一种优化的面向对象软件复杂性度量方法

谢超超，杨 柳
（中南大学 软件学院，湖南 长沙 410000）

摘 要：首先结合面向对象技术特性，对面向对象软件类级别的CK度量方法和系统级别的MOOD度量方法进行了分析，并就CK度量提出了优化的度量方法OCK。然后结合OCK度量和MOOD度量的优点提出了较优化的面向对象软件复杂性度量方法OSCM。OSCM度量可以有效地弥补CK度量和MOOD度量的不足，优化度量结果。

关键词：面向对象；软件度量；CK度量；MOOD度量；OCK度量；OSCM度量

中图分类号：TP311.5 **文献标识码**：A **文章编号**：1674-7720(2013)21-0062-03

An optimized measurement method of object-oriented software complexity

Xie Chaochao, Yang Liu
(School of Software, Central South University, Changsha 410000, China)

Abstract: Combining the object-oriented technology characteristics, this article analyzed object-oriented software class-level CK metrics and system-level MOOD metrics. In addition, it proposed OCK. Then, combining the advantages of OCK and MOOD, an optimized measurement method of object-oriented software complexity, which is called OSCM, was proposed. OSCM metrics can supplement CK metrics and MOOD metrics, and optimize the result of measurement.

Key words: object-oriented; software metrics; CK metrics; MOOD metrics; OCK metrics; OSCM metrics

软件的应用领域越来越广，软件的质量也越来越受到关注和重视。软件复杂性很大程度上影响到软件质量的好坏，其度量是软件度量的重要方面。随着面向对象软件技术的广泛应用，面向对象软件复杂性度量也显得尤为重要。面向对象度量的基本目标[1]和已存在的传统软件度量的目标一致：即更好地理解产品的质量，评估过程的效果，从而控制开发过程，以提高软件质量。

当前，已有很多面向对象软件度量方法被提出，并在不断被验证及成熟。这些度量方法包括LK度量[2]、CK度量[3]、Li度量[4]和MOOD度量[5]等。但是这些度量方法依然存在缺陷，需要不断进行研究和改进，以使这些度量更易于应用，从而更好地指导面向对象软件的设计、开发，提高软件质量[6]。

本文分析了针对类层面的CK度量和针对系统层面的MOOD度量。这两种度量都只是分散地针对软件的某一个特定层面进行，在实际应用过程中，难以让人们同时综合类与类之间关系和系统级别这两个层次的度量，致使无法更加系统全面地掌握软件系统的复杂度。因此本文在分析CK度量和MOOD度量的基础上，对CK度量进行改进，提出优化的度量方法OCK（Optimized CK），并与MOOD度量相结合，提出了一个较为优化的面向对象软件复杂性度量方法OSCM（Optimized Software Complexity Metrics），以方便人们更快捷、有效地分析面向对象软件系统的复杂度。

1 面向对象软件度量方法

自20世纪90年代以来，面向对象技术兴起并被广泛应用起来，人们逐渐开始研究有关面向对象软件的度量[7]。不断有面向对象软件度量方法被提出，目前主要的面向对象软件度量方法有：LK度量、CK度量、Li度量以及MOOD度量等。本文介绍面向对象软件复杂性度量：CK度量和MOOD度量。

1.1 CK度量

Chidamber和Kemerer等人于1994年提出的CK度量[3]，是目前使用最为广泛的度量体系之一，是面向对象软件类级别度量方法，其中包括6条适用于面向对象设计的度量准则[8]。

（1）每类加权方法数 WMC（Weighted Method per Class）。WMC 是一个类方法复杂度的加权总和。类 WMC 越大，对子类的可能影响越大，但其通用性和可复用性越差。

（2）继承树深度 DIT（Depth of Inheritance Tree）。DIT 指从本类节点到根节点的继承树中路径的最大深度，根节点值为 0，以下各级依次递增。DIT 值越大，则其可能继承方法数越多，复用程度越高，但预测其行为将更困难，同时设计越复杂。

（3）每类孩子数 NoC（Number of Children）。NOC 是继承树中一个类的直接孩子数。NOC 越大，重用性越好，但其父类抽象性减弱，测试越困难。

（4）对象类之间耦合度 CBO（Coupling Between Object Classes）。一个类的 CBO 指的是和其有耦合关系的类的数目。CBO 越大，则类的可重用性越弱，且修改和测试越复杂。

（5）类响应 RFC（Response For a Class）。RFC 是本类方法数加上被本类方法调用的方法的个数总和。RFC 越大，类越复杂，且对该类进行测试和调试也越困难。

（6）方法内聚缺乏度 LCOM（Lack of Cohesion in Methods）。LCOM 是相似度为零的方法对数量减去相似度不为零的方法对数量，相似度是两个方法访问相同属性的程度。类的 LCOM 越大，方法内聚度越弱，则类可以分解为两个或更多的子类。

1.2 MOOD 度量

MOOD 度量是另一个著名的度量体系，是由 Abreu 等人于 1994 年针对软件系统层次提出的[5]。MOOD 度量从面向对象的封装性、继承性、耦合性和多态性 4 个方面给出面向对象软件 6 个度量指标。

（1）封装性度量。封装性由类中的属性和方法实现，因此封装性通过属性隐藏因子 AHF（Attribute Hiding Factor）和方法隐藏因子 MHF（Method Hiding Factor）表示系统中所有类的属性和方法的隐藏程度。隐藏因子的值越大，系统中信息隐藏得越好。

（2）继承性度量。继承性通过属性继承因子 AIF（Attribute Inheritance Factor）和方法继承因子 MIF（Method Inheritance Factor）表示系统中所有类的属性和方法的继承程度。继承因子的值越大，系统中信息继承的程度越高。

（3）耦合性度量。耦合性通过耦合因子 CF（Coupling Factor）表示系统中所有类之间的耦合程度，但不将继承关系考虑进去。CF 越大，类之间耦合越频繁。

（4）多态性度量。多态性通过多态因子 PF（Polymorphism Factor）表示系统中所有类方法使用多态机制的程度。

2 面向对象软件复杂性度量方法

面向对象软件复杂性度量方法虽已得到发展和完善，但依旧存在一定的缺陷。首先分析 CK 度量的不足，并在 CK 度量的基础上提出改进的度量方法 OCK。然后结合 OCK 度量和 MOOD 度量的优点从而提出较为优化的面向对象软件复杂性度量 OSCM。

2.1 CK 度量的分析与改进

（1）WMC 只考虑方法成员，没有考虑属性成员对类复杂性的影响[9]，也没有根据类成员可见性的不同区别看待各成员对类复杂性的影响，类公有成员，保护成员和私有成员各自对类复杂性影响程度大小不同。因此，在 WMC 的基础上提出类的复杂性 CPC（Complexity Per Class）度量指标。

定义 $CPC=aWAC+bWMC$，$WAC=cWAC_u+dWAC_o+eWAC_v$，$WMC=cWMC_u+dWMC_o+eWMC_v$

式中 a、b、c、d、e 为调节因子，a、b 可以适当地反映出属性和方法对类复杂性的影响程度不一样；c、d、e 则可以反映出公有成员，保护成员和私有成员对类复杂性的不同影响程度，这些值是通过大量的类复杂性度量实践获取的经验值，且 $a<b$，$c>d>e$。WAC（Weighted Attributes per Class）是每类加权属性数，WAC_u 是加权公有属性数，WAC_o 是加权保护属性数，WAC_v 是加权私有属性数，WMC_u 是加权公有方法数，WMC_o 是加权保护方法数，WMC_v 是加权私有方法数。CPC 值越大，类越复杂，其通用性和可复用性越差。

（2）DIT 无法度量多重继承，遇到多重继承时会出现歧义[6]。而且当考虑多重继承时，仅根据继承树深度难以全面地判断类继承复杂性，还需要结合类的祖先类个数来进行分析，故提出多重继承树复杂性 CMIT（Complexity of Multi-Inheritance Tree）度量指标。

定义 $CMIT=a\mathrm{DMIT}+b\mathrm{NOF}$

$$\mathrm{DMIT}=\max(\mathrm{DMIT}_i)+1,\ \mathrm{NOF}=\sum_{i=1}^{n}(\mathrm{NOF}_i+1)$$

式中 a、b 为调节因子，可以表示多重继承树深度 DMIT（Depth of Multi-Inheritance Tree）和祖先类数目 NOF（Number Of Fathers）对 CMIT 值的影响程度大小。DMIT_i 是指本类的每个父类的 DMIT 值，继承树根节点的 DMIT 值为 0，max 为取最大值函数。NOF_i 是指本类的每个父类的 NOF 值，本类中没有任何父类时，其 NOF 值为 0。CMIT 综合考虑了多重继承树深度和祖先类数目两个因子，其值越大，则复用程度越高，但不可预测性也越高。

（3）NOC 只计算了直接继承的孩子数，并未将间接继承的孩子数计算进来。因此可定义每个类的所有子孙数 NOAC（Number Of All Children）用来计算所有子孙数目。

定义 $\mathrm{NOAC}=\sum_{i=1}^{n}(\mathrm{NOAC}_i+1)$

式中，NOAC_i 是指本类的每个直接孩子类的 NOAC 值。当没有直接孩子类时，其 NOAC 值为 0。

(4)CBO 只是计算和本类耦合的类的数目，没有对不同类型耦合的强度进行区分，而是假设所有的耦合关系强度是相同的，而且忽略了最强的耦合关系即继承耦合。为了考虑到不同的耦合关系，包括关联、继承和实现耦合，提出了类型间所有耦合 ACBT（All Coupling Between Type）度量指标。

定义 ACBT=aNAC+bNIC+cNRC

式中 a、b、c 为调节因子，是度量实践中获取的经验值，可反映出关联耦合、继承耦合和实现耦合在 ACBT 计算中的权值。关联耦合数 NAC（Number of Association Coupling）是系统中所有被本类关联的类型（类或接口）的数目，继承耦合数 NIC（Number of Inheritance Coupling）是系统中类型（类或接口）所继承的所有类型（类或接口）的数目，实现耦合数 NRC（Number of Realization Coupling）是系统中本类所实现的所有接口的数目。ACBT 越大，类的可重用性可能越弱。

(5)RFC 没有考虑本类方法和被本类方法调用的方法分别对类复杂性的影响程度，只是简单地计算它们的总和。因此，提出类加权响应 WRFC（Weighted Response For a Class）度量指标。

定义 WRFC=aNMC+bNCMC

式中 a、b 为调节因子，是度量实践中获取的经验值，可表现出本类方法和被本类方法调用的方法对类加权响应值的影响程度的大小。NMC（Number of Methods a Class）是本类方法数，NCMC（Number of Called Methods a Class）是被本类方法调用的方法数。

(6)LCOM 存在一定的缺陷，没有将类的实例变量数计算进来，但实际上类的实例变量数对类内聚性有一定的影响。假设一个类的 LCOM 较小，甚至为 0，根据 CK 可以推断出该类具有较好的内聚性，但实际上可能因为该类拥有大量的实例变量，所以其内聚性及封装性不容乐观。因此，在 LCOM 的基础上提出类内聚缺乏度 LCOC（Lack of Cohesion in Class）度量指标。

定义 $LCOC=LCOM\sum_{j=1}^{n}|I_j|$

LCOM 同 CK 度量里定义的 LCOM。假设一个类有 n 个方法 $M_1,\cdots,M_n$，I_j 是方法 M_j 中实例变量的集合，$|I_j|$ 为计算 I_j 集合中的元素个数。LCOC 值越大，则类的内聚性及封装性越弱。

根据上述对 CK 度量方法的分析与改进，提出优化的 CK 度量方法 OCK，其包括 CPC、CMIT、NOAC、ACBT、WRFC、LCOC 等 6 个相对应的度量指标，每个度量指标针对性地弥补每点缺陷。例如，采用 OCK 度量的 CPC 能综合考虑一个类的属性和方法以及不同可见性成员对该类复杂性的影响；CMIT 度量指标解决了多重继承的度量问题，消除歧义；NOAC 值涵盖了一个类的所有子孙；ACBT 依据耦合类型的不同分配不同的权值从而度量类的耦合性等等。

2.2 面向对象软件复杂性度量方法 OSCM

OCK 度量方法虽在 CK 度量的基础上进行了改进，但仍存在一些缺陷。首先 OCK 度量没有对多态性进行度量，多态对整个系统的复杂性有很大影响；其次 OCK 与 CK 度量一样主要是针对类层面，在系统层面没有很好的度量指标。MOOD 度量也存在一定的不足。比如，MOOD 度量没有对类方法和属性以及类之间的关系进行研究，也没有完整定义抽象性和复杂性[10]。

此外，继承性和耦合性也是面向对象的基本特性，将软件类层面和系统层面的继承性、耦合性度量结合起来可以更好地反映软件整体的复杂度。但单独的 OCK 度量和 MOOD 度量都只考虑了继承和耦合的单个方面的复杂性，不能可靠地对软件进行整体评价。因此针对这两种度量方法进行改进，补充每个度量方法缺少的方面，从而提出新的面向对象软件复杂性度量方法 OSCM 如下，与 OCK 度量和 MOOD 度量的比较如表 1 所示。

表 1 OCK、MOOD、OSCM 度量汇总

面向对象的特性	OCK	MOOD	OSCM
类复杂性	CPC		CPC
继承性	CMIT	AIF	AIF
	NOAC	MIF	MIF
			CMIT
			NOAC
封装性	LCOC	AHF	AHF
		MHF	MHF
			LCOC
多态性		PF	PF
耦合性	ACBT	CF	CF
	WRFC		ACBT
			WRFC

度量 1 使用 CPC 度量指标对类复杂性进行度量。

度量 2 使用 AIF，MIF，CMIT 和 NOAC 等 4 个度量指标对继承性进行度量。既有对系统级别继承性的度量，又有对类级别继承性的度量。

度量 3 使用 AHF，MHF 和 LCOC 等 3 个度量指标对封装性进行度量。既有对系统级别封装性的度量又有对类级别封装性的度量。

度量 4 使用 PF 度量指标对多态性进行度量。

度量 5 使用 CF，ACBT 和 WRFC 等 3 个度量指标对耦合性进行度量。CF 的度量粒度不够细[6]，加上类级别的 ACBT 和 WRFC 就能更好得衡量整体耦合性。

上述 5 条度量法则即为改进提出的 OSCM 度量方法。OSCM 度量综合考虑了 OCK 度量和 MOOD 度量的优缺点，可同时针对软件类级别和系统级别进行度量，有效地弥补 CK 度量和 MOOD 度量的不足。

本文在 CK 度量和 MOOD 度量的基础上，对 CK 度

量进行改进完善，提出OCK度量方法。再集成OCK度量和MOOD度量的优点，提出面向对象软件复杂性度量OSCM。OSCM可较全面、可靠地对面向对象软件复杂性进行度量，优化度量结果。

参考文献

[1] PRESSMAN R S.软件工程实践者的研究方法[M].梅宏译.北京：机械工业出版社，2002.

[2] LORENZ M，KIDD J. Object-Oriented Software Metrics：A Practical Guide[M]. New Jersey：Prentice-Hall，1994.

[3] CHIDAMBER S R，KEMERER C F. A metrics suite for object oriented design [J]. IEEE Transaction on Software Engineering，1994，20(6)：476-493.

[4] Li Wei. Another metric suite for object -oriented programming[J]. Journal of Systems and Software，1998，44(2)：155-162.

[5] ABREU F B. MOOD-metrics for object-oriented design[C]. Proc of the 9th Annual Conference on Object -Oriented Programming Systems，Languages，and Applications. New York：ACM Press，1994.

[6] 张伟.面向对象软件复杂性度量研究[D].武汉：武汉理工大学，2007.

[7] 吴光金. 面向对象软件复杂性度量方法的研究 [D].重庆：重庆大学，2008.

[8] 伦立军，丁雪梅，李英梅.面向对象软件度量技术研究[J].计算机应用研究，2002，19(12)：40-42.

[9] 马志新，徐德启，杜伟杰.面向对象软件度量C&K方法的研究与改进 [J]. 电子科技大学学报，2006，35(3)：396-398.

[10] 李大鹏，郭平，陈新宇.一种集成类层次和系统层次的面向对象软件复杂性度量集 [J]. 计算机研究与发展(增刊)，2010，47：237-242.

（收稿日期：2013-07-07）

作者简介：

谢超超，男，1992年生，软件设计师，主要研究方向：软件工程、软件度量。

杨柳，女，1979年生，博士，讲师，主要研究方向：软件度量、语义Web。