现代软件工程

中

基本方法篇

周之英 编著

科学出版社

现代软件工程（中）

基本方法篇

周之英 编著

科学出版社

2000

内 容 简 介

本书分上、中、下三册，每册独立成篇，上册为管理技术篇，中册为基本方法篇，下册为新技术篇。中册集中介绍了软件开发过程中最重要的两个阶段（需求分析阶段和设计阶段）中的主要软件工程方法，并讨论了各种不同类型方法的来历、特点、优缺点、目前的发展、应用状况和一些实例。读者可以从不同方法中了解软件工程方法的发展历程，从而灵活地选用适合特定要求的方法，甚至必要时能创造性地发展自己独特的软件工程方法。

中册可作为学习计算机软件工程和信息系统工程的大学生、研究生的教材或参考资料。对从事软件开发的技术人员来说，本书是提高软件开发技术水平的重要参考资料。同时，该书也可帮助软件工程管理人员提高技术能力。

图书在版编目（CIP）数据

现代软件工程（中）：基本方法篇/周之英编著．-北京：科学出版社．2000

ISBN 7-03-007704-0

Ⅰ．现… Ⅱ．周… Ⅲ．软件工程-基本知识 Ⅳ．TP311

中国版本图书馆 CIP 数据核字（1999）第 27360 号

科学出版社 出版

北京东黄城根北街 16 号

邮政编码：100717

新蕾印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

2000 年 1 月第 一 版 开本：787×1092 1/16

2000 年 1 月第一次印刷 印张：32

印数：1—3 000 字数：745 000

定价：42.00 元

（如有印装质量问题，我社负责调换〈环伟〉）

前　言

我们即将迎来的21世纪信息社会将高度地依赖于信息系统。事实上，在现代化的社会中已经很难想象没有“计算机”、没有“软件”会是怎样，面对着那无穷无尽的现实的和潜在的计算机应用需求，研究如何更快、更好、更多、更方便地开发出各种不同类型、不同目的的软件，这就是软件开发技术和软件工程技术所要解决的一个问题。

软件开发技术一直是软件工作者的主要研究方向。50多年来，随着计算机系统的发展，软件开发技术也发生着变化。软件工程首先是为了解决软件危机而提出的。其目的是用成功的、卓越的开发经验来指导，通过类似于工业化的管理，把一般程度的开发人员的水平提高到优秀水平。软件开发技术的巨大成就，已经使软件开发不再是少数逻辑天才或专家的专利，而是广大用户可以参与和直接开发自己的应用项目。因此，软件工程技术和设计方法将会受到更多人的关注。

90年代以来，软件工程不仅从方法论的角度为管理人员和开发人员提供可见的结构和有序的思考的方式，而且大量的成功软件总结出的设计经验，使软件开发人员在面对新的项目时，不必从头做起，而可以充分利用设计模式、框架、部件库等等。网络计算环境提供了经济全球化的新倾向，硬件技术以每6个月为周期的速度在发展，这些新的需求要求软件能具有充分的适应能力，去适应各种不同类型的连通和变动要求。

因此，老的软件工程教学体系已基本不能反映这些新技术和新需求的现状。从1996年我开始逐步对清华大学研究生校级学位课“软件工程技术与设计”的教学内容进行更新。教学的基本方针强调软件技术发展的变动性。我认为在急速变化的技术与社会环境中，无法想象还坚持不折不扣地照搬以前的成功经验会在新环境下继续成功，也无法想象不了解过去条件下的成功与失败，就可以迅速地创造出全新的方法与产品。因此，当前“软件工程技术与设计”的新的教学内容是以软件开发技术的发展史为纲，试图说明每种软件工程技术发展的原因、解决的问题及局限性，以使学生不仅学到技术知识，更强调能根据具体情况灵活应用，甚至创造性地推动技术的更进一步发展。目前“软件工程技术与设计”课程讲课大纲如下：

- 软件开发技术发展史
- 软件危机及风险研究的重要性
- 软件工程技术方法的基本原则
- 软件过程的CMM模型
- 软件过程改进的实例分析
- 软件过程改进现状
- 需求工程的意义及现状
- 重要的需求分析及规格说明技术（功能性为主，结构化方法）
- 重要的需求分析及规格说明技术（控制为主，状态机，Petri Net）
- 重要的需求分析及规格说明技术（形式方法）

- 软件体系结构研究的意义及现状
- 基本软件体系结构
- 软件设计方法的发展概述
- 面向对象设计方法概述
- Use Case OOD
- Design Pattern
- 软件过程标准化问题
- CORBA（软件部件接口标准）
- DCOM/COM（OLE）（软件部件接口标准）
- 软件度量
- 实例分析（实时系统的设计）
- 自选（必须与教师协商）

当然，现在学生上计算机课已经不满足于单单在课堂上学习理论。理论如果不与实际应用相结合，是无法深刻地理解和应用的，更难于发扬创造精神。但由于条件的限制，这一点还有待于今后的改进。

本书是我多年从事软件工程教学的总结。全书共分为五部分：第一部分是总论，涉及一些全局性的问题，如软件开发技术发展史、软件风险、软件工程技术的基本原则和软件生命期过程及改进问题。第二部分讨论与需求工程有关的各种问题。第三部分是与软件系统的设计与构造有关的问题。第四部分涉及分布系统，它对网络计算环境有特殊意义。第五部分是三个专题，其中有关标准和软件度量是更高层次上的问题。我们说，大部分软件技术和方法提供了软件的可见性，标准和度量提供了判断上的工具和准则，因此正文部分和专题部分是相辅相成的，但传统上更强调方法论。本书没有涉及编码和测试问题。这虽然也很重要，但前者有其他课程讨论，后者更需要针对性地讨论，所以暂时不纳入本书范围内。

本书分上、中、下三册出版。除了要全面学习软件工程的学生和从事软件工作的人员外，分册出版可以帮助他们（她们）更方便地找到最需要的内容。上册包括第一部分和三个专题。软件开发的管理层次的人员会对这些内容有兴趣。中册是需求工程（第二部分）和设计方法（第三部分的大部分内容）。软件系统的开发技术人员可以在其中找到灵感，获取进行软件需求分析和设计方面的可选用的方法。下册则是现代软件工程成就的集中体现。包括第三部分中代表了优秀的软件设计经验复用的重要途径——设计模式方面的内容和第四部分——分布系统方面的成果。对于在基本软件工程理论和方法方面有相当造诣，需要了解90年代以来最新发展的工程技术人员来说，一定会感兴趣。

选修本课程的有清华大学计算机系和其他系的近百名的研究生和进修教师，他们在学习本课程后收集了大量资料和报告。在本书的内容中，吸纳了其中一部分内容。没有他们的积极参与和共同讨论，本书是不可能现在就面世的。协助本书编写工作的还有清华大学计算机系软件教研组的谢若阳老师和计算机系的部分大学生。在此一并表示感谢。

由于目前国内有关软件工程技术与设计方面的资料比较缺乏，因此本课程新的教学内容大都参考国外的书籍和资料。当前的技术发展可说是日新月异，而教学任务要求尽快把教学内容整理出版，以供急需。因时间和水平所限，一定有许多不周到和不准确之

处，恳切希望读者提出批评和建议。

清华大学计算机科学与技术系　周之英

1999 年 4 月 5 日于清华园

目　录

第五章 需求工程

5.1 概 述

一、有关软件错误的一些事实

我们或许会产生这样的疑问：为什么要浪费时间来担心需求呢？为何不跳过这一步以便节省开销呢？通过下面的5点事实，自然会理解进行软件需求分析不仅是可能的而且也是值得的。

事实1 在软件生命周期中，一个错误发现得越晚，修复错误的费用越高

在20世纪70年代，GTE，TRW和IBM三家计算机公司对这个现象进行了独立的研究，最后它们都得出差不多同样的结果，如表5-1所示。

表5-1 生命周期中修复软件的相对费用

阶 段	相对修复费用
需求阶段	0.1～0.2
设计阶段	0.5
编码阶段	1
单元测试阶段	2
验收测试阶段	5
维护阶段	20

从表5-1可以看出，在需求阶段检查和修复一个错误所需的费用只有编码阶段的1/5到1/10，而在维护阶段做同样的工作所付出的代价却是编码阶段的20倍。这就意味着在需求阶段和维护阶段修复一个错误的比值可高达1∶200。对于这个结果有两种可能的解释：

- 如果我们认为绝大部分错误在它们产生之后马上就被检查出来，那么我们所能得出的唯一结论是检查和修复一个编码错误的费用要比检查和修复一个设计错误的费用高，而后者的费用又比消灭一个需求错误的费用高。
- 如果我们认为绝大部分错误是在它们产生之后很长时间才被检测出来，那么额外的费用不仅用在修正这个错误本身，而且还要用在改正这个错误对后续阶段的一系列负面影响上。

事实2 许多错误是潜伏的，并且在错误产生后很长一段时间才被检查出来

Boehm从TRW公司所做的软件项目中得出结论：所有被检测出来的错误中的54%实际上是在编码和单元测试阶段以后才被发现的；更糟糕的是，此类错误中的绝大部分

(占 45%)是属于需求和设计阶段的,而编码阶段的错误只占 9%。

事实 3 在需求过程中会产生很多错误

DeMarco 在一份研究报告中指出,被检查出来的错误的 56%产生的根源可以追溯到需求阶段。AIRMICS 所进行的一项调查发现,在一份美国军方大型管理信息系统的需求规格说明书(SRS)中存在着 500 多个错误,当然这仅仅是一个软件项目中的一次调查。

事实 4 在需求阶段,代表性的错误为疏忽、不一致和二义性

美国海军研究实验室从 20 世纪 70 年代起就对软件开发技术不断地进行研究。他们对海军 A-7E 飞机上的飞行操作程序进行实地测试,以验证许多新设想的可行性。得出的研究数据表明:A-7E 项目中 77%的需求错误特点是不明确 ---疏忽、不一致和二义性。按错误类型对这些错误分布进行分析的结果是:

49%不正确的事实

31%疏忽

13%不一致

5%二义性

2%放错位置

事实 5 需求错误是可以被检查出来的

让我们看以下三个独立的研究。

- Bruggere 认为"不应该浪费时间去分析软件中的不可执行部分(例如需求和设计),因为计算机在运行这些部分时会很容易地发现其中的错误"的说法是荒诞的,在软件中发现错误的最有效办法是检查它。有一份研究数据指出:

表 5-2 发现错误的比例

发现错误的方法	发现错误的比例(%)
检查	65
单元测试	10
集成测试	5
演进	6
其他	14

- Basili 和 Weiss 的数据表明:在 A-7E 的软件定义文档中,33%的需求错误是通过人工检查出来的。其中:

表 5-3 人工检查出的错误比例

类型	比例(%)
作者自查	23
非作者检查	10
维护参考	2
设计参考	45
编码参考	1
其他	19

• Celko 觉得利用自动分析工具(如 CADSAT,PSL/PSA 等)能够从 SRS 中检查出来相当数量的错误。表 5-4 指出三种自动分析工具检查出来的错误类型及数量:

表 5-4 自动分析工具检查出来的错误类型及数量

错误类型	REVS	IORL	CADSAT
不一致	101	143	115
二义性	70	126	
遗失	53		
无逻辑性	38		
不完整	26		52
存在问题	0		79
其他	14	273	4
总计	302	542	250

由上面这些事实,能得出如下四点结论:

• 在需求过程中会产生很多错误(事实 3 和 4)。
• 许多错误并没有在早期被发现(事实 2)。
• 这样的错误是能够在产生的初期被检查出来的(事实 5)。
• 如果没有及时检查出来这些错误,软件费用会直线上升(事实 1)。

需求错误会使最后交付的软件产品不能满足用户的需要,因为建立一个错误的系统不仅浪费时间,也浪费投资。对需求的不同理解会使用户和软件开发人员的意见不统一,浪费时间和金钱甚至会产生法律纠纷。1979 年美国政府的调查表明,投资于软件中的大部分钱都被浪费掉了。在 9 个被调查的软件项目中,只有不到 2%的投入得到预计的产出。进一步的调查发现,那 2%成功的原因在于所有参与者都很好地理解了项目需求的详细情况,并且这些需求在整个软件设计过程当中都没有改变。虽然这项调查已是许多年以前的事了,不过美国科罗拉多大学软件工程权威戴维斯却坚信当今世界软件行业依然存在着这种需求工作被轻视的现象。这必须改变。需求工程对后续开发工作所起的指导性作用以及对软件工程的最终交付使用所起的评价、审订、鉴定性作用,都使得它在整个软件工程中的地位日益突出,并受到越来越多的重视。需求工程保证"软件产品"恰如用户所期望得到的,从而使我们的工作价值得以完全体现;也就是说,高质量的需求工程是软件项目得以正确、高效完成的前提。这也是人们为摆脱"软件危机"而做出的明智选择。

二、什么是需求工程

1. 无法说明的问题是无法解决的

软件工程师所面临的要解决的问题常常极为复杂,理解问题的本质也很困难,特别当要求开发一个新的系统时更是如此。在正式采取行动之前,必须研究问题,解决系统应该提供什么服务,工作将受什么条件约束。需求工程就是确定系统"做什么"的问题,它并不

涉及系统“怎么做”(即如何实现)的问题。需求工程的成果是产生重要文档——“软件需求规格说明书”,作为软件开发的依据。需求工程主要分三个步骤:需求获取、需求分析和编写软件规格说明书(SRS)及验证。

2. 需求工程的有关概念

需求是什么?需求就是以一种清晰、简洁、一致且无二义性的方式,对一个待开发系统中各个有意义方面的陈述的一个集合。需求必须包含有足够多的信息,足以使设计师和工程师来产生一个使客户方和使用者都满意的产品,仅此而已。(尽管从定义上来看,很明显,需求必须包括有关系统必须完成的功能的描述,然而通常它还不得不包括更多额外的信息。)

需求工程一般指应用已证实有效的原理、方法,通过合适的工具和记号,系统地描述出待开发系统及其行为特征和相关约束;通常是一些过程的集合:需求获取(需求引出)、需求分析和编写软件规格说明书(SRS)及验证(包括鉴定和证实)。需求引出是一个确定客户或使用者的要求是什么的信息收集过程;规格说明,这个词既指这个过程也指这个过程的结果,即在“需求规格说明书”的文档中描述收集来的信息;鉴定,这个过程用来确信需求规格说明书中不包含任何不一致的条款;证实,这个过程用来确信需求规格说明书中描述的东西确实正是客户所期望的。需求工程的主要目的是给待开发系统提供一个清晰的、一致的、精确的并且无二义性的模型(model),通常以需求规格说明书的形式来定义待开发系统的所有外部特征。

模型是对现实系统的一种描述,是它的抽象和简化;模型必须反映出现实系统的本质和实际,由其所有有关元素组成,反映其内在联系。

需求工程过程涉及一个对待开发系统的需求的清晰理解,其中包括对系统要求实现的服务的理解、对系统的用户情况的理解、对系统环境以及相关约束的理解。需求工程过程还涉及对各种细节层次上的许多观点预见和关系获取、分析及解决的方法。

系统环境是指本系统以外的、但对本系统产生很大影响的一些因素。系统的输入来自于环境;系统的输出返回环境。

三、需求工程中涉及的角色

需求工程中涉及的角色主要有需求者、分析员和实现者。有时某一个(或一组)人会充当其中的多个角色,但无论从技术的角度还是管理的角度来看,角色之间的明确划分将有利于需求工程的进行。

需求者(或广义的用户):包括客户和使用者,以及需要或对系统起决定性作用的主管。

系统分析员:其工作是通过适当的引导、规格说明、鉴定和证实技术来开发一个需求者所要的对该系统的精确描述;系统分析员是完成需求分析的主体。

开发者:由设计人员、编程人员和项目管理者组成。一旦需求规格说明产生,由开发者来构造系统。

系统分析员又称需求工程师。他们是用户和程序设计人员的中介,负责沟通用户和开

发人员的认识和见解，起着桥梁的作用，如图 5-1 所示。

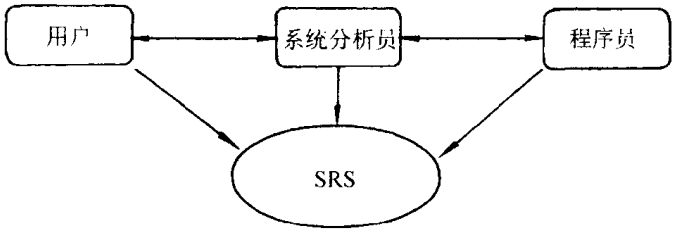

图 5-1　系统分析员的角色

系统分析员必须对产品的需求负责，应能把当今技术融合于应用问题之中，有能力既熟悉计算机信息处理思想，又了解应用业务领域的要求。系统分析员应该对产生的问题有全局把握的能力，集中精力于任务的关键部位。一个优秀的系统分析员必须能够深入地理解用户的环境，并且用简洁的语言完整地表述一个问题。

为了能胜任上述任务，分析员应具备如下几个方面的素质：

(1) 能力

系统分析员必须能够从冲突的原始材料中掌握事物的本质；能够深入地理解用户的环境，把各项需求组织起来，使最初的需求能在一个不断前进的基础上最终被实现。

系统分析员具有总体和局部的观念。

有抽象思维能力，善于由点到面考虑问题。不过早陷入细节，能找到主要问题。

需求工程的成果应该是一个精心构筑的框架。由于系统分析员熟悉计算机技术，所以在设计过程的开始就能够被使用，最终也能被用户所接受。

(2) 过程

系统分析员必须保证需求工程能顺利地进行。然而经常会由于需求过程进展缓慢导致软件产品被推迟交付；另一方面，在最初阶段或维护阶段拒绝对需求做任何重要的更改也会使得项目被迫中断。

系统分析员必须学会使得一个软件项目有始有终，既要满足用户的愿望，又要使开发人员理解他的作法。

(3) 交流

一个系统分析员必须综合考虑各种技术和非技术方面的意见。他或她也应该是一个优秀的协调员，善于表达思想，进行交流，把各种观点集中起来以便寻求一个最终解决的途径。

(4) 技术

一个成功的系统分析员应该有见地地了解软件项目的应用领域。他应该理解该领域的专门技术用语的含义。为了确保成功，这样的理解应该在 SRS 中得以很好地体现，并且用一种严格的或形式化的需求语言来表达。所采纳的语言既包括文本形式又不乏图形注释。

一个优秀的系统分析员应该博学多闻，受过严格的数学、科学和工程教育。不管怎样，一个系统分析员必须具备数学表达能力，这样才能把用户的需求翻译给软件开发人员。

四、需求分析与程序设计的不同

需求分析主要是解决软件产品应该达到的各项功能和非功能要求，即用户要求做什么。软件需求分析工作是软件开发人员与用户紧密配合、充分交换意见，系统在广大的相关人群中谋取平衡与折衷，最终达到互相谅解的过程。系统分析员是需求分析的主要角色。程序设计（即通常我们所说的编程）是软件设计的实现，是解决"怎么做"问题的具体方法中的实现步骤。程序员是程序设计的主要角色。二者的差别如表 5-5 所示。从表 5-5 可以看出，需求分析是十分复杂而困难的。

表 5-5　需求分析与程序设计的不同

程序设计	需求分析
工作单纯、明确	存在不确定性（有很多讨论的余地）
有确定的成功标准	无确定的成功标准（一般无最佳方案）
对自己工作自信心强	由于其复杂性，难于使各方满意
人与人关系简单	人与人关系复杂

五、需求工程的作用

1. 支持项目开发

需求工程过程是软件开发阶段的前提和基础。需求工程过程中产生的文档资料——软件需求规格说明书（software requirement specification）是软件开发的依据，也是软件开发者与用户评估产品软件质量的一种手段。软件开发人员根据需求工程的成果，确定并设计系统体系结构及各功能部件的性能、功能和接口。而且需求工程中所归纳的环境因素也会极大地影响各功能部件设计的复杂性。

高质量的需求工程能较好地刻画用户需求的各个细节特征，并产生出清晰、完备、精确且易于开发人员实现的需求规格说明书，引导开发人员正确地进行产品设计，全面地考虑各方面的影响因素（如系统环境因素的影响等），对开发进度、人员分配、软件成本等进行合理安排，从而提高工作效率和工作质量，同时减少了以往常常因误解或曲解而造成的反复修改。用户也能充分利用软件功能，使得软件产品的价值得以体现。

另一方面，在没有实现高质量需求工程的情况下，需求分析人员所产生的需求规格说明书往往不仅不能起到应有的指导作用，反而会因为其对需求描述的不彻底、含混或者二义性使开发人员对用户需求产生误解，或者由于它的不完备而导致系统缺乏对某种情况下的考虑，结果设计出的产品得不到用户的认可，前面的工作被否定，系统需要重新来做，这就造成了不必要的额外耗费。更糟糕的是，用户和工作小组可能开始互相埋怨，合作氛围变得紧张，工作效率下降，开发工作受到交付期限的威胁，开发人员负担加重。在这种情况下，姑且不说在软件开发成本上造成的额外负担，最终交付的软件产品虽然可能满足用户的基本需求，但其产品质量的保证却值得怀疑。

2. 支持测试和验证

需求规格说明书将为项目测试和验证提供基准，可以用来检查设计、验证系统。显然，一个有二义性的功能是无法测试或验证的。不论验证过程是采用测试或形式方法，最关键的对照物是需求规格说明书。

3. 支持维护

维护阶段的工作同以下几个方面紧密联系：

(1) 修改在测试阶段中尚未检查出来的少量残留编码错误。

(2) 软件运行一段时间后，因环境因素的改变而产生的软件的适应性维护。

(3) 用户在软件交付使用后又重新提出扩充功能的需求时的软件维护。

首先，我们可以看到，需求工程做得越彻底，质量越高，即充分考虑了软件的适应性问题和软件的功能扩展问题，在软件维护阶段的工作量就会大大减少，软件维护所需的人、财、物力和时间耗费也会大大降低。

其次，我们考虑在已出现因环境因素改变或因用户需求更新而引起的软件维护问题时，维护小组的工作很有可能是开始新的需求工程过程，因为只有在做好需求工程的基础上问题才会得以圆满解决。

4. 支持项目承包商

项目承包商必须采取一定步骤来确信他们正在构造的系统是正确无误的(build the right thing)，并且他们是在正确地构造这个系统(build the thing right)。需求的证实过程为拟构造系统的正确性提供了进一步的根据；而根据需求工程产生的基准(baseline)，可以检验系统是否被正确地构造。要拥有有效的鉴定和证实过程，清晰、一致和完备的需求很重要。

5. 支持管理

为了保证项目的顺利进行，项目管理者必须有一个完备的项目计划，包括开销、交付日期、可用资源、交付条件等等；而一个完整的需求规格说明将包含以上所有信息。这又一次清楚地说明：要使项目可管理，必须有完善的需求工程。

5.2 需求工程的内容

一、需求获取

1. 需求获取过程

制定软件的需求规格说明不只是软件开发人员的事，用户起着至关重要的作用。用户必须对软件功能和性能提出初步的要求，并澄清一些模糊概念。这就要求软件开发人员和用户要保持紧密的工作协作关系。软件分析人员在同用户交流的过程中收集各种用户的信息，认真理解用户的各项要求，包括该软件项目的功能性要求和非功能性要求。但是要

注意，用户的要求和用户的需求不是一回事。例如，一个组织可能决定要一个软件系统来支持财务处理工作，显然，这样一句话不足以让软件工程师去开发一个可以接受而且有用的软件系统。软件工程师必须收集解决此问题的一切有关信息，作为进行分析、产生一个全面解决方案的基础。

在进行需求获取时，包括要从各种客户（实际应用系统的直接领导者、行政上有批准权的领导；将要使用系统的具体操作者、使用者，包括负责新系统维护的技术人员等）获取信息和分析信息。

在把获取的信息进行分析时，分析员要对收集反馈回来的各种信息细致地进行调查分析。对于其中一些模糊的要求还需要向用户作进一步的澄清，然后才决定是否接纳。由于大部分用户不是计算机专家，所以所提出的要求从软件角度来看并非全部合理，对于那些不合理的部分或者目前暂时无法实现的要求应该向用户做充分的解释，以求得谅解。

在分析过程中，分析员会随时向有关方面专家请教，并且明确该解决方案的约束条件。在这个时候，有关此问题的各种信息和知识都会得到很大的扩展。

这期间的主要工作是：归纳和整理用户提出的各种问题和要求，弄清用户企图通过软件达到的目的，并把它作为要求和条件予以明确；即分析人员借助于各种工具和方法，获取对用户需求的基本理解，然后在需求获取方法的驱动和指导下，从非形式需求陈述中提取出用户的实际需求，并由此确定软件的功能、性能、接口关系及有关属性、软件条件、限制和边界等，标定软件的作用范围，确认支持性的软、硬件环境及辅助工具与条件。此阶段还为软件需求分析过程提供了相应的过程控制机制。

2. 需求获取方法

需求获取方法包括两个方面：

（1）指导开发小组获得用户需求的方法框架。

（2）支持控制此项活动进展的过程控制机制。

根据应用领域、用户性质和系统规模的不同，需求工程具体所采用的方法框架和过程控制机制也不完全一样，但基本的规律可以大致表示如下：

图 5-2　需求的方法框架

需求分析员应该牢记两个信条：第一，只有当你彻底弄懂了用户的全部意图之后，才有可能建立起成功的软件系统；第二，一切从用户的角度着想，在条件（人力、物力、财力和技术力量等）允许的情况下尽可能保证用户从所构造的软件系统中获得最大利益。

这样，对需求工程工作小组一般有较为严格的要求。工作组成员不仅需要具备高超的技术水平、丰富的实践经验，而且还必须具有较高的人际交往的能力；有强烈的事业心，能把自己的意志放在小组利益之下，坚决服从整个小组的决定。工作小组人数不应太多，且最好是单数，这样容易进行表决。整个工作小组应该与用户坦诚相待、融洽相处；必须设法

让用户意识到，这个系统的成败并不仅仅是分析员的事，也是用户自己的事，分析员、工程技术人员和用户的利益是紧密联系在一起的。

3. 当前状况

当前的软件系统，特别是大型系统的需求从根本上是难于建立的。具体情况如下：

(1) 误解。大多数情况下，分析员并非该应用领域的专家。许多在需求规格说明中出现的问题，可以一直追溯到需求获取时分析员和软件工程师误解了用户潜在的隐含假设。

(2) 交流障碍。需求分析员的作用是理解需求和表达需求。由于需求分析员在用户领域的知识远比领域专家少，那么交流问题就成为理解的一大障碍：领域专家同一个新手交谈时的用词往往并不足以完全解决问题。

(3) 缺乏共同语言。由于需求分析员和系统用户的经历和教育背景不同，他们之间通常缺乏足够的沟通。也就是说，他们之间缺乏共同语言。

(4) "完整性"问题。在需求定义中，"完整性"这个词是有问题的。我们没有一个简单的解析过程，可以用来决定何时用户已经把每一件为开发恰如需求的系统所必须知道的事都告诉给了开发人员。软件工程师当然希望提出系统需求的用户领域专家能清晰、简洁和完备地表述出确实可行的系统需求，以使得这种需求是方便实现的。然而，所要的需求知识在其最初阶段可能是模糊、不完备甚至不正确的，况且某些对系统有影响的实际需求知识可能往往只有通过一定的"引导"之后，用户才能提出来。

(5) 需求永远不会稳定。大型软件系统往往要求改变用户环境的现状。系统工程环境因素一旦改变，系统的操作环境就要随之改变，甚至用户本身也难于预料改变以后的系统会对应用有何影响。这种情况有时甚至在系统开始建立之前就会发生。用户对软件的需求也会改变，而且无法预测。这种需求的改变带来的影响随着改变发生的时间不同而不同：需求修改提出得越迟，系统为修改而耗费的成本越多，而且基本是呈指数量级上升。

(6) 用户意见不统一。大系统往往有各种不同类型的用户集体，他们往往有互相冲突的需求和不同的需求优先次序，寻求折衷是不容易的。

(7) 错误的要求。系统的定购者(付钱的人)和系统的用户经常不是一个人，定购者由于受组织或经费的限制，提出的需求会与最终用户的需求相冲突。

(8) 认识混淆。有时系统的目标与系统的需求会发生混淆。目标是系统应达到的更为一般的特征，而需求应是可测试的。例如，目标可以是"系统应做到用户界面友好"，而这是无法测试的；相应的需求可以写成"所有用户命令选择应采用命令菜单方式"。

4. 解决问题的建议

解决问题的最好办法是进行调查研究：要了解用户的意图，就要进行大量的调查研究，没有调查研究就没有发言权。调查研究是需求获取的基础工作之一，其中包括对应用系统的理解、与用户的交流和材料的收集等。

(1) 为了做好调查研究，一般应考虑这几个方面：

1) 了解系统需求。软件开发常常是作为系统开发的一部分，因此仔细研究系统分析的文档资料以了解系统的需求中对软件的需求是很有必要的。

2) 市场调查。了解市场上对开发软件的需求形势，掌握市场上流行的相关软件产品