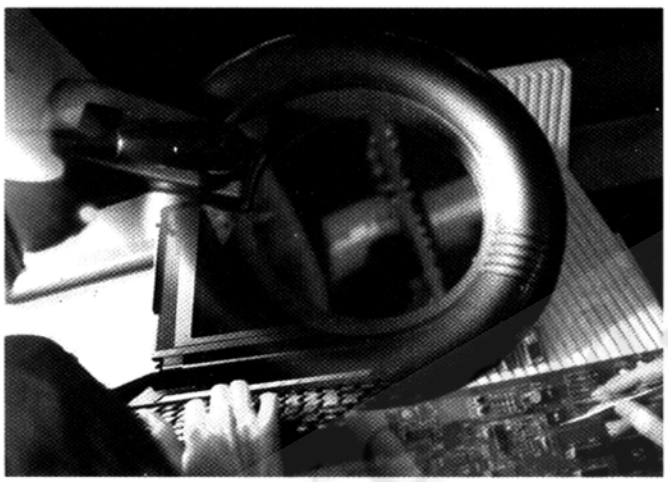
多层数据库与MIDAS技术的应用开发



贾代平 （烟台东方电子信息产业股份有限公司 264001）

摘要：多层分布式结构是当前大型数据库应用的一个热点，本文首先分析了数据库应用采用多层结构的客观原因，接着给出了多层结构的基本划分方法，最后重点结合多层分布式应用服务即MIDAS技术探讨了多层数据库应用的构架及其开发方法。

关键词：多层分布式结构 MIDAS 企业对象 应用服务器

数据库应用经历了从单机到网络、从网络到客户/服务器结构、再从C/S结构到多层（Multi-Tier）分布式结构这样几个发展阶段。目前C/S结构如火如荼，多层数据库应用初见端倪。

1 为什么需要多层结构?

客户/服务器结构推动了数据库应用的普及，但随着数据库应用规模的扩大，特别是今天的应用范围由LAN扩展到WAN，甚至扩展到Internet时，传统的客户/服务器两层结构日益显示出它的弊端来。主要表现在以下几个方面：

(1) 执行效率无法满足日益膨胀的客户需求。客户/服务器结构中的客户数量受到一定的限制，一般而言，当客户数量由数十个增加到数百个时，系统的执行效率便开始显著下降，主要原因是大量的数据库并发连接消耗了宝贵的数据库资源，而这种连接往往使用率非常低。

(2) 客户/服务器结构的维护成本较大。首先是硬件成本，在数据库服务器端相对不变的情况下，要提高系统的执行效率，往往需要对客户机的硬件进行升级；其次是软件成本，包括应用软件的维护成本和系统的扩充成本，在客户/服务器的体系结构下，随着系统规模的扩大这两种成本会显著地增加。另外，随着应用需求的变化，整个系统缺乏灵活性。

(3) 大规模的应用系统在负载平衡能力上显得力不从心。虽然目前多数的数据库系统都支持分布式应用，但仍然需要客户端的应用来实现负载的均衡。当数据库连接成为系统的负担时，负载平衡能力就显得非常必要了。

上述客户/服务器结构的缺点在多层应用系统中都会得到比较好的解决，除此之外，两者相比较，后者还具有如下几个明显的优势：

① 多层应用的客户端属于瘦客户模式。从应用推广的角度来看，瘦客户意味着无需在客户端做复杂的安装和配置，大大减轻了软件使用者的负担。这种模式正发展成一种趋势，它代表着系统应用逐渐走向成熟、简洁、灵活。

② 多层应用可以实现更好的安全性。对于大多数数据库应用来说，系统的安全性都是被放在很重要的位置上。多层结构采用安全验证多级划分的分布式安全管理机制，不仅使系统的数据都得到更加安全的保护，也使系统的应用软件得到保护，因为软件的关键部分都集中在中间层。

③ 多层结构可以更好地支持分布式计算环境。从应用的发展趋势来看，分布式的业务促使我们的软件采用分布式的体系结构，在这种体系结构下我们可以做到游刃有余地满足绝大部分的应用需求。

④ 多层结构可以减少网络开销。在两层结构中，对于典型的一次的事务处理（Transaction），客户机与数据

库往往要进行多次交互。在多层结构中，可以通过中间层，客户机的数据交互只需要一次，数据的多次交互可以限定在中间层与数据库之间，这样网络上的数据流量就会大大减少。

⑤ 多层结构可以部分地消除数据库瓶颈。虽然数据库的并发度不能无限制的提高，但多层结构中可以通过增加中间层的并发度、分解数据库承担的计算任务、共享数据库连接等措施来降低数据库的负荷，从而提高数据库系统的运行效率。

2 数据库应用系统的任务分割

任何数据应用从逻辑上可以概括地分为用户界面、商业规则和访问控制、数据存储这样三个基本任务。这三种不同的任务可以在一个程序里实现，也可以分割在几个程序里实现。在数据库的多层应用中，三层结构是应用的基本方式。如果我们将上述三种任务分别用不同的软件来实现的话，这就是典型的三层应用结构，这三种软件都是为整个应用系统服务的，我们称之为系统的表示层、商业规则层、数据层。

表示层是一个人机交互的接口，它提供给用户一个可视化的界面，借此使用者可以下达系应用软件操作的指令，如数据的获取、数据的输入、数据的修改、数据的有效性验证等一系列操作。

数据层是一个相对独立的层面，它负责执行数据的存储管理、数据的完整性、数据的安全性等任务，另外它还需要响应数据操作的请求，产生数据操作的结果。这一层大多数的应用系统采用成熟的数据库管理系统来实现，如 Oracle、Sybase、MS SQL Server 等。

商业规则层位于表示层和数据层之间，是两者沟通的桥梁，因此又叫中间层。从表示层看，它是服务器，从数据层看，它又是客户端，因此它是两套客户/服务器结构的交汇点。从功能上看，它控制数据的访问、传递表示层和数据层进行信息交互所需要的指令和数据。从物理上看，它把用户和数据隔离开来，通过其内部封装的企业对象来完成表示层和数据层之间的数据交易，从而实现应用系统更好的安全性和灵活性。为了减轻表示层和数据层的负担，在多层结构中，可以把原先两层结构中存在于客户端或数据库端的数据处理和业务逻辑迁移到商业规则层（比如将原先数据库端普遍存在的存储过程移向中间层），这样在整个应用系统中，数据的处理趋向集中，非常有利于后期的软件维护和系统升级。

在多层结构的应用系统中，典型的情况是采用三层结构（Three-Tier）。如果根据实际需要将中间层再分成若干个层次，如将承担负载平衡的部分和负责数据存取的部分从中间层分离出来，则构成真正的多层结构（Multi-Tier）。

3 MIDAS 中间件技术

所谓中间件（Middleware），通俗地说，就是在多层结构中，任何不在客户端和（数据库）服务器端执行并且担负有特定功能的应用程序，例如在B/S结构的动态网站中，Web Server就是一种我们常见的中间件。在数据库的多层应用中，中间件是系统实现的核心部件，它担负着系统的关键性任务。在传统的采用两层结构的系统中，所有的客户端都是直接与数据库服务器连接的，这种连接属于有状态的连接[1]，这就决定了系统的主要瓶颈取决于数据库服务器。正是由于中间件的出现，使得我们可以克服在两层结构中的诸多弊端。在多层结构中，由中间件技术实现绝大部分的商业规则和业务处理，由此诞生出一个或多个担负企业运算的服务器，我们统称为应用服务器。

MIDAS 是 Multi-Tier Distributed Application Services Suite的英文缩写，是Borland/Inprise公司用来实现分布式多层数据库应用的通用中间件产品，它是我们在视窗条件下用 Delphi 或 C++Builder 来开发多层应用系统的中介透明引擎，它提供客户端程序和应用服务器之间互通数据库信息的机制。通过 MIDAS，我们不仅可以开发出高效率的应用服务器，也可以通过它来改造我们的客户端，为应用程序“减肥”，使之可以连接不同类型的应用服务器，如独立于平台的 CORBA 服务器或微软的 MTS 服务器。另外，经过减肥的瘦客户应用程序可以比较方便地发布到互连网上。

MIDAS 在开发数据库的多层应用方面，有如下几个主要功能：

(1) 远程数据代理（Remote Data Broker）：通过这个中介桥梁，实现客户端非直接地存取数据库，并管理客户端与远程数据库服务器之间的数据流。

(2) 数据约束代理（Constraint Broker）：部分地代替数据库本身的 Constraint 功能，让数据在未提交到数据库前就执行必要的数据校验和有效性检查，从而提高系统效率。

(3) 企业对象代理（Business Object Broker）：允许我

们开发出符合商业逻辑和企业规则的智能对象，为客户端提供可重用的服务。

事实上，上述三项功能也是任何中间件产品所应该具备的基本功能，在此不再赘述。除此之外，MIDAS还为我们提供了另外两个重要功能：容错能力（Fault Tolerance）和负载平衡能力（Load Balancing），这是大型的关键性多层数据库应用所必须赋予的专门功能。实践中，我们可以通过MIDAS为应用服务器设置Broker Server来实现这两项功能。为了说明方便，我们给出图1所示：

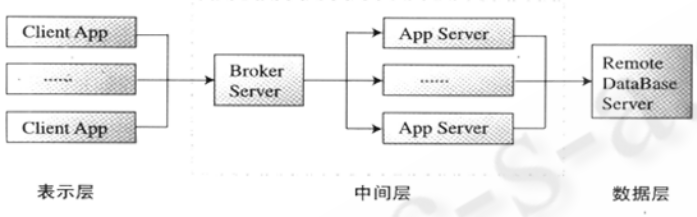


图 1

从上图我们可以看出，在一个大型应用中，我们可以为中间层设置多个应用服务器Application Server，其中的容错能力和负载平衡能力主要通过Broker Server来实现。

在上述系统结构中，每个应用服务器需要向Broker Server进行注册。当客户端应用程序需要连接服务器时，它不是直接与应用服务器打交道，而是通过Broker Server寻找与之相匹配的应用服务器，一旦找到，客户端就会与这个特定的应用服务器建立通信连接。Broker Server的作用主要体现在这个查找过程中，系统的容错能力和负载平衡能力就是在这个查找过程中得以实现。

首先来看负载平衡能力。在一个繁忙的分布式系统中，一个应用服务器往往会不堪重负，此时就需要有多个应用服务器来协同承担服务任务。此时大量客户端的访问需求如何在应用服务器之间分配呢？这就需要有一个调度机制。当某个客户端需要建立连接时，Broker Server就会根据各个应用服务器的负荷情况分配一个适当的应用服务器与之建立连接。从整体上看，大量的服务需求就被分布到不同的应用服务器中去执行。通过这个“调度”机制从而实现系统的负载平衡。

再来看系统容错能力。当连接的应用服务器在运行过程中由于某种原因出现故障时，客户端的程序调用就会出现错误，一旦发现这种状态，Broker Server就会启动连接"向导"功能，在网络中寻找另外一个提供同样服务且运行正常的应用服务器，如果找到，Broker Server就会将客户端的连接引向这台无故障的应用服务器。这一过程对于客户端应用来说是完全“透明”的，也就是说客户端应用程序不会意识到这个搜寻的过程。这就是三层结构通过Broker Server实现系统容错的基本原理。当然实现这个容错功能的前提是客户端应用程序与应用服务器之间的连接必须是无状态的连接，否则客户端应用在执行过程中更换应用服务器会导致运行错误。

构造好了应用服务器，客户端的应用程序如何与之建立连接呢？这就要依靠通信协议。MIDAS支持多种通信协议，如TCP/IP、DCOM、IIOP（CORBA）、HTTP等，并且将这些协议的实现细节封装成组件的形式供开发者使用，这就大大减轻了开发人员的负担，减少了客户端应用程序为建立通信连接而需要编写的大量代码。

4 MIDAS多层应用的体系结构

有了对多层分布式数据库应用的总体认识后，我们来具体介绍MIDAS在开发应用服务器和客户端应用程序的框架结构。

目前在Windows平台下支持MIDAS技术的开发工具主要有Borland/Inprise公司的Delphi和C++Builder，两者在很多方面都采用了相同或相似的技术，如它们共用相同的基础类库VCL，体系结构一脉相承，为开发人员提供的组件和类也都是一致的。不同之处在于具体的编程语言，Delphi使用的是Object Pascal，C++Builder使用的则是Borland/Inprise公司扩展的C++语言。

开发多层数据库应用主要涉及数据集和MIDAS两个方面的组件，图2是基本的结构示意图。

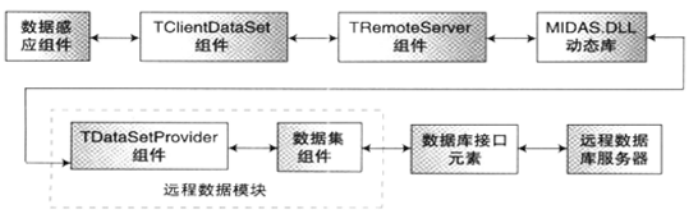


图 2

在这个结构示意图中，我们给出Borland/Inprise公司构造三层应用的基本框架，为简化起见，图中没有包含承担负载平衡的Broker Server部分。下面就各个部分的功

能和作用做出简要说明。

客户端应用程序主要有三部分构成：数据感应组件、TClientDataSet组件和TRemoteServer组件。数据感应组件如TDBGrid、TDBNavigator等，它负责数据的界面显示，构成软件的用户界面。TClientDataSet和TRemoteServer主要负责连接远程应用服务器，其中TClientDataSet是远程数据模块中数据集组件在客户端的映象，TRemoteServer通过特定的连接协议负责与指定的应用服务器建立连接，实际连接的组件有：TDCOMConnection、TSocketConnection、TWebConnection、TCORBAConnection等。除此之外，要连接客户端和应用服务器，实现数据的正确交互，两端都需要MIDAS构架的一个核心动态库MIDAS.DLL，它负责将客户端与应用服务器端需要传递的数据转化为数据封包（Data Packet），然后再通过网络发送给对方，当然数据封包到达对方后，MIDAS.DLL还需要解包。这种数据传递形式，一方面压缩了网络上需要传递的数据量，另一方面也一定程度上保证了数据的安全性。

应用服务器主要由远程数据模块（Remote Data Module）和封装了商业逻辑的企业对象（Business Object）两部分组成，其中企业对象在上述图示中没有明确画出。整个应用服务器对客户端来说表现为一个Automation Server，由远程数据模块实现IAppServer接口，并为客户端提供DCOM、TCP/IP等多种连接方式。远程数据模块中的TDataSetProvider和数据集组件TDBDataSet则通过数据库接口（如BDE、ADO、SQL Link等）连接远程数据库服务器，并将取得的数据通过IAppServer接口以数据封包的形式传递给客户端。另外可以根据需要在远程数据模块中加入TDatabase组件和TSession组件，以便对数据库连接和会话做进一步的控制。这个IAppServer接口在MIDAS技术中扮演着非常重要的角色，只有通过它位于另外一台机器中的MIDAS客户端程序才能通过网络远程调用应用服务器提供的一系列服务，而这些服务则主要由内涵于应用服务器中的各种企业对象提供，应用需求中的任何业务规则和企业计算都可以编写为专门的企业对象封装起来。这里的企业对象与使用何种语言编写是无关的，但必须符合OMG组织（Object Management Group）定义的规范标准，可以是COM/DCOM/COM+对象，也可以是独立于平台的CORBA对象，这是从技术角度划分的。如果从企业对象在服务器中的作用来分，在一个大型应用系统中，这些企业对象可能包括负责数据存取的数据对象（Data Objects），代表业务主体的实体对象（Entity Objects），负责处理企业逻辑的规则对象（Rule Objects），提供特别计算或服务的功能对象（Functional Objects），以及为了系统协调运行的控制对象（Control Objects）等。在此需要特别说明的是，在MIDAS应用服务器中，IAppServer接口和企业对象是两个关键因素，企业对象提供后台服务资源，而接口负责对外输出服务的功能调用，两者相互配合，才能完成应用服务器的使命。

从以上的介绍可以看出，开发一个实际的分布式多层数据库应用并不困难，关键是要理解多层体系结构的设计思想和一系列相关的概念，特别是对应用服务器与企业对象的深刻理解。在此基础上，采用适当的开发工具和技术（如本文介绍的C++Builder/Delphi和MIDAS）就会达到事半功倍的效果。■

参考文献

1 Delphi 5.x分布式多层应用系统篇，李维，机械工业出版社，2001.1。

2 Borland C++Builder5 Developer's Guide Borland/Inprise公司，2000。