基于 Java 技术的 Web 环境下分布式数据库互操作性的实现

李炽明 莫倩 徐明 (长沙 国防科技大学并行与分布处理实验室 410073)

摘要:本文首先简要介绍了 Java 技术中 JDBC 和 RMI 机制的工作原理,讨论了如何应用 JDBC 实现在 Web 环境下对异种数据库进行透明访问,如何采用 RMI 机制消除 Java 对 Applet 程序的安全性限制,实现 Web 环境下分布式数据库的互操作,最后给出了一个我们基于 Java 技术设计和实现的 Web 环境下 N 层 Client/Server 结构分布式数据库应用系统的模型。

关键词:RMI JDBC WWW 分布式数据库互操作性

1. 背景

计算机技术和网络技术的发展使以网络为中心的计算日益得到重视,WWW 系统和数据库是网络化信息服务的基础。Web 能用一种及时和友好的方式向人们提供大量有用的信息,而伴随着大量信息,就需要大量的数据库的管理,Web 和数据库的结合是 WWW 信息服务技术和分布式数据库技术发展的大势所趋。把数据库同 WWW 服务器连接起来,这种一体化的信息网络系统:数据库 + WWW 服务器,将成为下一代的 Internet 开发的新领域。

Java 是近几年随着网络的发展而流行的,它能够在短短几年间迅速成为计算机界的热点,这跟它的种种良好的特点是分不开的。随着 Java 标准的确立,Java 技术更是不断的成熟。在数据库处理方面,Java 提供了 JDBC API,为数据库开发者开发数据库应用提供了标准的应用程序编程接口。还有新兴的 RMI 技术,它提出了远程对象引用的概念,这跟当年过程式编程语言年代提出面向对象语言一样具有革命性的意义。RMI 技术更适合 Java 这种纯面向对象语言,它使得分布式环境下不同 Java 程序之间的通信更加简单自然,符合面向对象的语义。Java 已经成为开发网络数据库应用的理想工具之一。

Java Applet 是 Java 与 Web 结合的产物,它给 Web 带来了动态效果和极大的扩展,因为它本身就是一种编程语言,功能上比静态 HTML, CGI, JavaScript 等都强大得多。于是,Java Applet 理所当然的成为 Web 数据库应用和理想途径。当然,网络上对 Applet 的限制,是我们所无法避免的,这也就是我们采用 RMI 技术的原因。

本文给出了一个利用 JDBC 和 RMI 技术实现 Web 环境下分布式数据库应用的方法。下面先让我们来介绍一下 Java 的 JDBC 和 RMI 技术。

2. JDBC 工作原理简介

JDBC(Java Database Connectivity)是一个 Sun 公司注册了的商标,代表用来执行 SQL 语句的 Java 语言应用程序编程接口 API,它包括一系列用 Java 语言编写的类和接口,为数据库开发者开发数据库应用提供了标准的应用程序编程接口。

(1)JDBC 概念。采用 JDBC 可以很容易用 SQL 语句访问任何商用数据库,也就是说,通过 JDBC API,数据库应用开发者就不需要编写一个程序访问 Sybase 数据库,编写另一个程序访问 Oracle 数据库,再编写第三个程序访问 Informix 数据库,他只需编写一个使用 JDBC API 的程序,就可以将 SQL 查询语句送往合适的数据库。同时,采用 Java 语言编写应用程序,具有平台无关性,不需要为不同的平台编写不同的应用程序。因此,采用 Java 语言和 JDBC 编写数据库应用程序的开发者可以真正做到"编写一次,随处可用"(write it once and run it anywhere)。

JDBC 扩展了 Java 语言的功能。例如,使用 Java 和 JDBC,可以制作一个能够从远程数据库上获取信息的具有 Java 应用小程序(Applet)的 Web 主页,或者开发一个能够使企业的员工通过企业内部网(Intranet)访问一个或多个内部数据库的应用。

采用 Java 和 JDBC 将使信息的收集和发送非常容易和经济。公司和企业可以继续保留他们原先的数据库系统,并且哪怕这些数据库系统是互不相同的,仍然可以轻

松地访问上面的信息。同时，开发一个新的数据库应用的时间也将大大缩短。

(2)JDBC 的体系结构。JDBC 的体系结构包含四个组件如图 1 所示：

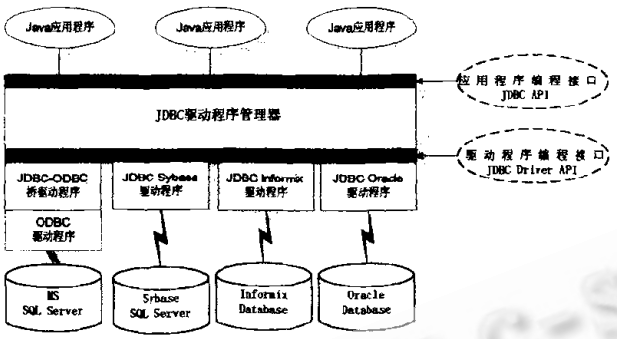

图 1　JDBC 体系结构

·JDBC 应用程序。JDBC 应用程序负责用户与用户接口之间的交互操作，以及调用 JDBC 的对象方法以给出 SQL 语句并提取结果。

·JDBC 驱动程序管理器 JDBC 驱动程序管理器为应用程序加载和调用驱动程序。

·JDBC 驱动程序。JDBC 驱动程序执行 JDBC 对象方法的调用，呈送 SQL 请求给指定的数据源，并将结果返回给应用程序。驱动程序业负责与任何访问数据源的必要软件层进行交互作用。

·数据源。数据源由数据集和与其相关联的环境组成，主要指各数据库厂商的数据库系统。

象 ODBC 一样，JDBC 提供给程序员的编程接口是由两部分组成，一是面向应用程序的编程接口 JDBC API，一是供底层开发的驱动程序接口 JDBC Driver API. JDBC API 是为应用程序员提供的，而 JDBC Driver API 则是为各个商业数据库厂商提供。各个商业数据库厂商的 JDBC 驱动程序是由 JDBC 驱动程序管理器自动和统一管理的。

3. RMI 工作原理简介

在一个分布式系统中，不同主机上的程序之间的通信，可以通过直接对 Socket 编程，这是比较底层的方法；也可以使用层次更高一些的 RPC(Remote Procedure Call)机制来实现。但是对于 Java 这么一种纯面向对象的网络编程语言来说，这些方法和机制都不可避免地破坏了 Java 的面向对象的特性。怎样去解决这个问题，实现分布式对象环境下的不同虚拟机上的对象之间的相互引用呢？这就是 Java 的 RMI 技术所要做的事情。

(1) RMI 概述。RMI 就是远程方法调用(Remote Method Invocation)，是 Java 体系中新兴的一门技术。Java 是一种纯面向对象的语言，每一个程序实体都是以对象的形式提交的。为了使分布式环境下的 Java 编程保持这一特性，Sun 公司提出了分布式对象的概念，并且为分布式对象(即远程对象)之间互相引用提供了适合对象语义的基于抽象的对象层次的机制，这就是 RMI。它使得分布式环境下的 Java 编程或者说 Java 程序之间的网络通信，不再涉及底层的协议和数据的编码解码，从而更加简单而自然。

(2) RMI 的体系结构。如图 2 所示，RMI 系统由以下三层组成：存根/架构(stub/skeleton)层，远程引用层，和传输层。每一层的边界是由特定的界面和协议来界定的；因此，每一层相对于它的相邻层是独立的，也就是说，某一层在实现上的改变，是不会影响到其他层的。

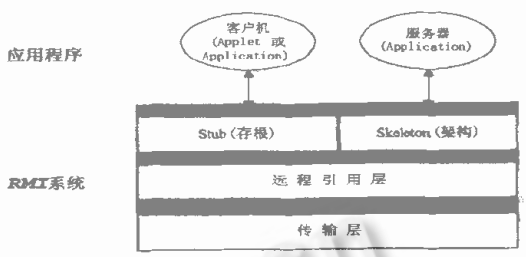

图 2　RMI 的体系结构

stub/skeleton 层是 Java 程序跟 RMI 系统其他部分的接口。在 RMI 系统界面，这一层实际上是扮演着代理的角色的。stub 和 skeleton 分别是客户程序和服务器程序的代理人。在客户端，客户程序向 stub 发出远程方法调用请求，stub 则负责把这些请求传递给服务器，并且把调用结果返回给客户程序。而在服务器端，来自客户端的调用申请为 skeleton 所接收，并且由 skeleton 负责调用相应的对象方法为之服务，调用结果也是由 skeleton 返回给客户端。

远程引用层是具体的从语义上实现远程对象引用的部分，所有的包括对象的复制等对远程对象操作都由远程引用层来管理。

传输层则负责侦听进来的调用请求，为它们建立连接，并且负责管理和监视这些连接；同时为所有存在于该

地址空间的远程对象维持一张远程对象表，以便定位远程调用的目标。

(3)对象序列化技术。RMI 系统中还用到一种技术：对象序列化技术(object serialization)。对象序列化实质是对 Java 对象的编码，在远程方法调用过程中，它负责把发送方要传送的 Java 对象先编码成字节流，然后再进行网络传送，而在接收方它负责把接收的字节流还原成 Java 对象，从而实现了不同地址空间上的对象的透明传送。

(4)RMI 的目标。分布式对象的互操作性。RMI 技术是针对 Java 语言的，它的目标就是实现对存在于不同虚拟机上的对象的无缝的远程调用，把分布式对象模型自然地集成到 Java 语言里，尽可能的从语义上保留 Java 的面向对象的特性，也保留 Java runtime 环境所提供的安全性。

在 RMI 体系里，所有的远程方法调用的实现细节对 Java 程序来说都是透明的，远程对象之间的引用，就跟引用本地的 Java 对象一样的简单自然，符合面向对象的语义。

在 RMI 里，很重要一点，对远程方法的调用，在语法上，跟对本地方法的调用是一致的。

4.基于 JAVA 技术的 Web 环境下分布式数据库互操作性的实现

(1)JDBC 对异种数据库互操作性的实现。从 JDBC 的体系结构图上我们可以看出，在 JDBC 里要实现对各种异种数据库的访问，有一种简单而又快捷的方法，就是利用 ODBC. ODBC 是目前较成熟的数据库应用技术，它提供了对各种异种数据库的统一的访问接口。因此，我们只需要通过 JDBC－ODBC 桥驱动程序，就以实现对各种关系数据库的透明访问，从而使 JDBC 应用程序以及其驱动程序都获得了对数据库的独立性。

(2)传统 Client/Server 二层结构对 Web 环境下分布式数据库应用的局限性。传统的 Client/Server 二层结构在开发 Web 环境下的分布式数据库应用时(如图 3)，因为客户端与服务器端采用的是直接访问方式，如果客户端运行的是 Java Applet，由于 Java 语言只允许 Java Applet 程序访问它原来所驻留的主机源节点，不能对分布在网络上其他的数据库进行存取，这就使得应用大受限制，而且当大量的客户程序都对同一服务器主机进行访问时，会导致服务器节点的负载过重。

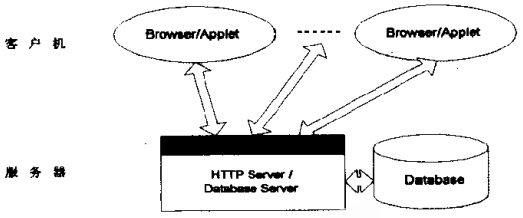

图 3　传统的二层 Client/Server 结构在 Web 数据库的应用模型

(3)利用 RMI 解决 Java 对 Applet 的安全性限制问题。对于 JDBC 应用程序来说，RMI 最大的优点是它可以超越 Java Applet 程序所受的网络安全方面的限制。Applet 作为 RMI 系统的客户端部分，是从 RMI 服务器所在的主机节点下载的，它可以访问 RMI 服务器上的资源，可以向服务器发出数据库访问请求，和接收服务器返回的数据；而作为 Java Application 的 RMI 服务器程序，它没有象 Applet 那样的限制，它可以访问其他主机上的资源，包括数据库。它负责接收客户 Applet 的数据库访问请求，根据请求通过 JDBC 去访问数据库资源，并且把访问结果数据返回给客户 Applet。实际上，RMI 服务器程序成了客户 Applet 访问数据库的一个代理，屏蔽了 Java 对 Applet 的安全限制。

(4)利用 RMI 和 JDBC 实现一个三层 Client/Server 结构模型。对传统的二层结构的一个改进办法，就是采用 Client/Server 三层结构，在客户端与数据库服务器之间增加一个“中间层”，客户端所有的数据库访问都由中间层执行，访问的结果数据由中间层从数据库服务器接收，并负责转发给客户端。中间层实际上是起着一个代理的作用。

在(3)里面我们所说的其实就是这么一个三层结构(如图 4)。

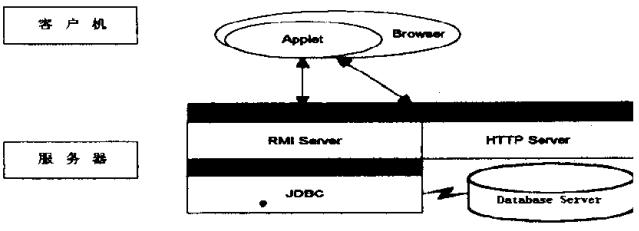

图 4　利用 RMI 和 JDBC 实现的层 Client/Server 模型

RMI 服务器是作为客户和数据库服务器之间的中间层，负责它们之间的数据通信。客户 Applet 的数据库访问请求，通过 RMI 服务器来向数据库提交，而访问结果，也是由数据库服务器通过 RMI 服务器向客户 Applet 传递。RMI 服务器其实也是一个 JDBC 应用程序，它通过 RMI 系统与客户 Applet 进行通信并为它服务，通过 JDBC 与数据库服务器建立连接，从而实现了客户 Applet 与数据库服务器之间的连接。

使用 RMI 与 JDBC 结合的方法实现 Web 环境下的远程数据库访问，还有一个明显的优点，就是作为客户的 Java 程序，不再需要进行低层的 JDBC API 调用，而是由服务器提供了相应的高层 API，而作为 RMI 服务器，完全可以对客户 Applet 的请求加上一定的约束，增加了系统的灵活性和安全性。

(5)实现一个 n 层 Client/Server 结构。上面所述的三层模型，是我们开发 Web 环境下的分布式数据库应用的基础。基于 RMI 支持点对点通信机制的特点，即一个节点上的 RMI 程序既可作为服务器程序又可以作为客户机程序，所以，我们完全可以实现一个 n 级客户机/服务器模式的应用(图 5)。

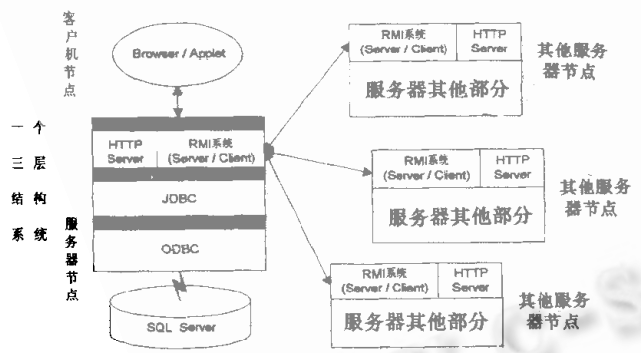

图 5　N 级客户机/服务器模式在分布式数据库中的应用

这种应用系统，由许多分布在不同主机节点上的服务器以及它们的直接客户机所组成，每一个服务器与其直接客户机构成如(4)所述的一个三层结构系统。服务器由 HTTP 服务器、RMI 系统(Server/Client)、JDBC、ODBC 和 SQL 数据库服务器组成，每一个服务器通过 RMI 系统与和它相连的其他服务器以及直接客户机进行通信，在整个系统里面，它既作服务器，为直接客户和其他服务器提供数据服务，又作客户机，可以向其他服务器申请数据服务。

在实际应用时，每一个服务器上都维持着一张表，上面记录了与它相连的其他服务器的地址，每一个服务器的直接客户机可以向其服务器发出数据库访问申请，服务器首先到自己节点的数据库去查询，如果存在相应信息，则将它返回给客户机，否则，就按照地址表向其他服务器发送查询请求；如果其他服务器上有相应信息，也把它们返回给客户机，否则，告知客户机查询失败。明显地，所有的客户机都能够透明访问所有服务器上的数据库。这样，我们就实现了一个 Web 环境下的分布式数据库的应用。

5. 总结

利用 JDBC 和 RMI 实现 Web 环境下的分布式数据库互操作性，是我们应用 Java 技术的一个尝试，同时，也是对传统 C/S 体系结构的一种改进。它具有着简单高效的优点，结合 Java 的各种良好特性，特别是新兴的 RMI 技术，和为业界所支持的 JDBC 技术，相信它会向着标准化的方向发展，并且会得到更加广泛的应用的。

参考文献

[1] Rasmussen. B, "WDB - A Web Interface to SQL Databases", European Southern Observatory, http://arch - http. hq. eso. org/bfrasmus/wdb/wdb. html, 1997

[2] Tan Naguyen, V, Srinivasan, "Accessing Relational Database From the World Wide Web", ACM SIGMOD '96 Montreal, Canada

[3] Sun JDK1.1.X Release Documents, 1997

[4] Art Yaylor, "JDBC Developer's Resource", Prentice Hall PTR, http://www.prehall.com, 1997

[5] Don Burleson, "APIS: Linchpin of Database Connectivity", Database Programming & Design, 1996

(来稿时间:1998 年 6 月)