第32卷 第13期 计 算 机 工 程 2006年7月
Vol.32 № 13 Computer Engineering July 2006
·软件技术与数据库· 文章编号：1000—3428(2006)13—0041—03 文献标识码：A 中图分类号：TP311.5

面向对象软件度量结果的分析方法与评价

冯素云[1,2]，侯惠芳[1,3]
(1. 河南工业大学信息科学与工程学院，郑州 450052；2. 北京理工大学生命科学与技术学院，北京 l00081；
3. 解放军信息工程大学 NDSC 研究所，郑州 450002)

摘 要：软件度量是保证软件质量的重要方法，该文主要针对面向对象软件经过分析度量后得出结果进行分析与评价，分析与评价主要基于面向对象程序中类的耦合度、内聚度、继承性、复杂度等进行展开，从数学的角度来分析度量指标的可信度，以保证从整体上把握面向对象软件产品的质量。

关键词：面向对象；耦合；内聚；继承；软件质量

Analysis Method and Evaluation of the Results for Object Oriented Software Metrics

FENG Suyun[1,2], HOU Huifang[1,3]
(1. College of Computer Science and Eng., Henan University of Technology, Zhengzhou 450052; 2. School of Life Science and Technology, Beijing Institute of Technology , Beijing 100081; 3. NDSC Institute, PLA University of Information Engineering, Zhengzhou 450002)

【Abstract】 Software metrics is an important method for software qualities, but this paper mainly introduces the analysis of the results for object-oriented software metrics, which is based on classes analysis, inheritance analysis, coupling between object classes analysis, cohesion in methods analysis, complexity analysis and so on, and it analyzes the possiblity of metrics-quota with math knowledge in order to ensure the whole quality of object-oriented software.

【Key words】 Object-oriented; Coupling; Cohesion; Inheritance; Software quality

面向对象软件度量是针对计算机面向对象软件的度量，是对一个软件系统、组件或过程具有的某个给定属性的度的一个定量测量。通过度量，可以对面向对象软件给出客观的评价，指出面向对象软件属性的趋势，能有针对性地进行改善。确认已知度量的有效性是目前开展较多的度量研究课题。常使用的确认方法是用度量对软件质量特性作预测。

1 问题的提出

目前使用软件产品度量对质量特性做预测的工作都是使用统计学的方法，如方差分析和回归尤其是一元线性回归，是通过对实践数据进行分析完成的。例如，Li 和 Henry 在 1993 年成功地使用 C&K 度量集来预测软件模块的维护工作量。Basili 等在 1996 年使用几乎是同一组度量成功地预测了软件模块的出错率。Subramanyam 和 Krishnan 在 2003 年也用 C&K 度量集，在控制了模块大小的条件下，成功地预测了有缺陷的模块。这些研究大多数是基于实践方法的。度量的形式化研究目前处于在零星的个人探讨阶段，并未形成科研成果。而度量当中的一个重要环节就是要对度量结果进行分析与评价。

本文从以下几个方面进行分析与评价：

(1)软件质量要素：软件质量可分解成 6 个要素，这 6 个要素是软件的基本特征。

1)功能性：软件所实现的功能满足用户需求的程度。功能性反映了所开发的软件满足用户称述的或蕴涵的需求的程度，即用户要求的功能是否全部实现。

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。可靠性对某些软件是重要的质量要求，它除了反映软件满足用户需求正常运行的程度，且反映了在故障发生时能继续运行的程度。

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时，所做努力的程度。易使用性反映了与用户的友善性，即用户在使用本软件时是否方便。

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。效率反映了在完成功能要求时，有没有浪费资源，此外“资源”这个术语有比较广泛的含义，它包括了内存、外存的使用，通道能力及处理时间。

5)可维护性：在一个可运行软件中，为了满足用户需求、环境改变或软件错误发生时，进行相应修改所做的努力程度。可维修性反映了在用户需求改变或软件环境发生变更时，对软件系统进行相应修改的难易程度。一个易于维护的软件系统也是一个易理解、易测试和易修改的软件，以便纠正或增加新的功能，或允许在不同软件环境上进行操作。

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

(2)是在达到以上 6 个基本的软件质量要素后，面向对象软件还应重点放在类的分析上，面向对象软件质量的度量在

基金项目：河南省教育厅基金资助项目(2003520264)
作者简介：冯素云(1983—)，女，硕士生，主研方向：生物医学工程，软件工程；侯惠芳，副教授、博士生
收稿日期：2006-02-03 **E-mail：**houhf72@163.com

对类的分析过程中，应采用多个视角分析一个类的特点，即从类的耦合、内聚度、继承性、复杂度等几个方面考虑。

2 问题的分析方法与评价

本文针对面向对象的类的耦合、内聚度、继承性、复杂度的特点，通过利用 Java 语言开发的面向对象软件度量工具对 18 个小型 Java 语言做成的系统进行度量，同时部分采用了数学分析方法对一些度量指标对软件质量造成的影响加以分析，并给出了具体的改进措施。

2.1 基于耦合的分析

如果一个类作用于其他类，比如一个类的方法使用了另一个类的方法或实例，称为耦合，耦合分为几下几种：

(1)通过继承性带来的耦合：继承性有利于重用，但以为没有严格规定从超类中继承属性的方式，有可能破坏封装性和数据隐藏。

(2)通过消息传递带来的耦合 MPC：MPC 是类中对象间传递消息的数量，通常用来测类中消息传递的复杂性。MPC 表示在一个类中，方法的实现对其它类的依赖性，但没包含类接收消息的数量。如果 MPC 较大，说明其它类的耦合越强；类的响应 RFC 不仅计算了类之间的消息传递，而且对于类的实现方法中，调用类本身方法的情况也加以考虑，因此一般情况下 MPC 的值要小于 RFC 的值。在系统的设计实现过程中 RFC 的值越大，说明是系统中起重要作用的类。

(3)类之间的耦合 CBO 是全面考核类与类之间的各种耦合，表示类 C 使用其它类中属性和方法的数量。CBO’表示类 C 使用除祖先外的其它类中属性和方法的数量，即是 CBO 中除去因继承引起的耦合。类与类之间的非继承耦合比继承耦合更可能解释为软件体系结构的一种交互，因此当 CBO 与 CBO’的数值越低，则表明系统类一级的分解适合，且复杂度控制性越好，越易于进行系统的部件化开发，维护。

由系统中部分类的 CBO、CBO'的值对比可发现二者的值都呈现正态分布 N（μ，σ^2）。其中μ，σ为常数。对比结果见表 1。得到如下评价：

(1)CBO 和 CBO’的值太大，不利于模块化设计和阻碍软件的复用。CBO 和 CBO’的值越低，表明该类影响到的类越少，独立性越强，则修改是所涉及的类也越少，维护的代价越小，类的质量越好，所以一个设计良好的系统应该避免较大的 CBO 值。

改进措施：查看 CBO 较高的类，检查与该类相关的设计，尽量改进设计降低 CBO 的数值；对于 CBO 值高，且 CBO’值也高的类，可以配合继承度量的情况，检查该类的继承层次。

(2)RPC 的值越高，意味着类的复杂度很高，且可理解性、可维护性很低，因此 RPC、MPC 的数值越低，程序分解合理，不易出错，模块质量好。

改进措施：对于 MPC 和 RPC 数值较高的类。需要结合该方法的复杂度，对复杂度超过标准的，必须进行分解，以便于软件的实现和测试。

表 1　CBO、CBO’在部分类的值对比

CBO	CBO’
5	5
15	15
30	30
10	10

2.2 基于内聚的分析

内聚度量是一个模块内部各成分之间相互关联的程度，内聚的度量应分为：(1)类的内聚(通常用 LCOM)；(2)方法的内聚；(3)继承内聚。部分类的 LCOM1 和 LCOM2 的对比结果见表 2。

表 2　LCOM1 与 LCOM2 的对比结果

LCOM1	LCOM2
5	0
10	2
30	15
90	60
50	20

在面向对象软件的质量度量中采用 LCOM1、LCOM2 指标衡量软件设计中类的内聚度。由于 LCOM1、LCOM2 是直接度量类内聚缺乏的程度，值越大其内聚度反而不好。LCOM1 是类方法间通过使用相同属性增强内聚性的。LCOM2 是在方法 M1 和方法 M2 间的相似程度为 0 的方法对个数减去方法 M1 和方法 M2 间的相似程度不为 0 的方法类个数，因此相似方法越多，类的聚合度越高，LCOM2 的值反映了方法间相对不同程度。

概率论中，一元线性回归是考虑 y 为线性函数时的情况，y=a + b x。假设对于 x (在某个区间内)的每一个值有

$$y \sim N(a + b\,x, \sigma^2) \quad (1)$$

其中，a，b 及σ^2，都是不依赖于 x 的未知参数。相当于设

$$y = a + b\,x + \varepsilon \sim N(0, \sigma^2) \quad (2)$$

式(2)称为一元线性回归模型。

如果从样本中可以得到 a，b 的估计 a，b，就可以获得 y 的估计

$$\hat{y} = \hat{a} + \hat{b}\,x \ldots \quad (3)$$

y 关于 x 的线性回归方程或者回归方程，其图形称为回归直线。需要强调的是线性回归不仅适用于线性关系，通过变量代换可进行其他关系的估计。如对 y=a + b sin(t)关系模型(y 是 x 的函数)，可用 x 替换 sin(t)，获得 x 对 y 的估计后，再用 sin(t)替换 x 获得。

通过表 2，可以大致看出 LCOM1、LCOM2 在部分类的值之间呈现一元线形回归关系即式(1)，其中 y 为 LCOM1 的值，x 为 LCOM2 的值。

所以得出评价：LCOM 值大的，类的设计质量下降。

改进措施：LCOM 值越大，说明方法之间的聚合度不是很高，类的封装性比较差，应分析是否可能存在通过微小改动而产生不相关的方法集合和属性集合，如果存在就应该进行正确类的分解。

2.3 基于继承的分析

继承度指标中继承树的深度(DIT)、类到叶子的深度(CLD)是有关继承树纵向深度的度量。特别当一个类的 CLD=0 时，表示一个叶子类，当一个类的 DIT 为 0 时，表示一个根类。特别地 DIT、CLD 二者的值在系统类中呈现均匀分布，即不随类的改变而改变，可以通过表 3 的值及二者合成图观察出其规律性。

表 3　DIT 和 CLD 对比结果

DIT	CLD
1	2
1	2
1	2
1	2
1	2

评价：DIT+CLD 的值越小，则该类的抽象级越高，值越高，表明该类来自于继承的方法可能越多，其设计越复杂，类的实现质量越难以保证。

改进措施：对于 DIT 值高的类，要控制该类在实现中的

用法，避免由于继承深度大，造成方法多，从而引起误用；对于 CLD 值高的类，需要多加确认和测试，减少对其的修改。

2.4 基于复杂度的分析

每个类的方法的权重和 WMC，是传统基于模块化、结构化设计度量方法在面向对象度量的应用，是对类的整体复杂度的衡量。方法的数量和它们的复杂性是对实现和测试类所需要的工作量的合理指标，方法数量越多，继承树也越复杂。类是面向对象抽象的结果，对于一个功能性很强的类，在实现过程中可以有几个不同的类分别完成各项任务，在有一个类将它们联合起来，这样每一个类的实现复杂度都不高，而且易于实现和测试。当把一个系统当作一种对象时，是没有理由因为它是一个对象，就把它实现为一个单独的类，因此任何类的实现复杂度都不应该太大，从程序的实现中容易出错、可维护性、可理解性等质量特性来看都是这样。

评价：为 WMC 设定一个警戒值，当 WMC 大于警戒值时，类的设计和实现质量降低，需要检视和修订。

改进措施：对于方法高于警戒值的类，需要考虑其方法的复杂度情况，是否存在复杂度特别高的方法，如果存在则分解该方法；如果每个方法的实现复杂度并不高，则参考类的内聚度，查看是否需要分解该类。

3 总结

面向对象程序设计是以类为基本单位的，所以则应计算与一个类有耦合关系的类的个数，类的内聚度，类的代码行数以及类的复杂度。目前面向对象软件度量的研究现状与面向对象的分析、设计技术以及程序设计语言的研究相比尚显薄弱，软件度量学理论还缺少坚实的理论基础。在此基础上提出的度量技术及结果的评价还不够精确。寻找新的和更为有效的度量指标，来确认已知度量的有效性和度量的形式化。软件度量的理论基础至关重要，数值分布和回归分析等数学理论对面向对象软件质量度量结果的分析与评估有很大的帮助。因此，应在面向对象软件度量的数学理论基础上，进一步提高我们的软件度量水平，使面向对象软件度量的结果分析与评价更为准确、可靠。

参考文献

1 Chidamber R, Kemerer F. A Metrics Suite for Object-oriented Design[J]. IEEE Trans. on Eng., 1994, 20(6).

2 Chen J Y , Liu J F. A New Metric for Object-oriented Design[J]. Information and Software Technology, 1993, 35(4).

3 Weyuker E. Evaluating Software Complexity Measures[J]. IEEE Trans. on Software Eng., 1988, 14(9):1357-1365.

4 中国软件协会. 软件质量及其评价技术[M]. 北京: 清华大学出版社, 1990.

5 王振宇. 程序复杂性度量[M]. 北京: 国防工业出版社,1997.

6 弓惠生. 面向对象设计中软件度量学的进展[J]. 计算机科学, 1996, 36(3).

7 李　新. 一个面向对象软件度量工具的实现与度量使用研究[J]. 计算机学报, 2000, 23(11).

8 侯惠芳, 彭成寒. 面向对象软件度量工具的设计实现[J]. 计算机工程与设计, 2005, 26(6).

（上接第 19 页）

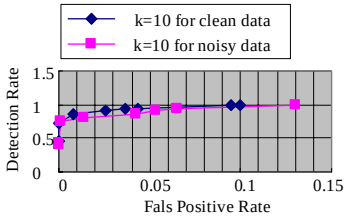

图 4 无干扰和有干扰情况的 ROC 曲线

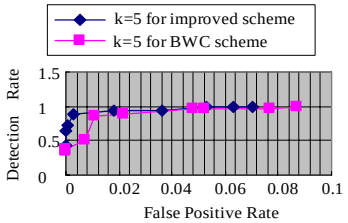

图 5 本模型和 BWC 模型抗干扰能力比较

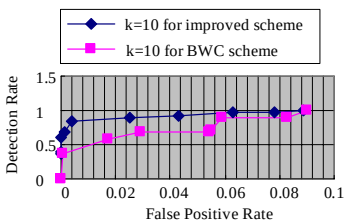

图 6 本模型和 BWC 模型抗干扰能力比较

5 结论

本文将系统调用的顺序特性和频度特性相结合，构建了一种新的入侵检测模型，并提出了一种改进的相似度计算方法，从而，在应用 kNN 算法时，提高了相似度的判定精度。实验表明，该系统不仅可以提高入侵检测的准确度，降低误报率，而且还可以提高系统的抗噪声干扰能力。

参考文献

1 Liao Y, Vemuri V R. Use of K-Nearest Neighbor Classifier for Intrusion Detection[J]. Computers & Security, 2002, 21(5).

2 Sanjay R, Pujari A K, Gulati V P, et al. Intrusion Detection Using Text Processing Techniques with a Binary-weighted Cosine Metric[Z]. http://www.cs.ucdavis.edu/~vemuri/publications.htm.

3 Ye N, Li X, Chen Q, et al. Probabilistic Techniques for Intrusion Detection Based on Computer Audit Data[J]. IEEE Trans. SMC-A, 2001, 31(4): 266-274.

4 Forrest S, Hofmeyr S A, Somayaji A, et al. A Sense of Self for Unix Processes[C]. Proceedings of the IEEE Symposium on Security and Privacy, Los Alamitos, CA, 1996.

5 Lee W, Stolfo S, Chan P. Learning Patterns from Unix Process Execution Traces from Intrusion Detection[C]. AAAI Workshop: AI Approaches to Fraud Detection and Risk Management, 1997-06.

6 DARPA Data Set[Z]. http://www.ll.mit.edu/IST-ideval data 1998/1998_data_index.html, 1998.