软件质量及其评价技术

中国软件行业协会上海分会
上海计算机软件技术开发中心 编

清华大学出版社

软件质量及其评价技术

中国软件行业协会上海分会
上海计算机软件技术开发中心 编

清华大学出版社

内 容 简 介

本书主要介绍软件质量科学的评价方法以及软件质量保证的技术。从介绍软件质量的度量方法与软件质量管理内容入手，重点介绍软件质量的评价体系，以及如何具体地、综合地评价软件质量。并综合国际上对软件复杂性的研究工作，详细说明有关软件开发、测试等文件的国家标准，还介绍软件系统的测试方法以及用户对软件系统的性能评价。

本书适于广大的计算机软件开发、管理与应用人员阅读，也可供理工科高年级大学生及研究生参考。

软件质量及其评价技术

中国软件行业协会上海分会
上海计算机软件技术开发中心 编

☆

清华大学出版社出版
北京 清华园
北京京辉印刷厂印刷
新华书店总店科技发行所发行

☆

开本：787×1092 1/16 印张：16.25 字数：409 千字
1990年2月第1版 1990年2月第1次印刷
印数：0001—6000
ISBN 7-302-00582-6/TP·206
定价：6.40 元

前　言

计算机已日益广泛地应用于各行各业，而计算机软件的质量将直接影响计算机应用的深度和广度。目前国内一些地区和部门已开始进行软件成果与产品的登录工作以及优秀软件的评审工作。如何科学地对计算机软件进行评价、测试和鉴定已成为不可回避的问题。因此必须积极开展软件成果和产品的评价、测试方法的研究，进一步改革和完善现行的计算机软件鉴定方法，逐步形成衡量软件质量的统一标准，促进软件质量的提高，以加速我国计算机软件的产业化、商品化的进程。

中国软件行业协会上海分会（即上海软件行业协会）和上海计算机软件技术开发中心近年来组织部分专家与技术人员开展了有关的研究工作。本书汇集并总结了其研究成果，以期推广和应用。

本书共分 8 章。第一章介绍软件质量的概念以及软件质量的度量方法，是全书的基础。第二章介绍软件质量管理的内容与管理活动。第三、四章是全书的重点，分别介绍软件质量的评价体系、具体的评价方法，以及如何对软件进行综合评价。这些科学的评价方法对于评选优秀软件具有直接的指导作用。第五章综合了国际上对软件复杂性的研究工作。第六章简要介绍用户对软件系统的性能评价。第七章详细说明了国家标准“计算机软件开发规范”、“计算机软件产品开发文件编制指南”、“计算机软件测试文件编制指南”与“计算机软件需求说明编制规范”，可作为编制软件产品各类文件的指南。第八章讨论系统测试与质量保证的关系以及系统测试的方法。

本书各章的编写者如下。第一章：王家坛、高毓乾，第二章：刘光龙、胡嘉宇，第三章：严洪范、刘光龙、周庆隆、方原、陈敏，第四章：刘小升、高汉卿，第五章：朱三元、周庆隆，第六章：张然，第七章：陈敏，第八章：居德华、陆斌。全书由朱三元、张然汇总和审阅，由朱三元定稿。

软件质量及其评价技术还有许多课题尚待进一步研究。由于我们执行与应用这项技术的时间较短，书中难免存在缺点和问题，热诚欢迎读者批评指正。

朱三元　1989.3.11

《软件质量及其评价技术》

编辑委员会名单

目　录

第一章　软件质量的基本概念

第二章　软件质量管理

第三章　软件质量评价

第四章　软件综合评价

第五章 软件复杂性度量

第六章 性能评价

第七章 文件标准

第八章 系统测试和质量保证

第一章　软件质量的基本概念

§ 1.1　软件质量的评测和管理

1.1.1　软件质量的重要性

产品质量是产品的生命。计算机软件与其它产品一样，也有质量的问题。例如，1981年4月10日美国准备发射一枚空间回收装置，在离发射时间尚有20分钟之际，计算机实时控制系统的软件突然出现一个故障，迫使发射延期进行。事前尽管花了数千小时进行测试与模拟，但仍未检测出这个隐患。从这个例子可以窥见软件质量的重要性。

如果开发单位为用户开发一个软件系统，在经费预算范围内，能按期完成并移交给用户使用，而且此系统也能正确执行用户规定的功能。那么，用户是否一定满意呢？不一定。其原因如下：

（1）软件产品可能难以理解，难以修改，因而在维护该软件过程中，费用大幅度增加；

（2）软件产品可能难以使用，或者容易出错。据1976年国外一篇报告统计，由于软件用错而使得美国政府在这一年额外增加不必要的支出一千多万美元；

（3）软件产品可能不必要地依赖机器，或难以与其它程序组合，使得系统难以扩充或移植。随着计算机类型的迅速增加，这类问题也愈加严重。

上面的情况说明如果软件有故障，将可能造成人力、物力和财力的巨大浪费。一个软件的质量不高，其维护费用也将大大超过其开发费用。另外，对一个软件而言，即使它没有故障，但如果它难以让用户理解和使用，将势必增加用户培训时间，从而耗费更多经费，维护也更困难。所以保证软件质量对开发者和用户来说都是必不可少的，也是十分重要的。

1.1.2　软件质量管理的重要性

良好的软件质量管理是获得高质量软件的重要保证，所以在软件开发自始至终的全过程，都必须不懈地加强软件质量管理工作。首先要了解、重视软件质量管理工作，更为重要的是确定、设计、实施和评价所期望的质量，并为此而制定一个工作计划。

人们在开发软件系统中曾有过这样一些经验：

（1）两个软件程序模块可以通过相同的确认和验证测试，并且对某规格说明提供同样的性能，但是在总体质量上仍有可能存在差别。究其原因是由于人们对初始系统概念和软件结构不太注意以致产生对有关环境的整体结构定义不完整。

系统概念涉及测试与试验要求、系统寿命、接口要求与配置情况等。软件结构包括操作系统、应用软件、高级语言、软件开发和调试工具、一般诊断手段、现有软件库与实用程序等。

（2）大约45%的软件问题在系统集成测试（联调）时发现，而且发现问题过程中所耗费用几乎占联调和运行使用时进行软件调试费用的80%。其原因在于对软件需求考虑不完整，在测试时涉及到更多的文件。如果分阶段开发和测试，则联调时错误会大为减少，费用也大大降低。

（3）开发工具和程序员手段不同，则所得到的软件质量与耗费的各种资源也不同。如使用汇编语言与使用高级语言就会有差别。

（4）开发过程所形成的各种文件（document）的有无以及文件是否简明、完整，也极大地影响软件质量。

软件质量管理，除了以上这些技术内容外，还包括人事方面内容，如软件开发课题组人员配备、技术水平、组织形式、工作计划与步骤等等。如果人员技术水平低，或者组织不当都会直接或间接地影响软件质量。由此可见，软件质量管理的重要性。

1.1.3 软件质量的科学管理和评测

计算机软件质量是计算机软件的所有内在属性的组合。“质量”不依赖于人们对软件要求什么，规定什么或测量什么，它仅依赖于软件的本质。应当注意的是软件质量包括计算机程序、数据和文件等多方面的质量。

软件质量管理是一项系统活动，它可分为质量设计和质量控制两大类。软件质量的管理和评测包括如下内容：

（1）软件质量要求和质量控制指标的规定

这是软件质量管理活动中的质量设计部分。质量设计的第一步是对最终软件产品规定一些所期望的质量特性或二级特性。当然，这些质量特性的提出应服从用户对性能的需求，此需求与软件所完成的功能有关。显然，在需求分析阶段要进行质量设计，提出质量要求并规定一些质量指标。例如，希望有“85%的灵活性”以及“99%的可靠性测量”（这些都是假设的量度）。

（2）软件开发过程中的质量控制

实践证明，在目前软件开发环境中，分阶段开发软件可及早发现错误，避免大规模返工，降低软件开发费用，提高软件质量。软件系统的分阶段开发过程，实际上是从用户要解决的问题到取得最后软件产品的一系列转换过程，即需求分析→设计→编码→集成与验证。每个阶段结束时都有相应的产品（阶段结果和最后结果）。要保证最后软件产品的质量，必须确保各阶段结果的质量。就是说对阶段结果要进行检验与评审。由于各阶段结果的形式与内容不同，因而检验和评审的对象以及合格的标准也有差别。为确保最终软件质量符合要求，必须在一个阶段工作完成之后，对其结果进行充分检验和评审，在改正所发现的错误之前，不要急于开始下一阶段的工作。一旦发现某阶段的错误与前面的阶段有关则应及时返回前阶段进行错误分析和修正。

（3）软件产品质量评价和验收标准

软件质量优劣直接影响着计算机应用的深度和广度，因而科学地评价软件产品质量就显得十分重要。只有科学地评价，才能客观地介绍软件的质量和特性，便于用户选择使用，从而保护用户利益；只有科学地评价，才能改革和完善现行的软件鉴定方法，以利于制定软件质量评价测试标准，推动软件开发的产业化、商品化进程；只有科学地评价，才能促进软件

开发单位重视全面质量管理，促进软件质量的提高，同时也有利于软件评价测试队伍的建立和成长。

软件质量由功能性、可靠性等若干质量特性决定，这些特性是否具有，以及具有的强弱程度决定着软件质量好坏，因此，对软件产品质量的评价和验收也以这些特性为标准。由于各个课题性质不同，对特性要求也就不同，同时还由于这些特性相互之间有矛盾的方面．因此．不可能要求每个软件产品对每个特性都达到可能达到的最大度量值，也就是说，使各项特性都同时达到最大值是难以做到的。一般来讲，如果软件产品的质量特性达到了能保证软件产品正常使用的程度，那末该软件产品的质量就是合格的。

（4）软件质量的可测性和定量化

软件质量由若干特性组合而成，通常用这些特性来描述软件的优劣程度。人们期望只有这些特性可以测量，而且测量结果必须是数值的。只有软件质量特性定量化，才能比较出该软件是否具有某种特性，以及具备该特性程度的强弱，从而推断出此软件质量的优劣。一般而言，测量某软件特性，其数据来源于软件的产品（中间阶段产品或最后产品），产品可以是程序、数据或文件。

§ 1.2 软件质量的定义和基本特性

1.2.1 软件质量的定义

参照ANSI/IEEE Std 729—1983，软件质量定义为："与软件产品满足规定的和隐含的需求的能力有关的特征和特性的全体"。或者

（1）软件产品中能满足给定需要的性质和特性的总体。例如，符合规格说明；

（2）软件具有所期望的各种属性组合的程度；

（3）顾客或用户觉得软件满足其综合期望的程度；

（4）软件的合成特性，它确定软件在使用中将满足顾客预期要求的程度。

除了上述定义外，还有其它一些定义。例如，M.J.Fisher 将软件质量定义为："所有描述计算机软件优秀程度的特性的组合。"所以计算机软件质量是软件的一些内部特性的组合。其中"质量"与人们对软件期望什么、规定什么或者测量什么无关，它仅依赖于软件的本质。因此，软件质量与那些描述计算机软件优秀程度的特性密切相关。

1.2.2 软件质量特性的定义

目前，软件质量由下列 6 个特性来定义：

（1）功能性：软件所实现的功能达到它的设计规范和满足用户需求的程度。

（2）可靠性：在规定的时间和条件下，软件所能维持其性能水平的程度。

（3）易使用性：对于一个软件，用户学习、操作、准备输入和理解输出所作努力的程度。

（4）效率：在指定的条件下，用软件实现某种功能所需的计算机资源（包括时间）的有效程度。

（5）可维护性：在一个运行软件中，为了满足用户需求，当环境改变或软件发生错误

时，进行相应修改所作努力的程度。

（6）可移植性：软件从一个计算机系统或环境转移到另一个系统或环境的容易程度。

§ 1.3 软件质量的二级特性

1.3.1 软件质量的二级特性

§1.2所定义的软件质量特性是面向管理的观点，或者说是从使用观点引入的。为了引进定量量纲，必须将这些面向管理的特性转化为与软件有关的观点，或是面向技术的观点。这种转化是通过对每个特性定义一组二级特性来完成的。二级特性进一步刻画了软件质量特性，并且有助于描述各个软件质量特性之间的关系，因为一个二级特性可以定义一个以上质量特性。这些二级特性是软件产品或软件生产过程的独立特性。利用这些二级特性，软件质量可被判定、解释和测量。

从软件设计观点出发，软件质量特性由下列二级质量特性所决定：

（1）可追踪性：在特定的开发和运行环境下，提供从实现到用户需求可追溯思路的软件属性。

（2）完备性：所需功能全部实现的软件属性。

（3）一致性：提供软件设计、实现技术和记号一致的软件属性。

（4）精确性：在计算和输出时提供所需精度的软件属性。

（5）简单性：在不复杂、可理解方式下，提供功能的定义和实现的软件属性。

（6）可操作性：决定与软件操作有关的规程，并提供有用的输入／输出的软件属性。

（7）培训性：提供对现行操作熟悉程度的软件属性。

（8）通信有效性：在执行功能时，使用最少的通信资源的软件属性。

（9）处理有效性：对于实现某种功能，提供最少处理时间的软件属性。

（10）设备有效性：对于实现某种功能，提供使用最少设备（包括存储设备和外部设备）资源的软件属性。

（11）模块性：提供高内聚，低耦合的软件属性。

（12）软件系统无关性：提供不依赖于软件环境（操作系统、例行程序和输入、输出子程序等）的软件属性。

（13）硬件系统无关性：提供与现行系统的微码及计算机结构无关的软件属性。

（14）自描述性：对功能实现进行自我说明的软件属性。

（15）结构性：提供软件结构良好程度的软件属性。

（16）清晰性：提供不复杂的、可理解的方式对程序结构作出清楚明了描述的软件属性。

（17）可扩充性：提供适应数据存储和计算功能扩充要求的软件属性。

（18）产品文件完备性：软件文件齐全、描述清楚及满足国家标准的软件属性。

（19）健壮性：在发生意外时，能继续执行和恢复系统的软件属性。

(20) 公用性：提供使用协议、例程、数据表示的接口标准的软件属性。

(21) 可见性：提供开发与操作状态监控的软件属性。

(22) 安全性：防止软件受到意外或蓄意的存取、使用、修改、毁坏或泄密的软件属性。

1.3.2 软件质量特性与二级特性的关系

软件质量特性与软件质量二级特性的关系如图 1-1 所示。其中质量特性用椭圆表示，二级特性用矩形表示。

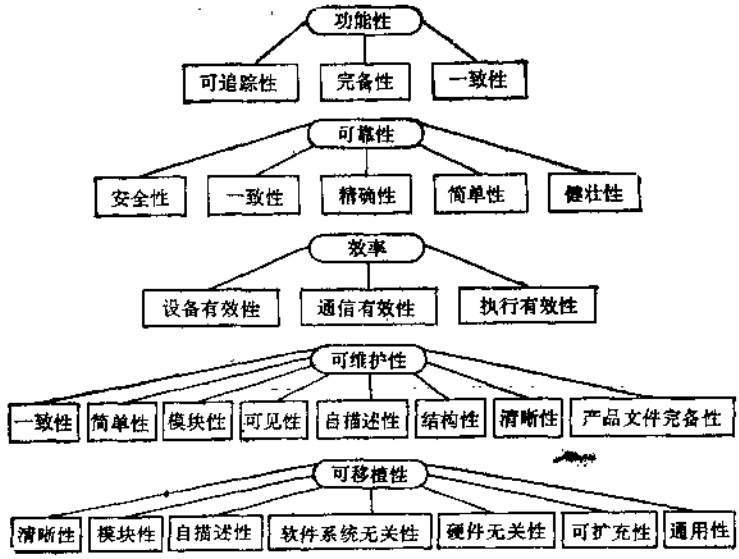

图 1-1 软件质量特性与质量二级特性的关系

§ 1.4 软件质量的度量

1.4.1 软件质量特性的定量化

从上述内容可知，软件质量由 6 个特性定义来描述并通过它们来衡量软件的优劣程度。这些特性又可进一步用22个二级特性描述。这些二级特性只是定性概念，而在实际测定某一软件质量时，必须对这些二级特性进行定量评测。

为此，我们引进“度量”（metric）概念，利用度量对二级属性进行定量测定。即软件质量度量可定量测量软件具有某给定属性的程度，进而可测定软件具有某质量特性的程度。显然，特性与度量是同时进行开发的，软件质量特性的定量化通过度量来进行。

1.4.2 度量的类型和例子

度量类型有预测型（predictive-type）和验收型（acceptance-type）两种，其中验收度量作为独立验收与确认，以及开发过程中预测评介的辅助工具，它可看作预测度量的确认。

预测度量是面向软件开发过程的，在开发阶段使用。预测度量基本上有两种。第一种象

一把尺子，是相对量测量。第二种为二元测量，它确定一种属性的存在或不存在。

为避免二义性并得到有意义的测量，相对量测量采用如下规则：实际个数与可能出现个数之比。

相对量度量的一个例子是复杂性测量，它在设计阶段应用于设计图，在实现阶段应用于源代码。这种度量基于路径流分析和沿着每个路径变量设置／使用信息。如果一个变量在程序中沿着某个路径可再次被使用，则该变量在该节点就被认为是“活的”（live）。复杂性测量是这样计算的：所有变量沿着该程序所有路径的“活性”（liveness）的总和。复杂性度量的规则是：上述复杂性测量值与该程序的最大可能复杂性度量值（即沿着所有路径的所有变量数）之比。

二元测量的一个例子是一张检验清单，它用于评价设计文件是否完全。清单中每一项都须给以记分，这依赖于特定信息存在（1）或不存在（0）。这种度量是对清单所有项记分的归一化累加。

上述的相对量度量与二元测量的实例如图 1-2，图 1-3 所示，以度量数据集中记分单形

特性：可维护性

二级属性或子属性	度量	需求		设计		实现	
		是/非 1或0	值	是/非 1或0	值	是/非 1或0	值
数据和控制流复杂性	复杂性测量(对每一模块)				□		□
	系统度量值：$\frac{\text{每一模块复杂性测量之和}}{\text{模块数}}$				□		□
⋮							

图 1-2 相对量测量例子

特性：功能性，可靠性

二级属性或子属性	度量	需求		设计		实现	
		是/非 1或0	值	是/非 1或0	值	是/非 1或0	值
完备性	完备性检验清单： (1) 无二义性引用(输入、函数、输出)	□		□		□	
	(2) 所有数据引用由一个外部源确定、计算或得到	□		□		□	
	(3) 所有定义的函数被使用	□		□		□	
	(4) 所有引用的函数被定义	□		□		□	
	(5) 对每一个判定点所有的条件和处理被定义	□		□		□	
	(6) 所有定义的引用调用的参数顺序一致			□		□	
⋮							

图 1-3 二元测量例子

```
1        SUBROUTINE  tabsch  (nsym  i, nflag)
2  C
3        COMMON/poltab/npt
4        DIMENSION npt (200)
5  C
6  C     serch  thru  data  block  npt  for  the
7  C     lenght  unit  of  the  symbol
8  C
9
10       nflag=0
11       maxsym=200
12 C
13       DO  200  j=i,  maxsym
14 C
15       IF (npt(j).EQ.0) GOTO 300
16       IF (nsym.EQ.npt (j)) GOTO 100
17 C
18       j=j+2*npt(j+1)+4
19       GOTO  200
20 C
21  100  i=j
22       nflag=1
23       j=maxsym
24 C
25  200  CONTINUE
26
27  300  RETURN
28       END
```

(a)

可能的度量
GOTO数/卡片数
GOTO数/语句数
GOTO数/例行程序
GOTO数/可执行语句数
GOTO数/块程序

(b)

图 1-4 选取一个度量

式，来记录测量结果。

采用相对量测量，要加小心否则会产生不正确结果而导致度量失败。例如，无条件转移个数（GOTO语句数）可与二级属性的清晰性相关联。图1-4(*a*)是用FORTRAN语言写的程序段，(*b*) 说明 (*a*) 中无条件转移数的一些可能度量。GOTO 语句数/可执行语句数是没有二义性的度量，因为GOTO语句是可执行语句集合中的真子集。而空白卡、注解卡和说明语句则不可能是GOTO语句，因此，GOTO语句数/卡片数以及 GOTO 语句数/语句数这些度量，对于GOTO语句与相关质量特性的关系而言会带来二义性。而GOTO语句数/例行程序和GOTO 语句数/块程序并不能显示出可能出现集合的规模，因而其本身是二义性的。相对量度量的值是在 0 与 1 之间变化。

1.4.3 软件质量度量体系

软件质量是复杂的，不同人对质量有不同的理解。管理人员谈质量，可能指服从一些标

准并在经费与进度限制范围内实现所需功能。终端用户可能指使用方便，响应时间快。程序员可能指正确执行并符合程序标准。而维护人员则可能指文件清晰、代码是可理解的。此外不同的软件系统其关键特性也会有差别。可移植性是某系统的重要特性，然而对另一个系统则不是。内存的有效使用对某系统可能是重要的，而对另一系统则不然。同时，质量的不同特性相互之间也可能是矛盾的。例如使用高级语言可提高可维护性，但是运行速度和内存空间利用率就不如用汇编语言。

软件质量的复杂的、捉摸不定的性质是难以获得预想软件的主要原因。没有实用的软件质量的体系或模型、项目，管理人员就无法知道如何为系统建立质量需求。没有清楚地叙述系统的质量需求，开发人员就不知道要建立哪些质量特性，验证人员就不知道要测量哪些质量特性，最后管理人员也无法确定已经获得哪些质量特性。

因此，建立一个软件质量度量体系是极其重要的。体系可以对一软件系统建立一些质量需求，在软件生存期每个阶段利用一些方法测定这些软件特性，以保证满足这些需求，并验证一些质量测定结果。

至此，我们已经建立了软件质量度量体系，如图 1-5 所示。

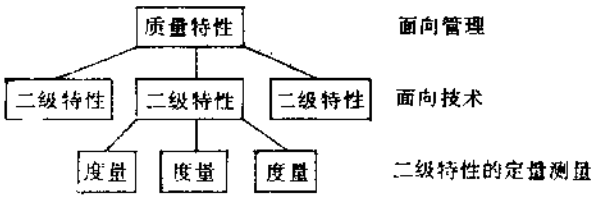

图 1-5　软件质量度量体系

此体系由质量特性，二级特性和度量三个层次组成。体系第一层是质量特性，它代表软件质量面向管理的观点。这些特性对各类管理员直接有用，它为系统开发初期建立质量需求提供工具。例如，如果希望软件生存期长些，那么寻找并确定一些错误成为考虑费用的关键。如果某系统为试验系统，它的软件规格说明显然更改频繁，那么软件产品的灵活性必须引起高度重视。

体系第二层为二级特性，它是质量特性分解和转换的结果，是面向技术的观点。二级特性是软件的独立属性，它对各类技术人员，如分析员、设计员、程序员、测试员和维护员更为有用。将一个质量特性分解为若干个二级特性，有利于管理员与有关技术人员的对话。

体系第三层是度量。每个二级特性分解为若干度量。一个度量可以对应于几个二级特性。度量是一些信息或数据，它可从软件产品的开发过程取得。有些度量很简单，可以直接观察得到，其它则要通过详细计算才能取得结果。

体系的从顶到下的机制有利于下述管理活动：在系统生存期早期，管理员利用质量特性建立一些质量需求；对建立起的特性，利用二级特性与技术人员对话；建立起质量特性的二级特性与特定项的关系。因此，这种从顶到下机制为建立、通信和评价质量需求提供一种工具。反过来，体系的由底向上方式，基于产品或过程评价为管理和技术人员提供反馈信息。

从图中也可看到，该体系很灵活，允许在它上面对软件质量特性进行增、删、改，对二级特性和度量亦是如此。因此，该体系可应用于所有的软件。

1.4.4　度量的应用

由质量度量所表示的测量结果可以应用于开发的各个阶段，对于所希望的产品质量提供

增长（质量增长）的指示。在生存期的早期，这些度量可以使用，并具有较多的“指示”意义。显然，如果设计说明是高质量的，而没能很好地实现，那么得到的软件产品将不是高质量的。在开发各阶段连续运用度量有助于防止偏离开发目标的情况。软件度量概念如图 1-6 所示。软件度量的应用如下所述。

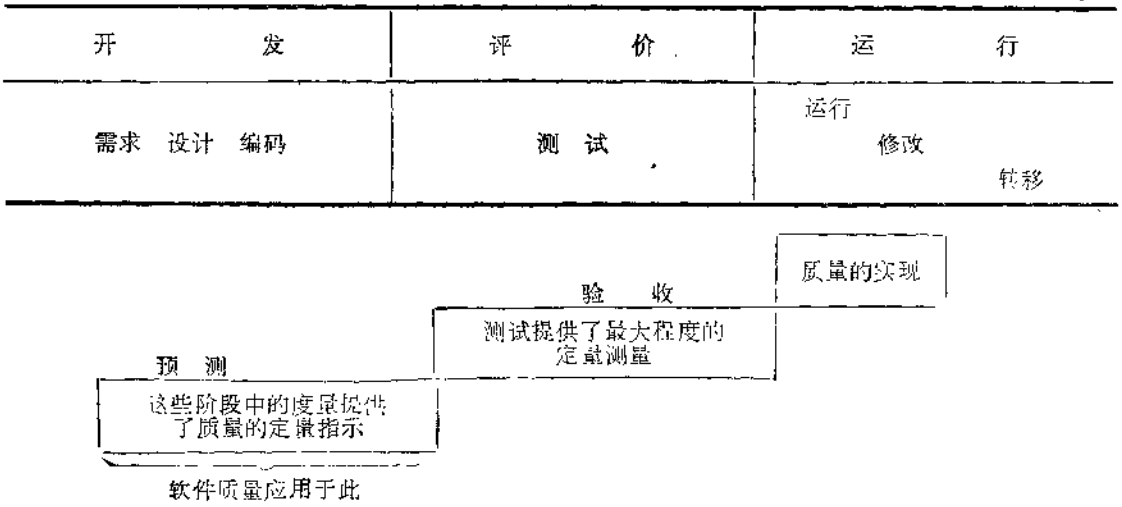

图 1-6 软件度量的概念

1. 用于质量规定

当开发一个新系统，特别软件是其很大一部分时，开发组织面临的主要问题是如何确定哪些软件质量是重要的以及随后如何以需求形式规定它们。这一点在我国当前的状况下没有引起足够的重视。在需求分析阶段时，仅仅注重用户功能需求，而对性能要求以及质量需求很少提及，是一个很大的错误。对于一个软件产品来说，不提质量需求，那就意味着潜伏了危机。下面讨论如何提出软件质量特性需求。

每一个软件系统与规定质量相关的软件质量需求都是唯一的。有一些基本的系统特征影响着质量需求，而且每个系统必须对其基本特征进行分析。这些基本特征和相关的质量特性的例子如表 1-1 所示。

表 1-1 系统特征与相关的质量特征

特征	质量特征
人类生活受影响	功能性 可靠性 可维护性
延长生存期	可维护性 可移植性
实时应用	效率 功能性 可靠性
保密的信息处理	可靠性