一个集成了COM和CORBA的脚本语言*

付岩，白硕，李国杰

（中国科学院 计算技术研究所，北京 100080）

E-mail：fuyan@ict.ac.cn；bai@ncic.ac.cn

http://www.ncic.ac.cn

摘要：提出了一个集成了COM和CORBA两种分布式对象系统的简单的脚本语言——GSCRIPT.该语言使用自动化编程接口和动态调用接口来分别操纵在网络中的COM对象和CORBA对象.GSCRIPT是平台独立的，同一GSCRIPT程序可以在多种操作系统和硬件平台上运行.同时，也详细介绍了GSCRIPT解释器的体系结构和提供事件服务的方法，它们也是GSCRIPT实现平台独立性的途径.

关键词：面向对象；分布式对象；脚本语言；COM；CORBA

中图法分类号：TP311 **文献标识码**：A

面向对象设计的概念在20世纪70年代出现以后，各种面向对象的语言和分析、设计方法[1]相继面世并成熟起来.面向对象技术被认为是继结构化设计技术之后，在软件工程领域中一次阶段性的飞跃.然而，在面向对象语言（如C++）中的传统对象是依赖于程序的本身而存在的.另外，一个对象的方法和属性也不能被除该对象的宿主程序之外的其他程序所使用.这就使得在面向对象技术中最具吸引力的特点之一的可重用性[2]受到了极大的限制.分布式对象则很好地解决了这些问题.一般来说，分布式对象是具有清晰定义的接口的自完备（self-contained）的二进制软件模块，它可以独立于对象的使用者而存在于网络中的任何地方.同时，对象本身及其使用者可以不受编程语言、操作系统和硬件平台等环境的限制.分布式对象的这些特点使其逐渐成为客户/服务器计算模式[1,2]和构件化软件设计[3]的主要选择.目前，分布式对象的实现有两个标准，component object model（COM）[4]和common object request broker architecture（CORBA）[5].前者由美国微软公司提出，主要在Windows平台上实现.后者由国际标准化组织Object Management Group（OMG）提出，目前已经在多种软、硬件平台上实现.尽管在CORBA规范[5]中给出了一个两者之间的互操作模型和类型映射规则，但仍然没有一种简单的方法可以同时使用网络上的在COM对象和CORBA对象中所蕴藏的计算资源.而且两者在概念和使用中的复杂性也限制了它们的实际应用.一般认为，熟练地掌握COM技术至少需要6个月的时间[4].CORBA在应用上也有类似的困难.针对这些问题，本文提出了一种脚本语言GSCRIPT.它可以方便地同时使用在网络中基于COM和CORBA的分布式对象.

1 GSCRIPT的设计目标

GSCRIPT是一种脚本语言.相对于C++和Java这样的系统编程语言来说，脚本语言是一类

* 收稿日期：1999-10-15；修改日期：2000-03-06

基金项目：国家863高科技发展计划资助项目（863-306-ZD02-01-3）

作者简介：付岩（1974－），男，内蒙古人，博士，主要研究领域为面向对象系统，分布式计算；白硕（1956－），男，辽宁人，博士，研究员，博士生导师，主要研究领域为Internet/Intranet应用软件，计算语言学，人工智能；李国杰（1945－），男，湖南人，研究员，博士生导师，中国工程院院士，主要研究领域为并行计算，人工智能.

具有更高抽象层次的简单易用的编程语言[6].脚本语言一般假定已经存在一些有用的软件构件,它的任务是把这些构件有效地组合在一起,而不是像传统的编程方式那样一切都从头做起.脚本语言很少用于实现复杂的算法和数据结构.这些工作通常是由系统语言在构件中实现的.由于存在这些特点,使用脚本语言可以在一定的程度上提高软件的生产效率.所以,我们选择用一种脚本语言来实现GSCRIPT.

由于脚本语言一般是解释执行的,没有一个能够使得GSCRIPT预先获得对象类型信息的编译的过程,所以GSCRIPT在连接COM对象时主要使用自动化编程接口(automation)[7];在连接CORBA对象时使用动态调用接口(dynamic invocation interface,简称DII)[5].事实上,这也是在CORBA规范中COM-CORBA互操作模型的一种主要实现方式[5].该模型试图实现这样一个目标:在一种对象模型中的对象使用者可以把另外一种对象模型中的对象当作本对象模型中的对象一样来使用.这种方法在灵活性和效率上都存在着不足.

另外,GSCRIPT还要实现这样两个目标:一个是实现平台独立性.同一个GSCRIPT脚本程序可以运行在多种软、硬件平台上,不受操作系统和硬件等环境的限制,包括在某些不支持COM和CORBA的平台上;另一个是提供事件服务支持.当发生某些预先定义的事件时,COM对象或CORBA对象可以回调一段与这些事件相应的GSCRIPT程序.

2 GSCRIPT语言

考虑到一种新语言在应用中的困难,GSCRIPT没有重新定义一套自己的词法和语法规则,而是扩展了微软的脚本语言visual basic scripting edition(VBS)[8].这样,GSCRIPT语言的解释器不但能执行已有的VBS程序,而且能执行同时使用COM对象和CORBA对象的GSCRIPT脚本程序.关于VBS语言的详细内容可以在文献[8]中找到,这里只介绍GSCRIPT的扩展部分.

与其他脚本语言一样[6],GSCRIPT是一种无类型的语言.它只支持一种数据类型Variant.一个Variant类型的变量在不同的上下文中可以代表不同的子类型,如整数类型、字符串类型等.在VBS5.0中,Variant可以表示13种子类型[8].其中的Object子类型主要用于表示自动化对象(IDispatch接口指针).在GSCRIPT语言中,Object子类型同时也可以用于表示CORBA对象.在运行过程中,它既可以是一个虚IDispatch接口指针,也可以是一个虚CORBA对象引用(见第3节).当把一个代表CORBA对象的变量作为参数传递给用于动态地判断变量类型的函数VarType时,该函数返回一个在GSCRIPT中定义的常量vbCorbaObject.

GSCRIPT把在VBS中用于创建对象引用的函数CreateObject扩展为如下形式:

```
stmt createobject: CreateObject (objectname [,location]_,objecttype_)
objectname: com_objectname|corba_objectname
com_objectname: servername.typename
servername: STRING typename: STRING
corba_objectname:[namingmethod]name
namingmethod: NamingService|IOR|Visibroker
name: STRING
location: STRING
objecttype: OT_COM OT_CORBA
```

其中objectname是一个字符串,表示所要引用的对象.COM对象分成两个部分来描述servername

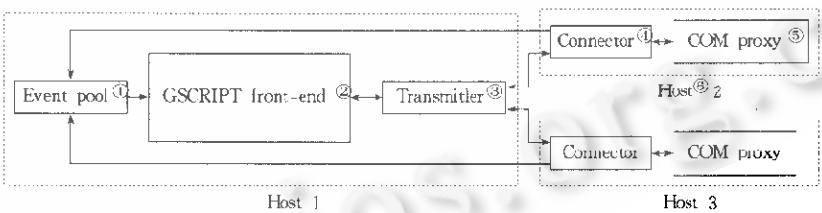
和 typename. 前者是提供该对象的服务器名(应用程序名),后者是该对象的类型. 与之相关的信息一般保存在 COM 代理所在主机上的注册薄中(见第 3 节). 有很多种方法可以从一个给定的字符串得到它所对应的 CORBA 对象引用. 最常用的方法是 CORBA 命名服务[9]. 另外,也可以使用 IOR 字符串[5],或者使用由 CORBA 产品供应商所提供的扩展方法. GSCRIPT 程序可以在 objectname 字符串的开头附加相应的对象绑定方法信息. 否则,GSCRIPT 将按照某种固定的顺序试图创建一个 CORBA 对象引用,直到第 1 个成功的对象绑定或者所有的方法都失败而返回为止.

Location 是一个可选项,用于表示创建由 objectname 所代表的对象的主机在网络中的位置. 它一般是一个 DNS 名或 IP 地址. 如果不提供该选项,GSCRIPT 就使用在 COM 代理主机或 CORBA 代理主机中的对象注册信息来创建对 objectname 对象的引用.

Objecttype 也是一个可选项,用于表示由 objectname 所代表的对象所属的分布式对象标准. OT_COM 表示该对象是一个 COM 对象(自动化对象),OT_CORBA 表示该对象是一个 CORBA 对象. 如果不提供该选项,GSCRIPT 将按照某种固定的顺序试图创建对 objectname 对象的引用,直到第 1 个成功的对象绑定或者所有的分布式对象类型都失败而返回为止. 如果成功地创建了一个绑定,还要在 GSCRIPT 解释器中符号表里的相应项上记录所返回的虚对象所属的分布式对象标准.

3 GSCRIPT 体系结构

GSCRIPT 是一种平台独立的脚本语言,同一 GSCRIPT 程序可以在多种操作系统和硬件平台环境下运行. 只有这样才能最大程度地利用在网络上的各种蕴藏在 COM 对象和 CORBA 对象中的计算资源. GSCRIPT 体系结构就是为了实现平台独立性而设计的. 图 1 是该体系结构的一个示意图.



①事件池,②GSCRIPT前端,③转发器,④连接器,⑤代理,⑥主机.

Fig. 1 Architecture of GSCRIPT

图1 GSCRIPT体系结构

GSCRIPT 前端包括词法分析器、语法分析器、中间代码生成器和执行器. 在整个解释器中,所有平台独立的任务都在这里完成,如语法分析、算术运算和流程控制等. GSCRIPT 前端并不真正操作 COM 对象或 CORBA 对象. 类型 Variant 的子类型 Object 记录的是虚对象引用. 在 GSCRIPT 前端的符号表中有两个区域专门用于所使用的 COM 对象和 CORBA 对象. 主要的表项有变量名、对象编号和事件注册信息等. 其中的对象编号在同一个 GSCRIPT 程序的整个运行周期里是惟一的,并与 COM 代理或 CORBA 代理中的相应表项保持一致. 虚对象引用实际上就是符号表中用于表示一个对象的表项的入口点,通常是一个索引值或指针. 执行器并不真正地执行在 COM 对象或 CORBA 对象上的方法调用,而是通过与 GSCRIPT 前端紧耦合的转发器把相应的调用请求传递给 COM 代理或 CORBA 代理. 这些对象代理真正执行相应的方法调用.

COM 代理的主要任务是创建 COM 对象(自动化对象)、调用对象的方法(属性)、向 GSCRIPT

前端返回执行结果,以及向 GSCRIPT 前端发送事件服务请求等.COM 代理一般在 Windows 平台上实现.在创建 COM 对象时要使用在该平台上的注册薄中所记录的对象信息.在 COM 代理中维护了一份与 GSCRIPT 前端的符号表中 COM 对象区域相一致的对象登记表.该表记录了真正的对象引用,即 IDispatch 接口指针.当与 COM 代理紧耦合的连接器接收到 GSCRIPT 前端通过转发器而传送来的命令后,COM 代理就根据对象登记表中的信息执行相应的操作.对象方法调用是主要的命令类型.该命令的格式如下:

虚对象引用	方法名	参数 1 类型	参数 1 值	参数 2 类型	参数 2 值	...

COM 代理在接收到该命令之后,从对象登记表中找到相应的 IDispatch 接口指针,处理命令参数,最后通过调用 IDispatch 接口上的 Invoke 方法向 COM 对象发送调用请求.如果调用成功返回,还要通过连接器把返回的结果传送给 GSCRIPT 前端.

CORBA 代理的结构与 COM 代理类似.当它在一个 CORBA 运行环境的支持下使用与自动化编程接口类似的对象方法调用命令之后,从对象登记表中找到相应的 CORBA 对象加以引用.然后根据接口池(IR)[5]中的信息将方法的参数值和返回值作必要的类型转换,并通过 DII 接口上的 invoke 方法向 CORBA 对象发送方法调用请求.

在转发器与连接器之间所交换的数据使用一种独立于硬件体系结构的表示方式,例如外部数据表示方法(XDR).另外,在转发器与连接器之间维护一个简单的接口的同时,还要支持包括进程内过程调用、各种类型的网络通信直到串行口连接在内的各种通信方式.这样的设计使得 GSCRIPT 前端、COM 代理和 CORBA 代理三者不必在同一台主机上实现,而可以根据需要灵活地进行配置.GSCRIPT 前端一般用标准 C/C++或 Java 语言编写,另外,由于 GSCRIPT 语言本身的简单性,从而实现了 GSCRIPT 平台独立的特点.

4 事件服务

GSCRIPT 的事件服务是由 GSCRIPT 前端、对象代理和分布式对象协作完成的.GSCRIPT 前端是事件响应代码的载体和执行者,对象代理接受来自分布式对象的事件服务请求,并提供事件和 GSCRIPT 前端中事件响应代码之间的映射关系,分布式对象则是事件源.COM 和 CORBA 对事件服务支持的原理是类似的.前者使用连接点协议(connection-point protocol)[4],后者使用 CORBA 事件服务协议(event service)[9].GSCRIPT 使用这两种协议分别在对象代理中动态地创建事件响应对象.该对象分别对应于 COM 连接点协议中的事件槽(event sink)和 CORBA 事件服务中的事件消费者(consumer).事件响应对象建立了事件类型与 GSCRIPT 前端中事件响应代码之间的映射关系.在创建分布式对象时,对象代理把事件响应对象的引用传递给该对象.这样,在发生某种预先定义的事件时,分布式对象就可以通过事件响应对象而使 GSCRIPT 前端开始执行一段与该事件相对应的代码.这段 GSCRIPT 代码可以自由地使用存在于网络中的 COM 对象或 CORBA 对象.

5 一个 GSCRIPT 示例

我们在 Windows NT4.0 的平台上实现了一个 GSCRIPT 的实验版本(无事件服务支持).通过简单地修改词法分析器,我们还把该 GSCRIPT 语言解释器集成到了微软的 Web 服务器 IIS4.0 中.这样,Web 站点的设计者可以在网页中嵌入一些同时使用 COM 和 CORBA 两种分布式对象的 GSCRIPT 代码.这些网页在接受请求时可以被 GSCRIPT 解释器处理而生成动态的 HTML 页面

返回给浏览器.下面是一个嵌入了GSCRIPT代码的页面:

```
<html><head><title>This is the original web page</title></head><body>
<%dim a as object%><%set a=CreateObject("COMServer.MyServer")%>
<%a.SetEntry 101, "Zhang San"%><%a.SetEntry 102, "Li Si"%>
<%print a.GetName(101), ""%><% print a.GetID("Zhang San")%><p>
<%dim b as object%><%set b=CreateObject("IDL:ClassServer:1.0")%>
<%b.SetClass 101,"Advanced Algorithms"%><%b.SetClass 101, "Computer Architecture"%>
<%print b.GetClass(a.GetID("Zhang San"),1), ""%><%print b.GetClass(a.GetID("Zhang San"),2)%>
</body></html>
```

在"<%"和"%>"之间嵌入的代码是GSCRIPT程序.当GSCRIPT解释器处理这个页面时,只执行被嵌入的GSCRIPT程序,而把页面中其他部分原样地输出到所生成的动态HTML页面中.

在上面的Web页中,变量a是一个COM对象,它维护了一个学号和姓名之间的关系.变量b是一个CORBA对象,它维护了一个学号和与该学号对应的学生所选的课程之间的关系.这两个对象可以分别操纵在不同主机上的数据库.由GSCRIPT解释器所生成的动态页面如下(删去了一些换行符):

```
<html><head><title>This is the original web page</title></head><body>
Zhang San
101 <p>
Advanced Algorithms
Computer Architecture
</body></html>
```

这个动态页面将被返回给申请原Web页的浏览器.该示例说明,GSCRIPT可以把COM和CORBA两种不同的分布式对象标准和谐地集成到一个脚本语言中来.

6 结束语

GSCRIPT之所以能够在一种脚本语言中集成COM和CORBA两种分布式对象标准,是因为尽管它们无论在设计上还是在实现上都有很大的区别,但是它们都把对象看成是具有一定功能的独立的软件实体,每个对象都有清晰定义的接口,并向对象使用者屏蔽了接口的实现细节.脚本语言本身的简单性使得GSCRIPT可以通过一种简单的体系结构设计而实现平台独立的要求.另外,由于COM和CORBA在事件服务的概念上所存在的类似性,GSCRIPT可以在脚本语言中提供一种一致的事件响应机制.在未来的研究工作中,我们还会致力于将Java对象、遗留系统(legacy systems)和嵌入式系统集成到GSCRIPT中来.

References:

[1] Fowler, M. UML Distilled. Reading, MA: Addison-Wesley Publishing Company, 1998.

[2] Lewandowski, S. M. Frameworks for component-based client/server computing. ACM Computing Surveys, 1998,30(1):3~27.

[3] Orfali, R., Harkey, D., Edwards, J. Essential Distributed Objects Survival Guide. New York: John Wiley & Sons, Inc., 1996.

[4] Box, D. Essential COM. Reading, MA: Addison-Wesley Publishing Company, 1998.

[5] Object Mangement Group. CORBA/IIOP 2.0 Specification. http://www.omg.org, 1996.

[6] Ousterhout, J. K. Scripting: higher-level programming for the 21st Century. IEEE Computer, 1998,31(3):23~29.

[7] Microsoft Corperation. OLE Automation Programmer's Reference. Washington, DC: Microsoft Press, 1996.

[8] Visual Basic Scripting Edition Papers. http://www.microsoft.com/scripting.

[9] Object Management Group. CORBA Services: Common Object Services Specification. 1996. http://www.omg.org.

A Script Language Integrating COM and CORBA*

FU Yan, BAI Shuo, LI Guo-jie

(*Institute of Computing Technology, The Chinese Academy of Sciences, Beijing* 100080, *China*)

E-mail: fuyan@ict.ac.cn; bai@ncic.ac.cn

http://www.ncic.ac.cn

Abstract: A simple script language, GSCRIPT, which integrates distributed object systems of COM and CORBA is presented in this paper. The language manages COM objects and CORBA objects on network by using automation programming interface and dynamic invocation interface respectively. GSCRIPT is platform-independent, one GSCRIPT program runs on many operating systems and hardware platforms. The architecture of GSCRIPT interpreter and the mechanism to support event service are introduced in detail, which enables the platform-independence of GSCRIPT.

Key words: object orientation; distributed object; script language; COM; CORBA

* Received October 15, 1999; accepted March 6, 2000

Supported by the National High Technology Development Program of China under Gr