混源软件质量模型与度量方法研究

刘启林[1] 董 威[1] 尹良泽[1] 齐 璇[2] 杨沙洲[1]
(国防科学技术大学计算机学院 长沙 410073)[1] (北京系统工程研究所 北京 100101)[2]

摘 要 由于混源软件包含自主代码、开源代码等不同来源代码,从而具有更高的多样性和复杂性,对其质量的度量评估与传统软件存在极大区别。为了度量混源软件质量,建立混源软件质量度量模型和方法是非常必要的。通过分析混源软件质量特性,提出混源软件质量模型。然后利用层次分析法、幂性法及线性法构建度量方法体系。最后对UbuntuKylin操作系统进行了实验性的度量评估,验证了模型与方法的可行性和有效性。

关键词 混源软件,质量模型,软件度量,自主可控,度量方法
中图法分类号 TP311.5 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.04.018

Research on Mixed Source Software Quality Model and Measurement Method

LIU Qi-lin[1] DONG Wei[1] YIN Liang-ze[1] QI Xuan[2] YANG Sha-zhou[1]
(School of Computer, National University of Defense Technology, Changsha 410073, China)[1]
(Beijing Institute of System Engineering, Beijing 100101, China)[2]

Abstract Because the mixed source software(MSS) contains independent code, open source code and other different codes, the MSS has higher diversity and complexity, and quality measurement is greatly different between traditional software. In order to measure the MSS, it is very essential to establish measurement model and method. Based on MSS quality property, this paper put forward mixed source software quality model. Then we used the analytic hierarchy process, power method and linear method to build measurement method system. Finally we performed an experimental measurement on UbuntuKylin operating system. The experimental results shows the effectiveness, feasibility of the model and method.

Keywords Mixed source software, Quality model, Software measurement, Independence and controllability, Measure methods

1 引言

随着开源软件的迅猛发展,越来越多开源软件与闭源软件进行紧密的交互,跨平台的开源环境与Linux和Windows的衔接也越来越多,使得混源软件广泛应用到各领域,形成了一个混源时代。混源软件是伴随着开源软件的产生而产生的,其是指兼有闭源代码和开源代码的混合体。

软件质量是指在特定条件下使用软件产品满足明确或隐含需求的程度[1]。国外对软件质量进行了很多有益的探索,并形成了一些相对成熟的理论成果。ISO/IEC 25010产品质量模型(Product Quality Model)[1]对闭源软件的质量属性考虑得非常全面,但缺乏对开源软件开发过程、社区、演化等的度量,不能反应开源软件质量特性。OpenBRR(Open Business Readiness Rating)模型、QualOSS(Quality of Open Source Software)模型和QSOS(Qualification and Selection of Open Source software)模型[2-3]的结构严谨,度量指标完整,度量方法完备,但度量过程复杂,度量的可操作性差。OMM(QualiPSo Opensource Maturity Model)模型[4]结构完整、评估充分详细,很好地反映了开源过程开发和持续管理维护的特性,但整个评估过程复杂,而且OMM模型偏向开发过程的评估,忽视对软件产品本身特性的评估。此外,这些开源软件度量模型并未过多考虑加入自主代码后的一些特征,并不能直接用于度量混源软件质量,且很多混源软件对自主可控的要求很高。

国内在软件度量领域也做出了一些有益的贡献,一些适合于相关应用的模型被提出。CSIP开源软件质量模型[5]只针对开源软件,度量评估单一,不适用于混源软件。还有一些面向特定领域的模型(如一些面向军事、航天、金融、商务等领域的具体应用模型)面向特殊需求,无法兼顾通用需求。

自主可控(Independence and Controllability, IC)是指软件产品由自己主导设计、研制、生产、维护,关键技术不受制于他人,能够保证产品的安全性、使用的可靠性、供货的稳定性。由于开源软件受到一些许可协议或者开源商业公司的附加条件制约,无法参与自主软件的演化[6],这给自主可控带来了巨大挑战,严重影响了混源软件质量,自主可控已经成为衡量软

到稿日期:2015-11-30 返修日期:2016-02-26
刘启林(1988—),男,硕士,主要研究领域为软件度量、软件质量工程,E-mail:chilinliu@163.com;董 威(1976—),男,博士,教授,主要研究领域为软件工程、软件分析与验证;尹良泽(1986—),男,博士,讲师,主要研究领域为程序分析与验证、代码质量保证;齐 璇(1973—),女,博士,副研究员,主要研究领域为软件工程;杨沙洲(1975—),男,博士,副研究员,主要研究领域为操作系统。

件质量的一个重要指标。

总之,一方面,现有的软件度量模型都只单纯地考虑闭源或开源软件,将二者孤立起来则无法适用于混源软件;另一方面,当前度量方法都是只针对传统软件、开源软件或者面向某个特殊领域的软件的度量,没有与混源软件质量度量相关的方法,而且通常度量都需固定要素和权重,对度量的限制很大,影响了度量的实际效果。

本文首先通过广泛的调查研究,提取各种软件的质量特性,在此基础上建立混源软件质量模型;然后,以混源软件质量模型为基础,设计度量算法,并实现度量工具;最后,对 UbuntuKylin 操作系统进行研究性验证。具体研究框架如图 1 所示。

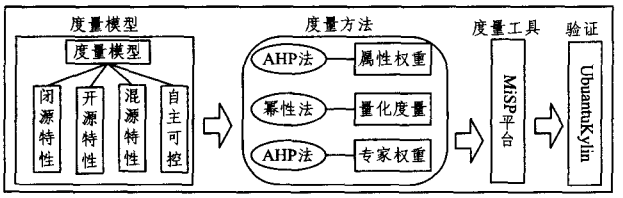

图 1　研究框架

2　混源软件质量模型

在混源时代,企业主导着混源软件的开发与应用,决定着混源软件的发展演化,其规模、技术实力与混源软件的质量有一定相关性。混源软件开发与应用会带来开源软件授权、软件产权问题,合理地选择合适的开源许可有利于提升混源软件的质量。混合源代码本身的可靠性、可信性是混源软件质量的基础,且如何关联、组织、管理大量多源异构的代码(如自主代码、开源代码、库代码等)将制约混源软件质量。混源软件对互操作性、适配性提出更高的要求,影响着混源软件质量。不同代码的更新演化反映混源软件的持续性和扩展性,说明混源软件不是静态产品,而是永远演化的。

根据"目标—质量属性—质量子属性"的 3 层结构设计模型,定义混源软件质量模型(Mixed Source Software Quality Model,MSQ),具体如图 2 所示。

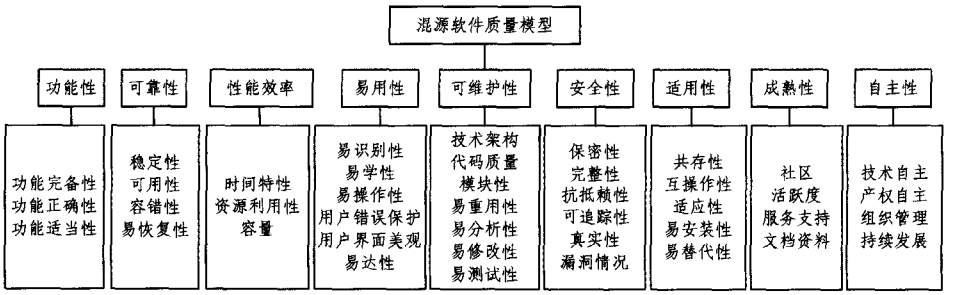

图 2　混源软件质量模型

对于功能性、可靠性、性能效率、易用性、可维护性和安全性,主要参考了 ISO/IEC 25010 产品质量模型。其中可维护性下的技术架构是指软件技术和架构的合理性、成熟性、可扩展性的程度;代码质量是指代码的可信、规范程度。技术架构和代码质量从源代码层面反映软件的质量。安全性下的漏洞情况是指一段时间内发现漏洞与修复漏洞的情况,突出开源软件频繁发布修复漏洞这一特性。下面主要对与传统模型差异较大的几个特性进行说明。

(1)适用性,指软件所能适用的范围,反映软件的兼容性和可移植性程度。主要从共存性、互操作性、适应性、易安装性和易替代性等方面衡量软件的适用性。有些软件直接跟设备绑定,对兼容性和可移植性不做要求,适用性可以根据使用环境进行有针对性的度量评估。

(2)成熟性,指软件产品的应用支持程度。社区主要反映社区的大小、社区的参与模式、社区组织管理等方面;活跃度主要从版本发布周期、软件下载频率、交流活动情况等方面进行评估;服务支持主要包括企业支持和应用程度;文档资料主要包括软件文档、书籍和网络资源的丰富程度。成熟性从侧面反映了一个软件产品的质量,是软件质量中一个不可或缺的部分。

(3)自主性,指软件自主可控程度。主要从技术自主、产权自主、组织管理和持续发展等 4 个方面衡量。技术自主度是指软件的技术自主化程度;产权自主度指软件版权自主化程度;组织管理是指软件运行、维护、管理的自主程度;持续发展主要度量其未来发展、演化的自主程度。

MSQ 模型统一了开源与闭源特性,突出自主可控。模型可以针对闭源、开源分别定制,或者针对嵌入式、非嵌入式等分别定制,亦可针对典型的行业应用定制模型。下面的度量方法灵活支持定制后的质量评估。MSQ 模型综合考虑了混源软件的各种质量特性,满足了度量的充分性。根据使用环境定制度量模型,能满足差异化需求,增强了度量的针对性和灵活性,提高了模型的实用性。

3　混源软件质量度量方法

混源软件质量度量的难点主要有:属性权重的设置、属性评分的整合、多个度量人权重的确定。针对上面的困难,利用 AHP 方法获取属性权重和度量人权重,利用幂性法计算整合属性评分,用线性法复合度量人权重,具体方法如下。

3.1　AHP 方法

层次分析法[7](Analytic Hierarchy Process,AHP)是美国运筹学家 T. L. Saaty 于 1971 年提出的一种将定性与定量分析相结合的决策方法。层次分析法的核心思想是:把复杂的问题分解成一个层次结构,对每一层次的要素进行定量的相对重要程度的判断,再用数学方法计算出相对权重,并进行检验,以验证其合理性。具体计算如下。

(1)假设第 $k-1$ 层选择了 n 个质量属性,根据各质量属性的重要程度,进行两两比较,得到一个 $n\times n$ 的判断矩阵:

$$A=\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

(2)计算判断矩阵每行所有元素的几何平均值：

$$\overline{\omega_i}=\sqrt[n]{\prod_{j=1}^{n}a_{ij}},\ i=1,2,\cdots,n \tag{1}$$

得

$$\bar{\omega}=(\overline{\omega_1},\overline{\omega_2},\cdots,\overline{\omega_n})^{\mathrm{T}}$$

(3)将$\overline{\omega_i}$归一化，即计算：

$$\omega_i=\frac{\overline{\omega_i}}{\sum_{j=1}^{n}\overline{\omega_j}}$$

得

$$\bar{\omega}=(\omega_1,\omega_2,\cdots,\omega_n)^{\mathrm{T}}$$

即为所求特征向量的近似值，$\bar{\omega}$也是第$k-1$层中各质量属性关于总目标的相对权重。

由于软件的复杂性以及人的思维判断的差异性，判断会出现偏差，需要对判断矩阵的合理性进行验证以检验其一致性。接着利用层次总排序和总排序一致性检验以获取第k层各质量子属性关于总目标的相对权重。具体内容可以参考文献[7]。

用户基于AHP法可以自主地给出各属性的相对权重，对于用户而言，这比给出各属性绝对权重更易于操作，体现了算法的灵活性。

3.2 幂性法

采用AHP法获取各质量子属性的权重后，再用幂性法将子属性评分与子属性权重幂性进行复合，获得单人的度量结果。具体幂性计算模型如下：

$$\begin{cases} F=S_1^{\omega_1}S_2^{\omega_2}\cdots S_n^{\omega_n} \\ \sum_{i=1}^{n}\omega_i=1 \\ 0\leqslant\omega_i\leqslant 1 \end{cases} \tag{2}$$

其中，S_i表示各子属性得分，若S_i中有一项为0，则F为0；n表示子属性的个数；ω_i表示子属性对应的权重。

幂性计算模型满足单调性、凝聚性、灵敏性以及稳定性[8]，用幂性法完成单人度量评估量化。使用幂性法使度量评估更加科学、客观，能准确地反映度量量化特性，提高度量的有效性。

3.3 线性法

通过对多家单位、企业进行调查研究，发现在实际度量时同一个项目会有多个度量人，而且每个度量人也具有权重。由于各方利益的协调比较困难，因此需要客观、科学的方法以确定度量人的权重。首先，用AHP方法进行度量人相对重要程度的判断，获得度量人权重h_i，再用线性法复合单人度量结果F_i，得到最终的度量值。线性计算模型如下：

$$\begin{cases} G=h_1F_1+h_2F_2+\cdots+h_nF_n \\ \sum_{i=1}^{n}h_i=1 \\ 0\leqslant h_i\leqslant 1 \end{cases} \tag{3}$$

其中，F_i表示单个度量结果，n为度量人的个数，h_i表示度量人的权重。

线性复合增强了度量的稳定性和可靠性。

3.4 “层次分析－幂性－线性”度量算法体系

至此，完成了“层次分析－幂性－线性”度量算法体系设计，如图3所示。

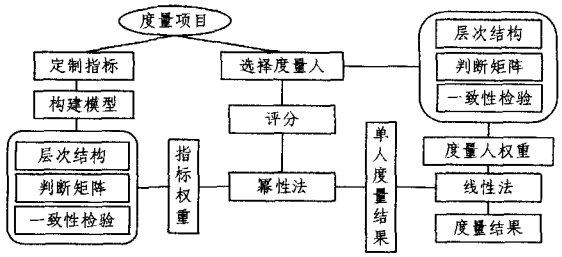

图3 “层次分析－幂性－线性”度量算法体系

本算法体系具有一定的通用性，很好地支持了多人多指标的度量评估，可以根据不同的需求和使用环境实现快速化的定制，适用于多种度量评估。根据混源软件质量模型，应用本算法体系，设计开发了混源软件质量度量平台(Mixed Source Software Quality Measurement Platform，MiSP)。

4 实验分析

度量具有极强的引导性，可以促使软件质量有针对性地改善和提高。本度量实验的指导思想是：同一个项目、同样的属性、同样的权重、不同类型的度量人进行探索性的度量实验。我们示意性地定制UbuntuKylin操作系统的质量属性体系，如图4所示。

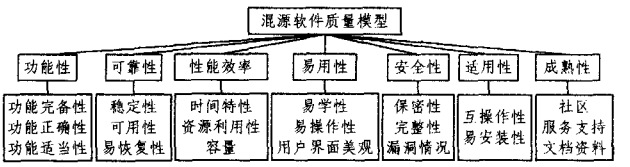

图4 UbuntuKylin质量属性体系

分别随机邀请5位用户、5位专家以及5位UbuntuKylin操作系统开发者实施度量。用户都使用过UbuntuKylin操作系统，且专家都对UbuntuKylin操作系统比较熟悉，保证了度量对象具有一定程度的认知。假设5位用户之间、5位专家之间及5位开发者之间的都是平等的，即他们之间权重是相同的。具体度量结果如表1所列。

表1 UbuntuKylin操作系统度量结果表

目标	度量类别	度量人	单个度量结果
UbuntuKylin	用户	用户A	80.404759455300
		用户B	80.752960941483
		用户C	79.964922521981
		用户D	79.470332244957
		用户E	64.783791721217
	专家	专家A	81.855921123138
		专家B	83.216793554744
		专家C	79.304492764211
		专家D	80.754639365169
		专家E	79.813027834441
	开发者	开发者A	89.502679427133
		开发者B	92.995133397400
		开发者C	89.977072219384
		开发者D	88.396464664735
		开发者E	87.960803102815

(下转第95页)

Journal of Frontiers of Computer Science and Technology, 2012, 6(2): 109-117. (in Chinese)
潘森，谭曦，彭鑫，等. 综合包级和类级度量的软件缺陷预测方法[J]. 计算机科学与探索，2012，6(2)：109-117.

[8] TU Y M, MAO J P, YU J, et al. Analysis of Software Defect Prediction Model of System Testing Process[J]. Journal of Computer Research and Development, 2010, 47(Suppl.): 108-112. (in Chinese)
涂亚明，毛军鹏，余静，等. 系统测试阶段的软件缺陷预测模型分析[J]. 计算机研究与发展，2010，47(Suppl.)：108-112.

[9] WANG P, JIN C, GE H H. Mutual information-based feature selection approach for software defect prediction[J]. Journal of Computer Applications, 2012, 32(6): 1738-1740. (in Chinese)
王培，金聪，葛贺贺. 面向软件缺陷预测的互信息属性选择方法[J]. 计算机应用，2012，32(6)：1738-1740.

[10] SHIVAJI S, WHITEHEAD E J, AKELLA R, et al. Reducing Features to Improve Bug Prediction[C]// 2009 IEEE/ACM International Conference on Automated Software Engineering. IEEE Computer Society, 2009: 600-604

[11] XU G C, LIU X Z, HU L, et al. Software reliability assessment models incorporating software defect correlation[J]. Journal of Software, 2011, 22(3): 439-450. (in Chinese)
徐高潮，刘新忠，胡亮，等. 引入关联缺陷的软件可靠性评估模型[J]. 软件学报，2011，22(3)：439-450

[12] JIANG H Y, ZONG M, LIU X Y. Research of Software Defect Prediction Model Based on ACO-SVM[J]. Chinese Journal of Computers, 2011, 34(6): 1148-1154. (in Chinese)
姜慧研，宗茂，刘相莹. 基于 ACO-SVM 的软件缺陷预测模型的研究[J]. 计算机学报，2011，34(6)：1148-1154

[13] KIM S, WHITEHEAD E J, ZHANG Y. Classifying Software Changes: Clean or Buggy? [J]. IEEE Transactions Software Engineering, 2008, 34(2): 181-196.

[14] SLIWERSKI J, ZIMMERMANN T, ZELLER A. When do changes induce fixes?[J]. ACM Sigsoft Software Engineering Notes, 2005, 30(1): 1-5.

[15] SEBASTIANI F. Machine learning in automated text categorization[J]. ACM Computing Surveys, 2002, 34(2): 1-47

[16] OSTRAND T J, WEYUKER E J, BELL R M. Where the bugs are[J]. AcmSigsoft Software Engineering Notes, 2004, 29(4): 86-96.

[17] SCOTT S, MATWIN S. Feature Engineering for Text Classification[C]// International Conference on ICML. 1999: 379—388.

[18] MOCKUS A, VOTTA L G. Identifying Reasons for Software Changes Using Historic Databases[C]// Proceedings of IEEE International Conference on Software Maintenance. 2000: 120-130

[19] CUBRANIC D, MURPHY G C. Hipikat: recommending pertinent software development artifacts[C]// Proceedings of 25th International Conference on Software Engineering. 2003: 408-418

[20] ZIMMERMANN T, NAGAPPAN N, GALL H, et al. Cross-project Defect Prediction A Large Scale Experiment on Data vs. Domain vs. Process[C]// European Software Engineering Conference. 2009: 91-100.

[21] NAM J, PAN S J, KIM S. Transfer defect learning[C]// International Conference on Software Engineering. 2013: 382-339.

(上接第 84 页)

UbuntuKylin 操作系统质量度量最终结果：基本都浮动在 80 左右，属于优良级别。该度量结果与广大用户对 UbuntuKylin 操作系统的评价相符，一定程度上反映了用户的使用体验度，也说明了模型的合理性及度量算法的有效性。

结束语　通过分析混源软件的质量特性，提出了混源软件质量模型。然后，以混源软件质量模型为基础，设计了一套实施度量的算法体系，并实现了 MiSP 度量工具。最后，对 UbuntuKylin 操作系统进行了探索性的度量实验。下一步工作将深入优化度量模型，规范度量指标，丰富度量平台。

参 考 文 献

[1] ISO/IEC. 2011 Systems and Software Quality Requirements and Evaluation: ISO/IEC 25010[S]. 2011.

[2] PETRINJA E, SILLITTI A, SUCCI G. Comparing OpenBRR, QSOS, and OMM Assessment Models[C]// Open Source Software: New Horizons. 2010: 224-238.

[3] HAALAND K, GROVEN A K, GLOTT R, et al. Free/Libre Open Source Quality Models — a comparison between two approaches[OL]// http://publications.nr.no/FLOS34workshop.final.pdf.

[4] PETRINJA E, NAMBAKAM R, SILLITTI A. Introducing the OpenSource Maturity Model[C]// Proceedings of the 2009 ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development. IEEE Computer Society, 2009: 37-41.

[5] CHEN Y, HU C J, WU T. Open Source Software Maturity Assessment-Part One[J]. Information Technology & Standardizaton, 2011(9): 62-67. (in Chinese)
陈越，胡昌军，吴桐. 开放源代码软件成熟度评估(上)[J]. 信息技术与标准，2011(9)：62-67.

[6] NI G N. Independence and Controllability is the premise to enhance security internet[J]. Business Culture, 2014(30): 42-45. (in Chinese)
倪光南. 自主可控是增强网络安全的前提[J]. 商业文化，2014，(30)：42-45.

[7] 《运筹学》教材编写组. 运筹学(第 4 版)[M]. 北京：清华大学出版社，2012：522-527.

[8] WANG J, CHEN Y X, GU B, et al. An approach to measuring and grading software trust for spacecraft software[J]. Science China Press, 2015, 45(2): 221-228. (in Chinese).
王婧，陈仪香，顾斌，等. 航天嵌入式软件可信性度量方法及应用研究[J]. 中国科学：技术科学，2015，45(2)：221-228.