基于 COM+的协同集成设计环境下用户管理模型*)

马 勇[1,2] 张玉清[2] 孟 波[1]

(武汉大学计算机科学与技术学院 武汉 430072)[1]

(中科院研究生院国家计算机网络入侵防范中心 北京 100039)[2]

摘 要 近年来,计算机支持的协同设计(CSCW)已经成为一项热门的研究话题。本文提出了协同设计环境下用户管理的模型,探讨了不同逻辑网络结构,特别是在对等网络结构下,用户动态加入和退出的算法。通过使用 COM+技术实现了客户和服务器位置的透明性、角色的划分和用户之间的安全交互等功能。实验结果进一步验证了设计的正确性和有效性。

关键词 计算机支持的协同设计,用户管理模型,COM+技术

COM+-Based User Management Model for Collaborative Design Environments

MA Yong[1,2] ZHANG Yu-Qing[2] MENG Bo[1]

(School of Computing Science and Technology, Wuhan University, Wuhan 430072)[1]

(National Computer NetWork Intrusion Protection Center, GSCAS, Beijing 100039)[2]

Abstract In the recent years, CSCW has become a hot research. The algorithm that users can dynamically join and exit the task is discussed in Client/Server and especially in peer to peer, and designed. Role and security are put forward in this paper. Furthermore location transparency, role and interaction of the peers are carried out via the COM+. The validity and correctness of the user management model has been proved and testified by an experiment.

Keywords CSCW, User management model, COM+ technology

1 引言

现代的许多应用系统和产品的设计与开发非常复杂,往往需要很多人员同时参与,并彼此协作才能完成。为此,人们研究了 CIDES(Collaborative and integrated design environments),它为产品设计和开发提供了一个计算机支持的在线集成开发环境。IDES 经历了三个发展阶段,包括本地集成、分布集成和协同集成。传统的 IDES 系统只能在本地运行,一次只能由一个用户使用,提供了一个人-机交互的接口。后来,人们借鉴分布式数据库、分布式操作系统的思想,开发了一系列具有分布式功能的集成开发环境,各个设计人员通过文件共享的方式交流设计结果。文件共享只是提供了简单的分布式功能,不能支持用户在线协同的产品设计。因此,开发支持在线协同 IDES 成为一种迫切的需求,它能提供一个人-人交互的接口来达到多个用户协同完成设计任务。

实时群组编辑已经成为一个协同集成设计环境的一个典型应用[1]。自从 CSCW 概念诞生以来,已经出现了许多群组编辑系统,例如 GROVE[2]、REDUCE[3]。在这个领域的研究者,提出了一系列的操作转换算法[4]。这些算法可以很好地执行本地响应、一致性保持、灵活地 UNDO[5] 操作等协同集成开发环境下的关键技术,同时开发了许多关于协同编辑的工具(如白板工具[7,8])。同时,出现了许多商业应用软件,如 Oval 和 Lotus Notes 系统。它们通过共用系列文档,使用群件技术来支持用户之间的互操作,其文档管理系统通过访问和版本控制、文档研究、状态跟踪等技术促进团队合作。这些环境集成了以协同为核心的通信和工作流机制[9]。近年来,人们在把现有的单机版的商业软件改造成多用户协同工作的软件方面做了不懈的努力,并且取得了很大的成功,例如现有的绘图软件、CAD 软件、Office 办公软件[10]。

上世纪 90 年代后期,PC 机的性能在速度和处理能力上突飞猛进。人们开始意识到如果把以前各个独立工作的 PC 组织起来协同工作,各个 PC 既可以并行工作,也可以促进参与者之间交流,迅速提高工作效率。2000 年用于共享 MP3 音乐的 Napster 软件与美国唱片界的一场官司将 P2P 技术重新带回人们的视线。继 Napster 之后,各种基于对等网的应用风起云涌。文件交换方面比较有代表性的有 Gnutella、FreeNet 等;对等计算方面有著名的 Distribute0. Net 案例,利用 100,000 台分布在互联网上的 PC 机对 56 位 DES 数据加密算法进行了强力攻击;协同工作方面有 Intel 内部处理器开发工具 Netbatch,允许工程师们利用没有使用的 10000 台遍布全球的 Intel 公司工作站为芯片的设计执行计算机仿真,在 10 年里已经为 Intel 节省了约 5 亿美元。互联网最基本的 TCP/IP 协议并没有客户端和服务器的概念,在通讯过程中,所有的设备都是平等的一端。P2P 正好体现了对等思想,因此如果把 P2P 技术应用到产品设计领域,协同集成设计环境下用户工作效率会得到明显提高。

以往的方法主要集中讨论了协同集成环境下如何实现一致性、软件的执行效率等问题。然而,却很少讨论在不同的网络结构,尤其是对等网络结构下用户自由加入和退出项目的算法,也很少涉及用户角色的划分、位置透明性、安全等同样

*)Supported by the National Natural Science Foundation of China under Grant No. 603773040(国家自然科学基金);the National High-Tech Research and Development Plan of China under Grant Nos. 863-317-01-04-99,2003AA142150(国家高技术研究发展计划(863))。马 勇 博士生,主要研究方向为网络与系统安全;张玉清 副研究员,硕士生导师,主要研究方向为网络安全;孟 波 教授,博士生导师,主要研究方向为信息技术与决策支持系统。

重要的课题。本文对此进行了详细探讨，给出了在不同网络结构下协同集成设计环境的用户管理模型，并且对协同集成设计环境下的用户管理关键技术进行了讨论，主要包括：

① 在不同的网络结构下，用户自由选择、加入、退出讨论；

② 同一个讨论小组内实现位置透明的用户管理；

③ 用户角色的划分和访问安全的设计。

2 定义

本节给出了用户管理模型设计中所涉及的相关定义和一些基本性质，主要包括：用户、用户集合、用户不同连接方式的定义，在此基础上给出了集中模式和对等模式应当满足的条件和具有的性质，以及这些定义的数学表达形式。

定义 1 设 T 代表讨论，N 是加入讨论 T 的人数，u_i 是第 i 个加入讨论 T 的用户，则加入讨论 T 的用户集合 U_N^T 可以定义为：$U_N^T=\{u_p \mid 0<p\leqslant N\}$，显然，$u_1$ 是讨论创建者。

定义 2 设 $i\neq p$，u_i 是第 i 个加入讨论 T 的用户，则有 $u_i \Rightarrow u_p$，表示建立了用户 u_i 到服务器 u_p 的连接。$u_i \Leftrightarrow u_p$ 表示 $u_i \Rightarrow u_p$ 和 $u_p \Rightarrow u_i$ 同时成立，u_i 和 u_p 是点到点的对等模式下的用户，既是服务器又是客户端。$u_p \Leftrightarrow u_p$ 表示建立了自连接。

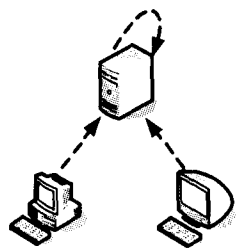

图 1 客户/服务器集中模式

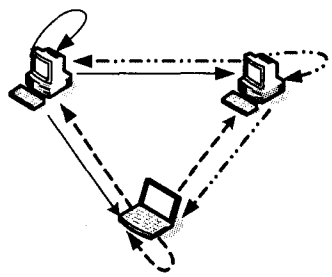

图 2 点到点对等模式

定义 3 客户/服务器集中模式满足以下条件：

$$u_i \Rightarrow u_1 \ (1\leqslant i\leqslant N) \tag{1}$$

$$\overline{\exists}(u_i \Rightarrow u_p)(1\leqslant i\leqslant N, 1<p \text{ or } p>N) \tag{2}$$

定义 3 显然具有如下性质：

U_N^T 存在 N 条连接。 (3)

图 1 形象、直观地说明了客户/服务器集中模式网络结构。

定义 4 点到点对等模式满足以下条件：

$$\forall(u_i,u_p)\rightarrow u_i \Leftrightarrow u_p \ (1\leqslant i\leqslant N, 1<p \text{ or } p>N) \tag{4}$$

$$\overline{\exists}(u_i \Rightarrow u_p)(1\leqslant i\leqslant N, 1<p \text{ or } p>N) \tag{5}$$

定义 4 显然具有如下性质：

U_N^T 存在 N^2 条连接。 (6)

图 2 形象、直观地说明了点到点对等模式网络结构。

3 用户管理模型的总体设计

以前服务器的创建“讨论”，客户连接到服务器大多采用 TCP/IP 协议的 SOCKET。这样，客户需要知道服务器的地址或名字，在点到点对等模式下用户的管理更加困难，很难达到人与人之间流畅、自由交互和站点的分布透明性要求。而基于组件、中间件、群件技术可以很好地满足 CIDES 对用户管理的要求。现有的组件技术主要是微软开发的公共对象模型(COM)和 OMG 布的公共对象请求代理结构(CORBA)。COM+的多套件、例集和解例、远程过程调用等技术实现了分布式环境下群组通信。COM+通过定期 Ping 和引用计数在一定程度上实现了组件的生命周期管理，集成了 Windows 操作系统的安全机制和分布式环境下远程过程调用安全机制。我们采用 COM+组件技术，设计了一个方法(UserMangeMod)对 CIDES 下用户管理模型进行探讨和研究。

由于对等模式包含了多个集中式结构，本文仅仅讨论对等模式下 UserMangeMod 的用户创建和加入讨论的过程，以及总体结构图。UserMangeMod 由多个 Peer 和一个 TaskBroker 组成，Peer 由 Server 和 Client 组成。TaskBroker 负责建立讨论主题和协调建立组内 Peer 之间的连接，产生全局操作序列。Client 负责接收用户的输入，从 TaskBroker 接收一个全局的命令序列号，向本地 Server 发出命令调用，同时也执行来自组内任何一个 Server 的调用，由协调控制器根据全局操作序列在 Client 按序绘制输出。所有的 Peer 都属于一个 Task 任务组，每一个 Peer 可以在 Task 组内创建一个新的小组，也可以选择参加自己感兴趣的小组。在同一个小组内，任何一对 Client 和 Server 之间都存在一个连接。图 3 显示了建立讨论和用户加入的流程，图 4 说明了对等模式下用户管理模型的结构图。任何一个 PeerX 选择讨论、加入讨论的步骤如下：

(1)ServerX 通过接口 IRegisterSer，在 TaskBroker 注册自己，可以创建讨论话题。

(2)ClientX 连接到 TaskBroker，ClientX 通过接口 Italk 请求加入讨论，TaskBroker 通过 ServeX 向该 ClientX 发送组内所有的 Server 的指针，同时通过其它的 Server 向其对应的 Client 发送 ServerX 的指针。

(3)Client 查找所有 Server 的连接点，建立到 Server 连接点的连接。Client 和 Server 建立连接后，不再通过 TaskBroker，而进行直接通信。

(4)Client 通过接口 IsendCommand 向 Server 发送命令。

(5)Server 在收到 Client 发来的命令后，通过接口 IreceiveCom，调用 Client 的命令。

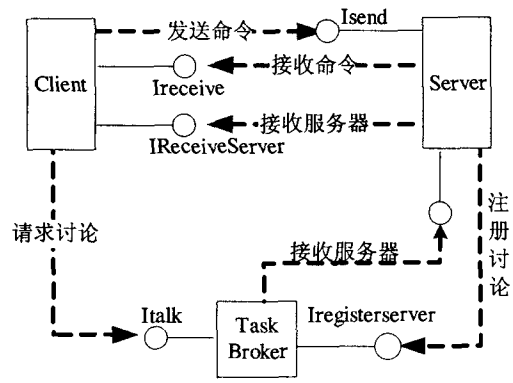

图 3 建立讨论和用户加入流程

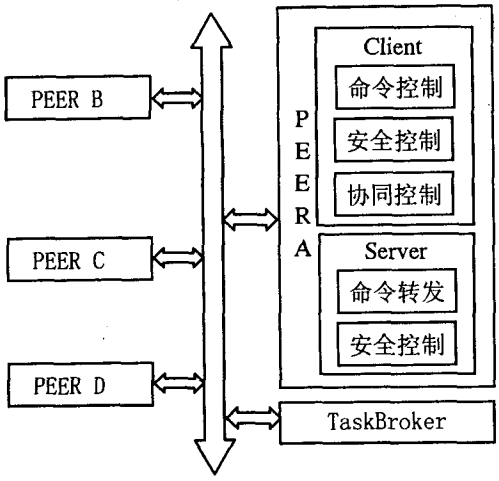

图 4　用户管理模型的结构图

4　关键技术的设计

透明的用户位置、用户的自由加入和退出讨论，以及角色和安全机制等关键技术是用户管理的核心内容。下面详细地讨论这些关键技术的实现。

4.1　用户位置透明性

COM 组件通过接口实现组件的相互连接和相互调用，而所有的接口由 IUnknown 继承而来。IUnknown 接口包括三个成员函数：QueryInterface，AddRef，Release。对象调用接口时，通过调用 AddRef 增加接口的引用计数；对象调用接口时，Release 函数用来减少接口的引用计数，表明对象不再调用该接口了；QueryInteface 函数查询该接口提供的功能。在上述用户管理模型和流程步骤(1)～(3)里，Client 和 Server 通过 TaskBroker 进行连接，Client 和 Server 并不知道对方的地址，实现了 Client 和 Server 位置透明性。位置透明性拉近了客户和服务器的距离，让用户感觉服务器距离客户足够地“近”，运行在服务器上的组件就像运行在本地一样。

COM 组件通过连接点机制可以实现客户和服务器的双向通信和调用。上述用户管理模型和流程步骤(4)说明 Client 通过接口 ISend 向服务器发送命令，步骤(5)说明 Server 通过接口 IReceive 向所有连接到该连接点的 Client 发送命令，很好地实现了 Client 和 Server 动态通信，从而为 Client 之间自由流畅的人-人交互和人-机交互的实现提供了支持。

4.2　用户的动态加入和退出

4.2.1　客户/服务器集中模式下用户动态加入和退出

(1)算法(C-S-J)描述了在集中模式下用户动态加入。

如果 u_i 是 u_1，u_1 是服务器

u_1 向 TaskBroker 注册自己，建立 $u_1 \Rightarrow u_1$　　C-S-J-1

否则　u_i 从 TaskBroker 获取 u_1 指针，建立 $u_i \Rightarrow u_1$　　C-S-J-2

(2)算法(C-S-E)描述了在集中模式下用户动态退出。

如果 u_i 是 u_1，u_1 是服务器

取消 $\forall u_i \Rightarrow u_1 (1 \leqslant i \leqslant N)$，$u_1$ 向 TaskBroker 取消注册　　C-S-E-1

否则　取消 $u_i \Rightarrow u_1$　　C-S-E-2

很显然，算法(C-S-J)和算法(C-S-E)符合定义 3。

4.2.2　点到点对等模式下用户动态加入和退出

(1) 算法(P-P-J)描述了在对等模式下用户动态加入。

如果 u_i 是 u_1，u_1 是服务器

u_1 向 TaskBroker 注册自己，建立 $u_1 \Rightarrow u_1$　　P-P-J-1

否则　u_p 向 TaskBroker 注册自己，从 TaskBroker 获取 $\forall m\{u_m \mid 0<m \leqslant N\}$ 指针，建立 $u_m \Rightarrow u_p$。　　P-P-J-2

$\forall m\{u_m \mid 0<m \leqslant p\}$，建立 $u_p \Rightarrow u_m$。　　P-P-J-3

建立 $u_p \Rightarrow u_p$　　P-P-J-4

算法(P-P-J)满足定义 4 的两个条件(4)和(5)，同时满足性质(6)。可以证明如下：

当用户 u_1 加入时，执行 P-P-J-1，很明显满足定义 4。假如 U_k^T，算法(P-P-J)满足定义 4，那么用户 u_{k+1} 加入时，执行 P-P-J-2；建立 $\forall m\{u_m \mid 0<m \leqslant K\} u_m \Rightarrow u_{k+1}$，生成了 K 个新的不同连接，执行 P-P-J-3；建立 $\forall m\{u_m \mid 0<m \leqslant K\}$ $u_{k+1} \Rightarrow u_m$，生成了 K 个新的不同连接，执行 P-P-J-4；建立 $u_{k+1} \Rightarrow u_{k+1}$，生成 1 个新的连接。如果 U_{k+1}^T 不满足定义 4 的要求，只可能是 $\exists u_m (0<m \leqslant K+1)$ 和 u_{k+1} 不满足 $u_m \Leftrightarrow u_{k+1}$，否则和假设 U_k^T 相矛盾。如果不满足 $u_m \Rightarrow u_{k+1}$，与 P-P-J-2 相矛盾；如果不满足 $u_{k+1} \Rightarrow u_m$，与 P-P-J-3 相矛盾；如果不满足 $u_{k+1} \Rightarrow u_{k+1}$，与 P-P-J-4 相矛盾。所以，$u_{k+1}$ 加入到 U_k^T，U_{k+1}^T 满足定义 4 的要求。连接数目：$K^2+K+K+1=(K+1)^2$ 个连接，满足性质(6)。

(2)算法(P-P-E)描述了在对等模式下用户动态退出。

在对等模式下如果讨论的创建者 u_1 退出，u_2 成为 u_1。

如果 u_i 是 u_1，u_1 是服务器

u_1 从 TaskBroker 取消注册，对于 $\forall m\{u_m \mid 1<m \leqslant N\}$，取消 $u_m \Rightarrow u_1$。　　P-P-E-1

对于 $\forall m\{u_m \mid 1<m \leqslant N\}$，取消 $u_1 \Rightarrow u_m$，u_m 成为 u_{m-1}，$N=N-1$。　　P-P-E-2

取消 $u_1 \Rightarrow u_1$。　　P-P-E-3

否则

u_i 从 TaskBroker 取消注册，对于 $\forall m\{u_m \mid 1 \leqslant m \leqslant N$ 且 $m \neq i\}$，取消 $u_m \Rightarrow u_i$。　　P-P-E-4

对于 $\forall m\{u_m \mid 1 \leqslant m \leqslant N$ 且 $m \neq i\}$，取消 $u_i \Rightarrow u_m$，u_m 成为 u_{m-1}，$N=N-1$。　　P-P-E-5

取消 $u_i \Rightarrow u_i$。　　P-P-E-6

利用同样的方法，可以证明算法(P-P-E)也是正确的。

4.3　角色和安全机制

根据上面的讨论，无论在客户/服务器集中式网络结构还是点到点对等式网络结构，服务器创建讨论并且发布服务，客户端加入讨论，订阅服务。用户可以划分成两种角色：发布者、订阅者。在客户/服务器集中式网络结构下，服务器创建讨论，发布服务，承担发布者角色。各个客户端加入讨论，订阅服务，承担订阅者角色。在点到点对等模式下，每个用户既是服务器，发布服务，担当发布者的角色；又是客户端，订阅服务，担当订阅者的角色。在这两种不同的网络结构下，用户角色的划分正好反映了这两种网络结构各自的特点。客户访问服务器需要得到服务器的授权，服务器回调客户端同样需要扮演客户。根据访问权限的大小划分为不同的等级：表 1 说明了客户端访问服务器端的访问安全性，表 2 显示了服务器访问客户端的激活安全性。

通过分别在客户端和服务器端实施不同的安全性等级，有效地解决了客户在安全等级的权限内访问服务器。服务器访问客户端时同样需要得到客户端的认证和授权，避免了客户端收到恶意的攻击。整个系统具有了很好的安全性。

5　系统实现

通过对现有 CAD 软件的封装和改造，使其具有协同设计

的功能;多个用户在协同集成环境下进行协同设计来检验CIDES下用户管理模型的效用性。

表1 客户端访问服务器端的访问安全性

等级	权限
None	认证没有被使用
Connect	客户将通过初始的连接被认证
Default	认证将使用缺省的认证级别
Call	每一次调用都会被认证
Packet	每一个包都会被认证
Integrity	每一个包都会被认证并确认数据在传输过程中没有被修改
Privacy	包含完整性所做的全部事情，并且同时所有的都被加密了

表2 服务器访问客户端的激活安全性

等级	权限
Anonymous	客户是匿名的，服务器不能获得客户的身份确认，扮演客户
Identify	可以扮演客户的角色，唯一的目的是获得客户的身份认证
Impersonate	可以扮演客户，同时也可以替代客户
Delegate	完全扮演客户，同时能够访问客户可访问的网络资源

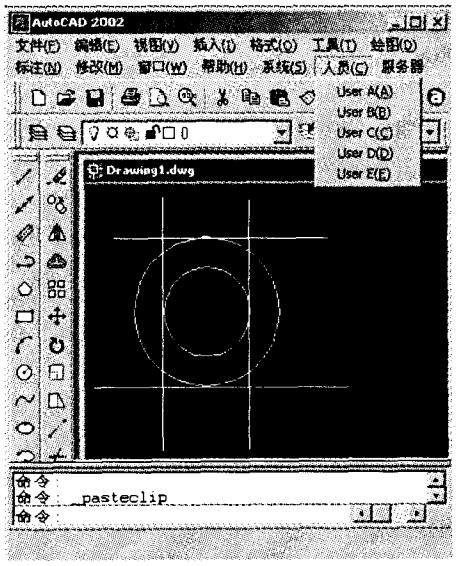

图5 AutoCAD 2002协同工作示意图

用户User A创建了一个生成几何形体的任务，从而在该任务里承担创建者的角色。用户User B、User C、User D、User E加入该任务，承担工程师角色。User A创建了两个同心圆，User B、User D分别描绘了两条和大圆相切的平行直线，User C、User E分别绘制了两条和小圆相切的平行直线。图5显示了在AutoCAD 2002环境下多用户协同开发的运行实例。在Windows任务管理器进程标签里显示的DrawServer是客户端的代理(Proxy)，同时DrawServer进程支持用户进行协同设计。不同DrawServer相互连接，构成了一个协同总线，每个DrawServer进程是该总线上不同的插槽，很好地实现了客户端的软件的热插拔，很好地达到了基于组件的协同集成设计环境下用户管理的目标。

总结 本文探讨了服务器创建"讨论"以及在客户端和服务器之间建立连接的过程，分析了客户和服务器的位置透明性原理和实现，提出了在集中和对等两种网络结构下用户动态的加入和退出算法，对角色进行划分并且确立了客户端、服务器的安全机制，最后实现了在CSCW系统下的用户管理。但是在广域网实际系统里，往往采用的是集中式结构和对等结构相结合的混合结构，在混合结构下的用户管理仍需要讨论。另外，本文提出的安全机制是基于接口级的安全机制，不同的用户对接口下面的函数需要拥有不同的等级，这也是在以后的工作中需要加以改进和完善的地方。

参考文献

1 Dewan P, Choudhary R, Shen H. An editing-based characterization of the design space of collaborative applications. Journal of Organizational Computing, 1993, 4(3): 219～240

2 Ellis C A, Gibbs S J. Concurrency control in groupware systems. In: ACM SIGMOD'89 preceedings, 1989. 399～407

3 Sun C, Jia X, Zhang Y, et al. Achieving convergence, causality-preservation, and intentionpreservation in real-time cooperative editing systems. ACM Transactions on Computer-Human Interaction, 1998, 5(1): 63～108

4 Sun C, Ellis C. Operational transformation in real-time group editors: issues, algorithms, and achievements. In: Proceedings of ACM CSCW '98 Conference, 1998. 59～68

5 Sun C. Undo any operations at any time in group editors. In: Proceedings of ACM CSCW '2000 Conference, 2000

6 Grudin J. Eight challenges for groupware developers. Communications of the ACM, 1993, 37(1): 92～105

7 Moran T P, Palen L, Harrison S, et al. I'll get that of the audio. A case study of salvaging multimedia meeting records. In: Proceedings of CHI '97, 1997. 202～209

8 Fox A, Johanson B, Hanrahan P, et al. Integrating information appliances into an interactive workspace. IEEE Computer Graphics and Applications, 2000, 20(3): 54～65

9 Malone T W, Lai K-Y. Toward intelligent tools for information sharing and collaboration. In: Bostrom R P, Watson R T, Kinney S T, eds. Computer Augmented Teamwork: A Guided Tour. New York, Van Nostrand Reinhold, NY, 1992

10 Campbell J D. Collaboration Transparency Workshop: Experience. wit