基于 JNDI 的分布式对象技术
Distributed Object Technology Based on JNDI

吕立坚,赵 明,李静滨,谭 政
Lü Lijian,Zhao Ming,Li Jingbin,Tan Zheng
(广西大学计算机与电子信息学院,广西南宁 530004)
(Coll. of Comp. & Elet. Info. ,Guangxi University,Nanning,Guangxi,530004,China)

摘要:介绍分布式应用系统和 JNDI 分布式对象技术的基本原理,给出 Microsoft Windows 2000 Server+j2sdk1.4.1_01+Sun ONE Studio 4 Update1 Enterprise Edition 环境下,基于 JNDI 的文件系统的分布式对象访问技术的应用实例。实例显示,JNDI 具有将对象绑定到名字,并通过 1 个名字查找对象引用的能力。

关键词:分布式对象 JNDI 命名服务
中图法分类号:TP393

Abstract: The distributed application system is discussed. The basic principle of JNDI distributed Object Technology is introduced. An application instance of distributed Object Access Technology based on the file system of JNDI is givenunder the environment of Microsoft Windows 2000 Server and j2sdk1.4.1_01 and sun one studio 4 update enterprise edition. In the instance, JNDI reveals its ability in binding the object to name and through one name definitely finding out the object quoting.

Key words: distributed object, JNDI, naming service

随着互联网技术的飞速发展,人们期望能更充分利用网络资源以及企业应用软件、信息数据库、组件、计算机设备、打印设备、网络设备等现有资源。但分布式系统在异构环境下存在互操作、系统管理、系统安全,以及大访问量下的系统可靠性等问题。随着分布式网络体系架构日趋成熟,分布对象技术使面向对象技术能够在异构的网络计算环境中得到全面实施,从而能够有效地控制系统的开发、管理和维护。本文主要讨论 JNDI(Java Naming and Directory Interface)分布对象访问技术,介绍 JNDI 技术的基本原理,并给出利用 JNDI 技术访问分布式对象的应用实例。

1 分布式应用系统

分布式对象技术指在分布式异构环境下建立应用系统平台和对象构件,在应用系统平台的支撑下,开发者可以将软件功能包装成更易管理和使用的对象。这种技术将分布在网络上的资源看作对象进行组织,每一个对象都有定义明晰的访问接口。创建和维护分布对象实体的应用称为服务器,按照接口访问该对象的应用称为客户。服务器中的对象不仅能够被访问,而且其自身也可作为其他对象的客户。在分布式环境下,分布式对象常被称为组件,它既可以是一个简单的对象,也可以是一组相关的对象复合体。这些对象可以跨越不同的软硬件平台进行互操作,而对于外界来说这种互操作透明。用户只需按照相应的协议来访问对象接口即可取得对象提供的相关软件功能。典型的分布式应用系统如图 1 所示。

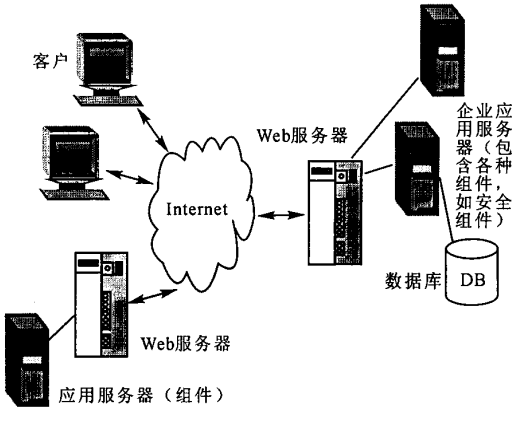

图 1 分布式应用系统

2 JNDI 分布式对象技术

命名服务是将名字和计算机中的一个对象相关联,通过名字可以方便地找到对应的对象(图 2)。在

2004-06-08 收稿。

分布式系统和非分布式系统中，应用程序引用对象的主要机制是命名服务。Sun公司的JNDI为多种多样的命名与目录服务提供一个通用的接口。与其他的分布式对象访问技术相比，JNDI具有以下优点：(1)JNDI为用户提供了1个资源定位的接口，用户可以方便地访问Java应用程序，可获取到Java事务API的User Transaction接口的一个引用，连接到资源工厂(如JDBC、URL和消息服务)，查找组件从而获取相应的服务等。(2)JNDI可以与RMI、EJB、CORBA相联系并综合RMI和CORBA的优点，使得程序员能更方便地编写分布式程序设计，实现分布式计算。(3)JNDI的对象——存储能力允许它成为分布式网络的资源管理者[1]，它具有简单、可管理对象等特点。

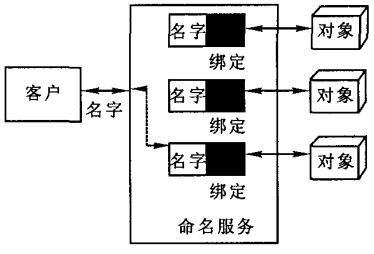

图2 命名服务

JNDI体系结构[1]包括应用编程接口API和服务提供者接口SPI。API由javax.naming和javax.naming.directory 2个包组成，提供应用程序访问名字和目录服务；SPI是名字和目录服务透明接入的挂钩，仅包括javax.naming.spi。利用JNDI实现分布式应用的Java应用程序至少需要4部分软件——Java虚拟机、JNDI软件包、命名或目录服务系统以及与其相应的服务提供者软件包，其中服务提供者软件包是SPI对特定的命名或目录服务系统的具体实现。这是JNDI与特定服务进行交互的纽带。不同的服务对应不同的软件包。如果用户需要的特定服务在Sun公司上没能找到对应的SPI包，则用户可以尝试根据JNDI规定的SPI规范编写自己的SPI实现。

这种设计模式隔离用户API和底层服务提供者之间的关系，通过可插拔的配置方式，JNDI客户可以使用任何底层的服务提供者提供的名字和目录服务实现，而不需改变客户程序的代码。通过这种方式，各种名字服务实现可以通过JNDI SPI的包装很轻易地集成到JNDI体系中，而客户能够通过1个统一的标准的访问接口进行访问。JNDI体系结构如图3所示。

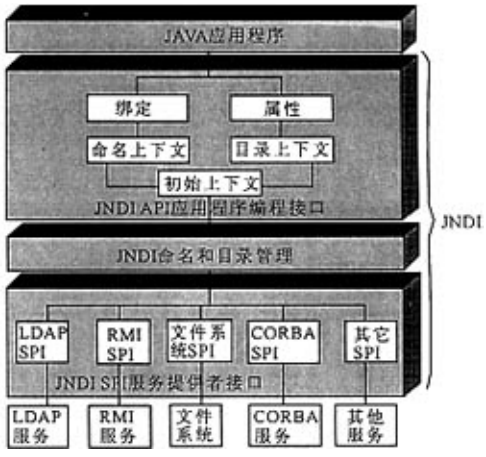

图3 JNDI体系结构

3 应用实例

在Microsoft Windows 2000 Server + j2sdk1.4.1 _ 01 + Sun ONE Studio 4 Update1 Enterprise Edition环境下，基于JNDI的文件系统的分布式对象访问技术的应用实例[2~6]显示JNDI通过将对象绑定到名字并通过1个名字查找对象引用的能力。

3.1 JNDI环境的配置

(1)安装操作系统Microsoft Windows 2000 Server。

(2)安装j2sdk1.4.1 _ 01，将其安装于C:\j2sdk1.4.1 _ 01的路径下，安装完毕后，在Windows 2000 Server的环境变量中将path变量的值设为%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;c:\j2sdk1.4.1 _ 01\bin。

新建变量CLASSPATH，其值设为:.;c:\j2sdk1.4.1 _ 01\lib\dt.jar;c:\j2sdk1.4.1 _ 01\lib\tools.jar。

(3)安装Sun ONE Studio 4 Update1 Enterprise Edition。

(4)从http://java.sun.com/products/jndi下载进行JNDI编程所需的JNDI SPI：providerutil.jar，fscontext.jar，ldap.jar，rmiregistry.jar把这个SPI包放到c:\j2sdk1.4.1 _ 01\jre\lib\ext目录，该目录是Java运行时环境在开发过程中用来对运行库进行扩展的目录，其他要扩展的Java包也可以放入目录。在Sun ONE Studio 4 Update1 Enterprise Edition的启动程序启动后，程序就会自动加载这些添加的包，这样相应的JNDI服务提供者(文件服务、LDAP服务、RMI注册系

统)被安装到 Sun ONE Studio 4。如果需在 DOS 下编程,则需要把这些包的路径也添加到 CLASSPATH 环境变量中。

3.2 实现过程

(1)初始化上下文。JNDI 对命名或目录服务的操作对应特定的上下文对象,必须先获得 1 个初始化上下文的引用。而要获得该引用,必须设置对应服务提供者的环境参数,诸如 SPI 名、服务提供路径等,如果该服务需要授权,则还要指定用户名、密码等。

(2)利用初始化的上下文对当前文件系统进行操作。

(3)应用程序打包。由于 JNDI 的 SPI 包并不包括 Java 2 核心包,如果要把编写的 JNDI 应用程序移植到其他的 Java 虚拟机上运行,应该把用到的非核心类包括其全路径一齐打包进应用程序中形成 jar 文档。

3.3 应用程序的封装

应用程序提供了 1 个图形界面,其执行后将初始化一个针对文件系统的上下文,它允许用户在文本框中输入 1 个字符串形式的名字,程序将调用上下文的 lookup()方法,对所输入的名字进行查找,如果能够找到该名字绑定的对象,则显示该对象的信息,如果出错或者找不到,则显示查找过程中出现的异常。

(1)编写应用程序并编译成 class 文件。这里的程序包含在 jndi 包内的程序 Lookup. java,编译这个文件生成 Lookup. class 文件。

(2)组织应用的类文件。把 Lookup. class 文件放到 jndi 文件夹;Lookup 程序调用文件系统的 SPI 包,其定义的类包含在 fscontext. jar 和 providerutil. jar 2 个文档,将 2 个文档解压得到的类文件,它们包含在 com 的文件夹内。把 jndi 文件夹和 com 文件夹放到同一个目录下,如 tan 目录。

(3)编写一个清单文件 t. mf。在 t. mf 文件中包含这样一行文字:Main-Class:jndi. Lookup。

(4)将应用程序打包。在命令提示符下,进入 tan 目录,输入:jar cvfm lookup. jar t. mf *。

(5)执行应用程序。在命令提示符下,输入:java-jar lookup. jar。

运行的结果如图 4 所示,在命令提示符下,输入 autoexec. bat 点击 ok 按纽,运行的结果如图 5 所示。输入 jndi 点击 ok 按纽,运行的结果如图 6 所示。

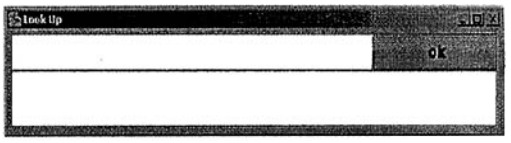

图 4 程序执行的初始界面

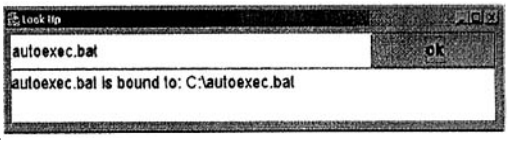

图 5 查找名字是一个文件名时的显示结果

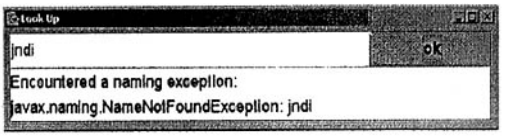

图 6 查找名字不存在时的显示结果

该实例显示 JNDI 的对象绑定和查找能力,为应用程序引用分布式对象(组件)提供了强大的技术支持。

4 结束语

JNDI 分布式对象技术是一个支持访问 Java 程序中命名和目录服务的 API,它提供标准的与协议无关的 API,为访问不同的命名和目录服务提供一个通用的框架,实现对用户访问的完全透明性,并能够将这些服务有效的组织在一起,融为一个整体,为实现对分布式对象的方便、高效管理提供了可能。相信在不远的将来 JNDI 将会在基于网络的分布式访问技术中占有一席之地。

参考文献:

1 Paul J Perrone, et al. J2EE 构建企业系统——专家级解决方案. 北京:清华大学出版社,2001.

2 Joseph L Weber. Java2 编程详解. 北京:电子工业出版社,1999.

3 Cay S. Horstmann Gary Cornell. Java 2 核心技术卷Ⅱ:高级特性. 北京:机械工业出版社,2000.

4 David Flanagan, JimFarley, William Crawford. Java Enterprise 技术手册. 北京:中国电力出版社,2002.

5 How to Java——JNDI overview, http://www.javaworld.com/javaworld/jw-03-2000/howto/jw-0331-howto.zip, 2003.

6 Sun Microsystems Inc. Java™Naming and Directory Interface™ Application Programming Interface (JNDI API), 2003.

(责任编辑:黎贞崇)