进行扩充时。一种情况是，可能有大量独立的扩展，为支持每一种组合将产生大量的子类，使得子类数目呈爆炸性增长。另一种情况可能是因为类定义被隐藏，或类定义不能用于生成子类。

✧ Façade（外观），为子系统中的一组接口提供一个一致的界面，Façade 模式定义了一个高层接口，这个接口使得这一子系统更加容易使用。适用于：要为一个复杂子系统提供一个简单接口时。子系统往往因为不断演化而变得越来越复杂。大多数模式使用时都会产生更多更小的类。这使得子系统更具可重用性，也更容易对子系统进行定制，但这也给那些不需要定制子系统的用户带来一些使用上的困难。此模式可以提供一个简单的缺省视图，这一视图对大多数用户来说已经足够，而那些需要更多的可定制性的用户可以越过 Façade 层。客户程序与抽象类的实现部分之间存在着很大的依赖性。引入 Façade 将这个子系统与客户以及其他的子系统分离，可以提高子系统的独立性和可移植性。需要构建一个层次结构的子系统时，使用 Façade 模式定义子系统中每层的入口点

✧ Flyweight（享元），运用共享技术有效地支持大量细粒度的对象。适用于：一个应用程序使用了大量的对象；完全由于使用大量的对象，造成很大的存储开销；对象的大多数状态都可变为外部状态；如果删除对象的外部状态，那么可以用相对较少的共享对象取代很多组对象。

✧ Proxy（代理），为其他对象提供一个代理以控制对这个对象的访问。适用于：在需要用比较通用和复杂的对象指针代替简单的指针的时候。

■ Behavioral Patterns（行为型模式）

✧ Chain of Responsibilty（责任链），为解除请求的发送者和接收者之间耦合，而

使多个对象都有机会处理这个请求。将这些对象连成一条链，并沿着这条链传递该请求，直到有一个对象处理它。适用于：有多个的对象可以处理一个请求，哪个对象处理该请求运行时刻自动确定；想在不明确指定接收者的情况下，向多个对象中的一个提交一个请求；可处理一个请求的对象集合应被动态指定。

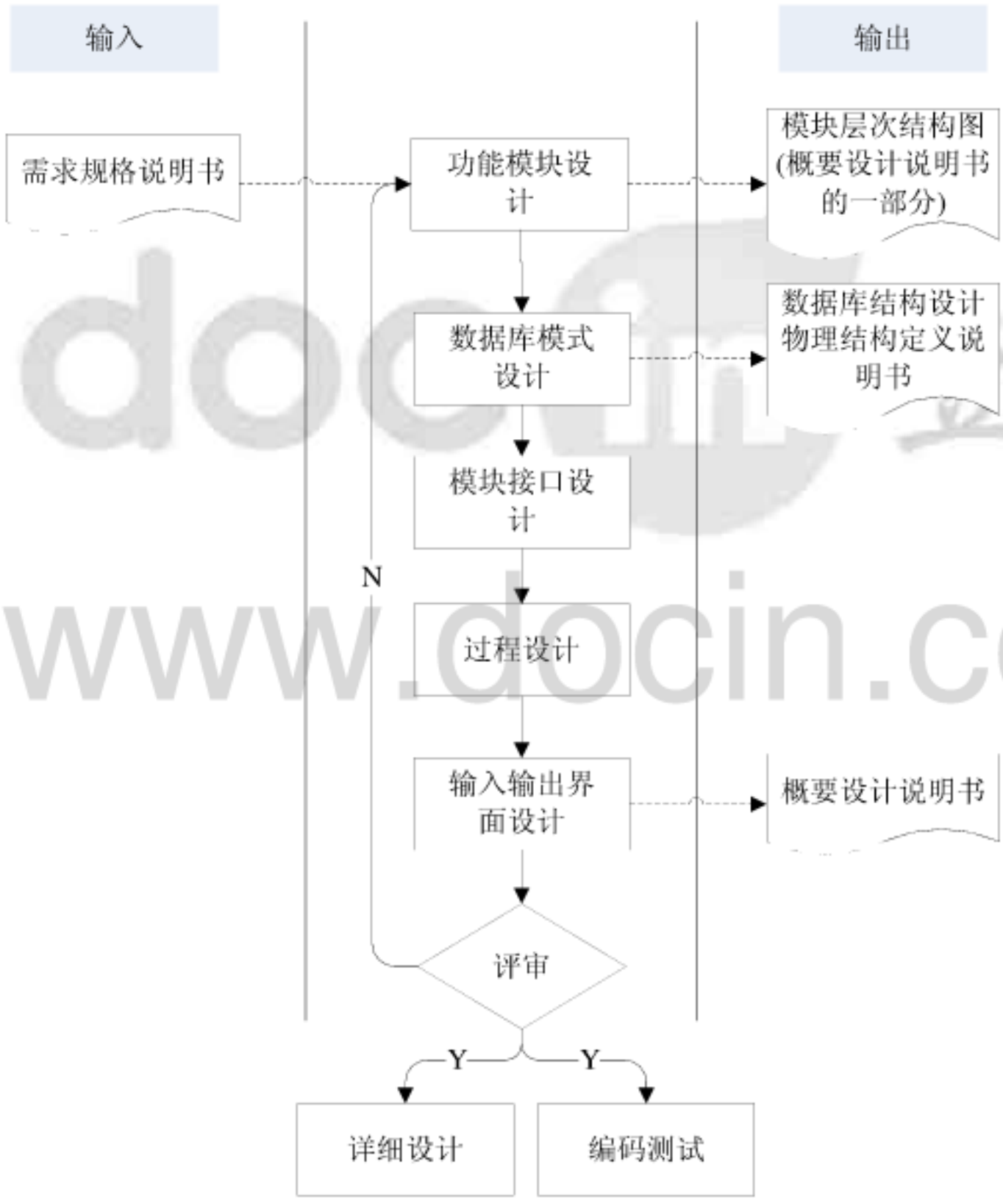
- ✧ Command（命令），将一个请求封装为一个对象，从而使你可用不同的请求对客户进行参数化；对请求排队或记录请求日志，以及支持可取消的操作。
- ✧ Interpreter（解释器），给定一个语言，定义它的文法的一种表示，并定义一个解释器，该解释器使用该表示来解释语言中的句子。适用于：当有一个语言需要解释执行，并且可将该语言中的句子表示为一个抽象语法树时，可使用解释器模式。
- ✧ Iterator（迭代器），提供一种方法顺序访问一个聚合对象中各个元素，而又不需暴露该对象的内部表示。适用于：访问一个聚合对象的内容而无需暴露它的内部表示；支持对聚合对象的多种遍历；为遍历不同的聚合结构提供一个统一的接口。
- ✧ Mediator（协调器），用一个中介对象来封装一系列的对象交互。中介者使各对象不需要显式地相互引用，从而使其耦合松散，而且可以独立地改变它们之间的交互。适用于：一组对象以定义良好但是复杂的方式进行通信。产生的相互依赖关系结构混乱且难以理解；一个对象引用其他很多对象并且直接与这些对象通信,导致难以复用该对象；想定制一个分布在多个类中的行为，而又不想生成太多的子类。
- ✧ Memento（备忘录），在不破坏封装性的前提下，捕获一个对象的内部状态，并在该对象之外保存这个状态。这样以后就可将该对象恢复到保存的状态。适用于：必须保存一个对象在某一个时刻的(部分)状态，这样以后需要时它才能恢复到先前的状态；如果一个用接口来让其它对象直接得到这些状态，将会暴露对象的实现细节并破坏对象的封装性。

- ✧ Observer（观察者），定义对象间的一种一对多的依赖关系，以便当一个对象的状态发生改变时，所有依赖于它的对象都得到通知并自动刷新。适用于：当一个抽象模型有两个方面，其中一个方面依赖于另一方面，将这二者封装在独立的对象中以使它们可以各自独立地改变和复用；当对一个对象的改变需要同时改变其它对象，而不知道具体有多少对象有待改变；当一个对象必须通知其它对象，而它又不能假定其它对象是谁。换言之，你不希望这些对象是紧密耦合的。
- ✧ State（状态），允许一个对象在其内部状态改变时改变它的行为。对象看起来似乎修改了它所属的类。适用于：一个对象的行为取决于它的状态，并且它必须在运行时刻根据状态改变它的行为；一个操作中含有庞大的多分支的条件语句，且这些分支依赖于该对象的状态，这个状态通常用一个或多个枚举常量表示。
- ✧ Strategy（策略），定义一系列的算法,把它们一个个封装起来，并且使它们可相互替换。本模式使得算法的变化可独立于使用它的客户。
- ✧ Template Method（模板方法），定义一个操作中的算法的骨架，而将一些步骤延迟到子类中。Template Method 使得子类可以不改变一个算法的结构即可重定义该算法的某些特定步骤。适用于：一次性实现一个算法的不变的部分，并将可变的行为留给子类来实现；各子类中公共的行为应被提取出来并集中到一个公共父类中以避免代码重复；控制子类扩展。
- ✧ Visitor（访问者），表示一个作用于某对象结构中的各元素的操作。它使你可以在不改变各元素的类的前提下定义作用于这些元素的新操作。适用于：一个对象结构包含很多类对象，它们有不同的接口，而对这些对象实施一些依赖于其具体类的操作；需要对一个对象结构中的对象进行很多不同的并且不相关的操作，而想避免让这些操作"污染"这些对象的类；定义对象结构的类很少改变，但经常需要在此结构上定义新的操作。

11.4 概要设计活动

概要设计一般包含功能模块设计、数据库模式设计、模块接口、过程设计和界面设计等内容，但在开始设计之前，必须确定项目组概要设计所使用的模板、设计的要求、设计通过准则等内容。由项目经理安排概要设计人员及制定进度计划。具体流程如图 11-1 所示。

图表 11-1 系统概要设计流程图



项目经理确定参与概要设计的人员，并讨论确定阶段准出准则，通盘考虑并跟踪上游顺延下来的进度、技术难度等风险、问题，与相关人员沟通，制定阶段计划。必要时，请高层

参与问题解决。评估、确认需求分析结果，并对可重用的软件或功能模块进行测试，通盘考虑整个系统结构、功能扩展性。主要完成如下的设计工作：

1. 系统体系结构设计

- 用选定的工具和开发计划设定的交付方式（如小版本渐进交付）及设计方法，结合设计原则（如功能模块化等），将系统分解为若干子系统、功能模块，并确定子系统、功能模块及其间的关系；
- 确定子系统、功能模块间的约束、假设和依赖（如系统运行环境和开发、测试环境等，并考虑系统并发性和分布性要求），设定子系统、功能模块的优先级排序；
- 结合以上内容，对系统的模块逻辑实现和集成方法进行设计，降低使软件难于实现、测试（必要时测试人员参与讨论）、维护因素，形成高内聚、低耦合的系统体系结构，建议考虑采用三明治或变体实现和集成方法；
- 定义错误处理和恢复策略，对可能出现的故障进行分解，进行优先级排序并确定处理对策；
- 确定项目数据库设计规则以便于系统统一，其中包括：库命名，逻辑设计，物理设计，安全性设计及优化，管理规则等；
- 数据库设计一般要经过“逻辑设计→物理设计→安全性设计→优化”等步骤，通常要迭代进行。

2. 逻辑设计

逻辑设计：分析软件系统模块及其之间的数据操作，使用抽象数据类型设计，转换数据对象的属性及其关联、接口等内容，设计并完善数据字典及其约束条件，实现数据的变量封装结构设计。面向结构设计方法中为创建与数据库相关的数据流图或实体关系图；若采用面向对象方法，则分析类信息传递内容，并创建类图。

3. 物理设计

物理设计主要完成：设计表结构，与实体关系图或类图相结合；对表结构进行规范化处理。

4. 安全设计

安全性设计考虑数据库的登陆访问限制，用户密码加密，操作访问权限等系统安全设计；分析并优化数据库的“时—空”（即性能，容量等）效率，尽可能“提高处理速度”并且“降低数据占用空间”。

- 分析“时—空”效率的瓶颈，找出优化对象（目标），并确定优先级；
- 消除对象（目标）间的对抗性，必要时给出折中方案；
- 给出优化的具体措施，如逐步评估、优化数据库环境参数，对表格进行反规范化处理等，坚持信息隐蔽等原则，加强数据设计可维护性。

5. 接口设计

接口（包括用户界面）设计是指与用户、测试人员交流界面设计需求，明确用户界面、接口设计规则，包括：标准控件的使用规则，通用界面（包括主界面和子界面等）、接口设计原则等。接口设计原则：

- 扩展子系统或功能模块及其之间的关系和限制条件，实施系统所需的接口设计，并消除冗余后，完善系统的数据流图，必要时形成功能说明和操作方式。若面向对象方法，则为子系统包、类间的属性、方法等设计；
- 由测试人员参与完善测试接口设计；
- 结合系统错误处理和数据验证方法，验证接口设计结果，并逆向需求求证。

6. 界面设计

界面设计：分析需求说明中对用户界面的需求，实施用户界面设计，包括界面及其关系、工作流程等，必要时采取原型设计，并请用户或同行评估后细化改进。

7. 整合及评审

- 根据设计方法及其设计结果，项目经理负责采用指定的概要设计说明书、数据库设计说明书模板（必要时结合数据字典或类图）描述设计体系结构内容；
- 根据设计结果由文档人员完善、更新、充实用户手册（草稿）相应内容；
- 指定需求跟踪人负责跟踪系统设计结果，更新《用户需求跟踪矩阵》，若发现问题，

形成过程审计报告，提交项目经理或高层经理寻求解决方案；

- 测试人员负责对系统设计结果进行可测试性验证；
- 项目经理或用户委派专人负责组织对设计工作产品的进行评审或同行评审，按项目评审过程执行。

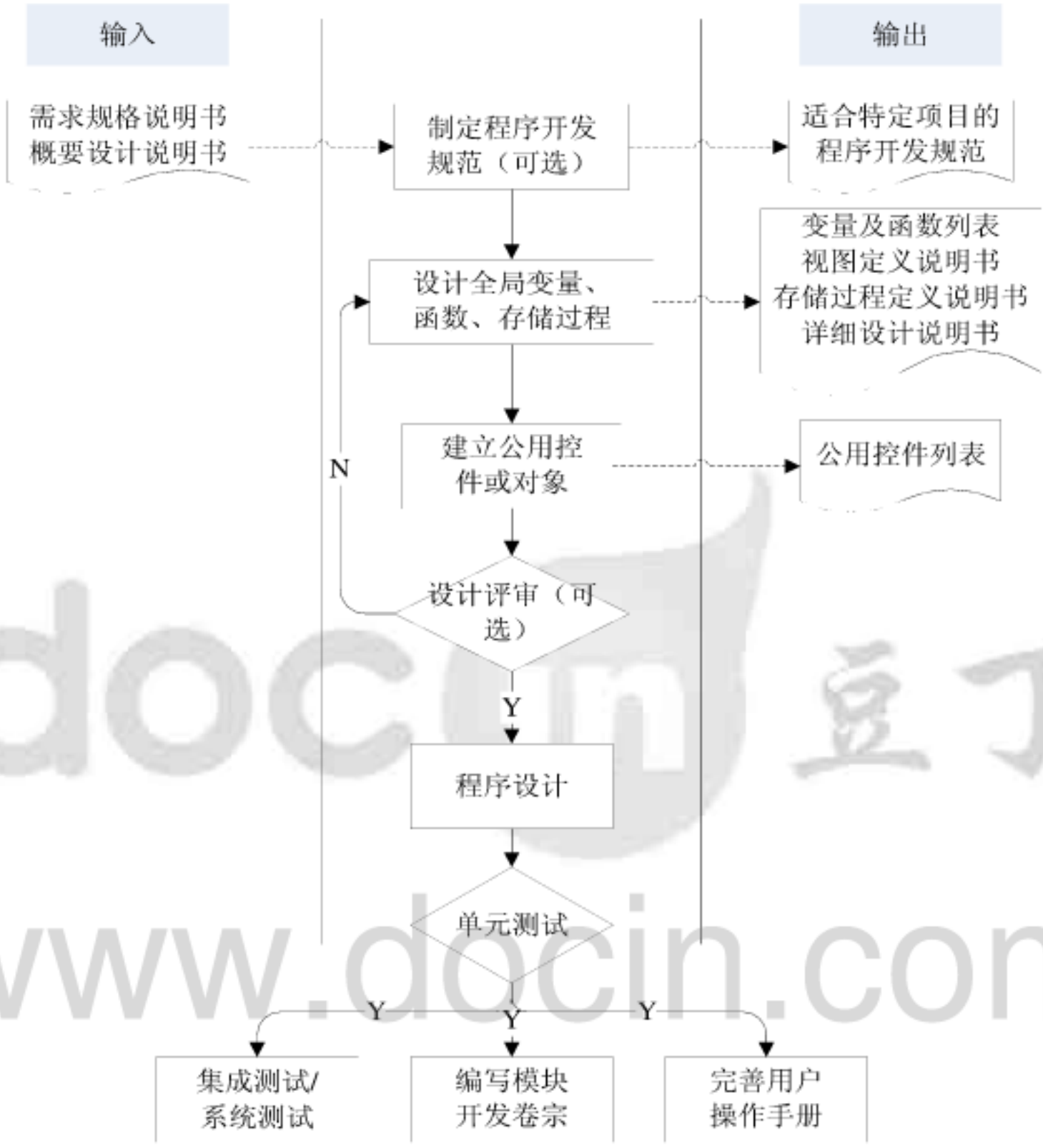
11.5 详细设计活动

详细设计的根本目标是确定应该怎么具体地实现所开发的系统，应当对目标系统的实现方法、算法进行精确描述，从而在编码阶段可以把这个描述直接编写成用某种程序语言书写的程序。因此，详细设计的结果基本上决定了最终程序代码的质量。详细设计不仅仅是逻辑上正确地实现每个模块的功能，更重要的是设计出的处理过程应该尽可能简明易懂。在开发过程中，详细设计的活动流程如图 11-2（见下页）所示。

具体执行步骤为：

- 项目经理确定详细设计人员，并通盘考虑上游顺延下来的进度、技术难度风险、问题，制定阶段工作计划，确定阶段准出准则；
- 项目经理配合详细设计人员对概要设计方案进行评估，项目组间或组内达成共识；
- 结合设计方法、工具、需求文档和软件系统体系结构设计文档，逐步细化设计每个功能模块的主要接口与属性，必要时还须细化每个用户界面；若采用面向对象方法，则为设计类的函数和成员变量，并明确对象之间的相互关系；
- 细化设计每个功能模块的数据结构与算法（若存在的话），并提高其效率，确认并完善重用软件及模块单元的算法和处理流程，确保系统一致性；
- 处理数据流程并充分考虑系统限制，逐步完善系统集成方案；
- 指定需求跟踪负责人对需求状态进行跟踪，完善需求跟踪矩阵，若发现问题，形成过程审计报告，提交项目经理或高层经理寻求解决方案；
- 重复执行以上步骤直到达到准出准则；

图表 11-2 系统详细设计流程图



■ 整合及评审

✧ 项目经理负责或指定专人负责组织整合设计内容，编制详细设计说明书，指定专人完善用户手册；

✧ 项目经理负责组织对阶段工作产品的验证和评审，按项目评审过程执行。

11.6 设计方法简介

对大多数软件系统而言，60%以上的软件费用都用于软件维护，因此，优秀软件设计的一个主要特点就是容易维护。所谓优秀设计，就是权衡了各种因素，从而使得系统在其整个生命周期中的总开销最小的设计。

11.6.1 面向结构（数据流）设计方法

结构程序设计的概念最早由 E.W.Dijkstra 提出。1965 年他在一次会议上指出："可以从高级语言中取消 GO TO 语句","程序的质量与程序中所包含的 GO TO 语句的数量成反比"。1966 年 Böhm 和 Jacopini 证明了，只用 3 种基本的控制结构就能实现任何单入口单出口的程序。这 3 种基本的控制结构是"顺利"、"选择"和"循环"。从而引起大家对程序设计思想、方法和风格的争论，最终认为，应当创立一种新的程序设计思想、方法和风格，以显著地提高软件生产效率和降低软件维护代价。1972 年 IBM 公司的 Mills 进一步提出，程序应该只有一个入口和一个出口，从而补充了结构程序设计的规则。[12]

1971 年 IBM 在两个项目中使用了结构程序设计，代码行数分别为 8300 行和 40 万行。而且在设计过程中用户需求又曾有过很多改变，但是这两个项目都按时、高质量完成。并且表明，软件生产率比以前提高了一倍，结构程序设计技术成功地接受了实践的检验。

在软件开发过程中，使用结构化设计方法时，可以按如下讲解的步骤来实施，分别完成系统的概要设计和详细设计。

结合需求规格说明及系统不同层次的数据流图并利用最高输入\输入抽象点，把整个系统分解为模块并确定模块功能，确定每个功能模块的输入、转换和输出数据流，并考虑各方面数据接口、存储方式等。

迭代逐步分解各系统模块，直到确定每个功能模块只执行一个行为为止，同时细化数据结构化设计，完善数据字典以确保系统能获得所需的输出结果，从而完成系统构架设计。

[12] 引自《软件工程导论》（第四版），张海藩编著，清华大学出版社。

在详细设计的时候，对每个功能模块中的函数进行详细的算法设计，理想情况下应当达到伪代码的程度，以方便开发人员在详细设计的基础之上进行编码。

11.6.2 面向对象设计方法

使用面向对象的方法进行设计时，有几条准则可以做为指导方针。一是模块化；二是抽象，实际上类就是一种抽象数据类型；三是信息隐藏，实现对象的封装性，分离了接口与实现；四是松耦合，对象之间紧密程序尽量降低；五是强内聚，类内部属性和方法应该是高内聚的，一个类应该只有一个用途；六是可重用，从设计阶段就应当考虑。

结合需求规格说明（包括用例模型、类模型）和系统选用的编程语言，信息隐藏、责任驱动的设计原则，确定用户类（包括子类）及类的各种行为，并使用设计模型及相应的列表，确定每个对象的客户（模块）关联，并给予必要的方法设计。

迭代逐步检查对象及其方法，直到系统中每个对象拥有所有必需的方法，从而完成系统构架设计。给出详细的类图，包括 Actor 与类及各类之间的关系图、类的属性、方法等。

详细设计是来设计类的每个方法及决定他们实现什么；详细设计与具体实现语言必须关联。

在进行详细设计时，应当注意把整个系统拆分为若干个子系统来进行。详细设计的文档，可以使用伪代码方式来编写，必须注意保持与整体技术框架的一致性，注意检查与概要设计内容的一致性。

第12章 软件测试简介

容提要：

- 软件测试基本概念
- 软件测试分类
- 自动化测试
- 常见测试工具
- **BUG** 管理流程

软件的质量很重要，当前大家都对此有了一定的认识。但是，怎样才能保证软件的质量呢？雇佣高水平的程序员能解决此问题吗？答案是否定的。不过，只要条件允许，设置专门的软件测试部门和机制对保证软件质量是十分必要的，通过改进软件开发过程及加强软件测试对提高软件质量是很重要的。

软件测试有自己一套完整的、严格的理论与实践体系，测试人员工作重点与程序员不一样，对他们的技能要求及培养也不一样。在某种意义上来说，测试人员除了受了软件测试相关培训之外，还应当有更广的知识技能要求，对所要测试的软件结构、功能、要求及应用领域有深入的认识。对于软件测试，我们应当清醒地认识到，测试只能证明软件有错，而不能保证软件程序没错。

1983 年 IEEE 给软件测试给出的定义为：使用人工或自动的手段来运行或测定某个软件系统的过程，其目的在于检验它是否满足规定的需求或弄清预期结果与实际结果之间的差别。是帮助识别开发完成（中间或最终的版本）的计算机软件（整体或部分）的正确度(Correctness) 、完全度(Completeness)和质量(Quality)的软件过程；是 SQA(Software Quality Assurance，软件

质量保证)的重要子域。简单的来说，软件测试是为了发现程序中的错误而执行的过程[13]。

12.1 软件测试基本概念

12.1.1 软件测试背景[14]

1947 年，哈佛大学制造的 Mark II，需要大批程序员定期维护，某日运行过程中突然停止了工作，大家爬上去找原因，把其腹内的一组继电器接通后，可以开始工作了。缺陷（BUG）产生了，然后被消灭了。

软件测试是伴随着软件的产生而产生的。早期的软件开发过程中，测试的含义比较狭窄，将测试等同于“调试”，目的是纠正软件中已经知道的故障，常常由开发人员自己完成这部分的工作。对测试的投入极少，测试介入也晚，常常是等到形成代码，产品已经基本完成时才进行测试。

直到 1957 年，软件测试才开始与调试区别开来，作为一种发现软件缺陷的活动。由于一直存在着“为了让我们看到产品在工作，就得将测试工作往后推一点”的思想，测试仍然是后于开发的活动。潜意识里，我们的目的是使自己确信产品能工作。所以，20 世纪六十年代，在软件工程理论建立之前，大家对测试的理解是：为表明程序正确而进行测试。

1972 年，在美国北卡罗来纳大学举行了首届软件测试正式会议。1979 年，Glenford Myers 的《软件测试艺术》（The Art of Software Testing）中作出了当时最好的软件测试定义：“测试是为发现错误而执行的一个程序或者系统的过程。”

1980 年,在美国俄勒冈计算机会议上软件测试被正式确认为软件工程的一部分。1981 年，Bill Hetzel 开设“Structured Software Testing”公共课。1982 年，在美国北卡罗来纳大学召开首次软件测试的正式会议。1983 年，Bill Hetzel 在《软件测试完全指南》（Complete Guide of Software Testing）一书中指出：“测试是以评价一个程序或者系统属性为目标的任何一种活动。测试是对软件质量的度量。”Myers 和 Hetzel 的定义至今仍被引用。

---

[13] 引自于百度百科，http://baike.baidu.com/

[14] 本节内容使用互联网相关资料整理编写得到。

1988 年，David Gelperin & Bill Hetzel 在“Communications of the ACM”发表“The Growth of Software Testing”，介绍系统化的测试和评估流程。上世纪 90 年代，测试工具终于盛行起来。人们普遍意识到，工具不仅仅是有用的，而且要对今天的软件系统进行充分的测试，工具是必不可少的。1996 年提出测试能力成熟度 TCMM(Testing Capability Maturity Model)，测试支持度 TSM（Testability Support Model)，测试成熟度 TMM（Testing Maturity Model)。

2002 年，Rick 和 Stefan 在《系统的软件测试》（Systematic Software Testing）中对软件测试做了进一步定义：“测试是为了度量和提高被测软件的质量，对测试件进行工程设计、实施和维护的整个生命周期过程。”

近 20 年来，随着计算机和软件技术的飞速发展，软件测试技术研究也取得了很大的突破。测试专家总结了很好的测试模型，比如著名的 V 模型、W 模型等，在测试过程改进方面提出了 TMM（Testing Maturity Model）的概念，在单元测试、自动化测试、负载压力测试以及测试管理等方面涌现了大量优秀的软件测试工具。

12.1.2 软件测试著名案例[15]

狮子王案例：1994 年的秋天，Disney 为孩子们发布了它的第一个多媒体 CD-ROM 游戏——狮子王动画书。虽然在这个市场上，已经有公司涉足多年，可是狮子王却是 Disney 进入这个市场的第一次尝试。为了赢得市场，Disney 不惜美元为这款产品做广告和推广活动。这些活动无疑是成功的，这款游戏成了那个假期孩子们必买的游戏。然而，12 月 26 号，圣诞节的第二天，Disney 的客户支持电话开始响个不停，很快，电话支持人员就淹没在了家长们的抱怨声中，因为他们的孩子无法运行游戏而开始哭个不停。大量的报道接着出现在了报纸和电视新闻上。事后的调查发现，Disney 没有在市场上卖的众多 PC 平台上去测试这款软件。软件只能在类似开发人员开发软件的平台上运行，而不是大多数普通人所使用的。现在我们很明确的知道了，Disney 没有做配置测试。

Intel 浮点除法软件缺陷：1994 年，英特尔发布的一批奔腾中央处理器，存在如下问题，

[15] 本节的案例均来自于互联网资料，并由作者整理编辑而成。

在计算器中输入(4195835/3145727)×3145727-4195835，得到的计算结果不为零。这个软件缺陷被刻录在 CPU 中，并在生产过程中反复制造，问题是 Intel 的测试人员在实验室时发现了该缺陷，只是没有在发布之前得到重视并修改。最终，Intel 道歉并拿出 4 亿美元招回更换芯片，并产生了很大的反面影响。

美国航天局火星登陆：1999 年美国“极地登陆号”登陆后毁坏，通过分析认定出现错误动作的原因极有可能是某一个数据位被意外更改。原因是，登陆器经过了多个小组测试，其中一个小组测试脚落地过程，另一个小组测试此后的着陆过程。前一个小组不去注意着地数据位是否置位，这不是他们负责的范围；后一个小组总是在开始测试之前重置计算机、清除数据位。双方独立工作都很好，但从未集成在一起过，从而导致在实际环境下工作时出错。

爱国者导弹防御系统：美国爱国者导弹防御系统是里根总统星球大战计划的一部分。第一次使用这个系统是在海湾战争中（1991 年），用来对付伊拉克的飞毛腿导弹。虽然有很多吹捧这个系统成功的故事，它却没能防住所有的导弹，包括在沙特多哈炸死 28 名美军的一枚。专家发现问题出现在软件错误上，系统时钟一个很小的记时错误经过了 14 个小时的积累，最终导致了跟踪系统不再精确，在多哈的那次袭击中，系统已经运行了 100 小时。

12.1.3 软件缺陷

我们可以把所有的软件问题统称为软件缺陷，但为了能更严格的定义软件缺陷，避免开发组内产生不同的认识，可以从以下 5 点来定义软件缺陷：

- 软件未达到产品说明书（简称，SPEC）标明的功能；[16]
- 软件出现了产品说明书指明不会出现的错误；
- 软件功能超出产品说明书指明范围；
- 软件未达到产品说明书虽未指出但应达到的目标，此条的目的是抓住产品说明书上遗漏之处；
- 软件测试员认为软件难以理解、不易使用、运行速度缓慢，或者最终用户认为不好。

[16] 此定义参照《软件测试》一书，Ron Patton 著，周予滨、姚静译，机械工作出版社，2006 年 第 1 版

【说明：我们可以把这里的产品说明书理解为包含：《用户需求说明书》、《软件需求规格说明书》等在内的各类资料。只要是项目组共同评审或协定通过，对开发的产品进行定义和描述的文档、资料、口头约定（不建议养成口头约定的习惯）等均可以称之为 SPEC，并不是单指某一个《产品说明书》。】

在软件开发过程中必定会产生缺陷，而且缺陷不可被全部消除。产生软件缺陷的原因很多，一般可以归纳为以下几点：

- 软件模型或者说业务建模制定不正确，更直观的理解是，SEPC 本身不明确或有错误，没有能很好的描述要开发的软件，这类原因占了 70%左右，并且很难于纠正；
- 软件庞大，功能十分复杂；
- 编程过程出错，此类原因导致的错误大概占 20%，一般来说比较容易纠正；
- 个别功能要求改变而影响到其他部分；
- 与要开产的软件对接的第三方软件有缺陷；
- 人为因素，常见的因素包括：项目组管理方法、项目进度要求时间紧、项目组配备人力不足、组内及组外沟通不充分等几种情况。

修正不同阶段产生的缺陷，需要投入的费用是不要一样的，一般来说同样的一个缺陷越到软件开发的后期，修正所花费的费用越大，下表给出了一个不同阶段修正同一个缺陷所花费用比率：

图表 12-1Bug 发现阶段修正花费对照表[17]

	纠错阶段	单位费用
1	功能需求搜集分析/软件设计阶段	1 单位费用
2	编程或分块测试阶段	5 单位费用
3	整体或系统测试阶段	10 单位费用
4	早期用户试用或 Beta 测试阶段	15 单位费用
5	软件推出市场后	30 单位费用

[17] 引自互联网资料。

12.1.4 软件测试的原则[18]

为了能够更好的进行软件测试，提高测试的整体效率，降低项目的整体成本，我们在执行软件测试过程中可以参照以下几点原则：

- 完全测试程序是不可能的，不可能找出软件的所有缺陷，这是因为：
  - ✧ 输入量太大
  - ✧ 输出结果太多
  - ✧ 软件实现途径太多
  - ✧ 软件说明书没有客观标准，从不同的角度来看，软件缺陷的标准不同。
- 软件测试是有风险的行为，如果决定不去测试所有的情况，那就是选择了风险。软件测试人员要学会的一个主要原则是如何把无边无际的可能减少到可以控制的范围，以及如何针对风险制订作出明智抉择，去粗存精。
- 测试无法显示潜伏的软件缺陷，软件测试工作与防疫员的工作极为相似，可以报告已发现的软件缺陷，却无法报告潜伏的软件缺陷，更不可能保证找到全部的缺陷。
- 找到的软件缺陷越多，就说明软件缺陷越多。生活中的寄生虫和软件缺陷几乎完全一样，两者都成群出现。发现一个附近就会有一群，这是因为：
  - ✧ 程序员怠倦，第一天编写代码还不错，第二天就会烦燥不安了，那么一个软件缺陷很可能会表明附近有更多的软件缺陷。
  - ✧ 程序员往往会犯同样的错误，每个程序员都有自己的偏好及编码习惯。
  - ✧ 某些软件缺陷是大灾难的征兆，一开始某些缺陷似乎毫无关联，但其有可能是由一个极其严重的原因造成的。
- 杀虫剂怪事，与农药杀虫是一样的，软件对测试方法及技术也有免疫力，只有发明新的杀虫剂（测试技术或方法）去找虫子。
- 并非所有软件缺陷都能修复，主要原因为：

[18]本节内容引自于《软件测试》一书，Ron Patton 著，周予滨、姚静译，机械工作出版社，2006 年 第 1 版

- ✧ 没有足够的时间，所有项目都有工期的要求，而又不可能投入足够多的开发及测试人员；
- ✧ 不算真正的软件缺陷，有些缺陷可能是由于错误的理解、测试错误或说明书变更等引起的，有时也会把缺陷当作附加功能来对待；
- ✧ 修复的风险太大，在紧迫的产品发布进度压力下，修改软件将冒很大的风险，可能会导致更多的软件缺陷出现。不去理睬未知软件缺陷，以避免出现未知新缺陷的做法也许是安全之道；
- ✧ 不值得修复，有时不常出现的软件缺陷和在不常用功能中出现的软件缺陷可以放过。

■ 难以说清的软件缺陷，因为开发小组使用的最佳工作方式千差万别，大家对缺陷的理解也不一致。

■ 产品说明书不断变化，整个行业变化太快，同时软件变得更庞大、更复杂，功能越来越多，这些都会导致用户描述和定义软件的产品说明书一变再变。

■ 软件测试员在小组中不受欢迎，软件测试员的任务是检查和批评同事的工作，挑毛病，公布发现的问题。为了保持小组成员和睦，可以采纳以下建议：

- ✧ 早点找出软件缺陷；
- ✧ 控制情绪，试想，你给别人编写的程序找出了很多缺陷，如果自己情绪再不能得到很好的控制，那么很容易与程序员产生对生，这样即对项目组不利，对测试出的缺陷修复也不利；
- ✧ 给程序员，不要总是报告坏消息；

■ 软件测试是一项讲究条理的技术专业，当前软件行业已经发展到强制使用专业软件测试员的阶段了，因为生产低劣软件的代价太高。虽然并不是所有的软件开发公司如此来做，但大多数软件都采用井然有序的开发方式，把软件测试员当作必不可少的核心小组成员。

12.1.5 软件的版本

在整个软件开发的生命周期中，可能会出现各种版，每个公司对版本的定义也不一样，通常情况下有以下的几个版本是比较通用的。

- Alpha 版——公司内部测试的版本，该版本的特征为：
  - ✧ 软件的所有功能已基本实现
  - ✧ 所有的功能已通过测试，一般情况下推向市场前不再增减（一般为集成测试）
  - ✧ 已到的缺陷中，严重级别的已修正并通过复测
  - ✧ 软件性能测试可提供基本数据
- Beta 版——对外发布公测，该版本的特征为：
  - ✧ 次严重缺陷基本完成修正并通过复测
  - ✧ 完成测试计划中的每一项具体测试（一般为系统测试计划）
  - ✧ 一段时间内缺陷的发现离低于修正率
  - ✧ 所有相关文件（用户指南、软件说明、版本说明等）得到最后修正
- 发布版——正式发布版本，一般在 Beta3 之后软件正式发布，该版本的特征为：
  - ✧ 缺陷发现率低于修正率，此距离逐渐拉开并一直保持稳定的一段时间
  - ✧ 测试部门对所有已修正的缺陷重新测试并通过
  - ✧ 技术支持部门对产品的提出认为可行
  - ✧ 所有用户反馈都已妥善处理
  - ✧ 所有文件准备就绪
  - ✧ 得到测试部门认可

【说明：除此之外，大家还会常听说 CTP 版本，即 Community Test Preview 社区测试预览版，可以理解为处于 Alpha 版与 Beta 版之间的一个版本，主要是供各类技术社区里的爱好者使用，并提供反馈意见的一个版本。RC 版本，即 Release Candidate 发行候选版。RTM 版本，即 Release to Manufacturing 零售版。RTM 将紧随最后一个 RC 版本出现并将代码最终发布给客户。】

12.1.6 优秀软件测试员必备

要想成为一名优秀的软件测试员需要艰苦的过程，应当具备的一个基本素质是：打破沙锅问到底，一般可以下几方面去努力：

- 探索精神，软件测试员不会害怕进入陌生环境。有较强的学习能力，可以用最快的速度成为一个新的行业的专家。
- 故障排除能手，软件测试员善于发现问题的症结，喜欢猜谜。可以迅速的通过事物的表面现象发现事物的本质，能够从琐碎的现象中发现内部的联系和规律。
- 不懈努力，软件测试员总是不停尝试。他们可能会碰到转瞬即逝或者难以重建的软件缺陷；他们不会心存侥幸，而是尽一切可能去寻找。只要出现过的缺陷，就说明一定是存在的，找不到只能说明没有能够真的重新当时的环境和全部的操作细节。测试人员要能够敏感的察觉到细微的变化，并立即开始在大脑中努力重现之前的整个场景。把残存的瞬间记忆整理在纸上，通过分析，把这些碎片整理起来，最终找到缺陷重现的场景和规律。牢记：在做这样的事情之前给自己制定一个规则，例如只花费 N 多时间来努力重现这个缺陷，如果超过这个时限还没有找到，那么就把当前的工作整理成一份文档保留下来，然后去按计划继续进行下面的工作，直到再次“偶遇”这个缺陷。
- 创造性，测试显而易见的事实，那不是软件测试员；他们的工作是想出富有创意甚至超常的手段来寻找软件缺陷。虽然创造性是必需的，但是还是更建议把大多数时间放在熟悉真实用户的工作上，测试的基础是现实中已经存在的场景，在冥思苦想新的场景的时候，先同用户沟通一下，试图发现一些新的场景效率会更高一些。有很多事实并不是那么显而易见。
- 追求完美，他们力求完美，但是知道某些无法企及时，不去苛求，而是尽力接近目标。做任何事情都应当有一个策略，分配给每项任务一个指标或者一部分资源（也就是说如果这件事情成功，那么它带来的收益值得我们付出的最大成本），当这部分

资源耗尽时，就停止这项任务。

- 判断准确，软件测试员要决定测试内容、测试时间，以及看到的问题是否算作真正的缺陷。要不断的提高自己的专业素养，除了行业知识、测试专业知识以外，还要尽可能的去学习一些软件行业的基础知识，例如操作系统、数据库、程序设计开发、计算机网络等。
- 老练稳重，软件测试员不害怕坏消息，必须告诉程序员，你的孩子很丑，知道和不够冷静的程序员怎样合作。
- 表达能力，软件测试员要善于表达观点，表明软件缺陷为何必须修复，并通过实际演示力陈观点。测试工作开展的好坏，很大程度上就靠沟通能力和展示自己工作的能力了。
- 在编程方面受过教育。一个有过开发经历的测试人员，对系统的领悟能力和学习速度同没有开发经历的测试人员是截然不同的。

12.2 软件测试分类

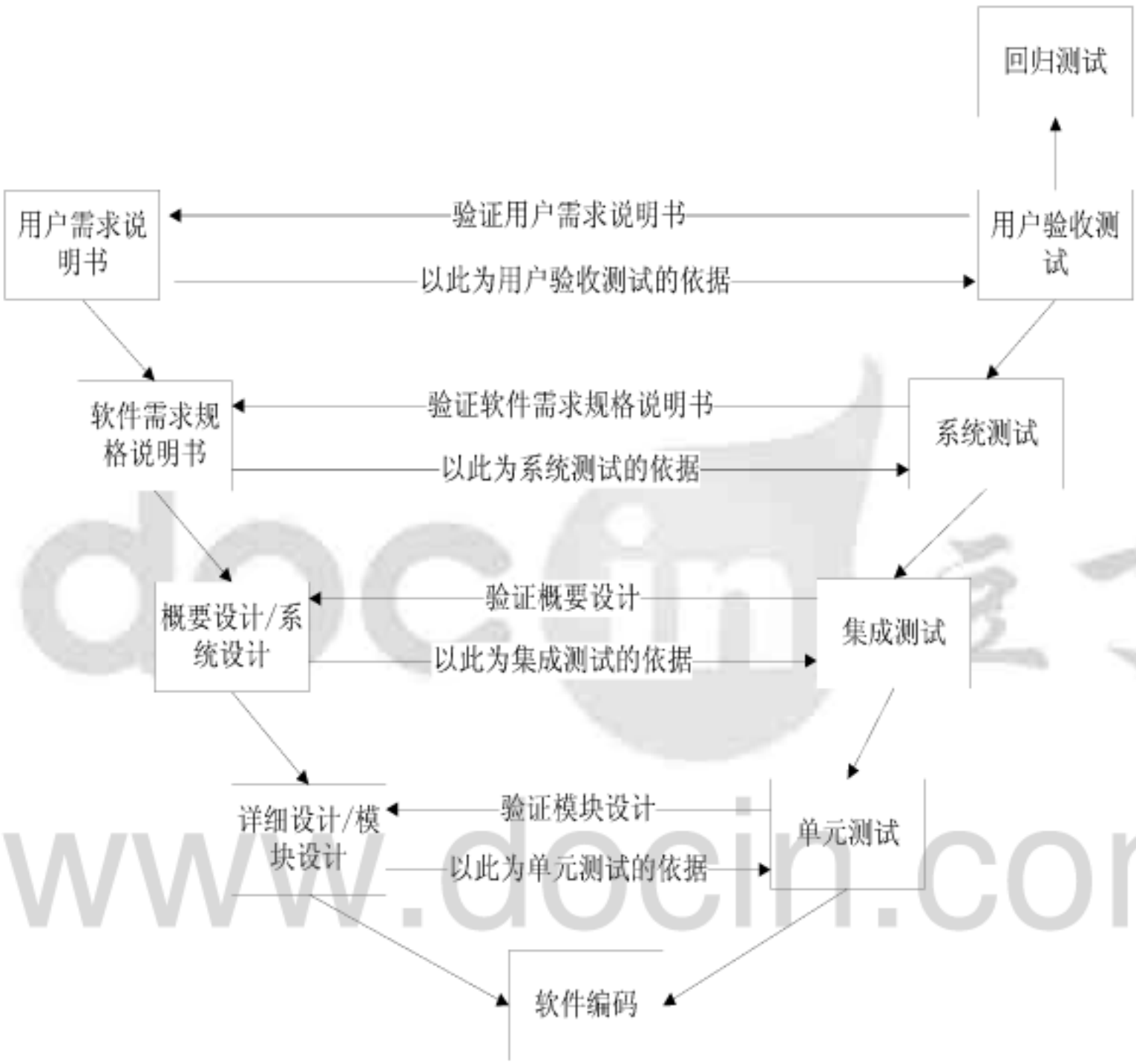
按软件测试特性可以把软件测试分为白盒测试、灰盒测试和黑盒测试三种，其特征及包含的内容如下：

- 白盒测试——测试人员直接在软件的源程序上进行测试、修改、复测。要求测试工程师对软件的内部结构及逻辑有深入的了解，并掌握写成该源程序的语言。分为：语句测试；分支测试；路径测试；条件测试；目测
- 灰盒测试——介于白、黑两者之间，是两者的结合。测试工程师对软件程序结构有一定了解，但了解的程度又不需要达到白盒测试的深度。
- 黑盒测试——测试人员不必深入了解软件的内部设计，只是从一个终端用户的角度，根据产品说明书的指标，从外部测试软件的各项功能及性能。黑盒测试主要是功能测试。

按软件开发过程可以把软件测试分为单元测试、集成测试、系统测试、用户验收测试以

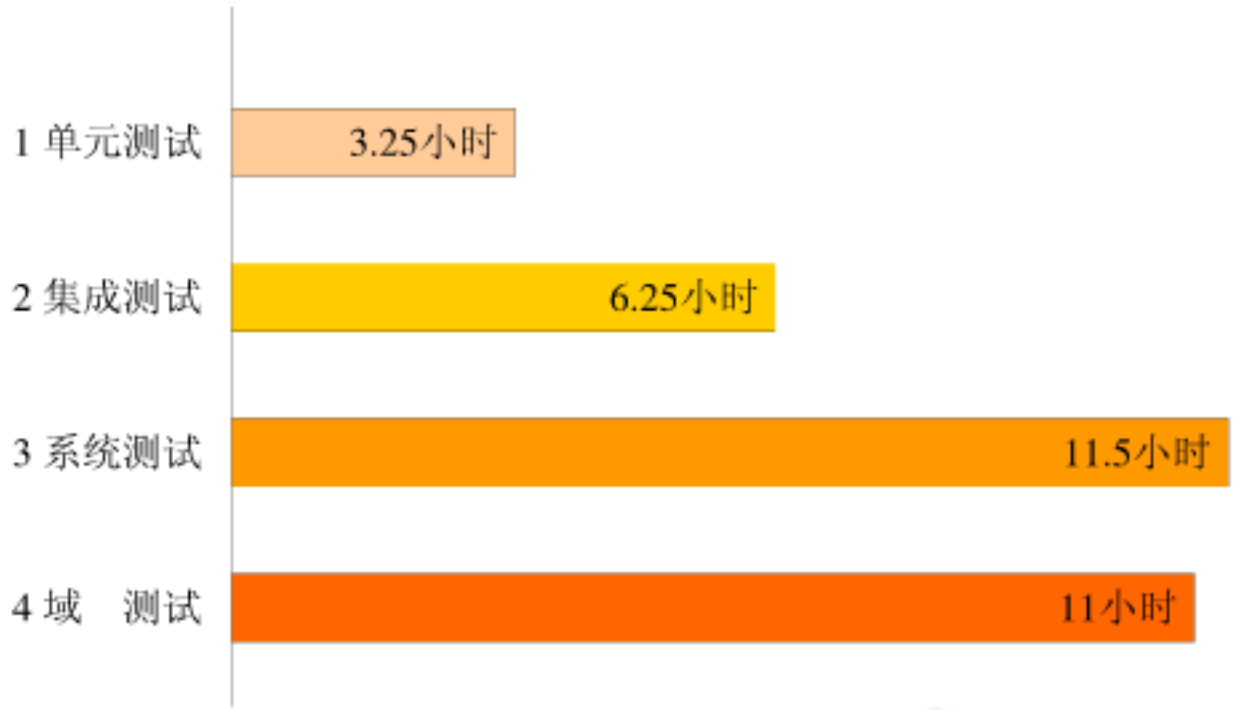
及回归测试。回归测试一般是在缺陷修改之后执行，保证原缺陷不在重现，并且缺陷的修改不影响其他功能。此分类一般可以使用 V 模型来表示，如下图所示：

图表 12-2 软件测试 V 模型图



用此种方法分类时，个阶段的用时差别比较大的，根据国外的统计数据，每功能点（此概念请参见第 7 章）测试用时如下图所示，供大家参考：

图表 12-3 各类测试用时表



【说明：域测试可以理解为交付测试，意思是在软件投入使用以后，针对某个领域所作的所有测试活动，对应与这里的用户验收测试。】

按软件测试要求可以把软件测试分为基本功能测试、全面测试和基准测试。按此方法分类的各种测试解释如下：

- 基本功能测试（Smoke test）：只对软件的关键功能做测试，而不必卷入细致的测试，不必面面俱到。
- 全面测试（Sanity test）：不仅对软件关键功能测试，还要覆盖软件的全部功能，是回归测试的主要组成部分。
- 基准测试（Benchmark test）：对指定的一个或一组程序及数据在不同的计算机上执行测试，以测定其在标准情况下、特定配置下的工作性能，并将其执行速度、完成需时等加以比较。

按软件特性可以把软件测试分为功能测试和非功能测试：

- 功能测试主要包括：等价区间测试，把输入空间划分几个“等价区间”，在每个区间中只需要测试一个典型值即可；边界值测试；随机测试；状态转换测试；流程测试等。
- 非功能测试主要包括：安装/卸载测试；使用性测试；恢复测试；兼容性测试；安全测试；性能测试；强度/压力测试；容量测试；任意测试等。

12.3 自动化测试

一般认为就是使用（自动化测试）工具来进行的测试叫自动化测试，一般不需要人干预。自动化测试有如下优点：

- 一旦积累了一套自动化测试的程序，日后自动化测试节省大量的时间和资源；
- 没有时间限制——一般安排在下班后；
- 可以反复执行；
- 保证测试执行过程的一致性及准确性；
- 有较高的功能测试覆盖率；
- 模拟操作，进行压力测试，这是手测很难实现的。

自动化测试并不能完全取代手测，与任何事物一样，自动测试也有它的不完美之处，其缺点也是显而易见的：

- 并非所有的测试都可用自动测试来实现，比如使用性测试、兼容性测试等；
- 没有创造性，只能安排设计好的用例去测，碰到新问题不会应变；
- 受具体项目资源限制：受时间及人力的限制，因为自动化测试编程很费时；受资金预算的限制，商用测试软件价格比较高；对测试工程师要求比较高。

综上所述，自动化测试与手测各有优缺点，应该是互补、并存的。根据自动化测试的特点，建议以下测试可以优先考虑自动测试：

- 回归测试，每次有新版本发布前都必须执行，在整个开发过程中需要多次执行，很适合编写成自动测试程序。
- 涉及大量不同数据输入的功能测试。如各种各样的边界值测试，需要大量时间去完成的网页连接测试等等。
- 用手测完成难度较大的测试，如性能测试、压力（负荷）测试、强度测试等。例如：对于一个网站，要测试 1 万个用户在某一时间内同时登录时，服务器运行是否正常及速度是否仍然可以接受，这是手测很难完成的。

为了实现软件测试自动化，首先要有一套自动化测试的软件或工具。在使用 Visual Studio 2005 进行授课时，建立使用 VSTS 2005 for Tester 版本，来支撑部分的自动化测试工作。但是，一般自动化测试可按如下步骤进行：

1. 编写测试用例；
2. 分析、分析、验证测试用例；
3. 对已有测试用例归类，编写测试自动化计划方案；
4. 编写自动化测试程序；
5. 尽量用“数据驱动”来提高测试覆盖率；
6. 将测试用例编写成自动测试程序；
7. 执行测试程序，记录并反馈 BUG；
8. 不断完善自动化测试系统或程序。

12.4 常见测试工具

以下几类我们在日常开发过程中可能经常使用，大家也不会把他称为专门的测试工具，但确实起到了辅助测试的作用。

- 查看器和监视器，各类编译器的代码调试器均可看作查看器；任何能够洞察系统，看到一般用户看不到的数据的工具，都可以称之为查看测试工具；
- 驱动程序，用于控制和操作测试软件的工具，最简单的是批处理文件；
- 仿真器，为测试工具或程序提供数据或响应软件发送的数据；
- 分析工具，电子表格软件、文件比较软件、抓屏软件和比较软件、计算器、秒表等；
- 干扰发射器和噪声发生器，模拟由于数据中断、干扰能产生的通信错误。

以下是在公司里常用的专业测试及测试管理工具：

- 窗口/网络软件用户界面测试：WinRunner、QuickTest Professional、SilkTest、Functional Tester、Test Partner、VisualTest。

- 性能测试：LoadRunner、SilkPerformer、Rational Performance Tester、QALoad。
- 软件测试管理工具：TestDirector、SilkCentral Test Manager、Rational TestManager/ClearQuest、QADirector/TrackRecord
- X 窗口软件测试：X Runner
- 自开发测试软件，适用于特定领域

12.5 BUG 管理流程

12.5.1 微软研发中的 BUG 管理[19]

微软有一个研发框架叫 MSF（微软解决方案框架），在此解决方案中，有一个小组模型，对在研发过程中的小组角色进行了分工，在我们开发过程中主要有三个角色：PM(程序规划经理)、Dev（软件开发工程师）、Tester（软件测试工程师）。在研发过程中，三者分工明确、接口清晰。

- PM 来定义需求、书写每个功能特性的设计文档(SPEC)；
- Dev 写代码来实现这些 SPEC；
- Tester 来测试 Dev 做出来的东西是否符合 PM 定义的 SPEC。

在这个过程中形成了“三权分立”，开发人员不能否决测试人员提出的 Bug，当开发人员与测试人员对 Bug 认识不一致的时候，就以 SPEC 为准来确定 BUG 是否修改；若大家就 SPEC 理解不一致，则需要 SPEC 的负责人即 PM 给出权威解释。

微软不少项目都使用完善的研发管理工具，其中 Bug 管理系统（原来叫 Raid 系统，现集成在 VSTS 中）居于核心地位。整个软件研发过程中，特别是在测试产品、修复 BUG 的中后期，团队中所有人都生活在 Raid 中。

- Tester 只要发现问题就立即新建一个 Bug 予以跟踪并指派给相关的开发小组长（Dev Leader)

[19] 整理自《程序员》杂志，2005 年合订本。

- 开发小组长会判断这个 BUG 属于某个特定的开发人员并指派他处理。
- 开发人员会根据 BUG 的详细描述信息找到问题所在，修改程序解决这个 BUG，并把 BUG 返回给当初的测试人员；或者有争议的时候，把 BUG 指派给这个需求定义者 PM，要求澄清说明。
- 测试人员在看到某个 BUG 被解决后，就去验证这个 BUG 是否真的不存在了，根据最初的发现步骤去证实问题真的解决了就关闭这个 BUG；否则，可以激活这个 BUG，返回给当初的开发人员做进一步处理。
- 当测试人员与开发人员无法达成一致意见时，由对应的 PM 出面做协调，判断这个 BUG 的严重程序，对用户可能的影响。根据产品的进度和项目资源做出评估，是否真的需要解决这个问题。
- 管理团队利用 BUG 管理系统来跟踪整个进度，单个人的工作、小组的进度、整个产品研发进度。

每月、每周、每天所有研发人员都会收到一个当前 BUG 状态的 EMAIL：每个人有多少个 BUG，前 5 名都是谁，哪个子产品、子模块中的 BUG 还处于上升阶段。

在微软的 BUG 管理或者说研发管理思想里有以下几点需要注意：

- 报告 BUG 不仅仅是测试人员的事情，团队的每个人发现问题时都会提交一个 BUG 来跟踪；
- BUG 管理系统不仅仅是跟踪软件功能方面的 BUG，其他各种问题，如需求文档的变更、界面上错别字、帮助文档的语言、某项任务指源等都可以通过此来跟踪。在 VSTS 中全部被称之为工作项。
- Everything should be tracked in VSTS（Raid）。

12.5.2 通用 BUG 管理流程

比较通用的 Bug 管理流程如下：

1. BUG 登记——测试工程师，初始；

2. 指派任务——项目经理，激活；

3. 修改 BUG——开发工程师，修改；

4. 验证——测试工程师，通过则转第五步，否则转二步，状态为再激活；

5. 关闭——测试工程师。

【说明：项目经理可以与测试人员讨论或在总工程师参与下讨论，直接把 BUG 关闭】

为了能使开发人员准确理解 Bug，对其准确描述是测试人员的基本功之一。BUG 的描述应该短小、单一、明显和通用，并且能再现。

- 短小——只解释事实和演示、描述软件缺陷必需的细节；
- 单一——每个报告只针对一个软件缺陷，切记不要把几个软件缺陷放在一起描述，这样修复人员很容易漏掉缺陷；
- 明显和通用——用简单步骤描述软件缺陷，得到修复的机会较大；
- 再现——按照预定步骤可以使软件达到缺陷再次出现相同状况。

【注意：

1. 在报告软件缺陷时不做评价，测试员和程序员之间很容易形成对立关系，BUG 报告需要不带倾向性、个人观点和煽动性。BUG 报告应当针对产品，只陈述事实。

2. 补充完善 BUG 报告，良好测试员发现并记录许多软件缺陷；优秀测试员发现并记录了大量软件缺陷之后，继续监视其修复的全过程。

3. 尽快报告缺陷，软件缺陷发现的越早，留下的修复时间就越多。】

12.5.3 BUG 的分类

- 按缺陷状态分为：

1. Open：确认提交的缺陷，等待处理

2. Rejected：不需要修复或不是缺陷

3. Resolved：缺陷被修复

4. Reopen：回测后，缺陷没有被修复

5. Closed：回测后，缺陷被修复，将其关闭

- 按缺陷严重级分为：

1. 严重：不能完全满足系统要求，基本功能未完全实现；或者危及人身安全。比如：系统崩溃、数据丢失、数据毁坏等。
2. 较严重：严重地影响系统要求或基本功能的实现，且没有更正办法（重新安装或重新启动该软件不属于更正办法）。比如：操作性错误、错误结果、遗漏功能等。
3. 一般：严重地影响系统要求或基本功能的实现，但存在合理的更正办法（重新安装或重新启动该软件不属于更正办法）。比如：小问题、错别字、UI 布局、罕见故障等。
4. 轻微：使操作者不方便或遇到麻烦，但它不影响执行工作功能或重要功能。

- 按缺陷优先级分为：

1. 高：立即解决，否则将影响进一步测试
2. 中：正常排队，在产品发布前按正常安排进行修复
3. 低：可暂缓解决，如果时间允许应该修复，不修复也能发布

【说明：在我们实训时，各个小组可以根据以上分类建议对确定自己项目的缺陷类别，并且明确自己使用的 BUG 管理工具的操作方法及流程。】

第13章 系统实现与测试过程

内容提要：

- **CMMI** 中对应实践
- 系统实现与测试过程简述
- 编码流程
- 测试流程
- 缺陷管理与改错
- 建立产品支持文档

系统实现与测试过程，不是只根据设计编写软件代码，然后再对软件代码进行测试，它是“编码、调试、测试、改错、完善”的综合过程。通常此阶段是投入人员最多、花费时间最长、工作量最大的开发阶段，项目组的人员峰值一般也是在该阶段。

对于现代软件开发而言，超级程序员或英雄式程序员不再是法宝，逐步形成了软件工厂模式的大规模软件开发。在此阶段是“人多、活多”，必须制定软件实现及集成测试的规范，让所有人员都按照规范执行，才能顺利完成实现与测试任务。

13.1 CMMI 中对应实践

本章由三部分组成，一部分是对设计的编码实现并进行单元测试，第二部分是把实现的系统集成在一起——产品集成，第三部分是对集成好的产品进行集成测试——验证，所以在 CMMI 中与之对应的实践也包括了三大部分。分别在技术解决方案（Technical Solution，简称为 TS）过程域、验证（Verification，简称 VER）过程域、产品集成（Product Integration，简称 PI）过程域。在 TS 中有一个特定目标为“实现设计”，是根据设计来实现产品组件及相

关的支持文档，主要是与该章的系统实现相对应，其包含两个具体的实践；在 VER 中，有两个特定目标与本章内容相关，分别为：“验证准备”和“验证选择的工作产品”，其中的“验证准备”还与 14 章测试用例的编写有关，我们把测试方案的制、测试计划的制定、测试用例的编写都是验证准备的相关活动。

TS 中的实践：

SG3 Implement the Product Design（实现产品设计），目的是依照设计，实现产品组件及相关的支持文件。

实现通常包括集成之前的单元测试，一般是对产品或产品组件进行单元测试之后才给其他人员去集成，去编写用户文档。

SP3.1 Implement the Design（实现设计），目的是实现产品组件设计，一旦完成设计，就需要将其实现为产品组件，其主要由以下几个工作组成：第一，使用有效的方法实现产品组件，比如，结构化编程、面向对象编程、自动代码生成、软件代码复用、应用合适的设计模式等；第二，遵循适当的标准与准则，比如：编码规范、过程及质量标准，编码时遵循模块化、明确、简单、可靠、安全、可维护等准则；第三，对选定的组件产品，进行同行审查，可以通过代码走查、测试等多种方式来实现；第四，适当时，对产品执行单元测试，这里单元测试不局限于软件，涵盖个别硬件、软件单元或先前已整合的相关组合；第五，必要时修订产品组件，在实现阶段发生了未能于设计阶段预见的问题时，就是修订产品组件时机的范例之一。

SP3.2 Develop Product Support Documentation（建立产品支持文档），目的是开发并维护用于产品安装、操作及维护的相关文档。一般会形成的文档有：最终用户使用培训教材、用户操作手册、维护手册、安装手册、在线帮助等。其主要由以下几个工作组成：第一，审查需求、设计、产品及测试结果，以确保影响安装、操作及维护等项目文件的相关议题已被界定并解决；第二，使用有效的方法，制作安装、操作及维护的文件；第三，遵循适当的文件制作标准，比如：某公司内部制订的《技术文档编制规范》等；第四，在生命周期的初期阶段就制作安装、操作及维护等文件的初始版本，以供相关的干系人评

审；第五，执行安装、操作及维护等文件的同行审查；第六，必要时修订安装、操作及维护文件，一般需求变更、设计变更、产品实现变更等都会导致安装、操作及维护文件的修订。

产品集成（PI）的目的是，从产品组件装配（编译或组装）成产品，当前集成的时候，确保产品功能并且交付产品。此过程域重点关注把产品组件集成为更复杂的产品组件或整个产品，在产品组件编译进行集成的过程中，可以采用一次性集成，也可以采用增量集成，由项目集成计划中定义的集成顺利和集成过程来确定。在产品集成中一个重要方面就是管理产品或产品组件的内部、外部接口，必须保证这些接口的兼容性。在整个项目过程中，均需要对接口管理加以注意。

SG1 Prepare for Product Integration（产品集成准备），目的是完成产品集成的准备，产品集成的准备包括建立和维护集成顺序，执行集成的环境，集成的过程等。

SP1.1 Determine Integration Sequence（确定集成顺序），确定产品组件集成顺序。通常形成如下工作产品：产品集成顺序说明，选择或拒绝集成顺序的基本原则。可通过如下几步完成该实践：一是识别要集成的产品组件；二是识别在产品集成过程中需要执行的验证工作；三是识别产品集成顺序的候选方案；四是选择最合适的集成顺序；四是定期审查产品集成顺序，并且根据需要进行修订；六是记录制定或延缓决策的基本原则。

SP1.2 Establish the Product Integration Environment（建立产品集成环境），为了支持产品组件的集成，建立和维护必须的环境。通常形成如下工作产品：验证过的产品集成环境，产品集成环境的支持文档。可以通过如下几步来完成该实践：一是识别产品集成环境的需求；二是识别产品集成环境的验证过程及准则；三是决定必须的集成环境中哪些需要自制哪些需要购买；四是如果合适的集成环境不能得到，开发一个集成环境（比如自动编译程序开发等）；五是在整个项目过程中，维护集成环境；六是废除集成环境中不在使用的部门。

SP1.3 Establish Product Integration Procedures and Criteria（建立集成规程及准则），建立和维护产品组件集成规程及准则。产品集成规程及准则重点考虑如下：构造组件的测试

级别，接口的验证，性能偏差的阀值，组装的派生需求及其外部接口，允许替代的组件，测试环境参数，测试的成本限制，集成操作的质量和成本权衡，适当运作的可能性，交付率及其变化，从订单到交付的时间，人员的可用性，集成设施或环境的可用性。通常会产生如下工作产品：产品集成规程，产品集成准则。可以通过如下几步完成该实践：一是为产品组件建立和维护产品集成规程；二是建立和维护产品集成和评估准则；三是建立和维护集成后的产品确认和交付准则。

SG2 Ensure Interface Compatibility（确保接口的兼容性），目的是确保产品组件的内部和外部接口都是兼容的。

统计表明，许多产品集成的问题，是由未知或未控制的的内、外部接口导致的。有效的产品组件接品需求、规格和设计管理可以帮助确定实现接口的完整性及兼容性。

SP2.1 Review Interface Descriptions for Completeness（审查接口描述的完整性），审查接口描述的覆盖率和完整性。注意，接口除了包含产品组件接口之外，还包括与产品集成环境的接口。通常会产生如下工作产品：接口分类，每个分类下的接口列表，接口与产品组件及产品集成环境的对应关系。可以通过如下几步完成该实践：一是审查接口数据的完整性，并确定完全覆盖所有的接口；二是确保产品组件和接口得到标识，确保与相关交互的产品组件容易及正确连接；三是定期审查接口描述的充分性。

SP2.2 Manage Interfaces（管理接口），管理产品及产品组件内、外接口的定义、设计和变更。通常会形成如下工作产品：产品组件与外部环境关系表（比如：电力供应、固定产品和计算机总线系统），不同产品组件之间的关系表，各方产品组件同意的接口定义列表，接口控制工作组会议报告，更新接口的活动项，应用程序接口（API），更新后的接口描述和协议。可以通过如下几步完成该实践：一是在整个产品生命周期中确定接口的兼容性；二是解决冲突、不兼容和变更问题；三是维护一个项目参考者都能存取的接口数据库。

SG3 Assemble Product Components and Deliver the Product（装配产品组件并交付产品），目的是组装验证过的产品组件以及交付通过集成、验证和确认的产品。

SP3.1 Confirm Readiness of Product Components for Integration（确认用于集成的产品组件准备注绪），装配前，确认用于装配产品的每个产品组件被恰当定义，功能与其描述一致，产品组件接口遵从接品描述。通常可能会产生如下工作产品：收到的产品组件的接收文档，交付收据，检查过的产品列表，异常报告，取消（撤消）说明（通不过确认需要出具此文档）。可以通过如下几步来完成该实践：一是当产品组件为集成可用时，开始跟踪他们的状态；二是确保产品组件是按照产品集成顺序和可用的过程交付给产品集成环境；三是确认每个被正确标识产品组件的收据；四是确保每个收到的产品组件满足其描述；五是根据预期的配置来检查配置状态；六是对所有的物理接口，在产品组件连接进来之前执行前期检查。

SP3.2 Assemble Product Components（装配产品组件），根据产品集成顺利及适用的过程装配产品组件，在此活动中会产生集成后的产品或产品组件。可以通过如下几步完成该实践：一是确保产品集成环境准备就绪；二是确保集成顺序得到正确的执行；三是根据需要修订集成顺序和适用的过程。

SP3.3 Evaluate Assembled Product Components（评估已装配产品组件），评估集成后的产品组件以保证接口的兼容性。可能会产生如下工作产品：异常报告，接口评估报告，产品集成总结报告。可以通过如下几步完成该实践：一是根据产品集成顺序和适用的过程对集成后的产品组件进行评估；二是记录评估结果。

SP3.4 Package and Deliver the Product or Product Component(打包并交付产品或产品组件)，打包集成后的产品或产品组件，并交付给适当的客户。可能会形成如下工作产品：打包的产品或产品组件，交付文档。可以通过如下几步完成该实践：一是审查需求、设计、产品、验证结果和文档，确保影响打包和交付的问题得到识别和解决；二是使用有效的方法打包和交付集成后的产品；三是打包和交付产品需要满足一定的需求及标准，比如安全性、保密性等。

验证（VER）的目的是，确保选定的工作产品符合其指定的需求。验证过程域包括：验证准备、验证执行及纠正措施识别。验证及确认过程域相似，但强调不同重点，验证确保“你

把事做对了(you built it right)”，确认确保“你做了对的事(you built the right thing)”。具体讲解如下：

SG1 Prepare for Verification（准备验证），目的是确保验证措施已植入于产品及产品组件需求、设计、开发计划及进度中，并对支持工具、测试设备及软件、模拟、原型系统及设施加以定义。

验证方法包括(但不限于)检查、同行审查、审计、逐步审查、分析、模拟、测试及展示。为达到此目标，需要完成如下 3 个实践：

SP1.1 Select Work Products for Verification（选择待验证的工作产品），需要选择待验证的工作产品及每一工作产品使用的验证方法。对于软件开发来说，常见的验证方法包括：路径覆盖测试，压力、强调和性能测试，基于决策树的测试，基于功能分解的测试，测试用例重用，接收测试等。可能会产生如下工作产品：被选择进行验证的工作产品列表，对每个选定工作产品的验证方法。可以通过如下几步完成该实践：一是识别要验证的工作产品；二是识别每个选定工作产品满足的需求；三是识别可用的验证方法；四是定义每个选定工作产品的验证方法；五是提交集成到项目计划，产品得到标识、需求得到满足、选定的方法得到应用。

SP1.2 Establish the Verification Environment（建立验证环境），建立和维护支持验证所必须的环境。可以通过如下几步来完成该实践：一是识别验证环境需求；二是识别那些可以重用或修改的验证资源；三是识别验证设备和工具；四是获取验证支持设备和环境，比如测试设备和软件。

SP1.3 Establish Verification Procedures and Criteria（建立验证规程及准则），为选定的工作产品建立并维护验证的规程及准则。

SG3 **Verify Selected Work Products**（验证选定的工作产品），根据他们稳定的需求验证选定的工作产品。

SP3.1 Perform Verification（执行验证），对选定的工作产品进行验证。通常会形成如下工作产品：验证结果，验证报告，演示程序，运行过程日志。可以通过如下几步来完成该

实践：一是根据他们的需求执行选定工作产品的验证；二是记录验证活动的结果；三是识别验证工作产品导致的活动项；四是文档化运行时验证方法和在执行过程中发现的与使用的验证方法及规程背离之处。

3.2 Analyze Verification Results（分析验证结果），分析所有验证活动的结果。通过会形成如下工作产品：分析报告（比如，性能统计、非一致性原因分析），故障报告，验证方法、准则和环境的变更请求。可以通过如下几步来完成该实践：一是比较期望结果与实际结果；二是基于已建立的验证准则识别不能满足需求的产品，或者用验证方法、规程、准则和验证环境识别问题；三是分析关于缺陷的验证数据；四是把所有分析结果写进报告；五是使用验证结果比较实际度量和性能与技术性能参数；六是提供解决缺陷的信息并采取纠正措施。

13.2 系统实现与测试过程简述

在软件开发的整个过程中，系统实现与测试过程至关重要，也是我们在学习软件工程时比较容易忽视的地方，特别是针对系统实现，不少软件工程教材上都不会对其进行详细描述。由于本阶段是研发过程中投入人力最多的阶段，所以能否组织好系统的实现，对于软件产品能否按时交付及最终质量是至关重要的，这也是本章内容的重点。在此过程中，主要是达到如下几个目的：

- 实现产品组件的编码并产生相应的支持文档。
- 准备产品/系统集成，确保接口兼容性，组装产品组件。
- 同时适时对产品组件进行单元测试和集成测试，实现对产品组件及集成的产品构件的验证。

系统实现阶段开始的时机，一般会有如下的约定：系统设计阶段的《数据库设计》、《模块设计》、《用户界面设计》已完成，且通过同行评审后才开始系统实现过程。但是，对于不同的公司或不同的项目类型，此约定也不尽相同。在此过程中，项目组内各类人员及角色承担的职责如表 13-1 所示，可以由项目经理根据项目实际情况进行调整。

【注意：对于比较大的系统，可能划分多个子系统，可能实现并行开发模式，完成其中一个子系统的模块设计、用户界面设计即对该子系统的设计进行评审，然后由开发人员进行系统编码；系分人员继续对另外的子系统进行设计。】

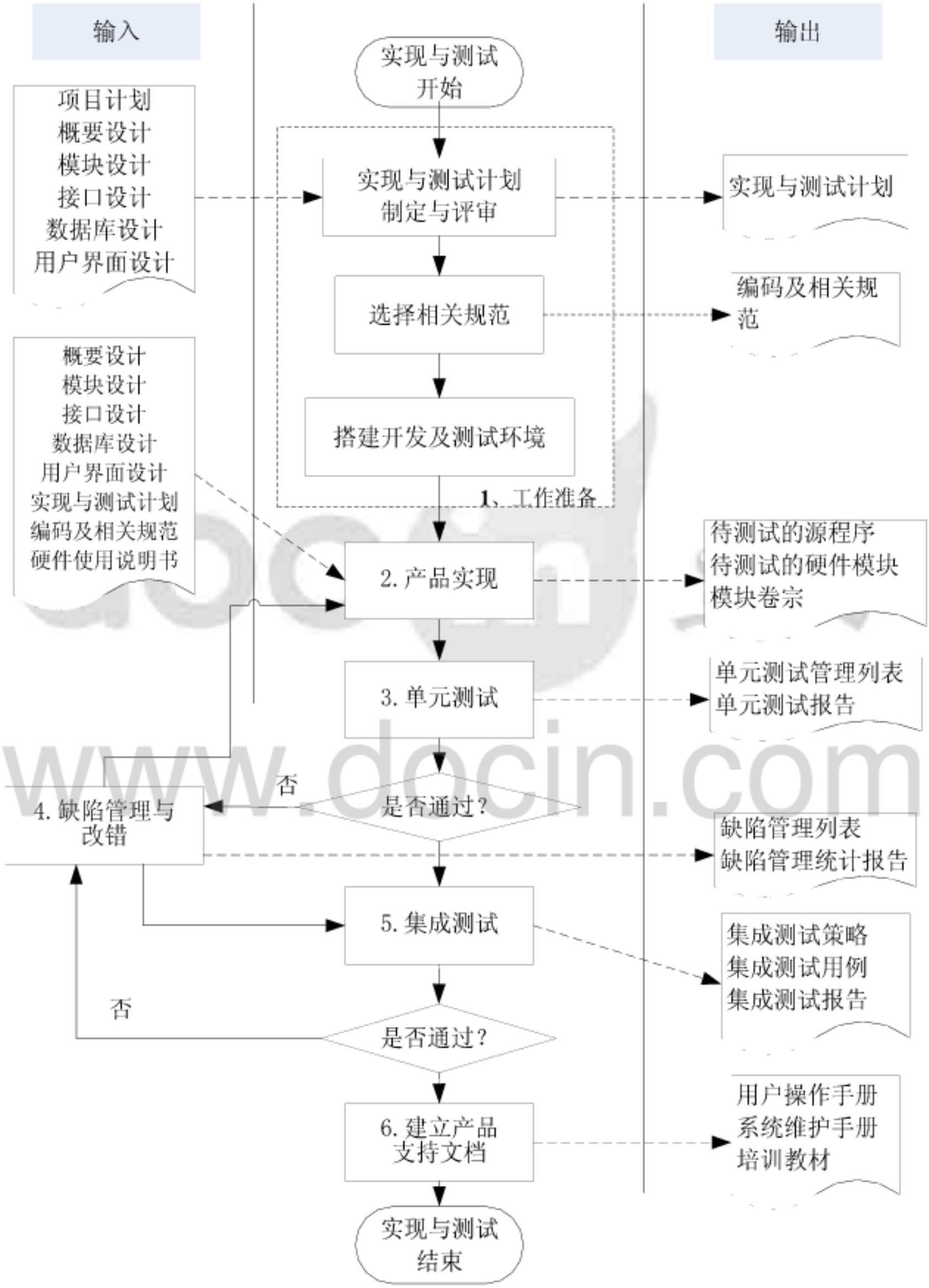
图表 13-1 实现及测试过程角色职责对照表

序号	角色	职　　责
1	项目经理	负责监督项目实现与测试过程活动，并有义务向 EPG 组提出过程改进建议
2	开发组长	管理开发相关活动，在项目经理协助下制订系统实现/编码计划。
3	开发人员	依据计划编写代码，对自己的代码进行必要的测试、调试
4	系分人员	制定项目的集成策略，根据项目组规模，也可以参与集成测试工作。
5	测试人员	依据单元测试管理列表或集成测试计划以及测试用例进行单元测试和集成测试
6	质量保证工程师	对系统实现及测试过程执行 QA 检查，并跟踪各类检查出现的问题，协助项目经理收集该阶段的数据。
7	配置管理员	把本阶段产生的各类工作产品纳入配置库，并执行配置库的日常维护。

整个实现及测试过程的活动可以分为 6 部分，分别为：准备工作、产品实现、单元测试、缺陷管理与改错、系统集成及集成测试、建立产品支持文档。通过这几部分的工作，实现“编码、调试、完善、内部测试、改错、再完善”的目的，这六部分的关系及具体操作流程如图 13-2（见下页）所示。

在这些活动中，需要重点强调采用统一的编码规范的重要性，其主要用途是统一编程风格、提高代码质量、给已婚懂得编程的人用。良好的编码习惯无论对程序员个人还是公司均有很大的帮助，其为代码复用的基础，统一的编码规范有助于代码在项目组内复用，也方便公司内其他项目使用。除此之外，良好的编码习惯还有如下好处：1、方便代码的交流和维护；2、不影响编码的效率，不与大众习惯冲突；3、使代码更美观、阅读更方便；4、使代码的逻辑更清晰、更易于理解。对于程序员个人而言，养成良好的代码编写习惯，是成为优秀程序员的基本功。

图表 13-2 实现及测试活动流程图



13.3 编码流程

13.3.1 工作准备

1. 与其他各阶段工作一样，在开始本阶段具体工作之前，需要细化并更新项目进度表，制定详细的实现与测试计划：项目组依据《项目计划》共同协商实现与测试计划，开发组长起草《实现与测试计划》，《实现与测试计划》。该计划主要包括对开发环境配置方案、编程计划和集成测试计划的描述；
2. 评审该计划：《实现与测试计划》需进行同行评审；如果通过，则转向（3）；否则退回（1）。
3. 确定相关规范：项目组根据所选用的软件开发工具，确定各模块的编程语言；若机构已经存在合适的编程规范，则采用之；否则由开发小组共同制定新的编程规范（需留存）或对已有的编程规范进行修改定制。
4. 根据《实现与测试计划》中对开发环境要求的描述，由开发组长指定人员搭建将要使用的开发环境，并保证在整个开发过程中该环境的一致性。各小组成员不得在未经项目经理或开发组长同意的情况下安装任何软件，以保证开发环境的纯洁性。

13.3.2 编码活动

1. 开发人员根据《实现与测试计划》中“编程计划”、《模块设计》、《用户界面设计》、《数据库设计》和《编码规范》编写模块代码。
2. 开发人员在编写完成每个模块时，必须对自己的代码进行必要的自查和测试。
3. 开发人员在完成代码自查和测试后，编制《模块卷宗》；《模块卷宗》的编写参照《模板_模块卷宗》。
4. 在编码实现过程中，项目经理指定专人对系统程序代码进行抽查，并将审查意见记录在《模块卷宗》中。

13.3.3 编码中常见问题[20]

一、如何避免开发阻塞？

项目经理在安排开发任务的时候，不要让能干人忙死，不能干的闲死（这种现象在学生实训中更为突出）。尽可能不要使多个任务串行，否则万一某个任务延误，将导致后面的任务全部延误。怎么避免该问题呢？

首先，在编码过程中可能会遇到各种技术问题，可能会导致整个编码任务的阻塞。这就需要使用本书实训指导中提出的思路去认真解决。即在需求分析及设计阶段，开发人员与系分人员紧密配合沟通，开发人员为主把在项目中可能会遇到的技术难题通过技术预演的方式解决掉，并且制定技术预研计划，编写必要的技术预研文档。

其次，识别所有任务中的关键任务，即有其他任务有重大影响的任务，对这些任务的完成进度纳入风险管理体系，加强跟踪及提出应对措施。以避免因为关键任务的延迟导致开发的阻塞。

二、有最好的编程语言吗？

答案很肯定，没有？实际上只有最适合。如今的 Visual Basic、Delphi、Visual C++、Java 和 C#等语言各有所长，很难明确说哪个好哪个不好。使用 VB 开发出来的大型商用软件也很多，在对日软件外包单子中，作者现在遇到的使用 VB 的也很多。

在学习编程语言时，精通一门就可以了，在开发过程中使用到其他新的编程语言上手也会比较快。如果都学个皮毛，实际上是什么语言都不会用。

在开发商用软件时，能很好地解决问题的编程语言就是好语言。公司应该根据实际情况，选择业界推荐的并且自己擅长的编程语言来开发软件，才能保证有较好的质量与生产率。

三、换用更快的计算机还是开发更快的算法？

如果软件运行较慢，是换一台处理性能更高的计算机，还是对算法进行性能调优？要根据“成本—收益”来决定。因为性能调优虽然可以从根本上提高软件的运行速度，但可能引

[20] 本节摘录自林锐主编的《IT 企业研发管理——问题、方法和工具》，电子工业出版社。然后由本书作者根据自身经验及授课过程中的心得体会整理编写而成。

入错误以及进度的延误。由于目前计算机硬件相对比较便宜，如果换一台处理性能更高的计算机可以解决问题，而且费用比较低的话，也是一种不错的选择。

四、要多用新技术和技巧吗？

我们开发软件是为了满足客户的需要，而不是自己闹着玩或追求技术挑战。为了提高质量、提高开发效率并且降低成本，应当尽可能采用成熟可靠的技术来开发软件。

在编程时要尽量少用技巧，技巧的优点是能另辟蹊径解决问题，缺点是技巧并不为其他人熟知。在程序中使用太多的技巧，如果相应技术资料再跟不上的话，可能会留下错误隐患，别人也难以理解。一个局部的优点对整个系统而主是微小的，而一个错误则可能对整个系统产生致命的影响。因此，建议用自然的方式编程，不要滥用技巧。

五、夜里编程效率更高吗？

编程和调试有这样的特点：干活要一鼓作气，不要中途停下来做其他事情之后再接着编程调试，否则思路可能被打乱，重新捡起来很费劲。所以程序员经常在夜里干活，这样效率比较高。但是，从个人健康来说，这样长久的话绝对伤害身体，只能偶尔为之。每个软件开发人员都要学一点养生之道，不要让不良的工作习惯影响健康。

六如何提高团队编程的质量？

首先，在组建开发团队时，尽量多一些编程老手（至少有 2 年以上编程经验），少一些编程新手。由于开发工具越来越先进，现在的编程技术门槛也越来越低。有些公司为了省钱，往往低薪招聘编程新手来干活。这种做法无疑会大大降低团队的战斗力。公司支付低薪而省下来的钱，远不及开发团队修补软件质量带来的额外成本。笔者建议在实际开发时，编码人员中新手的比例不能超过 50%，可以采用一对一的“传、帮、带”方式尽快提高新员工的编程能力，提高整个团队的编程质量。

其次，如果每个开发人员的技能都是合格的，编程规范就显得比较重要。让开发人员都按照既定的规范进行编程，是提高代码质量、降低代码维护代价的简单有效的方法。

最后，编写高质量的程序离不开责任心，这一点也是做任何工作所必须的。每个程序员都应对自己的代码进行仔细的跟踪调试，进行严格的自我测试，然后再提交给测试人员进行

单元测试或集成测试。

13.4 测试流程

13.4.1 单元测试

在设计阶段，整个系统被细分为许多模块（或类），这里可以把模块或类理解为单元。每个单元的接口、数据结构与算法都已经设计完成。在编码完成之后，把这些单元集成起来之前，需要先执行单元测试，以保证单元本身正确无误，保证单元是否符合设计要求，可以采用白盒测试的方法，也可以采用黑盒测试的方法，根据公司测试人员的安排及投入来确定。笔者建议在测试投入许可的情况下，对单元尽量做白盒测试。在实际执行时，可以按如下流程来开展单元测试：

- 项目经理根据开发人员开发进展情况，安排测试人员或系统分析人员（有些公司可能是开发人员）编写《单元测试管理列表》或直接使用相关测试工具来编写，具体采用什么方式，由各项目组根据实际情况在项目开发计划里确定。
- 项目经理审批《单元测试管理列表》，并指定测试人员进行单元测试，并记录在《单元测试管理列表》中；若使用专门的测试管理工具，则把结果记录进该工具中。
- 测试人员依据已审批的《单元测试管理列表》进行相应的单元测试，产生《单元测试报告》或登记进测试管理工具，然后由测试管理工具产生相关的单元测试报告。

13.4.2 集成测试

什么叫集成测试？集成测试（也叫组装测试，联合测试）是单元测试的逻辑扩展，最简单的形式：两个已经测试过的单元组合成一个组件，并且测试它们之间的接口。

集成测试是在单元测试的基础上，测试在将所有的软件单元按照概要设计规格说明的要求组装成模块、子系统或系统的过程中各部分工作是否达到或实现相应技术指标及要求的活动。集成测试所持的主要标准是《软件概要设计规格说明》。由于软件系统需要把单元模块集

成到一起才能形成协同操作的功能，所以软件项目/产品都不能摆脱系统集成这个阶段。

- 当开发进程达到《实现与测试计划》中预期集成点时，且《实现与测试计划》中“集成测试计划”中涉及的单元模块均通过单元测试，开始集成测试活动。
- 系统分析员及开发组长共同制定本次集成的《集成测试策略》，主要包括对本次集成范围、集成顺序、集成环境、集成方法等内容的描述，为产品集成做好准备工作。
- 开发组长组织开发人员及测试人员按照本项目的《集成测试策略》中集成环境的描述，建立产品集成环境，依据其中产品集成顺序和产品集成方法的描述，进行产品集成活动，同时搭建集成测试环境。
- 此项工作准备就绪后，开发组组长在《集成测试报告》1.基本信息表：“集成及测试环境”表项签字，说明项目已具备产品集成及可以进行集成环境确认的工作。
- 测试人员根据本项目的《集成测试策略》和《概要设计》编写《集成测试用例》或把集成测试用例放进测试管理工具，并进行同行评审。
- 开发组长组织测试人员根据《实现与测试计划》中“集成测试计划”和《集成测试用例》进行系统的集成测试，将测试的结果填写到《集成测试报告》或测试管理工具中。

【说明：单元测试用例、集成测试用例的编写时机、编写方法及用例包含的内容，放到第 14 章进行讲解，此处主要是让大家了解单元测试、集成测试的工作流程及步骤。】

13.5 缺陷管理与改错

如果在测试时发现了缺陷，开发人员应当尽早消除缺陷，并且需要对缺陷的全生命周期进行详细的跟踪及管理。通常缺陷管理及改错的指导原则为：

- 在单元测试和集成测试过程中，测试人员发现系统中的缺陷时，必须将缺陷记录在《缺陷管理列表》或记录进 BUG 管理工具（一般的软件测试管理工具均带有 BUG 管理功能，也可采用专门的 BUG 管理工具）。
- 开发人员及时消除已经发现的缺陷，若使用 BUG 管理工具，则可以设置查询条件，

查询由自己负责并且还未解决的缺陷。

- 开发人员消除缺陷之后，测试开发人员应当马上进行回归测试，确保不会引入新的缺陷。
- 集成测试人员在完成一次集成测试后，依据《缺陷管理列表》统计填写《缺陷管理统计报告》或由 BUG 管理工具对缺陷进行统计分析。

在缺陷管理与改错时，各公司可能会制定不同的管理流程，以下是本书建议使用的缺陷管理与跟踪流程。

- 测试人员发现缺陷后，填写《缺陷管理列表》或 Bug 管理工具中缺陷信息项，并将其状态置为“Open”，提交项目经理。
- 项目经理确认缺陷内容后，将其转为相关人员解决或指派给相关人员解决。
- 当缺陷解决人员认为缺陷已经修复后，即可填写《缺陷管理列表》或 Bug 管理工具中相应款项，然后将此《缺陷管理列表》及修复后的程序提交给测试人员进行回测。
- 测试人员进行回测，填写《缺陷管理列表》或 Bug 管理工具中验证信息项：
  - ✧ 如果该缺陷被修复了，则将此缺陷状态置为 Closed, 并在《缺陷管理列表》的“缺陷状态”一栏填写“Open/Closed”（若同一个缺陷无论回测过多少次均在此状态栏中填写其关状态，直到最后一个状态标识为 Closed 或 Rejected 为止，如：Open/Reopen/Reopen/Close）；
  - ✧ 如果该缺陷未被修复，则将该缺陷状态置为 Reopen，状态标识格式同上；
  - ✧ 不论该缺陷有没有被修复，若在回测过程中，又测试出新的缺陷，则在原《缺陷管理列单》中创建新的工作表(EXEL 表格中 sheet)，并标识为复测，以及第 3(4,5，…，n) 次测试；
- 开发人员查询状态为 Open 和 Reopen 的缺陷，不是缺陷，由项目经理确认后，可置状态为 Rejected。
- 对于不能解决和延期解决的缺陷，开发人员要提出申请，争求项目经理的同意后，才能将其状态置为“Rejected”。

【注：若项目组比较小，测试人员清楚各个模块由哪位开发人员负责，则可以在填写缺陷时直接指派相关人员，而不再让项目经理进行指派。只有当测试人员与开发人员就缺陷解决达不成一致时，才由项目经理及系分人员共同讨论解决方案。】

开发人员在改错时，要注意如下事项[21]：

找到错误的代码时，不要急于修改，先思考一下：修改此代码会不会引发其他问题？如果没有问题，可以放心修改；如果有问题，那么可能要改动程序结构，而不止于一行代码。

有些时候，软件中可能潜伏同一类型的许多错误（例如由不良的编程习惯引起的）。好不容易逮住一个，应当把同类的错误全部找到并且修改，并不一定非要等测试人员提出缺陷时才去解决。

在改错之后一定要马上重新测试，以免引入新的错误。改了一个程序错误固然是喜事，但要防止乐极生悲。更加严格的要求是：不论原先程序是否绝对正确，只要对此程序作过改动（哪怕是微不足道的），都要重新测试。

上述事情做完后，应当好好反思：我为什么会犯这样的错误？怎么能够防止下次不犯相似的错误？最好能写下心得体会，与他人共享经验教训。

13.6 建立产品支持文档

在整个系统实现及测试过程中，负责文档的人员应当根据开发的进展及时的编写并调整相关产品支持文档，具体如下：

- 文档人员在开发人员的协助下编制《用户操作手册》、《系统维护手册》、《培训教材》、联机帮助、系统安装包等。
- 《用户操作手册》、《系统维护手册》、《培训教材》、联机帮助等完成之后，由项目经理组织同行评审。

当满足以下条件时，系统实现与测试整个过程可以结束：

- 软件代码已经编写完成，软件集成在一起可以运行。

[21]摘录自林锐等主编的《IT 企业研发管理——问题、方法和工具》，电子工业出版社。

- 满足集成测试结束准则，集成测试通过。
- 本过程所有文档已经完成，《用户操作手册》、《系统维护手册》、《培训教材》通过同行评审。

第14章 制定测试方案及编写测试用例

内容提要：

- **CMMI** 中对应实践
- 测试资料收集与整理
- 检查产品说明书
- 测试方案的制订
- 测试计划书的编写及要素
- 测试用例编写

在整个软件开发过程中，测试占有大量的工作。比如：微软的 SharePoint Team Services 项目组的组成，产品部门经理（product unit manager）：1 人；程序经理（program manager）：10 人；开发人员（developer）：29 人；测试人员（tester）：27 人；可用性工程师（usability engineer）：1 人（合用）；用户培训（user education）：2 人（合用）；产品经理（product manager）：1 人（合用）；本地化（localization）：4 人（合用）；行政助理（administrative assistant）：1 人；其中测试占了近 40%。但是，在国内不少企业中，对测试不是那么重视，对测试人员的培养及待遇也有一定的认识偏见。实际上，测试员应当比程序员更资深，有助于发现系统中一些隐性的重大缺陷。

在整个软件测试活动中，重点工作有时在测试的策划，而不是执行测试。测试方案及测试用例设计质量对测试的效果有着非常重要影响，也是测试活动能否顺利开展的基础。本章重点讲解怎么样进行测试策划，制定测试计划及设计测试用例。

14.1 CMMI 中对应实践

本章主要是完成 CMMI 中 VER 和 VAL（Validation——确认）两个过程域的相关准备活动，其中 VER 的相关实践在上一章已经进行了讲解，本节主要解释与 VAL 中准备活动相关的实践。

在 VAL 中，与测试或客户验收（系统试运行）准备有关的是 SG1 Prepare for Validation（确认准备），其目的是为将要进行的确认活动准备，准备活动主要包括，选择要确认的工作产品或产品组件；建立并维护确认的环境、过程、标准。为了达到此目的需要完成如下三个实践：

SP1.1 Select Products for Validation（选择需确认的产品），目的是选择需确认的产品及产品组件，及对每一个产品及产品组件使用的确认方法。根据用户的需要来选择相关要确认的产品或产品组件，一般以下几种，产品或产品组件的需求及设计；包含系统、硬件、软件、服务文档在内的产品或产品组件；用户界面；用户手册；培训资料；过程文档。

SP1.2 Establish the Validation Environment（建立确认环境），目的是建立并维护支持确认活动所需的环境。环境一般包含如下类别，与将要被确认的产品交互的测试工具；临时的嵌入式测试软件；用于转储或作进一步分析和重放的记录工具；用软件等模拟的子系统或组件；实时接口系统；熟练操作或使用前面提到的元素的人员。

SP1.3 Establish Validation Procedures and Criteria（建立并维护确认过程和准则），为了确保产品或产品组件放到预期的环境里达到预期的应用，测试用例及测试过程需要满足确认过程的需要。

14.2 测试资料收集与整理

测试资料收集与整理是软件测试策划的一个重要组成部分，软件测试策划的目的：规定测试活动的范围、方法、资源和进度；明确正在测试的项目、要测试的特性、要执行的测试任务、每个任务的负责人，以及与计划相关的风险。由此可见，测试策划是制作产品详细计

划过程的副产品，重在策划过程，而不是产生测试计划的文档。测试计划过程的最终目的是交流（而不是记录）软件测试小组的意图、期望，以及对将要执行的任务的理解。我们需要收集与整理的内容包括：

一、通用的信息

- 一般信息：公司大体情况；所在测试部门的大体情况；周围的人与事、工作环境；公司的企业文化。
- 技术信息：软件的类别及构成；软件的用户界面；所测软件涉及使用的第三方软件。

二、被测软件的类别及构成

为制定恰当的测试方案，需要了解清楚软件的类别及结构：

- 软件的类别及用途；
- 软件的技术结构；
- 所支持的平台；
- 软件的主要构成部分，各自功能及各部分之间的联系；
- 每一构成部分所使用的计算机语言；
- 若进行白盒测试，则了解各部分已建立的函数库的函数或类的接口，以及他们的用途、输入、输出值。

三、被测软件的用户界面

还需了解最终用户的软件界面：

- 用户界面类别（Windows 窗体、命令行、网页类）
- 用户界面各部分功能之间的联系
- 界面中组成控件的特性及操作特点

以上信息来源：《用户需求说明书》、《软件需求规格说明书》、系统设计说明书等各类文档，及与设计或编码等人员沟通。

为了能有效的展开测试，在测试开始前，需要准备好如下资料，也正是测试计划/策划所要解决的重点：

- 软件及测试基本情况（进度、进展、分工）
- 软件目前主要存在的问题
- 测试管理流程（特别是 BUG 管理流程）
- 使用的测试软件、BUG 管理软件、配置管理软件
- 测试的环境
- 软件产品的文件、说明等

14.3 检查产品说明书

产品规格说明书（简称 SPEC）可以保证大家都知道最终会得到什么样的产品。确保最终产品符合客户要求以及正确计划测试的唯一方法是用产品说明书完整描述产品。软件测试人员将其作为测试项目的书面材料，并且还可以在编写代码之前找出缺陷。

对 SPEC 进行检查一般分为两类，一类称为高级审查；一类称为低级审查。两者的出发点不太一样，其中高级审查要做的事情为：

- 测试产品说明书的第一步不是钻进去找软件缺陷，而是在一个高度上审视，以找出根本性的大问题、疏忽或遗漏。
- 通过研究更好地了解软件要做什么，为了可以做更好的测试。
- 怎么进行高级审查？

1. 设身处地为客户着想，把自己当客户，通过与市场人员、销售人员或实际客户进行沟通，了解客户所想；熟悉软件应用领域相关知识极有好处
2. 研究现有的标准和规范，惯用语和约定、行业要求、国家标准、图形用户界面、硬件和网络标准等等

- 审查和测试同类软件，研究同类软件有助于制订测试条件和测试方法，审查竞争产品重点注意问题是：
  - ✧ 规模。软件是小型的还是大型的？这在测试中有何不同？
  - ✧ 复杂性。软件是否复杂？会影响测试吗？

✧ 测试性。是否有足够的资源、时间和经验来测试软件

✧ 质量/可靠性。软件是否完全依据质量标准计划编写的？可靠性如何？

✧ 对同类软件动手实践，通过试用去了解，为测试自己的软件积累经验。

SPEC 的低级审查，主要是关注被审查对象——产品说明书——编写的质量如何？编写中有没有可能存在问题的用语等。具体包含的内容为：

产品说明书属性检查清单，优秀产品说明书应具有 **8** 个重要属性：

- 完整；
- 准确，解决方案正确吗？目标明确吗？有没有错误？
- 精确、不含糊、清晰；
- 一致，是否自相矛盾，与其他功能有没有冲突？
- 贴切，描述是否必要？有无多余信息？
- 合理，以现有人力、物力能否实现？
- 代码无关，坚持产品定义；
- 可测试，特性能否测试？是否为操作提供足够的信息？

产品说明书用语检查清单，找出问题描述用语，一下用语的表述可能表明 **SPEC** 本身存在问题，在开始测试之前要进行纠正：

- 总是、每一种、所有、没有、从不：此类绝对描述，要设计针锋相对的测试案例；
- 当然、因此、明显、显然、必然：诱使接受假定情况，不要中圈套；
- 某些、有时、常常、通常、惯常、经常、大多、几乎：太模糊，无法测试；
- 等等，诸如此类、依此类推：也无法测试，需要明确；
- 良好、迅速、高效、小、稳定：不确定说法，不可测；
- 已处理、已拒绝、已忽略、已消除：可能隐藏大量需要说明的功能；
- 如果…那么…：缺少“否则”则会出现大问题。

14.4 测试方案的制订

测试方案是软件测试的总体规划。包括：测试的方针、策略、系统建立、人员分配、进度等。一般由测试组长或测试部门的主管来完成。

不同的测试方案，制订时机也不一样，单元测试一般可以在详细设计或编码阶段制订；集成测试一般可以在系统设计结束后或单元测试结束前制订；系统测试一般可以需求分析完成或集成测试结束前制订。根据不同项目类型、不同项目人员结构来确定。在制定测试方案的时候，需要主要考虑以下四个因素：

- 软件的现状及将来可能的发展，现状包括软件的复杂程度、规模、现有缺陷的难度及缺陷发现的频率等；软件将来可能的发展，需要在决定测试结构时预留一些空间。
- 现有资源及将来可能获得的补充资源，包括测试用计算机、测试用软件、测试人员；将来公司对测试工作的重视程度、投入资金的计划、测试队伍建设计划等。
- 风险分析：软件系统与选定的硬件或某些第三方软件的不兼容性；软件的功能并不能达到其在用户说明书中的列项；软件的实际技术指标与设计要求和用户说明相去甚远；软件存在致使的安全漏洞；开发费用、开发时间比预期高。
- 制订测试的策略，确定采用的测试方法、范围，一般功能测试、安装测试、兼容性测试是必不可少的；强度测试、容量测试、数据库测试等视软件特性而确定

14.5 测试计划书的编写及要素

在软件测试工作中，测试计划作为整个项目计划的一部分，根据开发所处的阶段不同编写不同的测试计划。不论什么阶段的测试计划，其作用一般为：

- 测试工作的依据，使测试工作有目标、有计划地进行。
- 帮助我们以科学的方法管理测试工作，从测试资源分配、测试进度、测试方法、步骤、测试单元划分、测试覆盖程度、缺陷报告及管理方式等都明确规定；
- 及早发现软件规格说明书的问题，以便及早修正；

- 有利于日后测试部门工作的相互协调；
- 有利于日后测试自动化，需要列出所有的测试用例，帮助确定自动化测试程序编写顺序。

14.5.1 测试计划书衡量标准

测试计划书必须有明确的测试目标、范围和深度、具体实施方案及测试重点；提供大体的测试进度及所需的资源（人力、物力、软件、硬件等）。好的测试计划应具备的特点是：

- 有效达到最终目的，引导整个软件测试工作正常运行，配合编程部门、保证软件质量、按时将产品推出。
- 列举的所有数据必须是准确的，比如：兼容性所要求的数据、输入、输出数据等。
- 提供的方法使测试高效运行，较短时间内找出尽可能多的缺陷。
- 提供明确的测试目标、测试的策略、具体步骤及测试标准。
- 既强调测试重点，也重视测试的基本覆盖率。
- 测试方案尽可能充分利用公司现有的、可以提供给测试部门的人力物力资源，而且是可行的。
- 测试安排有一定的灵活性，可以应付一些突然的变化。

14.5.2 测试计划内容

测试计划做为开展软件测试的纲领性文档，如同项目开发计划对一个软件开发一样重要，其内容一般包括如下几点：

- 测试计划书的文件名及版本号
- 基本情况介绍（测试目的、背景、测试范围及参考文献等）
- 测试的具体目标（要测的是软件的哪些部分）
- 具体执行的测试类型

- 测试通过的判断准则
- 测试用例
- 测试准备工作及测试结果的处理
- 测试工作中涉及的相关事项（测试工具、硬件、第三方软件等）
- 部门责任分工
- 测试人力资源分配
- 测试进度列表
- 测试工作中可能面临的偶发事件及危机处理（即测试过程中的风险管理）
- 缺陷管理流程

14.6 测试用例编写

14.6.1 单元测试用例编写

什么叫单元测试？工厂在组装一台电视机之前，会对每个元件都进行测试，这就是单元测试。那软件开发中什么应当叫单元测试？

一般来说，在结构化程序时代，单元测试所说的单元是指函数，在当今的面向对象时代，单元测试的所说的单元是指类（不论是私有函数还是公用函数都应当纳入单元测试）。在进行开展单元测试工作之前，需要理解如下的两个几个问题：

1、什么时候进行单元测试？

- 一般来说，对于开发单元测试越早越好；
- 先编写产品函数的框架（概要设计），然后编写测试函数，针对产品函数的功能编写测试用例，然后编写产品函数的代码，每写一个功能点都运行测试，随时补充测试用例。

2、谁来执行单元测试？

单元测试与其他测试不同，单元测试可看作是编码工作的一部分，应该由程序员完成，

也就是说，经过了单元测试的代码才是已完成的代码，提交产品代码时也要同时提交测试代码。测试人员可以作一定程度的审核。在我们项目中，单元测试也由开发人员来做。一般单元测试用例包含以下内容：

- 用例编号、被测对象
- 测试用例的核心是输入数据，输入数据包括四类：参数、成员变量、全局变量、IO 媒体，这四类数据中，只要所测试的程序需要执行读操作的，就要设定其初始值，其中，前两类比较常用，后两类较少用。输入数中，要包括：正常输入、边界输入（最大值及最小值）、非法输入
- 期望输出，在给定的输入下，单元应当给的反映。

【注意：单元测试用例通常没有操作步骤。为什么？】

14.6.2 集成测试用例编写

集成测试是为确保各单元组合在一起后能够按既定意图协作运行，并确保增量的行为正确。它所测试的内容包括单元间的接口以及集成后的功能，所以集成测试用例中一般包括接口测试用例。一般使用黑盒测试方法测试集成的功能，对应于设计来说编写出来的代码能否正确执行，并且对以前的集成进行回归测试。

集成测试的必要性：一些模块能够单独地工作，但并不能保证连接起来也能正常工作。程序在某些局部反映不出来的问题，有可能在全局上会暴露出来，影响功能的实现。在某些开发模式中，如迭代式开发，设计和实现是迭代进行的。在这种情况下，集成测试的意义还在于它能间接地验证概要设计是否具有可行性。通常情况下，集成测试用例内容包括：

- 用例编号、被测对象、场景等。
- 测试用例的核心是输入数据：输入数据包括四类：参数、成员变量、全局变量、IO 媒体，这四类数据中，只要所测试的程序需要执行读操作的，就要设定其初始值，其中，前两类比较常用，后两类较少用。输入数中，要包括：正常输入、边界输入（最大值及最小值）、非法输入

- 测试时操作步骤。
- 期望输出，在给定的输入下，系统应当给的反映。

14.6.3 系统测试用例编写

系统测试（System Test，简称 ST）是将经过测试的子系统装配成一个完整系统来测试。它是检验系统是否确实能提供系统方案说明书中指定功能的有效方法。系统测试的目的是对最终软件系统进行全面的测试，确保最终软件系统满足产品需求并且遵循系统设计。有时，集成测试与系统测试可以不做很严格的划分。

系统测试采取了一种不同的测试用例编写方法，因为不可能使用一项技术来设计系统测试用例，需要根据不同类型的测试来设计测试用例，那么系统测试用例的设计需要大量的创造性。事实上，设计好的系统测试用例比设计系统或程序需要更多的创造性、智慧和经验。一般来说，系统测试包含 15 种类型，本书中重点讲述了如下九种类型测试用例的编写：

对外的接口测试用例、与《需求规格说明书》对应的功能测试用例、程序的健壮性测试用例、系统性能测试用例、系统图形用户界面测试用例、系统信息安全性测试用例、压力测试用例、系统可靠性测试用例、系统安装/反安装测试用例等。每种用例的编写请参见实训指导。

第15章 系统测试

内容提要：

- **CMMI** 对应实践
- 系统测试简述
- 系统测试活动内容

系统测试有着特定的目的：将系统或程序与其初始目标进行比较，比如：《用户需求说明书》等。由于可见其包含了两方面的含义：一是，系统测试并不局限于系统，如果产品是一个程序，那么系统测试就是一个试图说明程序在整个系统中是如何不满足其目标的过程；二是，如果产品没有一组书面的、可度量的目标，系统测试也就无法进行。

15.1 CMMI 对应实践

系统测试主要是对应于 CMMI 的确认（Validation，简称 VAL）过程域，此过程域共达到两个目标，一个是确认的准备，二是对产品或产品组件进行确认。其中确认的准备在上章中已经进行了解释，本章重点解释第二个目标及为了实现该目标执行的两个实践。整个 VAL 的目的是：演示一个产品或产品组件当放进预期的环境中可以充分达到预期的功能。具体为：**SG 2 Validate Product or Product Components**（确认产品或产品组件），目的是确认产品或产品组件，以确保在预期作业环境中可适用。

确认方法、程序及准则是用来确认所选择的产品与产品组件，以及相关的维护、培训及支持服务。此项确认是在适当的确认环境中进行。整个产品生命周期中，都要执行确认活动。

SP 2.1 Perform Validation（执行确认），为让使用者接受，产品或产品组件置于预期作业环境中，其工作表现必须完全符合预期要求。主要是：执行确认活动，并依据已建立的方法、程序及准则，搜集结果资料；适当时，记录已执行的确认程序及执行时所发生的

偏差。在此实践中，常会产生以下的一些资料：确认报告、确认结果、确认对照表、运行过程日志、操作演示文件。

SP 2.2 Analyze Validation Results（分析确认结果），依据定义好的准则对确认测试、检查、演示或评估产生的结果数据进行分析。分析报告应指出需求是否符合，如有偏差，报告需记载成功或失败的程度，并将可能失败的原因分类。分析报告或确认文件可能指出，测试结果不佳乃是确认程序或执行环境的问题。通过分如下几个步骤来执行：一是比较实际及预期结果；按已建立的确认准则识别问题，包括识别于预期作业环境下执行不佳的产品或产品组件，或识别确认方法、准则及(或)环境问题；分析确认缺陷数据；记录分析结果并识别问题；利用确认结果，将实际度量及性能，与预期使用或操作需要进行比较。

15.2 系统测试简述

系统测试（System Test，ST）的目的是对最终软件系统进行全面的测试，确保最终软件系统满足产品需求并且遵循系统设计的标准和规定。采用黑盒测试的方法进行测试，主要内容有：功能性测试、健壮性测试、性能-效率测试、用户界面测试、安全性测试、压力测试、可靠性测试、安装/反安装测试等。

制定测试计划和设计测试用例活动的进入准则是：产品需求和系统设计文档完成之后，系统测试小组就可提前开始制定测试计划和设计测试用例，不必等到“实现与测试”阶段结束，以提高系统测试效率。

执行系统测试计划活动的进入准则是：集成测试已通过。

在集成测试的时候，已经对一些系统的功能、接口进行了测试，那么在系统测试时是否能跳过相同内容的测试？答案是，不能。因为集成测试一般是在开发环境下中开展，可以由开发人员或专门的测试人员来执行，那不是真正的目标系统，所以集成测试不能做为系统已经通过测试的依据。而系统测试强调的是在用户实际使用环境或模拟用户实际使用环境下进行测试，并且为了保证测试的客观性，应当有机构的独立测试小组来执行系统测试。所以在

进行系统测试的时候，测试小组的成员一般有如下几个来源：

- 委托外部测试机构进行测试，比如：软件测评中心；
- 与项目组独立的测试小组或测试部门人员；
- 邀请其他项目的开发人员参与测试；
- 本项目的部分开发人员（但绝对不能以本项目开发人员为主进行系统测试）；
- 技术支持或工程实施人员（更能清楚了解用户的实际使用环境及需求）。

15.3 系统测试活动内容

15.3.1 系统测试内容

1. 用户层，主要是面向产品最终的使用操作者的测试，重点突出的是从操作者的角度，测试系统对用户支持的情况，用户界面的规范性、友好性、可操作性，以及数据的安全性。
   - 用户支持测试，用户手册、使用帮助、支持客户的其他产品技术手册是否正确、是否易于理解、是否人性化。
   - 用户界面测试，在确保用户界面能够通过测试对象控件或入口得到相应访问的情况下，测试用户界面的风格是否满足用户要求，例如：界面是否美观、界面是否直观、操作是否友好、是否人性化、易操作性是否较好。
   - 可维护性测试，可维护性是系统软、硬件实施和维护功能的方便性。目的是降低维护功能对系统正常运行带来的影响。例如：对支持远程维护系统的功能或工具的测试。
   - 安全性测试，安全性主要包括了两部分：数据的安全性和操作的安全性。验证只有规定的数据才可以访问系统，其他不符合规定的数据不能够访问系统；验证只有规定的操作权限才可以访问系统，其他不符合要求的操作权限不能够访问系统。
2. 应用层，针对产品应用的测试,重点在系统应用的角度，模拟实际应用环境，对系统的兼容性、可靠性、性能等进行的测试。

- 系统性能测试，针对整个系统的测试，包含并发性能测试、负载测试、压力测试、强度测试、破坏性测试。并发性能测试是评估系统交易或业务在渐增式并发情况下处理瓶颈以及能够接收业务的性能过程；强度测试是在资源情况低的情况下，找出因资源不足或资源争用而导致的错误；破坏性测试重点关注超出系统正常负荷 N 倍情况下，错误出现状态和出现比率以及错误的恢复能力。
- 系统可靠性、稳定性测试，一定负荷的长期使用环境下，系统可靠性、稳定性。
- 系统兼容性测试，系统中软件与各种硬件设备兼容性，与操作系统兼容性、与支撑软件的兼容性。
- 系统网络测试，网络环境下，系统软件对接入设备的支持情况。包括功能实现及群集性能。
- 系统安装升级测试，安装测试的目的是确保该软件在正常和异常的不同情况下进行安装时都能按预期目标来处理。例如，正常情况下，第一次安装或升级、完整的或自定义的安装都能进行安装。异常情况包括磁盘空间不足、缺少目录创建权限等。还有一个目的是核实软件在安装后可立即正常运行。另外对安装手册、安装脚本等也需要关注。

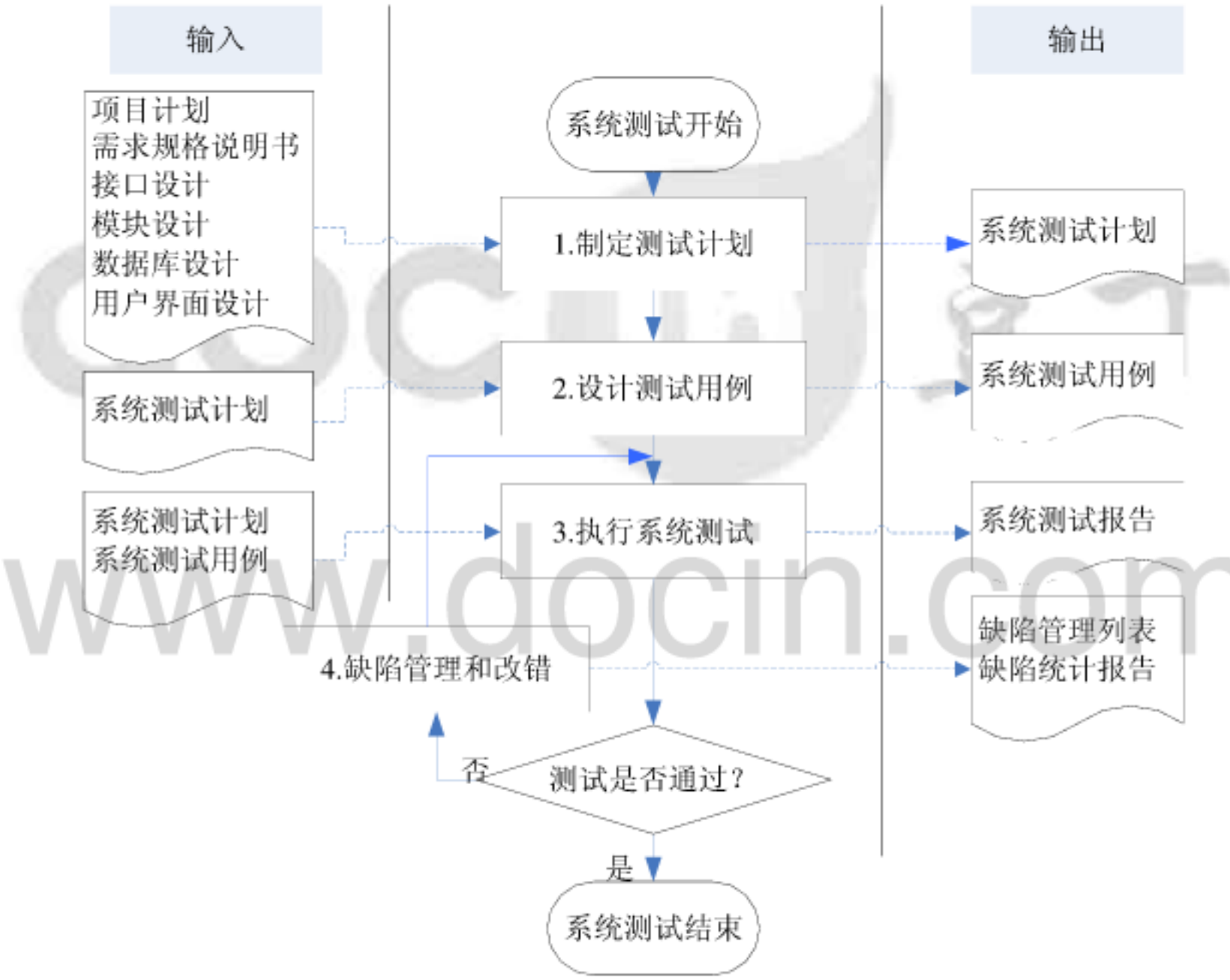
3. 功能层，针对产品具体功能实现的测试。
   - 业务功能的覆盖，关注需求规格定义的功能系统是否都已实现。
   - 业务功能的分解，通过对系统进行黑盒分析，分解测试项及每个测试项关注的测试类型。
   - 业务功能的组合，主要关注相关联的功能项的组合功能的实现情况。
   - 业务功能的冲突，业务功能间存在的功能冲突情况。比如：共享资源访问等。
4. 子系统层，针对产品内部结构性能的测试，关注子系统内部的性能，模块间接口的瓶颈。
   - 单个子系统的性能，应用层关注的是整个系统各种软、硬件、接口配合情况下的整体性能，这里关注单个子系统。
   - 子系统间的接口瓶颈，例如：子系统间通讯请求包的并发瓶颈。

■ 子系统间的相互影响，子系统的工作状态变化对其他子系统的影响。

5. 协议/指标层，针对系统支持的协议、指标的测试（主要是测试协议或指标的一致性及互通性）。

软件开发过程的系统测试，可以分为：制定测试计划、设计测试用例、执行系统测试、缺陷管理和改错四大步骤，其中的“缺陷管理和改错”请参见本书的 13.5 节，此处不做详细讲解。具体可以按图 15-1 所示的流程来执行。

图表 15-1 系统测试流程图



15.3.2 制定系统测试计划

系统测试小组各成员共同协商测试计划。测试组长起草《系统测试计划》，详细内容请参见第 14 章的相关内容。该计划主要包括：

- 测试目标
- 测试范围
- 测试方法
- 测试环境与辅助工具
- 测试完成准则
- 人员与任务表

《系统测试计划》根据情况进行正式或非正式评审。项目经理根据项目计划及评审结果审批《系统测试计划》。该计划被批准后，方能进入下一步工作。

15.3.3 设计测试用例

系统测试人员根据《系统测试计划》和指定的模板，设计《系统测试用例》并进行同行评审；《系统测试用例》的编写参照第 14 章提供的模板及编写指导。测试组长邀请开发人员和同行专家，对《系统测试用例》进行技术评审。该测试用例通过技术评审后，进入下一步工作，根据测试用例执行系统测试。

15.3.4 执行系统测试

系统测试人员依据《系统测试计划》和《系统测试用例》执行系统测试，并对测试过程中发现的缺陷进行跟踪，及时解决验证。

将测试结果记录在《系统测试报告》中，用《缺陷管理列表》或专门的缺陷管理工具来记录所发现的缺陷，并及时通报给开发人员。

【注：此活动必需在单元测试、集成测试完成之后进行。缺陷管理的流程及细节请参见第 13 章的实训指导】

在如下情况可以结束系统测试活动，结束准则各公司根据所研发系统类型的不同，可自行设定，此处是给出的两个建议结束准则：

- 对于非严格系统可以采用“基于测试用例”的准则：

1. 功能性测试用例通过率达到 100%；

2. 非功能性测试用例通过率达到 95%。

■ 对于严格系统，应当补充“基于缺陷密度”的规则：

相邻 n 个 CPU 小时内“测试期缺陷密度”全部低于某个值 m。具体值根据项目的类型来确定。

在系统通过系统测试之后，一般可以进入如下阶段：安装到客户使用环境下试运行、发布 Beta 版；系统验收。

第16章 客户验收

内容提要：

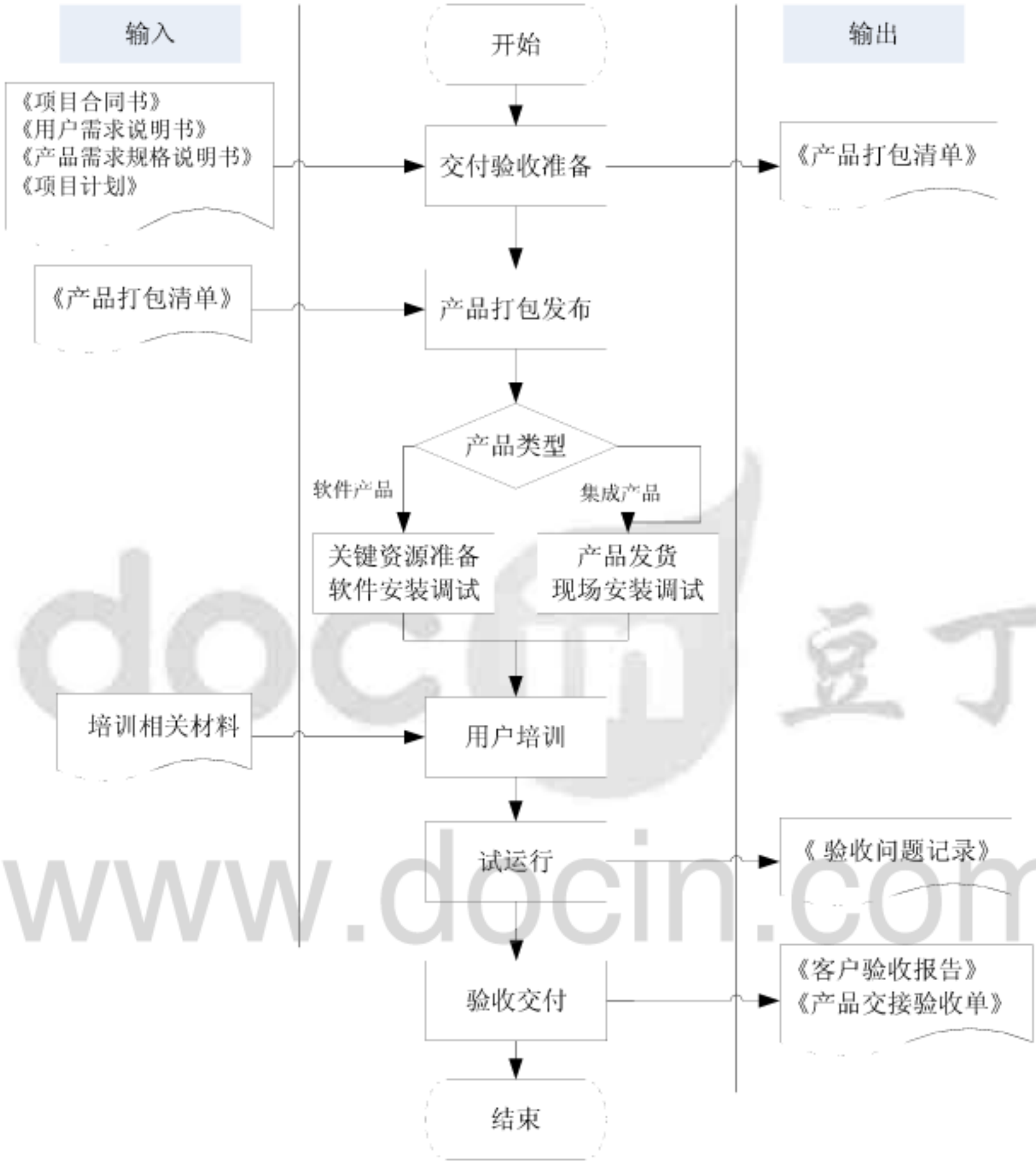
- 客户验收简述
- 系统试运行
- 验收流程

16.1 客户验收简述

客户验收（Customer Acceptance，CA）是指公司和客户依据合同及相关附件（如相对应的需求规格说明书等准确表达双方共同约定的有效文件），规范项目的验收和交付过程，保证公司各项目在交付阶段，对产品进行审查和测试，保证项目达到客户要求。合同订制类项目必须系统试运行之后进行客户验收阶段，然后再进入项目总结阶段；产品类项目要在 Beta 版之后或试点单位运行之后再进入项目总结阶段，没有客户验收阶段。该阶段的进入准则为：产品的系统测试已经完成，《系统测试报告》已经评审通过之后，方可进入该阶段。

系统开发好（通过系统测试）之后，可以安装到用户环境下进行试运行。在此过程中，将发现的缺陷以及对软件的建议及时报告给开发方负责系统试用的人员。负责研发项目的项目经理汇总这些缺陷或问题，组织开发人员及时处理这些反馈。对于客户发现的缺陷，应当立即修改，并在通过测试后交付给客户；对于一些难以马上实现的建议，可以由上级领导（研发部经理或总工程师）决定如何处理。但无论修改与否，都应当有客户的反馈尽快答复，这是提高客户满意度所必须的，可是能否保证顺利验收所必须的。客户验收及试用的流程图如下所示：

图表 16-1 客户验收及试用流程图



16.2 系统试运行

1.交付验收准备

项目经理按照《项目计划》，参考《项目合同书》、《用户需求说明书》、《产品需求规格说明书》的内容，明确验收交付活动的参加人员、进度安排和验收交付地点等内容；必要时与

用户协商；同时，项目经理要确定需要发布的产品的组成，并制定《产品打包清单》，《产品打包清单》要求罗列需要交付的各种内容，包括产品、文档、手册等，如果这些内容还不具有交付时所应有的特征，则在《产品打包清单》中需要说明产品打包发布所要完成的工作。

2.产品打包发布

项目经理指定专门的负责人完成《产品打包清单》中所列项目的集成、打包、美工设计等内容，形成最终交付给用户的产品

3.试运行环境搭建

软件类产品：开发方依据《用户需求说明书》、《产品需求规格说明书》对用户的使用环境、实施条件进行调研，确定所需的关键资源；协同用户准备所需关键资源。

集成类产品：依据《用户需求说明书》、《产品需求规格说明书》对用户的使用环境、实施条件进行调研，确定所需的关键资源；协同用户准备所需关键资源；将准备好的集成产品发送到用户，并取得用户接收的确认。

4.用户培训

项目经理与用户进行协商，确认用户培训的时间、地点、参与人员和培训内容，由项目经理或项目组成员依据培训相关材料，例如《培训教材》，对用户进行培训，要求培训后用户能够初步掌握产品的使用方法。

5.试运行

调试安装好的软件或系统集成产品在用户环境中按照用户所期望的模式进行试运行，试运行要求能够覆盖到《用户需求说明书》中所有与产品性能有关的内容，试运行期间，由用户对产品的运行情况进行记录，如果用户验收人员在试运行期间发现产品存在问题，则由用户填写《验收问题记录》，并反馈给项目经理，项目经理应当视问题的严重性与客户协商是否需要将产品返回开发商，以及是否需要二次验收，针对发现的问题给出合适的处理措施，并跟踪问题直到关闭。

16.3 验收流程

在系统验收之前，一般需要作以下准备工作：

- 若系统由多个子系统组成，首先由集每个子系统使用情况报告，由用户确认每个子系统均可使用。
- 与客户沟通，确认验收时间，验收参与人员。
- 编写验收所需的相关技术资料。
- 在验收准备工作完成之后，一般可按如下的过程进行系统的验收：
- 用户成果审查。用户验收人员审查开发方应当交付的成果。确保这些成果是完整的，并且是正确的。
- 用户验收人员根据试运行的记录，确保产品功能、质量符合需求。
- 由用户编写《客户验收报告》，双方在《产品交接验收单》上签字确认

对于不同类型的项目，验收方法及流程可能不一样，各个公司也有自身的特点来制定不同的验收方法。如下是对不同类项目的验收流程或方法说明，供读者参考。

- 针对具体行业的产品类项目，在试点单位运行之后，用户验收通过，可以认为项目结束。
- 通用类产品，比如：文字处理软件、杀毒软件等。在通过最多三次 Beta 版公测之后，即可进行发布，或者确定候选发布版。
- Web 网站的开发，可以采用原型法，在公测过程中逐步完善相关功能，至客户认可。

若满足以下条件，则可以通过客户验收：试运行过程中发现的所有缺陷均已得到解决；产品满足用户需求，并得到用户确认；《客户验收报告》已经生成，并得到客户的确认。在系统验收之后，交付给公司的维护人员负责，在维护期间发现的一些缺陷或问题，由维护人员负责处理，必要的时候可以让研发人员给予配合。

第17章 项目总结

内容提要：

- 项目总结简述
- 代码复用总结
- 项目结项

合同执行，作为市场项目整个生命周期的一个特定阶段，单从工程实施或技术开发角度看，阶段结束的主要标志是：

- 合同类项目——合同所含的所有工程都通过竣工测试，每个工程均有用户正式确认的竣工报告；
- 新产品项目和产品升级类项目——通过系统测试，系统试运行取得用户正式确认的验收报告，或通过 Beta 测试，其中发现的缺陷均得到处理。

当然，这是理想情况。实际上还可能有其他的情况发生，造成（工程或开发）执行阶段结束或项目终止，例如：

- 合同包含的多数工程已竣工，个别工程由于各种原因无法在一定期限内实施，双方商定排除这些个别工程；
- 开发类项目的部分（功能）目标由于各种原因无法按期实现，双方确定作为下一期项目的附加目标而先结束当前的项目；
- 测试已通过，必须补救的问题也已解决，合同执行阶段的实质性工作已经停止，但尚未取得用户正式确认的竣工报告或验收报告或确认意见；
- 其他各种因素造成项目失败，非正常终止项目计划甚至终止合同。

应特别强调，不论以哪一种方式结束项目执行阶段，从整个生命周期看，事情并未了结，接下去必须做的就是项目总结，这个阶段也称项目结束阶段。

17.1 项目总结简述

软件通过验收测试（对合同项目而言），或者通过系统测试和用户试用（对新产品项目和产品升级项目而言），并不表示项目结束（从项目管理的角度看），只是表示可以进入软件生命周期的下一个阶段或项目生命周期的最后一个阶段——项目总结阶段。

项目总结目的是：对项目的有形资产和无形资产进行清算；对项目进行综合评估；总结经验教训等；机构过程资产积累。具体的讲，项目总结的目的为：

- 通过项目分析、总结和会审，对项目工作进行评价，使项目组的经验成为机构过程资产，并促进软件过程的不断改进。
- 通过技术归档，为公司加强知识产权保护提供了依据，不断增加公司的技术积累。
- 通过技术交接，为做好产品进入市场后所必需的产品维护和客户服务做好必要的准备。
- 通过产品会签和发布，确保公司向用户提供符合市场需求的软件产品。

虽然，项目结束很少对项目本身的成败能产生重大影响，并且在此阶段项目相关各方（用户、项目组、管理层等）对项目的评价以及对项目成败原因的认识可能很不一致，只有通过总结，形成共识，才能妥善解决项目结束后必需面对的诸多问题，例如：

- 丰富公司资产库；
- 产品投入正常使用，减小公司应承担的售后服务压力；
- 建立与用户的长期合作关系；
- 项目团队及每一个相关人员的绩效评价；
- 项目管理的成功经验和失败教训作为无形资产长期积累；
- 项目成果的进一步产品化，已有产品的进一步商品化；等等。

17.2 代码复用总结

17.2.1 代码复用简介

代码复用就是把现有的代码、算法、方法、思想、技术等，拿到当前项目中加以利用。一般是直接使用或调用，不作修改。代码复用一种是同一项目内的复用，另一种是不同项目间的复用。代码复用总结的目的是收集项目中的完成特定功能的代码，编辑成册，为其他项目组提供参考和借鉴，减少后续系统的开发工作，减少相同功能代码的开发工作。既提高了效率，又节约了成本。

代码复用主要有两种形式，即二进制代码复用与源代码复用。前者是通过创建和使用对象来实现的；后者，顾名思义，是通过继承实现的，后者在 C++等面向对象的语言中被广泛使用。最直接的一个例子，Java、DotNet 提供了大量的类库，比如 XML 存取、WebService 实现、WinForms、WebForms，其实严格来说，这些东西与虚拟机有什么关系吗？即使全部关系。去掉，都是完全可以的，这些库与 Java、DotNet 语言开发没有任何关系。

写出可以重用的代码对开发者的要求非常之高。代码重用可遵守如下原则：

- 代码重用要先从在当前项目中实现代码重用开始。
- 应该从小模块开发。
- 可重用的代码一定跟业务无关，跟业务相关的代码无法重用。到了代码阶段只有算法和逻辑，不要将业务引入代码重用。
- 对接口编程。
- 优先使用对象组合，而不是类继承。
- 将可变的部分和不可变的部分分离。
- 减少方法的长度；消除 case / if 语句；减少参数个数。
- 类层次的最高层应该是抽象类。
- 尽量减少对变量的直接访问。
- 子类应该特性化，完成特殊功能。

- 拆分过大的类：作用截然不同的对象应该拆分。
- 尽量减少对参数的隐含传递。

一般有两种代码复用技术，一是改写类的实例方法；二是把参数类型改成接口，分别介绍如下：

1. 改写类的实例方法

任何方法，只要它执行的是某个单一概念的任务，就其本身而言，它就应该是首选的可复用代码。为了重用这种代码，我们必须回归到面向过程的编程模式，把类的实例方法移出成为全局性的过程。为了提高这种过程的可复用性，过程代码应该象静态工具方法一样编写：它只能使用自己的输入参数，只能调用其他全局性的过程，不能使用任何非局部的变量。

2. 把参数类型改成接口

代码复用真正的要点在于通过接口参数类型利用多态性，而不是通过类继承。从技术上说，可重用的是方法，而不是传递给方法的对象。选择最简单的参数接口类型；描述参数对象要求的接口越简单，其他类实现该接口的机会就越大。

17.2.2 代码复用活动

要想做到代码复用，须有几个前提：一是，统一的代码规范；二是，员工高度的代码复用意识；三是，员工的编码水平；四是，公司和员工级的高水平的代码库。

一般代码得用实施会经验两个阶段，分别为：第一阶段，复用以前的代码：一方面每个项目组的开发人员，在开发过程中，从公司或个人的代码库中提取可以直接利用的代码；另一方面，参考代码库中的代码和思想，加以修改和利用，写出适用于当前项目的代码。第二阶段，提取代码以备复用：每个项目成员总结自己开发过程中用到的算法、类、方法、函数等可以被其他项目借鉴或复用的代码段。按用途、功能等分门别类，撰写《代码复用总结》。

在进行代码复用时，可以按如下过程操作：

- 了解代码库：每个项目成员在了解了项目设计之后，开始代码编写之前，快速浏览公司级的代码库，可以看分类部分，主要目的是了解有哪些方面的代码库，为以后

开发过程中能记得公司的代码库是否有类似的代码作准备。

- 了解代码规范：熟读本公司的代码规范，在开发过程中严格按代码规范来编码，改变自己编码的习惯。
- 了解代码复用原则：在开发过程中，通过学习，了解代码复用的原则，尽可能地以此为准来编写代码。
- 参考代码库来编写代码：在编写代码过程中，可以借鉴公司或个人的代码库来编写代码，通过自己了解的方法，活用这些以前编写好的代码。
- 开发过程中的交流：在开发过程中，与同事交流，参考同事的同功能的代码来编程。
- 开发结束后的总结：在开发结束后，总结自己的代码，提取可以借用的部分，作为公司或个人的代码库。项目成员可以用讨论的方式来总结代码的提取。这样有利于代码的提炼。提取的代码要有完备的代码说明，有举例等。
- 评估复用代码：项目成员在总结出自己的可复用代码之后，项目经理应组织人员对可复用的代码进行评估，去其糟粕取其精华，得到真正的可以复用的代码，并提交给公司代码库的相应管理人员。
- 管理公司级的代码库：EPG 人员对公司级代码库进行管理，增加、备份和提取管理。

在项目结项时，都需要对项目中的代码进行复用总结，一般代码复用总结的工作流程如下：

1. 每个项目成员总结自己开发过程中用到的算法、类、方法、函数等可以被其他项目借鉴或复用的代码段。按用途、功能等分门别类，撰写《代码复用总结》。要求能清楚的知道代码的功能，用处，实现的算法，测试的方法等。
2. 项目经理组织项目成员讨论每个成员的《代码复用总结》，提炼出真正有意义的可复用的代码，编写出本项目的《代码复用总结》。
3. 评审《代码复用总结》：项目经理组织项目成员对《代码复用总结》进行评审
4. 通过评审后，将项目的《代码复用总结》，提交 EPG，作为过程改进的项目贡献，由 EPG 整理，收入机构级代码复用库。

17.3 项目结项

1.结项准备

项目经理与项目组成员、质量保证工程师、配置管理员共同收集并汇总项目执行过程中产生的数据，完成下列事项：项目经理撰写《结项报告》；质量保证工程师撰写《QA 总结报告》。QA 经理、配置管理员作如下验证：

- QA 经理对《结项报告》 、《QA 总结报告》之间的相关数据做有效性、正确性、一致性检查；
- 配置管理员对《结项报告》、《QA 总结报告》与该项目的配置库、度量数据库的数据做有效性、正确性、一致性检查；
- 验证完成后，如果发现问题，项目经理组织项目组成员进行分析解决，然后修改报告，转向下一步。

2.结项评审

为了结束项目，项目经理须把《结项报告》、《QA 总结报告》提交给总工程师。然后进行结项评审，结项时的评审可以按如下步骤进行：

1. 总工程师发起结项评审会，具体过程参见管理评审过程。参加评审人员包括：总工程师、EPG、项目经理、项目组成员、质量保证工程师、配置管理员等。
2. 项目资产检查与处理，参加评审人员在结项评审会上检查该项目的有形资产和无形资产，并和项目经理共同商讨如何有效地利用这些资产。
3. 项目综合评估，在过程改进时用项目度量数据库中的量化的指标作为评判项目的依据。主要包括：项目性能指标、过程质量、产品质量、项目需求、项目风险、生产率、资产积累。
4. 总结经验教训，结项评审会上项目组成员共同总结经验教训，添加在《结项报告》中，将其充实进机构级的过程资产库共享。
5. 参加评审人员在《结项报告》附录中签署意见并签字，并交付给总工程师。

6. 总工程师审阅资料。总工程师批准，项目正式结项，否则项目组修改资料并重新申请。

最后，项目配置管理员依据《结项报告》把项目资产移交给机构级的 CMG 组长。机构级的人员（EPG 组长、配置管理员、质量保证工程师）把项目移交的资产确认后纳入机构级的过程资产库中（更新 OMR、OPAL、风险库、检查列表库等，具体操作方法及包含的内容将在第 20 讲解）。

第18章 产品及过程质量保证

内容提要：

- **CMMI** 对应实践
- **PPQA** 简述
- **PPQA** 活动内容

过程和产品的质量保证活动贯穿项目生命周期的全过程，有二个特定目标：

- 客观评价项目组执行的过程及其工作产品或服务相对于适用的过程描述、标准和规程的符合性和不符合性；
- 客观分析不符合项，跟踪不符合项直到解决。

所谓客观，有这么几层含义：

- 过程和产品质量保证活动的执行者，习惯上独立于项目组，由专职的质量保证工程师（质量保证工程师）承担。项目组与质量保证工程师不能是领导与被领导的关系。
- 评价的标准是由 EPG 制定的机构标准过程集 OSSP 以及相关的过程资产；项目组实际执行的过程，可能与机构标准有差异，但必需符合机构制定的“裁剪指南”的规定。
- 评价、审计活动的计划以及活动内容（检查单）事先制定并公开。

显然，EPG 和过程改进管理部门是标准的制定者，产品研发部门和项目组是标准的执行者，而过程和产品质量保证工程师及其管理部门则是标准的监督者，三者的互相促进、互相制约的关系正好体现了西方发达国家“三权分立”的社会架构。

【说明：对于第三类和第二类学员的班级，在实训的时候，如果设置了质量保证工程师角色，本章的内容可以调整到第 6 章及第 7 章之间讲解。】

18.1 CMMI 对应实践

在 CMMI 中有一个专门的过程域对应于产品及过程质量保证，即产品及过程质量保证（Process and Product Quality Assurance，简称 PPQA），其目的是向项目组成员和管理部门提供对（项目实际执行的）过程及其工作产品的客观评价。

在项目整个生命周期内，通过向项目组成员和管理者提供有关过程和相关工作产品不同程度的可视性和反馈信息，以支持高质量产品和服务的提交。过程和产品质量保证过程域的实践确保计划的过程被实施，而产品验证过程域（VER）的实践确保满足指定的需求。这两个过程域有时会从不同的角度出发处理相同的工作产品，项目组应该注意尽量减少重复工作量。

过程和产品质量保证评估活动的客观性是项目成功的关键。客观性是通过独立性和使用标准来实现的。质量保证人员习惯上独立于项目组，但也有的机构将质量保证纳入过程而强调同行评审。质量保证应该开始于项目的早期，比如在建立计划、过程、标准时，有利于项目更好地符合机构方针和项目需求。

SG1 Objectively Evaluate Processes and Work Products（客观评价过程和工作产品），已执行的过程及其相关工作产品和服务相对于所用的过程描述、标准和规程的符合性得到客观的评价。

SP1.1 Objectively Evaluate Processes（客观评价过程），基于采用的过程描述、标准和规程，客观地评价指定的已执行过程。一般会产生如下工作产品：评价报告，不符合项报告，纠正措施。可以通过如下几步来完成该实践：一是创建鼓励员工参与确定和报告质量问题的环境（项目管理的一部分）；二是建立和维护评价准则；三是采用准则评价已执行过程，看是否遵循过程描述、标准和规程；四是识别在评价过程中发现的每一个不符合项；五是识别经验教训，以便改进未来产品和服务的过程。

SP1.2 Objectively Evaluate Work Products and Services（客观评价工作产品和服务），基于采用的过程描述、标准和规程，客观地评价指定的工作产品和服务。一般会产生如下工

作产品：评价报告，不符合项报告，纠正措施。可以通过如下几步来完成该实践：一是基于文档化的采样准则，选择将被评价的工作产品；二是建立和维护工作产品评价标准；三是在工作产品评价过程中使用已建立的标准；四是在工作产品提交给客户前评价工作产品；五是在工作产品的选定里程碑处评价工作产品；六是基于过程描述、标准和规程，对工作产品进行连续或增量的评价；七是识别在评价过程中发现的每一个不符合项问题；八是总结经验教训，以便改进未来产品和服务的过程。

SG2 Provide Objective Insight（提供客户的认识），不符合项得到客观地跟踪和沟通，并确保得到解决。

SP2.1 Communicate and Ensure Resolution of Noncompliance Issues（沟通并确保解决不符合项），通过项目组内部或与管理者之间的沟通，确保解决不符合项和经趋势分析而发现的质量问题。一般会产生如下工作产品：纠正措施报告，评价报告，质量趋势报告。可以通过如下几步来完成该实践：一是尽可能多的项目组成员参与解决不符合项；二是记录项目组内不能解决的不符合项；三是把项目组内不能解决的不符合项提交给上层管理部门；四是分析不符合项，以判断是否存在应该指出并处理的质量趋势；五是确保项目相关各方及时地意识到评价的结果和质量趋势；六是定期地与负责接收和管理不符合项的管理部门一起评审不符合项和趋势；七是跟踪应解决的不符合项。

SP2.2 Establish Records（建立记录），建立和维护质量保证活动的记录。一般会产生如下工作产品：评价日志，质量保证报告，纠正措施状态报告，质量趋势报告。可以通过如下几步来完成该实践：一是尽量详细地记录过程和产品质量保证活动，以便尽可能了解其状态和结果；在必要时修订质量保证活动的状态和历史。

18.2 PPQA 简述

结合 CMMI 的相关实践可以看出质量保证工程师的作用主要有：

- 帮助产品研发部门和项目组选择合适的过程描述、标准和规程，遵循裁剪指南，定义项目过程。

- 按计划进行质量保证活动（如，评审 Review、检查 Inspection 和审计 Audit），客观评价项目组实际执行的过程及其工作产品相对于适用过程、标准和规程的符合性和不符合性。
- 分析不符合项及质量趋势，向项目组和上极管理者报告审计结果和发现的问题，跟踪问题的最终解决。

可见，执行过程和产品质量保证的工程师应该具备比较好的项目管理能力、执行能力和专业水平。可惜，实际上许多企业往往不太重视这一点。当然，这里也有一些认识上的问题，例如：

- 软件产品质量与软件过程质量是二个不完全一样的概念。软件产品的质量目标决定了软件过程的质量目标；软件过程的质量为软件产品的质量提供了保证。好的过程质量不等于就一定有好的软件质量，不好的软件过程则很少能有好质量的软件产品。
- 软件质量工程师与质量保证工程师的工作侧重点有所区别：软件质量工程师的很大一部分工作由软件测试工程师承担，另外一部分则由参与各类工作产品评审、同行评审和审批的人员承担；质量保证工程师的主要工作目标是保证过程和产品相对于标准的符合性，质量保证工程师不是与项目组相对立的监督者，而只是从独立、客观的角度将项目工程活动过程及其结果的实况反映给项目组及各级管理者(所谓“眼睛和耳朵”的作用)，让管理者及时了解过程及产品与标准之间存在的偏差，及时纠正，因此是管理者特别是项目组长的协助者。
- “质量保证工程师不保证质量”，也就是说，质量保证工程师发现不符合项，并跟踪不符合项问题的解决，但纠正不符合项本身是项目组的责任。
- 质量保证工程师的活动也有一个质量保证问题，包括：
  - ✧ 未发现不符合项；
  - ✧ 只注意发现、不重视分析；
  - ✧ 纠正不符合项的建议未能妥善兼顾产品质量、成本进度、用户满意和项目组员满意等多方面因素；

- ✧ 不善沟通，与项目组相对立；
- ✧ 未能兼顾严格执法与适度灵活的关系，等等。

在企业里执行产品及过程质量保证活动时，一般会制定一些必须遵循的方针，针对本书案例的研发机构，制订的质量保证方针如下：

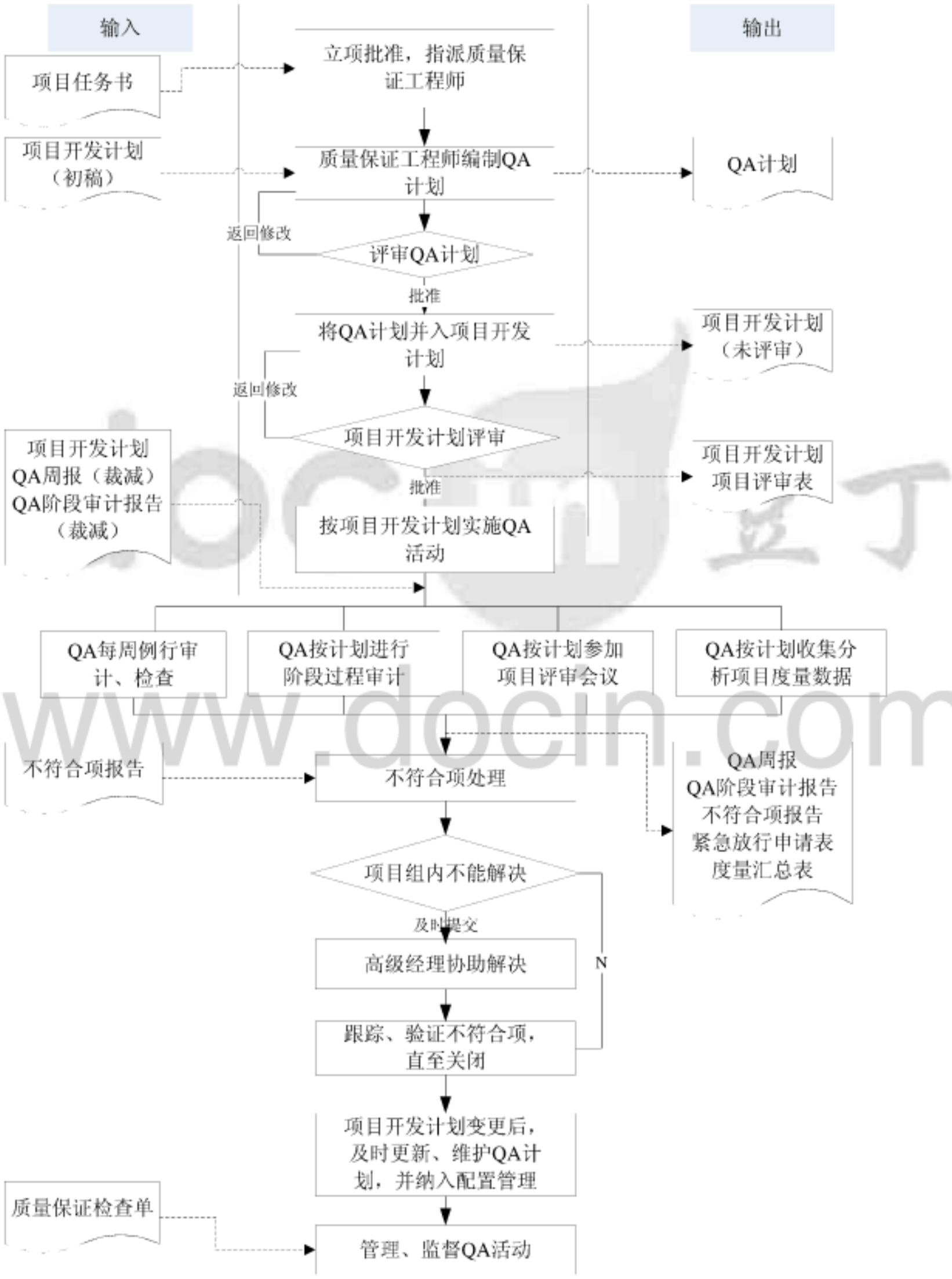
- ■ 总工程师为每个项目指定质量保证工程师负责项目的 QA 活动，进行过程检查与监督。
- ■ 在项目计划阶段，质量保证工程师编制质量保证计划，计划经总工程师确认后，做为项目开发计划的一部分，一起提交评审、确认和批准并纳入配置管理。
- ■ 质量保证工程师的活动应坚持独立、客观的原则，以既定的质量保证计划、标准和规程为基准，客观的评价和报告软件过程活动和工作产品相对于规范、规程和标准的符合性和不符合性。
- ■ 质量保证工程师审计主要针对生成工作产品的活动过程而不是产品所采用的技术，并将审计结果及时报告相关组和个人，并有向上级直接报告的渠道。
- ■ 质量保证工程师的主要职责是检查软件过程活动和工作产品的偏离、不符合项，并按规定的方式、步骤报告跟踪不符合项直至关闭。
- ■ 总工程师或 QA 经理及时处理项目组内部解决不了的不符合项问题，并定期审查质量保证工程师的活动和结果。

质量保证的目的是为了有效的实施软件项目的质量保证工作，客观验证软件过程活动和工作产品对适用标准、规程和需求的遵从性，保证项目过程活动和工作产品的可视性，让管理者及时了解实际过程活动情况，确保项目质量与计划保持一致 。质量保证内容：

- ■ 制定项目的质量保证计划；按计划执行 QA 活动；
- ■ 客观验证软件过程活动和工作产品相对于标准、规程和需求之间遵从性，以及不符合项的报告、解决、跟踪直至关闭；
- ■ 质量保证计划随着项目开发计划的变更及时更新维护；
- ■ 总工程师/研发部经理、QA 经理定期监督、检查 QA 活动。

在项目开发过程中，QA 相关的活动可以用图 18-1 来表示。

图表 18-1 产品及过程质量保证活动流程图



18.3 PPQA 活动内容

18.3.1 制定质量保证计划

项目立项审批通过后，QA 经理为项目指派质量保证工程师。在项目立项后的两周内（根据项目规模的大小而定，小规模的项目可以一周），质量保证工程师在项目开发计划初稿的基础上，编制项目的质量保证计划初稿。

质量保证计划的主要内容有，详见《质量保证计划》模板：

- QA 工作的目的、范围、工作职责和权限；
- QA 工作在项目中的活动资源（人员、培训、设备和工具等）；
- 质量保证工程师在项目组中的各项活动的活动内容和时间表；
- 确定本项目每周 QA 例行检查时间；确定质量保证工程师或 QA 经理独立向上报告的途径、与相关项目组或个人的通报方式；确定须进行的过程检查、过程检查表的内容及依据标准；确定项目进行质量检查的工作产品；
- 确定项目需收集的度量表格；确定检查结果的保存方式；

在项目计划阶段，根据细化的《项目开发计划》，质量保证工程师调整质量保证计划，完成后提交 QA 经理审核；审核同意后并入项目开发计划，作为项目开发计划的一部分与项目开发计划一起提交相关组或个人评审，质量保证工程师关注评审中提出的问题和意见，必要时对质量保证计划进行修订。

评审后形成的评审文档和质量保证计划、项目开发计划，由项目经理递交给配置管理员纳入配置管理。

18.3.2 实施 QA 活动

质量保证工程师按既定的质量保证计划（或项目开发计划），完成以下工作内容：

- 每周例行审计、检查软件过程活动和工作产品；

- 按计划进行软件过程/工作产品的阶段审计；
- 协助项目经理组织并参与项目评审会议；
- 收集和分析项目度量数据。

整个项目生命周期过程中，质量保证工程师形成三类文档，QA 周报、不符项报告、QA 阶段审计报告，从立项后开始，一直到项目总结结束。具体的执行方法如下：

一、**QA** 每周例行活动

- 在项目计划之前，则根据项目开发计划/质量保证计划初稿进行，项目开发计划书评审通过后根据项目开发计划/质量保证计划进行，每周对项目进行例行检查，形成 QA 周报，有不符合项时，则同时编写不符合项报告提交项目经理、总工程师、项目组成员、以及相关组或个人。
- 每周检查完毕，填写 QA 活动度量表收集 QA 每周检查总人时、不符合项解决总人时、验证总人时等度量数据，及时填写在《项目度量数据库》中（具体包含内容请参见第 19 章的相关内容）。
- 把 QA 周报和不符合项报告提交配置管理员纳入配置管理。
- 检查的依据为 QA 周报模板中的检查要点，检查要点详见《QA 周报》模板 。

二、**QA** 阶段审计

审计，是质量保证活动的一项基本形式。实际执行时，QA 审计活动往往与基线审计、里程碑评审相关联，只是目的、内容有区别，如下表所示：

时间点	策划阶段结束	设计阶段结束	集成测试结束	系统测试结束	技术归档结束
基线建立	✓	✓	✓	✓	✓
基线审计	✓	✓	✓	✓	✓
QA 审计	✓	✓	✓	✓	✓
里程碑评审	✓	✓	×	✓	×

在生命周期不同阶段的 **QA** 审计活动，有不同的重点：

- 项目计划阶段结束 QA 审计内容：需求阶段的工作是否按计划进行；验证《用户需

求说明书》和《软件需求规格说明书》是否符合模板要求；验证软件需求评审、需求规格说明书同行评审过程是否符合规程。项目计划是否按计划进行；《项目（开发）计划》是否符合模板要求；《项目（开发）计划》的评审过程是否符合规程。

- 设计阶段结束 QA 审计内容：概要设计和评审是否按计划进行；《概要设计说明书》是否符合模板的要求；评审过程是否符合规程；详细设计和同行评审是否按计划进行；《详细设计说明书》是否符合模板要求；评审过程是否符合规程。
- 编码阶段结束 QA 审计内容：编码和代码同行评审是否按计划完成；验证代码同行评审的过程是否符合规程。
- 测试阶段结束 QA 审计内容：测试阶段工作是否按计划完成；测试计划、测试规格说明书、测试记录、测试报告是否符合模板要求；测试计划和测试规格说明书的同行评审过程是否符合规程；集成测试用例和规程是否符合模板要求；测试报告的确认/批准是否符合规程。
- 项目总结阶段结束 QA 审计内容：验证最终产品生成过程是否符合规程；验证各类手册（维护手册、用户手册）是否符合模板要求，同行评审的过程是否符合规程；验证项目总结报告是否符合模板要求，评审过程是否符合规程。
- 配置管理（各阶段均涉及）的 QA 审计内容：配置项的标识、状态和存放的路径是否符合规程的要求；配置状态报告填写是否及时；配置项的变更是否受控，变更是否留有记录，如：需求是否有变更，因变更引起的相关约定是否已通过协商，并按改变后的约定执行，变更后的工作产品是否纳入配置管理，并保留变更记录；《项目开发计划》的变更是否符合规程等；基线生成表和审计报告是否符合模板，基线生成过程和基线审计过程是否符合规程。

QA 阶段审计的具体执行过程为：

- 根据项目开发计划中裁剪的项目过程和生命周期，在各个生命周期阶段结束之前进行 QA 阶段审计，尤其在基线审计后，里程碑评审前 QA 阶段审计更加关键，审计不通过则项目无法继续开展 。

- QA 阶段审计主要审计的内容为项目开发过程中的过程活动及其过程中的产生的相关工作产品，验证其是否符合机构制定的标准、规程和模板的要求。质量保证工程师按项目开发计划/质量保证计划进行阶段审计，审计要点详见《QA 阶段审计报告》中的各阶段检查表，不同的软件过程阶段采用不同的阶段检查表 。
- 质量保证工程师在检查后，及时将 QA 阶段审计报告和不符合项报告（有不符合项时生成），递交给项目经理、项目组成员、研发部经理、总工程师、以及相关组或个人；必要时，质量保证工程师协助项目经理向项目组成员或相关人员说明发现的不符合项问题，取得共识并采取相应措施 。
- 填写《项目度量数据库》中的相关数据。

三、协助并参与项目评审

按项目开发计划/质量保证计划参加项目评审会议，客观公正地验证评审会议是否按照机构制定的规程、标准进行，将评审过程及审核结果记录到 QA 周报，若发现不符合项时，则记录到不符合项报告中。步骤如下：

- 在评审会议之前，进行评审资料预审检查工作产品是否符合规程、标准或文档模板的要求，并填写预审问题清单反馈给项目经理。
- 参加评审会议，检查评审过程是否符合规范，如评审相关人员是否参加；评审是否指定了主持人、记录员；主持人是否已经熟悉评审产品/过程活动的相关内容等。
- 评审会议后，检查项目经理提交的《项目评审表》中的缺陷记录等是否填写正确，并签字确认。
- 在指定的复审日期检查《项目评审表》中的缺陷是否按时解决并经过验证人验证，确认后填写质量保证工程师意见。
- 填写《项目度量数据库》中与评审相关的数据。
- 在正式评审结束后，在每周例行检查时对项目评审过程是否符合机构制定的规范进行评价，形成 QA 周报，发现评审中产生的问题，持续改进评审流程。

1. 收集和分析项目度量数据

质量保证工程师协助制定项目的度量目标、度量数据的来源、收集频度、度量标准及所需资源等，经评审通过后，由项目经理和质量保证工程师负责实施。

项目经理侧重于收集和分析项目规模、进度、工作量、成本、风险等方面的度量数据。质量保证工程师则侧重于收集和分析项目评审、缺陷、QA 活动工作量等方面的度量数据。步骤如下：

- 根据项目开发计划/质量保证计划定期收集项目度量数据，填写项目度量数据库。
- 填写完成后，检查度量数据是否收集正确，保证数据的完整性和正确性。如发现当前度量汇总表不能完整收集本项目度量数据，则填写变更申请表，提交总工程师审核。
- 质量保证工程师按计划执行每周例行检查、阶段审计、不符合项报告验证等活动后，及时填写项目度量数据库中的 QA 活动度量表。具体内容详见 QA 活动度量表。
- 质量保证工程师在项目评审缺陷验证解决后，填写项目度量数据库中的项目评审度量表。
- 质量保证工程师定期或事件驱动分析度量数据，形成文档提交项目经理、研发部经理、总工程师及其他相关人员通报度量结果，以取得支持决策和采取有效的纠正措施，同时为过程持续改进提供基础。

18.3.3 不符合项处理

质量保证工程师在每周例行检查、阶段审计或参加项目评审过程中发现的不符合项，须按照以下步骤进行及时处理。

- 及时记录不符合项，并进行编号
- 识别不符合项的严重等级
- 不符合项的分类
- 不符合项的处理
- 特殊情况处理

具体的处理步骤如下：

1. 质量保证工程师在审计或评审结束后，及时整理形成 QA 阶段审计报告/或 QA 周报，有不符合项时同时填写不符合项报告，提交项目经理、研发部经理、总工程师或相关组及个人；
2. 质量保证工程师协助项目经理识别不符合项报告中的不符合项，对不符合项取得共识并进行分类，并进行编号；
3. 根据不同情况对不符合项进行分类处理，结果可以有（解决、不能解决、拒绝）。
4. 不符合项处理过程：
   - 项目经理根据不符合项报告，及时与项目组成员及相关人员进行分析商量，及时采取措施，确认问题的修改方式、责任人，修改完成日期和再次审核日期 ，记录到《不符合项报告》中；
   - 质量保证工程师进行不符合项处理的跟踪、验证，确认不符合项是否已经关闭；
   - 一般在项目组内进行沟通，达成一致处理意见，若在项目组内不能解决时，质量保证工程师需要提交研发部经理/总工程师协助解决；
   - 总工程师/研发部经理收到质量保证工程师提交的不符合项报告后，应及时和项目组进行沟通、协调，制定限期整改计划，并反馈给质量保证工程师 ，质量保证工程师跟踪直至不符合项得到解决 ；
   - 收集项目度量相关数据，在项目度量数据库中的 QA 活动度量表中记录不符合项的检查、解决、验证的总工作量

在实际开发过程中，比较常见的不符合项有以下几种：

- 没有根据个人周报、项目组周报及时更新项目进度表；
- 没有进行需求、成本、关键计算机资源等内容的跟踪；
- 评审发现的缺陷、不符合项没有按时解决处理；
- 配置项的放置、标签方法等没有按规程处理；
- 没有按计划完成工作且没有文档化的陈述；

- 没有按计划进行评审且没有具体的理由；评审没有按规范、没有评审记录；
- 没有按期举行项目例会、没有会议记录。

18.3.4 维护质量保证计划

对质量保证计划进行维护管理和变更控制，保持质量保证计划和项目开发计划的一致性。具体步骤如下：

- 当项目开发计划发生变更时，质量保证工程师对质量保证计划进行相应的变更控制，保持与项目开发计划一致。
- 质量保证计划的变更必须得到总工程师和项目经理的确认。
- 调整并确认后的质量保证计划，质量保证工程师需要及时通知相关组或个人。（如配置管理员，项目组成员，研发部经理或总工程师）。
- 变更后的质量保证计划由质量保证工程师提交配置管理员纳入配置管理。

质量保证工程师的日常工作、项目组对 QA 活动的支持、项目经理和 QA 经理对质量保证工程师工作的监督，可以很好的保证质量保证工程师有效的开展活动。一般至少包含：

- 为项目组提供有关质量保证的培训。在项目初期或进展过程中，为使 QA 活动能够有效的实施，根据实际情况，由项目经理协助质量保证工程师对项目组成员或相关人员进行有关质量保证工程师义务和活动的培训 ，可以包括不符合项处理、跟踪监督等内容。
- 总工程师或 QA 经理定期检查质量保证工程师活动。定期评审项目软件质量保证过程与方针；定期检查质量保证工程师提交的 QA 周报、QA 阶段审计报告、不符合项报告等 QA 活动是否按照质量保证计划和机构制定的 QA 规程及其他相关规程执行 。

第19章 度量分析

内容提要：

- **CMMI** 对应实践
- 度量分析简述
- 度量活动内容
- 项目度量数据库结构

在 CMMI 中进一步强化的数据度量与分析的相关过程域，把其提升到 CMMI2 级中，通过数据的收集、度量、分析，方可为整个公司的过程改进提供基础数据，为决策提供科学的参考依据，更是公司量化管理的基础。

将度量与分析活动纳入项目进程，可为项目的以下活动提供支持：项目计划和估计；根据已建立的计划和目标跟踪过程的实际性能；确定和解决与过程有关的问题；为将来把度量集成到其他过程提供基础。度量能力可以体现在单个项目中，也可以体现在其他机构职能中（如质量保证）。度量活动的最初焦点是在项目一级。然而，事实表明度量能力对解决机构以及企业一级的信息需求也非常有用。项目组可以将项目的度量数据先存放在项目特定的数据库中，当度量数据用于其他项目共享时，则可将其存入机构一级的度量数据库。

19.1 CMMI 中对应实践

在 CMMI 中有一个专门的过程域（PA）对应于度量分析，其目的开发和维持测量能力，为各类管理活动提供必需的信息。初始的度量活动关注于项目级，但是度量能力对公司级信息获取及分析是非常重要的。度量分析过程域（简称，MA）包含如下内容活动有，配合各类管理活动的目的和对信息的需求，确定测量和分析的目标；制定度量定义、数据采集和存

储机制、分析方法以及报告和反馈机制；实现数据的采集、存储、分析和报告；提供客观结果，用作决策依据，必要时采取合适的纠正措施。共计要达到两个特定目标，执行 8 个特定实践，具体为：

SG1 Align Measurement and Analysis Activities（安排度量与分析活动），安排度量活动和目标，使其满足特定的信息需求和目标。

SP 1.1 Establish Measurement Objectives（建立度量目标），建立和维护从特定信息需求和目标衍生而来的测量目标。可以通过如下几步来完成该实践：一是文档化信息需要和目标；二是对信息需要和目标进行排序；三是记录、评审和更新度量目标；四是必要时为提炼和清晰信息需要和目标提供反馈；五是维护度量目标对信息需要和目标的可跟踪性。

SP 1.2 Specify Measures（定义度量，详细说明度量来说明度量目标），形成基本度量和派生度量规格说明。可以通过如下几步来完成该实践：一是基于文档化的度量目标确定可选度量指标；二是识别已有的涉及度量目标的度量指标；三是为度量指标制定操作定义；四是对度量指标进行排序、评审和修改。

SP 1.3 Specify Data Collection and Storage Procedures（定义数据收集及存储的具体过程），详细说明和定义度量数据是如何获得和保存的。一般会形成如下工作产品：数据采集和存储规则，数据采集工具。可以通过如下几步来完成该实践：一是识别从当前工作产品、过程或活动产生数据的已有数据源；二是识别哪些需要但是当前不用的度量指标；三是针对每一个度量指标，给出如何采集和存储数据的说明；四是创建数据采集机制和规则指南；五是在合适和方便时支持数据的自动采集；六是对数据收集和存储规则进行排序、评审和更新；七是在必要时更新度量指标和测量目标。

SP 1.4 Specify Analysis Procedures（定义分析过程），详细说明和定义度量数据是怎样分析和报告的。一般会形成如下工作产品：数据分析方法规格说明，数据分析工具。可以通过如下几步来完成该实践：一是制定应执行的分析活动和需要准备的报告，并对其排定优先顺序；二是选择合适的数据分析方法和工具；三是制定分析数据和沟通结果的管理制度；四是评审和更新分析和报告的内容和格式；五是必要时更新度量指标和度量目

标：六是制定评价分析结果以及度量和分析活动实施情况的准则。

SG 2 Provide Measurement Results（提供度量结果，此度量结果说明了已确定的信息需要和目标）

SP 2.1 Collect Measurement Data（收集度量数据），获取指定的度量数据。一般会形成如下工作产品：基本的和派生的度量数据集，数据完整性测试结果。可以通过如下几步完成该实践：一是获取基本度量数据；二是生成派生度量数据；三是尽可能结合数据来源检验数据完整性。

SP 2.2 Analyze Measurement Data（分析度量数据），管理和存储度量数据、度量规格说明和分析结果。可以通过如下几步来完成该实践：一是执行初步分析、解释结果，形成初步结论；二是在必要时执行附加的或事先未料想到的度量和分析；三是与项目相关各方一起评审初步结果；四是细化准则，用于未来分析。

SP 2.3 Store Data and Results（存储数据和结果），管理和存储度量数据、度量规格说明和分析结果，形成存储数据清单。可以通过如下几步完成该实践：一是评审数据以确保数据的完备性、完整性、准确性和现势性（currency）；二是使得只有合适的成员组和个人使用被存储的内容；三是防止被存储信息的不正当使用。

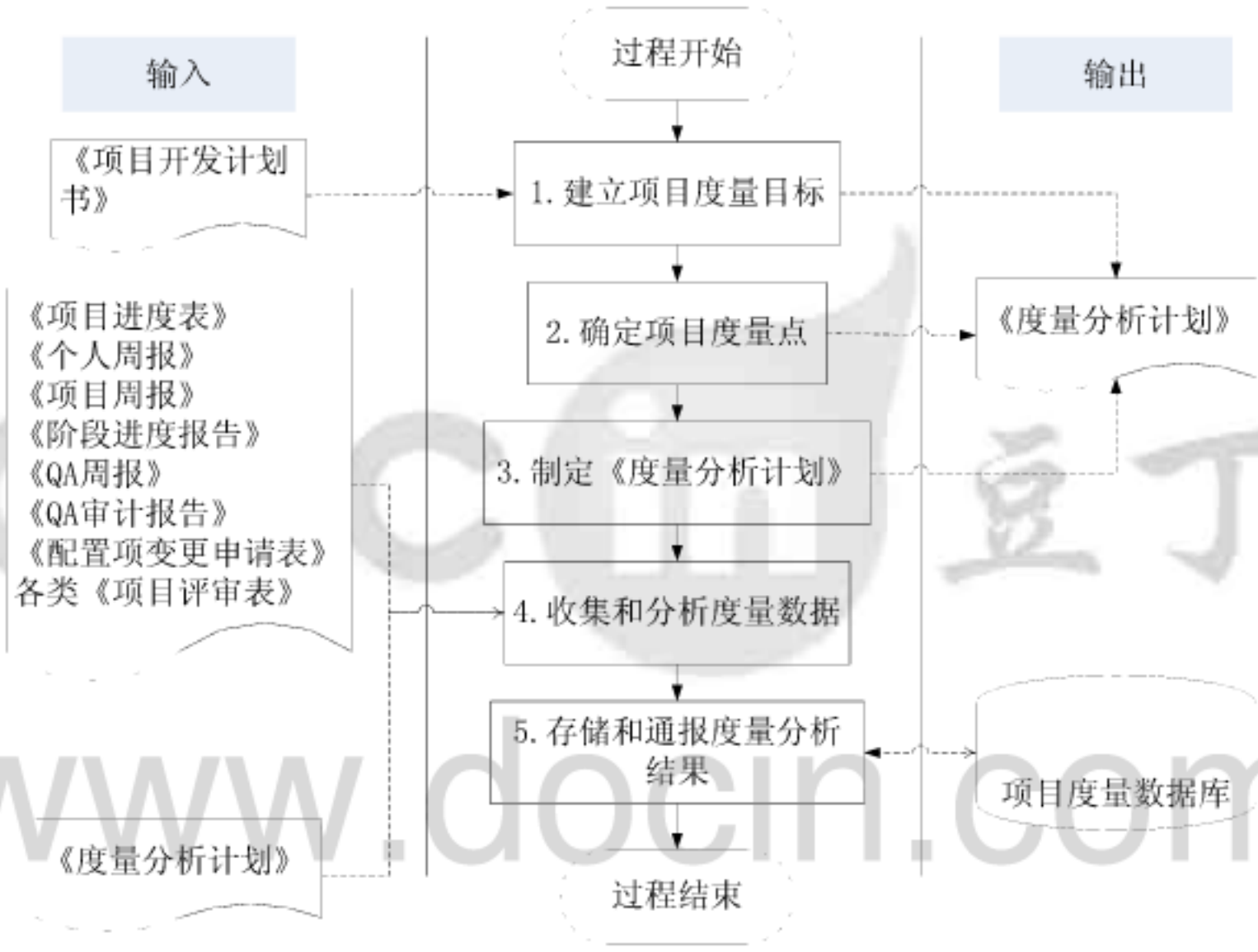
SP 2.4 Communicate Results（沟通结果），向所有的相关人员报告度量与分析活动的结果。一般会形成如下工作产品：提交的报告和相关的分析结果，有助于解释分析结果的背景材料或者指南。可以通过如下几步来完成该实践：一是使项目相关各方及时了解度量结果；二是帮助项目相关各方理解结果。

19.2 度量分析简述

度量分析过程的目的是开发和维护管理信息所需的度量能力。协调度量和分析活动，提供度量和分析的结果。建立项目过程数据库和丰富机构过程数据库。收集项目相关数据，并对数据进行分析，识别项目偏差，发现问题并及时做出相关预测。为公司内部相似项目估算提供参考依据。

度量分析活动贯穿项目生命周期全过程，在项目组实际执行度量分析之前，必须先进行度量策划，包括：确定度量范围和目标；定义度量元素及其规格说明；制定数据采集、存储规则；选定数据分析方法。通过策划，形成项目度量计划，以后的度量活动均以该计划为指导。度量分析在项目开发过程中的活动流程如图 19-1 所示。

图表 19-1 度量分析活动流程图



19.3 度量活动

19.3.1 建立项目度量目标

项目经理负责根据机构的度量分析相关要求， 以及机构标准度量目标和度量点，建立本项目的度量目标，并将其记录在《度量分析计划》中；项目的度量目标除了必须包含的所有的机构级度量目标外，还可以根据项目特点增加特有的度量目标；度量目标一般是在公司级的度量体系说明资料里进行描述。

项目经理负责根据项目度量目标和机构级的度量点定义，确定本项目的度量点，并将其记录在《度量分析计划》中；本项目度量点源于机构级度量点，必须包含其中的所有度量点，但不仅限于此，本项目可以根据自身特点增加新的度量点，或根据项目新的度量目标定义对应的度量点；机构级的度量点一般是在度量体系说明资料里进行描述。

19.3.2 收集和分析度量数据

项目经理、质量保证工程师、项目组成员在项目开展过程中，将与其相关的度量数据填写在以下文档中：

	文档	说明
1	《项目进度表》	由项目组成员定期更新其任务进度的完成情况，并由项目经理确认、接受后汇总形成更新后的整个项目的进度表。
2	《个人周报》	由项目组成员每周按照模板填写或在工具上生成。
3	《项目周报》	由项目经理每周生成
4	《阶段进度报告》	由项目经理在每一个里程碑评审前后生成。
5	《QA 周报》	由质量保证工程师每周生成
6	《QA 审计报告》	由质量保证工程师在每周生成。
7	《配置项变更申请表》	由配置管理员在每次配置相进行变更过程中生成。
8	各类《评审报告》	由指定的人员在各类评审后生成。
9	各类《测试报告》	由测试人员在各类测试结束后生成。
10	《项目度量数据库》	由项目经理或质量保证工程师定期对度量和分析数据进行汇总而生成。

在项目开发过程中，需要开始如下几类度量活动：日常度量活动、里程碑度量活动、变更度量活动、评审度量活动、项目结项度量活动。各项目可根据规模等实际情况对下述各项度量活动进行适当裁减，由项目经理在《度量分析计划》中写明。如项目规模较小，可不进行日常度量活动，但所有项目都必须有里程碑度量活动和项目结项度量活动。各类活动具体开展的方法及时机如下：

日常度量活动：每周项目例会前，由经理向项目组成员汇总一周工作执行情况及工作量，项目经理以此为依据编写《项目级周报》。其中包括本周工作量、人力投入状况。质量保证工程师每周在项目例会前出具《QA 周报》和《不符合项报告》。

里程碑度量活动：质量保证工程师在里程碑评审前生成《QA 阶段审计报告》。项目经理按照模板要求编写《阶段进度报告》，生成以下内容：《需求跟踪报告》及其汇总；工作量统计及偏差分析；进度统计及偏差分析；成本统计及偏差分析；本阶段的项目人员投入状况表；本阶段的不符合项和缺陷状况表。项目经理根据《阶段进度报告》和《QA 阶段审计报告》的结果，生成本阶段的《项目度量数据库》中的度量数据，并通告给质量保证工程师。质量保证工程师对《项目度量数据库》中的度量和分析数据进行有效性和完整性验证。项目经理确保在里程碑评审前将经过质量保证工程师验证过的《项目度量数据库》通报给相应的人员。

变更度量活动：当项目发生变更时，将变更过程记录《项目度量数据库》中，其中包含了变更阶段、变更对象和缺陷个数等信息。这样就可以方便地对项目的所有变更活动进行度量。

评审度量活动：每次对工作产品进行评审后，项目经理都要将评审的结果记录在《项目评审表》中，并将评审相关的度量数据汇总到《项目度量数据库》中。

项目结项度量活动：在项目结项时，EPG 要指定成员参加项目结项会议。项目经理要将整个项目阶段完整的度量数据记录在《项目度量数据库》中，供会议评审。在项目结项后，EPG 要指定成员将该项目完整的《项目度量数据库》中的数据加以汇总、分析和提取，纳入到《机构级度量数据库》中，并负责对新的《机构级度量数据库》进行分析，并将汇总和分析结果通报给总工程师，以便为公司提供直观、有意义的分析汇总结果，并为后续项目提供有价值的参考数据。

19.3.3 存储和通报度量分析结果

在里程碑点，项目经理根据《项目度量数据库》中的度量和分析数据完成《阶段进度报告》并通报给相关人员。项目经理确保在里程碑评审前将经过质量保证工程师验证过的《项

目度量数据库》通报给相应的人人，如总工程师、研发部经理、项目组成员等

在里程碑评审会议上，项目经理负责宣读本阶段的度量和分析结果数据，供总工程师及会议其它人员评审。

在项目结项时，EPG 要指定成员参加项目结项会议。项目经理要将整个项目完整的《项目度量数据库》在项目结项会议上宣读，供会议评审。

在项目结项后，EPG 要指定人员将该项目完整的《项目度量数据库》中的数据加以汇总、分析和提取，纳入到《机构度量数据库》中，并负责对更新后的《机构度量数据库》进行分析，并将汇总和分析结果通报给总工程师。

19.4 项目度量数据库结构

《项目度量数据库》一般由质量保证工程师或项目经理，每周或定期进行更新，其中数据的来源一般为《项目组周报》、各类阶段报告、评审表等，具体在模板里均有详细说明。但是在填写时应当注意三点：

- 白色框部分可以填写，其他颜色的框不需要填写数据，可以自动计算得到；
- 在《项目度量数据库》中出现最多的是“数据覆盖率”，这个指标是表示每类需要填写的数据，实际填写的比率，用来反映项目组数据收集能力的一个指标。比如：项目工作量统计表中需求开发阶段，共有 10 个数据应当采集填写，如果实际只采集了 6 个数据，那么数据覆盖率就为 60%。
- 项目度量数据库模板可以根据项目的裁剪进行调整。

19.4.1 项目综合数据表

项目综合数据表是项目度量点的集合，本页汇集了所有需要统计分析的度量数据，内容分为 8 类，分别是项目特征信息、项目性能指标、过程质量度量、产品质量度量、项目需求度量、项目风险度量、团队规模度量和资产累计度量。如果使用本书提供的 Excel 模板，这

些数据中除“项目特征信息”外，均可自动填充。

19.4.2 项目性能度量

可以从以下几个指标来衡量项目性能：项目工作量统计表；项目进度表；项目成本统计表；项目生产率统计表。其中几个比较重要的指标及含义解释如下：

- 工作量估算偏差：实际工作量-估算（计划）工作量。
- 工作量估算偏差率：$\frac{\text{实际工作量}-\text{估算（计划）工作量}}{\text{估算（计划）工作量}}$，是一个用来反映项目组对工作量估算能力的指标。
- 工期差异：实际工作日期-计划工作日期。
- 已完成工作量的实际成本（Aactual Cost of Workload Performed，简称 ACWP）：是指项目实施过程中某阶段实际完成的工作量所消耗的工时(或费用)，主要是反映项目执行的实际消耗指标。在软件开发项目中，我们可以用某个阶段实际所用工作人时来表示该指标。
- 已完成工作量的预算成本（Budget Cost of Workload Performed，简称 BCWP）：是指项目实施过程中某阶段按实际完成工作量按预算定额计算出来的工时(或费用)。在软件类项目中，一般使用工时来表示成本，所以 BCWP 为本阶段已完成的任务“计划工作量”之和，由于本阶段计划的所有任务不一定全部完成，所以 BCWP 与 BCWS 之间就有可能存在不一致，进度就会有偏差。
- 计划工作量的预算成本（Budget Cost of Workload Schedule，简称 BCWS）：是指项目实施过程中某阶段计划要求完成的工作量所需的预算工时(或费用)，主要是反映进度计划应当完成的工作量而不是反映应消耗的工时(或费用)。在软件项目是为计划某个阶段的所用工作人时。
- 进度偏差（Schedule Variation，简称 SV）：完成某阶段工作量实际用的人时与计划用人时的差。计算公式为，SV= BCWP-BCWS。
- 进度偏差率（Schedule Variation Percentage，简称 SVP）：SVP=（BCWP-BCWS）/BCWS

×100%。

- 进度绩效指标（Schedule Performance Index，简称 SPI）：SPI=BCWP/BCWS。
- 成本差异：由于在软件项目中软件开发成本一般是用工作量（人时）来表示成本的，所以在此与工作量估算偏差是一样的，即：成本差异=实际工作量-估算（计划）工作量。
- 成本偏差（Cost Variation，简称 CV）= BCWP – ACWP，即本阶段已完成的工作计划投入的工作量与实际投入的工作量之差。
- 成本偏差率(Cost Variation Percentage，简称 CVP)：CVP= (CV / BCWP) × 100%。
- 成本性能指标(Cost Performance Index，简称 CPI) = BCWP / ACWP。
- 工作量估算偏差率＝=（实际工作量-计划工作量）/计划工作量
- 人月成本=项目总成本/实际工作量
- 各阶段工作量分配百分比=各阶段实际工作量/实际工作量

【说明： SPI==1，且 CPI==1：表示项目的实际进展与计划完全符合；SPI>1，且 CPI>1 ：表示项目实际进展情况是“少花了钱多办了事”；SPI<1，且 CPI>1 ：表示项目实际进展情况是“少花了钱少办了事”；SPI<1，且 CPI<1 ：表示项目实际进展情况是“多花了钱少办了事”；SPI>1，且 CPI<1 ：表示项目实际进展情况是“多花了钱多办了事”。理想的区域是：0.95<SPI<1.08 ，且 0.95<CPI<1.08。】

为了更好的理解以上概念，举例如下：在修路时，按计划 2008 年 7 月 21 日-28 日一周时间修 10 公里，每公里预算费用为 1 万。在实际执行过程中，本周修了 12 公里，实际费用为 15 万元。那以，由此可以计算出以上的各个数据：ACWP=15 万元；BCWP=12 公里×1 万元/公里=12 万元；BCWS=10 公里×1 万元/公里=10。那么本周的：进度偏差（SV）= BCWP-BCWS=12-10=2；进度偏差率（SV%）=（BCWP-BCWS）/BCWS×100%=2/10×100%=20%；进度绩效指标（SPI）= BCWP/BCWS=1.2；成本偏差（CV）= BCWP – ACWP=12-15=-3；成本偏差率(CV%)= (CV / BCWP) × 100=-3/12×100%=-25%；成本性能指标（CPI）= BCWP / ACWP=12/15=0.8。

19.4.3 项目参数图表分析

本部分主要是根据项目性能指标度量数据自动生在相关图表，以方便项目组更直观的了解项目性能的相关指标。包含的图表主要有：项目工作量按阶段分布图、项目工作量按类别分布图、质量成本比例分布、工作量偏差趋势分析图、项目进展盈余分析图、项目进度成本趋势图、项目进度成本性能指标趋势图。

其中：在质量成本比例分布图中，正质量成本包含：项目评审、测试、培训，可以理解成为提高质量主动采取活动所花费的工作量；负质量成本是指返工所占比例，可以理解成为提高质量被动采取活动所花费的工作费。公式为：正质量成本=（评审工作量+测试工作量+培训工作量）/实际工作量；负质量成本=返工工作量/实际工作量。

19.4.4 产品评审度量

此处主要是对开发过程中主要工作产品评审时产生的数据进行度量，分为两类，一是评审时发现的缺陷；二是被评审工作产品的规模及评审用时。由这两类数据得到“总评审效率（个/人时）”、“重大评审效率（评审时发现严重缺陷的效率）”、“评审总缺陷密度（个/规模）”、“评审重大缺陷密度”等度量指标。同时生成了两个统计图表，分别为：评审缺陷按严重程度统计、项目技术评审效率统计。

对单个项目来说，这些数据最多也只能表明某个工作产品的评审效率。但对整个公司来说，把类多项目放在一起进行对比分析的时候，就可以得到若想评审好某类工作产品，达到什么样的评审效率才是符合公司要求的。这也正是 CMMI4 级中研发定量管理的基础。

19.4.5 产品测试度量

此处主要是收集三种数据，分别为：每阶段的测试按严重程度分类缺陷数据、每阶段的测试按优先级分类缺陷数据、每阶段测试花费工时。有时，一些公司可能还会按缺陷技术分类进行数据的收集，比如把缺陷分为界面缺陷、接口缺陷、算法缺陷、数据访问缺陷等。由

这些数据得到“总测试效率（个/人时）”、“重大测试效率（测试时发现严重缺陷和较严重缺陷的效率）”、“总测试缺陷密度（个/规模）”、“重大测试缺陷密度”等度量指标。

同时由这些数据生成了三个统计图表，分别为：测试缺陷按严重程度统计、测试缺陷按优先级统计、测试效率和测试缺陷密度统计。

19.4.6 过程质量度量

过程质量度量主要般包括：项目不符合项(NCR)统计表，用于统计每个项目阶段发现不符合项的数据，主要分为“本阶段新”、“正在解决”、“完成”、“提交高级经理（总工程师/研发部经理）”四种情况；项目阶段 NCR 汇总，等于本阶段新+正在解决+完成+提交高级经理。在这些数据基础之上，可以得到“项目阶段 NCR 密度”、“项目 NCR 密度”两个度量指标。计算方法分别为：项目阶段 NCR 密度=项目阶段 NCR 汇总/项目组阶段实际工作量；项目 NCR 密度=项目 NCR 汇总/项目实际总工作量。除此之外，还可以得到项目不符合项(NCR)阶段统计表，以图表的方式表示每个阶段 NCR 数量及趋。

19.4.7 项目需求度量

在进行需求度量时，主要是收集软件的需求总数以及每个阶段需求变更情况，这些数据均可以通过《用户需求跟踪矩阵》得到。在这些数据的基础之上，主要得到以下度量指标：

需求稳定度=未更改需求数/现有需求总数

需求变化率=需求变更总数/现有需求总数

未更改需求数=初始需求数-删除需求数-修改需求数

现有需求总数=初始需求数+增加需求数-删除需求数

需求变更总数=增加的需求数+删除的需求数+修改的需求数

19.4.8 其他度量

项目风险度量：根据《首要风险列表》对每个阶段的风险进行收集，主要是把每个阶段“已识别风险数”、“规避成功风险数”、“规避失败风险数”、“未识别风险数”、“异常风险数”详细填写在《项目度量数据库》。在这些数据之上，得到如下度量指标：

风险总数=已识别风险数+规避成功风险数+规避失败风险数+未识别风险数

风险识别率=(风险总数-未识别风险数)/风险总数

风险控制力=规避成功风险数/风险总数

团队规模度量：把每个阶段实际投入人数及所用的工期这两类数据收集填写进《项目度量数据库》，在此基础之上得到人员投入的平均值。形成“项目人员数量按阶段分布图”、“项目各阶段投入工时分布图”两个图表。

生产率度量：生产率= 软件规模/实际工作量

资产累积度量：文档累积率=实际提交的文档个数/计划提交的文档个数；数据累积率＝实际的数据个数/计划填写的数据项个数

缺陷注入率=项目总缺陷数/软件规模

第20章 软件开发过程管理

内容提要：

- **CMMI** 中对应实践
- 过程改进活动
- 过程资产维护
- 过程性能改进

软件开发过程过程管理（Organization Software Development Process Management，简称OPM）包括三个部分内容：

过程定义与维护：其目的是创建与维护机构可用的过程资产。并且通过收集各个项目的信息，使过程资产不断得到积累，持续改进 OSSP，使机构长期受益。

机构过程改进：其目的是在全面了解机构过程及过程资产存在的强项和弱项的基础上，计划和实施机构的过程改进工作。

过程性能管理主要是：根据机构商业目标和过程的特点，定义过程度量；然后，采集过程度量数据，采用统计学原理和方法，分析项目实际执行的过程性能(process performance)，识别和消除造成过程性能不好的可归属原因(assignable cause)，确保过程稳定或受控。所谓一个过程是受控的，系指根据过去和现在的度量数据，可以预测该过程的未来变化趋势，至少使度量数据因难以完全消除的普通原因(common cause)而造成的随机变化不会超过一定的门限值(Control Limits)。在过程稳定的前提下，采取措施，改进过程性能，提高过程能力，以满足机构商业目标或用户需求。通常，过程度量的均值和方差用于过程能力的量化表示。所谓提高过程能力，通常的做法是通过某种改进措施，达到移动(提高)均值和减小方差。刻划过程固有特性的多个过程能力的量化指标组成过程能力基线(Process Capability Baseline，简称PCB)。

20.1 CMMI 中对应实践

在 CMMI 中有三个过程域（PA）与本章内容相关，分别是：机构过程焦点（Organizational Process Focus，简称 OPF），机构过程定义（Organizational Process Definition，简称 OPD），机构过程性能（Organizational Process Performance，简称 OPP），共有 20 个过程域，在公司进行软件过程改进必须遵循的一些准则及有效实践。分别介绍如下：

OPF 的目的是在基于对当前机构过程及过程资产中强项、弱项的完成理解之上，计划、实现和推广机构过程改进。

SG 1 Determine Process Improvement Opportunities（确定过程改时机会），定期或者根据需要识别机构过程中的强项、弱项和改进机会。

SP 1.1 Establish Organizational Process Needs（建立机构过程需要），建立和维护机构过程需要及目标的描述，形成机构过程需要和目标文档。可以通过如下几步完成该实践：一是识别与机构过程相关的方针、标准和业务目标；二是调查相关过程标准和模型中的最佳实践；三是确定机构过程性能目标，可以使用量化或非量化的术语来表达，比如研发周期、缺陷移除率、生产效率等；四是定义机构过程的重点特征；五是文档化机构过程需要和目标；六是根据需要修订机构过程需要和目标。

SP 1.2 Appraise the Organization's Processes（评价机构过程），根据需要或定期对机构过程进行评价，以保持对其强项和弱项的理解。一般会形成如下工作产品：机构过程评估计划，针对机构过程强项和弱项评估发现记录，对机构过程的改进建议。可以通过如下几步完成该实践：一是从高级经理处获得对过程评估的支持；二是定义过程评估的的范围；三是确定过程评估的方法和准则；四是为过程评估制定计划、安排进度并进行准备；五是执行过程评估；六是文档化及发布评估活动及评估发现。

SP 1.3 Identify the Organization's Process Improvements（识别机构过程改进项），识别机构过程和过程资产的改进项。一般会形成如下工作产品：候选过程改进项的分析，机构过程改进标识记录。可以通过如下几步完成该实践：一是确定候选过程改进项；二是对候

选过程改进项进行排序；三是识别并文档化将要执行的过程改进项；四是修订计划的过程改进项，以保持时效性。

SG 2 Plan and Implement Process Improvements（计划并执行过程改进），针对机构过程及过程资产改进的过程行动得到计划并执行。

SP 2.1 Establish Process Action Plans（建立过程行动计划），针对机构过程及过程资产改进建立和维护过程行动计划（PAP），形成组成批准过的过程行动计划。可以通过如下几步完成该实践：一是针对已识别的过程改进项确定策略、方法和行动；二是建立执行行动的过程行动团队（组）；三是文档化过程行动计划；四是与相关干系人协商及审查过程行动计划；五是必要时对过程行动计划进行评审。

SP 2.2 Implement Process Action Plans（执行过程行动计划），一般会形成如下工作产品：过程行动组中各方面的承诺，过程行动计划执行中的状态和结果，试点计划。可以通过如下几步完成该实践：一是使过程行动计划对相关干系人就绪可用；二是商议并文档化过程行动组中的承诺，并且根据需要修订过程行动计划；三是根据过程行动计划跟踪进展和承诺；四是过程行动组和相关干系人执行交叉评审，以监控过程行动的时展和结果；五是计划试点以测试选择的过程改进；六是审查过程行动组的活动及工作产品；七是识别并文档化在执行过程行动计划时出现的问题，并跟踪至关闭；八是确保执行过程行动计划的结果满足机构过程改进目标。

SG 3 Deploy Organizational Process Assets and Incorporate Lessons Learned(推广机构过程资产并将过程相关经验教训纳入该资产中)，

SP 3.1 Deploy Organizational Process Assets（推广机构过程资产），一般会形成如下工作产品：机构过程资产推广计划以及整个机构中过程资产的变更，推广及其变更的培训资料，机构过程资产变更文档，机构过程资产推广及其变更的支持资料。可以通过如下几步完成该实践：一是在整个机构推广机构过程资；二是文档化机构过程资的变更；三是在机构中推广机构资产中的变更之处；四是对使用机构过程资产提供指导和咨询。

SP 3.2 Deploy Standard Processes（推广标准过程），在项目开始阶段推广机构的标准过程

集，并且在每个项目生命周期的适当时机推广他们的变更。一般会形成如下工作产品：机构项目列表以及在每个项目中过程推广的状态（比如：现有项目及计划的项目），在新项目中推广机构标准过程集的指导手册，在确定的项目中执行机构标准过程集的裁剪记录。可以通过如下几步完成该实践：一是在机构确定处于起始阶段的项目；二是确定将从执行机构当前标准过程集受益的活动项目；三是编制计划，以便在已确定的项目中执行机构当前标准过程集；四是协助项目通过裁剪机构标准过程集以满足项目需要；五是对于已确定项目保持裁剪及执行过程的记录；六是确保从过程裁剪得到的定义过程纳入过程遵从性审计计划；七是当前机构标准过程集更新时，识别项目应当执行哪些变更。

SP 3.3 Monitor Implementation（监控执行情况），监控在所有项目上机构标准过程集的执行及过程资产的使用。一般会形成如下工作产品：项目中监控过程执行记录，过程遵从性评估记录和状态，对选用过程中创造典型结果（做为过程裁剪和执行一部分）的评审记录。可以通过如下几步来完成该实践：一是监控项目中机构过程资产及其变更的使用情况；二是评审在每个项目生命周期中选用过程创造的典型结果；三是评审过程遵从性评估结果，以确定机构标准过程的推广情况；四是识别并文档化与执行标准机构过程相关的问题，并跟踪直至关闭。

SP 3.4 Incorporate Process-Related Experiences into the Organizational Process Assets（把过程相关的经验纳入机构过程资产），把从计划和执行过程中得到的与过程相关的工作产品、度量标指、改进信息纳入到机构过程资产。一般会形成如下工作产品：过程改进建议，过程中的经验教训，机构过程资产的度量，机构过程资产的改进建议，机构过程改进活动记录，机构过程资产及其改进相关信息。可以通过如下几步完成该实践：一是结合机构商业目标，对机构标准过程集及相关机构过程资产的有效性和适宜性进行定期评审；二是获取使用机构过程资产的反馈；三是得到从定义、试点、执行和推广机构过程资产产生的经验教训；四是得到对机构中人员合适有用的经验教训；五是分析机构通用度量集；六是评估在机构中使用的过程、方法和工具，得到改进过程资产的建议；七是得到对机构中人员最适用的过程、方法和工具；八是管理过程改进建议；九是建立并维护机

构过程活动改进记录。

OPD 的目的是创建与维护机构可用的过程资产及工作环境标准。并且通过收集各个项目的信息，使过程资产不断得到积累，持续改进 OSSP，使机构长期受益。

SG1 Establish Organizational Process Assets（建立机构过程资产），建立和维护一个机构过程资产集。

SP 1.1 Establish Standard Processes（建立标准过程），建立和维护机构的标准过程集。在一个企业里标准过程可以在多个层面上进行定义，他们可能类似于一个分层的方式。机构的标准过程集通常包括技术、管理、行政、支持和机构级的过程，应当从机构或项目需要的所有过程中进行收集。

SP 1.2 Establish Lifecycle Model Descriptions（建立生命周期模型描述），建立和维护经核准在机构内使用的生命周期模型描述。常见的生命周期有：瀑布模型、螺旋模型、进化模型、增量模型、迭代模型，各个公司可以根据自身研发的特性制订出适合自己的软件开发生命周期模型，也可以采用业界现有的生命周期模型。

SP 1.3 Establish Tailoring Criteria and Guidelines（建立裁剪准则及指南），建立和维护针对机构标准过程集的裁剪准则和指南。裁剪指南主要是解决如下问题：怎么样使用机构标准过程集及机构过程资产产生项目的定义过程；定义过程必须满足强制的需求（比如：对任何定义过程必须是机构过程资产的子集）；必须按照文档化的过程进行裁剪。

SP 1.4 Establish the Organization's Measurement Repository（建立机构度量数据库），一般会形成如下工作产品：针对机构标准过程集的产品和过程通用度量指标集的定义，机构度量数据库的设计说明，机构度量数据库（包括，库结构和支持环境），机构度量数据。

SP 1.5 Establish the Organization's Process Asset Library（建立机构过程资产库），以下均可以放进机构过程资库（不局限于这些内容）：机构方针、定义过程描述、规程（比如：估算规程）、开发计划、获取计划、质量保证计划、培训资料、过程帮助资料（比如：检查列表）、经验教训报告等。

SP 1.6 Establish Work Environment Standards（建立工作环境标准），工作环境标准使用项

目组和机构从通用工具、培训和维护中受益，降低成本。

OPP 的目的是建立和维护对机构标准过程集性能的量化理解，以支持过程性能目标及质量目标，并且为定量的管理机构的项目提供过程性能数据、基线和模型。由此可以，此过程域的目标能否顺利有效执行，是建立在公司的度量数据库中是否有足够的样本数据或候选项目数据基础之上的。

SG 1 Establish Performance Baselines and Models（建立性能基线和模型），建立和维护表明机构标准过程集期望的过程性能的基线和模型。

过程性能是过程所达到的实际结果的度量。过程性能由过程度量和产品度量二种度量表示。机构的共同度量由过程度量和产品度量组成，被用于总结机构中单个项目的过程实际性能。分析这些度量数据以建立结果的分布和范围，这些分布和范围表示了单个项目过程的期望性能。期望的过程性能可以被用于建立项目的质量目标和过程性能目标，并可被用作基线与实际项目性能进行比较。基线被用于项目定量管理。每个进行定量管理的项目提供的真实性能结果可以成为机构过程性能资产基线数据的一个组成部分。

SP 1.1 Select Processes（选择过程），从机构标准过程集中选择包含进机构过程性能分析的过程或子过程。通过常情下，对机构标准过程集中的所有过程或子过程应用统计管理技术是不可能的、无用的或不经济的。可以基于机构和项目需要及目标来选择过程或/和子过程。

SP 1.2 Establish Process-Performance Measures（建立过程性能度量），建立和维护包含在机构过程性能分析中的度量指标定义。可以通过如下几步完成该实践：一是确定哪些应进行度量的质量和过程性能方面的机构商业目标；二是选择能反映机构质量和过程性能的度量；三是将所选度量纳入机构共同度量集；四是必要时修订度量集。

SP 1.3 Establish Quality and Process-Performance Objectives（建立质量和过程性能目标），为机构的质量和过程性能建立和维护定量目标。可以通过如下几步完成该实践：一是评审与质量和过程性能相关的机构商业目标；二是定义质量和过程性能的机构定量目标；三是定义质量和过程性能机构目标的优先级；四是评审、协商质量和过程性能机构目标

及其优先级，并获取来自于项目相关各方的承诺；五是必要时，修订质量和过程性能机构定量目标。

SP 1.4 Establish Process-Performance Baselines（建立过程性能基线），建立和维护机构过程性能基线。可以通过如下几步完成该实践：一是从机构的项目收集度量数据；二是利用所收集的度量数据和分析结果，建立和维护机构过程性能基线；三是评审机构过程性能基线，并获取项目相关各方的同意；四是使存放在机构度量数据库中的机构过程性能信息可为全机构使用；五是把机构过程性能基线与相关目标进行比较；六是必要时，修订机构过程性能基线。

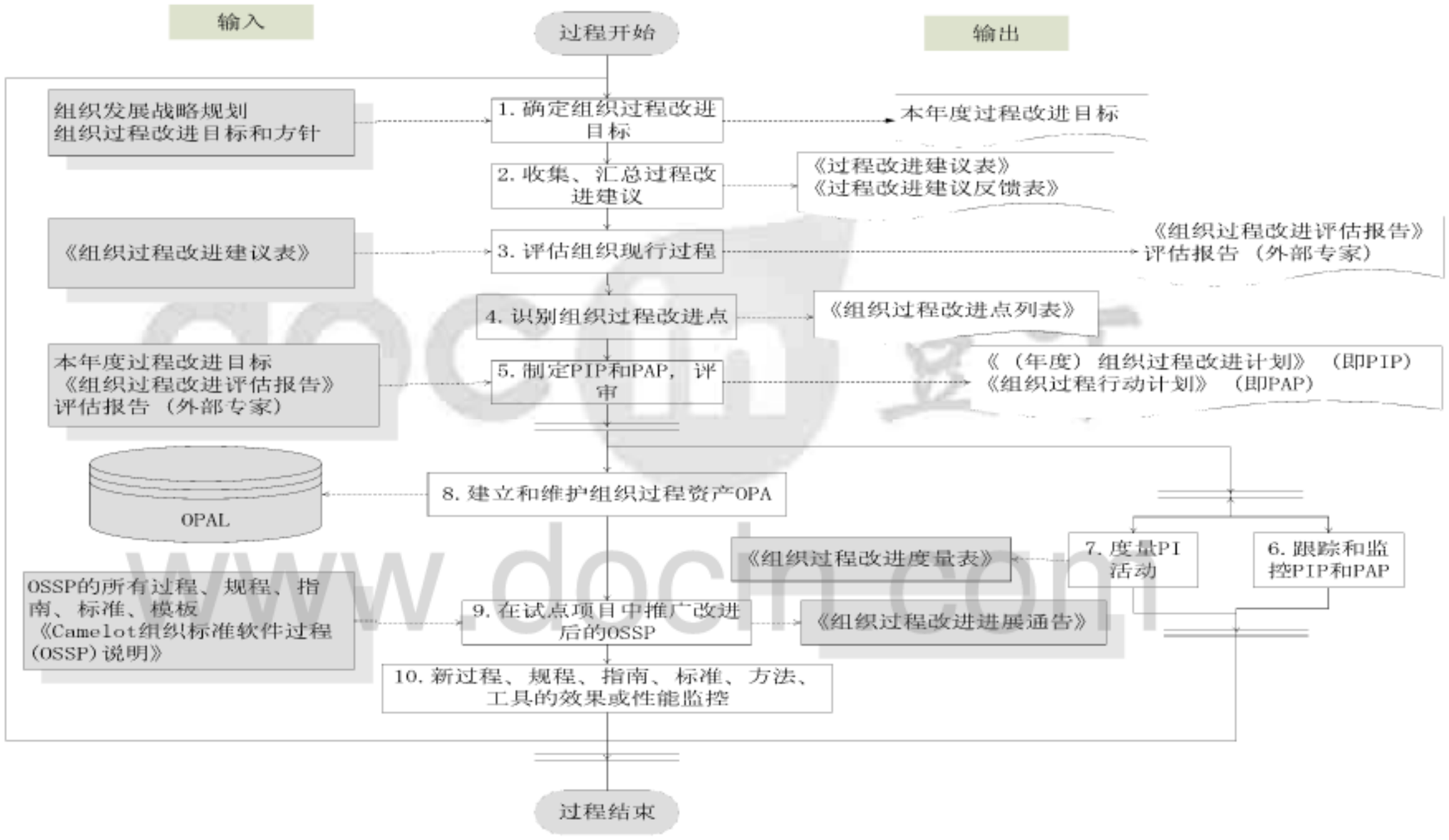
SP 1.5 Establish Process-Performance Models（建立过程性能模型），建立和维护机构标准过程集的过程性能模型。可以通过如下几步完成该实践：一是基于机构标准过程集和机构过程性能基线，建立过程性能模型；二是基于机构过去的结果和未来的需要校正过程性能模型；三是评审过程性能模型，并获取项目相关各方的同意；四是支持项目组使用过程性能模型；五是必要时修订过程性能模型。

20.2 过程改进活动

过程改进（Process Improvement，简称 PI）的目的是：建立和维护对机构级和项目级的过程，并就该过程在机构内达成改进共识；协调机构过程的实施，并对过程进行评估、维护以及改进的各项工作；全面识别并了解机构过程及过程资产已经存在或潜在的强项及弱项。

在公司推行过程改进时，一般是按如下流程图来执行的（参见表 20-1），其中每个步骤活动的执行方法及各类人员的职责解释如下。

图表 20-1 研发过程改进活动流程图



一、确定机构过程改进目标

EPG 组长在每年年底制定来年机构过程改进的目标，应当建立在公司的商业目标基础之上。EPG 组长通过与机构领导、总工程师交流，获得和充分理解机构的发展战略规划，并结合机构过程改进目标和方针，制定出合理可行的本年度机构过程改进目标，该年度目标写在《（年度）机构过程改进计划》（Process Improvement Plan，简称 PIP）的“年度目标”中。

二、收集、汇总过程改进建议

公司制定统一的《过程改进建议表》，所有部门及人员均可通过该表提交对公司当前研发过程的改进建议。然后由 EPG 向机构内所有相关人员收集过程改进的建议，用于制定下一轮改进的目标。

项目经理在每个项目结项后，收集项目组成员的过程改进意见，汇总在《过程改进建议表》中提交 EPG；另外，EPG 还可以不定期的召开 OSSP（机构标准软件过程）使用讨论会，请相关项目经理、项目组成员、质量保证工程师、CM 充分参与讨论，在会上给出改进意见，也可以采取发放纸质调查问卷或网上调查的方式，收集得到的过程改进建议汇总在《过程改进建议表》中。EPG 在每次收到《过程改进建议表》后，进行分析汇总，作为过程改进的参考。

三、评估机构现行过程

EPG 每年年底或在过程改进计划执行中的里程碑点，针对上一年或上一阶段公司标准软件过程的应用情况进行评估，通过的评估有以下三个级别：

- A 类验证，质量保证工程师对项目经理、与某些数据相关的项目组成员进行面对面的直接访谈，准备一些可以验证数据正确性、完整性和一致性的问题向被访谈人员提问，根据被访谈人员的回答检查相应数据，做出结论
- B 类验证，质量保证工程师除了对相关人员进行访谈外，还需检查数据来源的工作产品，如项目周报、个人周报、风险跟踪表、测试报告等等，核对这些数据与项目度量数据库中大的数据是否一致；
- C 类验证，质量保证工程师根据正式评审的要求，对相关人员、相关工作产品进行

正式的评审

评估时，EPG 和相关人员针对《过程改进建议表》中的建议逐项识别与讨论，并确定过程的强项、弱项及待改进项，由此生成《机构过程改进评估报告》作为本年度或本阶段机构过程改进的依据，并在下一年度或下一阶段对此机构过程改进的效果进行验证。如果机构条件允许，可以每隔一年或一年半请外部专家对机构的现行过程进行一次评估。在此外部专家的《评估报告》就是机构进行过程改进时一个比较重要的依据。

四、识别机构过程改进点

EPG 组长及相关成员根据过程改进的建议和对机构内现行的过程评估情况，找出机构执行 OSSP 时的强项和弱项，这些强项和弱项应该是各项目间的共性，识别出的改进点。

五、制定 **PIP** 和 **PAP**

EPG 组长根据本年度机构过程改进目标、《机构过程改进评估报告》和（或）外部专家的《评估报告》，制定出《（年度）机构过程改进计划》；PIP 的内容包括过程改进相关人员及职责，实施的步骤，进度表（可使用 project 等软件制作，作为该计划的附件），跟踪控制的计划（评审计划、里程碑计划等）。

在年度过程改进计划的基础之上，为了确定具体项目组实施过程改进的实践活动的计划，由 EPG 制订《过程行动计划》（Process Action Plan，简称 PAP）。在 PAP 中明确试点项目，人员和职责、权限，新过程、新工具、新方法的推广及验证，行动风险等；PIP 和 PAP 须经过评审和总工程师审批后，方可执行。在实际执行过程中，在每次对 OSSP 改进之后，为了对改进后的过程进行试点，均需要制定 PAP。然后对 OSSP 在中试点项目应用情况来确定在整个机构内怎么推广新版的 OSSP。

六、跟踪和监控 **PIP** 和 **PAP**

EPG 组长作为过程改进项目（即 PI 项目）的项目经理，需要对 PIP 计划和 PAP 计划的执行情况进行跟踪和监督，对出现的偏差或风险要及时采取纠正措施。跟踪和监控时可采用的手段：

- EPG 组长定期召开例会，总结本阶段工作内容做出下阶段工作计划，会后生成《EPG

会议记录》，EPG 成员定期提交个人工作报报，EPG 组长收集后汇总生成《EPG 工作报告》。另外，根据公司过程改进的实际进展情况，EPG 也可以召开的不定期的会议，在会议上解决在过程改进中出现的问题，并形成《EPG 会议记录》；

- 在 PIP 的里程碑处 EPG 组长召集总工程师、PAP 中的相关项目经理及相关质量保证工程师、CM、主要项目成员等，对前一阶段的过程改进活动进行评审，生成《机构过程改进里程碑报告》；
- 进行过程改进活动时，EPG 需要对已改进的方面进行详细记录，定期提交机构领导、总工程师及相关人员审阅；
- 当因偏差或风险导致 PIP 发生变更时，需参照项目规划过程中关于项目计划变更控制的部分来进行。需要注意的是，变更后的 PIP，需经过管理评审和机构领导审批认可后，方可执行。

七、度量过程改进活动

EPG 组长带领 EPG 成员执行 PIP 的过程中，定期收集过程改进相关活动所花费的工作量、质量、成本、规模、进度偏差、变更次数等。度量活动可以每季度或事件驱动的进行，并把收集的数据结果提交给公司领导、总工程师及相关人员审阅。

八、建立和维护机构过程资产库（**OPAL**）

具体操作方法请参见下一节“过程资产定义与维护”

九、在试点项目中推广改进后的 **OSSP**

将改进后的 OSSP、方法、工具等，根据 PAP 中确定的试点项目，进行验证，确认有效后在机构的其他所有项目中实施。对实施过程中的进展情况向机构全体成员汇报。

十、新过程、规程、指南、标准、方法、工具的效果或性能监控

对机构决定使用的或 EPG 推荐试点项目使用的新过程、规程、指南、标准、方法、工具，EPG 要对其使用效果及效率进行监控和定量的评价，以此来评判上述内容是否可以使用或推广。如果可以在机构范围内推广使用，则将其完善到机构标准软件过程中，并在合适时，推广到机构的其他项目中。对于 EPG 阶段性的工作进展，EPG 组长定期向机构内全体人员公告。

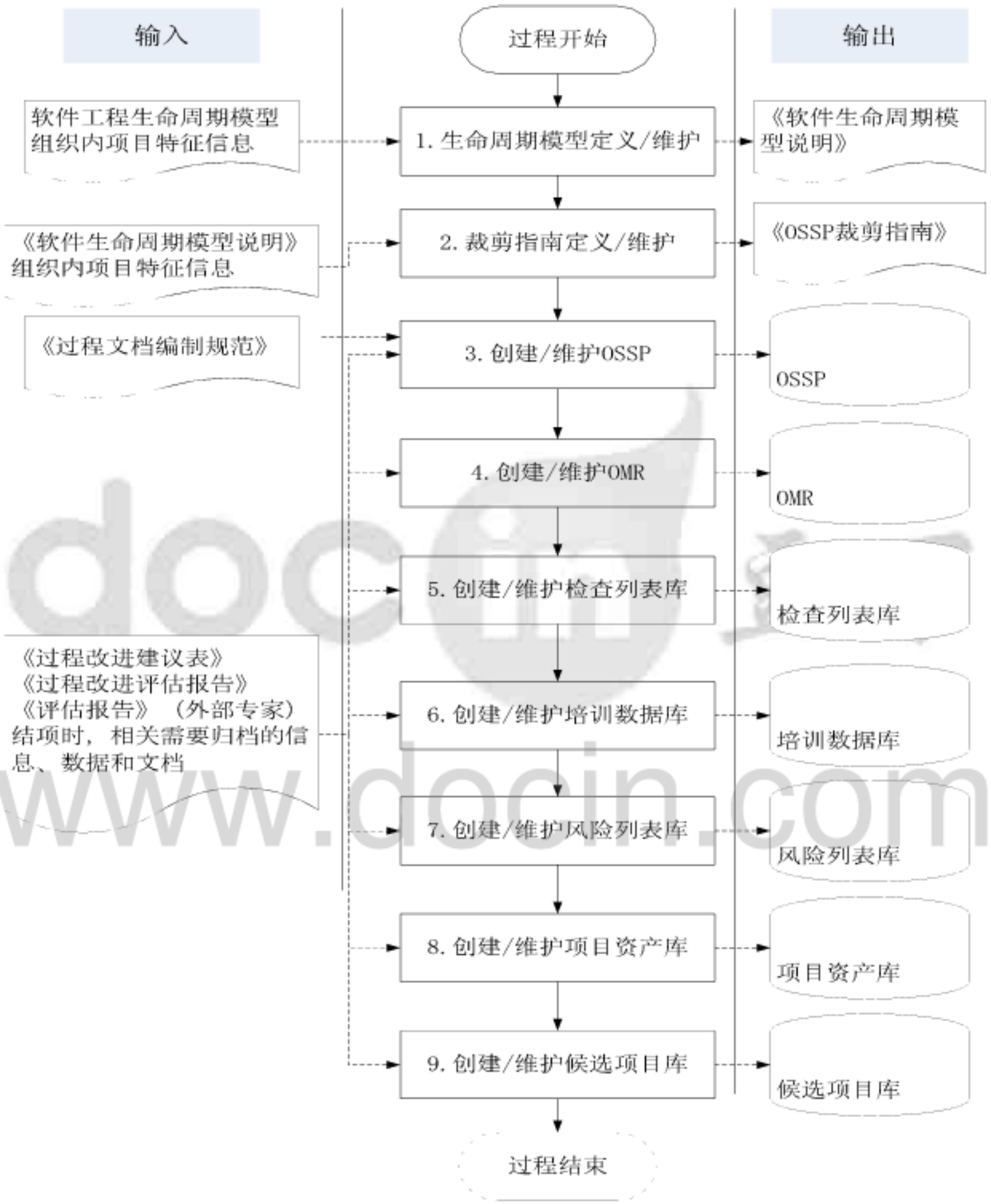
20.3 过程资产定义与维护

过程资产定义与维护的目的是：通过对机构过程资产的创建与维护，使得从项目中收集和提取上来的数据、信息和经验得以保存和积累，并经过一定的整理、汇总和加工后可以有效的应用到机构的其它项目或过程中。对机构过程资产库（Organization Process Asset Library，简称 OPAL)的创建和维护能够使机构的过程资产得到固化，使机构从中长期受益。本节相关的活动分为两个阶段：一是过程资产开发（定义）阶段，二是过程资产维护阶段。在对 OPAL 管理时，我们建议：

- OPAL 可以放进专业的配置管理工具中进行版本管理及变更管理。
- EPG 周期性或事件驱动地对 OPAL 相关实体的合理和易用性进行评估与改进工作，并对项目组成员及相关小组和个人提交的过程改进建议进行评审，改进或修正 OPAL 结构及其实体内容。
- OPAL 为 EPG 所拥有，且只有 EPG 有写数据库的权利；项目经理及其级别以上的人员都只有阅读的权利。每个项目结项时，项目经理需要向 EPG 提交《项目结项报告》及相关过程管理报告。在数据被输入 OPAL 之前，EPG 组长需要与项目经理、项目的质量保证工程师、配置管理员共同验证数据的正确性及有效性。
- 只有经过 EPG 组长批准后的数据，才能为相关人员可见，才可以用于管理和决策之用。为保证数据的安全性，EPG 需要定期对 OPAL 进行备份。

以下是执行该过程的流程图（见下页图 20-2），在其中共分为九大类活动，在实际操作过程中，这些活动可以并行执行。

图表 20-2 研发过程资产定义与维护流程图



一、生命周期模型定义/维护

创建过程：EPG 收集、整理机构内现行和可能开展的项目的特征信息（包括：立项类型、

项目类型、项目周期、项目阶段信息），参照软件工程学中常用的生命周期模型，定义符合本机构的生命周期模型（可能是多类模开，以适合公司内不同类型的项目），并将定义好的软件生命周期模型在《软件生命周期模型说明》进行详尽的描述。描述的内容包括：各种软件生命周期模型及各自的适用范围、阶段描述、生命周期模型的图形说明及注意事项等。

维护过程：当有机构《软件生命周期模型说明》中未提及的、新的项目类型产生时，EPG 为此项目类型创建新的生命周期模型并将其更新到《软件生命周期模型说明》中。当现有项目生命周期模型不符合机构大部分项目使用或不能够很好地为项目服务时，EPG 需要征求项目经理及技术骨干的意见，对《软件生命周期模型说明》进行修订。

公司的《软件生命周期模型说明》创建或更新后，需由高层经理、EPG、相关业务部门经理、项目经理等组成评审小组对《软件生命周期模型说明》进行评审。

二、创建/维护 **OSSP**

创建过程：当创建一个新的过程时，EPG 来负责定义新的过程。EPG 收集机构现行的项目管理和软件工程方面的过程、方法和使用的工具，进行综合汇总、分析，定义出来一套符合机构给定的软件开发策略、软件过程标准和产品标准的通用的软件过程。在该 OSSP 中描述了经规范后的代表机构当前水平的软件工程过程和方法以及软件工程科目（如，软件需求分析、软件设计、程序编码、软件测试等等）之间的内部过程接口和软件工程过程和其他过程之间的外部过程接口（此处的接口即每个过程的输入和输出）。最终，EPG 将文档化的 OSSP 实体（过程文档、规程、指南、标准、模板）和《机构标准软件过程(OSSP)说明》文档作为本活动的结果和输出，供机构中所有项目使用。

维护过程：EPG 周期性的或事件驱动的对 OSSP 进行评估与改进工作，并根据《过程改进建议表》、《过程改进评估报告》和（或）外部专家的《评估报告》对 OSSP 进行修订。该活动被触发有如下几个时机：

- 组织提出且需要纳入到 OSSP 中新的过程、规程、标准、模板、方法或工具时；
- 组织成员和（或）相关负责人提出的修改建议；
- 在项目结项时，EPG 听取项目经理、项目组成员对由 OSSP 裁剪生成的 DSP 的评价，

识别出来问题时。

评审：OSSP 制定完成或版本更新后，由 EPG、相关业务部门经理、项目经理等组成的评审小组对 OSSP 进行评审。

创建/维护 OSSP 注意事项：

- 建立、修订各个过程的标准流程，可采用流程图加文字描述的方式。如果过程中某个活动涉及的步骤较多，较复杂，则使用单独的规程来描述这个活动，使之更加清晰具体；进行某项活动时可供参考或选择的做法称作指南；标准是进行某一项活动时需要遵循的准则。合理的规程、指南和标准能使 OSSP 的指导性和可操作性增强；
- 建立、修订各过程的文档模板，使 OSSP 转化为文档实体；
- 将 OSSP 所涉及到的所有过程及过程之间的关系，即 OSSP 过程体系框架说明在《机构标准软件过程(OSSP)说明》文档进行详细的描述；
- 每次新增和修订 OSSP 的结构、内容后，需要进行详细记录。

三、裁剪指南定义/维护

机构标准软件过程是在一般意义上所进行的描述，因各实际项目的特殊性通常不直接使用完整的 OSSP。因此，EPG 需要根据项目的具体特征信息，结合软件生命周期模型来制定 OSSP 的裁剪规则，用以指导特定项目从定义好的软件生命周期模型的集合中选择合适的生命周期，并对 OSSP 进行剪裁和精化，使其适用于项目的具体特征。

创建过程：EPG 根据《软件生命周期模型说明》和机构内项目的特征信息（如项目周期、项目来源、项目规模、管理特征、技术特征等，以及影响过程裁剪的其他因素），制定项目执行的过程的裁剪原则，针对每一种特征的项目建议采取的裁剪原则和裁剪方法，将其详述在《裁剪指南》中。

维护过程：当识别出《裁剪指南》中未提及的项目特征，或识别出新的影响过程裁剪的因素，并且原有的裁剪原则和方法、裁剪影响因素不再适用于具有这种特征的项目，则需要将新的影响因素、原则和方法更新到《裁剪指南》中。同样，创建和更新后的裁剪指南也需要进行评审，然后方可纳入机构过程资产进行使用。

四、创建/维护机构度量数据库（**Organization's Measurement Repository**，简称 **OMR**）

创建过程：创建 OMR 时，EPG 根据 OSSP 中各个过程的特征，提炼出机构关心的，能体现质量、效率、工作量、成本等的有效数据，放在《机构度量数据库》中。

维护过程：维护 OMR 时，分为两种维护：对库结构即度量项的维护和新增已结项项目的度量数据。对于度量项的维护，一方面，EPG 在评估现有过程后，根据形成的《过程改进评估报告》对机构度量数据库的库结构（度量项）进行增删改中的进行删减、扩充或修改；另一方面，当项目结项后，项目经理把经过验证的项目度量数据库提交给 EPG，EPG 组长提取项目度量数据库中新识别的度量项，经 EPG 评审后更新到《机构度量数据库》。对于新增已结项项目的数据，在每次项目结项时，EPG 汇总分析项目经理/度量分析员提供的相关工作报告，如《项目度量数据库》等，将该项目的度量项数据更新到《机构度量数据库》中。

技术评审：机构度量数据库创建或更新后，需由总工程师、EPG、相关业务部门经理、项目经理等组成评审小组对 OMR 的结构及度量项和新增数据进行评审。

五、创建/维护检查列表库

创建过程：创建检查列表库时，EPG 根据 OSSP 中各个过程的特征和各个工作产品的特征，分别制定出针对每个过程和产品的检查项，编写《机构过程审计检查表》和《机构产品审计检查表》。检查项是与本过程或产品的正确性、完整性、有效性、及时性等相关的关注点，将这些关注点用是非问句的方式列出，答案只能为“是”、“否”、“不适用”。《机构过程审计检查表》中列出针对每个过程的检查项，《机构产品审计检查表》中列出针对每个工作产品的检查项。《机构过程审计检查表》和《机构产品审计检查表》是质量保证工程师对过程和产品进行检查的依据。

维护过程：维护检查列表库时，分为两种维护：对库结构即检查项的维护和新增已结项项目的数据。对于检查项的维护，一方面，EPG 在评估现有过程后，根据形成的《过程改进评估报告》对检查列表库中的检查项进行删减、扩充或修改；另一方面，在每次项目结项时，EPG 组长将项目质量保证工程师提交的《QA 总结报告》中的“新增问题列表”进行分析，根据以下原则考虑是否将新的问题纳入到检查列表库中：

- 原则 1：是否对提高产品质量有帮助；
- 原则 2：是否能够帮助有效控制过程；
- 原则 3：是否是共性问题；

对于新增已结项项目的数据，在每次项目结项时，EPG 汇总分析质量保证工程师提供的相关工作报告，将该项目的检查项数据分别更新到《机构过程审计检查表》和《机构产品审计检查表》中。

在机构过程审计检查表及机构产品审计检查表创建或更新后，需由总工程师、EPG、相关业务部门经理、项目经理等组成评审小组对检查列表库结构及检查项和新增数据进行评审。

六、创建/维护培训数据库

培训管理员或公司的人力资源部门负责创建和维护培训数据库。培训数据库的结构包括：培训计划部分、培训教材部分、培训记录部分、培训总结报告部分。

创建过程：创建培训数据库时，培训管理员创建存放培训计划、培训教材、培训记录、培训总结报告等文件的空间，可按照培训的类别（如技术培训、管理培训）、等级（机构级培训、项目级培训）、年度分别创建。

维护过程：维护培训数据库时，培训管理员定期（每年/每季度）将年度、季度培训计划文档放置在培训计划文件夹中，培训结束后将本次培训时使用的电子教材、PPT 等放置在培训教材文件夹中，定期（每年/每季度）或培训结束后将项目培训申请表、培训需求表、培训需求调查报告、培训申请汇总表、免修记录、培训签到表、培训通知、培训资源表、职位技能对应表放置在培训记录文件夹中，培训结束后将将培训总结报告、培训评估报告放置在培训总结报告文件夹中。在每次培训结束后，培训管理员汇总本次培训的各项数据，将这些数据更新到《机构培训记录表》中。同样，创建和更新后的培训数据库也需要进行评审，然后方可纳入机构过程资产进行使用。

七、创建/维护风险管理库

公司的 EPG 负责创建和维护风险管理库。风险管理库包括：商业风险部分、管理风险部分、技术风险部分。

创建过程：创建风险管理库时，EPG 先确定风险的分类，如商业风险、管理风险、技术风险等，识别出各类风险的关注点，将这些关注点用是非问句的方式列出。同时，制定风险的等级，如严重性等级、可能性等级、优先级等作为风险的参数，写在《机构风险检查表》中。

维护过程：维护检查管理库时，分为两种维护：对库结构即风险检查项的维护和新增已结项项目的数据。对于风险检查项的维护，一方面，EPG 在评估现有过程后，根据形成的《过程改进评估报告》对风险管理库中的检查项进行删减、扩充或修改；另一方面，在每次项目结项时，将本项目中新发现的有助于避免和降低风险、有助于决策的风险检查项，经评审后补充到风险管理库中。对于新增已结项项目的数据，在每次项目结项时，EPG 汇总分析项目经理提供的相关工作报告，如《风险管理报告》，将该项目的风险检查项数据更新到《机构风险检查表》中。同样，创建和更新后的风险管理库也需要进行评审，然后方可纳入机构过程资产进行使用。

八、创建/维护项目资产库

创建和维护项目资产库的目的在于：保存对现在和将来的项目可能有用的过程文档，特别是与 OSSP 有关的过程文档；在机构范围内共享使用。

创建过程：创建项目资产库时，按照项目资产的分类，如：项目计划及其下属计划，项目定义的标准、检查表、指南、模板和设计规范，已经开发的工具和相关注释，培训材料，项目核心的技术文档（需求文档、设计文档、源代码、测试用例等），能被以后项目重用的其他文档等，为每类工作产品建立空间。也可以按照项目建立空间。

维护过程：维护项目资产库时，分为两种维护：对库结构的维护和新增已结项项目的数据。对于库结构的维护，一方面，EPG 在评估现有过程后，根据形成的《过程改进评估报告》对项目资产库的结构进行删减、扩充或修改；另一方面，在每次项目结项时，也可以根据该项目的新增特征对库结构进行调整。对于新增已结项项目的数据，在每次项目结项时，将该项目中可以借鉴的所有工作产品，分门别类的放置到项目资产库中，为以后的项目提供参考借鉴。对项目资产库内容及库结构均需要进行评审。

九、创建/维护候选项目库

创建和维护候选项目库的目的：保存那些未通过立项决策和因其他原因停止的项目的资料，特别是对 OSSP 的改进有用的资料；在机构范围内共享使用。由 EPG 来负责，具体方法如以上 1-8 所述。

20.4 过程性能管理[22]

一个公司的过程性能如何，不只是体现在过程执行能力和过程符合性上，更重要的是体现在过程与性能自我改进能力上。对于软件类的过程性能管理及改进主要是由 EPG 来负责开展，具体流程如图表 20-3（见下页）所示。对于整个公司的过程性能管理及改进，可能会有专门的机构或部门来负责，比如：质量管理部、规范化推进委员会等。

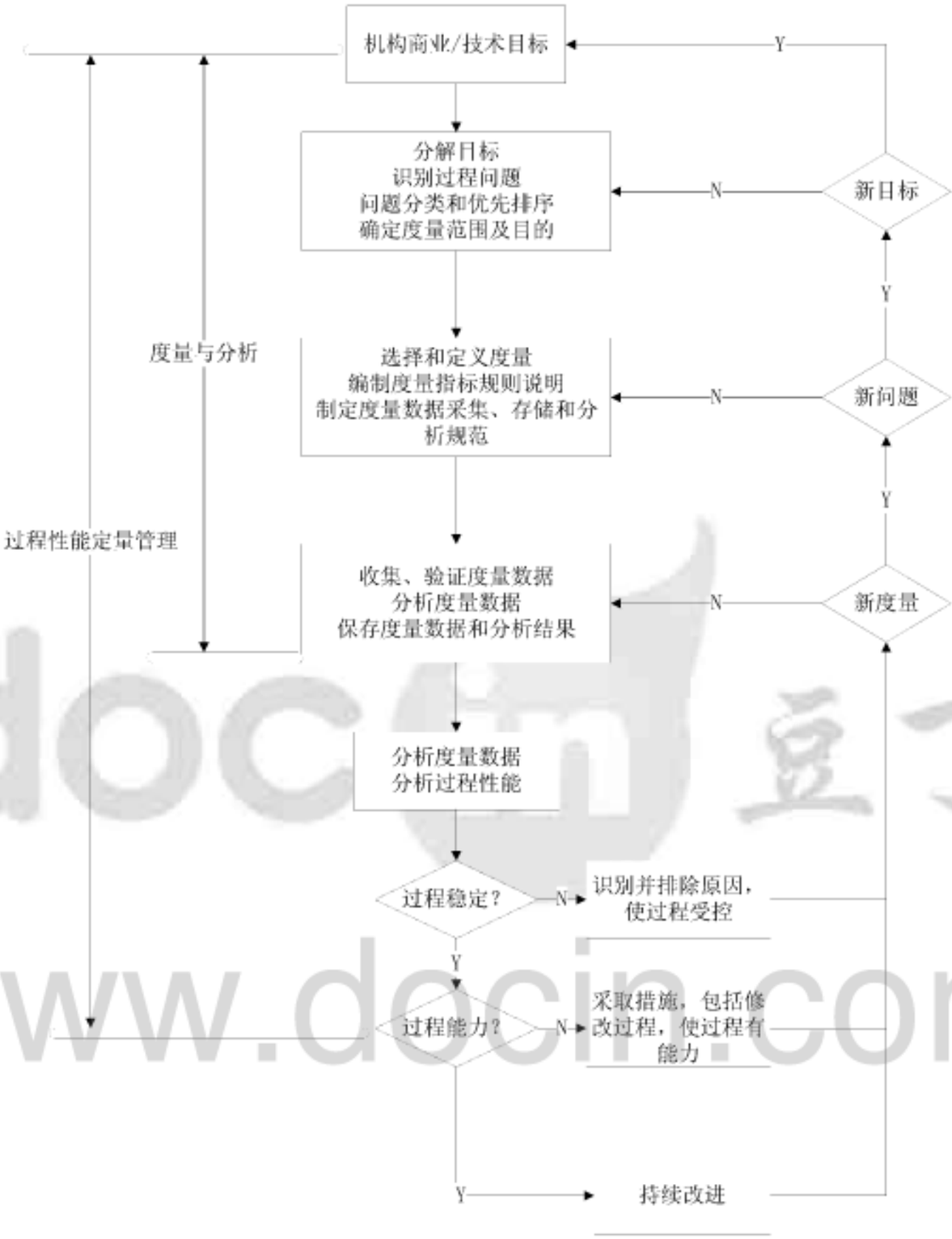
在进行过程性能管理及对公司已有数据进行定量分析时，均需要了解一些统计学的知识，现介绍如下：

一、用以分析过程行为的统计学方法很多，常用的工具有：

- 散点图(Scatter Chart)，一般用于分析二个变量之间的因果关系，比如规模和工作量的关系。
- 趋势图(Run Chart)，用于观察某一变量的观察值随时间(或观察点)的变化趋势。
- 因果图(Cause and effect diagrams)，用于研究一个问题与多种原因之间的关系，又称鱼刺图，也称石川(Ishikawa)图。
- 直方图(histogram)，用以观察变量值的频率分布。
- Pareto 图，用以观察和识别影响过程性能的大量无关紧要的因素(trivial many)和少数至关重要的因素(vital few)，以便针对后者，有针对性地采取改进措施。

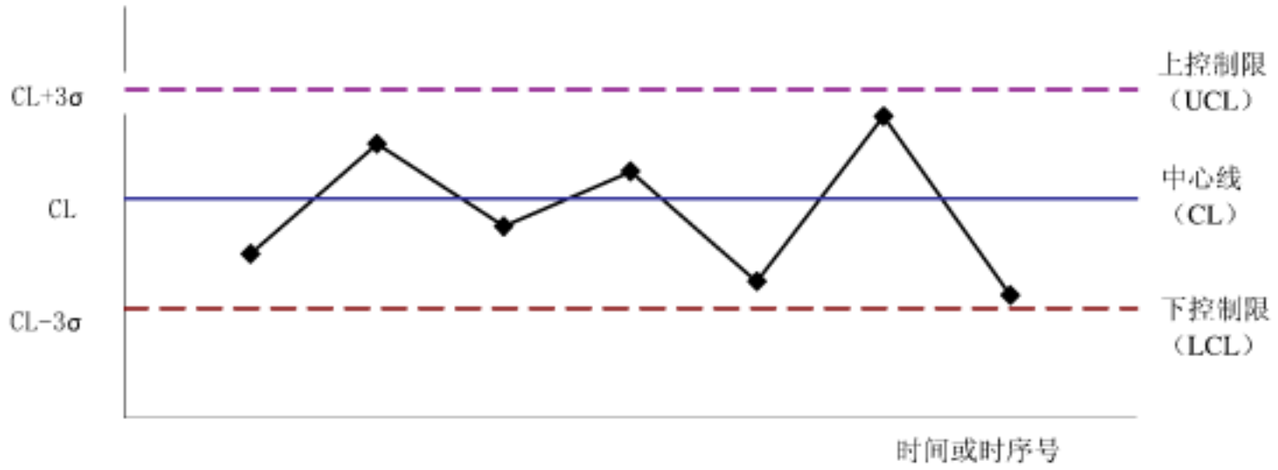
图表 20-3 研发过程性能改进流程图

[22] 本节内容引自于胡希明老师的培训讲义。



二、控制图，也称过程行为图（**process behavior charts**），一般由一条中心线、在中心线两侧的控制限以及相邻观测点的联线组成，如图 **20-4** 所示：

图表 20-4 控制图示例



图中，σ 为标准偏差，或均方差；中心线和控制限由采样值按规定公式计算所得，不能人为设置。

三、XmR 图

有多种形式的控制图，适合于变量数据或属性数据，并分别适用于不同的随机分布模型，例如：X 图和 R 图、X 图和 S 图、XmR 图、np 图和 p 图、c 图和 u 图，等等。其中适用性比较广的是单点值和移动值域图，简称 XmR 图，原文是 individuals and moving range charts，比较确切的译法是：单点值和（二点）波动幅度图。

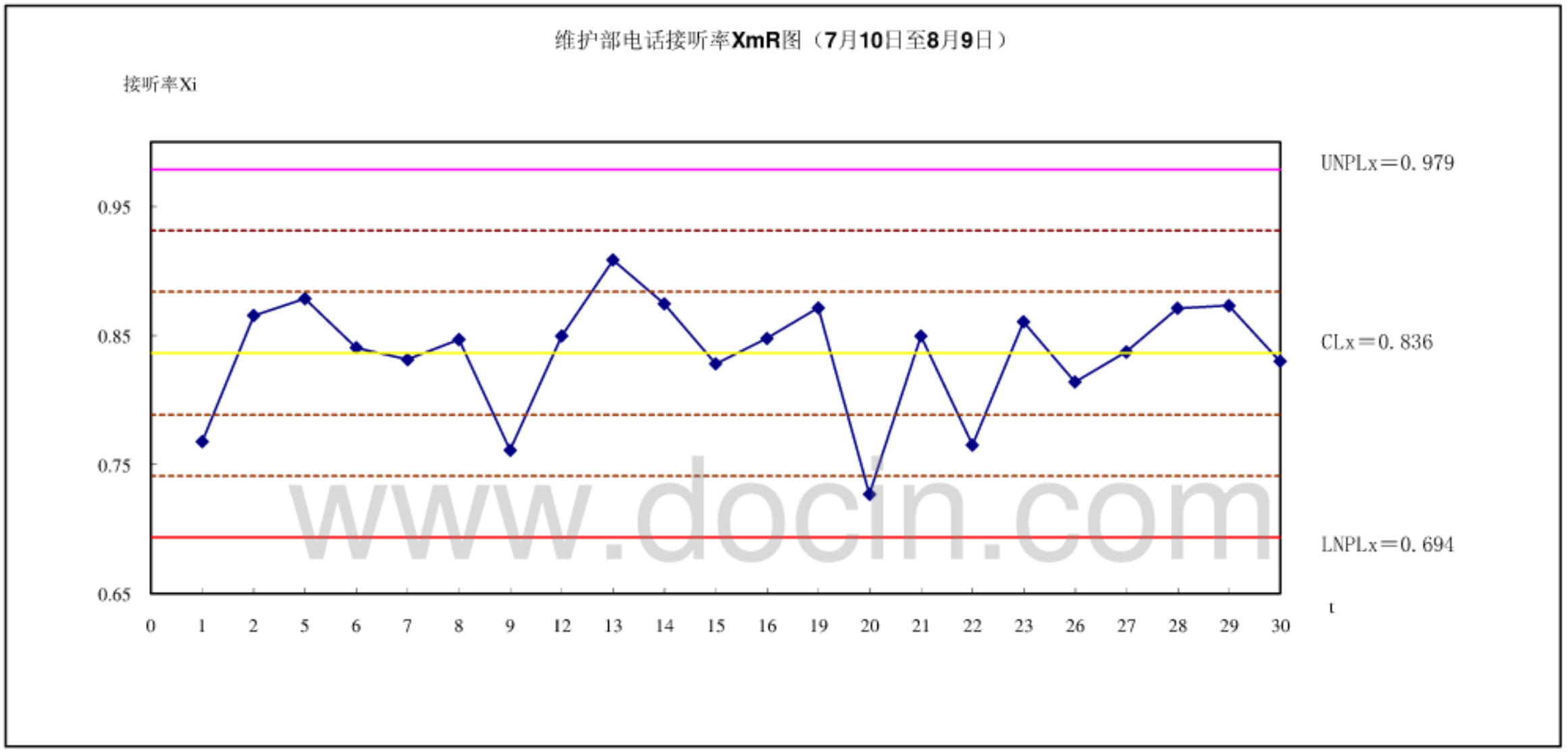
例 1：作为实例，取某公司客户中心从 7 月 10 日到 8 月 9 日之间的每日用户来电记录（如表 20-5 所示），用 XmR 图（如图 20-6 和图 20-7 所示）表示维护过程的行为，包括平均接听率、接听率波动的上下限以及过程稳定性。

图表 20-5 例 1 原始数据表

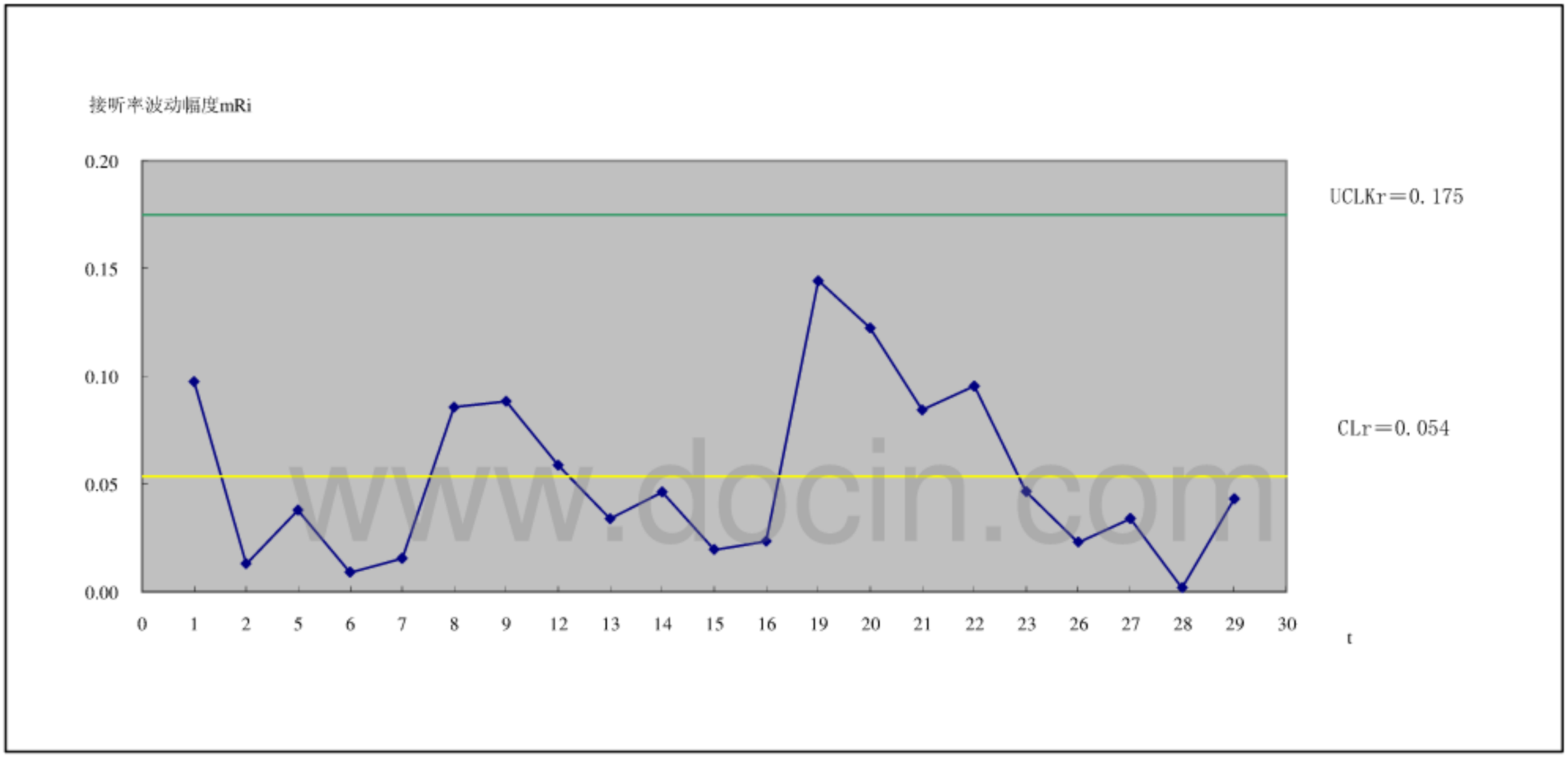
序号	日期	呼入次数	接听次数	平均谈话时间	座席数	Xi 接听率	说明
0							
1	20030710	857	658	3.8 分	18	0.77	
2	20030711	698	604	3.7 分	18	0.87	
5	20030714	674	592	3.8 分	17	0.88	
6	20030715	971	816	3.7 分	19	0.84	
7	20030716	640	532	3.9 分	17	0.83	
8	20030717	705	597	4.0 分	20	0.85	

9	20030718	1331	1013	2.8 分	16	0.76	
12	20030721	718	610	4.0 分	16	0.85	
13	20030722	720	654	3.6 分	18	0.91	
14	20030723	661	578	3.7 分	18	0.87	
15	20030724	698	578	3.8 分	18	0.83	
16	20030725	742	629	3.8 分	20	0.85	
19	20030728	699	609	3.7 分	17	0.87	
20	20030729	780	567	4.2 分	16	0.73	
21	20030730	704	598	3.7 分	17	0.85	
22	20030731	766	586	4.0 分	19	0.77	
23	20030801	767	660	3.8 分	18	0.86	
26	20030804	758	617	3.6 分	17	0.81	
27	20030805	712	596	3.6 分	17	0.84	
28	20030806	698	608	3.6 分	16	0.87	
29	20030807	630	550	3.6 分	17	0.87	
30	20030808	676	561	3.5 分	19	0.83	

图表 20-6 例 1 维护部电话接听率 XmR 图



图表 20-7 例 1 接听率波动幅度图



四、±3σ 经验规则

从例 1 的 XmR 图（见图 20-6）可以看出，全部观测点均落在平均值二边、上下控制限以内，可以说过程是稳定的；如果过程不稳定，则可能有完全不一样的分布情况。研究表明，在多种不同的随机分布模型下，一组同质的观测数据，均符合以下经验规则：

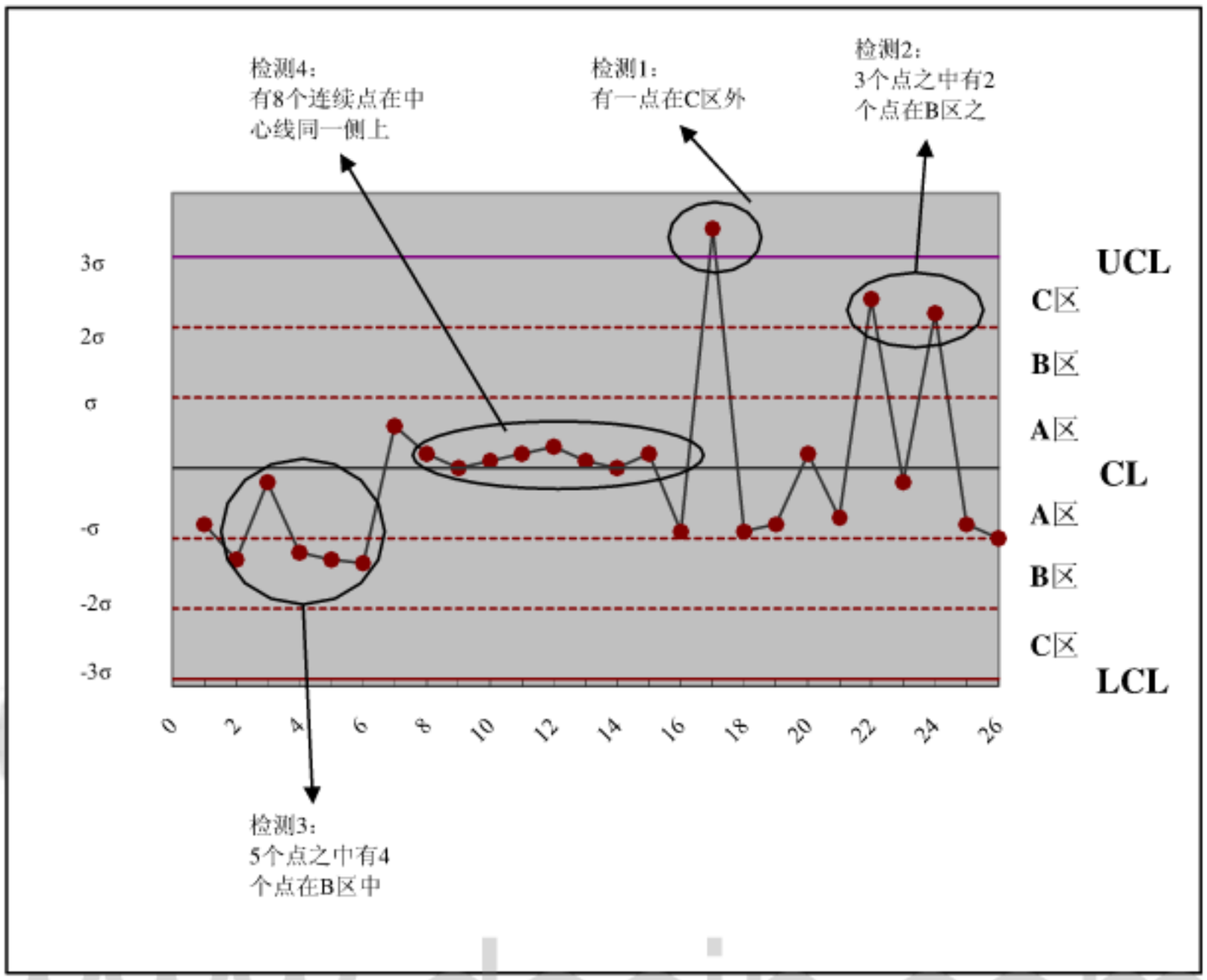
- 规则 1：大约 60%～70%的数据会处在平均值二边 1σ 单位距离内；
- 规则 2：大约 90%～98%的数据会处在平均值二边 2σ 单位距离内；
- 规则 3：大约 99%～100%的数据会处在平均值二边 3σ 单位距离内。

五、过程不稳定检测准则

按照±3σ 经验规则，粗略地说，分布在中心线二侧、上下限之间的观测点，都是属于正常的观测点，它们之间的度量偏差是由不可避免的随机因素造成的，此类偏差称普通原因偏差（common cause variation）。还有另外一类造成过程不稳定的原因，称可归属原因（assignable cause），为了改进过程行为，使其稳定，就应该识别这些可归属原因，消除这些原因使其不再重复出现。

利用控制图 XmR，怎样去找可归属原因呢？有人提出了四条检测准则，如图 20-8 所示（见下页）。一旦 XmR 图中出现这四种情况之一时，就应寻找可归属原因。

图表 20-8 归属原因检测准则图



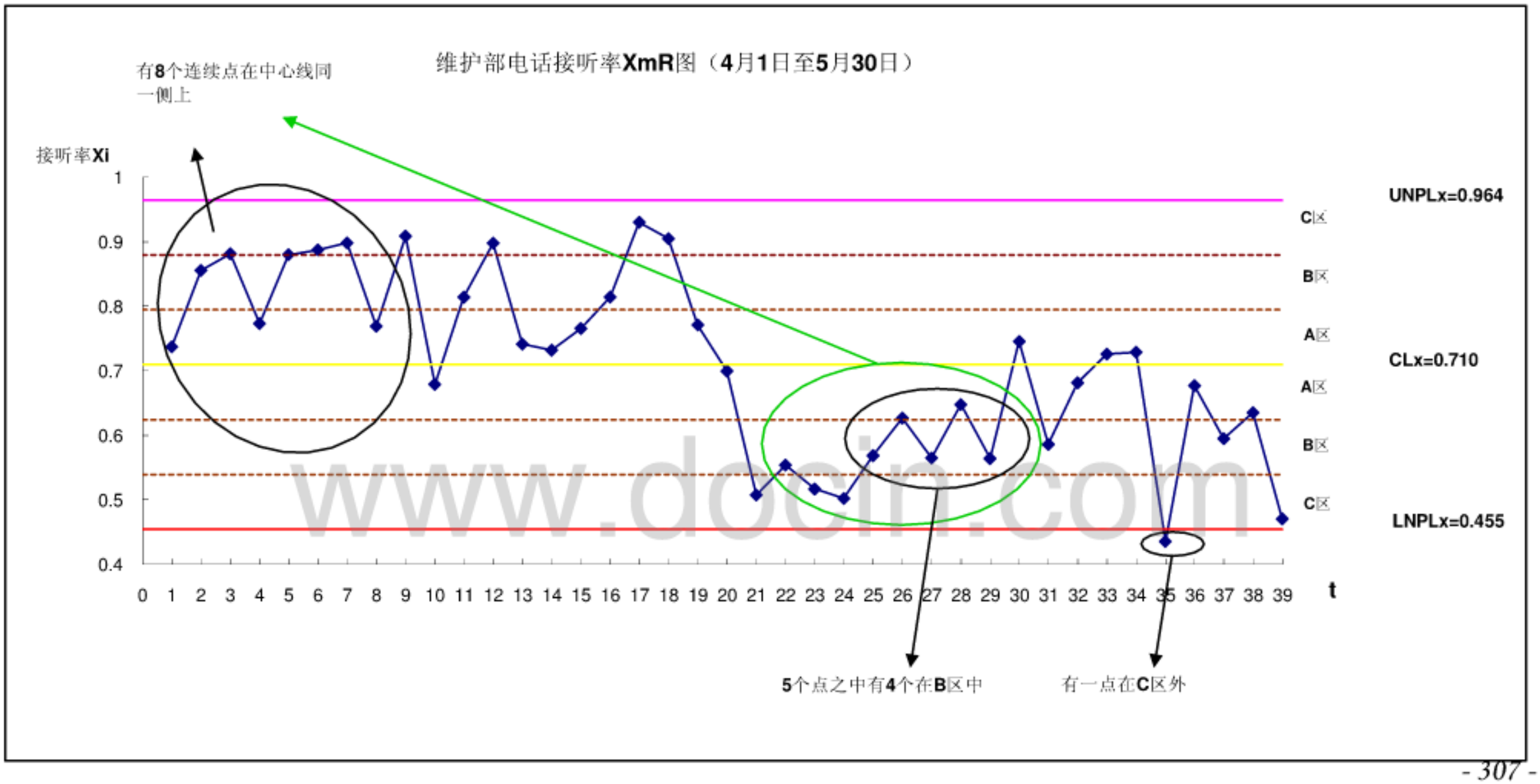
例 2：下表给出某公司客户服务中心从 4 月 1 日到 5 月 30 日之间的用户来电记录，用 XmR 图表示过程行为，并利用四条检测准则，识别可归属原因。

图表 20-9 例 2 原始数据表

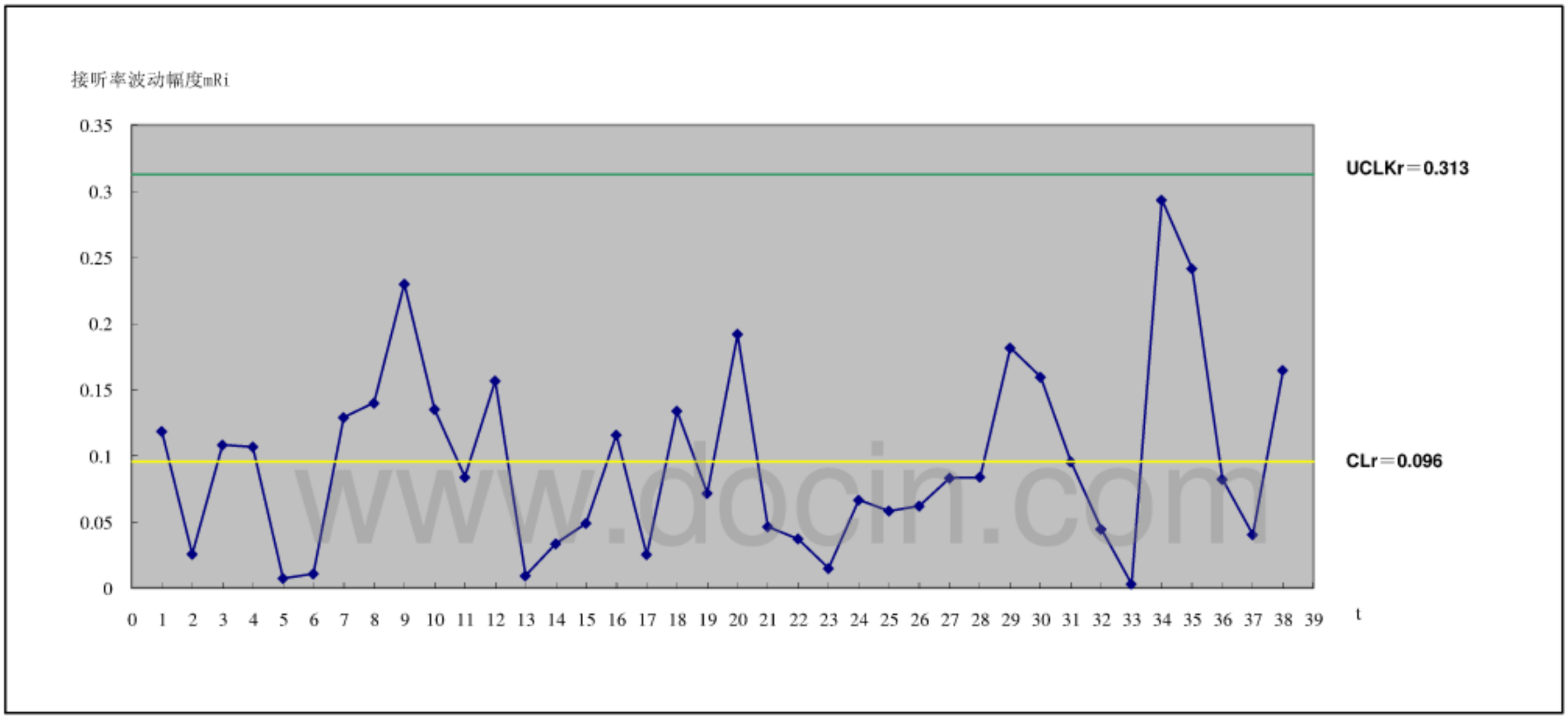
序号	日期	呼入次数	接听次数	谈话时间	座席数	Xi 接听率	说明
0							
1	401	925	682	3.9 分	19	0.737	
2	402	839	718	3.3 分	17	0.856	
3	403	1518	1338	2.9 分	17	0.881	
4	404	741	573	3.7 分	15	0.773	
5	407	683	601	3.7 分	15	0.880	
6	408	683	606	3.8 分	17	0.887	
7	409	588	528	3.9 分	17	0.898	
8	410	814	626	3.8 分	18	0.769	

序号	日期	呼入次数	接听次数	谈话时间	座席数	Xi 接听率	说明
9	411	757	688	3.4 分	17	0.909	
10	414	1006	683	4.0 分	15	0.679	
11	415	898	731	3.8 分	17	0.814	
12	416	744	668	3.9 分	16	0.898	
13	417	1171	868	3.6 分	19	0.741	
14	418	1228	899	3.5 分	17	0.732	
15	421	938	718	3.6 分	16	0.765	
16	422	781	636	4.0 分	17	0.814	
17	423	701	652	3.6 分	18	0.930	
18	424	1598	1446	2.8 分	18	0.905	
19	425	1603	1236	3.0 分	18	0.771	
20	426	772	540	3.5 分	15	0.699	
21	428	1393	707	3.5 分	19	0.508	
22	429	1087	602	3.5 分	18	0.554	
23	430	1140	589	2.9 分	17	0.517	
24	512	1285	645	3.3 分	18	0.502	
25	513	936	532	3.8 分	16	0.568	
26	514	857	537	3.3 分	16	0.627	
27	515	758	428	3.3 分	15	0.565	
28	516	792	513	3.6 分	17	0.648	
29	519	915	516	3.7 分	17	0.564	
30	520	668	498	3.9 分	17	0.746	
31	521	942	552	3.2 分	17	0.586	
32	522	788	537	3.7 分	17	0.681	
33	523	1226	890	2.9 分	16	0.726	
34	524	867	632	2.8 分	12	0.729	
35	526	1270	553	3.9 分	16	0.435	
36	527	808	547	3.8 分	17	0.677	
37	528	928	552	3.9 分	15	0.595	
38	529	866	550	3.7 分	16	0.635	
39	530	1252	589	3.4 分	15	0.470	

图表 20-10 例 2 维护部电话接听率 XmR 图



图表 20-11 例 2 接听率波动幅度图



从例 2 的 XmR 图（见图 20-10 和图 20-11 所示）可以看出，过程并不稳定，性能（过程行为）并不理想。左右二半呈明显的阶梯状；右半的数据造成左半数据的均线下移，从而使左上角 8 个连续点落在中心线上侧；同样，左半的数据造成右半数据均线上移，从而右下角 8 个连续点分布在中心线下侧。实际上，问题要从第 18 点到第 21 点的接听率明显下降寻找原因。查当时值班记录，发现如下意外事件：

图表 20-12 例 2 中通过 XmR 图查找到意外事件列表

序号	日期	意外事件
18	4/24	深交所通知错发 101903 的国债利息记录
19	4/25	海通证券上三板系统
20	4/26	B 股新版数据接口规范全网测试
21	4/28	QYB1.0 优化版将当晚的股票 000930 余额误删，涉及所有相关用户
34	5/24	全网测试，包括 B 股接口规范和上市公司收购业务
35	5/26	深交所正式启用要约收购业务

从上表看，除 4 月 28 日（第 21 点）由程序错误引起以外，其余均由外部原因引起而又没有及时采取应对措施（如增加席位、及时修改程序等），造成接听率明显下降。

六、过程能力分析

过程能力是一个受控（稳定）过程可预测性能的量化表示，对于适宜于用 XmR 图表示其行为的过程而言，过程能力的具体体现则由过程控制图的中心线和上下控制限组成。至于说某个过程是“有能力的”，则必须满足以下二个条件：

- 过程进入受控（即稳定）状态必须有一段足够的时间，以便在运行过程中可以容易发现异常行为；
- 过程能力必须达到或超过规定值，并且此规定值必须满足机构商业/技术目标或用户需求。

以例 1 为例，从 XmR 图可以得知，此时过程能力—接听率 X，达到如下水平：

0.836-0.143<X<0.836+0.143，即 0.69<X<0.98

如果根据商业目标或用户需求，公司规定接听率应满足如下要求：

0.63<接听率<0.93

那么，从 XmR 图可以得知，过程是稳定的，并且还是有能力的。可是如果机构规定：

0.80<接听率<0.99

那么过程能力就达不到要求了。应该采取措施，增加均值 CLx 或者缩小偏差。问：在技术支持过程没有得到改进的前提下，增加席位数、或者按时段分布科学安排值班，是否可以改进接听率？看下面二个例子。

例 3:将某公司客户服务中心从 2 月 10 日到 8 月 9 日的用户来电记录按席位数重新分组，然后采用回归分析方法，找出接听率对席位数的依赖关系。

图表 20-13 例 3 原始数据表

席位数 / 接听率	15	16	17			18		19	20	21
	0.773	0.858	0.792	0.898	0.591	0.898	0.502	0.988	0.914	0.961
	0.88	0.797	0.916	0.909	0.708	0.96	0.735	0.927	0.933	0.872
	0.679	0.858	0.921	0.814	0.782	0.983	0.768	0.96	0.871	0.928
	0.669	0.898	0.849	0.732	0.774	0.847	0.865	0.737	0.847	
	0.565	0.765	0.94	0.814	0.888	0.908	0.908	0.741	0.848	
	0.595	0.568	0.927	0.517	0.712	0.919	0.874	0.508	0.889	
	0.47	0.627	0.98	0.648	0.713	0.938	0.828	0.63		
	0.684	0.726	0.961	0.564	0.773	0.926	0.86	0.84		
		0.635	0.946	0.764	0.707	0.941				
		0.899	0.943	0.586	0.586	0.936				
		0.697	0.926	0.681	0.811	0.924				
		0.728	0.863	0.677	0.878	0.769				
		0.651	0.856	0.591	0.831	0.93				
		0.743	0.881	0.735	0.871	0.905				
		0.768	0.887	0.61	0.849	0.771				
		0.763	0.87	0.84	0.81	0.554				
		0.761								
		0.85								
平均值	0.664	0.755	0.79			0.85		0.791	0.884	0.92

经统计，可知：接听率＝0.03664×席位数＋0.148

从上列公式可以看出，增加一个席位可以增加接听率 3.664 个百分点。可见，保持一个一定数目的席位数（例如 20 个）并且确保每个席位的有效服务时间（不会长时间离岗），明显有利于提高接听率。特别是，当外部条件有变化（如交易规则修改）或者优化包发放时，

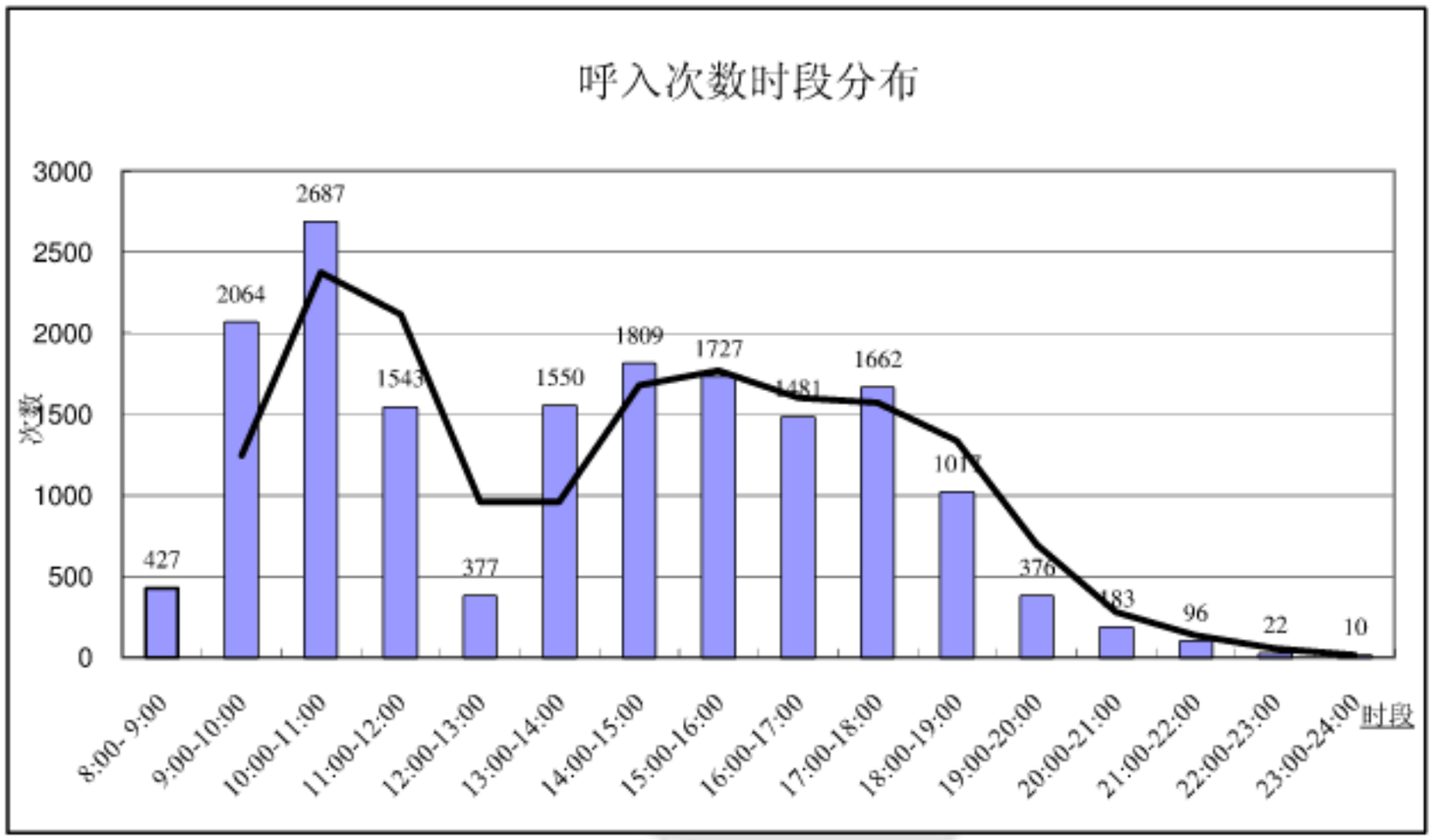
临时增加席位（如质保人员参加一、二天维护值班），明显有利于保持或提高接听率。

例 4：将某公司客户服务中心从 07 月 10 日至 08 月 09 日的来电记录，按一天内不同时段分组，看看能得到什么启示。

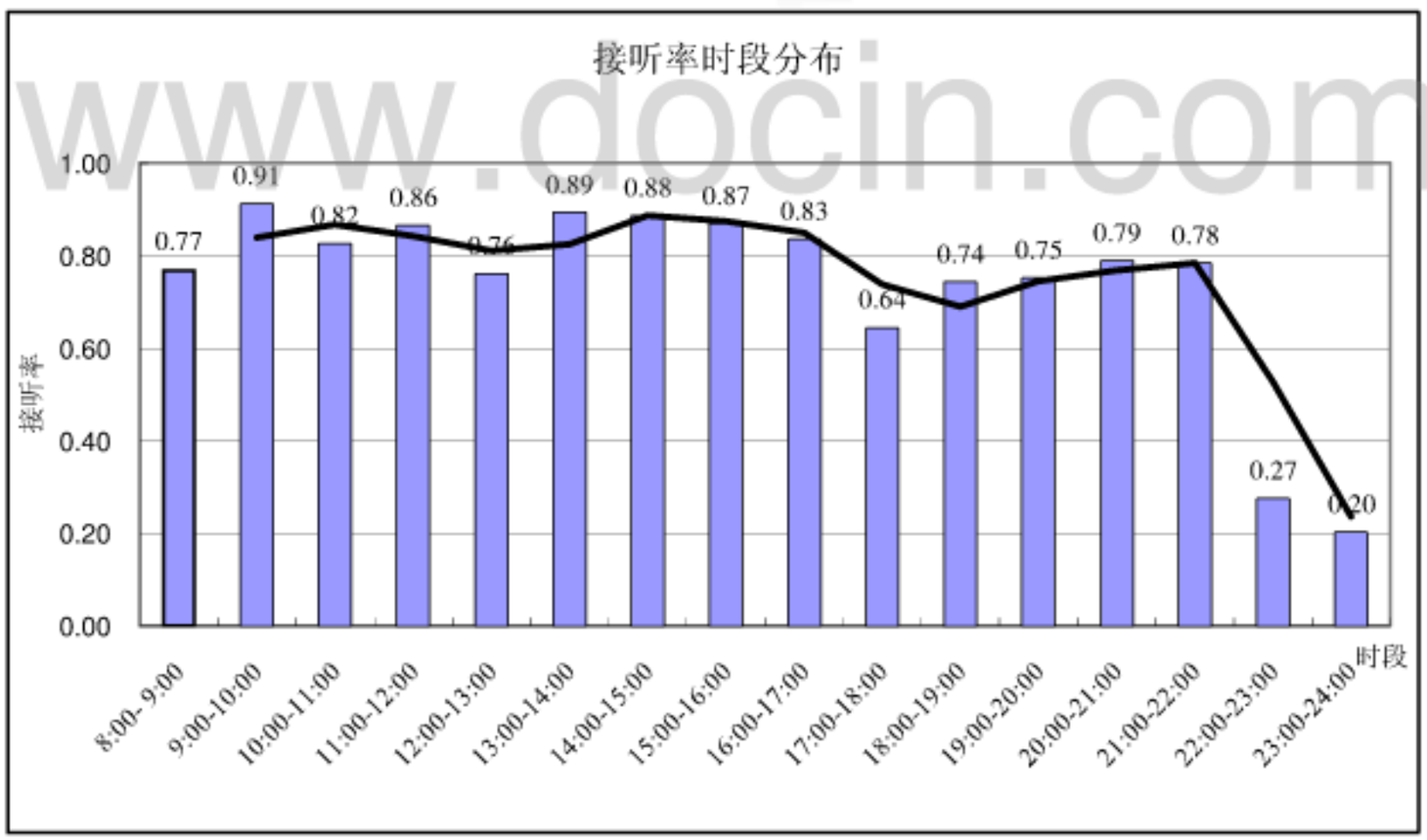
图表 20-14 例 4 原始数据表

时段	呼入次数	接听次数	谈话时间	平均谈话时间	接听率
8:00- 9:00	427	328	18 时 1 分 5 秒	3.3 分	0.77
9:00-10:00	2064	1879	109 时 55 分 20 秒	3.5 分	0.91
10:00-11:00	2687	2215	136 时 17 分 33 秒	3.7 分	0.82
11:00-12:00	1543	1332	77 时 41 分 57 秒	3.5 分	0.86
12:00-13:00	377	286	16 时 41 分 44 秒	3.5 分	0.76
13:00-14:00	1550	1381	85 时 54 分 53 秒	3.7 分	0.89
14:00-15:00	1809	1600	97 时 48 分 43 秒	3.7 分	0.88
15:00-16:00	1727	1497	94 时 41 分 5 秒	3.8 分	0.87
16:00-17:00	1481	1235	73 时 0 分 16 秒	3.6 分	0.83
17:00-18:00	1662	1068	69 时 0 分 33 秒	3.9 分	0.64
18:00-19:00	1017	753	48 时 22 分 42 秒	3.9 分	0.74
19:00-20:00	376	282	19 时 11 分 55 秒	4.1 分	0.75
20:00-21:00	183	144	10 时 53 分 31 秒	4.5 分	0.79
21:00-22:00	96	75	5 时 16 分 29 秒	4.2 分	0.78
22:00-23:00	22	6	20 分 40 秒	3.5 分	0.27
23:00-24:00	10	2	5 分 50 秒	2.9 分	0.20
	17037	14083	863 时 14 分 16 秒	3.7 分	0.83

图表 20-15 例 4 呼入次数时段分布图



图表 20-16 例 4 接听率时段分布图



从呼入次数时段分布图（见图 20-15）中看出：呼入次数的分布有两个比较明显的波峰，第一个出现在早上 9：00－11：00，第二个出现在下午 14：00－18：00，午餐时段（12：00－13：00）和晚餐时段（18：00－19：00）以及其他时段的呼入次数相对来说就少了。

比较接听率时段分布图（见图 20-16），有三个比较明显的波峰，主要是因为午餐和晚餐时段席位数相对较少。在呼入次数时段分布图中，晚餐时段的呼入次数并没有明显的下落，这个时段的接听率有明显的下降。呼入较为集中的早上 9：00－10：00 和下午 14：00－15：00，接听率在全天的接听率中属于较高的。但是 10：00－11：00 的接听率有所下降，这个时段的呼入次数是一天中最高的。

此外，还有几个情况值得注意：

- 8：00－9：00 时段，呼入次数并不高，但是为什么接听率偏低？
- 同样，12：00－13：00，呼入次数也不高，但为什么接听率也偏低？午餐和午休时间能否科学安排？
- 下午临下班时，17：00－18：00 呼入次数较高，但接听率偏低，是否与公司 17 点下班或晚饭时间有联系？是否可以适当安排而改进？
- 晚上 6：00 到 8：00 接听率低，能改进吗？

第21章 决策分析

内容提要：

- **CMMI** 中对应实践
- 决策分析简述
- 决策分析活动
- 关于“蓝海战略”

决策，就是人们对未来的行为确定目标，同时为实现目标选择最佳的行动方案和策略，并在实施方案的过程中跟踪监督，确保目标的有效实现。各级管理者的诸多职责中，首要职责就是决策；决定管理者核心能力的诸多因素中，首要因素就是决策能力。不论管理者做什么，都是通过决策进行的。管理就是决策，管理就是一个决策过程。

决策可以并且应该从不同维度进行分类，分类不同决定了职责和权限的区别，分类不同决定了方法和流程的差异。例如，应该由哪一级制定决策；涉及哪些活动或影响哪些活动，因此哪些管理者应参与决策，至少决策前应征询他们的意见；决策制定后应告知哪些决策者；等等。可以从几个不同维度考虑决策的分类；

- 决策影响的未来时间跨度；
- 决策对公司或其他部门的影响程度，一个部门流程或绩效的优化不能以牺牲其他部门为代价；
- 决定决策的关键因素，例如关系到企业价值观的决策就应由最高一级决定；
- 经常性决策或者偶尔为之的特殊决策；
- 决策实施涉及的纵向关系和横向关系。

就决策涉及的范围而言，大体分四级：公司一级、事业部或职能部门二级、研发部三级和项目组四级。尽管无法详尽预测决策的内容和方式，但经过分析，可以发现：十有八九的

决策基本上均可纳入为数不多的几个典型类型，并且往往将决策级别定得过高。甚至由于个别决策者的个性偏好，造成决策延误，甚至该决策而不决策但紧握着决策权不放，丢失机会。原则上，决策权应尽可能下放到最低一级，越接近决策实施行动的现场越好。

21.1 CMMI 中对应实践

在 CMMI 中有一个专门的过程域对应于决策分析，即 DAR（决策分析与解决方案），其目的是通过正式评估过程，根据已建立的准则，评价被识别的多种候选方案，分析可能的决策，确定最佳的解决方案。

SG 1 Evaluate Alternatives（评估候选解决方案），利用评估准则，评估候选方案，依据评估结果做出决策。

SP 1.1 Establish Guidelines for Decision Analysis（建立决策分析的指导），建立和维护（决策分析）指南，决定哪些问题应通过正式评估过程。可以通过如下几步完成该实践：一是建立（决策分析）指南；二是将指南集成到已定义过程的合适地方。

SP 1.2 Establish Evaluation Criteria（建立评估标准），建立和维护评价候选解决方案的准则以及这些准则的分级排序。一般形成“文档化的评估准则”和“准则重要性分级”两个工作产品，可以通过如下几步完成该实践：一是定义评估候选解决方案的准则；二是为了对评估准则进行分级，定义标尺和幅度；三是对准则分级；四是评价准则及其相对重要性；五是改进评估准则以改进其有效性；六是记录选择和拒绝评估准则的原因。

SP 1.3 Identify Alternative Solutions（识别候选方案），识别解决问题的候选解决方案。可以通过如下几步完成该实践：一是检索文献资料；二是识别多种可能的候选解决方案；三是记录被推荐的候选解决方案。

SP 1.4 Select Evaluation Methods（选择评估方法），可以通过如下几步完成该实践：一是基于分析决策的目的以及用于支持该方法的信息的可获得性，选择评估方法；二是选择评估方法，还应考虑使用该方法时能否专注于问题本身而不会受到次要问题的过多影响；三是定义用于支持评估方法的度量。

SP 1.5 Evaluate Alternatives（评估候选方案），使用已建立的准则和方法，评估候选解决方案。可以通过如下几步完成该实践：一是使用已建立的准则和选定的方法，评估被推荐的候选解决方案；二是评估与评估准则相关的假设以及支持假设的论据；三是评价候选解决方案评分值的误差是否影响评价（结果），并进行必要的处理（如重新评估、修改评估准则）；四是必要时，执行模拟、模型化、原型和典型试验，演习评价准则、方法以及候选解决方案；五是如果被提议的解决方案测试结果不理想，则考虑新的候选解决方案、准则或方法；重复评价过程直到候选解决方案测试成功；六是记录评价结果。

SP 1.6 Select Solutions（选择解决方案），可以通过如下两步完成该实践：一是评估实现推荐解决方案的相关风险；二是记录推荐解决方案的结果和理由。

21.2 决策分析简述

决策分析与解决方案是提供一种正式的处理过程，对两个或两个以上的候选方案进行决策分析，以此尽量去除主观的判决，使用客观数据，达到选择最优方案的目的。

一般应用于机构内所有重大决策过程，不只是软件开发类的工作，其中包括机构的发展方向、项目中组件的“Make or Buy”、新产品或新业务立项等。

按决策目标的不同特征，决策可分为四种类型：

- 接受/拒绝性决策，决策者必须接受决策或拒绝决策，二者必选其一。
- 评价性决策，决策者必须基于对某些实体的价值评估而进行一系列的活动。例如，人事任免、级别升降、先进评选、荣誉授予，以及只有一个可选方案的决策评价等。
- 选择性决策，决策者必须通过对二个或多个可选方案的对比评价，按约定的准则择优选定一个方案或可选方案的一个子集。例如，员工住房公积金调整、设备采购毛利底线的选定，等。接受/拒绝性决策可视作选择性决策的特例。
- 结构性决策，决策者必须依照问题需求及其目标、边界约束、使用可用的资源、恰当组织各个可选的主题、做出决策并确保决策的有效实施。例如，质量体系建设，CMMI 引入、项目成本管理、信息化推进等。

决策者和决策管理者，应共同对决策的有效性负责，努力提高决策质量，敢于承担责任，勇于修改错误。决策质量，取决于下列因素：

- 确实满足公司实际发生的需求；
- 可以达到决策者事先制定的目标；
- 决策相关各方对决策所带来结果的综合评价满意（或积极）；
- 选定方案的结果至少不会比选择其他方案的结果更差；
- 决策过程成本低，即决策过程消耗最少的资源。

21.3 决策分析活动

公司的决策分析过程中，不只是一个简单的操作活动，实际上是要考虑到博弈论、社会心理学等各方面的基础之上进行。决策者和决策管理者应遵循科学的决策过程，努力提高决策过程质量。应当注意避免出现失误，出现失误的原因主要有：不科学的决策方法；迫于压力的仓促判断；不恰当的资源配置。

重大问题的完整决策过程，经历如下五个阶段，分别是建立评估准则、确定候选方案、确定评估方法、评估候选方案、做出决策，具体流程如图 21-1 所示（见下页）。

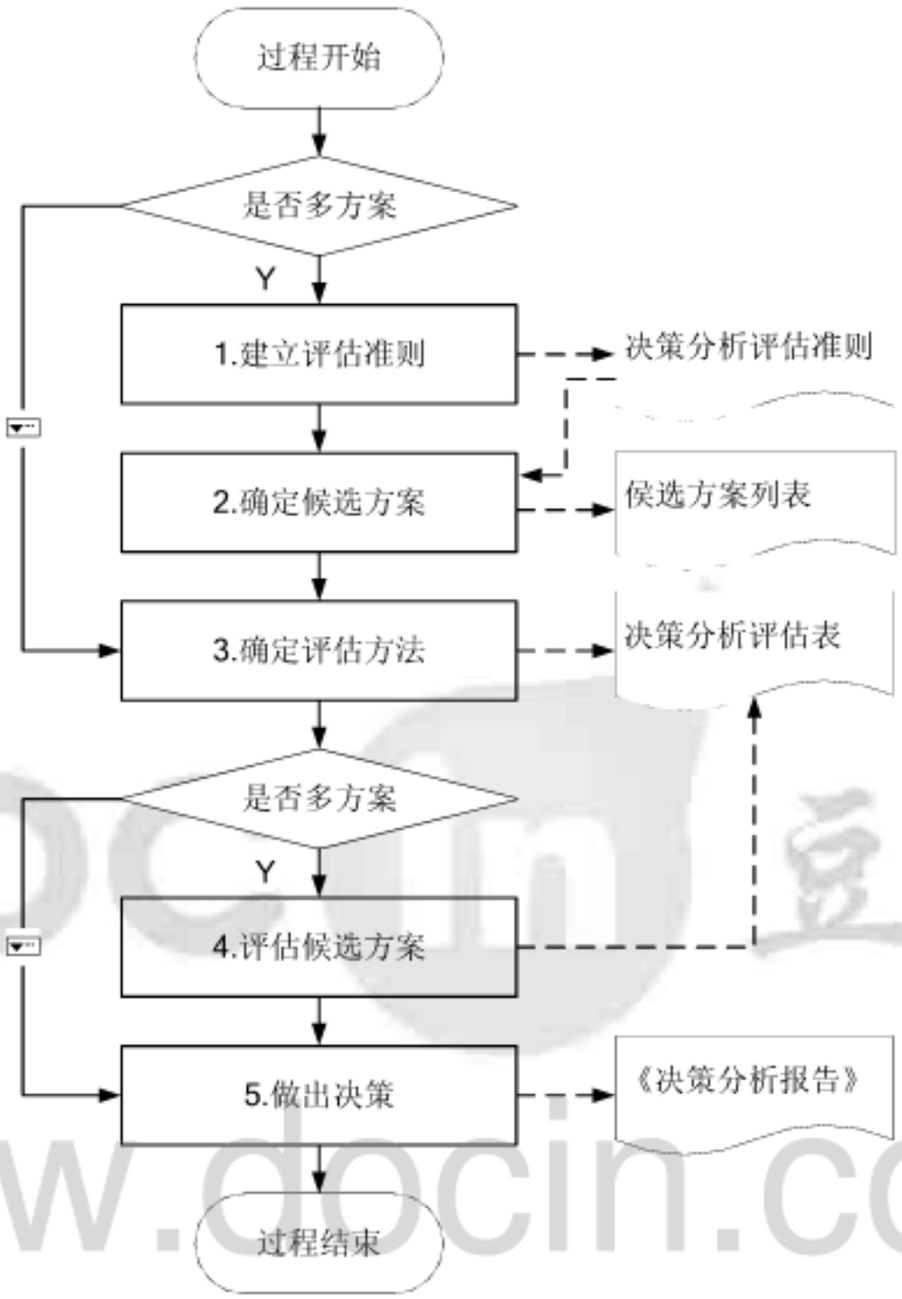
一、建立决策分析评估标准

决策问题的提出者从现象出发，分析关键因素，确定问题性质，明确决策需求。在决策分析评估过程中，首先需要确定本次评估过程中用以确定候选方案的评估准则，并以准则的先后顺序体现准则的优先级。当需要对单一方案进行决策（例如：是否立项、“Make or Buy”）时，要确定评估方法并做出决策即可。

二、确定候选解决方案

参与评估的人员按照评估准则讨论并确定需要进行决策的解决方案。将方案填写在《决策分析报告》的“候选方案列表” 中。按照决策分析评估准则识别方案与准则的符合度，确定候选方案。此处需要注意：不采取任何行动也是一个可能的方案，并且防止过多方案的制定和评选而造成决策成本的浪费。

图表 21-1 决策分析活动流程图



三、确定评估方法

决策参与人员，根据待决策问题的特征，确定评估指标。并为每项评估指标分配适当的权重。将信息填写到《决策分析报告》的“决策分析评估表”中。在此过程中，需要注意以下几个方面：

- 预测决策可能带来的后果；进而判断后果发生的可能性多大。
- 了解、分析决策相关各方对决策后果可能作出的价值判断。
- 优选最合适的解决方案。按照问题的重要性、紧迫性、执行难度和成本等多个方面，从风险、效益、时机和资源等多个因素，综合权衡。

四、评估候选方案

决策参与人员根据其经验为每项候选解决方案的每项评估指标打分。并将结果记录在到方案评估表中。 计算公式一般为：候选方案 1 得分=指标 1×权重 1×得分 + 指标 2×权重 2×得分 + …… +指标 n×权重 n×得分。

在这个过程中，不只是投投票、打打分那么简单，参与决策评估的人员应当注意避免 7 个决策陷井：不能充分了解并妥善协调决策相关各方的关注点和真实需求；迫于压力，仓促决定，最终只能用劝服加/或命令手段实施决策；决策目标模糊不清；急于求成，缺乏多个可选方案特别是创新方案的研究；错误运用评估手段，忽视不同意见或反面意见；忽视道德风险和社会责任；不善于从错误中学习。

五、做出决策

比较每个方案的综合分值，选择最优的方案。最优方案的判定方法为候选解决方案中得分最高的那个方案。如果出现评分相近的候选方案，则需要考虑相应评估准则。再根据其满足评估准则的情况选择最优方案。如果候选方案不能满足要求，则考虑新的方案、准则和方法，直到选出合适的方案。决策问题的提出者文档化推理过程和判断结果，编写《决策分析报告》。

在这个过程中各级管理者，特别是高层管理者，应十分重视营造民主、科学的决策氛围。特别强调以下几点对公司决策是非常重要的：

- 决策参与者应以公司/部门整体利益为准则，对每一个决策问题均应认真准备并发表意见，充分沟通，冷静协商，让步、妥协，适时做出有效决策。
- 尊重个人见解。个人见解常常是因个人的经历、经验、知识、专业、分管范围等多个方面的不同而产生的对问题的直观想法，不一定要有充分的事实为依据。许多有效的决策，往往是从多种不同且互相冲突的个人见解中产生的。因此，应努力营造一个人人都能发表个人见解的氛围。提倡“见解为先”。遵从决策的首要原则：没有不同的见解，就不可能有决策。
- 善用反面意见。在重大问题的决策过程中，应允许、鼓励并善用反面意见。与其怀疑反面意见提出者个人的不良动机，不如相信个人对公司或公司事业的忠诚。要注

意，在公司里“什么是正确的”的问题比“谁是正确的”的问题更重要。

- 重视决策相关各方的关注点、实际需求和价值判断。力求避免仓促决策后单纯使用说服和命令的手段实施决策，造成决策失效。
- 涉及公司/部门整体利益的重大决策，应重视风险评估和应对措施；同时，涉及公司/部门整体利益的风险评估和应对措施，本身就应列入重大决策的范畴。
- 决策需要勇气。不论是规避灾难的决策，还是抓住机会的决策，都应强调：该决断就要决断。
- 一经决策，就应着力于决策的实施，化决策为行动。为此，必须明确：谁应了解这项决策、应采取什么行动、谁采取行动以及这些行动为何进行。决策者不能或明或暗的支持另外一种与决策不相符合的行为，从而为决策的有效实施增加障碍，并且久而久之会使员工和下级管理者产生一种印象：说一套，做的又是另一套。
- 每一项重要决策，均应有相应的信息反馈措施，对决策实施过程进行跟踪，对决策的预期结果做实际印证。需知，决策是人做的，人难免会犯错误；再英明的决策，也不可能永远正确；再有效的决策，总有一天会被新的决策所替代。强调，决策者应对决策的实施结果亲自检查、定期回顾，及时地修改早该修改的决策，及时地补充早该补充的决策，及时地废弃早该废弃的决策。

21.4 关于“蓝海战略”[23]

公司愿景、使命和战略发展规划的制定和实施，是公司一级的最大决策。按哪一种战略思路进行决策，是一个大问题，特别要认真对待。

传统的战略思路，以竞争为前提，聚焦生产力，采用产品差异化或成本领先或者二者折中的常见手段，追求产出最大化、投入最小化，通过产品、价格、质量、成本、服务、产品差异、市场分割等等方式打败竞争对手，确立企业自身的产品或服务在市场中的独特定位，最大限度地占有市场份额。

[23] 本节内容引自于胡希明老师的培训讲义

可惜，产品差异化意味着成本增加，而成本领先的策略往往限制了利润率的增长。随着越来越多的企业参与拼抢和瓜分有限的市场份额和利润，企业最终取得的获利空间就越来越小。近 20 年来，中国有多少 IT 企业特别是软件企业总是重复着各领风骚三五年的悲剧命运，究其原因，很大程度上与传统的战略思路有关。

【金伟灿（W.Chan Kim，美籍韩裔）、芮妮.莫伯尼（Renée Mauborgne，美籍欧裔）合著《蓝海战略》（Blue Ocean Strategy）一书，由哈佛商学院出版社于 2005 年 2 月出版。出版后，很快就在世界范围内获得了热烈反响，先后被评为“全美畅销书”、“全球畅销书”，据统计迄今已被译成 27 种文字，打破了该出版社有史以来国际版权出售记录。《蓝海战略》所引起的迅速和热烈的反响，充分反映了在当今的激烈竞争市场环境中，全球企业界对寻求新的战略思路以实现生存、发展以至长盛不衰的强烈渴望。】

《蓝海战略》为企业提供了新的战略思路，即蓝海战略。这是一种新的战略思路，要求企业：

- 将视线从市场的供给一方移向需求一方；
- 从关注并比超竞争对手转向为买方提供新的价值；
- 分析、筛选和重新排序决定买方价值的关键因素，剔除和降低现有的某些价值因素，增加和创造现有产业未提供的某些价值因素，创造新的价值曲线；
- 跨越现有市场边界，发现潜在需求，开创、重建市场和产业边界。

最终使企业摆脱红海“已有市场空间”的血腥竞争，开创无竞争或少竞争的蓝海“新的市场空间”。红海战略与蓝海战略的根本区别，如表 21-2 所示：

图表 21-2 红海战略与蓝海战略区别表

红海战略	蓝海战略
竞争于已有市场空间	开创无人竞争的市场空间
打败竞争对手	摆脱竞争
开发现有需求	发现、获取和开创新需求
只能在差异化（效用价值）和低成	增加效用价值与降低成本可以同

蓝海战略六原则，包括：

- 重建市场边界，摆脱竞争，开创蓝海。降低机会寻找风险。

- 注重全局而非数字。降低计划风险。
- 超越现有需求。降低规模风险。
- 遵循合理的战略顺序。降低商业模式风险。
- 克服关键的组织障碍，将战略贯彻在行动中。降低机构风险。
- 要想取得员工的信任和忠诚，自愿合作，企业应战略执行建成战略的一部分，采用公平过程制定和执行战略。降低管理风险。

检验蓝海的准则，概括说，有三：

- 发现和创造新的需求；
- 增加消费者的好处和企业的利润；
- 建立有效的进入障碍，将潜在竞争者挡在外面，使他们看得到但一时不容易跟进来。

《蓝海战略》一书有许多新的观点、理念和论点，此处摘抄几段。

- 企业家们追求卓越，希望基业常青。然而，历史表明：没有永远卓越的产业；也没有永远卓越的企业。
- 想在竞争中求胜，不能只顾着打败对手；要在未来赢得胜利，打败对手的唯一方法就是摆脱或停止竞争。
- 大部分蓝海是通过扩展现有红海的产业边界而创造出来的；由于蓝海的竞争规则尚未形成，因此无从竞争。
- 价值创新，聚焦于为顾客和公司创造价值提升，进而开创无人竞争的市场空间，是蓝海战略的基石。
- 只有当企业把创新与效用、价格、成本综合权衡时，才能实现价值创新。如果创新不与买方价值（效用和价格）、企业价值（价格和成本）相结合，那么技术创新者和市场先驱者往往会落到为他人做嫁衣的下场。
- 采用“战略草图”（Strategy Canvas）和“四步行动框架”（Four Actions Framework），分析产业现有的战略逻辑和商业模式，重购买方价值要素，才能创造新的价值

曲线。

【四步行动框架：第一步，剔除（eliminate）所在行业中企业长期竞争攀比而形成的价值要素。这些要素往往被认为是理所当然的，虽然它们不再具有价值，甚至还减少价值。第二步，降低（reduce）某些要素的要求到行业标准以下。例如，为了打败对手，现有产品或服务是不是在功能设计上过了头，企业给顾客的超过了他们的需求，徒增企业成本却没有好效果。（以上二步，旨在明白如何将成本降到竞争对手之下。）第三步，提升（raise）某些要素到行业标准以上。第四步，创造（create）未曾有过的某些要素。（后二步，旨在提升买方价值。）

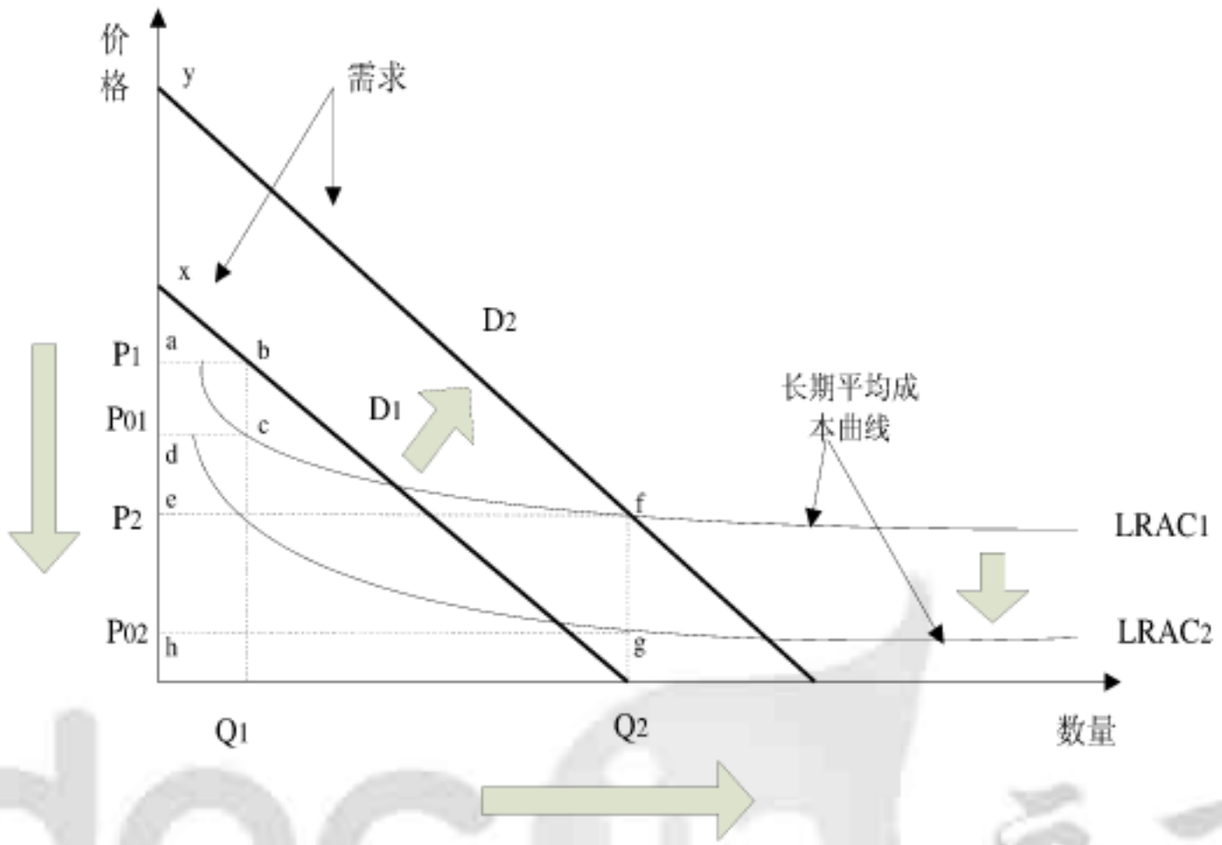
- 蓝海战略的基本特征有三点：重点突出，独树一帜，主题词令人信服。
- 制定蓝海战略之后，必须执行这个战略。无论是在红海中还是蓝海中，企业就如同个人一样，将想法转变为行动都很艰难。然而，与红海战略相比，蓝海战略意味着对现状和习惯的重大改变，就加大了执行的难度。必须克服战略执行中面临的四重组织障碍：认识障碍，资源障碍，动力障碍，政治障碍。
- 只有当企业的所有成员都团结到企业新战略旗帜下并在任何情况下都支持它时，企业才能脱颖而出。克服战略执行中的组织障碍只是向此目标靠近的重要一步。企业最终需要求助于企业广大员工的态度和行动。因此，企业应遵循公平过程，贯彻三 E 原则，强调广泛参与，做好解释沟通，明确目标，使尽可能多的员工接受或认同公司战略，从被迫执行转向自愿合作去实施公司战略。

【三 E 原则：Engagement，广泛参与；Explanation，解释沟通；Expectation，明确期望。】

- 一项蓝海战略自然会给模仿者设置许多进入壁垒。当企业的新的价值曲线重点突出、另辟蹊径、有令人信服的主题时，就应克服再次创新的诱惑，而要集中力量，不断扩大规模和范围，乘胜追击，将优势保持越久越好。
- 随着竞争加剧，蓝海又将变成红海。当新对手的价值曲线趋向重合时，企业就应开始寻求又一次的价值创新，开创新的蓝海，再次超前一步。

其实，从经济学理论角度看，蓝海战略的关键可以用图 21-3 来表示：

图表 21-3 蓝海战略关键因素图



从图 21-2 可见，“蓝海战略”的价值创新，具体表现在需求曲线（D1/ D2）、长期平均成本曲线（LRAC1/ LRAC2）、价格（P1/P2）、销售数量（Q1/Q2）、消费者感受到的好处（△Yef/△Xab）及获利（□efgh/□abcd）的改变。概括地说，蓝海战略的秘诀就在于：

- 通过价值曲线的改变，$D_1 \rightarrow D_2$；
- 使成本和价格下降，$LRAC_1 \rightarrow LRAC_2$，$P_1 \rightarrow P_2$；
- 而买方价值提升，△Xab→△Yef；
- 最终结果是产出扩大，利润增加 ，$Q_1 \rightarrow Q_2$，□abcd→□efgh。

第22章 附录

22.1 CMMI 模型的部件[24]

CMMI 模型，从标准文本角度看，如何构成？要点如下：

- CMMI 模型是一个模型系列，包括多个学科。
- 每个学科模型都有二种表示法，相应的有二套标准文本，每种表示法均包含相同数量和相同名称的过程域。
- 每个学科的二种标准文本结构一致，内容部分相同。
- 阶梯式表示法，将全部过程域划分为 5 个成熟度等级，每个等级包含若干个数量不等的过程域，每个过程域包含若干个目标以及为达到每个目标应执行的若干个实践。
- 连续式表示法将全部过程域按内容相关性分成 4 类，每个过程域的能力等级分为 6 级，每个过程域包含若干目标，每个目标包含若干个实践，每个能力等级分别对应 1 个到 5 个目标。

更深入一层，每个过程域的标准文本又是如何构成的？这就是本章要讲的模型部件，即构成模型的基本要素。

每个过程域的标准文本均由三大部分文字组成，分别称为：

- 必需部件（Required Components）
- 期望部件（Expected Components）
- 解释性部件（Informative Components）

在机构过程改进活动中，不同角色的人员可以有重点地了解上列部件的内容。例如，对于机构的高层管理者而言，可以浏览全部过程域的必需部件，从而使自己对 CMMI 的了解达到“主管概览”级；而部门经理而言，可以浏览全部过程域的必需部件和期望部件，使自己

[24] 本节内容引自于胡希明老师的培训讲义。

对 CMMI 的了解达到“经理概览”级。

22.1.1 必需部件

每一个过程域都规定了一个或多个应达到（或满足）的目标，用以表征机构成熟度或过程能力所达到的程度，所有这些目标的描述构成必需部件。例如，需求管理（REQM）的必需部件：

图表 22-1 需求管理必需部件

SG 1 Manage Requirements
GG 2 Institutionalize a Managed Process
GG 3 Institutionalize a Defined Process

每个过程域的目标又可分二类：

- 只适用于一个特定过程域的目标，称特定目标（Specific Goal），每 个过程域的特定目标的文字陈述前面（或后面）均有一个编号，SG1、SG2 等等。
- 当一个目标可适用于所有过程域时，称共性目标(Generic Goal)，整个模型，共有 5 个共性目标，编号分别为 GG1、GG2、…、GG5。

读者请注意：

- 特定目标的文字陈述十分简练，应下功夫理解它的真正含义，特别是要通过对该目标所含实践的了解来加深对目标的理解。
- 共性目标的文字陈述除了十分简练还十分抽象（因为它适用所有过程域）。 例如，GG2：

GG 2 Institutionalize a Managed Process

为了理解这个共性目标，首先要清楚：

- 这里的过程(域）(Process)是泛指的、抽象的。这个目标 GG2 出现在需求管理（REQM）中，则是指 REQM 过程域；同一目标 GG2 出现在需求开发（RD）中，则是指 RD 这个过程域。

- 制度化（Institutionalize）是什么意思？
- 一个过程域怎么样才算是受到管理的（Managed Process）？

CMMI 模型的 5 个共性目标，分别表示了机构整体开发能力的成熟度等级（在阶梯式表示法中），或者过程能力等级（在连续式表示法中）。第一个共性目标 GG1 表示达到全部特定目标。共性目标 GG2 表示过程（域）制度化到受管理的程度，如何才能达到或者怎样才算达到受管理的程度？GG2 所包含的 10 项实践正好就是为了达到受管理的程度而应执行的活动。共性目标 GG3 表示在达到共性目标 GG2 之后，应进一步制度化，使过程（域）成为已定义过程，而 GG3 所含的二个共性实践 GP3.1 和 GP3.2 则规定了为了达到已定义级的制度化程度应执行的活动。注意，GG2、GG3 是针对所有相关过程域而言的，但 GG4、GG5 就不一样了。共性目标 GG4 所规定的目标只针对某些过程或子过程，至于是哪些过程或子过程，则要根据机构的实际情况而精心选定，（不同机构因具体情况不同可以选定不同的子过程），然后对这些子过程进行定量管理，使其制度化程度达到定量管理级。最后一个共性目标 GG5 也一样，应根据实践情况，分析问题根源，然后持续优化相关的过程。

22.1.2 期望部件

为了达到过程域的全部目标，每个目标下面都会给出若干个为了达到该目标而应执行的活动，即实践（Practices），所有这些实践的陈述，构成了该过程域的期望部件。与二类目标相对应，也分二类实践：

特定实践（SP-Specific Practices），

共性实践（GP-Generic Practices）。

22.1.3 解释性部件

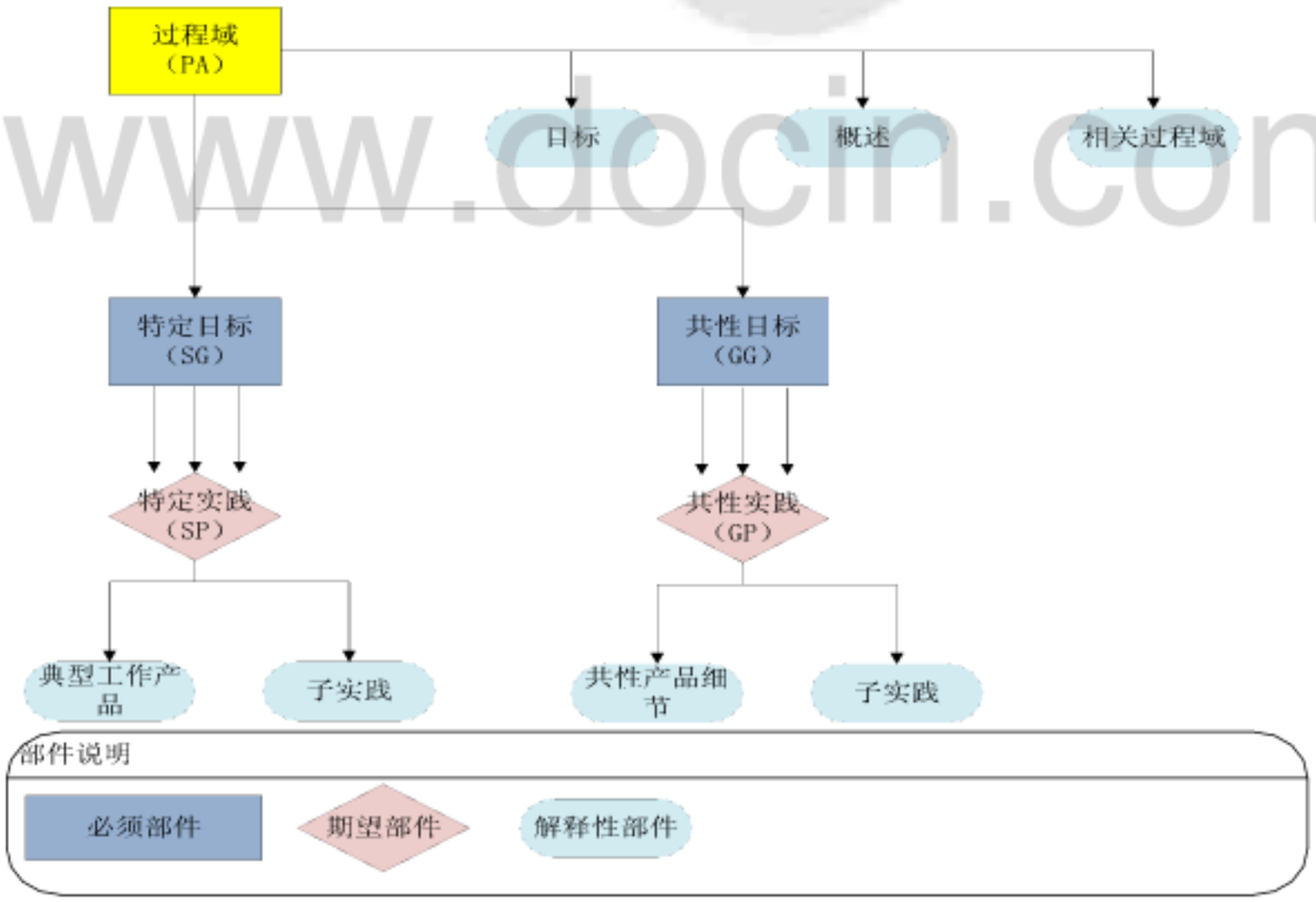
CMMI 每个学科模型的标准文本中，除了上面所讲的必需部件和期望部件之外，还有大量的解释性文字，这一部分内容统称解释性部件，细分为 10 种，列表说明如表 22-2 所示：

图表 22-2 解释性部件分类表

分类	说明
Purpose 目的	由一、二句话组成，用以概述该过程域的特定目标
Introductory Notes 概述	概述该过程域的范围、重要性、识别最佳实践的方式、专有名词以及与其他过程域的联系
Related Process Areas 相关过程域	针对某些主题，指明应参考其他过程域的相关内容
Names 名称	每个目标或实践均有一个简短的名称，以利于检索，紧接名称之后又有一句或几句进一步说明或描述
Practice－Goal Relationship Table 实践与目标联系表	概要列出实践与目标的对应关系，可以看作该过程域的摘要
Notes 注释	对目标、实践或子实践的补充说明
Typical Work Products 典型工作产品	执行某项实践时应输出的典型的工作产品，是验证执行效果的主要文件
Subpractices 子实践	某项实践的进一步分解或详细说明，但不属于期望部件
Discipline Amplifications 学科扩充	说明某些特定实践或目标适用于哪种学科
Elaborations 详细说明	针对某些共性实践的详细说明

有了前面三节的分类说明，最后可以给 CMMI1.2 版本模型部件的示意图（如图 22-3 所示）：

图表 22-3CMMI 1.2 版模型部件示意图



22.2 CMMI 共性实践详细说明[25]

CMMI 的每个过程域都包含了只适用于该过程域的一个或多个特定目标，每个特定目标都有若干个特定实践。如何理解、如何执行这些特定实践，将在后面各章中分别说明。

本节只对适用于各个过程域的共性实践作进一步说明。请读者注意，对共性实践的理解，很大程度上影响了 CMMI 模型的有效实施。可惜国内已出版的许多有关 CMMI 的著作或者各式各样的培训，重点往往只放在特定目标和特定实践上（这当然是重要的），而对共性实践的讲解则一带而过，结果是读了以后往往不知道如何实施，例如，不知道该从何入手、不知道该写些什么文档，等等。本节用较多的篇幅企图弥补这一点。

先看一下共性目标和共性实践的清单，见表 22-4。

图表 22-4 共性目标和共性实践列表

GG 1 Achieve Specific Goals		
	GP 1.1	Perform Specific Practices
GG 2 Institutionalize a Managed Process		
	GP 2.1	Establish an Organizational Policy
	GP 2.2	Plan the Process
	GP 2.3	Provide Resources
	GP 2.4	Assign Responsibility
	GP 2.5	Train People
	GP 2.6	Manage Configurations
	GP 2.7	Identify and Involve Relevant Stakeholders
	GP 2.8	Monitor and Control the Process
	GP 2.9	Objectively Evaluate Adherence
	GP 2.10	Review Status with Higher Level Management
GG 3 Institutionalize a Defined Process		
	GP 3.1	Establish a Defined Process
	GP 3.2	Collect Improvement Information
GG 4 Institutionalize a Quantitatively Managed Process		
	GP 4.1	Establish Quantitative Objectives for the Process
	GP 4.2	Stabilize Subprocess Performance
GG 5 Institutionalize an Optimizing Process		
	GP 5.1	Ensure Continuous Process Improvement
	GP 5.2	Correct Root Causes of Problems

下面从第 2 个共性目标开始，逐个说明每个共性实践。

GP2.1 建立机构方针

为了策划并实施该过程域所含的过程，机构应建立并维护（针对该过程域的）机构方针。

[25] 本节内容引自于胡希明老师的培训讲义。

此项共性实践的目的在于：确定机构对该过程域的期望和要求，并使机构内所有相关人员能明确获知此类期望和要求。机构方针应由机构的高层管理者负责制定并慎重颁布。当然，并不是所有来自高层管理者的信息都必需作为机构方针。总的来说，机构方针是公司高层管理者对过程目标的态度、意愿和承诺的文字表述，是对每一个该过程活动参与者的行为约束，必须遵循的、具有权威性的纪律。因此，机构方针的制定和发布，应满足如下需求：

- 每一条机构方针应力求做到必要、明确、权威和可实施。
- 机构方针应写成书面文档。
- 机构方针可由机构内部负责过程改进的专门小组（如工程过程组 EPG 或软件过程专家组）起草，既要体现高层管理者对过程管理的意图，又要征求部门和项目组员工的意见，通过评审，由机构最高负责人批准，正式发布。
- 机构方针的发布必须是充分的，在必要的范围内、在必需的保密前提下，所有过程活动参与者均可以通过各种方式随时取用。
- 机构方针应定期评审，持续改进，以确保反映变化了的机构内外环境或高层管理者的指令，同时确保符合 CMMI 标准的要求。
- 机构方针将随着成熟度等级的逐步提高或所选过程域能力的提高而分次颁布，逐步扩充、完善。

GP2.2 策划过程

制定并维护过程实施计划。制定计划，包括计划的文档化并提供过程描述；维护计划，是指必要时修改计划，以便使计划能适应过程需求和目标的变化，以及响应其他纠正措施引起的变更需求。

一个过程实施计划，应包括如下内容：

- 过程描述；
- 过程工作产品和服务的标准；
- 过程工作产品和服务的需求；
- 过程性能的特定目标（如质量、时间进度、需用资源等）；

- 过程活动、工作产品和服务之间的相互依赖关系；
- 实施该过程所必需的资源（包括资金、人员和工具）；
- 职责分配和授权；
- 执行和支持该过程所必需的培训；
- 应纳入配置管理的工作产品以及每个配置项的配置管理层次说明；
- 为了深入了解过程及其工作产品和服务的性能所必需的度量需求；
- 经标识的项目干系人应参与的活动；
- 过程监督和控制活动；
- 对过程及其工作产品的客观评价活动；
- 对过程及其工作产品的管理评审活动。

过程策划，作为一项共性实践，又可以划分为如下 5 个子实践（步骤）：

1. 获取高层管理者对过程实施的倡议，以便得到必要的支持和承诺。
2. 详细规定并文档化过程描述。过程描述应包括相关的标准和规程；可以作为过程实施计划的一部分，也可以作为计划的附件。
3. 确定并文档化过程实施计划。实施计划可以作为一个单独的文档，也可以纳入一个更全面的文档中，还可以分散在多个文档里。如果分散在多个文档中，则应确保在整个过程活动中谁做什么能保持清晰连贯的图景。
4. 与相关项目干系人一起评审实施计划并取得他们的同意。评审活动应包括评审经过策划的过程能满足机构方针、计划、需求和标准，以提高相关项目干系人的信心。
5. 必要时修订实施计划。

关于本项共性实践，还有几点注意事项，补充说明如下：

- 过程需求包括本过程的特定工作产品和可能来自其他过程需求的工作。
- 过程目标可能来自其他计划（如项目开发计划），也可能是其他特殊情况的要求，包括质量、成本和进度等。
- 此处所说的“策划过程”，概念上容易造成混淆。例如，在“项目计划”过程域中也

包含了共性实践 GP2.2，此时，GP2.2 所讲的“策划过程”与项目“策划过程”有什么不一样，表面上看似乎是“策划一个策划”的递归格局。其实，本质上二者不是一个概念。很简单，项目计划本身应该进行策划，谁负责以及如何估计规模、成本，谁负责制定进度表、谁负责编制项目计划，等等，本身就应先进行策划并进而制定计划，然后按照这个计划去进行“项目计划”过程域所规定的各项策划活动。再例如，某些过程域的某项特定实践也可能涉及制定一个计划，但这个计划往往是针对某项工作产品或某项活动而言的，显然，这个事情本身（由谁及如何制定这个计划等等）也应进行策划并作为文档化的过程实施计划的一项内容。

- 过程描述，也有一定的格式，一个有良好定义的过程，从过程描述上看，一般应包含如下信息：目的，输入，就绪准则，重要活动，角色分工，度量，验证，输出，结束准则。
- 过程实施计划，最好也有规范的格式，将在下一节结合具体的过程域再作介绍。

GP2.3 提供资源

为了实施过程域所含的过程，开发工作产品，提供服务，机构应确保提供足够的资源，包括资金、物理设备、具有一定技能的人员以及必要的工具。所谓“足够”，取决于许多因素并随时间而变化。如果没有足够的资源，应增加资源，或者减小一部分需求、去除某些约束条件以至改变某些承诺。

提供足够的资源，讲起来简单，实现起来很难。例如，人员不够但工期不能缩短，只好加班加点，而实际上往往是以牺牲过程规范性为代价，满足一时的商业目标，最终不得不支付无限制的交付后的维护成本。可能这不是某一个二个企业家的问题，而几乎是一个国家软件产业发展初期一代二代企业家应支付的成长的代价。当然，国家所能提供的产业环境也是问题的一个原因。

GP2.4 分配任务

为了实施过程域所含的过程，开发工作产品，提供服务，应明确划分职责并合理并明确授权。职责分配和授权可以是固定的方式写入相关文档，如写入过程实施计划；也可以是动

态的方式，按不同的角色进行分工。

分工涉及以下三个要点：

1. 总的职责划分和授权，主要指项目管理、过程管理和支持任务的分工和授权。
2. 具体任务的分工和授权。
3. 确认被分配任务和授权的员工理解并接受分工。

GP2.5 培训人员

应对执行或支持该过程的人员进行必要的培训，以便使相关人员能充分理解应执行或支持的过程及其活动、工作产品和目标，并赋于执行或支持过程所必需的知识和技能。培训可以采取多种方式，课堂讲授只是其中的一种方式。此处所讲的培训与“机构培训”过程域所讲的培训有所不同，项目组应充分利用机构级的培训。

GP2.6 管理配置

将该过程的指定工作产品纳入配置管理的相应层次。本项共性实践的目的在于：建立并维护该过程指定工作产品在其整个可用周期的完整性。指定工作产品应在过程实施计划中特别加以标识并同时给出配置管理层次的详细说明。

不同的工作产品或同一工作产品在不同时间点上，应有相应的不同配置管理层次。某些工作产品只要维持版本控制就足够了（例如，在过去或将来的某个给定时间使用的工作产品版本是知道的，并且版本的变更也以受控方式得到体现）。版本控制通常由工作产品的所有者负责控制（所有者可以是个人、小组或团队）。

有时候，某些工作产品必需纳入基线。此时涉及在预先定义的时间点上定义和建立基线。基线应经过正式评审并得到确认，作为指定工作产品下一步开发的基础。

除了以上二种层次外，还可能会有附加的配置管理层次。一个指定的工作产品在不同时间点上可能会纳入不同的配置管理层次。

注意，“配置管理”过程域的某些特定实践，将会为此处所讲的配置管理的实现提供支持。

GP2.7 识别相关的项目干系人并使其介入过程活动

本实践的目的是制定和维护项目干系人在过程执行期间介入过程活动的计划，以便使相关的项目干系人能按计划介入过程活动。应介入的活动包括：

- 策划，
- 决策，
- 沟通，
- 协调，
- 评审，
- 评价，
- 需求定义，
- 问题或争议问题解决方案。

策划项目干系人介入过程活动的目的在于：确保各方对过程的相互要求均能得到满足，但不要有太多的受影响的组或个人介入，以免妨碍过程的执行。

本实践可分成以下几个子实践：

1. 识别项目干系人对本过程的关系并决定其应介入的活动类型。相关的项目干系人可在下列人员中识别：输入提供者、输出的用户以及过程活动的执行者。相关的项目干系人一经确定，就应将其应介入过程活动的层次列入计划。
2. 必要时，项目计划或其他计划制定者可以共享上列识别结果（相互项目干系人名单及其应介入的活动层次）。
3. 使相关项目干系人按计划介入过程活动。

GP2.8 监督与控制过程

按照过程实施计划监督与控制过程，必要时采取纠正措施。

本实践的目的是对过程进行日常的监督与控制。应保持对过程的必要的可视性，以便及时采取纠正措施。过程的监督与控制涉及到对过程及其工作产品和服务的相应属性进行测量。更详细的说明，可参考“项目监督与控制”和“度量分析”二个过程域的相关内容。

本实践可划分成如下几个子实践：

1. 依据过程实施计划，度量过程及其工作产品和服务的实际性能。
2. 依据过程实施计划，评审过程进展（完成的程度、进度）和结果。
3. 与对本过程负有责任的中层管理者一起评审过程活动、状态和结果，并识别争议问题。评审目的在于向中层管理者提供对过程的必要的可视性。评审可以是定期的或事件驱动的。
4. 识别并评估相对过程实施计划的明显偏差所造成的影响。
5. 识别过程实施计划和过程执行中的问题。
6. 当过程的需求和目标不能满足时，当发现了争议问题时，当实际进展明显偏离过程实施计划时，应采取纠正措施。在采取任何一种纠正措施之前，应考虑连带的风险。可能的纠正措施包括：
   - 采取补救措施修复有缺陷的工作产品或服务。
   - 修改过程实施计划。
   - 调整资源，包括人员、工具或其他资源。
   - 协商修改已有的承诺。
   - （在有把握的情况下）修改过程应满足的需求和目标。
   - 终止项目（工作）。
7. 跟踪纠正措施直到关闭。

GP2.9 客观评价遵循度

客观地评价过程相对其过程描述、标准和规程的遵循程度，并处理不符合项。

本实践的目的是，为过程按计划执行并遵循其过程描述、标准和规程，提供可靠的保证。

通常的做法是，由不直接负责该过程管理或执行的人员负责评价遵循情况。许多情况下，可以由机构内部的人员负责评价，但这些人员应与项目或该过程没有直接关系；也可以由机构外部的人员负责。作为评价的结果，可为过程的遵循程度提供可靠的保证，即使过程处在受到种种压力的情况下执行（例如，项目工作落后于进度要求，超过预算等等）。

有关客观评价遵循性的更多信息，可参考“过程与产品质量保证”过程域。

GP2.10 高层管理者参与过程状态评审

高层管理者应参与评审过程活动、状态和结果，并解决争议问题。

本实践的目的是：为高层管理者提供对过程的必要的可视性。此处所说的高层管理者包括机构内部那些比直接负责管理该过程的管理者层次更高的人员，如总工程师，特别是那些制定机构方针和过程改进方向的管理者，但不包括那些负责对该过程进行日常监督和控制的人。不同层次的管理者对过程信息有不同的需要。此类评审有助于高层管理者对过程策划和实施情况作出正确的判断或决策。

高层评审可以定期也可以随事件驱动。

GP3.1 制定（项目）定义过程

制定和维护（项目）定义过程描述。

本实践目的是：制定和维护（项目）定义过程的详细描述，此定义过程是根据具体项目的需要从机构标准过程集 OSSP 中经裁剪而得到的。机构应有覆盖全部过程域的标准过程，并且还有裁剪指南，利用裁剪指南就可以从标准过程集中经裁剪而得到符合项目或部门需要的定义过程（此过程称项目定义过程）。通过这种方式得到的项目定义过程，减少了机构各个项目之间如何实施过程的差异性，并且可以有效共享机构过程资产、数据和经验。项目定义过程详细描述为过程及其活动、工作产品和服务的策划、实施提供了基础。详细说明可参考“机构过程定义”过程域。

分五个子实践：

1. 从机构标准过程集 OSSP 中选择覆盖该过程域并且最符合项目或部门需要的标准过程。
2. 依据机构裁剪指南，裁剪选出的过程，制定项目定义过程。
3. 确保机构过程的目标恰当地转换成项目定义过程的目标。
4. 项目定义过程和裁剪记录应文档化。
5. 必要时，评审项目定义过程详细描述。

GP3.2 收集改进信息

收集工作产品、度量、度量结果以及其他来自过程策划和过程实施的改进信息，以支持将来的使用和机构标准过程及机构过程资产的改进。

本实践的目的是：收集过程策划和过程实施的信息和各类人工制成品。通过这项实践的执行，将使这些信息和人工制品纳入机构过程资产，并且使其他策划和实施相同或相似过程的人员能共享机构过程资产库。这些信息和人工制成品应存入机构变量数据库和机构过程资产库。相关信息的例子，包括各类活动所付出的工作量、某些特殊活动中注入或消除的缺陷数以及获得的经验教训。详细说明可参考“机构过程定义”过程域。

子实践包括：

1. 将项目定义过程和产品的度量（数据）存入机构度量数据库。（OSSP 中应包括通用度量集的定义）
2. 将相关信息和制品的结论性文档存入机构过程资产库。
3. 将相关经验教训的文档存入机构过程资产库。
4. 提交机构过程资产改进建议。

GP4.1 制定过程定量目标

根据用户的需要和机构商业目标，针对（过程产品的）质量和过程性能，制定和维护过程定量目标。

本实践的目的是：确定过程的某些定量目标，并取得相关项目干系人的同意。这些定量目标可以通过产品质量、服务质量以及过程性能来表示。详细说明可参考“项目定量管理”过程域。定量目标可以只针对一个过程或者一组过程来定义，若是后者则应将定量目标分配到组内的每个相关过程。

过程定量目标作为一组准则，据以判断产品、服务和过程性能是否满足用户、最终用户、过程管理和过程实施的要求。过程定量目标应超过习惯上的最终产品的目标，应包括中间目标以便用以管理最后目标的实现。过程定量目标部分地反映了机构标准过程的性能范例。过程定量目标应设置量化值，当过程稳定运行并在预期范围内波动时此量化值很可能达到。

子实践包括：

1. 制定从属于（所有相关）过程的定量目标。

2. 将定量目标分配到每个相关的过程或其子过程。

【注意，过程定量目标的设定，应力求做到符合 SMART 标准：Specific（确定、明确）、Measurable（可度量）、Aggressive yet Achievable（有挑战性但能达到）、Result_oriented（面向结果，即有效果）、Time_bound（限定时间）】

GP4.2 稳定子过程性能

稳定一个或多个子过程的性能，确定该过程满足质量和过程性能定量目标的能力。

本实践的目的是，采用适当的统计学方法或其他定量技术，稳定项目定义过程所包含的一个或多个子过程的性能，这些子过程的性能对整个过程的整体性能具有关键性的影响。通过稳定选定的子过程的性能，以支持过程的预期能力满足已制定的质量和过程性能的定量目标。

一个稳定的子过程给出了过程偏差特殊原因的非显著性表示。一个稳定的子过程可以预期在上下限范围内运行，上下限则由子过程的预期边界决定。一个稳定的子过程的偏差由偶然因素造成，偏差幅度可大可小。

所选过程及其产品的度量应纳入机构度量数据库，以支持过程性能分析以及将来可能必需的基于事实的决策分析。

子实践有三个：

1. 统计管理一个或多个对该过程的整体性能有显著影响的子过程的性能。

2. 依据受统计管理的子过程性能，预测过程能达到已经制定的定量目标的能力。

3. 将所选过程的性能度量结果纳入机构过程性能基线。

GP5.1 确保过程持续改进

为了满足机构商业目标，应确保过程持续改进。

本实践的目的是，选择并统一部署过程改进和技术改进，为满足质量和过程性能的已定目标做出贡献。

过程优化的方式或程度（灵活的方式，或创新性的），取决于为了配合机构商业目标能有多少授权的员工可以参与过程改进。通过经验共享，可以提高机构对变化和机会的快速响

应能力。在持续改进过程中，过程改进是每个角色的固有责任。

子实践包括：

1. 制定和维护用以支持机构商业目标的过程改进定量目标。

过程改进定量目标可以只针对某个特定过程，也可以针对一组过程，如果是后者则应将目标分解到组内的每个相关的过程。

过程改进定量目标最初可能来自机构商业目标，也可能来自对过程能力的深入理解。过程改进定量目标可以作为一组准则，用以判断过程性能是否定量地改进了机构满足其商业目标的能力。为定量目标设定的值，一般应超过当前的过程性能，并且为了满足这些目标可能涉及局部的、渐进式的技术改进和全局性的、创新性的技术改进。目标应经常修订，以推动过程改进。（当目标已达到时，再设定一个新的目标值，这个新的值超过已达到的新的过程性能。）

过程改进定量目标可以与 GP4.1 共性实践所规定的过程定量目标一致，或进一步细化，只要它们能起到二个作用，即推动过程改进以及作为成功过程改进的准则。

2. 识别可以使过程性能得到可测量的改进的过程改进活动

过程改进活动包括二种方式，渐进式的改进和创新式的改进。通常总是尝试采用创新式的技术改进措施，并独立地加以计划、执行和管理，特别是先进行典型试验。

3. 根据量化的期望收益、成本和效果的估计以及过程性能的可量化改进，决定过程改进活动的推广策略和管理措施。

应定量估计改进活动的成本和收益，并测量实际的成本和收益。收益主要应从机构级的过程改进定量目标角度加以考虑。而改进活动则既针对机构标准过程集又针对项目定义过程。

有关过程改进推广活动的管理措施，包括：变更试验、实施必要的调整、处理潜在的和实际存在的对于推广活动的障碍、减小正在进行的工作的破坏作用，以及管理风险。

GP5.2 纠正问题的根本原因

识别并纠正出现在过程中的缺陷和其他问题的根本原因。

本实践的目的是，当发现问题或缺陷时，分析缺陷和其他问题的根本原因，纠正其根本原因，以防止这些缺陷和问题以后再出现。

本节最后，请读者注意，上列共性实践中有一部分共性实践与部分过程域有直接的联系，后者的有效实施为相关实践的实现提供了必要的支持。例如，GP2.6 管理配置项这个共性实践就与配置管理过程域有直接联系，后者为前者提供了支持。表 22-5 中给出了二者之间的联系：

图表 22-5 共性实践与过程域之间联系表

共性实践	过程域
GP2.6 管理配置	配置管理（CM）
GP2.7 确定相关的共利益者并使其介入	项目计划（PP）
GP2.8 监督和控制过程	项目监督与控制（PMC）
GP2.9 客观评价遵循情况	过程和产品质量保证（PPQA）
GP3.1 建立项目定义过程	机构过程定义（OPD）
GP3.2 收集改进信息	机构过程定义（OPD）

22.3 本书章节与 CMMI 映射

图表 22-6 教材章节与 CMMI 映射表

本书章节与 CMMI3 过程域的映射关系列表		
过程类别	本书章节	CMMI3 的 18 个过程域
过程管理 Process Management	第 20 章 软件开发过程管理	CMMI3：Organizational Process Definition CMMI3：Organizational Process Focus CMMI4：Organizational Process Performance
	第 4 章 项目评审管理	CMMI3：Verification CMMI3：Decision Analysis and Resolution
项目管理 Project	第 3 章 立项管理	
	第 17 章 项目总结	

Management	第 5 章 项目初步计划 第 7 章 项目估算及详细计划	CMMI2：Project Planning CMMI3：Integrated Project Management
	第 10 章 项目跟踪及控制	CMMI2：Project Monitoring and Control CMMI3：Integrated Project Management
	第 9 章 风险管理	CMMI3：Risk Management
工程过程 Project Development	第 6 章 需求开发及管理	CMMI2：Requirements Management CMMI3：Requirements Development CMMI3：Validation CMMI3：Verification
	第 11 章 系统设计	CMMI3：Technical Solutions CMMI3：Verification CMMI3：Decision Analysis and Resolution
	第 13 章 实现与测试	CMMI3：Technical Solution CMMI3：Product Integration CMMI3：Verification
	第 14 章 制定测试方案及编写测试用例 第 15 章 系统测试	CMMI3：Verification CMMI3：Validation
	第 16 章 客户验收	CMMI3：Product Integration CMMI3：Verification CMMI3：Validation
项目支撑 Project Support	第 18 章 过程及产品质量保证	CMMI2：Process and Product Quality Assurance
	第 8 章 软件配置管理	CMMI2：Configuration Management
	第 19 章 度量分析	CMMI2：Measurement and Analysis
	第 21 章 决策分析	CMMI3：Decision Analysis and Resolution