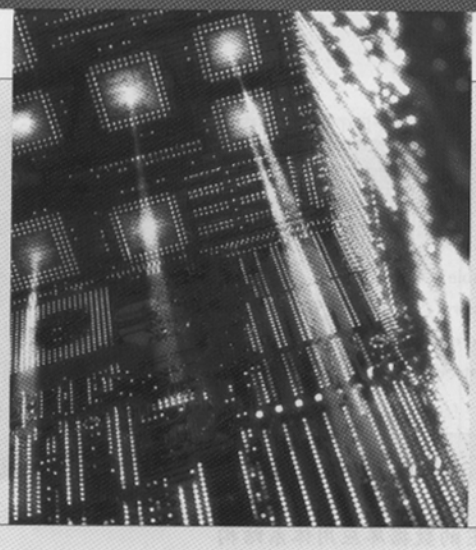

基于ClientDataSet的数据库应用系统设计

Designment of ClientDataSet-based DB Application System

张国权（长沙市劳动保障局信息中心　410011）
彭江平（湖南大学会计学院信息管理系　410079）

摘　要：本文讨论了ClientDataSet组件在多种形式的数据库应用体系统结构中应用的方法与意义，并就若干关键技术问题进行了探索。

关键词：ClientDataSet组件 Delphi MIDAS 数据库应用

1 引言

由Borland公司提供的开发工具，如Delphi与C++ Builder等，由于其在开发数据库应用及网络多层分布式应用方面的特殊表现，再加上系统的广泛开发性，它们正得到越来越多程序员的青睐。本文主要讨论在简单的单层文件数据库系统以及多层分布式数据库系统中都必不可少的一个重要组件ClientDataSet，探讨针对一些在数据库应用设计与开发中可能遇到的一些具体技术问题。

2 ClientDataSet在数据库应用系统设计与开发中的应用

不论是单层、两层C/S或是多层结构中，都可使用ClientDataSet组件，作者认为在使用Delphi或C++ Builder开发数据库应用时，程序员应该多使用该组件。一方面是因为使用ClientDataSet组件能使数据库应用获得强大的功能；另一方面是Borland公司已许诺要在将来不断地增强其功能。事实上，在新版本Delphi与C++ Builder中相比老版本而言，该组件的功能就正在不断增强。借助MIDAS技术的支持，该组件本身就是一个数据库引擎，除没有提供对SQL的支持外，它提供了几乎所有关系数据库的特征。在这里，限于篇幅不可能对此展开更深入的讨论，仅给出ClientDataSet组件在单层或多层数据库应用中的体系结构，并就使用该组件所带来的优势进行简要的评述。

2.1 单机本地“瘦”数据库应用

虽然使用Borland开发工具设计基于BDE的数据库应用简单方便，但是在发布与配置时需要有经验与技巧，而且使用安装工具对所需文件进行打包时有不灵活与冗余文件多等缺点。这些对于开发单机的、小型的应用系统而言，是一个比较大的制约。因此，很多开发者都通过使用第三方组件及其他数据库引擎来部分地解决发布与配置等复杂问题。但是，使用ClientDataSet组件可从根本上解决这些问题。该组件的最新版本可同时支持CDS及XML格式的数据表文件。其体系结构如图1所示。

这种体系结构的数据库应用不需要附带BDE，也不需要进行BDE别名等的设置，应用程序的分发与维护都相当简单，一般只需要进行简单的文件复制就可使用；能够使用ClientDataSet组件所提供的多种高级功能；在数据量不是很大的情况下，因为组件操作的是内存数据，速度非常快。要注意的是这种结构不适于需要操作数据量很大的应用。

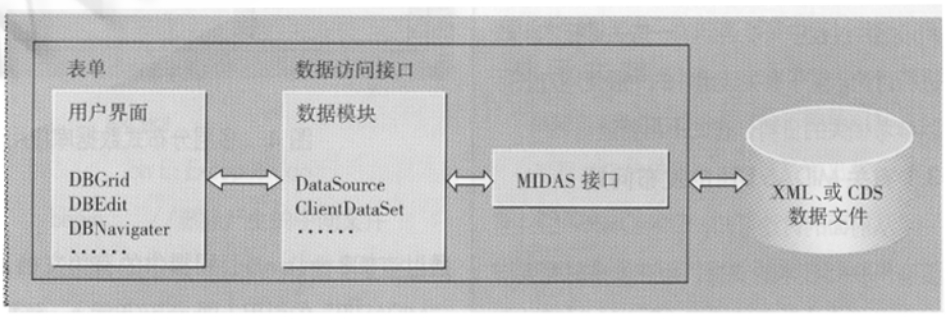

图1　基于ClientDataSet组件的单层数据库应用体系结构

2.2 基于BDE的单机及C/S结构数据库应用

ClientDataSet组件具有许多一般的数据集所不具备的功能，因此，可以在基于BDE的单机及C/S结构数据库应用中使用它来增强应用程序的功能。它主要是通过在图2所示的一般的基

于BDE的单机及C/S结构数据库应用中使用的体系结构中增加“ClientDataSet”与“DataProvider”组件的使用，其结构如图3所示。

在基于BDE的单机及C/S结构数据库应用中，使用ClientDataSet组件可以充分利用其强大的动态统计、过滤及灵活的索引功能；可以代替数据集组件上的Catched Updates功能，实现更有效的缓冲更新；能使应用程序控制数据存取的流量与种类；可以更方便地将多层分布式应用体系结构移植。

2.3 多层体系结构数据库应用

多层分布式体系结构作为大型数据库应用中最为关注的系统结构，随着网络应用普及和管理信息系统的广泛应用，已为越来越多的设计者所关注。MIDAS技术作为Borland公司的多层分布式应用体系结构的核心的不断成熟，已成为实现多层分布式应用最方便的解决方案。其体系结构如图4所示。

这种体系结构在一个共享的中间层封装商业规则；实现数据的分布式处理；客户端可在线或离线工作；远程数据代理、约束代理、商业规则代理等的应用简化客户端的部署与升级功能。

3 ClientDataSet若干技术问题研究

上一部分讨论了ClientDataSet组件在各种系统体系结构中应用的可能性，这一部分就使用该组件过程中可能遇到的一些关键技术问题进行讨论，至于有关该组件的一般使用方法可以参考相关的资料，在此不再赘述。

3.1 有关MIDAS.DLL的发布问题

从前面的讨论可知，ClientDataSet组件需要有MIDAS技术的支持，即在发布时需要同时发布相应的动态链接库（Delphi 3和Delphi 4中是DBClient.dll，Delphi 5以后是Midas.dll），当然，可以随应用同时发布它，但为了保证其正确注册，一般需要将它复制到操作系统的系统目录中，这一工作虽然谈不上复杂，但无形中增加了应用分发的工作量。

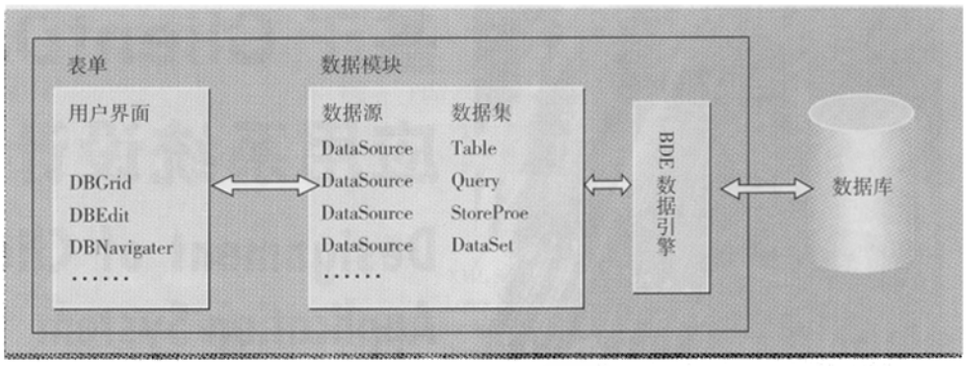

图2 普通的基于BDE的数据库应用体系结构

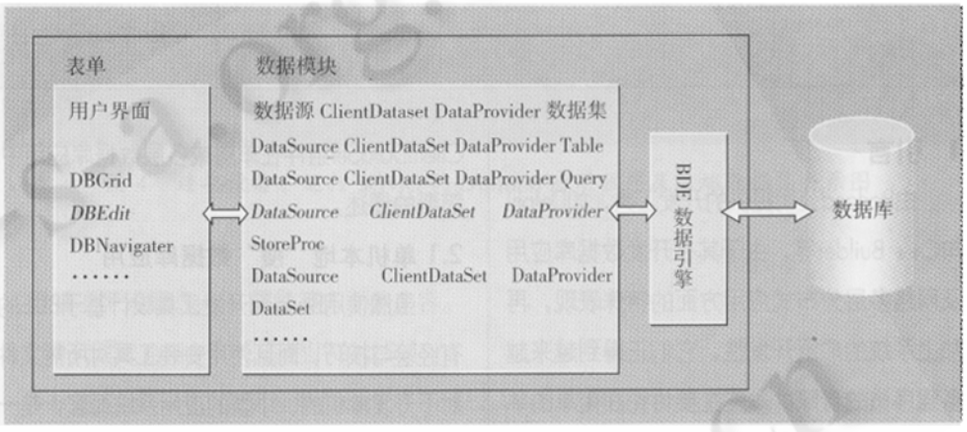

图3 借助ClientDataSet增强型的基于BDE的数据库应用体系结构

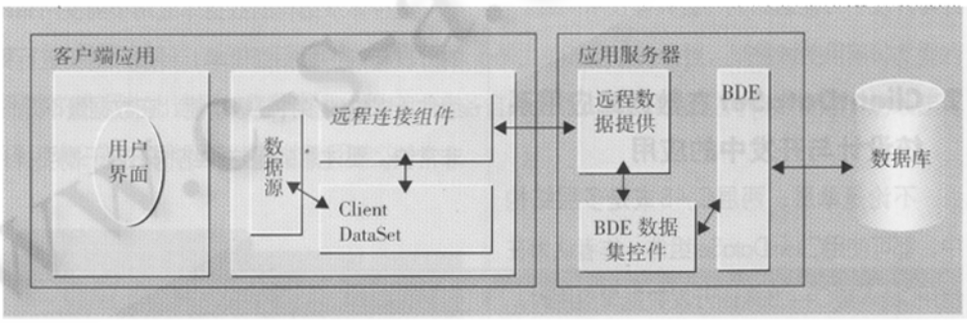

图4 多层分布式数据库应用体系结构及ClientDataSet的应用

通过查阅Borland公司提供的有关文档，可以在应用程序发布前，在主程序中增加对“MIDASLIB”的引用（即use该单元），这种处理虽然会增加应用程序的大小，却可以简化发布的工作，要分发时仅需要分发所设计的应用及应用所处理的数据（XML或CDS）文件。

3.2 灵活的索引与排序机制

在数据库应用中，经常希望能够依任意字段进行排序，这在基于BDE或ADO的普通的数据库应用中实现比较麻烦，但ClientDataSet的内存索引功能却能方便地实现这一功能。通过设置ClientDataSet的“IndexFieldNames”属性为包含一个或多个（用分号分隔）字段，创建

ClientDataSet的即时动态索引，就可很方便地实现依一个或多个字段的排序功能，这在普通的数据库应用中一般需要多次执行不同的SQL语句，实现不方便且效率低。同时，也可以通过代码在ClientDataSet中创建普通索引，并根据相应的“IndexName”设置要使用的索引。例如，下面的代码段动态地为数据集增加了唯一索引，通过与异常处理代理相结合，就可保证插入到数据集中的记录的主键的唯一性。

```
//增加特殊处理的唯一索引
        cds.IndexDefs.Clear;
        cds.IndexDefs.Add('TX01','TX01',
[ixUnique]);//设置由字段TX01建立的唯一索引
//索引名为TX01
        cds.IndexName:='TX01';}
      ......
//错误处理，确保主键的唯一性。
try
        cds.Post;
      except
        on E: Exception do begin
        ShowMessage('主键冲突，处理不成功！');
        Exit;
      end;
```

3.3 事务处理技术的实现

除了作为ClientDataSet组件一个重要特性的缓冲更新(CatchedUpdate)机制外，它还提供了一个非常有用的属性SavePoint这一可读写的属性，通过它可以恢复到上一次保存数据的变动点，所有在该点后的数据变动都可取消。通过巧妙地应用该属性，就可以在单层或多层数据库应用中不使用数据库服务器所提供的事务处理功能，而实现某种处理级的事务处理，简化数据库的事务处理过程与工作量。例如，下面的过程代码实现将cdsCopy到cds两个结构完全相同的ClientDataSet组件间的数据复制，并通过使用“SavePoint”属性将整个数据复制作为一事务处理，如果在复制过程中，出现某一记录不能复制的异常，则放弃整个复制工作。在整个代码段中，要注意的是“SavePoint”的使用以及使用它的前提条件：数据集必须是记录修改日志的，具体请注意代码中粗体的部分。

```
procedure CDSDataSetCopy() //支持事务处理的数据复制
  begin
    cdsCopy.Close;
      cdsCopy.LoadFromFile('.\CopyOfsqsb03_50.xml');
    cds.Close;
    cds.LoadFromFile('.\sqsb03_50.xml');
     cds.LogChanges:=True;//设置由ClientDataSet组件记录数据修改
    savePo:=cds.SavePoint;//在进行数据处理前，设置保存点
     while not cdsCopy.Eof do
  begin
   cds.Append();
      for i:=0 to cds.FieldCount-1 do
  begin
        cds.Fields[i].ReadOnly:=False;
         cds.Fields[i].Value:=cdsCopy.Fields[i].Value;
      end;
      try
        cds.Post;
      except
        on E: Exception do
  begin    //删除产生的中间文件
  cds.SavePoint:=savePo;//数据复制出现异常，恢复到保存点
  cds.LogChanges:=False;//取消修改日志记录
        ShowMessage('主键或字段值不符合约束条件，数据复制不成功！');
        Close();
        Exit;
      end;
    end;
    cdsCopy.Next;
  end;
  cds.MergeChangeLog;//数据复制成功，合并所作改变，并清除修改日志
  cds.LogChanges:=False;// 取消修改日志记录
    cds.SaveToFile('.\sqsb03_50.xml',dfXMLUTF8);
  end;
```

4 结论

本文讨论了ClientDataSet组件在多种形式的数据库应用体系结构中的方法与意义，并对若干关键技术问题进行了探索，在多个层面展示了应用ClientDataSet组件的优势。此方法已应用于某省大型社会保险退休人员社会化管理系统的设计中，大大简化了维护与服务的工作量。相信对正在使用Delphi或C++ Builder开发数据库应用的程序员有一定的启发作用。

参考文献

1 张晓东、李敬，C++ Builder 5程序设计—数据库应用实务篇，中国铁道出版社，2000年。

2 李维，Delphi 5.x 分布式多层应用:系统篇，机械工业出版社，2000年。

3 李维，Delphi 5.x 分布式多层应用:电子商务篇，机械工业出版社，2000年。

4 王涛，多层分布式数据库实践，清华大学出版社，2000年。