软 件 工 程 技 术 丛 书

HZ BOOKS

质量管理系列

PEARSON Addison Wesley

软件质量工程的度量与模型

Metrics and Models in Software Quality Engineering

（美） Stephen H. Kan 著

王振宇 陈利 余扬 等译

机械工业出版社
China Machine Press

软 件 工 程 技 术 丛 书

质量管理系列

软件质量工程的度量与模型

Metrics and Models in Software Quality Engineering

（美） Stephen H. Kan 著

王振宇 陈利 余扬 等译

本书详细讲述了软件质量工程中的基本问题和技术，除软件度量、软件可靠性模型和程序复杂性的模型和分析外，还讨论了过程中度量、缺陷排除有效性、顾客满意度等问题。理论、技术和实例的结合是本书的显著特点，书中有许多来自 IBM、摩托罗拉、惠普和 NASA 软件工程实验室的例子。通过这些实例，读者可进一步了解如何把书中所讲的理论和技术用于实际工作中，以测量和改进整个软件开发过程的质量。

本书版权登记号：图字：01-2002-5448

图书在版编目(CIP)数据

软件质量工程的度量与模型/(美)凯(Kan, S. H.)著；王振宇等译. -北京：机械工业出版社，2003.10

(软件工程技术丛书　质量管理系列)

书名原文：Metrics and Models in Software Quality Engineering

ISBN 7-111-12792-7

Ⅰ.软…　Ⅱ.①凯…②王…　Ⅲ.软件质量-质量管理　Ⅳ.TP311.5

中国版本图书馆 CIP 数据核字(2003)第 068439 号

机械工业出版社(北京市西城区百万庄大街 22 号　邮政编码　100037)

责任编辑：李　[illegible]

北京昌平奔腾印刷厂印刷·新华书店北京发行所发行

2003 年 10 月第 1 版第 1 次印刷

787mm×1092mm　1/16　·19.25 印张

印数：0001-4000 册

定价：39.00 元

译者序

本书自1995年出版以来，到2002年5月已经是第9次印刷，可见颇受欢迎。Kan博士曾在IBM Rochester长期工作，是有认证资格的软件质量工程师和可靠性工程师，是Application System /400(AS /400)产品开发的质量管理过程的过程经理。自1988年以来一直关注AS /400计算机软件系统的质量问题。这些技术经历和背景，使他能在写这本书时把理论、技术和实例恰当地融合。的确，理论同实际的结合是本书的显著特点。

对于软件质量问题的重要性从不认识到认识、从不重视到重视，似乎是个规律——不以人的意志为转移的客观规律。相信这本书的翻译出版是同这个大潮相适应的。

在这本书的翻译过程中，关于若干术语的译法碰到不少问题。这一方面是由于国外一些作者和机构对这些术语的用法历来就不同，另一方面是由于中西文化背景的差异。我们的原则是尽可能在这本书中统一译法，不致使读者混淆，并在必要时加注原文。另外，“信、达、雅”能全面兼顾，当然很好，但由于种种原因无法兼顾的时候，只有牺牲“雅”了。相信翻译过科技著作的人会同意这种取舍。

参加本书翻译工作的人员除封面署名外还有王志海、王晓军、杨士林、满益智、段姗、周一帆等。陈靖参与了录入工作。

2003年6月1日

序

在过去的几些年里，质量管理和工程有了多种多样的应用。例如：

- 在五年的时间里，由于进行了很好的质量管理，使原先一个保守估计要1780万美元投资的医院教学系统最终节省了250万美元。
- 美国空军军事空运司令部在海湾战争期间通过解决质量改进问题而使运输能力得到极大提高，避免了使用民用飞机(因而避免了中断次日邮件的投递，还有其他好处)。
- 美国劳务统计部为了生成月消费价格指数(Consumer Price Index，CPI)，需要五个部门的650个人的努力，现在所需的时间减少了33%，而且不降低准确性。
- 宾夕法尼亚大学从一个减少邮费的项目节省了6万美元。

这样的例子从电讯、保健、法律、医院、政府、制药、铁路和学校之类的产业接踵而来。TQM种子成功扎根的领域之繁杂几乎使人难以形容。

在其他领域都以革命性的速度投入质量改进时，软件开发人员和工程师却落后得很远很远，还在花时间辩论度量和过程模型，方法学和CASE工具。造成这种局面的原因是：确定软件系统用户需求的特有困难、许多软件机构中流行的“大人物”思路，以及软件工程应用的相对不成熟性。前两个原因成为有吸引力的议题有它们自身的原因，而Stephen Kan的这本书旨在针对第三个原因的挑战。

试设想用片段的航空工程知识去设计飞机，忽略机械工程的知识去设计汽车，没有化学流程工程知识去运作提炼厂会是一种什么情景。提供可信软件质量解决方案的顾问们的第一条建议就是软件界首先应该使用经过证明的软件工程方法。可是，软件开发界接受这些方法很缓慢，同样，接受质量工程方法也很缓慢。

造成这种接受缓慢的一个原因是，领先的软件开发机构使用这些方法的同时，却相对缺乏清晰描述这些方法基本原理的有关文献。Stephen Kan的这本书代表了满足这种需要的一个值得称赞的行动。

Kan博士提供了一本软件质量工程领域独有的全面的参考书。他在所需的技术细节和模型与技术的实际应用之间做了令人愉快的平衡。此书充满了业界实例，不仅有来自Kan所在的Malcolm Baldrige国家质量奖获得者IBM Rochester的例子，还有来自NEC的交换机系统分部、惠普、摩托罗拉、NASA软件工程实验室和IBM联邦系统分部的例子。用软件业的例子说明概念和理论，使本书阅读内容丰富得多。

读者大概会在其职业生涯中反复阅读此书多次，我却想引起对第12章的特别注意，该章描述了AS/400系统。这个活生生的案例研究将本书其他地方介绍的要素集中到一起，并为这些要素提供一个成功故事的重要背景。最后一章为数据质量这个正在形成的领域提供了令人激动的一瞥。

现代质量管理和工程的关键创始人之一Joseph Juran博士描述了“质量漂亮外衣后面的生活。”随着社会对技术更加依赖，技术的失效增加了不利影响。质量工程可以帮助企业远离这些

危险。质量在软件开发中的作用肯定能同质量在商业竞争的作用相媲美，并且这也是阅读、理解并应用本书中的思想的另一个引人入胜的理由。

Brian Thomas Eck 博士
Juran 研究所副所长
Wilton，康涅狄格州

前　言

从历史的角度看软件工程，20 世纪 60 年代及以前可以看作是功能时代，70 年代是进度时代，80 年代是费用时代。20 世纪 60 年代我们学会如何利用信息技术去满足机构的需要，并开始将软件同机构的日常运作联系起来。到 20 世纪 70 年代，由于当时软件产业的状况是大量的进度延迟和费用超支，关注重点是软件项目的计划和控制，引进了基于阶段的生命周期模型和出现了像人月神话(mythical man-month)之类的分析。到 20 世纪 80 年代，硬件费用持续下降。信息技术遍布到我们机构的每个方面并成为个人可用的东西。随着产业中的竞争变得激烈并广泛实现了低费用的应用，软件开发生产率的重要性显著增加，开发并使用了软件工程的各种费用模型。在 20 世纪 80 年代后期，也认识到了质量的重要性。

20 世纪 90 年代及其以后肯定是质量时代。由于最新的技术能够提供丰富的功能，顾客们对软件提出了高质量的要求。我们的社会对软件日益增加的依赖更进一步加强了对质量的要求。账单上的错误、大范围中断的电话服务、甚至最近海湾战争的导弹失败，都可以追踪到软件质量问题。在这个时代，质量已经成为软件开发过程的核心。从软件提供商的立场看，质量不再仅是公司成为市场优胜者的有利因素，它已经成为公司成功参与竞争的必要条件。

测量在有效的软件开发中起着关键作用。它为软件工程成为真正的工程学科提供了科学基础。本书描述了软件质量工程中的度量和模型：质量计划、过程改进和质量控制、过程中质量管理、产品工程(设计和代码复杂性)，可靠性估计和预测、顾客满意度数据的分析。大多数测量书籍采取一种百科全书式的方法，包括了每一种可能的软件测量。本书将其范围限制为软件质量的度量和模型，不包括诸如成本估计、生产率、人员配备和性能测量之类的内容，这些方面已经有大量出版物。

预计本书的使用者是软件产品管理人员、软件开发管理人员、软件工程师、软件产品保证人员以及软件工程、管理信息系统、系统工程和质量管理的学生们。对于学生，本书意在向高年级本科生或研究生提供一门课程的基础，同时还提供软件开发质量工程实践中的实用指南和例子。虽然涉及了方程和公式，但本书重点是度量和模型的理解和应用(而不是数学推导)。通贯本书，使用了来自 IBM Rochester 的 IBM Application System /400(AS /400)开发和软件产业中其他公司的大量实例。(IBM Rochester 于 1990 年获得 Malcolm Baldrige 国家质量奖并在 1992 年通过 ISO 9000 认证。) 第 12 章详述了 AS 400 软件质量管理系统的一个案例研究并提供了同前面章节中 AS /400 例子的联系。

第 1 章讨论软件质量的定义和全面质量管理框架。第 2 章回顾在软件业中使用的各种开发过程模型并讨论过程成熟度框架和几个质量标准。第 3 章考察测量理论的基础，它对于软件测量的实践是很重要的。第 4 章给出同软件生命周期各阶段相关联的主要软件质量度量，描述几个大软件公司的度量程序，并讨论软件工程数据收集问题。第 5 章描述质量控制的基本统计工具在软件开发中的应用，它们被称为 Ishikawa 的七种基本工具。第 6 章考察了缺陷排除有效性的核心概念、它的测量和在质量计划中的作用。

第7章至第10章论述用于不同目的的三个类别的软件质量工程模型：（1）用于质量评估和预测的可靠性模型（第7章Rayleigh模型和第8章指数分布模型和可靠性增长模型），（2）在开发过程中用于管理质量的质量管理模型（第9章），和（3）由软件工程师用以改进设计和实现质量的复杂性度量和模型（第10章）。

第11章讨论顾客满意度数据的测量和分析。第12章描述用于AS/400计算机系统开发的软件质量管理系统。这一章或是明确地或是隐含地引用了前面各章讨论过的途径、方法、度量和模型以及AS/400例子。最后，第13章提供若干与软件测量有关的一般性评论意见以及有关软件质量度量和模型的具体评论意见，而且提出测量在软件工程中的未来前景。

我要感谢Richard Hedger、David Amundson和Kathy Dunham，他们在本书的写作期间给以我不断的支持和鼓励。还要感谢我过去和现在的同事们，特别是Lionel Craddock和在IBM Rochester的开发质量和过程技术部的各位成员，感谢他们就软件度量这个主题进行过的许多非正式的讨论和他们提出的许多深刻见解。我还要向提供了编辑帮助的IBM Rochester的技术活力计划(Technical Vitality program)表示感谢。

我要感谢审阅本书的人们，尤其是Juran研究所的Brian Eck博士、质量软件技术公司的Richard Hedger、IBM Westlake(得克萨斯州)的Alan Yaung博士、IBM Rochester的Lionel Craddock、明尼苏达大学的Wei-Tsek Tsai博士和Skill Dynamics公司的James Abraham。他们提供了许多建设性的建议。还要向获准复制本书中用到的图和例子的作者、期刊和出版社致谢，书中还分别表示了谢意。

Stephen H. Kan，博士
Rochester，明尼苏达州
1994年8月

目　录

第 1 章
什么是软件质量

如果想要改进质量，就必须给质量下定义并测量它。质量工程和管理中的一个主要问题是质量这个术语的定义有歧义，因而常被误解。这种混乱有多种原因。其一，质量不是个单一的概念，而是个多维的概念。质量的“维”包括：感兴趣的实体、对实体的观点和实体的质量属性。其二，任何概念都有不同的抽象级别。当人们谈论质量时，有些人可能用其最广的含义，另一些人可能用其特定的含义。其三，质量这个术语也是我们日常语言的一部分，而且这个词的大众化观点可能非常不同于工程和管理角度的专业观点。

本章我们讨论质量的大众化观点、来自质量专家的正式定义和它们的含义、质量在软件中的含义和具体使用以及全面质量管理的方法和关键要素。

1.1 质量：大众化观点

质量的大众化观点是：质量是一个无形的特征——可以讨论，可以感觉和评判，但不能称、也不能量。诸如“质量好”、“质量坏”和“生活质量”的说法是人们谈论含糊事物、又不想去定义它的例证。这种观点反映这样的事实：即人们察觉了质量并以不同的方式理解它。这种观点的含义是：质量不可控、不可管理、也不能量化。这同质量工程学科中的专业观点鲜明对立。专业观点认为质量能够而且应当被定义、测量、监控、管理和改进。

另一种大众化观点是：质量就是豪华、等级和品味。昂贵、精致和较复杂产品被认为比那些寒酸的同类产品提供了较高的质量。所以，卡迪拉克是高质量的车，而雪佛莱不是，这同它们的实际可靠性和维修记录无关；

或者，环绕声高保真系统是个高质量的系统，而单扬声器的收音机不是。按这种观点，质量只限于那些少数有精巧功能的昂贵产品类别和有点级别的名目。那些功能简单又不昂贵的产品很难归为质量高的产品。

1.2 质量：专业观点

大众化观点的误解和含糊无助于产业界的质量改进工作。要做到这一点，必须给质量以可用以工作的定义。Crosby（1979）将质量定义为“符合需求”。Juran 把它定义为“适于使用”（Juran and Gryna，1970）。这两个定义互相关联而且相符，如稍后会看到的那样。质量的这两个定义已经被质量专业人员采纳和使用。

“符合需求”隐含着需求必须明确陈述出来，使它们不被误解。然后，在开发和生产过程中，不断进行测量以确定对这些需求的符合性。不符合需求就被看成是缺陷——缺乏质量。例如，对某种收音机的需求之一可能是它必须能接收 30 英里之外的广播源的特定频率。如果收音机做不到这一点，它就不满足质量需求，应该被拒绝。同样，如果一辆卡迪拉克车满足卡迪拉克车的所有需求，那么它是一辆高质量的车。如果一辆雪佛莱车满足雪佛莱车的所有需求，那么它也是一辆高质量的车。这两辆车可能在风格、性能和经济实惠方面很不相同。但是如果二者都达到为它们设置的标准，那么它们都是高质量的车。

“适于使用”这个定义考虑了顾客的需求和预期。顾客的需求和预期包括产品和服务是否适合他们的使用。由于不同顾客以不同方式使用产品，这意味着产品必须具有适合使用的多种要素。按 Juran 的说法，这些要素中的每一个都是一个质量特性。它们可以被分类为多个类别，这些类别被称之为适于使用的参数。两个最重要的参数是设计质量和符合性质量。

在大众化术语中，设计质量被称为级别或型号，它同购买能力的大小有关。不同级别之间的差别是由预期或设计差别产生的。再以小汽车的例子说，所有汽车都向用户提供交通服务。然而，型号在尺寸、舒适度、性能、风格、经济实惠和授予状态方面有差别。相反，符合性质量是产品同设计意图符合的程度。换句话说，设计质量是由需求和规格说明决定的，而符合性质量是同需求的一致程度。

所以质量的两种定义（符合需求和适于使用）本质上是相似的。差别在于“适于使用”这个概念隐含了顾客需求和预期的更重要作用。

顾客的作用

就同质量的关系而言，顾客的作用决不会被说过头。从顾客的观点看，质量是顾客购买产品的感觉价值，其根据是诸如价格、性能、可靠性和满意度这样的多种因素。在 Guaspari 的书《I Know It When I See It》(1985)中，在有关顾客的相关部分中是这样讨论的：

顾客们以内行的姿态向你说起质量，因为这才是他们真正要买的东西，他们不是在买一个产品。他们在购买产品会达到他们的期望的保证。

所以，除了那些保证之外，你没有别的什么东西可卖。除了质量之外，你没有别的东西可卖。

按概念的高层次定义，从操作上定义一个产品要涉及许多步骤，每个步骤都可能留下缺点。例如，为达到“符合需求”这一状态，必须首先收集并分析顾客的需求，必须从这些需求产生规格，再据此开发和制造产品。在此过程中的每个阶段，发生的错误都会影响最终产品的质量。需求可能是错误的（对软件开发而言尤其如此），开发和制造过程可能受制于那些诱发缺陷的变量，等等。从顾客的角度看，购买产品后的满意度是产品“符合需求”和“适于使用”的最终验证。从生产者的角度看，一旦规定了需求，依据规格说明开发和生产产品是实现质量的基本步骤。通常，无缺陷和可靠性好已成为产品质量最基本的测度。

由于关于质量的两种观点（顾客满意度作为质量的最终验证的观点和生产者坚持的“符合需求”以达到质量的观点），质量的实际定义由两级组成，第一级是内在产品质量，限于产品缺陷率和可靠性。这个狭义定义被称为“小 q”（q 就是质量）。质量的较广义定义(第二级）包括产品质量、过程质量和顾客满意度，称为“大 Q”。人们可以看到质量的这种两级定义正被许多产业使用，包括汽车产业、计算机产业（软件和硬件两方面）和消费电子产业。

质量的两级概念被假定形成一个闭环：顾客的需求→需求和规格说明→按需求进行产品设计、开发和制造，并持续关注过程改进→优秀的产品质量加上好的供应和服务过程→取得全体顾客的满意。然而，在许多产业里并非总是如此。尤其在 20 世纪 80 年代末（在那个时候，现代质量时代才开始）之前。经常没有顾客的介入就产生了产品需求，顾客满意度也不一直是做出商业决策的因素。虽然最终产品符合需求，但是它们可能不是顾客需要的。所以，顾客在质量定义中的作用应当明确地讲清楚：符合顾客需求。

1.3 软件质量

对于软件，最狭义的产品质量就是产品中没有“bug”。这也是“符合需求”的最基本含义，因为如果软件包含太多功能性缺陷，提供所需功能这个基本需求就不会满足。这个定义通常以两种方式表达：缺陷率（例如，每百万行源代码、每个功能点或其他单元中的缺陷数）和可靠性（运行 n 小时的失效次数，平均无失效时间或在规定的时间段内无故障运行的概率）。顾客满意度是按顾客满意度调查中的满意或非满意（中性的和不满意的）的百分比度量的。为减少偏向性，经常使用盲式调查技术（采访者不知道谁是顾客，顾客不知道采访者代表哪家公司）。对于软件产品，除总体的顾客满意度以外，还测量针对特定属性的满意度。例如，IBM 监控其软件产品的 CUPRIMDSO 满意级别［能力（功能）（capability）、实用性（usability）、性能(performance)、可靠性（reliability）、可安装性（installability）、可维护性（maintainability）、文档/信息（documentation/information）、服务（service）和整体（overall）］。惠普则关注 FURPS［功能性（functionality）、实用性（usability）、可靠性（reliability）、性能（performance）和可支持性（supportability）］。其他公司使用类似的软件顾客满意度量度。这些质量属性被 Juran 称之为质量参数或“适于使用”参数。

为了提高整体顾客满意度以及针对各种质量属性的满意度，一定要将这些质量属性考虑到软件的规划和设计里去。然而，这些质量属性并不总是互相协调的。例如，软件的功能复杂性越高，进行维护就变得越困难。依赖于软件和顾客的类型，不同软件属性需要不同的权重因子。对于拥有复杂网络和实时处理的大顾客，性能和可靠性可能是最重要的属性。另一方面，对于拥有单机系统和简单操作的顾客，易用、可安装性和文档可能更重要。图 1-1 说明了某些质量属性之间的可能关系。有些关系是互相支持的，有些是互相否定的，有些还不清楚，这依赖于顾客和应用的类型。所以，对于拥有不同顾客群的软件，为各种质量属性设置目标并满足顾客需求不是一件容易的事情。

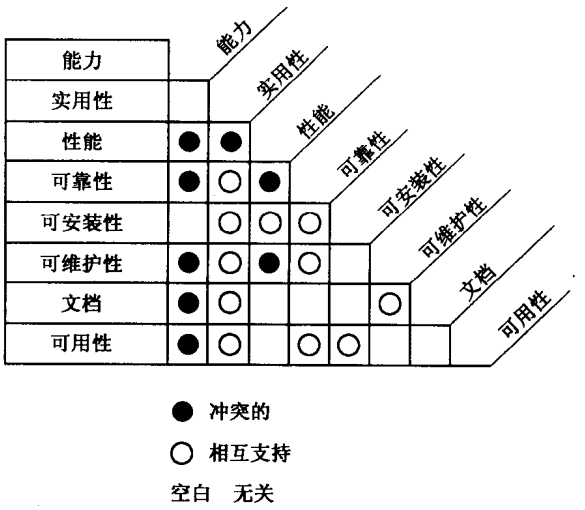

图 1-1 软件属性之间的内部关系——CUPRIMDA 的例子

按这些讨论的观点，质量的更新定义（即符合顾客需求）与软件产业是特别相关的。不用奇怪，需求错误成为软件开发中主要问题类别之一。按 Jones（1995）的说法，所有软件缺陷中有 15%（或更多）是需求错误。一个不解决需求质量的开发过程一定只能生产低质量的软件。

软件质量的另一种观点是关于过程质量对最终产品质量的观点。从顾客需求到软件产品的交付，开发过程是复杂的，而且经常涉及一系列的阶段，每个阶段又有反馈路径。每一阶段都为中间用户（下一阶段）生产中间交付物；每一阶段也从上一阶段接受中间交卑物。每个中间交付物有某些影响最终产品质量的质量属性。例如，图 1-2 展示了最常见的软件开发过程，即瀑布过程的简化表示。

有趣的是，如果我们将质量定义中顾客的概念展开成外部的和内部的顾客，这个定义也就适合于过程质量。如果开发过程的每一阶段都满足中间用户（下一阶段）的需求，这样开发和生产出来的最终产品也将满足规定的需求。当然，这种说法是现实的过分简化，因为每一阶段都存在影响这个阶段完全满足需求的众多因素。为了在开发期间改进质量，我们需要开发过程模型，并且在此过程中我们需要选择和部署具体的方法和途径，使用正确的工具和技术。我们需要这个开发过程及其阶段的特性和质量参数的测度，以及有助于确保开发过程受控于满足产品质量目标的度量和模型。质量度量和模型正是本书的关注焦点。

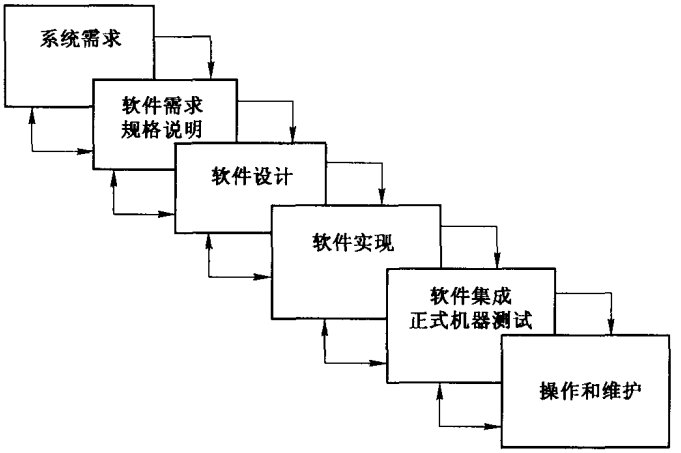

图 1-2 瀑布开发过程的简化表示

1.4 全面质量管理

全面质量管理（total quality management，TQM）最初是在 1985 年由（美国）海军航空系统部队用于描述其质量改进的日本式管理方法所创造的一个术语。它有许多意思，看谁在解释它并怎样使用它。然而，一般来说，它代表了一种旨在通过将质量同顾客满意度联系起来实现长期成功的管理风格。这种方法的基本点是创建一种文化，在这种文化中，机构中的所有成员都参与过程、产品和服务的改进。实现 TQM 思想的各种具体方法可在 Philip Crosby（1979）、W. Edwards Deming（1986）、Armand V. Feigenbaum（1961，1991）、Koru Ishikawa（1985）和 J. M. Juran（1970）等人的工作中找到。

从 20 世纪 80 年代开始，美国的许多公司在质量工作中开始采纳 TQM 方法，1988 年美国政府设立了 Malcolm Baldrige 国家质量奖（MBNQA），突显对这一思想和管理风格的接受。ISO 9000 被欧共体采纳为质量管理标准，在过去几年里，美国私人企业也接受这些标准，进一步说明了在当今的商业环境中质量思想的重要性。在计算机和电子工业中，TQM 成功实现的例子包括惠普的全面质量控制（Total Quality Control，TQC）、摩托罗拉