计算机科学2000Vol. 27№ 8

C/S 分布环境下核心软构件的设计与集成

The Design and Integration of Core Software Components in C/S Distributed Environment

刘 升[1] 游晓明[1] 陈传波[2]

(湖北师范学院计算机系 黄石 435002)[1] (华中理工大学计算机系 武汉 430074)[2]

Abstract This paper has analyzed and discussed the layered architecture of application system in C/S distributed environment. Then, the formal definitions of core software components in application system are presented. In addition, the design and integration method for them was given, this method can increase the efficiency of developing applications and it also has reusability widely.

Keywords Software component, Client/server, Distributed environment, Software reuse

1 引言

随着计算机硬件技术的高速发展,操作系统功能的加强,网络技术的完善,开放式网络环境下的客户机/服务器体系结构已成为分布式处理系统的主流,其体系结构如图1所示[1]。

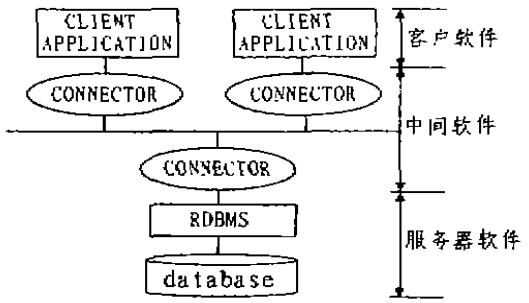

图1 Client/Server 模型体系结构图

C/S 结构要求将整个处理任务根据具体情况合理分布于逻辑上的二级或三级结构上,使 Client 和 Server 能够相互配合,密切协作以便最大限度地发挥 Client 和 Server 各自的工作潜力,更好地满足用户对应用系统的整体要求,C/S 的这种分布式处理思想顺应了90年代以来出现的系统规模适度优化(rightsizing)的潮流,因此得到了快速发展并被广泛采用。

2 软构件的分层结构和形式定义

通过前面的分析,我们知道:根据应用系统中软构件的不同地位和作用,Client/Server 模型试图把一个应用分离成:客户软件、中间软件、服务器软件,与之相对应的便是用户界面、事务处理逻辑和数据处理三大组成部分,这样我们就可以将一个分布式应用 DA 表示成如下三元组的集合[2,3]:

DA=(用户界面,事务处理逻辑,数据处理)

下面我们将给出其主要软构件成分的形式定义:

〈用户界面〉::=〈系统软构件层〉|〈领域软构件层〉
〈系统软构件层〉::=〈窗口对象〉|〈数据窗口对象〉|〈菜单对象〉|〈控件〉
〈领域软构件层〉::=〈领域通用软构件层〉|〈领域专用软构件层〉{〈领域通用软构件层〉|〈领域专用软构件层〉}
〈通用领域构件〉::=〈数据录入构件类〉|〈统计构件类〉|〈查询构件类〉|〈报表打印构件类〉|〈数据编码构件类〉|〈其他通用领域构件〉
〈事务处理逻辑〉::=〈通用中间件〉〈专用服务中间件〉{〈通用中间件〉〈专用服务中间件〉}
〈通用中间件〉::=〈通信堆栈〉|〈分布式目录〉|〈认证服务〉|〈远程过程调用〉|〈其他通用中间件〉
〈专用服务中间件〉::=〈ODBC〉|〈专用接口〉|〈对象中间件〉|〈事务处理中间件〉|〈数据事务处理对象〉|〈数据库转换中间件〉|〈分布式管理系统管理中间件〉|〈其他专用服务中间件〉
〈数据处理〉::=〈表或视图的定义〉〈存取控制〉〈查询优化〉〈一致性检查〉〈安全性检查〉〈完整性检查〉〈存储过程〉{〈表或视图的定义〉〈存取控制〉〈查询优化〉〈一致性检查〉〈安全性检查〉〈完整性检查〉〈存储过程〉}
〈表或视图的定义〉::=〈表或视图名〉〈域描述表〉
〈表或视图名〉::=〈标识符〉
〈域描述表〉::=〈域描述〉{〈域描述〉}
〈域描述〉::=〈数据域描述〉|〈非数型域描述〉
〈数型域描述〉::=〈域名〉〈数型〉〈域长度〉[〈小数位数〉]
〈非数型域描述〉::=〈域名〉〈非数型〉〈域长度〉
〈域名〉::=〈标识符〉
〈数型〉::=numeric | integer | decimal | float | tinyint |

刘升、游晓明 硕士,讲师,研究方向:软件工程和分布式处理。陈传波 教授,研究方向:计算机网络与信息管理工程。

smallint|double|real|smallmoney|money
〈非数型〉::=char|varchar|datetime|nchar|nvarchar|
smalldatetime|text|image|bit|binary|varbinary
〈存取控制〉::=select|update|insert|delete
〈查询优化〉::=〈物理优化〉|〈逻辑优化〉
〈物理优化〉::=索引|排序|散列
〈逻辑优化〉::=〈代数优化〉|〈分布优化〉
〈一致性检查〉::=〈锁模型〉|〈时间印模型〉
〈锁模型〉::=〈共享锁〉|〈排它锁〉
〈时间印模型〉::=〈基本时间印〉|〈保守时间印〉
〈完整性检查〉::=〈属性值完整性〉|〈参照完整性〉

3 软构件的通用性设计

针对模型框架三大组成部分的核心构件，在应用系统开发中，我们以数据库管理系统选择基于 Client/Server 结构的 Sybase，开发工具采用 powerbuilder5.0/6.0为例，给出其通用性设计，以便复用。

(1)**用户界面的设计** 用户界面也是一类构件，其设计大多是针对窗口的，界面的状态通过对其提供的方法和事件的引用而演变，它一般采用面向对象方法设计，具有继承性和封装性。下面具体给出用户界面 UI 的设计定义[5]：

〈UI〉::=W(〈W_name〉:〈method or event list〉,w(〈x〉,〈y〉,〈attr〉))|M(〈M_name〉:〈metbod or event list〉)|DW(〈DW_name〉:〈method or event list〉,DW(〈x〉,〈y〉,〈attr〉)|〈message〉|〈os〉|〈UI〉|〈空〉

〈method or event list〉::=(〈method or event ID〉,〈specification message〉|

〈message〉::=〈注释〉|〈出错信息〉|〈空〉

⋮

DW 在系统中有十一种具体表示形式，如 tabular 格式、freeform 格式、crosstab 格式以及 grid 格式等，菜单类分为3个子类：弹出菜单和下拉菜单、级联菜单，界面生成可以采用菜单树的数据结构将需求分析的结果归纳成多层的功能模型，用多层菜单树形结构进行描述，叶子便是可以执行的功能模块。

用图形界面编辑设计各种界面的对象类、构件和界面构造脚本的描述，脚本描述应包括：(1)各界面在菜单树中的关系；(2)各界面自身特性；(3)方法和触发事件的描述。把各种图形、图像、正文文本、按钮和音响都作为对象类放在窗口中，编写出其脚本描述，指出程序运行时的实际运作。

下面给出一个界面状态转换图(图2)的例子。

M000(系统设置：dw-单位编码维护，zc-职称编码维护，…exit-退出)

M001(单位编码维护：op-操纵，dis-显示，exit-退出)

F000：(操纵：ADD，DEL，UPDATE，SEL，EXIT)

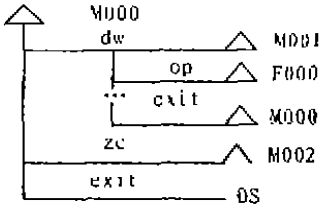

图2 界面状态转换图

构成界面的基本构件都作为系统构件，所以在设计界面时，只要定义所需构件的实例和自定义构件即可，界面定义完备后可由辅助工具生存这些实例。

(2)**数据事务处理对象** 相当于用户界面和数据处理之间的适配器，使用户界面具有很好的稳定性，powerbuilder 提供了缺省事务对象 SQLCA 充当应用程序与数据库之间的通信区域，执行对数据库的事务管理，下面创建了有关用于 Sybase 数据库的事务对象和一个用于 SQL anywhere 的事务对象，用户可以执行对本地数据库和远程数据库的操作，具有一定的通用性。

```
Transaction Tr_sqlanywhere
Transaction Tr_sybase
Tr-sqlanywhere=create transaction
Tr_sqlanywhere.dbms="odbc"
Tr_salanywhere.bparm="connectstring='dsn=rs_loca';
    UId=dba;PWD=sql"
Tr_sybase.dbms="SYC sybase system 10 CTLIB"
Tr_sybase.server name="@S:sybsrv"
Tr_sybase userid="dba"
Tr_sybase.dbpass="sql"
Tr_sybase.dbpamn="connect string='dsn=rs';
    UID=dba;PWD=SQL"
Connect using tr_sqlanywhere;//连接本地库
Connect using tr_sybase;//连接服务器上数据库
```

(3)**通用异构数据库之间的数据转换** 假设 Sybase 中有一名为 DB1的数据库，其中有一名为 table1 的表，本地使用的 sql anywbere 中有一名为 DB2的数据库，其中有一名为 table2表，下面的程序完成由 sql anywhere 到 sybase 的转换。

```
Transaction sqlserver
    Sqlca.dbms="odbc"
    Sqlca database="db2"
    Sqlca dbparm="dsn=db2"
    Connect;
    Sqlserver=create transaction
    Sqlserver.dbms="sybase"
    Sqlserver.database="db1"
    Sqlserver.servername="@s:sybsrv"
    Sqlserver.lodid="sa"
    Sqlserver.logpass="123456"
    Connect using sqlserver
    For id=1 to n
Select a1,a2,…am into:str1,str2,…:strm from table2
    where a1=:id
Insert into table1,(a1,a2,…,am)values(:str1,str2,…:
    strm)using
Sqlserver;
Next
Disconnect;
Disconnect using sqlserver;
```

在上述程序中对有关数据库的事务的对象进行修

改就可以实现其它支持 SQL 语句的 DBMS 之间数据的转换。

同时 powerbuilder 提供了一种数据转换工具 pipeline，利用它也可以方便地完成数据的转换任务。

(4)数据库中表的定义 开发者可以利用数据库服务器提供的数据完整性约束，将标准数据完整性规则的说明作为基表定义的一部分，即可维护数据参照完整性，保证了主细目之间级联更新的正确性。

```
Create table person(
      id char(6)notnull,
      xm cher(8)notnull,
       ⋮


      constraint primary key(id));
Create table rs_zckh(
      id char(6)notunll,
      zclb char(6),
          ⋮
      Serino char(8) notnull,
Constraint primary key serino,
Constraint Foreign key(id)references person(id));
```

(5)触发器 下面定义触发器 update_person，主要用于当 person 表发生更新时，相应的身份表如职称考核、学历教育等也发生变化。

```
create trigger update_person
on person
for update
as
if update(id)
begin
update rs_zckh
set rs_zckh.rybh=inserted.id
from rs_zckh,deleted,inserted
where rs_zckh.rybh=deleted.id
update rs_xljy
set rs_xljy.rybh=inserted.id
from rs_xljy,deleted,inserted
where rs_xljy.rybh=deleted.id
end
```

(6)规则 下面定义中文简拼的规则 RUL_ZWJP，可以绑捆到 PERSON 表的 ZWJP 字段，作为该字段的值域规则：

```
Create rule rule_zwjp
As @zwjp like"[A-Z][A-Z][A-Z][A-Z]"
GO
Sp_bindrule rul_zwjp,"person.zwjp"
```

4. 系统的动态集成

我们设计的构件都遵循 PBL 的形态，同时通过 OLE2.0，DDE 和 DLL 以及 powerbuilder5.0 提供的 library export 和 libraryimport 等函数，我们可以使市场上 OLE/COM/DCOM/Activex 的构件能与原系统实现无缝连接，能存取系统中原有的数据，新的数据库表结构也能通过动态的数据窗口，通过数据事务对象可以调用原有的构件，达到构件软件的桌面集成的目的，这样可以提高应用系统的开发效率，缩短应用系统的开发周期，并能提高软件的可重用性。

参 考 文 献

1 董荣胜. 两层和三层 Client/Server 结构的分析. 计算机工程，1998(6)
2 Gogolla M，et al. A Development Environment for an Object Specification Language. IEEE Trans. on knowledge and data engineering，1995，7(3)：505～507
3 Deubler Hans-Helmath，et al Introducing object orientation into large and complex system. IEEE Trans. on software engineering，1994，20(1)：432～440
4 Prieto-Diaz R，Freeman P. Classfying Software for Reusability. IEEE software，1987，4(1)：6～16
5 诸葛海 面向对象的 MTS 开发方法 DDD. 软件学报，1995，增刊
6 耿刚复，李渊明. 基于构件的应用软件系统的体系结构及其开发模型. 计算机研究与发展，1998，35(7)

(上接第34页)

```
    ch_bj.addItem(rslt.getString(1).trim());
    }
    stmt.close();
    …//其它代码
  }
  }//end init
    …
    public void paint(Graphics g){
    g.setColor(Color.black);
     g.drawLine(x0,y0,x1,y0);
       g.drawLine(x0,y1,x0,y0);
    g.setColor(Color red);
    g.fill3DRect (x0, y0-score [1], 40, score [1],
       true);
     …//直方图的绘制
  }
}
```

Intranet 数据库中分布式处理的实现方案见图3，正是因为 Java 能使我们在 Intranet 上能完成复杂、或简单的各式任务，同时它还可以帮助扩展服务器、客户端的功能，使服务器端能根据需要建立相应的各种通讯方式，又可以使客户端具有处理功能，而不是单纯地依靠服务器，我们可以采用有效的方式，使服务器、客户端各司其职，分工合作，以减轻服务器的负担，提高效率，同时减少不必要的网络数据传输，提高整个 Intranet 的系统性能。

参 考 文 献

1 Mobseu P. Web 数据库开发人员指南. 北京：机械工业出版社，1997
2 Ablan J. 用 Java 开发 Intranet 应用. 北京：机械工业出版社，1997
3 Uanhelsuwe L. Java 从入门到精通. 北京：电子工业出版社，1997
4 许伟. 传统 MIS 模式向 Intranet 转换. 北京·电子与信息化，2000，(3)：31