科学普及出版社

软件质量的鉴定

〔美〕B.W. 博姆　等著

软件质量的鉴定

〔美〕B. W. 博姆 等著
徐竹平 李邦 合译
尤定华 审校

科学普及出版社

内 容 提 要

随着计算机技术的发展，计算机软件工作变得日益庞大和复杂。如何研制一个高质量的软件工程，越来越引起人们的重视。但研制一个软件，鉴定其质量的好坏，必须有科学的方法。本书正是解答了这个问题。读者阅读本书，不要求有很深的计算机专业知识，只要有一定的程序设计基础和实践，就能读懂。本书叙述比较系统和完整，且有丰富的实例，无论对有经验的专业工作者，还是对刚入门的软件工作人员，都可以为研制高质量的软件提供指导和帮助。

«Characteristics of software quality»

BARRY W. BOEHM
JOHN R. BROWN
HANS KASPAR
MYRON LIPOW
GORDON J. MacLEOD
MICHAEL J. MERRITT

软件质量的鉴定

〔美〕B. W. 博姆 等著
徐竹平 李 邦 合译
尤定华 审校
责任编辑：吴之静
封面设计：王序德

科学普及出版社出版（北京海淀区白石桥路 32 号）
新华书店北京发行所发行 各地新华书店经售
中国科学院印刷厂印刷

开本：787×1092 毫米 1/32 印张：6.625 字数：167 千字
1987 年 7 月第 1 版 1987 年 7 月第 1 次印刷
印数：1—5,700 册 定价：1.30 元
统一书号：15051·1229 本社书号：1454

ISBN 7-110-00097-4/TP·4

译　者　序

随着计算机硬件的发展，计算机软件工作变得日益庞大和复杂。早期小型的软件技术、方法和经验已不适应于现代大型软件的开发。同时，计算机的广泛应用，导致了从事软件设计工作的人员不断增加，如何研制一个高质量的软件工程，越来越引起人们的重视。研制一个软件，对其质量的好坏，如程序的可靠性、可移植性、可维护性以及研制的费用和时间等等，必须有一个科学的质量评估标准和评估方法。为了适应这样的需要，我们翻译了《软件质量的鉴定》这本书。

本书不要求读者有很深的计算机专业知识，只要有一定的程序设计基础和实践，就能阅读。而且书中的叙述比较系统和完整，并有丰富的实例，是一本可供参考的读物，无论对有软件设计经验的专业软件工作者，还是对刚入门的软件工作人员，都可以为研制高质量的软件提供指导和帮助。

本书由英俄对照译出（徐竹平同志根据英文原本翻译，李邦同志根据俄译本翻译）对原文的编排和内容，根据俄译本作了某些删减和变动。为了便于初学者阅读，译者编写了102条名词解释（见附录 B）作为参考。由于译者水平有限，译文中错误及不当之处在所难免，恳请读者指正。

译　者

1985 年 12 月

目　录

序　　言

为什么要评价软件质量？

如果你接收了一个按时交付的软件产品，它能准确而有效地完成全部专门功能，而且费用也在预算之内。但是由于种种原因，你不一定感到满意。下面便是你可能遇到的几个问题：

（1） 软件产品可能很难看懂，也很难修改。这将使软件维护方面的预算费用超支，而且这些费用将是很高的。例如，由 Elshoff[1] 写的最新论文中指出，General Moter 软件研制费用的百分之七十五是化在软件的维护上（GM 软件工程是一个相当典型的大工业软件工程）。

（2） 软件产品可能很难使用或者很容易用错。美国国家统计局（U. S. A. GAO——General Accounting Office）的报告指出，为了解决自动数据修正处理（ADP——Automic Data Proccessing）问题，国家预算的管理费用超过一千万美元，其主要原因就是因为软件太容易用错。

（3） 软件产品可能与机器有着不必要的依赖性，或者与其它程序很难组合。由此可见，一个软件产品，尽管能完成全部所要求的专门功能，但并不能说这个软件产品的质量是高的。所以对软件的质量需要作出评价。

有许多常见的因素可能会对软件质量产生强烈的影响，所以，正确地理解软件质量的各种特性是很重要的。这里有几点需要注意：

（1） 对一个软件产品制定质量技术规范。把你需要软

件具有什么功能以及你需要软件具有什么性能（具有多高的速度，达到多高的精度等）要求，可以直接地列出公式。当然，指出软件需要具有可维护性和可理解性，这同样也是重要的，但在对可维护性和可理解性进行检验的方式方面，要列出公式是比较难的。

（2） 软件产品是否符合质量规范的检查。 如果质量规范已经制定，那么对软件产品与质量规范进行一致性的检查是必不可少的。虽然它可以通过大量投资由人工来进行，但这样做既贵又难。

（3） 做出适当的设计，使费用兼顾研制费与操作费，这也是极为重要的。因为在研制时，紧凑的预算和进度要求会影响软件产品的可维护性、可移植性以及实用性等。

（4） 软件包的选择。这里还要再一次说明，许多用户还要对每个软件包适应他们配置的变化以及硬件资源的难易程度等方面进行相对的估计。

提高软件质量性能的主要结果，将大大改进软件维护的费用——效率比。太低的质量性能（如可维护性），将直接转化为过高的费用支出（即生存周期的维护费用，如对错误的修正以及对新用户要求的响应）。这是一个简单的道理，可是到目前为止，在评价软件质量的领域里，还没有进行比较严密的和深入的工作，这一点是令人感到奇怪的。

本书的主要内容是建立一个有关软件质量分析的概念性体系和一些关键性的初步成果。其主要成果和结论是：

（1） 注重软件质量性能，必将使软件生存周期的消耗费用得到显著的降低。

（2） 当前软件的发展水平限制着对软件质量进行自动和定量的评价能力。

（3） 对软件的质量性能标准，提出了有明确定义和明确

分类的层次结构。这个结构的较高层特性反映了它的实用性，也就是在实用中可以确立对软件质量的评定。而它较低一层的特性，与能够实现的实用软件质量性能指标的评价是紧密相关的。

（4） 对大量软件质量特性的评价标准，也给出了定义和分类，而且对它们的潜在效益、定量计算和自动化的难易程度作了估计。

（5） 对专用软件生存周期的活动律进行了鉴定，它对于提高软件质量起着显著的杠杆作用。

我们认为，最有意义的是本书通过一致的和互相支持的若干组定义、特征、准则和已被引用的经验，对可能与软件质量经常起作用的有关因素，提供了一个清晰而又明确定义的体系。虽然这个体系还不一定完整，但有一点是可以肯定的，那就是它能作为进一步精炼和深入研究的有用基础。

本书的内容按下列顺序介绍：

第一章　软件质量评价问题的状况

第二章　软件和软件编制过程

第三章　优质软件的质量特性

第四章　软件质量性能的测定指标

第五章　编制优质软件的指导原则

附录A　备用的软件质量性能测定指标的详细定义（附有使用 FORTRAN 语言的部分例子）

附录B　名词解释

附录C　参考文献

第一章　软件质量评价问题的状况

1.1 过去的研究

最初研制软件质量评价方法的人大概是 Rubey 和 Hartwick[3]。这种方法（参考资料 3）是：定义一些程序的"属性"和它们的"度量标准"，前者是对所希望的质量特性给以文字的叙述，后者是一些参数的数学函数，这些参数是为该属性而设定的，用来直接或间接地反映具体确定的属性。所谓属性，就如：

《A_1——数学计算的正确性》

《A_5——程序的可理解性》

《A_6——程序的易修改性》

为了更具体地定义"可以被直接测定的属性"（也就是在 0 到 100 的尺度范围内，把属性表示成软件的某个等级），必须对上面的每个属性进一步加以分析。尽管在参考资料中对每个主要属性进行了详细的分类，但也只是对其中几项度量标准进行了说明，此外，也没有提到专门的应用实例。

在专家们最近的研究文章[1]里，将质量特性测定指标——即度量标准公式化了，并且给出了它们在一个控制试验中、对两个用同样的规范而又互相独立编制的计算机程序（每个程序约有 400个 FORTRAN 语句）进行的应用。在该项研究里，仅考虑了有限的属性，这些属性基本上相当于前面参考资料（3）里的 A_5 和 A_6。

此外，从这两个程序设计的效果来看，在质量的着重点上

有着显著的差别：一个是由一位“激动”型的编程员编制，促使他编程效率达到最高；另一个是由一位“深思”型的编程员编制，他却着重强调简单。这份研究报告的主要结论是：

（1） 在“高效编程”的程序里被探测到的错误比“简单编程”的程序里的错误多达 10 倍（超过同样的 1000 次连续测试运行）。

（2） 在“简单编程”的程序里，程序质量性能指标显著地高。因此，这些指标很好地指明了有关操作的可靠性（至少在这方面）。

其他专家同样也认识到，描述软件性能以及明了地论述软件质量属性的重要性。Wulf[5] 经过划分，提出了七个重要属性的简明概念，它们既合理又互不重叠，这就是可维护性或可修改性、健壮性、清晰性、可运行性、耗费性、可移植性以及人的因素。Abernath[6] 等人确立了若干操作系统的质量性能，同时也分析了它们之间的一些综合性能。Weinberg[7] 进行了试验，由几个程序员组，给予相同的任务而不同的完成标准（研制速度、语句数、存储量、输出清晰性以及程序表达性）。每个质量性能的最佳值都被相应的组所获得。

由此，越来越多的专家开始认识到软件质量的重要性，并且着手用编制《优良程序设计实用参考书》的方式，含蓄地论述了软件质量属性的必要性。这方面工作的最好例子是由 Kernighan 和 Plauger[8] 所编的有关程序设计风格的书。此外，还应该指出的是：由 CIRAD[9,10] 在实验领域里发表的一系列有关软件可维护性的报告中，提供了一些原始资料以及下面列出的实用检查表和提高软件可维护性的一些项目：性能数据组（把具有共同特性的数据整理成数据组）、自顶向下的程序设计模块化、一致性、紧凑性、可理解性、可移植性、注释、插入语和名称。在W arren 写的一篇有关软件可移植性

的报告中[11]，对可移植性提出了一个可供选择的多种形式的概括——例如：模拟、仿真、翻译、程序自展、使用高级语言——在语言处理程序的可移植性方面所具有的基本重点。

近来，许多开创者都已认识到，在软件工程里，需要明确考虑质量因素的重要性。例如，在最近的 AIAA/ACM/IEEE/DOD 软件管理学术年会上，有四位代表的报告涉及到了这个课题：Dekoze 介绍的《软件质量的技术规范和合理综合》[12]是国防部（DOD）首先做的工作；Kossiakoff[13] 的报告中叙述了"好"的软件产品应具备的规范中的七项特性；Whitaker[14] 为新的 DOD 编程语言提出了十二个明确的质量指标；最后在 Light[15] 的报告中谈到，为保证编制出高质量软件，整理出了五项重要的测定软件质量的指标。

此外，在软件质量性能指标的测定和质量分析方面；最近已经在好几篇长篇论述中都给予了强调。Myers 写的《软件的可靠性》[16]，专门研究了在软件产品里，建立高度可靠性以及估价软件可靠性的技术，这种技术的本质就是，把估价软件可靠性看成一个被监视到的测试结果和错误记载的函数。此外，还在关于程序设计风格和程序设计语言的章节里，论述了其它一系列软件质量的性能指标。Halstead 在《软件科学的原理》[17]一书中概括了一个研究的重大进展，这就是把建立和证实某个软件复杂程度的通用验证尺度，表示为软件中不同的操作和操作数的数目以及它们在软件中出现频率的函数。

Gilb 写的书——《软件质量性能指标的测定》[18] 内容上很接近于本书。这是一本讨论有关软件质量性能的书，它论述了以形式化的(可能的话以定量的)方法来研究这些标准的重要性。它也是一本范围广泛的书，它概括了这些研究以及在这个领域里其它大量的成果，其中也继承了 Gilb 本人大量的研究成果。

在软件质量评价方面的某些关键论点归纳如下：

（1） 有没有可能定义一种软件质量特性？ 这些特性是可以测定的，而且彼此间互不重迭，它们对软件质量是可以作出评价的。 有关这个问题将在本章的第二节“优质软件的质量指标”里加以讨论。

（2） 如何能更好地测定整个软件的质量或者是它们的个别特性？这将在第三节“程序质量的测定”中讨论。

（3） 如何能用软件质量特性的信息来改进软件的生存周期过程？这将在第四节里加以讨论。

1.2 优质软件的质量特性

软件质量特性研究的主要任务是要提出一组软件质量的特性，同时对每个特性要确定一个或多个测定指标，如：

（1） 对一个任意的程序来说，这些测定指标可以定量地判定该程序属于哪一级。对应于每个级，程序具有相应的性能。

（2） 引出整个软件质量的综合评价，它是这些质量特性测定指标值的函数。

虽然“软件工程”可能由许多部分组成，如例功能规范、调试计划以及操作指南等，但下边的研究仅着重于 FORTRAN 源程序的质量特性测定指标。

研究的初步结论归纳如下。首先，在软件产品的研制和评价方面，人们常常把兴趣注重于产品在什么地方和怎样出现故障，而不是注重于产品为什么经常出现故障。因此，对软件质量的分析最有价值的自动工具，通常只是注明在程序中有哪些缺损或异常，却并不注重于出现的次数。在过去，象编译这样的诊断程序，它的诊断结果只有一个语句：“你的语句

中有1.17% 的括号不平衡”。这种空洞的诊断，常常使编制人员深感欠缺。

第二，我们发现，在实际中，对任何简单的数字关系的表达式，很容易找到反驳的例证，而这些例证又正是针对着可靠性的，所以，这些例证就可以作为软件质量特性——可靠性的测定指针。下面有几个例子：

（1） 研制一个质量特性的测定指标，用来计算程序模块的平均尺寸，并以此作为程序结构性的一个度量。因而，假定有一个软件产品具有 n 个长度为 100 个语句组成的控制程序和一个由 m 个长度为 5 个语句的计算程序组成库。那么，这个软件产品则被认为对任何合理值 m 和 n 都具有比较好的结构。然而，若 n = 2，m = 98，则平均模块长度便是 6.9 个语句。而当 n = 10，m = 10 时，则平均模块长度便是 52.2 个语句。

（2） 研制一个“完整性”的测定指标。它是针对含有潜在计算异常（例如除法、平方根、对数等）的语句部分，对这部分语句，预先用针对计算异常的测试语句和补偿语句来加以排除，因而常常使程序是在计算异常不可能发生的范围内进行的。有一个简单的例子可以说明，这就是对直角三角形斜边的计算：

$$Z = SQRT(X**2 + Y**2)$$

（3） 针对程序中注释卡的数量和注释语句的平均长度等方面，制定了一些称作为“自说明性”的质量性能测定指标。然而，用比较少的和比较短的注译语句来检索程序并不算难，而这些短而少的注释语句，比那些冗长而内容贫乏的注释语句更容易理解。

第三，我们认为，在软件研制技术的领域里，现在依然处于迅速发展的状态中，以致在某些范围内，不可能建立软件质

量性能的测定指标。实际上，这样的情况会造成加强通常的编制软件的过程。当然，这样的工作方式不可能是好的。这儿有一个例子，当采用了数据组和自动打印检测时，会使某些可靠性的质量性能测定指标无效，如为检测混合型表达式及参数范围的干扰等用的测定指标。

最后，我们认为，计算和理解一个全面的软件质量性能测定指标的具体数值，可能要比它的价值更困难，主要的问题在于许多单项的软件质量特性是互相矛盾的。进一步说，就是效益常常依靠可移植性、精度、可理解性和可维护性而获得；计算机的计算精度常常又与可移植性相矛盾，因为它们都取决于计算机的字长；简明性与易读性也是相矛盾的。在这种互相矛盾的情况下，用户常常会发现，要从数量上区分哪个指标更需要是很困难的。另一个问题是，这些质量特性测定指标常常不能完整地度量软件质量的相关的性能。归纳这些观点为：

（1） 一个软件产品所希望的质量性能将随着未来用户的需要和测重点而变化。

（2） 因而，没有一个单一的软件质量特性测定指标，能对软件质量给出一个有普遍意义的评价。

（3） 一个希望使用软件的用户，最好能得到由整个检测表和优先级装备起来的评定系统对该软件所作出的有用的评价。

（4） 即使如此，由于这些质量特性测定指标并不是完善的，所以就是给出总的评价也只能是参考性的，而不具有明确的结论性。

（5） 因而，从这一点出发，对各种这样的测定指标，最好是作为计算异常的指示器来使用，用来对软件研制、调试计划、探测错误和维护等方面进行指导。

软件质量特性的鉴定和分类。

由于获得了上面的这些结论，我们确定研制的任务是：提出一个软件质量特性相互联系的性能相关组和建立一个能够反映软件异常情况的指标系统，即异常——探测系统。我们的步骤和任务是：

（1） 确定一组软件产品是重要的，并且是适当完整的和不重叠的质量特性组。

（2） 提出一组备用的质量特性测定指标，作为评价的等级，在各个等级中，软件具有确定的质量特性。

（3） 分析研究这些质量特性和有关的质量特性测定指标，以确定它们与软件质量的相关性、使用的潜在效益、定量的可能性以及自动化的难易程度等有关的各项。

（4） 按照上面的这些准则以及它们与其它的特性测定指标：重迭性、依赖性、缺陷等的相互作用，来评价每个备用的质量特性测定指标。

（5） 根据这些评价，重新归纳出一组软件质量特性组，使它们在评价软件质量时，能够更加独立和趋于完善。

（6） 改进并确定备用的质量特性测定指标，并按被修改的质量特性组的顺序重新组合它们。

作为第一步，提出和确定了下面的质量特性的初始组：（1）可理解性，（2）完整性，（3）简明性，（4）可移植性，（5）一致性，（6）可维护性，（7）可测试性，（8）可适用性，（9）可靠性，（10）结构性，（11）有效性。这些特性的定义将在本书的第三章内加以介绍。

作为第二步，我们确定对FORTRAN程序的测定指标，作为该程序的可理解性、可维护性等的有效指示器。在这个过程中，我们发现，任何可理解性的程度也是可维护性的程度，因为任何代码程序的维护都要求维护者理解代码。但另一方