软件配置管理中版本管理技术研究

刘燕秋　勉玉静　赵文耘
（复旦大学计算机系软件工程实验室，上海 200433）
E-mail：yanqiu@etang.com

摘　要　通过对软件配置管理系统现状的分析，提出了可变粒度的版本管理系统及其相应的实现技术，并对版本管理的并发控制进行了讨论。

关键词　构件　版本管理　配置　并发控制

文章编号 1002-8331-（2003）21-0068-04　**文献标识码** A　**中图分类号** TP31

Research and Development of WINGCM Version Management

Liu Yanqiu　Mian Yujing　Zhao Wenyun
（Software Engineering Lab.，FuDan University，Shanghai 200433）

Abstract：This article presents a variety granularity software configuration management system WINGCM and the mechanism of implementation of version management，which provides effective control over software developing process.

Keywords：Component，Version Management，Configuration，Parallel Control

1　引言

早在70年代初期，人们就提出了软件配置管理的概念和软件过程工程的一些思想，研究实现了第一代软件配置管理工具。这些工具通过在特定的时刻标识软件系统的配置，系统化地控制对配置的变更，维护了配置的完整性和可追溯性。

1.1　配置管理及版本管理

配置管理是软件过程的一个关键部分，是支持项目小组开发和维护，使软件产品演化过程趋于稳定的一系列控制规则[1]。根据IEEE的标准定义，配置管理包括（1）标识：反映产品结构，标识构件及其类型；（2）控制：在生命周期中，控制产品发布及变更；（3）状态统计：记录报告产品状态和变更请求，收集构件统计信息；（4）审计、审查：维持产品完整性和一致性。后来，随着异质平台开发、小组协作的出现，配置管理的定义被继续扩展。配置管理还包括（5）生产：管理产品组装和构建；（6）过程管理：执行组织的过程、策略和生命周期模型；（7）小组协作：支持小组人员之间的交流和协作[2]。

软件配置管理具有：标识和确定系统中配置项的过程，在系统整个生存周期内控制这些项的投放和变更，记录并报告配置的状态和变更要求，验证配置项的完整性和正确性的作用。软件配置管理的重要性在SEI提出的能力成熟度模型（Capability Maturity Model）和ISO 9000中都有体现。

在软件配置管理系统中，版本管理功能是实现其它功能的基础，也是配置管理系统的基础和核心。

版本管理是针对软件开发过程中涉及到的各种软件资源进行的管理。有效的版本管理有助于对软件开发过程中产生的各种中间产品进行有效的管理，有助于选择合适版本的构件组成软件的发布版本。

因此，能否实现有效的版本管理，现在已经成为判断软件企业是否专业化和正规化的重要标准。

1.2　配置管理工具的发展

第一代研究开发的软件配置管理系统以RCS和Make为代表。此时的配置管理系统可以说仅仅是基于文件的版本控制工具。有的只对源对象[3]提供版本控制，如SCCS；有的则提供了对导出对象[3]进行版本控制，如Make和RCS等。这些都是一些面向开发而非面向管理的、简单的、独立的工具。

第二代配置管理系统以DSEE和Adele为代表。它们能够处理复杂的元素，提供了不同的版本模型，是一个集成了版本管理和项目组建功能的工具。如DSEE在实现对文件进行版本管理的基础上，应用了系统模型（系统模型是指由源对象及其关系描述的软件系统）的描述和版本绑定等策略来支持项目组建。但这些工具仅是支持编码阶段的工具。

第一代和第二代配置管理系统，作为一种支持开发的工具，提供了帮助开发者对软件产品进行协作变更的功能。但是，软件配置管理系统还应该在某种程度上支持软件开发的管理功能，即对软件产品提供变更控制的功能。

ClearCase是第三代配置管理系统的代表。此时的工具已经发展成为面向管理的、大型的、复杂的配置管理系统。其版本管理的对象可以是文件，也可以是目录。与第一代、第二代配置管理系统相比，其版本管理的对象有了很大的扩展。但其管理对象也仅是文件及目录，对基于构件的软件开发支持不够。为了实现对软件开发过程及可变粒度构件的管理，文章提出并实现了WINGCM系统。

1.3　WINGCM系统介绍

WINGCM是一个基于过程的，以项目元素为配置管理对象，支持小组协作以及局域网内部分布式软件开发过程的软件配置管理系统。该配置管理工具具有如下特征：

基金项目：国家863高技术研究发展计划“基于Internet以构件库为核心的软件平台”项目（编号：2001AA1100241）资助
作者简介：刘燕秋（1978-），女，硕士研究生，研究方向：软件工程。勉玉静（1978-），女，硕士研究生，研究方向：软件工程。赵文耘（1964-），男，教授，研究方向：软件工程。

(1)通过灵活的过程定制、基线设置、变更控制等一系列过程管理方法,全面支持企业的过程式软件开发。

(2)通过构建企业内部实用构件库、项目级构件库,对基于构件和软件复用的开发过程提供有力支持。

(3)提供可变粒度的软件配置管理,提高了该软件配置管理系统的可理解性。

(4)具有完善、灵活的版本控制能力,通过可视化的版本树管理,对软件开发过程的各种资源(包括测试用例等)以及软件系统的不同演化方向提供全面的管理支持。

(5)提供强有力的权限控制机制及锁机制,支持团队并行开发,确保资源安全、有效的共享。

(6)WINGCM 充分考虑企业过程改进需要,为企业通过 ISO9000 质量体系认证和 CMM 认证提供有力的工具支撑。

2 WINGCM 版本管理技术

WINGCM 是基于过程的可变粒度的配置管理系统,其版本管理以项目元素为管理粒度。在 WINGCM 中,项目元素是指用户在开发过程中创建的目录、文件,以及从企业内部实用构件库中提取或者在项目开发过程中创建的构件等。因此,WINGCM 的管理对象既可以是目录,也可以是文件或者构件等。

2.1 项目元素版本树

WINGCM 中,所有对象都有版本,因此可以组成结构化的版本。为此,采取了一系列的封装和抽象机制。在该系统中,配置项包括任何可以版本化的实体、关系、属性等,是项目元素的一个特定的版本。

WINGCM 采用一种比较简单实用的版本编号机制,如图 1(数字 1.0 等为同一文件或者目录的版本编号):

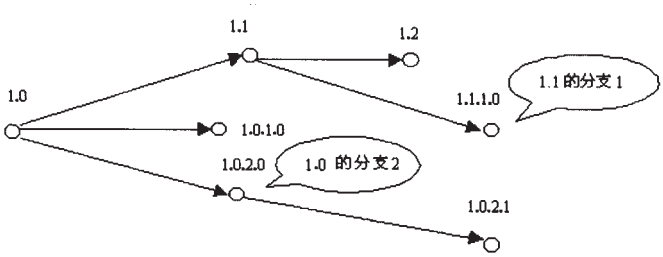

图 1 版本编号机制

2.2 配置数据库和资源库的组织

WINGCM 利用现代比较成熟的数据库技术,在实现中采用 Client/Server 二层组织结构和库的多层组织形式(如图 2)。

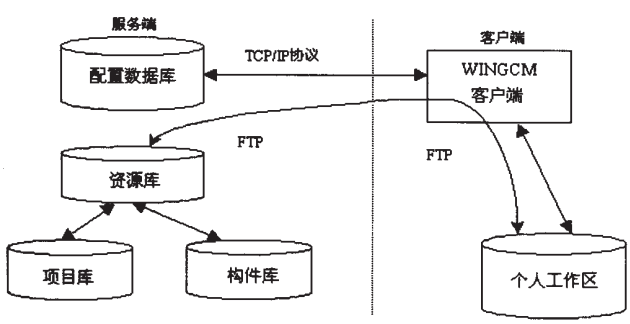

图 2 配置数据库和资源库组织图

资源库和配置数据库是位于最上层的库。配置数据库用来记录各种元数据,包括开发过程中产生的各种项目元素之间的相互关系和日志信息等,例如:何人在何时进行了何种操作,将产生何种影响等信息。这些信息被用来提供用户需要的审计和统计报告,追踪项目元素的一致性和完整性。

资源库是一个逻辑概念,整个资源库可分为项目库和构件库。资源库用来永久保存项目开发中产生的各种项目元素(包括文件、目录和构件等)。项目元素在资源库中采取增量存储[4]的方式。

WINGCM 中,项目元素及其版本关系以及在此基础上实现的增量存储如图 3。

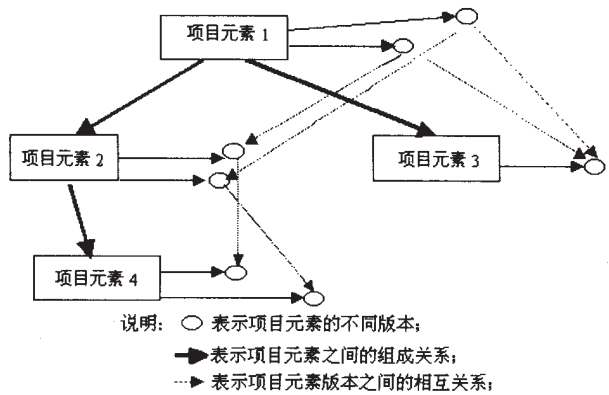

图 3 项目元素组成关系图

图中项目元素 4 有两个版本,导致项目元素 2、项目元素 1 也有两个版本。该图还显示:项目元素 1 由项目元素 2 和项目元素 3 共同组成;项目元素 1 存在 2 个不同版本;项目元素 1 的 2 个版本分别在项目元素 2、项目元素 3 版本的基础上进行版本提升或者分支而来。

WINGCM 中,对于目录的增量存储,依靠数据库技术,在数据库中存储组成目录的文件或者子目录的相对位置,对于不同版本中的相同子目录,在资源库中只存储一次。如图 3 中,项目元素 3 在资源库中只存储一次。

对于文件,传统的文件版本管理系统中,文件的增量存储技术已经得到了比较深入的研究,并且已经有实际的算法及实现。如最早的配置管理系统 RCS 采用的是最长公共子序列算法[4]。

2.3 WINGCM 版本管理模式

2.3.1 WINGCM 版本管理

WINGCM 主要采用检出/检入模式,同时也吸取了一些变更集模式[5]的思想。检出/检入模式支持项目元素版本的分支与合并,通过项目元素的权限管理和版本锁定机制来支持小组进行并行开发。

在 WINGCM 中,首先,用户在权限允许情况下,将项目元素从资源库中检出到个人工作区,然后,用户在个人工作区对所检出的项目元素进行修改,最后,再将修改后的结果检入到资源库中。同时,配置数据库中记录检入者、检入日期、修改原因等信息,以备用户要求提供审计、统计报告之需。

当个人工作区中检出项目元素的版本不是其所在版本树分支的最新版本,即用户检出的项目元素版本在该用户检入前已经由其他人员提供了更新的版本时,系统可以通过消息机制来通知用户。用户的检入操作将使得其检出的项目元素产生分支。

WINGCM 引入了变更集模式的思想来支持系统演化，提供变更控制功能。WINGCM 中，变更分为过程内变更和基线变更。过程内变更比较简单，基线的变更则要通过一系列严格的审批程序。

2.3.2 检出/检入算法

WINGCM 支持项目元素的检出/检入，即文件、目录的检出/检入。具体的检出算法如下：

(1)判断个人工作区是否可用：是否已经存在，是否需要新建等；如果不可用，转(6)执行；

(2)判断检出权限，在达到基线(WINGCM 中，基线是指项目的一个过程或者子过程进行到成熟阶段时的里程碑，它标志着项目可以进入下一个过程或者子过程)时，用户不能进行写检出；WINGCM 中，构件只能作为整体检出，构件内部的文件不允许单个检出，这些情况，都归为用户没有检出权限。没有权限，转(6)执行；

(3)处理未检入的同名项目元素，如果不存在与所检出的根配置项同名的子目录，则成功，否则，需要进行覆盖检出，其它情况视为失败；如果处理失败，则转(6)执行；

(4)检出配置项：确定检出模式，是否上锁，将配置项拷贝到个人工作区，取下一个子配置项，直到没有子配置项为止；在确定检出模式时，需要注意的是：配置项的检出模式有三种：只读、互斥写、共享写。如果对所检出配置项的子孙配置项的模式不进行具体指定，则其子孙项目元素的操作模式由所检出项目元素的操作模式和当时子孙项目元素的动态锁共同决定。同时，由于版本管理粒度可变，在检出项目元素时，对于其子孙项目元素，用户可以进一步具体指定其检出模式。系统还提供动态的修改已检出项目元素或者其下层子项目元素的上锁状态的功能。

(5)刷新项目视图；

(6)算法结束。

检入算法如下：

(1)检入前预处理，包括判断是否可以检入，属于何种性质的检入(不变、新增、删除或者修改)、查看检出日志等；

(2)检入核心处理：包括判断检入的是文件还是目录，检入目录，转入(4)执行；

(3)检入文件比较简单：设置检入结果，如果检入结果是增加或者修改，则拷贝项目元素到资源库中，生成新的配置项和检入结果；在对文件进行检入时，需要考虑配置项的版本是自动提升还是产生分支的问题：如果在文件检入前，配置库中已经存在比检出时更新的版本时，则产生分支，否则，只将文件版本进行提升；转(5)执行；

(4)检入目录比较复杂：首先判断检入的性质(不变、新增、删除或者修改)，递归地取下一个子项目元素，对子项目元素进行检入预处理、核心处理和善后处理直到所有的子项目元素处理完毕。在对目录进行检入时，同样需要对配置项的版本是自动提升还是产生分支进行的情况分别进行处理：如果子项目元素没有生成新的分支，并且原父配置项所在分支的最新版本中仍包含该子项目元素，那么就在这个最新的版本上进行版本提升，生成新的子配置项；但是当子项目元素生成了新的分支，那么父项目元素也要在原父配置项上生成新的分支，或者如果子项目元素没有生成新的分支，并且原父配置项所在分支的最新版本中已经没有该子项目元素了，那么就在原父配置项上生成新的分支。在检入时，我们采取一种“向上传递”的策略，递归的自下而上依次为每个祖先项目元素生成新版本的配置项，生成新的父子关系，直到项目。对于从企业内部构件库中提取到项目中的构件或者直接在项目中创建的构件，从“构件”到“项目”配置项的版本在原配置项所在分支的最新版本上进行提升。

(5)检入善后处理：包括是否需要解锁，写检入日志，设置权限，向相关开发者发送变更通知等。

(6)算法结束。

2.4 版本管理并发控制

为了提供项目的小组工作方式，必须提供版本管理的并发机制。在提供良好的版本管理的基础上，辅之以权限管理和锁机制。

2.4.1 权限管理

在软件开发过程中，不同的项目组成员将充当不同的角色。他们在项目生命周期的不同过程阶段进行工作，具有不同的对项目元素进行操作的能力。

在这里，需要强调的一点是：由于 WINGCM 是基于过程的配置管理系统，对项目有比较严格的权限控制，对于项目生命周期模型中某一具体过程阶段的开发人员来说，他可能根本就不具备检出其它过程产生的中间产品的权限。当然，更高层的管理人员在某些情况下可以将这种权限赋予需要的人员。

WINGCM 包括四种类型的权限：

(1)系统权限：指用户使用该系统时所能执行的不同的功能。如：系统管理员能够定义系统提供的基本过程框架等。

(2)过程权限：与过程操作相关的权限。如具体化、实例化系统提供的基本过程框架，建立配置，发布基线等。

(3)构件权限：指发布构件、维护构件基本信息等权限。这体现了配置管理系统对于基于构件的软件复用技术的支持功能。

(4)配置权限：对某个具体配置项进行操作的权限。是检查用户能否检出某个配置项以及确定其检出方式的依据。用户对构件的权限包括：无权限、读权限和写权限。

2.4.2 锁机制

WINGCM 中，用户对配置项的权限(即配置权限)被称为静态锁。在实际的项目开发中，项目元素可能因为被别的用户并发操作而被上锁，这种锁称为动态锁。配置项是否上锁反映了是否有其他用户正在对该配置项进行互斥写，以及该配置项是否处于基线上。

表 1 显示了静态锁和动态锁共同作用下，项目元素检出模式的改变情况(其中行表示静态锁，列表示动态锁。——表示无权限，即用户对项目元素不可见)。

表 1 静态锁和动态锁对检出模式的改变

	只读	互斥	共享
无权限	——	——	——
可读	只读	只读	只读
可写	只读	互斥	共享

从表 1 可以看出：当用户对项目元素无权限时候，不论项目元素被其他用户以何种方式检出，该用户对该项目元素均无检出权限。当用户对项目元素拥有可读的配置权限时，不论其他用户以何种方式对该项目元素进行检出，在静态锁和动态锁的共同作用下，该用户对项目元素只能进行只读检出。其他以此类推。

静态锁和动态锁的联合应用，可以有效地将那些目前暂时不可与其他用户共享的修改成果进行隔离，也可以将那些绝对

不可与其他用户共享的修改结果进行隔离。

3 结束语

该文提出了基于过程的、管理粒度为项目元素的版本管理系统的实现策略。与传统的配置管理工具相比，具有如下特点：

(1)WINGCM 的版本管理系统中，使用了多层次的资源库，支持软件复用技术在不同范围内、不同层次上的应用；同时，WINGCM 使用了现代成熟的数据库技术，加入一系列的封装和抽象机制，将项目元素和它们的元信息进行分离，支持可变粒度的项目元素级的版本管理。版本管理的对象可以大到整个项目，可以是传统意义上的构件，也可以小到某个单一的文件。这就使得版本化的对象在逻辑上更具有可理解性，用户能够快捷、方便、准确地了解软件在特定时刻的组成情况；也有利于满足不同用户对于版本管理的不同要求：项目管理者对项目的管理要求，程序开发人员对项目的开发要求。

(2)WINGCM 的版本管理建立在过程的基础上，操作具有明确的逻辑意义。WINGCM 提供了基本过程框架，对项目的控制和操作都以过程为基础。用户可以根据组织或者项目的具体需要，在基本过程框架的基础上对软件开发过程进行具体化[6]。由于每一过程阶段都有良好定义（具有开发进度、产品质量等目标的定义），项目管理者能够更好地把握项目开发进度，有利于提高软件产品的质量和生产率。

(3)WINGCM 在有效的版本管理基础上，提供了权限控制管理，以支持小组协作和并行开发。

(4)WINGCM 提供了较高自动化程度的版本管理，用户易于理解，易于操作。WINGCM 提供的版本管理功能，其实现对用户来说是透明的，并且，WINGCM 对于软件过程在很大程度上实现了自动控制。 （收稿日期：2002 年 9 月）

参考文献

1.A Brown，S Dart，P Feiler et al.The State of Automated Configuration Management[M].SEI，CMU，1991

2.朱三元，钱乐秋，宿为民.软件工程技术概论[M].北京：科学出版社，2002-01

3.Walter Tichy.Software Configuration Management Overview.http://www.ida.liu.se/~petfr/princprog/cm.pdf

4.James J Hunt，Kiem-Phong Vo，Walter F Tichy.Delta Algorithms：An Empirical Analysis.http://citeseer.nj.nec.com/hunt98delta.html

5.Peter H Feiler.Configuration Management Models in Commercial Environments[M].SEI，CMU，1991-03

6.Ivar Jacobson，Grady Booch，James Rumbaugh 著.周伯生等译.统一软件开发过程[M].北京：机械工业出版社，2002-01

7.Reidar Conradi，Berhard Westfechtel.Version Models for Software Configuration Management.1996-10-25

8.Walter F Tichy.RCS-A System for Version Control.http://citeseer.nj.nec.com/tichy91rc.html

---

（上接 67 页）

IOPCItemProperties-这个接口可以使客户在不向服务器添加项的情况下获得项的属性。

IConnectionPointContainer-此接口是连接点容器，使客户可以获得连接点，可以得到数据回传、关闭 OPCSERVER 条件。

IOPCBrowseServerAddressSpace（optional）-使客户可以获得 SERVER 所支持的所有项的名字和组织方法。

OPC 组接口说明：

OPCGroup 对象有 7 个接口，其中 6 个是必须的。OPCGroup 本身没有特殊的接口。当客户创建组时，它获得组的接口（7 个接口之一）。由于任何一个接口都可以从同一对象上的任何其它接口上查询，所以客户可以获得任一接口并保持它。

IUnknown-所有 COM 对象都支持此接口。

IOPCGroupStateMgt-获取并设置组参数，例如名字，更新频率，活动状态。

IOPCItemMgt-负责添加项、删除项和设置项参数。可以通过 ItemAttributeEnumerator 来列举项。

IOPCSyncIO-在多个项上执行同步读写。

IOPCAsyncIO2-在多个项上进行异步读写和刷新。当操作完成时通知连接点。

IConnectionPointContainer-这个标准接口是连接点容器，使客户登记一个数据订阅事件。

IOPCAsyncIO-在多个项上进行异步读写和刷新。当操作完成时通知数据槽。

IDataObject-标准的 COM 接口，登记接收通知的接口槽。

4 实现和利用 OPC 数据访问服务器的优势

OPC 规范以 OLE/DCOM 为技术基础，而 OLE/DCOM 支持 TCP/IP 等网络协议，因此可以将各个子系统从物理上分开，分布于网络的不同节点上。

OPC 按照面向对象的原则，将一个应用程序（OPC 服务器）作为一个对象封装起来，只将接口方法暴露在外面，客户以统一的方式去调用这个方法，从而保证软件对客户的透明性，使得用户完全从低层的开发中脱离出来。

OPC 实现了远程调用，使得应用程序的分布与系统硬件的分布无关，便于系统硬件配置以及，使得系统的应用范围更广。

采用 OPC 规范，便于系统的组态化，将系统复杂性大大简化，可以大大缩短软件开发周期，提高软件运行的可靠性和稳定性，便于系统的升级与维护。

OPC 规范了接口函数，不管现场设备以何种形式存在，客户都以统一的方式去访问，从而实现系统的开放性，易于实现与其它系统的接口。 （收稿日期：2002 年 9 月）

参考文献

1.OPCFoundation.OPCDataAccessCustomInterfaceStandard[S].Version2.03，1999-07-27

2.OPCFoundation.OPCOverview.Version1.0，1998-10-27

3.潘爱民.COM 原理与应用[M].北京：清华大学出版社，1999

4.Corry，Mayfield，Cadman.COM/DCOM 编程指南[M].北京：清华大学出版社，2000

5.TheComponentObjectModelSpecification[S].MicrosoftCo.&DEC，1995