软件复用度量综述

钱乐秋　张　涌

（复旦大学计算机科学系　上海 200433）

摘　要　软件复用度量是软件复用技术中不可分割的一部分，在软件复用开发中占据重要地位。软件复用开发与传统的软件开发方式不同，从而影响到软件度量，因此需要新的软件复用度量方法，软件复用度量的研究已经引起学术界的广泛重视。本文是一篇软件复用度量综述，阐述了软件复用对度量的影响以及有关软件复用的度量。

关键词　软件度量　软件复用　项目管理

A SUMMARIZATION ON THE SOFTWARE REUSABILITY METRICS

Qian Leqiu　Zhang Yong

(*Department of Computer Science, Fudan University, Shanghai 200433*)

Abstract　Software reusability metrics, being an integral part of software reuse technology, plays an important role in application system development based on software reuse. Application system development based on software reuse is different from the traditional software development, so we need a new metrics. More and more scientists are concerned about the software reusability metrics. A summarization on the software reusability metrics is presented in the paper. The impact of software reuse on the software measurement and the related software reusability metrics are discussed in detail.

Keywords　Software metrics　Software reuse　Software project management

1　引　言

软件是信息技术的核心，因而管理人员对软件开发过程中所涉及到的质量、效益度量越来越重视。这种重视引起了两种效果：

（1）要求新的、更好的软件开发方法和技术。

（2）在软件开发过程中，进行软件度量。

过去几十年间，软件生产率一直在持续、稳定地提高，产生了大量的代码、文档。其中许多代码、设计文档都是可以被重复利用的。软件复用利用现存的代码和文档来生产新的软件系统，是提高软件生产率和产品质量的一条有效途径。软件度量可以帮助管理人员控制、安排软件开发并利用反馈信息对软件进行改善，从而提高软件质量，因此软件复用度量是软件复用不可分割的一部分。

度量的研究可以追溯到 70 年代，其时，最为流行的软件开发方法是传统的结构化方法。早期著名的度量，如 McCabe 的环计数[1]、Halstead 的软件科学[2]和 Albretcht 的功能点[3]等，都是针对传统的面向过程的开发方法的。

基于复用的软件开发在软件生命期、系统结构及项目管理等方面同传统的软件开发有着明显的区别，这些区别极大的影响到软件度量。

本文是一篇关于软件复用度量的综述。

2　软件复用的概念及其对度量的影响

系统的进行软件复用的概念很简单：开发大小合适的构件系统并复用它们。合理的复用能够有效的减少项目开发的成本，Graham 指出复用策略能减少至少 20% 的开发成本[4]。复用可用于软件生命期的各个阶段，Jones[5]指出在软件开发过程中可复用的部分。可参见表 1。

表 1　软件项目中可复用的部分

1. 架构(Architecture)	6. 模板(Template)
2. 源代码(Source code)	7. 人机界面(Interface)
3. 数据(Data)	8. 计划(Plan)
4. 设计(Design)	9. 需求(Requirment)
5. 文档(Document)	10. 测试用例(Test case)

收稿日期：2001－02－14。钱乐秋，教授，博导，主研领域：软件复用，基于构件构架的软件工程，软件测试。

软件复用开发与传统的软件开发周期存在很大的不同:首先,软件开发不是从头开始,而是把软件的开发周期分解为两部分:一是可复用构件生产周期;二是利用构件组装应用系统的周期。在这两个周期中,要涉及到构件的识别、生产、查找、理解、扩展和组装等新的开发过程,同时利用复用技术开发新系统之前必须进行效益分析,这些变化都要求有新的质量、经济等度量模型及准则;其次,虽然软件复用不一定必须要面向对象技术的支持,但面向对象所具有的数据封装、继承、多态性等良好的机制促进了软件复用技术的发展,现今的软件复用技术更多的是以面向对象技术为基础的,面向对象技术的引入又赋予度量技术以新的内容,例如类的复杂性度量,类的偶合度度量等,软件复用也必须包括面向对象度量的内容。因此,许多传统的软件质量、经济等模型并不适用于软件复用开发。

软件复用开发与传统软件开发方法的不同及其对度量的影响,导致了新的软件复用度量模型的需求,软件复用度量的理论研究及实践应用已经成为软件工程研究的热点之一。

3 软件复用度量基本内容

目前,许多计算机科学家将软件复用看作是改善软件工程实践的一种潜在的强有力的手段。软件开发者再利用软件复用方法进行开发时,需要去度量开发进度,找出最有效的的复用策略,以及测度出复用的潜力和预期的收益,此外还要识别出现在系统中的可复用构件。

软件复用度量需要基于传统的度量准则和方法,结合其本身特性来构造出适当的度量模型来评价软件开发过程中与复用相关活动的特性。Karlsson 等人认为软件复用度量的研究设计三个方面:产品度量、过程度量和成本效益分析[6]:

- 产品度量主要用于判断构件的可复用性和度量,以及基于构件的软件系统的度量,它包括先期度量和后期度量。先期度量基于检查表和静态度量,在构件的开发期进行;而后期度量则基于复用者的反馈信息,在构件已被复用后进行。
- 过程度量判定复用对生产率、质量和开发时间的作用,它可在不同级别上进行度量,包括构件级、项目级、应用族级、企业级等。
- 成本效益分析用于估计生产可复用构件和基于可复用构件进行开发的成本以及预期的效益分析。

软件复用活动涉及到可复用构件的生产及消费两个阶段,上述三个研究方向在这两个阶段的内容各有不同。下面我们就来详细讨论各度量准则。

4 有关软件复用的度量准则

4.1 可复用性度量

可复用性度量的主要动机有:[6,7]:

- 学习遗产系统中的领域知识,为构件提取提供基础。
- 对构件库中的可复用构件进行客观的控制,保证构件库中保存的是高可复用性和高质量的构件。
- 复用构件库中若包含可复用性度量的信息,可以给使用构件库中构件的复用消费者提供有价值的帮助。

可复用性的度量具有相当的难度,其中之一就是可复用性的特征对于每个特定构件来说是特定的,即许多特征不具备普遍适用性。

Selby[8] 给出一些准则来识别可复用黑盒构件,这些准则包括:

- 相对少的模块调用/代码行。
- 相对少的 I/O 参数/代码行。
- 相对少的 read/write 语句/代码行。
- 大量的注释。
- 相对多的功能调用/代码行。
- 少量的代码行。

Selby 指出具有上述特征的模块是具有高可复用性的。

Basili[9] 报告了两个复用学习系统,第一个学习系统定义了利用数据共享度量来确认可复用构件,首先找出全局共享数据,然后进行串分析计算并给出耦合度的权重,那些强耦合的模块可复用性较差,独立性较强的模块是侯选可复用构件。第二个学习系统定义了一个对 Ada 语言构件的可复用性抽象度量,通过从现存系统转化为利用构件组装的系统所需的改变量,可得到构件的可复用性度量的指标。

Knight[10] 等人也提出了类似的利用模块耦合度信息来度量可复用性的方法。

除了以上的可复用性度量的方法之外,还有 REBOOT 方法,NATO 方法,Chen 和 Lee 方法,Hislop 方法及 Mayober 方法,他们都是根据构件的内部属性来得出度量结果。Jeffrey S. Poulin[7] 指出若研究发现构件所处语境也和其内部属性一样影响构件的可复用性时,就必须在度量的同时考虑其所处的领域属性和环境属性。

4.2 构件库的度量

为了有效使用构件,构件必须分类存储于构件库中。最普通的分类方案是枚举、刻面和索引[11]。分类方案的评价标准是:成本、查找效率、可理解性。

分类成本包括生成、维护分类方案,升级数据库的

费用。该成本可通过由总花费/库中的构件数来得到正规化。

查找效率可由获取准确度及查全率来度量。准确度可由相关构件获取数/总的构件获取数来获得;查全率可由相关构件获取数/库中相关构件总数来获得[12]。

可理解性度量是显示分类方法对用户理解可复用构件的影响度。Frakes 等人利用 7 个等级来度量分类方案对可理解性的支持度(7 = 最好)。

构件库的另一类度量准则是库效率。库效率包括内存占用率、索引文件大小、获取速度等。内存占用率用构件所占内存字节数来度量;索引文件大小用索引头占用率(原始数据大小/索引文件大小)来衡量,获取时间通常由找到时间来度量。

构件库中构件的质量是构件库度量的另外一个重要方面。Frakes[3]提出如下准则来评估构件库中构件的质量。

- 复用统计——构件已被复用次数。
- 复用评价——已复用该构件的机构或个人写的评语。
- 复杂性——过于复杂的构件不是高质量的。
- 内省机制——构件加入构件库时应被检查。
- 测试——构件至少应进行单元测试(达到语句覆盖率 100%,分支覆盖率 80%)。

4.3 经济模型

一旦企业想利用复用技术,首先考虑的问题就是成本和效益问题。

Gaffney 等人[14]提出了两个成本/生产率模型。一个简单模型反映了复用软件构件的花费,Cost - of - development 模型建立在简单模型之上代表了开发可复用构件的花费。

Margono 等人[15]应用 Cost - of - development 模型来估计一个大型 Ada 项目——美国联邦航空管理局先进自动化系统(FAA/AAS)的经济效益,结果显示开发可复用构件的花费是开发非复用构件花费的两倍。

Barnes 等人[16]检查了软件复用的成本和生产特性,提出了一个分析的方法,取得了好的复用投资效果。复用活动可分为生产者和消费者,生产活动是投资即成本发生的活动(生产可复用构件),消费活动是效益产生的地方(用节省的投资来度量)。投资质量(Q)是复用效益(B)与复用投资(R)的比例:Q = B/R,通过分析该模型,我们可用三个方法来改善成本效益比:1)增加复用率;2)减少平均复用成本;3)减少对于固定的效益所需要的投资。

Poulin 等人[17]提出了 IBM 公司使用的复用人力节省的度量准则:

- 复用比例;
- 复用后避免成本;
- 复用增加值;
- 额外开发开销——开发可复用构件增加的成本。

Agresti 等人[18]开展了利用复用对预计错误密度、软件质量的调查活动,得出的结论是:若复用比例在 26% ~ 28%,错误密度是 3.0 ~ 5.5/KSLOC(Kilo - Source Line Of Code)。

Boehm 在 1981 年提出 COCOMO(COnstructive COst MOdel)来度量软件开发成本,但随着软件复用技术应用于软件开发实践中,COCOMO 模型不能准确度量此类开发项目,因此 Boehm 又提出 COCOMO Ⅱ模型[19]。模型描述如下:

$$PA = A \times \{[(Size) \times (1 + Brak/100)]^{(1.01 + 0.01 \sum_{j=1}^{5} SF_j)}\} \times \prod_{i=1}^{17} EM_i + (自适应因子)$$

其中:

PM = 人 - 月;

A = 线性缩放比例常量;

Size = 为适应变化而产生的新代码或更改代码的的量;

Brak = 由于可变性而导致的构件组装过程中的代码丢弃量(按百分比计);

SF_j = 五个非线性比例因子;

EM_i = 十七个人力花费乘数。

SF_j 及 EM_i 的详细定义,由于篇幅所限请参看具体文献。

Favaro[20]提出应该把复用经济模型及投资分析按如下方式安排:

(1) 度量　确定收集原始数据,例如构件大小、复用水平和时间花费等。

(2) 成本效益估计　利用模型把原始数据转化为人力花费、成本、时间等。

(3) 复用投资分析　分析度量结果,并估计和确定如何开展复用业务,预计投资回收时间及决定是否投资。

4.4 复用成熟度模型

一个复用成熟度模型是计划复用、帮助企业了解他的过去,现在和将来的复用活动的目标的基础。复用成熟度度量类似于卡耐基·梅隆大学软件工程实验室的 CMM(Capability Maturity Model)。

Koltum 和 Hudson[21]提出了一个复用成熟度模型,可参见表 2。企业可利用上述模型来评估自己的复用成熟度,以便确定企业基于复用的技术水平和管理水平,然后企业可以根据它们来确定自己的复用计划以达到高水平的复用。

另外,软件生产力协会(Software Producitivity Consortium,SPC)开发出一个复用能力模型[22],它有评估模

表 2　复用成熟度模型

	1. initial/chaotic	2. monitored	3. coordinated	4. planned	5. ingrained
Motivation/ Cluture	Reuse discourage	Reuse encourage	Reuse incentivized re - enforced rewarded	Reuse indoctrinated	Reuse is the way we do business
Planning for reuse	None	Grassroot activity	Targets of opportunity	Bussiness imperative	Part of strategic plan
Breadth of reuse	Individual	Work group	Department	Division	Enterprise wide
Responsible for making reuse happen	Individual initiative	Shared initiative	Dedicated individual	Dedicated group	Corporate group with division liaisons
Process by which reuse is leveraged	Reuse process chaotic; Unclear how reuse comes in	Reuse questions raised at design reviews(after the fact)	Design emphasis placed on off the shelf parts	Focus on developing familitys of products	All software products aregenericized for future reuse
Reuse assets	Salvage yard(no apparent structure to collection)	Catalog identifies language and platform specific parts	Catalog organized along application specific lines	Catalog includes generic data processing functions	Planned activity to acquire or develop missing pieces in catalog
Classification activity	Informal, invidualize	Multiple independent schemes for classifying parts	Single scheme catalog published periodically	Some domain analyses done to determine categories	Formal, complete, consistent timely classification
Technology support	Personal tools, if any	Many tools, but not specialized for reuse	Clasification aids and synthesis aids	Electornic library separate from development envoirment	Automated support integrated with development envoirment
Metrics	No metrics on reuse level, pay - off, or costs	Number of lines of code used in cost models	Manual tracking of reuse occurrences of catalog parts	Analyses done to identify expected payoffs from developing reusable parts	All system utilities, software tools and accounting mechanising instrumented to track reuse
Legal, contractual, Accounting consideration	Inhabitor to getting started	Internal accounting scheme for sharing costs and allocating benefits	Data rights and compensation issues resolved with customer	Royalty scheme for all suppliers and customers	Software treated as key capital asset

型和实现模型两部分组成。评估模型包含一系列关键性的成功复用因素，企业可利用它们来评估企业复用实践的现有状态。这些因素划分为四大类：①管理，②应用开发，③构件开发，④过程及技术因素。实现模型帮助管理者划分复用目标的优先级别，并建立连续的实施步骤，SPC 在实现模型中定义了以下四个可以减少复用风险的步骤：

（1）机会主义　复用策略建立在项目级上，使用特定的复用工具并识别可复用构件。

（2）集成　确定标准的复用策略并把它集成到软件开发过程中，复用计划得到管理者和员工的支持，可复用构件得到分类。

（3）杠杆作用　复用策略发展到整个生命期，复用的效果得到度量并识别出薄弱环节。

（4）期望　利用复用能力及可复用构件进行新的业务，确定高回报率的构件。

5　今后的研究方向

软件复用度量目前仍处于探讨阶段，许多现有的模型应用于实际系统时偏差较大。软件复用度量还需要从以下几个方面来进一步研究：

• 软件属性清晰严格的定义　许多软件的属性，尤其是外部属性缺乏严格的定义，正如 Fenton 所指出："对于许多软件属性，我们还只有非常粗糙的经验关系系统"[23]。

• 形式化评估度量准则　Weyuker 提出的形式化定理有待于进一步完善，这组定理针对的是传统度量准则，随着软件复用的研究与发展，我们迫切需要一个适合于软件复用度量的形式化准则。

• 自动化支持　Grady 认为真正有用的度量应具有自动化功能[24]，否则，就无法收集和计算度量所用数据。

参　考　文　献

[1] A. J. Albrecht, "Measuring Application Development Productivity", Proc. Joint SHARE/GUIDE/IBM application Development Symposium, pp. 34 ~ 43. October 1979.

[2] T. J. McCabe, "A Complexity Measure", IEEE Trans. Software Eng. Vol. 2, No. 4, 1976 pp. 308 ~ 320.

[3] M. H. Halstead, "Elements of Software Science", New York: Elsevier North - Halland, 1977.

[4] I. Graham, Object - Oriented Methods, Addison - Wesley, Wokingham, UK, pp. 410, 1991.

[5] C. Jones, "Software return on investment preliminary analysis." Software Productivity Research, Inc., 1993.

[6] "Software Reuse: A Holistic Approach" - Measuring the Effect of Reuse Chapter, edited by Even - Andre Karlsson. Chichester; New York: Wiley, c1995, published by Wiley & Sons, Ltd. Baffins Lane, Chichester Sussex PO 191 UD, England.

[7] Jeffey S. Poulin, "Measuring Software Reusability" Third International Conference on Software Reuse, 1994.

[8] Richard W. Selby, "Quantitative Studies of Software Reuse," in "Software Reusability, Volume II," Ted J. Biggerstaff and Alan J. Perlis(eds). Addison - Wesley, Reading, MA, 1989.

[9] V. R. Basili, H. D. Rombach, J. Bailey, and A. Delis, "Ada reusability and measurement." Computer Science Technical Report Series, University of Maryland. May(1990).

[10] M. F. Dunn, J. C. Knight, "Software Reuse in an Industrial Setting: A Case Study." In 13th International Conference on Software Engineering, pp. 329 ~ 338. Austin, TX: IEEE CS Press 1991.

（下转第 32 页）

锁、远程控制、紧急保护等各种开关状态量的显示；③联锁单元工作模式设定；④报警及故障显示；⑤记录加速器运行历史数据和实时数据，并打印运行数据报告等等。

4 结束语

用 PLC 作为下位机使整个控制系统的稳定性、可靠性大为提高。自加速器进入调束阶段以来，控制系统运行可靠、平稳。不久前已调出束流。

参 考 文 献

[1] SIMATIC S7 - 200 可编程控制器参考手册，西门子公司 .

[2] 郭宗仁，可编程控制器及其网络通信技术，北京：人民邮电出版社，1999.

(上接第 4 页)

[11] W. Frakes, and P. Gandel, "Representing reusable software", Information and Software Technology, Vol.32, No.10, 1990, pp.653 ~ 664.

[12] W. Frakes, and T. Pole, "An Empirical Study of Representation Methods for Reusable Software Components." IEEE Transactions on Software Engineering 1994.

[13] W. Frakes, and B. A. Nejmeh, "Software Reuse Through Information Retrieval", Proceedings of the TwentiethAnnual Hawaii International Conference on Systems Sciences, Kona, pp.530 ~ 535, January 1987.

[14] J.E. Gaffney, and T.A. Durek, "Software reuse - key to enhanced productivity: some quantitative models", Information and Software Technology, Vol.31, No.5 1989, pp.258 ~ 267.

[15] T. Margono, and T. Rhoads, "Software reuse economics: cost - benefit analysis on a large ~ scale Ada project", In International Conference on Software Engineering ACM, 1993.

[16] Barnes, B. and Bollinger, T., "Making software reuse cost effective", IEEE Software, Vol.1, No.1 1991, pp.13 ~ 24.

[17] J.S. Poulin, J.M. Caruso, and D.R. Hancock, "The business case for software reuse", IBM Systems Journal, Vol.32, No.4 1993, pp.567 ~ 594.

[18] W. Agresti, and W. Evanco, "Projecting software defects in analyzing Ada designs", IEEE Transactions on Software Engineering Vol. 18, No. 11 1992, pp.988 ~ 997.

[19] B.W. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy and R. Selby, "Cost Models for Future Software Life Cycle Process: COCOMO2.0", Annals of Software Engineering Special Volume on Software Process and Product Management, J.D. Arthur and S.M. Henry, Eds., J.C. Balter AG, Science Publishers, Amsterdam, The Netherlands, 95, Vol.1, pp.45 ~ 60.

[20] Ivar Jacobson, Martin Griss, Patrik Jonsson, "Software Reuse——Architecture Process and Orgnization for Busssiness Success", ACM press, 1997.

[21] P. Koltun, and A. Hudson, "A reuse maturity model", In 4th Annual Workshop on Software Reuse in Herndon, VA, 1991.

[22] T. Davis, "The reuse capability model: a basis for improving an organization's reuse capability", In 2nd International Workshop on Software Reusability in Herndon, VA, 1993.

[23] N.E. Fenton, "Software Measurement: A Necessary Scientific Basis", IEEE Trans. Software Eng., Vol.20, No.3, pp.199 ~ 206, Mar. 1994.

[24] R.B. Grady, Practical Software Metrics for Project Management and Process Improvement, Prentice Hall, Englewood Cliffs, NJ, 1992.

(上接第 19 页)

项目跟踪与监督实施管理。它具有以下主要特点：

(1)采用 RUP(Rational Unified Process)方法，提供完整的需求分析及管理流程。

(2)以 Web 方式获取反馈，加强团队之间的有效沟通。

(3)用追踪图直观展现需求变化带来的影响。

现在，不少软件开发组织认识到软件需求管理的重要性，采用购买或自行研制的方式采用需求管理或项目管理的工具，这是我国软件产业发展中的一个可喜现象。

4 结 论

软件开发组织的项目管理者应充分认识软件需求不确定性的事实，根据实际情况采取相应的对策是十分必要的。特别是在把进行项目开发转向产品开发，把企业做大做强的背景下，把握软件需求不确定性的认识尤其重要。一次性开发成功一个软件产品几乎是不可能的，尤其在市场竞争激烈的今天。一个软件产品从商业的角度是尽快打向市场，但要被用户认可，真正占领市场，必须不断地从用户的反馈中持续改进，此时 SURCM 方法是可以采用的轻载方法。

参 考 文 献

[1] Roger S. Pressman, Software Engineering, A Practitioner's Approach, Fourth Edition, 1997, 32 ~ 36, 270 ~ 284.

[2] Martin Fowler, The New Methodology, http://martinfowler.com/articles/newMethodology.html.

[3] Mike Beedle, Martine Devos, etc. SCRUM: An extension pattern language for hyperproductive software development, http://www.controlchaos.com/.

[4] 卡内基梅陇大学软件工程研究所，软件能力成熟度模型 1.1 版(中文翻译版)，http://www.platinumchina.com/index_product_cmm.htm.

[5] 何新贵等，软件能力成熟度模型，清华大学出版社，2000, 11, 27 ~ 30.