C/S 结构业务系统平移到多层体系结构技术的研究

吴 勤　徐 进　（北京理工大学计算机科学与工程系 100081）

摘要：比较多层体系结构与传统 C/S 结构的特点，以商标注册与管理系统为例，探讨从 Client/Server 结构平移到多层体系结构中所涉及的技术。并具体说明客户端、应用服务器和数据服务器的设计、实现和优化。

关键词：C/S 结构　多层体系结构　中间件　Oracle 数据库　Delphi Midas

1　引言

目前，国内应用系统大多数是采用的 Client /Server 体系结构，在局域网环境下，应用被安装在客户端，数据服务器来存储和操作数据。随着网络技术的不断发展，Internet 技术的广泛应用和普及，大量应用系统从封闭式的内部处理，逐渐发展为更广泛的在线网络式服务。C /S 结构的应用系统，由于自身结构的局限性，在业务处理逻辑和交互表示逻辑方面的界定不清，成为许多应用软件进一步发展的瓶颈。多层分布式体系结构，将应用系统分为表示逻辑、业务逻辑和数据服务等多个层次，可实现异构环境下的事务操作，以及处理逻辑的平台无关性，成为事务处理应用系统软件的主流模式。

然而，重新开发新模式的应用系统，会带来大量的人力和资金投入，如何在保护已有资源的情况下，将局域网 C /S 结构的应用系统，经过尽可能少的改造，平移为多层结构，是本文的主要研究议题。

2　多层体系结构的主要技术特点

多层系统结构定义应用程序层没有一成不变的形式。系统可以支持多种的配置。在多层体系结构中包括：

(1) 表示逻辑：定义操作界面的显示内容以及界面的操作处理；

(2) 业务逻辑：确定应用的业务过程，通过业务的处理逻辑与具体的应用处理程序交互；

(3) 基础设施服务：提供应用程序要求的其他功能，如消息、事务支持、邮件服务等；

(4) 数据层：数据存储的位置。

多层体系结基于组件式开发，支持分布式应用。客户端运行用户界面；应用服务器上运行业务逻辑；数据库服务器运行数据库引擎。系统结构如图 1 所示。

在实现上，客户端可以是 JavaApplet，CORBARClient，AxtiveX，通过 IIOP 协议调用应用服务器组件；应用服务器端运用 EJB、AxtiveX 或 CORBAComponents，建立数据逻辑组件，通过 ODBC、JDBC 等访问数据库；数据库可以是 ORACLE、DB2 或 SYBASE 等 RDBMS。

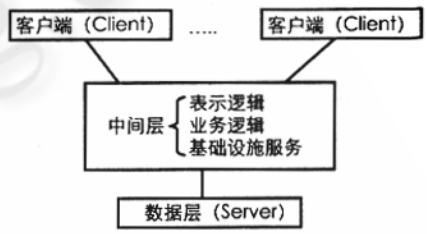

图 1　多层系统结构示意

多层体系结构，采用分离软件成份为相对独立的成份，分离策略和处理，分离接口和实现，使系统结构更趋于稳定和更具可维性，也为建立真正意义上的分布式协同计算结构打下基础，在当前的应用中占据主导地位。

3　具有多样化客户表现力的组件运行平台

目前，支持多层体系结构的分布应用框架主要有三项具有代表性的主流技术，即 Sun 公司的 J2EE（EJB /RMI）、OMG 的 CORBA 和 Microsoft 的 COM /COM + 标准。

J2EE 标准，提供平台无关、可移植、支持并发访问和安全、基于 Java 的服务器端中间件标准，给出了开发面向企业分布应用的规范，支持 JRMP 和 IIOP，包括 Java Servlet、JSP、EJB 等多种形式。

CORBA 是分布异构系统互操作的工业标准。定义了组件模型与元模型、容器结构、编程模型、组件定制与部署等。

COM 是微软支持基于组件的软件开发的核心，在此基础上扩展了 DCOM，COM +，MSMQ，ActiveX Controls 等相关技术。

从用户的角度看，信息和服务应该被有机地结合。简单的信息型服务，仅使用简单的导航和链接，提供只读的内容；而复杂的功能性综合型服务，则必须提供信息下载、交互、计算服务、数据库或数据仓库查询等复杂多样表现形式。要求组件运行支撑平台具有多样化的客户端表现能力。

综合分析三项主流技术，J2EE 和 COM 都提供了复杂多样表现形式的支持技术，如 JSP、ActiveX 等，而 CORBA 则需要集成其他的客户端表示机制。

由于待平移开发的项目是采用 delphi 开发的商标注册与管理信息系统，从资源和业务整合的角度看，应充分考虑现有资源的合理利用，以及平移的适应性。delphi 支持基于 COM 的技术，包括 COM 服务器和客户、ActiveX 控件、OLE 以及 MicrosoftTransaction Server（事务处理服务器）等。因此，采用 COM 可以充分利用现有资源，方便实现平移。本文将围绕 COM 技术讨论平移实现的有关技术。

4　平滑过渡到多层体系结构

商标注册与管理信息系统是典型的 C/S 结构的应用，系统客户端平台采为 Win2000 Professional，采用 Borland 公司的 Delphi5 开发的业务应用，数据库采用 Oracle 9i；应用服务器和数据库服务器平台采用 Solaris 8.0。业务的处理规则全部在 Client 端，即胖客户型结构。系统结构和应用结构划分如下：

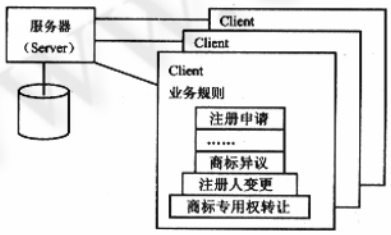

图 2　商标注册与管理信息系统应用结构示意图

4.1　拆分业务规则

拆分业务规则为表示逻辑和应用逻辑，并将它们映射到应用逻辑服务器侧，这是我们解决平移工作的关键。

必须明确，应用逻辑依据业务规则的具体处理，表示逻辑是操作数据的过程。应用逻辑和表示逻辑之间的控制是由业务的行为规则定义的，它决定数据怎样与图象用户界面程序相联系。参看图 3。

拆分后的应用逻辑部分，可以全部放到多层结构的中间层，用 Object Pascal 代码来实现；也可以在服务器端用 SQL 存储过程、触发器和其他数据库对象形式来实现，或者是两种方式的混合实现。

合理分配业务规则为应用逻辑和表示逻辑，需要考虑数据安全性、数据完整性的问题。

（1）拆分的数据访问安全性。由于数据的处理被分配为两部分，不同部分设置需要不同的操作任务，并设置不同的访问权限，以防止远端事务数据访问的不安全因素。

采用分别对视图和存储过程设置不同的对象的访问权限，来实现拆分后的分别管理。权限的设置和存储过程一样，都放在服务器端。通过双重的访问权限约束，可以限制非法的远端事务连接。

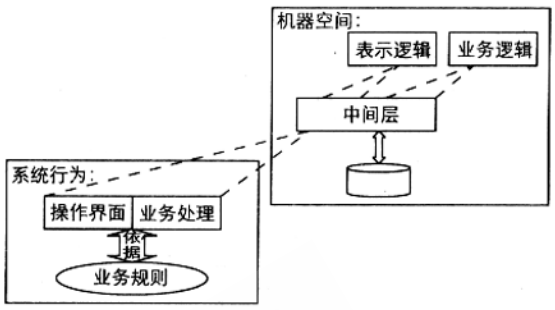

图 3　拆分映射示意图

例如：商标代理事务所帐户的管理和商标费用收取事务处理。在数据库服务端，设置特殊的用户 cw_user，对应收费规则的处理权限；而前端的财务部操作视图也设置相应的授权管理。如图 4 示意。

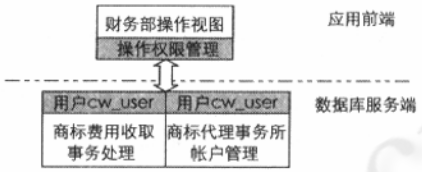

图 4　拆分的数据访问权限设置示例

（2）拆分的数据完整性（切断直接访问数据库的所有途径）。数据完整性是指数据的一次事务处理的正确性，特别指分布处理的出错回馈处理。在 C/S 结构中是强制使用存储过程对数据进行操作，以保证数据操作的完整性。在拆分后的多层结构中，业务逻辑是在服务器或中间层上，必须切断直接访问数据库的所有途径，通过中间层的业务规则，来保证事务处理的数据完整性。

例如，在商标代理事务所管理模块中，如要删除一个事务所，必须先从服务器上查看该事务所是否欠帐。拆分前的处理是在所有的前端，通过存储过程来处理该事务的检查和删除的。拆分后的处理，把商标代理事务所登记表的访问权限从所有的前端收回，并在服务器端，提供负责检查、删除等控制操作数据的存储过程。这样，避免了前端直接访问数据库。无论是正常情况还是出错回馈，对事务的数据处理都是一次性的操作，可以保证一致性和完整性。参见图 5。

4.2　具体实现

根据定义完成后的业务规则，分配客户端、服务器端需要完成的功能。Delphi 的 MIDAS 多层分布式应用服务技术，为开发多层结构的应用程序提供了强大支持。商标注册

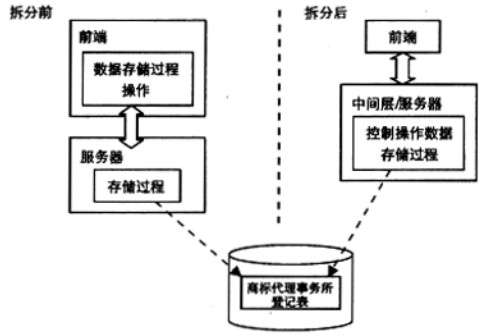

图 5　拆分前后存储过程示例

与管理信息系统平移前后的体系结构如图 6 所示。

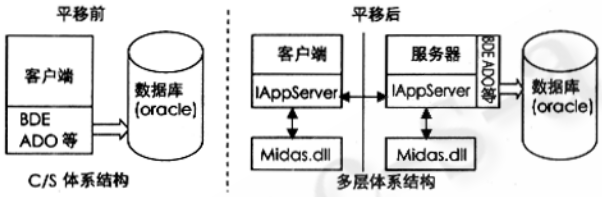

图 6　平移前后系统体系结构示例

客户端是用户界面，数据放在远程数据库服务器上，客户端通过应用服务器发出访问或更新数据请求。

(1) 调整数据库服务器(设置访问权限)。对 C/S 结构下定义的表结构和字段之间的关系、数据类型和用户安全性等进行分析，对数据库对象的访问权限做出相应调整。如：表 D_ACCOUNT 的授权 MODIFY 为：

GRANT INSERT, DELETE, SELECT, UPDATE ON D_ACCOUNT TO CW_USER;

(2) 创建应用服务器。远程数据模块(RDM)是创建服务器应用程序的核心。选择 File | New 菜单命令创建，在对象库的 Multitier 页中双击 RemoteDataModule 图标。设置初始化选项。命名为 SC_Server(审查服务器)。增加组件：

① TRemoteServer 组件：把客户端连到审查服务器 SC_Server 上。

② TProvider 组件：驻留在远程数据模块中，在网络上发布数据表，把 TProvider 组件与 TTable 或 TQuery 组件建立连接，网络上的其他程序可通过 DCOM 访问数据；C/S 应用下的 TABLE/QUERY/SP 移动到这里，一个数据集加 TProvider 构件；把 Dataset 属性设置为 TQuery。转移完成后客户端对 TQuery 中数据的使用，通过捆绑的 TProvider 实现。

③ TDispatchConnection 组件：与客户端通信，指定应用服务器的位置。

服务器应用程序建立之后，保存项目为 SC_Server，注册服务器为 COM。如图 7。

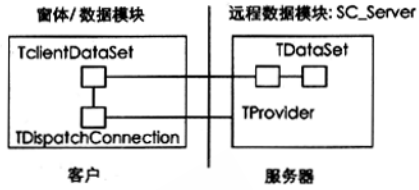

图 7　MIDAS 应用程序

(3) 调整客户端(应用程序的过程、函数改用数据库存储过程实现)。把涉及到数据访问链路的部分去掉。C/S 结构下，客户应用程序调用本地的函数、过程实现对数据库的访问，调整后不依赖于这种方式，改为向业务层发送命令，通过业务层处理事务，业务规则在数据库端用视图、存储过程和触发器实现，实现数据查询、更新。

例如：当商标转让业务核准转让时，修改转让申请表(a_zr)的商标状态为核准。

C/S 结构下：

```
Query1.close;
    Query1.Sql.Clear;
    Query1.Sql.Add("Update a_zr set states
='HZ' where AppNo='200405025'");
    Query1.execsql;
```

在多层体系结构下：

```
CREATE PROCEDURE UPDATE_STATUS(AppNo
Varchar2(9)) As
    Begin
      Update a_zr set states='HZ' where
AppNo=:AppNo;
    End;
```

客户端执行 execute update_status ('200405025')来实现。

5　优化

5.1　客户端的优化

(1) 限制数据包。打开 TClientDataset 时，TClientDataset 的 PacketRecords 属性设定了一次能检索的记录数。MIDAS 会为可视化组件检索足够的记录。可以跟踪客户的状态—通过向服务器传送上一条记录的位置。

(2) 使用公文包模式。把一个文件名赋值给 TClientDataset 的 Filename 属性，如果这个文件已经存在，那么 TClientDataset 会打开该文件的一个本地副本，从应用服务器中直接读取数据。

(3) 发送动态 SQL 给服务器。遵守多层结构的可靠性

原则,分两步进行。先把查询语句赋给 TClientDataset 的 ComandText 属性,选择 Provider. Options 属性的 poAllowCommandText 选项。然后打开 TClientDataset,语句就被传送到了服务器。

5.2 服务器端的优化

(1) 解决记录冲突。当两个用户操作同一条记录,第二个用户试图把记录保存回数据库时会发生错误。可以检测到这种冲突加以解决。

(2) 各种服务器选项。在中间层程序中,TDataset-Provider 的 Options 属性中有许多选项,控制 MIDAS 数据包的行为。例如,poReadOnly 选项可以使数据集在客户上成为只读的。poDisableInserts、poDisableDeletes 或 poDisableEdits 选项可以阻止客户执行相应的操作并触发对应的 OnEditError 或 OnDeleteError 事件等。

6 结束语

本文以商标注册与管理系统为例,比较了不同体系结构的特点,讨论如何将 C/S 模式平移到多层结构。

着重介绍了对数据拆分控制规则,把业务规则分布到多个中间层上,在保证服务器数据的安全性和完整性的前提下,对特定的部门进行职责分工,不同部门授予不同数据库对象的访问权限。多个部门可以共享数据,且只操作与实现他们的特定目标有关的数据。

对拆分数据的集中控制,把业务规则放在中间层,系统可在不影响客户应用程序的情况下对业务规则进行更新。修改存储过程代码时,只要客户端与服务器的接口不变,这些修改对用户来说是透明的。

对平移过程中涉及到的技术和实现细节进行了描述,最后简要介绍了优化的解决方案。通过以上介绍,我们可以看到从 C/S 到多层的平移是可以实现的,很多在 C/S 结构中的工作是可以保留或经过拆分得到利用的。

参考文献

1 J2EE 构件企业系统专家级解决方案,Perrone P J.张志伟、谭郁松、张明杰译,清华大学出版社,2001。

2 J2EE 服务器端高级编程,Subrahmanyam Allamaraju,闻道工作室译,机械工业出版社,2000。

3 Delphi 5.X 分布式应用系统篇,李维,机械工业出版社,2000。

4 Delphi 5 开发人员指南(美)Steve Teixeira,Xavier Pacheco 。