软件重新工程化 软件维护 软件工程 (15)

计算机科学 1995Vol. 22 №. 5

65-68

软件重新工程化方法

齐 越 柳军飞 TP311.5

（北方交通大学计算所 北京 100029） （中国科学院软件所 北京 100082）

摘 要 软件重新工程化方法与技术是九十年代软件维护自动化的重要内容，日益受到人们的重视。本文讨论软件重新工程化的概念、方法和过程。主要内容有逆向工程、系统重构和软件仓库。

关键字 重新工程、逆向工程、重构、软件仓库。

一、引 言

软件维护是软件生命周期的重要阶段。为了修改软件中潜伏的错误和缺陷；按用户新的需求提高软件性能和功能；将软件安装到新的软硬件环境中去等等都需要对软件进行维护。据统计软件维护费用在软件生命周期中所占比重已达到60～70%。美国空军F16喷气战斗机的软件开发费用是8500万美元，而它的软件维护费用高达2.4亿美元。软件维护开销大的原因是：①软件是逻辑产品，维护人员理解软件要占用47～60%的维护工作量。当软件缺乏必要的文档或文档质量很差时，软件维护只能依靠程序代码。②用新的软件系统替换旧的软件系统后，用户要承担新、旧两套软件费用。③缺乏十分有效的软件工具可以根本上减轻软件维护的工作量。④计算机厂家不断推出具有高性能价格比的计算机系统，促使计算机用户不断淘汰旧系统、建立新的系统。软件运行环境的改变、用户需求的变化、原有软件技术陈旧使软件维护的工作量越来越大。软件维护与硬件维护相比存在本质上的差别。

象的某一个个体，与其它对象无关，这大大减小了程序的复杂性。

4）Visual C++是一种面向对象的编程语言，并具有可视的特征。给OOCPN运行及动画系统的实现创造了良好的环境。

参考文献

[1]陆维明，林闯，Petri网研究：机遇与挑战，计算机科学，Vol. 21，No. 4，1994

[2]J. L. Villarroel et al.，Using Petri net models at the coordination level for manufacturing systems control，Robotics Computer-Integrated Manufacturing，Vol. 11，No. 1，1994

[3]M. C. Zhou and F. Dicesare，Parallel and sequential mutual exclusions for Petri net modeling of a manufacturing sysetms with shared resources，IEEE Trans. RA，Vol. 7，No. 4，1991

[4]G. Righini，Modular Petri nets for flexible production systems，Int. J. Prod. Res.，Vol. 31，No. 10，1993

[5]M. C. Zhou et al.，Design and implementation of a Petri net based supervisory for a flexible manufacturing system，Automatica，Vol. 28，No. 6，1992

[6]N. P. Huang and P. C. Chang，Specification，modeling and control of a flexible manufacturing cell，Int. J. Prod. Res.，Vol. 30，No. 11，1992

[7]R. Cossins and P. Ferreira，Celeritas，a colored Petri net approach to simulation and control of flexible manufacturing systems，Int. J. Prod. Res.，Vol. 30，No. 8，1992

[8]李晓鸥，徐心和，DES的面向对象着色Petri网模型研究，将于《信息与控制》1995年第六期发表

[9]Xiaoou Li and Xinhe Xu，Modeling DES with a modified colored Petri nets，Proc. of 1995 American Control Conference，Seattle，June，1995

[10]Z. A. Banaszak and B. H. Krogh，Deadlock avoidance in flexible manufacturing systems with concurrently competing process flows，IEEE Trans. RA，Vol. 6，No. 6，1990

齐 越 硕士研究生。 柳军飞 博士后。

近年来日益受到人们重视的软件重新工程化(Re-Engineering)企图用软件自动化技术支持软件维护。

二、软件重新工程化

软件重新工程化技术包括:分析技术、重构(Restructuring)技术、逆向工程(Reverse Engineering)和移植。软件重新工程化过程如图1所示。

分析技术用于研究当前软件系统的资料、理解系统的结构、了解系统的程序如何工作、为重新工程化标识顶层的对象和过程、测量软件的质量。逆向工程是,分析当前软件系统、从程序代码的物理表示出发,重新构造它的各个部件及它们之间的相互关系,逐步恢复程序的高层描述。逆向工程的目的是重新建立软件系统文档,寻找、收集、整理设计信息,提高软件的可理解性。重构是修改软件表示形式的过程。如,对数据的命名和定义,它不改变程序的功能。重构的主要目的是提高软件的可理解性,因为只有对软件理解准确才能着手维护。对于需要改善功能和性能的软件需要在原有设计的基础上重新设计。重新设计应尽量重用原有软件规范、设计、部件、测试题目等等,提高重新设计的质量和效率。软件移植指,把软件系统从一种语言描述转换成另一种语言描述,从一种运行环境安装到另一种运行环境。正向设计就是通常的软件工程过程,从软件系统的规格说明、详细设计出发,经过编码、调试、测试、确认生成软件系统目标代码。逆向工程过程、重构过程、正向工程过程构成了软件重新工程化的基本模型。软件重新工程化是在软件仓库(Repository)和软件工具的支持下完成的。当旧系统的功能或性能不能满足用户要求时,必须对旧系统进行修改或重新设计。

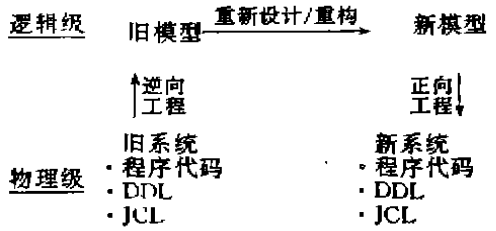

图1 软件重新工程化过程

三、逆向工程

从软件维护自动化的角度来看,人们希望软件维护工作不是在代码级,而是在软件规格说明和设计级完成。这样可以简化维护工作、提高维护质量、降低维护成本。通过逆向工程处理的程序可达到比较高的抽象层次。逆向工程过程中不仅收集、整理当前软件系统的过程和数据模型、部件,而且还有许多有价值的信息。人们把这些信息存放在软件仓库中,软件仓库提供信息存储、信息自动管理和信息的标准表示,为软件重构,重新设计和正向工程打下基础。

逆向工程分为数据逆向工程和逻辑逆向工程两类。数据逆向工程工具和逻辑逆向工程工具分别支持数据逆向工程过程和逻辑逆向工程过程。

数据逆向工程过程如图2所示,它不仅能从源代码、数据字典和数据定义语言(DDL)抽取实体、数据关系、属性和物理设计规格说明,而且还可以创造、修改、整理、验证物理数据库图和逻辑图。如E—R图,图的在线图形表示。它还可以根据更新的逻辑图和物理图重建DDL模式。数据逆向过程可分为以下几个步骤:

①读源代码,数据描述和DDL。

②生成物理设计图并存储在软件仓库中。

③对图进行修改并生成新的DDL。

④对物理设计图进行抽象,得到逻辑设计图并存放在软件仓库中。

⑤修改逻辑设计图,并按正向工程步骤生成新的物理设计图和新的DDL。

数据逆向工程工具不仅能帮助人们理解、修改数据库,而且还可以把这个数据库移植到新的数据库管理系统。数据逆向工程工具以软件仓库信息为基础,提供专用图验证、专家建议、文档生成、输出查询文件、建立规范的数据库描述文件等方面的服务。

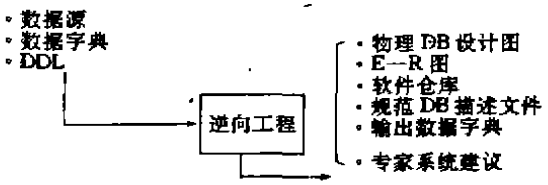

图2 数据逆向工程过程

逻辑逆向工程,从软件仓库的源代码和其它信息中抽取软件的物理层设计信息和逻辑层设计信息,并把这些信息存储在软件仓库中。可以使用逻辑逆向工程工具对软件仓库中的信息进行重建、修改、整理和验证。抽取的信息包括:程序的数据描述、程序结构、数据流图、控制流图等等。如图3所示,逻辑逆向工程过程可分以下几个步骤:

①收集信息。包括:源代码、设计文档、系统调用和引用外部过程的文档、与软件有关的个人经验和历史资料等等。

②审查、评审采集信息。

③提取、建立并标识程序的结构。结构图中结点表示程序对子程序的调用;它表示数据传送关系。

④记录结构图中每个结点的功能;并用PDL表示。涉及的子程序用自然语言或其它语言描述。

⑤记录数据流程。分析PDL，恢复程序的结构、标识程序中的数据变换、建立数据流图。

⑥标识程序的高层控制结构并用控制流图表示。

⑦评审再现设计。

⑧生成设计文档。

逻辑逆向工程抽取和整理出来的信息分物理设计、逻辑设计、代码三级存储在软件仓库中。物理设计级包括：数据结构、硬部件、模块调用关系、模块物理设计等；逻辑设计级包括：数据流图、过程模型、数据字典等；源代码级存储生成的代码。软件仓库中物理设计信息用基于对象的形式语言或伪码表示。在基于对象的语言中，对象是模块、记录或事件，关系是数据流和控制流。用伪码可以表示模块之间的关系、模块内部的逻辑结构和数据结构并存储在软件仓库中。人们借助软件工具可以对软件仓库中的物理设计进行修改，修改后必须进行错误检查。如，文法检查、完全性检查、一致性检查、复杂性检查、死锁检查等等。通过检查的物理设计和逻辑设计在软件工具支持下生成图形或报告文档。如，形式设计规格说明文档、模块层次报告、程序结构图、数据模型报告、控制流图等等。一旦对物理设计完成修改和检验以后，利用正向工程的CASE工具将这一新的物理设计转换成新的源代码。

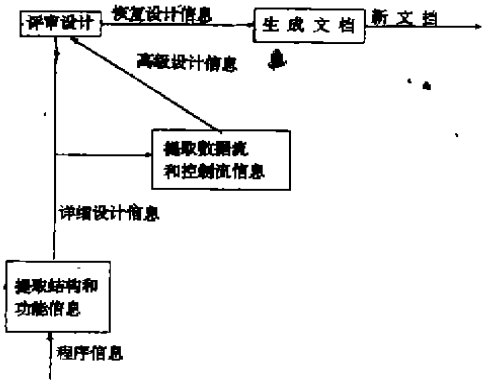

图3 逻辑逆向工程过程

代码可以用C语言、PASCAL语言、Ada语言或其它适宜的语言表示，从而实现程序在不同语言之间的转换。

四、软件重构

如果一个软件难读、难修改、难测试，代码质量差，错误比较多、必须配备高级的专门维护人员，而这个软件又非常重要，那么这样的软件就需要重构。软件重构是根据软件开发标准对程序中的数据、变量重新命名、重新定义，对程序逻辑重新设计并保持原程序功能和性能的过程。软件重构应该改善程序的可读性、简化程序逻辑、减少程序的测试和调试时间、提高软件质量、降低维护成本、减少对个别维护人员的过分依赖、保护软件可重用的成分等。程序的逻辑重构过程也是对源代码按照结构程序设计规则重新编排的过程。因此，重构后的程序应具有良好的结构，规范地使用程序设计语言及代码的排列格式。重构过程按下列步骤进行：

①使用程序代码分析器，分析非结构化程序，找出并删除其中缺陷。如死代码、GOTO语句、无限循环等。

②编译非结构程序，标识和校正程序中语法错误。

③评审编译和分析的结果、修正程序中的错误。

④借助软件重构工具的帮助对非结构化的程序按结构化程序的要求重新构造。注意寻求软件中能够重用的部分并生成可重用软部件。

⑤借助于优化编译器，重新编译结构化的源代码，提高代码的效率。

⑥重新确认结构化程序，确保非结构化和结构化程序有相同的功能。确认程序中可重用软部件并把它们存放在软部件库中。

经过逻辑重构的程序应消除GOTO语句、无用代码、失去控制的路径、无限循环等不合理的结构。建立模块和模块的层次结构使之成为结构化的程序。多数对代码重构的方法是以图论为基础的。用流程图模型表示程序的控制流。流程图由点和线组成。点表示一个或几个顺序执行的语句。线表示语句之间的控制路径。当对某一个程序进行重构时，需要在重构工具支持下将这一个程序的源代码转换为程序流程图，然后对非结构化的程序流程图进行重构，得到结构化的程序流程图。它经过自动或非自动编码可以得到结构化的程序。以图论为基础的流程图重构方法在数学上可以保证重构前后的程序在功能上是等价的。

在大多数软件维护活动中，需要对当前软件进行修改和设计。这项工作应该在逻辑设计级进行。这样可以降低软件维护的复杂性，提高软件维护质量。需要修改和重新设计的维护过程应该在逆向工程之后，软件重构之前完成。在逻辑级经过修改和重构的软件按照正向工程过程生成的源代码必要时可选用另一种程序设计语言，从而达到软件重构和移植的目的，正向工程过程应该在相应的CASE环境支持下进行，从而保证重构后的软件有比较高的质量。

五、软件仓库

软件开发和维护人员希望有一个能够支持软件

开发和维护的集成化的CASE环境。九十年代发展起来的软件仓库是这一环境的核心。软件仓库将软件开发和维护需要的各种软件工具集成在一起，按照公共的、一致的标准描述软件系统、共享与软件有关的信息。包括：管理信息、开发信息、维护信息等等。软件仓库支持软件重用，收集、整理、存储、管理可重用软部件和按领域划分的软件有价值的信息。软件仓库的信息包括：①逻辑数据和过程模型，②数据和过程的物理定义和代码，③软件开发的生命周期过程模型，④软件项目的管理信息，⑤软件仓库元模型和确认规则，⑥单位（或企业或公司）的组织结构模型、事务性数据、事务性规则及其相互关系等等。软件仓库中的信息具有标准的表示形式和公共框架，它独立于实现，便于人们理解和使用。软件仓库还提供公用数据的共享、访问和管理能力。软件仓库把一个单位的各种专用数据字典集成在一起，并对全部信息进行管理，保证数据定义的一致性。以软件仓库为核心，将支持软件生命周期的全部CASE工具和重新工程化工具集成在一起。多数单位都有一定的服务领域，软件仓库中按一定的数据关系模型存放这些信息，必要时可用这些信息生成应用数据库系统供整个单位共享这些资料。软件仓库按照软件开发和维护方法的特点和要求，将软件生命周期各个过程的信息集成在一起，有利于软件开发、维护、移植和重用。因此软件仓库可用于软件的开发、维护、管理和运行，支持软件工程的全部活动。软件仓库集成环境如图4所示。

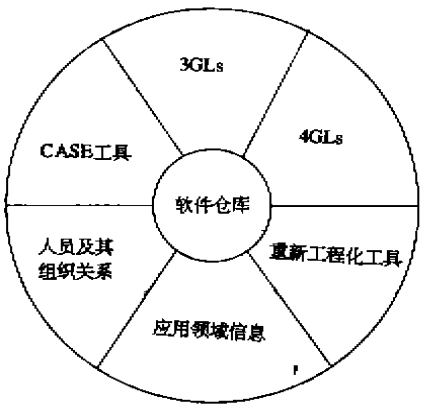

图4 软件仓库集成环境

软件仓库的元模型为软件仓库的系统信息提供公共的表示。因此在软件工程各个阶段使用的软件工具对软件仓库的所有信息都有一致的理解。软件仓库中存储的软件项目信息从抽象到具体，从项目的需求定义到生成的可执行代码，都是根据软件工程方法学的特点，分阶段存放的。除了过程的开始和终止外，上一阶段的输出，作为下阶段的输入。整个过程的信息和产品质量都是可以追踪和控制的。软件仓库中的方法学信息帮助用户选择软件开发和维护过程中需要的软件工具。在以软件仓库为中心的集成化CASE工具和重新工程化工具可以共享公共数据界面和用户界面，可以访问公共的软件仓库，可以借助软件仓库元模型提供的公共框架表示与工具有关的系统信息等等，从而简化了系统的复杂性，有利于人们对信息和工具的学习和理解。软件仓库采用的元模型主要是ER模型和面向对象模型两种。

当前多数软件仓库的实现都是基于ANSI SPARC的层次结构。整个软件仓库分三层。最外层是逻辑模型，是用户与系统的界面。逻辑模型从用户的角度来描述系统的信息和工具。中间层是概念模型，给出整个软件仓库的表示，而与具体的实现无关。软件仓库中的数据在中间层通常采用ER方式表示。最内层是物理模型，它按照数据的关系模型和具体实现的物理要求表示软件仓库的数据存储。因此可以用关系数据管理系统，如Oracle，DB_2支持软件仓库的数据管理。软件仓库的信息管理应包括：安全性、版本控制、变更管理、审计、数据访问和确认、查询和报告、多用户访问管理、配置管理等等。

至此，我们粗略地阐述了软件重新工程化的意义、作用和过程。阐述了软件仓库在软件重新工程化中的地位和作用。相信，软件重新工程化的进步必将大大推动软件维护自动化的进程。

参考文献

[1] PAV Hall, Overview of reverse engineering and reuse research, Information and software Technology, Vol 34 No 4, PP239-249, April. 1992

[2] N Mill and T Takeshita, Software Re-Engineering and reuse from a Japanese point of view, Information and software Technology, Vol 35 No 1, 1993

[3] Carma McClure, The Three Rs of Software Automation, 1993

[4] ERIC K. Byrne, Software Reverse Engineering: A Case Study, Software-Practice and experience, Vol. 21(12), 1991