多媒体支持的协同系统开发工具集(MediaCoKit)的设计与实现*)

The Design and Implementation of MediaCoKit

宗祥宇 魏明亮 吴敏强 茅 兵 谢 立

(软件新技术国家重点实验室 南京大学计算机科学与技术系 南京210093)

Abstract We illustrate MediaCoKit, a multimedia collaborative system development toolkit, in this paper. The collaborative system, developed with MediaCoKit, has a friendly GUI and a good accessibility for users. And group awareness is supported in terms of multimedia presentation, the presentation including both video and audio media. We discuss the key issues of the toolkit such as data sharing object, multimedia components, workspace awareness and then detail its implementation.

Keywords Multimedia, Collaborative systems, Workspace awareness, Data sharing object

1 引言

计算机支持的协同工作系统是一种人-人交互的群组工作系统,其设计除了考虑通常应用系统的功能设计外,还要着重考虑用户工作的协调一致和用户间的行为感知方式的设计。因此,协同系统的开发向开发者提出了更高的技术要求。我们设计并实现了一个多媒体支持的协同系统开发工具集(MediaCoKit)。基于本工具集我们开发了协同应用环境 Ashram、Asharm 具有友好的界面和简单的操作方式,且支持多媒体的协同感知。

2 相关工作

Rendezvous[1]是由 Bellcore(现称 Telcordia)公司开发的一个 CSCW 程序设计工具集,用于构造同步、图形界面的群件应用程序,Rendezvous 采用 Common-Lisp 书写,在 X 窗口系统上运行。Rendezvous 的核心是其 ALV(Abstraction-Link-View,抽象-链接-视图)结构。抽象是应用程序的共享语义部分,视图提供给各个用户以不同的界面和交互机制。链接是一种约束系统,起着联系抽象与视图的作用,负责将视图中用户界面的操作转换为对抽象的恰当修改,反之,当抽象发生变化时,负责相应地更新视图。Rendezvous 采用这种基于约束的机制来管理视图与抽象之间的所有通信。

Habanero[2]是伊利诺斯大学国家超级计算中心(NCSA,National Center for Supercomputing Applications)开发的一个同步协同系统框架和环境。通过采用 Habanero API 共享 Java 对象,在各个客户节点复制应用程序,Habanero 框架提供了跨平台的多用户 Java 软件的状态和事件同步,也提供会话管理的支持,包括创建、加入、离开、浏览会话的功能。Habanero 中还包括协同应用工具集,提供远程协同的基本功能,包括表决器、文本和语音聊天工具、文本编辑器、白板等。

现有的开发工具集通常仅支持协同系统设计的某个方面,或者仅支持某一领域的协同系统,未充分考虑工具集对协同工作系统快速反应性和对工作空间感知一致性的有效支持。因此,我们的工作集中在:研制支持灵活的协同模式和多种粒度的协同感知的协同策略控制模型,以及为协同系统的开发提供通用的方便的多层次开发工具,并且提供有多媒体支持的工作空间感知处理构件。

3 工具集的详细讨论

我们的协同软件开发工具集以 Java 实现,Java 语言具有平台无关、安全性好、支持多线程等特点,以及"一次编译,到处运行"的能力,使其适合于跨平台异构型网络应用。我们采用半复制式的体系结构(如图1)构筑协同系统。将协同系统划分为较为独立的三种节点

*)本文研究得到国家863高科技项目"实用化的 JAVA 开发环境的开发"(863-306-ZT02-03-01)的资助。宗祥宇 硕士研究生,研究方向:分布式多媒体系统,分布式计算,CSCW。魏明亮 硕士研究生,研究方向:分布式计算,分布式多媒体系统,CSCW。吴敏强 硕士研究生,研究方向 分布式多媒体系统,分布式计算,CSCW。茅 兵 副教授,研究方向:CSCW,分布式计算。谢 立 教授,博士生导师,研究方向:分布式计算与并行处理。

进行开发，即中央节点、应用服务器节点、用户节点，用户节点上运行应用程序的复制部分，并且与应用程序的集中部分通信。同时各个应用程序的复制部分有可能直接通信，即并非所有网络通信都由服务器转发。用户节点上还运行有会话管理程序，与中央节点的注册器程序通信。应用服务器节点是协同会话的集中节点，会话管理程序(session manager)和应用程序的集中部分运行在该节点上。会话管理程序负责会话的创建、删除、管理、加入和退出，并且将本会话的信息发送给注册器程序。中央节点是一个知名节点(well-known site)，其上运行注册器程序，负责为协同会话提供加入的门户，起着一个最初接入点的作用。

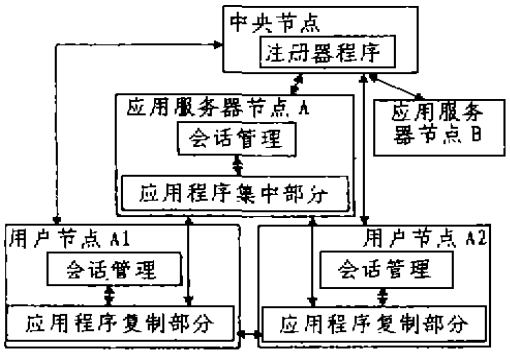

图1 协同系统运行时的体系结构

该工具集包括通信基础工具包、会话管理工具包、共享数据对象工具包、多媒体构件、工作空间感知处理工具包、共享文件工具包、协同界面构件，下面分别予以介绍。

3.1 通信基础工具包

通信基础工具包用于构造协同系统的底层通信结构，采用 TCP Socker 通信。在图1中，各节点间的通信有以下几类：应用服务器与用户节点间的通信、应用服务器与中央节点间的通信、用户节点与中央节点间的通信。这三种节点间的通信在设计与实现方式上都基本相近，因此我们只对应用服务器与用户节点间的通信进行介绍。该工具包中主要的类有：消息代理类 MessageAgent；客户代理类 ClientAgent（MessageAgent 的子类）；会议代理类 ConferenceAgent（ClientAgent 的子类）；服务器线程类 ServerThread；服务子线程类 ServerPeerThread（与 ClientThread 相对，为其服务的线程类）；服务子线程接收线程类 ReceiverServerThread（与 ServerPeerThread 形成一对）；客户线程类 ClientThread（与 ServerPeerThread 相对，是其客户线程类）；客户接收线程类 ReceiverClientThread（与 ClientThread 形成一对）。

如图2所示，服务器线程 ServerThread 在会话的服务器节点上运行，客户通过客户线程 ClientThread 与服务器连接，服务器端接受连接后，对于每一个客户

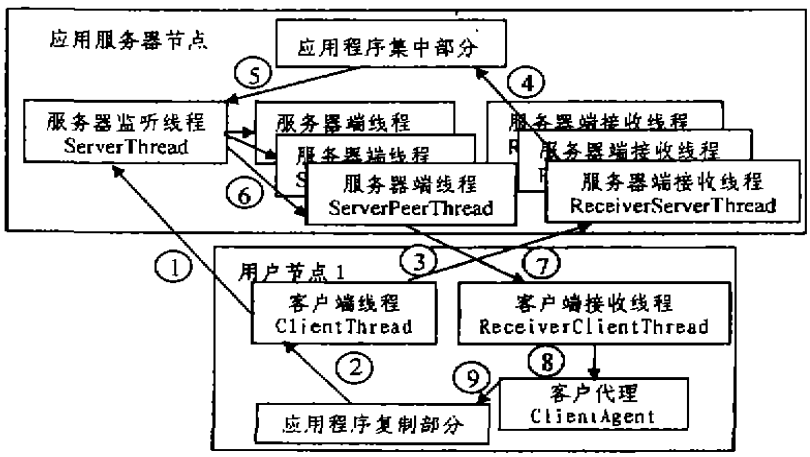

图2 节点间通信关系

有一个服务子线程 ServerPeerThread 负责与其通信，同时，与每一个 ServerPeerThread 配对有一个服务器接收线程 ReceiverServerThread，专门接收从对应客户发送来的消息；与每一个 ClientThread 配对也有一个客户接收线程 ReceiverServerThread，专门接收从对应服务器发送来的消息，对于接收到的消息由 ConferenceAgent 负责处理，其流程如下：

1）ClientThread 向 ServerThread 请求建立连接，后者创建一对线程 ServerPeerThread 和 ReceiverServerThread 负责与 ClientThread 通信；

2）应用程序复制部分将欲发送的消息传递至 ClientThread；

3）ClientThread 将消息发送至 ReceiverServerThread；

4)ReceiverServerThread 将消息传递至应用程序集中部分进行处理；

5)应用程序集中部分完成一定的计算后，将欲发送的消息传递至 ServerThread；

6) ServerThread 将消息传递至各 ServerPeerThread；

7)各 ServerPeerThread 将消息发送至与其对应的 ReceiverClientThread；

8)ReceiverClientThread 将消息交给 ClientAgent 处理；

9)ClientAgent 对消息进行处理，并调用相应的应用程序功能。

3.2 会话管理工具包

会话管理工具包的主要功能有：

(1)用户管理：用户在客户端进入系统时，先与注册器程序连接，通过身份认证后才加入，并且用户信息加入到在线用户列表中，其他用户可以与之通信，此时用户还未加入任何会话。

(2)会话信息查询：各个应用服务器节点的会话管理程序将本会话的加入用户、工作状态、时间等信息发送给注册器程序，由后者加入到当前会话列表中，方便用户的加入与感知。

(3)设备管理：各个用户的设备信息由注册器程序维护，这样可以实现视频和语音的远程请求连接。

(4)共享文件管理：注册器程序还负责维护共享的文件，实现文件的上传、下传功能。

3.3 共享数据对象工具包

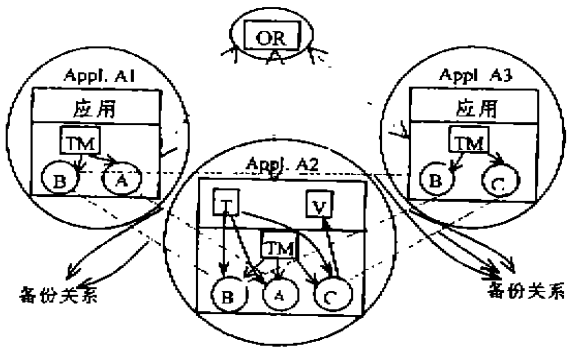

图3 协作共享对象模型结构

如图3所示，协同共享对象模型[7]CSOM(Collaborative Shared Object Model)由对象注册管理器(OR：Object Registrar)、模型对象(Model Objects)、事务处理(T：Transactions)、事务管理器(TM：Transaction Manager)、视域显示(V：View)和协同应用程序等组成。在该模型中，协同应用程序首先向 OR 登记欲与他人共享的对象，并且从中得到其共享对象组信息，之后按某一事务处理(T)激活事务控制对象，执行该事务处理，该事务处理包含对共享对象 A 和 B 的读或更新，由事务控制对象负责其所有共享对象组的并发控制和一致性维护，必要时将共享对象的值在视域中实时显示出来。

对象注册管理器(OR)是协作应用进行对象共享的入口点，负责协助共享对象组的查找、登记和删除，维持整个系统协作应用的所有共享对象的共享关系。

每个模型对象除了保存值信息外还保存有关联(Association)信息，该关联信息包含协作共享对象组成员的位置及标识信息，如成员的端地址、是否主副本和对象标识符。

事务处理是对模型对象的操作处理，由事务处理对象的 execute 方法激活，应用程序员可用 execute 方法和 handelAbort 方法来定义事务处理对象。execute 方法包含了对模型对象进行读和写的程序段，任何对模型对象的操作情况将自动地转发给对象组中的所有副本。handelAbort 方法定义了事务处理夭折时的处理方法。

事务控制对象控制事务处理的执行过程，包括消息的发送、接收、冲突检测，控制事务处理提交、夭折和重执行。

当一个视域对象依附于一个模型对象时，该视域对象通过更新通知消息调用视域更新方法可动态地跟踪模型对象值的变化。模型对象值的视域显示可分为乐观视域显示和悲观视域显示。前者主要为快速反映模型对象值的更新情况而设计；而后者仅为观察已提交的值更新而设计，它反映了某一事务处理执行后，整个系统中该共享对象的值一致性。

该工具包中主要类说明如下：

·类 SharedInt 提供加入共享整型对象、更新和获取共享整型对象值、退出共享等功能。

·类 SharedFloat 提供加入共享实型对象、更新和获取共享实型对象值、退出共享等功能。

·类 SharedString 提供加入共享字符串对象、更新和获取共享字符串对象值、退出共享等功能。

3.4 多媒体构件

这个类库为程序员实现了视频、音频的采集、传送和播放。多媒体构件主要是为多媒体信息在协同应用中的再现提供支持与服务，它包括多媒体存取、多媒体对象和多媒体表现三个层次。

多媒体存取层实现媒体处理所需的各种基本功能，提供媒体特定语义的存取服务，包括音频、视频等媒体数据的采集、播放、压缩、存储及传输等功能。该层利用系统低层提供的通讯和对物理资源存取服务进行信息处理。各种媒体的输入输出设备、数据组织形式以

及处理方式等均不相同，因此通过规范各种媒体数据的组织格式、处理功能等调用要素，以取得服务功能调用接口的一致，从而保证上层服务功能与具体环境无关。

多媒体对象层完成分布式多媒体处理功能的对象化封装，实现多媒体资源的分布式存取服务。多媒体对象服务提供多媒体对象的外部操作接口，实现对象内各种媒体之间的同步时序关系控制以及 QoS 协商控制等服务，为多媒体合成表现提供支持。

多媒体表现层为多媒体支持的协同应用提供多媒体表现的合成机制。按照应用语义要求，通过多媒体对象提供的媒体对象服务进行媒体合成表现。这里合成关系分为对象合成的时序关系和共享工作空间的感知处理。

多媒体构件的层次开发技术具有功能模块化、结构构件化和编程对象化的特点。我们的多媒体构件尽量用 Java 开发，并在调用接口上是 Java 化的，以满足整个系统用 Java 开发的需要。其主要的类包括 AudioStream 类和 VideoStream 类。完成的功能有启动音频视频数据的采样、传输和播放，暂停音频视频数据流和撤销音频视频流，以及取得视频窗口（JPanel）等。

3.5 工作空间感知处理工具包

我们采用文[8]中的感知处理模型。该模型结构主要包括场所映射、感知强度计算、感知表维护和分级感知处理等。需按用户职责范围选择专注对象场所，根据操作对象位置场所及影响场所，计算生成各用户对另一用户的行为感知强度表（简称感知表）。当用户引用对象时，根据感知表信息，分级收集、发送被感知信息，当远程感知处理信息到来时，也根据感知表内容，分级理解表达感知信息。

在多媒体支持的协同编辑器中我们实现了灵活、方便的感知。采用按需传输感知信息，可减少网络的信息流通量，在协作应用系统中，灵活、方便而又自适应的工作空间感知是提高工作组协同效率和加快协作任务进程的重要手段。

在协同编辑器中，我们用 CoEd_Workspace_Awareness 类实现了 Workspace_Awareness 接口，其功能是：分两个等级发送两个工作组成员之间的感知信息，以及一个工作组成员从另一个工作组成员那里接受感知信息并加以表达。即用户不在相邻段落编辑时，即以粗粒度的感知信息来处理，显示其所在哪个段落工作；用户在相邻段落编辑时，即以细粒度的感知信息来处理，除了显示其所在哪个段落工作外，还实时显示其视频信息。

该工具包中主要接口和类说明如下：

·接口 Workspace_Awareness 用于定义工作空间感知处理的等级，以及定义两个工作组成员间发送感知信息和接受感知信息并加以表达的方式。

·类 Working_Member 设置及查询工作组成员所处的协同工作名称和共享工作空间的位置。

·字符空间位置类 textposition 设置及获取字符空间位置。

·图形空间位置类 graphicposition 设置及获取图形空间位置。

3.6 共享文件工具包

针对协同应用中的共享文档的管理与处理，我们提供了共享文件工具包。这个类库实际上是一个集中式的远程文件管理系统。共享文件存储在服务器端，客户端能够实现对共享文件的浏览、上传和下载，并能对共享文件加锁/解锁。我们以类库的形式提供了这个工具包，包括文件上载、下载、加锁/解锁、修改等的 API。

例如文件客户机类 FileClient 提供建立连接、对文件操作和文件加锁/解锁的方法。

3.7 协同界面构件

GUI 工具集给软件开发人员提供了很大的便利，程序员可以方便地将一些精心设计并且已经经过测试和其他人使用过的界面构件加入到自己的程序中来，这些预制的界面构件提供给用户界面可以预制的观感（look and feel），并且可以比较放心地使用，CSCW 软件与单用户软件一样需要用户界面并且通常 CSCW 软件更需要图形界面，因此界面工具集作为 CSCW 软件开发工具的一部分是非常重要的。这种界面工具集可以有两种：一类是单用户界面的群件版本，如文[6]中通过耦合将单用户的按钮，菜单和文本编辑区扩充为多用户的版本；另一类就是专门开发针对群件的界面构件，如 GroupKit 中的成员状态工具、远程指针（telepointer）、多用户滚动条（multi-user scrollbar）和位置感知（location awareness）工具。

我们设计的协同界面构件，通过截取 Java AWT 界面构件的事件，由服务器转发给各协同客户端构件，并由封装好的相应协同界面构件处理这些事件。这些界面构件包括以下类：Co_Applet、Co_Button、Co_Dialog、Co_Canvas、Co_Checkbox、Co_CheckboxMenuItem、Co_CheckboxGroup、Co_FileDialog、Co_Choice、Co_List、Co_FloorMenuItem、Co_Frame、Co_Scrollbar、Co_Panel、Co_TextField、Co_Menu、Co_MenuBar、Co_MenuItem、Co_FloorButton、Co_Window、Co_ExtentionFilter。

4 实例系统

利用本工具集我们开发了一个多媒体协同环境 Ashram，集成了共享白板、协同编辑器、会议室、视频

工具、语音工具、用户管理、会话管理、共享文件等功能，Ashram 系统目前运行在由 Sun Ultra30、IBM RS/6000等工作站和 Pentium PC 等构成的异构分布式网络上（含各种多媒体设备），操作系统包括当前流行的 Windows 98/NT、AIX、Solaris 等。

Ashram 采用图1所示的半复制式体系结构，由3.1节所述的工具包开发其底层通信支持，采用会话管理工具包提供的 API 来管理和显示资源树。资源树中的共享文件功能采用共享文件工具包开发；另外，视频、音频的采集、播放、传输部采用了工具集的多媒体构件。

Ashram 以 Java 实现，使其适合于跨平台异构型网络应用。它主要有以下特点：①支持灵活耦合，允许用户对其与其他用户之间的交互粒度进行配置（如协同编辑器中用户进行的行、段为单位的协同粒度的配置）；②支持基于多媒体信息的组感知；③支持语音、视频及多媒体信息；④支持同步与异步工作的结合。

结论 本文讨论了多媒体支持的协同系统开发工具集的设计与实现。创新之处有：

1）根据用户所处工作位置相关性，采用了灵活的工作空间感知处理机制，特别是融入了实时视频信息，增强了协同工作组的协同感和真实感；

2）在多媒体多路传输中，对数据流和控制流采用清晰的层次结构，构件独立性和可扩展性好，并且将报文控制和 QoS 控制有机地结合起来，尽可能地满足实时性要求；

3）将协同系统中繁琐的协同方式的处理、网络信息处理、多媒体信息处理等过程以 API 类库形式提供给程序员，方便了协同系统开发，有效地缩短了开发周期。

参考文献

1 Hill R，et al. The Rendezvous Architecture and Language for Multi-User Applications. ACM Transactions on Computer-Human Interaction，1994，1(2)：81～125

2 NCSA Habanero，1996. NCSA Habanero Home Page. www.ncsa.uiuc.edu/SDG/Software/Habanero/

3 Dewan P A Tour of the Suite User Interface Software. In Scott Hudson，ed. Proc. of the 3rd ACM SIGGRAPH Conf. on User Interface Software and Technology，ACM，New York，1990. 57～65

4 Abdel-Wahab H，et al. An Internet Collaborative Environment for Sharing Java Applications. In Proc of the 5th IEEE Computer Society Workshop on Future Trends of Distributed Computing Systems（FTDCS'97），Tunis，Tunisia，1997. 112～117

5 Dourish P Using Metalevel Techniques in a Flexible Toolkit for CSCW Applications. ACM Transactions on Computer-Human Interaction，1998，5(2)，109～155

6 Dewan P，Choudhary R. A High-Level and Flexible Framework for Implementing Multiuser User Interfaces ACM Transaction of Information Systems，10(4)，345～380

7 詹永照，茅兵，吴敏强，谢立 协作共享对象模型及其实现技术. 计算机学报

8 詹永照，窦万峰，李成锴，茅兵，谢立. 基于共享对象划分的工作空间感知处理模型. 计算机研究与发展，2000，37(3)：353～358

（上接第20页）

行的子事务组成。往往顶层事务的截止期不能反映子事务的紧急程度，而子事务却需要自己的截止期以便获得合适优先级去竞争资源，这就需要有效的算法来由总截止期导出各子事务的截止期[7]。考虑到各子事务间的执行依赖关系，该问题可分解为两个子问题：

·串行子事务截止期计算 设顶层事务 T 的到达时间记为 AT，截止期记为 DT，宽松时间记为 ST，执行时间记为 ET，$T=\{T_1,T_2,\cdots,T_n\}$，则各个子事务的截止期可用以下公式计算：

$$D_i=AT+\sum_{k=1}^{n-1}E_i+(i-1)\times ST/n$$

·并行子事务截止期计算 公式如下：

$$D_i=AT+\sum_{k=1}^{n-1}E_i+(i-1)\times ST/(k\times n)$$

将这两种方法结合，就可进行一般嵌套子事务的截止期分配。

这只是目前提出的一种粗略的分配方法，实际上，由于子事务间相互依赖的关系很复杂，各个子事务的紧急程度也可能差别很大，我们还需要进行大量的工作以研究更适合于各种不同类型应用特征的方法。

参考文献

1 Ramamritham K. Real Time Database Int. Journal of Distributed and Parallel Database，1992

2 刘云生. 实时数据库系统. 计算机科学，1994，21(3)

3 Kohler W H. A Survey of Techniques for Concurrency Control. ACM Trans. on Database System，1981，6(2)

4 Sha L，Rajkumar R，Lehoczky J. Concurrency Control for Distributed Realtime Databases. ACM SIGMOD Record，March 1988

5 Carey M，Livny M. Distributed concurrency control performance：A Study of Algorithms，Distribution，and Replication. In：Proc. of 4thIntl Conf. on Very Large Databases，August 1998

6 Silberschatz A，Korth H F，Sudarshan S. Database System Concepts U. S. A.：The McGraw Hill Company，1999

7 刘云生，李国徽. 实时数据库嵌套事务的并发控制. 小型微型计算机系统，1998，19