计算机科学2002Vol. 29№. 11

基于 JAVA/CORBA 的曙光3000用户环境的设计与实现*)

The Design and Implementation of Dawning 3000 User Environment Based on JAVA/CORBA

孙耀晖　陈志辉

(中国科学技术大学计算机科学技术系　合肥230027)　(国家高性能计算中心　合肥)

Abstract This paper introduces the ideas for designation and implementation of DUET that is a user-side integrated environment developed for Dawning 3000 super server. The DUET choses the 3-tier client/server distributed objects style architecture proposed by CORBA, and enables users over the Internet access Dawning 3000 super server with a user-friendly interface.

Keywords User environment, Usability, Web, Java, CORBA

1 引言

长期以来,用户总是抱怨使用高性能计算机的困难,当前的使用方式主要采用难用的命令行界面,没有真正面向用户的集成环境和工具软件,因而无法实现良好的可扩展性和可移植性。为了简化其使用复杂度,尤其是在分布式环境下如何方便地使用高性能计算机,目前已提出了几种解决方案,如Connect Queue、LSF、NQE 和 DQS,但仅有少数提供了有限的在客户端使用的 GUI 特性,如 NQE 等。B. Buhlmann 与 H. Bieri 针对传统使用方式的弱点,提出了设计高性能计算机用户友好界面的基本需求,从而使高性能计算(High Performance Computing)培训和教学的代价大大降低[1]。

基于上述原因,我们开发了一套针对曙光3000超级服务器的用户环境,名为 DUET(Dawning3000 User-side-integrated Environment and Tools)。它的目标是简化用户对曙光3000超级服务器的使用步骤,使远程终端用户能够通过 Internet 方便地使用曙光3000。在这个环境中,我们屏蔽了访问超级服务器的具体细节,为用户提供一个基于 Web 浏览器的 GUI 界面,使其能够方便地使用高性能计算机。

本文介绍 DUET 的设计思想和系统结构;描述 DUET 的具体实现细节;对 DUET 的性能进行一些分析;最后给出结论和未来要做的工作。

2 DUET 用户环境的设计

2.1 基本思想

在 DUET 的开发过程中,我们从好用性(Usability)原则出发,结合用户的实际使用,提出了以下指导思想,这些原则对于一个良好的用户环境是至关重要的。

首先,作为一个替代传统使用方式的用户环境,为提高高性能计算机的好用性,DUET 应该具有一个良好的用户接口,同时只需要耗费很小的带宽,并且不能增加高性能计算机的系统负担。用户可以通过低速率的网络连接(如 Modem 连接)在 Web 浏览器中 DUET。DUET 将大部分高性能计算机不必要承担的功能(如各种工具软件的人机交互部分)分离出来在客户端上实现,这将大大减轻多用户使用时对系统资源和网络带宽的大量消耗,使得系统在支持多用户多任务的同时采用图形用户界面真正可行,也使得在采用图形用户界面的同时支持在 Web 异构环境下远程访问曙光3000成为可能,实现桌面的虚拟计算环境。

其次,作为面向 Internet 的应用,DUET 客户端必须是与平台无关的,可以使用在不同的硬件和操作系统环境中,独立于客户端和高性能计算机的硬件和操作系统平台。支持用户在本地各种类型的工作站/UNIX 平台或 PC/Windows/NT 平台上,通过该集成环境直接执行系统管理命令,加载执行应用程序,监视和控制程序运行,实现与高性能计算机的交互。用户可以用同样的方式使用系统提供给用户的其它系统操作和软件开发工具。

同时也要考虑到高性能计算机体系结构的变化和其系统软件、应用程序的升级,DUET 必须是可扩展的。我们希望代码是可通过继承重用的,这样在上述变化发生时,不用修改或只需做少量的修改就能使 DUET 适应新的环境。

另一个需要考虑的因素是安全性。由于客户端和服务器端存在敏感信息的传递,尤其是通过 Internet 访问和使用时,必须保证远程用户和高性能计算机的安全。DUET 环境中的消息传递过程必须有相应的安全机制的保护。

2.2 DUET 的系统结构

基于上述思想,DUET 采用如图1所示的三层 C/S 结构。我们选择 Java 语言和 Swing 组件包实现了 DUET 的客户端,通过 CORBA/IIOP 中间件来完成 DUET 的客户端与服务器端的通信,采用 SSL 保证通信的安全性。

Java 是一种面向对象的、与平台无关的分布式语言[2]。跨平台的可移植性使得纯 Java 程序无需修改和重新编译就能在不同的平台上运行。CORBA(Common Object Request Broker Architecture,公共对象请求代理结构)是 OMG[3]组织制定的分布式应用规范标准,可以提供在 Internet 环境中构造 DUET 所需的可移植性和可扩展性,并且与 Java 技术具有互补性。CORBA 作为 Client 与 Server 对象间的通讯总线,对双方具有透明性,可以使得无需知晓分布式对象的具体所在,就可以访问对象和调用对象上定义的方法。

IIOP(Internet Inter-ORB Protocol)协议是 GIOP(General Inter-ORB Protocol)协议在 TCP/IP 上的实现。GIOP 协议定义了标准消息格式,以及 IDL 数据类型映射到普通消息上的通用数据表示和互操作对象的引用格式[4]。SSL(Secure Sockets Layer Version 3)协议可以建立 Client 和 Server 间的安全连接,提供认证(Authentication)、验证(Identity)、加密(Encryption)等服务[5]。

*)本课题得到863-306项目(合同号为863-306-ZD07-02-3)和中国科学技术大学青年科学基金(编号为 KA1109)及中科院高水平大学建设项目(KY2706)的资助。孙耀晖　硕士研究生,主要研究方向:并行计算,分布式计算。陈志辉　博士研究生,主要研究方向:并行计算,高性能计算。

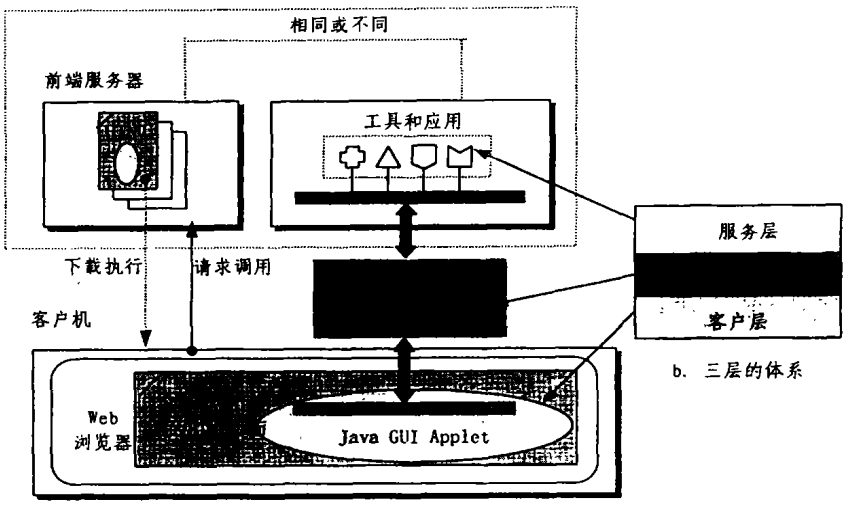

图1 DUET 系统的三层体系结构

DUET 系统采用三层体系结构，基于 CORBA 模型实现了工具的实现逻辑(在高性能计算机上实现)与图形的人机交互界面(面对终端用户)的分离。DUET 的前端集成了各种工具软件的客户部分(客户层)，采用 Java 小程序实现工具的图形用户界面，完成与用户的交互；中间是连接客户和服务部分的通信层 JC-Glue(Java/CORBA based Glue)；后端是随体系结构和操作系统不断变化更新的各种系统工具和应用的实现逻辑(服务层)。

下面将分别对它们作详细的说明：

1)*客户层* 设计曙光3000客户端图形用户界面系统时，我们考察了相应的用户和任务模型，并从好用性的角度出发，把人的认知、行为因素包含到系统的设计中去，使高性能计算机系统的外观和使用特点能适应高性能计算用户的习惯。DUET 的客户层运行在客户端平台上。用户通过 Web 浏览器以一个 URL 来访问服务器端时将其下载到客户端，运行在浏览器内嵌的 Java 虚拟机上。通过这样的环境视图，用户在浏览器中看到一台虚拟的高性能计算机就在自己的桌面上。浏览器中的每个图标或菜单项对应一个系统工具或应用。点击图标即可启动一个工具或应用在高性能计算机上的执行，用户与工具或应用的交互就在这个 Java GUI 小程序中进行。

2)*中间层* 主要实现客户层和服务层之间的信息传递和安全保护。在这里，客户层和服务层以中间层为界，互相隔离开来。我们基于 CORBA 的 IIOP 协议建立了一条软总线 JC-Glue，用于支持创建灵活的和可扩展的系统，方便地集成和配置新的部件。JC-Glue 实现为一个通信界面子系统，提供建立用户桌面系统与高性能计算机连接的统一机制，传递用户使用高性能计算机的请求，返回系统处理请求的结果。JC-Glue 既实现了工具或应用客户层与服务层的连接，也实现了这两部分在实现上的隔离，使得其中任何一方发生变化时对另一方的影响尽可能小。

我们在中间层采用基于 SSL 的安全认证(Authentication)和 UP 身份验证(Username/Password Identity)来解决 DUET 的安全性问题。当一个远程用户欲登录到 DUET 环境中时，服务器端要求客户端提供包含了用户名和口令的加密证书，服务器端收到此证书后，用自己的密钥将之解密并取出用户的身份信息，再与服务器上的 HostDB 或 NIS 数据库中的相应信息比较，通过认证后才允许用户登录[6]。

3)*服务层* 服务层运行在曙光3000高性能计算机上。考虑到硬件平台在体系结构和操作系统两方面可能的发展变化，服务层的实现采用了构件的思想来构造服务对象。通过对高性能计算机上各种系统工具和应用的接口的抽象，把原来传统方式的命令行操作请求，封装成一个个 CORBA 服务对象供客户端使用，通过输入输出重定向获得命令的输出结果并返回给客户端。

3 DUET 的实现

为了帮助用户实现对曙光3000超级服务器的全部访问，DUET 包括下面几个组成部分：

1)UM：用户登录管理，管理 DUET 客户端提交的登录请求。当收到一个登录请求时，获取验证证书，检查用户名和口令，如果用户通过身份验证，就为其创建一个用户对象，返回对象引用给客户端，否则拒绝登录。

2)DCDB：并行调试器，使用户可以在图形界面里调试用 PVM 或者 MPI 写的并行应用程序，支持所有的串行调试器功能，并在串行调试器的基础上支持并行调试功能，包括组管理，调试的回放，以及并行程序的插桩等。

3)JOSS：作业管理工具，使用户可以查看远程并行机上的作业状态，提交一个作业，暂停一个作业，删除一个作业。允许用户自主分配作业的计算资源如结点数和优先级，并且在负载平衡的考虑下，能够动态地调度作业以提高系统的使用率。

4)RFB：远程文件浏览器，提供了远程并行机的良好视图，为了使我们的远程文件浏览器具有良好的可理解性与易学习性，我们尽量将文件浏览器的界面设计成 Windows 的界面样式，以树的形式显示目录结构，以表格的方式显示每个目录的内容，并具有下拉菜单，弹出菜单和工具条等大家熟悉的窗口元素。用户使用这些窗口元素和鼠标可以很方便地对远程并行机的文件系统进行组织和管理，可以完成 UNIX 文件系统上的绝大多数操作，如 cp，rm，mv，mkdir，find，grep，chmod，chown，chgrp 等。

5)RNOTEPAD：远程文本编辑器，使用户可以方便地对远程并行机上的文本文件进行编辑，它支持块定义，块拷贝，块删除，块粘帖，字符串查找，字符串替代等功能。

6)PFM：性能监测工具，允许用户监视远程并行机系统上各个节点的性能，包括 CPU，物理内存，虚拟内存的使用情

况。用户可以利用它来选择合适的节点来加载自己的作业。

7)PMT:进程管理工具,允许用户查看远程并行机系统上的所有进程的状态,并对进程进行操作,包括杀死一个进程,暂停一个进程,重新一个进程。也允许用户加载一个程序,并获取执行这个程序的输出结果。

DUET 的实现可以被分成下面三个阶段:

3.1 接口定义

在这个阶段,我们抽象出一个曙光3000的用户模型,它覆盖了用户使用机器的整个生命周期。接口定义了 DUET 应该提供什么样的服务,对于每个服务,我们用 IDL 语言定义了一个 CORBA 对象,可以被 DUET 客户端访问。例如,我们定义了如下的接口:

```
.....
interface DuetFS{ //文件系统常用接口
    string getHomeDirectory();
    string getParent(in string dir);
    boolean chmod(in string filename,in long filemode);
    long chown(in string filename,in string newowner);
    long chgrp(in string filename,in string newgroup);
    boolean delete(in string filename) raises (DuetFileException);
    boolean mkdir(in string filename) raises (DuetFileException);
    boolean rm(in string dirname) raises (DuetFileException);
    boolean copyFile(in string filename1,in string filename2) raises
      (DuetFileException);
    boolean moveFile(in string filename1,in string filename2) raises
      (DuetFileException);
    FileAttr getFileAttr(in string filename);
    boolean getDirStruct(in string dir,out DirStruct dirstruct) raises
      (DuetFileException);
    DuetRF openfile (in string filename, in MODE mode) raises
      (DuetFileException);
};
.....
```

3.2 服务器端的实现

对于在接口定义中定义的每个 CORBA 对象,我们给出了具体实现。我们用这些实例对象把曙光3000上的一些应用和系统功能封装起来,成为可供客户端调用的服务。我们实现了 MPI(Message Passing Interface)、PVM(Parallel Virtual Machine)和 JMS(Job Management System)以及进程管理、文件系统等。

这些实例对象之间的关系如图2所示。DUET 客户端首先发送一个登录请求,UM 对象检查是否为曙光3000合法用户,通过验证后 UM 对象会为其创建一个用户对象,之后客户端通过这个用户对象再获得对 DCDB,FS,JOSS,PMT 和 PFM 等服务对象的引用,以调用这些服务对象的方法。

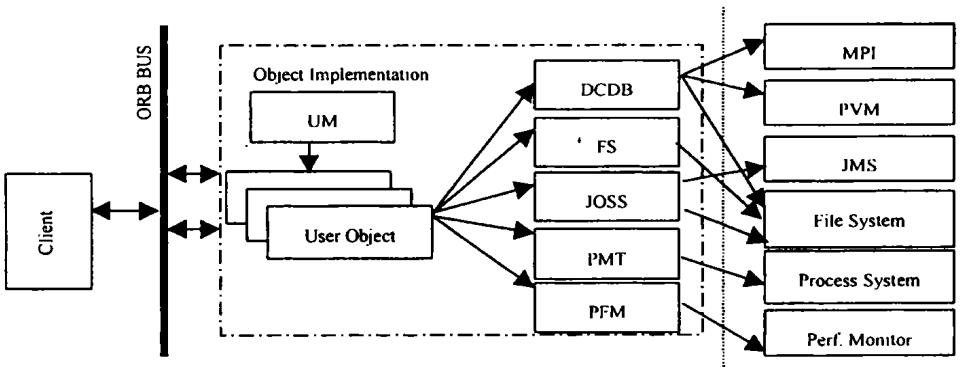

图2 服务器端 CORBA 对象的关系

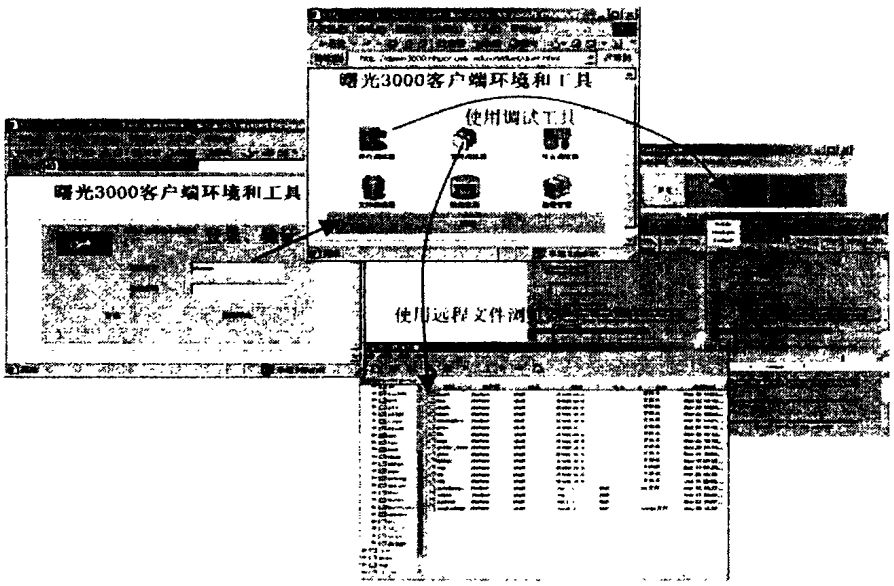

图3 DUET 客户端的界面

3.3 客户端实现

DUET 客户端为远程用户提供一个用户友好的本地 GUI 环境,屏蔽了高性能计算机的使用细节而代之以一目了然的图形界面,大大提高了曙光3000超级服务器的好用性。在这个 GUI 环境中,远程用户只要通过选择菜单和点击鼠标就可以方便地使用并行计算机的大部分功能。

DUET 客户端能够以 Applet 和 Application 两种方式运行，当以 Applet 方式运行时，远程用户的机器上只需要装有 Web 浏览器，就可下载至本地并在浏览器环境中使用；以 Application 方式运行时，远程用户必须下载 jar 打包文件并且将其设置类路径 CLASSPATH。

图3是几个 DUET 客户端的界面。

4 性能分析

因为在基于 CORBA 模型的 DUET 设计方案里，比较耗费时间和内存的操作，如产生图形界面和处理用户操作等，都被从服务器端实现中剥离出来在客户端完成，这就大大降低了服务器的工作负担。而且一个系统用户运行的不同系统工具在服务器端主机内存里将共用一个 Java 虚拟机，相应的服务进程只有一个，并不需要为每个连接新建立一个服务进程的拷贝，从而在根本上解决了以前那种 Java 语言编写的系统工具集在使用时占用过多内存空间的问题，尽可能地为用户计算保留了宝贵的内存资源，提高了系统的利用效率和对用户计算的支持能力。

表1 读取目录内容的网络传输性能的测试数据

目录	文件个数	传输字节数	传输时间(ms)	排序时间(ms)	响应时间(ms)
/	47	2242	50	0	50
/data2	5	302	0	0	0
/data2/home	80	4440	90	30	160
/data2/home/zhchen	59	2834	80	20	110
/bin	421	22384	600	200	930

同时，客户端和服务器端间的网络数据流量被降低到最小，使得多个用户在同时使用系统工具时不至于造成系统网络负担过重。例如在 RFB 远程文件浏览器中，文件的拷贝、删除、移动等操作完全在服务器上执行，服务器只返回执行该操作是成功还是失败的布尔值。从而在执行这类操作时，客户和服务器之间的通信量非常小，基本上可以忽略不计。而在读取一个目录或文件的内容时，客户和服务器之间的传输量是跟该文件或目录的大小成比例。表1是读取目录内容的网络传输性能的测试数据。

结论和未来的工作 相对于传统的高性能计算机用户环境，DUET 为用户提供了一个从本地局域网和 Internet 使用曙光3000的 GUI 接口，将显著提高曙光3000并行机的好用性。

未来我们将继续完善 DUET，使其能够包括高性能计算机几乎所有用户接口的功能。比如可以增加一个能支持编写、编译、运行和调试 pvm 和 mpi 程序的 GUI 工具，使用户能方便地在图形界面下开发并行程序；增加一个终端工具，允许用户执行在 telnet 中可以被执行的命令，这样用户可以在 DUET 中运行命令行交互的程序；在 RFB 远程文件浏览器中增加支持 put、get 文件功能，使曙光3000服务器和客户端可以传输文件，取代 ftp 等。

参 考 文 献

1 Buhlmann B, Bieri H. Supercomputing at the Desktop: An Improved Interface Using Internet Facilities. In: LNCS 1401, P. Sloot, et al. eds, Proc. of High-Performance Computing and Networking Intl. Conf. and Exhibition (HPCN Europe 1998), Springer, April 1998. 526～534

2 Sun Microsystems. Java Language - A White Paper. Sun Microsystems Computer Company, 1996

3 Orfali R, Harkey D. Client/Server Programming with Java and CORBA, Second Edition. Wiley Computer publishing, 1999

4 Bradley M. IIOP: OMG's Internet Inter-ORB Protocol: A Brief Description. OMG, May 1997

5 Westphall C M, et al. JaCoWeb Security - A CORBA Security Discretionary Prototype. CLEI Electronic Journal, 2000, 3(2)

6 Borland Corporation. VisiBroker SSL Pack 4–5 Programmer's Guide. Borland Software Corporation, June 1999

(上接第35页)

2 Deng J S Z. Scheduling Real-Time Applications in a Open System Environment. In: Proc. of the 18th IEEE Real-Time Systems Symposium (RTSS97), June 1997

3 Kalogeraki V, Melliar-Smith P, Moser L. Soft Real-Time Resource Management in CORBA Distributed Systems. In: Proc. of the Workshop on Middleware for Real-Time Systems and Services, (San Francisco, CA), IEEE, Dec. 1997

4 Smith B C. Reflection and Semantics in a Procedural Language: [Technical Report 272]. MIT Laboratory of Computer Science, 1982

5 Oliva A, Garcia I C, Buzato L E. The reflexive architecture of Guaraná: [Technical Report IC-98-14]. Institute of Computing, State University of Campinas, April 1998

6 Oliva A, Buzato L E. An Overview of MOLDS : A Meta-Object Library for Distributed Systems: [Technical Report IC-98-15]. Institute of Computing, State University of Campinas, April 1998

7 Costa F M, Blair G S. A Reflective Architecture for Middleware: Design and Implementation. In: ECOOP'99 PhDOOS Workshop, June 1999

8 Loques O, Leite J, Lobosco M, et al. Integrating Meta-Level Programming and Configuration Programming. In: Walter Cazzola, Robert J. Stroud, and Francesco Tisato, eds. Proceedings of the 1st Workshop on Object-Oriented Reflection and Software Engineer-ing(OORaSE'99), University of Milano Bicocca, Nov. 1999. 137～151

9 Kon F, Campbell R, Roman M. Design and Implementation of Runtime Reflection in Communication Middleware: the Dynamic TAO Case. In: Proc. of ICDCS'99 Workshop on Middleware, 1999

10 RM'2000. Workshop on Reflective Middleware of Middleware'2000. Available at: www. comp. lancs. au. uk/computing/RM2000. April 2000

11 Dowling J, Schafer T, Cahill V, et al. Using Reflection to Support Dynamic Adaptation of System Software: A Case Study Driven Evaluation. In Walter Cazzola, Robert J. Stroud, and Francesco Tisato, eds. Reflection and Software Engineering, Lecture Notes in Computer Science 1826, Springer-Verlag, Heidel-berg, Germany, June 2000. 171～190

12 Wang N, Schmidt D C, Kircher M, Parameswaran K. Towards a Reflective Middleware Framework for QoS-enabled CORBA Component Model Applications. IEEE Distributed Systems Online, 2001

13 Gill C D, Levine D L, Schmidt D C. The Design and Performance of a Real-Time CORBA Scheduling Service. Real-Time Systems, The International Journal of Time-Critical Computing Systems, special issue on Real-Time Middleware, 2001, 20