R

UANJIAN GONGCHENG JICHU JIAOCHENG

软件工程基础教程

潘广贞 杨剑 王丽芳 武瑞娟 ◎ 编著

国防工业出版社
National Defense Industry Press

软件工程基础教程

潘广贞　杨剑　王丽芳　武瑞娟　编著

国防工业出版社

·北京·

内 容 简 介

本书主要介绍了软件工程概述、需求分析、概要设计、详细设计及实现、软件测试与维护、Rational Rose 建模工具、面向对象方法学基础、面向对象的分析、UML 基本元素符号、类图、对象图与包图、用例图、活动图、交互图、状态机图、软件项目管理、软件工程的最新发展等内容。

本书内容循序渐进、深入浅出、概念清晰、结构条理，将软件工程的理论知识与软件工程的应用实践相结合，并配有适量的习题，帮助读者从不同的角度理解和掌握所学的知识，构建完整的软件工程知识体系。

本书可作为高等院校计算机、软件工程、通信或电子类等相关专业的本科生或高职高专院校专科生的教材，也可作为工程技术人员及计算机爱好者的自学用书。

图书在版编目（CIP）数据

软件工程基础教程 / 潘广贞等编著. — 北京：国防工业出版社，2013.10

ISBN 978-7-118-09103-8

Ⅰ. ①软… Ⅱ. ①潘… Ⅲ. ①软件工程-教材

Ⅳ. ①TP311.5

中国版本图书馆 CIP 数据核字(2013)第 225703 号

※

国防工业出版社出版发行

（北京市海淀区紫竹院南路 23 号 邮政编码 100048）

北京奥鑫印刷厂印刷

新华书店经售

*

开本 787×1092 1/16 **印张** 18½ **字数** 454 千字

2013 年 10 月第 1 版第 1 次印刷 **印数** 1—4000 册 **定价** 34.00 元

前　言

随着“十二五”规划纲要的颁布，我国软件和信息技术服务业在创新的驱动下进入了一个全新的发展阶段。新信息技术成为展示创新能力的主要平台之一，具有创新能力的软件工程人才将推动该领域技术模式、服务模式以及应用模式的全面转型，支持软件和信息技术服务业持续发展。软件工程作为指导计算机软件开发和维护的一门工程学科，自 1968 年被提出以来，经过四十多年的发展，已从面向机器、面向语言和面向中间件的形态，转为面向需求、面向服务和面向网络的形态，真正实现软件即服务。

在 20 世纪 90 年代初，中国计算机学会教育委员会及全国高等学校计算机教育研究会就把“软件工程”列为计算机学科中的专业必修课和主干课。现在各高校的软件工程、计算机科学与技术、网络工程、物联网工程等相关专业都将“软件工程”设为专业必修课。“软件工程”课程是一门综合性、实践性较强的课程，对培养学生的工程思维能力、实践能力、创新意识及沟通技能、团队合作精神等综合应用能力和素质起到重要作用，对培养能够应用和开发计算机软件系统，解决专业领域中实际问题的复合型人才起着不可替代的作用。

本书主要围绕传统的软件工程方法和面向对象软件工程方法两条主线组织内容，先介绍传统软件工程方法各阶段的过程、方法和工具，让学生对软件工程的实施有一个完整、清晰的认识；然后介绍面向对象软件工程方法的开发过程。这样可以使学生通过对两种开发方法的比较，理解两者的异同，理解相同的软件工程过程结合不同的软件工程方法，对软件分析、实现和维护的影响，以及对软件质量和管理发展的推动。

本书共 16 章，其中第 1 章 ~ 第 5 章主要介绍传统的软件工程方法，包括软件工程概述、需求分析、概要设计、详细设计及实现、测试与维护；第 6 章 ~ 第 14 章主要介绍面向对象软件工程方法，包括 Rational Rose 建模工具、面向对象方法学基础、面向对象的分析、UML 基本元素符号、类图、对象图与包图、用例图、活动图、交互图、状态机图；第 15 章介绍软件项目的管理；第 16 章介绍软件工程的最新发展。

由于本书的内容较多，各高校任课教师可以根据本校的实际要求，对授课内容作适当删减。本书文字通俗易懂、概念清晰、实用性强，并配有相应的习题，可作为各高等学校计算机及相关专业“软件工程”课程的教材或参考书，也可作为各类计算机爱好者的自学用书。

本书第 5 章、第 11 章由潘广贞编写，第 3 章、第 4 章、第 8 章、第 9 章、第 14 章、第 16 章由杨剑编写，第 1 章、第 2 章、第 6 章、第 7 章由王丽芳编写，第 10 章、第 12 章、第 13 章、第 15 章由武瑞娟编写。全书由潘广贞、杨剑统稿。本书的编写参考了大量的相关书籍和资料（详见参考文献），在此向这些参考文献和书籍的作者表示感谢。

本书在编写过程中得到了单位领导、同事的大力帮助和热情支持，他们对本书内容的修订提出了很多宝贵的意见，在此向他们一并表示感谢。

由于编者水平有限，书中难免存在疏漏之处，欢迎广大读者和同行专家，特别是使用本书的教师和学生批评指正。

编　者

2013 年 4 月

目　录

第1章　软件工程概述

1969年，IBM公司首次宣布除操作系统继续随计算机配送外，其余软件一律作为独立商品计价出售，从此开创了软件成为商品的先河。短短几十年，从PC到笔记本电脑，从互联网到移动通信设备，从军事到人们的日常生活，计算机软件几乎无处不在、无时不在，计算机软件的重要性无可替代。现在，软件产业已成为很多发达国家的支柱产业，软件工程师已成为受人青睐的职业。

本章主要介绍软件和软工程的相关概念及软件过程的相关知识。

1.1　软件与软件危机

和计算机硬件一样，从20世纪60年代以来，软件从规模、功能等方面得到了很大的发展，同时人们对软件质量的要求也越来越高。然而，软件的规模越大、越复杂，人们的软件开发能力越显力不从心，以致软件成本失去控制，软件质量难以保证。为了扭转软件开发的这种被动局面，人们逐渐开始重视对软件的开发、工具和环境的研究。

1.1.1　软件的定义、特点及分类

1. 软件的定义

计算机系统由硬件和软件组成。硬件是躯体，软件是灵魂。硬件在软件的支持和管理下，才能完成操作。因此，软件的发展与硬件的发展是相互联系的。在现代社会中，软件应用于多个方面。典型的软件有电子邮件、嵌入式系统、人机界面、办公套件、操作系统、编译器、数据库、游戏等。同时，各个行业几乎都有计算机软件的应用，如工业、农业、银行、航空、政府部门等。这些应用促进了经济和社会的发展，提高人们的工作效率，同时提升了生活质量。

在20世纪50年代，人们认为软件就是程序。20世纪60年代，人们认为软件 = 程序 + 文档。到了20世纪70年代，人们重新定义了软件的概念，认为软件应包含：

（1）一个或多个程序。程序执行时能够提供软件所期望的功能和性能。

（2）一个或多个数据。满足程序开发及运行所需要的任何数据，包括初始化数据、测试数据、研发数据、运行数据、维护数据、工程数据以及项目管理数据等。

（3）一个或多个文档。服务程序开发、测试、安装、运行、使用及维护等涵盖程序生命周期各阶段的细节描述，根据中国国家标准局颁布的《计算机软件开发规范》和《软件产品开发文件编制指南》，软件文档应包括可行性研究报告、项目开发计划、软件需求规格说明、数据需求规格说明、概要设计规格说明、详细设计规格说明、用户手册、操作手册、测试计划、测试分析报告、开发进度月报、项目开发总结报告、维护修改建议共计13种文档。

近年来随着软件在人类生活中的广泛应用，其规模与难度已经逐渐增大，上述定义已经不

能够完全表达软件的内容。1984 年,美国开始将软件定义为一个过程,该过程包括人、方法、规程、技术与工具。编制软件就是人们使用相应的方法、规程、技术、工具等原始材料转化为用户所需要的产品的活动。

2. 软件的特点

计算机系统中的软件与硬件是相互依存的,缺一不可。而软件与其他产品不同,它是一种特殊的产品,具有下列特点:

(1) 软件是一种逻辑产品。它是脑力劳动的结果,与物理产品有很大区别。软件产品是看不到、摸不着的,它以程序和文档的形式出现,保存在存储介质上,只有通过计算机的运行才能体现它的功能和作用。

(2) 软件产品的生产主要是开发。软件的开发是开发人员智慧的结晶,与硬件制造不同。硬件初期生产出试验产品,通过质量检验合格后,批量生产需要投入大量的人力、物力和财力。而软件一旦研发成功,批量生产就是进行简单的复制工作。对软件的质量控制,重在软件的研发过程中。

(3) 软件的开发和运行常受到计算机系统的限制。软件对计算机系统有不同程度的依赖性,软件不能脱离硬件而单独运行,在软件的开发和运行中必须依赖硬件提供的条件。

(4) 软件的成本昂贵。软件开发需要投入大量、高强度的脑力劳动,成本很高,风险也很大。目前软件开发的成本已远远高于硬件的成本。

(5) 软件开发尚未完全摆脱手工生产方式。虽然软件复用、构件技术已被提出,但像硬件那样基于已有零部件进行组装,实现软件生产的自动化仍有一定的难度。

(6) 软件不存在磨损和老化问题,但存在退化问题。软件不会因为磨损而老化,但软件为了适应运行环境及需求的变化需要进行维护,在维护的过程中不可避免地引入错误,导致软件失效率升高,从而出现软件退化。

3. 软件的分类

1) 按照软件的功能划分

按照软件的功能,软件可分为系统软件、支撑软件和应用软件。

(1) 系统软件:与计算机硬件紧密配合在一起,使计算机系统中的各个部件、相关的软件和数据协调、高效地工作的软件。如操作系统、数据库管理系统、设备驱动程序以及通信处理程序等。

(2) 支撑软件:协助用户开发软件的工具性软件,其中包括帮助程序人员开发软件产品的工具,以及帮助管理人员控制开发进程的工具。如编辑程序、程序库、图形软件包等。

(3) 应用软件:在特定领域内开发、为特定目的服务的一类软件,其中包括为特定目的进行的数据采集、加工、存储和分析服务的资源管理软件。如工程与科学计算软件、CAD/CAM 软件、CAI 软件、信息管理系统等。

2) 按照软件规模划分

按照软件的大小(源代码行)、开发软件所需人数和研制时间,可将软件分为微型、小型、中型、大型、甚大型和极大型 6 种。随着软件产品规模的不断增大,类别的参数指标也会发生变化,如表 1 - 1 所列。

表 1-1　软件的分类

类别	参加人数	研制期限	软件规模(源代码行数)
微型	1	1~4 周	0.5k
小型	1	1~6 月	1k~2k
中型	2~5	1~2 年	5k~10k
大型	5~20	2~3 年	5k~500k
甚大型	100~1000	4~5 年	1M
极大型	2000~5000	5~10 年	1M~10M

3) 按照软件工作方式划分

按软件的工作方式,可将软件分为实时处理软件、交互式软件、批处理软件和分时软件。

(1) 实时处理软件:在事件或数据产生时,立即予以处理,并及时反馈信息,控制需要监测控制过程的软件,主要包括数据采集、分析、输出三部分。

(2) 交互式软件:能实现人机通信的软件。

(3) 批处理软件:把一组输入作业或一批数据以成批处理的方式一次运行,按顺序逐个处理完的软件。

(4) 分时软件:允许多个联机用户同时使用计算机的软件。

4) 按照软件服务对象的范围划分

按软件的服务对象的范围,可以将软件分为项目软件和产品软件。

(1) 项目软件:也称定制软件,是受某个特定客户(或少数客户)的委托,由一个或多个软件开发机构在合同的约束下开发出来的软件。如军用防空指挥系统、卫星控制系统等。

(2) 产品软件:是由软件开发机构开发出来直接提供给市场,或为成百上千个用户服务的软件。如文字处理软件、财务处理软件、人事管理软件等。

1.1.2　软件开发的演变过程

软件是由计算机程序和程序设计的概念发展演化而来的,是在程序和程序设计发展到一定规模并且逐步商品化的过程中形成的。软件开发经历了程序设计阶段、软件设计阶段和软件工程阶段的演变过程。

1. 程序设计阶段

程序设计阶段出现在 1946—1955 年。此阶段的特点是:尚无软件的概念,程序设计主要围绕硬件进行开发,规模很小,工具简单,开发者和用户无明确分工,程序设计追求节省空间和编程技巧,除程序清单外无文档资料,主要用于科学计算。

2. 软件设计阶段

软件设计阶段出现在 1956—1970 年。此阶段的特点是:硬件环境相对稳定,出现了"软件作坊"的开发组织形式。开始广泛使用产品软件(可购买),从而建立了软件的概念。随着计算机技术的发展和计算机应用的日益普及,软件系统的规模越来越庞大,高级编程语言层出不穷,应用领域不断拓宽,开发者和用户有了明确的分工,社会对软件的需求量剧增。但软件开发技术没有重大突破,软件产品的质量不高,生产效率低下,从而导致了"软件危机"的

产生。

3. 软件工程阶段

自1970年起,软件开发进入了软件工程阶段。“软件危机”的产生,迫使人们不得不研究、改变软件开发的技术手段和管理方法。从此软件开发进入了软件工程时代。此阶段的特点是:硬件已向巨型化、微型化、网络化和智能化四个方向发展,数据库技术已成熟并广泛应用,第三代、第四代语言出现。

1.1.3 软件危机

20世纪中期,计算机刚被从军用领域转向民用领域使用,那时编写程序的工作被视同为艺术家的创作。当时的计算机硬件非常昂贵,编程人员追求的是如何在有限的处理器能力和存储器空间约束下,编写出执行速度快、体积小的程序。程序中充满了各种各样让人迷惑的技巧。这时的软件生产非常依赖于开发人员的聪明才智。到了20世纪60年代,计算机的应用范围得到较大扩展,对软件系统的需求和软件自身的复杂度急剧上升,传统的开发方法无法适应用户在质量、效率等方面对软件的需求。这就是所谓的“软件危机”。

最为突出的例子是美国IBM公司于1963—1966年开发的IBM/360系列机的操作系统。该软件系统花了大约5000人一年的工作量,最多时有1000人投入开发工作,写出近100万行的源程序。尽管投入了这么多的人力和物力,得到的结果却极其糟糕。据统计,这个操作系统每次发行的新版本都是从前一版本中找出1000个程序错误而修正的结果。可想而知,这样的软件质量糟糕到了什么地步。难怪该项目的负责人F·D·希罗克斯在总结该项目时无比沉痛地说:“……正像一只逃亡的野兽落到泥潭中做垂死挣扎,越是挣扎,陷得越深,最后无法逃脱灭顶的灾难,……程序设计工作正像这样一个泥潭……一批批程序员被迫在泥潭中拼命挣扎,……谁也没有料到问题竟会陷入这样的困境……”IBM/360操作系统的历史教训已成为软件开发项目中的典型事例被记入历史史册。

另一个例子发生在1963年,美国研制了一枚发射火星探测器的火箭,由于程序员将程序中的一个“,”误写为“.”,导致火箭升空爆炸,造成高达数亿美元的损失。

软件危机包含两方面的问题:如何开发软件以满足不断增长、日趋复杂的用户需求;如何维护数量及规模不断膨胀的已有软件。

1. 软件危机的表现

软件危机主要有以下几方面的表现:

1)软件成本日益增长

在计算机发展的早期,大型计算机系统主要是被设计应用于非常狭窄的军事领域。在这个时期,研制计算机的费用主要由国家财政提供,研制者很少考虑到研制代价问题。随着计算机市场化和民用化的发展,代价和成本就成为投资者考虑的最重要的问题之一。20世纪50年代,软件成本在整个计算机系统成本中所占的比例为10%~20%。但随着软件产业的发展,软件成本日益增长。相反,计算机硬件随着技术的进步、生产规模的扩大,价格却不断下降。这样一来,软件成本在计算机系统中所占的比例越来越大。到20世纪60年代中期,软件成本在计算机系统中所占的比例已经增长到50%左右。而且,该数字还在不断地递增,下面是一组来自美国空军计算机系统的数据:1955年,软件费用约占总费用的18%,1970年达到60%,1975年达到72%,1980年达到80%,1985年达到85%左右。

2）开发进度难以控制

由于软件是逻辑、智力产品，软件的开发需建立庞大的逻辑体系，这是与其他产品的生产不一样的。例如，工厂里要生产某种机器，在时间紧的情况下可以要工人加班或者实行“三班倒”，而这些方法都不能用在软件开发上。

在软件开发过程中，用户需求变化等各种意想不到的情况层出不穷，令软件开发过程很难保证按预定的计划实现，给项目计划和论证工作带来了很大的困难。

Brook 曾经提出：“在已拖延的软件项目上，增加人力只会使其更难按期完成。”事实上，软件系统的结构很复杂，各部分附加联系极大，盲目增加软件开发人员并不能成比例地提高软件开发能力。相反，随着人员数量的增加，人员的组织、协调、通信、培训和管理等方面的问题将更为严重。

3）软件产品质量差

由于缺乏工程化思想的指导，程序员几乎总是习惯性地以自己的想法去代替用户对软件的需求，软件设计带有随意性，很多功能只是程序员的“一厢情愿”而已，导致软件的可靠性差、故障率高，不能满足用户的要求。

4）软件维护困难

正式投入使用的软件，总是存在着一定数量的错误，在不同的运行条件下，软件就会出现故障，需要对软件进行维护。但是，由于在软件设计和开发过程中，没有统一的规范可以遵循，软件开发的随意性很大，没有完整的记录文档，给软件维护造成了巨大的困难。特别是在软件使用过程中，原来的开发人员可能因各种原因已经离开原来的开发组织，使得软件几乎不可维护。另外，软件修改是一项很“危险”的工作，对一个复杂的逻辑过程，哪怕做一项微小的改动，都可能引入新的问题。有资料表明，工业届为维护软件支付的费用占全部硬件和软件费用的 40% ~75% 。

5）软件产品开发的效率跟不上计算机硬件的发展

缺乏自动化的软件开发技术，软件生产率赶不上硬件的发展速度，更赶不上计算机应用的发展速度。软件产品进展缓慢使人们不能充分利用现代计算机硬件提供的巨大潜力。

2. 软件危机产生的原因

为了克服软件危机，需要分析导致软件危机的原因。通过软件危机的种种表现和软件作为逻辑、智力产品的特殊性可以发现，软件危机产生的原因主要有以下几方面：

1）用户需求不明确

在软件开发过程中，用户需求不明确问题主要体现在四个方面：

（1）在软件开发出来之前，用户自己也不清楚软件的具体需求。

（2）用户对软件需求的描述不精确，可能有遗漏、有二义性，甚至有错误。

（3）在软件开发过程中，用户还提出修改软件功能、界面、支撑环境等方面的要求。

（4）软件开发人员对用户需求的理解与用户本来的愿望有差异。

2）缺乏正确的理论指导

缺乏有力的方法学和工具方面的支持。由于软件不同于大多数其他工业产品，其开发过程是复杂的逻辑思维过程，其产品极大程度地依赖于开发人员高度的智力投入。由于过分地依靠程序设计人员在软件开发过程中的技巧和创造性，加剧软件产品的个性化，也是发生软件危机的一个重要原因。

3）软件的规模越来越大

随着软件应用范围的增加，软件规模越来越大。大型软件项目需要组织一定的人力共同完成，而多数管理人员缺乏开发大型软件系统的经验，而多数软件开发人员又缺乏管理方面的经验。各类人员的信息交流不及时、不准确，有时还会产生误解。软件项目开发人员不能有效地、独立自主地处理大型软件的全部关系和各个分支，因此容易产生疏漏和错误。

4）软件复杂度越来越高

软件不仅仅是在规模上快速地发展扩大，而且其复杂性也急剧地增加。软件产品的特殊性和人类智力的局限性，导致人们无力处理复杂问题。

3. 解决软件危机的途径

人们在认真地研究和分析了软件危机背后的真正原因之后，得出了“人们面临的不光是技术问题，更重要的是管理问题，管理不善必然导致失败”的结论，便开始探索用工程的方法进行软件生产的可能性，即用现代工程的概念、原理、技术和方法进行计算机软件的开发、管理和维护，于是诞生了计算机科学技术的一个新领域——软件工程。

1.2 软件工程

为了跟上硬件的发展速度、解决软件开发的进度，并实现成本可控制、质量可保证、系统可维护、过程可管理，进而缓解软件危机，1968 年 NATO 会议上首次提出“软件工程”（Soft wrae Engineeirng）的概念，提出把软件开发从“艺术”和“个体行为”向“工程”和“群体协同工作”转化。其基本思想是应用计算机科学理论和技术以及工程管理原则和方法，按照预算和进度，实现满足用户要求的软件产品的定义、开发、发布和维护的工程。

1.2.1 软件工程的定义

目前关于软件工程的定义一直都没有统一的说法，很多学者、组织机构都分别给出了自己的定义：

1. Barry Boedhm

运用现代科学技术知识来设计并构造计算机程序以及为开发、运行和维护这些程序所必需的相关文件资料。

2. IEEE 在软件工程术语汇编中的定义

软件工程包含两方面的含义：

（1）将系统化的、严格约束的、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件。

（2）上述方法的研究。

3. Fritz Bauer 在 NATO 会议上给出的定义

建立并使用完善的工程化原则，以较经济的手段获得能在实际机器上有效运行的可靠软件的一系列方法。

4.《计算机科学技术百科全书》中的定义

软件工程是应用计算机科学、数学及管理科学等原理，开发软件的过程。软件工程借鉴传统工程的原则、方法，以提高质量、降低成本。其中，计算机科学、数学用于构建模型与算法，工程科学用于制订规范、设计范型（paradigm）、评估成本及确定权衡，管理科学用于计划、资源、

质量、成本等管理。

5. CC2004 报告的定义

以系统的、学科的、定量的途径，把工程应用于软件的开发、运营和维护；同时，开展对上述过程中各种方法和途径的研究。该定义中明确提出了“把工程应用于软件”，突出了软件工程领域内的两类重要的研究和应用方向——工程学和方法学。

1.2.2 软件工程的基本原理

自“软件工程”这一术语提出以来，专家们对软件工程提出了 100 多条准则，著名软件工程专家 B. W. Boedhm（图 1－1）综合了这些专家的意见，并总结了多年来开发软件的经验，于 1983 年在一篇论文中提出了软件工程的七条基本原理，作为保证软件产品质量和开发效率的最小集合。具体如下：

1）用分阶段的生命周期计划严格管理软件工程过程

在软件开发与维护的漫长生命周期中，应该把软件生命周期划分为若干个阶段，并制订出相应的切实可行的计划，然后严格按照计划对软件的开发和维护进行管理。

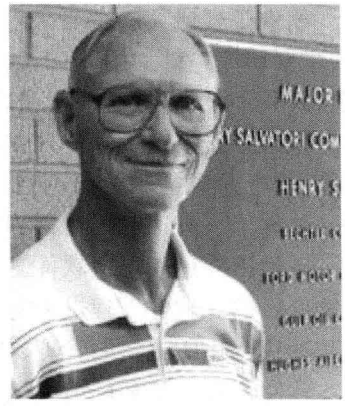

图 1－1　B. W. Boedhm

2）坚持进行软件过程中的阶段评审

根据 B. W. Boedhm 的统计研究表明，设计错误占软件错误的 63%，编程错误仅占 37%，软件的大部分错误是在编程之前造成的。软件中的错误被发现和纠正得越晚，所付出的代价就越高。软件的质量保证渗透在软件开发的各个阶段，因此，每个阶段都要进行严格的评审，尽早地发现软件中的错误，通过对软件质量实施过程评审，以确保软件在每个阶段都能够具有较高的质量。

3）实行严格的产品控制

在软件开发过程中，需求的变更是不可避免的，必须使用科学的产品控制技术，其中主要是实行基准配置管理，来保证软件的一致性。基准配置是指各个阶段产生的经过阶段评审的文档或程序代码。基准配置管理是指一切有关修改软件的建议，特别是涉及对基准配置的修改，都必须按照严格的产品控制来进行，使软件产品的各项配置成分保持一致，以适应软件需求的变更。

4）采用现代开发技术进行软件的设计与开发

从结构化软件开发技术到面向对象的软件开发技术，在长期的软件开发实践中，人们充分认识到：采用先进的软件开发技术可以提高软件产品的质量，降低开发成本，增加软件的使用寿命，提高软件的开发效率。

5）应能够清楚地审查开发过程中每个工作结果

软件产品是一种看不见、摸不着的逻辑产品，软件开发人员的工作进展情况可见性差，工作进展难以准确度量和控制，使得软件产品的开发过程难以评价和管理，为了对软件开发过程进行更好的管理，项目组应根据软件开发的总目标和完成期限，制订开发小组的责任和产品标准，使所得到的结果能够被清楚地审查。

6）开发小组的成员应少而精

开发人员的素质和质量是影响软件质量和开发效率的重要因素，高素质人员的开发效率