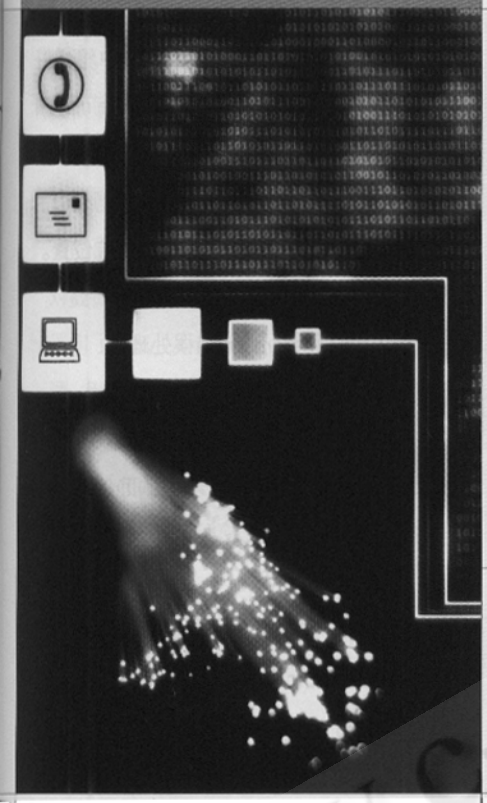

dbExpress 及 ClientDataSet 应用结构中异常处理研究

The Research On The Abnorm Disposal in the Application Sructure of dbExpress And ClientDataSet

彭江平 （湖南大学会计学院信息管理系 410079）

摘 要：基于dbExpress与ClientDataSet数据库应用体系结构的分析，定义了三个层次的异常，给出了不同层次的异常处理解决方案。

关键词：dbExpress ClientDataSet 异常 异常处理

1 前言

对于一个实际运行的大型信息系统而言，应用程序必须具有一定的容错功能。也就是说，在环境条件出现异常情况下，不会轻易出现死机和灾难性的后果，而应有正确合理的表现；而且在异常情况下，出现的用于帮助解决异常的提示信息也应该尽量简明直观。这一要求，也许是应用系统取得成功的关键要素。本文主要讨论在Delphi环境中基于dbExpress与ClientDataSet的应用体系结构中，各种异常的处理方法。

2 dbExpress 及 ClientDataSet 应用体系结构

dbExpress是Borland公司最新主张的数据访问引擎，BDE数据访问引擎现在已进入维护阶段，不会再有新版本出台。dbExpress是专门针对大型关系型数据库系统设计的轻型数据访问引擎，其本身的功能主要用于获取单向数据集，把真正的数据维护功能交给客户端进行处理。而ClientDataSet作为Borland公司成熟的多层分布式应用技术中的重要组件，从前面的讨论可知，它具有相当强大的客户端处理能力。因此，将它们结合起来，可以实现功能强大、部署简单的C/S数据库应用，而且可以方便地向多层分布式应用结构移植。其体系结构如图1所示。

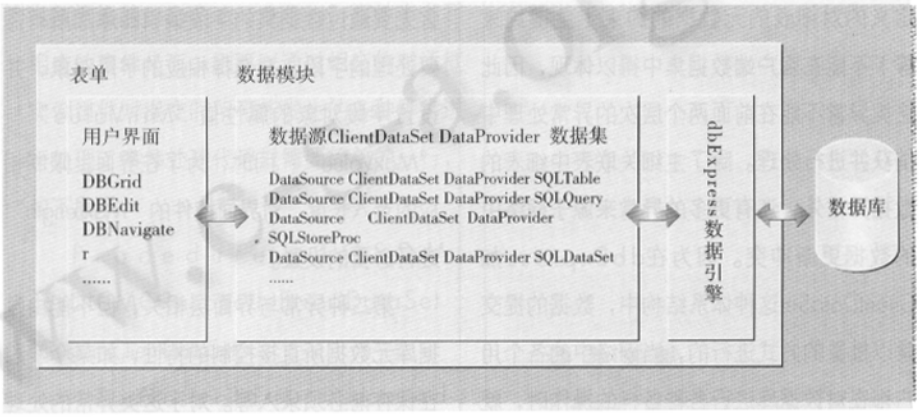

图 1 基于 dbExpress 与 ClientDataSet 技术的数据库应用的体系结构

3 异常处理解决方案

从上面给出的体系结构可知，出现异常可以是以下三种情形，在这里我称它们为三个层次的异常：

3.1 界面层的异常

在数据库应用中，一般界面层的许多组件通过数据感知组件直接绑定数据集中的字段，而依据应用的实际要求，数据表的字段需要设置适当的大小、类型、缺省值及是否为必须录入等。当界面层录入的项目不能满足这些要求时，这会引起异常。这类异常通常可在保存到客户端内存数据集（ClientDataSet）之前进行处理，因此，本文称这类异常为“beforepost异常”。

3.2 界面层到保存到客户端内存数据集层的异常

在将界面层的通过异常处理的数据保存到内存数据集时，可能出现：

（1）主键冲突，插入某些不能重复数据；

（2）主细关联表中，细表存在记录时，删除主表记录；

（3）主细关联表中，在细表中插入主表中不存在对应记录的数据。

这些异常都发生在将界面层数据保存到内存数据集时，因此本文称这类异常为“post异常”。

3.3 内存数据集层到数据库层的异常

在处理好前面讨论的两个层次的异常外，还可能有一些异常发生在客户端的内存数据集保存中到中心数据库时发生。例如，针对主细关联表中的细表，因为在dbExpress与ClientDataSet体系结构中，细表仅表现为主表的客户端数据集（ClientDataSet）的一个数据字段（DataField），在中心数据库中定义的对细表的元数据的约束（主键约束等）不能在客户端数据集中得以体现，因此这类异常不能在前面两个层次的异常处理中捕获并进行处理。除了主细关联表中细表的数据异常外，还有更多的异常来源于网络中的数据更新冲突。因为在dbExpress加ClientDataSet这种体系结构中，数据的提交是以批量的方式进行的，当网络中的各个用户都在对数据库进行各种各样的操作时，就可能引发冲突而产生异常。如在一个用户从数据库中获取数据并进行修改时，网络中的其他用户在此期间也对其中一部分信息进行了修改并提交到了中心数据库，那么当第一个用户修改完后提交数据时，就可能会引发冲突。因为这类异常只有在客户端内存数据集提交到中心数据库服务器时才能捕获，而客户端数据集提交到中心数据库的方法为“ApplyUpdates”，因此本文称这类异常为“ApplyUpdates异常”。

在给出这三个层次的异常后，下面将讨论三个层次的异常处理方法。

3.4 三个层次的异常处理方法

3.4.1 beforepost异常

在这个层次的异常中，又有两种类型。

第一种是当输入焦点离开所对应的数据录入界面控件时就触发的异常，如字段中录入值的范围超过了数据库中定义的字段值、字段中录入了非法的内容（如数值字段中录入了“-”等）。这类异常是由Delphi依据数据库中元数据的定义所控制，它们能由Delphi自动捕获，但是相应的提示信息都是英文，因此，不方便国人使用。对这类异常处理的关键是将这些提示信息本地化，本地化的具体方法是将随Delphi提供的VCL源代码（Source）中“**consts.pas”文件中相应的信息进行翻译。针对数据库错误的处理，仅需要处理“dbconsts.pas”。将修改后的文件复制到“Lib”目录，并删除相应的“.dcu”文件。另外，对于有输入范围控制的字段，首先要通过数据集的字段编辑器添加所有需要处理的字段，并选择相应的字段对象，并设置字段对象的属性值“MinValue”与“MaxValue”；同时，为了在界面上限制相应的录入长度，对界面控件的“MaxLength”进行必要的设置。

第二种异常与界面层相关，但不能由数据库元数据所直接控制的特性，如某个字段在保存前必须录入等。对于这类异常的处理可以采用两种方法，第一种方法是在相应的数据集（在大型应用中，数据集一般放置在单独的数据模块中）的“beforepost”事件程序中书写相应的代码，对相应的异常进行处理。但是这种处理方法在大型应用的开发中，有一个明细的弊端：为了在数据模块中的“beforepost”事件中，对异常进行很好的处理，可能需要经常性地引用相应的界面层的组件，这样，应形成了界面层与数据模块界的相互引用，很不便于在团队中组织开发。第二种处理方法是界面层的所有可能引发数据集保存“post”事件的代码中添加相应的条件检查，当不满足条件时，给出相应的中文提示信息，并将焦点设置到相应位置。例如下面的代码片断：

```
//数据BeforePost错误处理
If dbedtDicID.Text = '' Then
Begin
  ShowMessage('字典的ID值不能为空！');
  Exit;
End;
If dbedtDicValue.Text = '' Then
Begin
  ShowMessage('字典值不能为空！');
  Exit;
End;
```

采用这种异常处理模式，录入数据的检查完全由界面层完成，能避免界面层与数据层的交叉引用。

3.4.2 post异常

由前面的定义可知，“post异常”是在“beforepost异常”处理后，将界面层数据保存到到客户端内存数据集时可能引起的异常，虽然这类异常与“beforepost异常”有某些重叠，但这类异常的一个明显特征是：就数据集中的单一记录而言，本身并没有什么不正确之处，但是当把数据库看成一个具有一定约束关系的整体时，从界面层录入的单一记录可能会与当前数据集中的其他记录矛盾（如主键冲突），或者与关联表中约束冲突（如主细关联表中，细表存在相应记录时，删除主表记录；主表相关记录不存在时，录入细表记录等）。

对于这类异常的处理也有两种不同的方法：第一种处理方法是在数据集的“OnDeleteError”、“OnPostError”等事件中书写相应的事件代码，因为这是标准的处理

方法，在此不再赘述。第二种处理方法是借助Delphi提供的异常处理机制，并设计相应的通过异常处理函数来实现。下面是相应的异常处理代码。为了保持相应的完整性，重复了前面的代码片断，并在进行异常处理前，对数据集的状态进行了检查，对“post异常”采用了“try...except...”异常处理机机制，并调用了相应的异常处理过程“ShowErrorMessage(E : Exception)”。

```
If (GSDM.cdsDICMG.State = dsEdit) Or (GSDM.cdsDICMG.State = dsInsert) Then
  Begin
   //数据Post错误处理
   If dbedtDicID.Text = '' Then
   Begin
     ShowMessage('字典的ID值不能为空！');
     Exit;
   End;
   If dbedtDicValue.Text = '' Then
   Begin
     ShowMessage('字典值不能为空！');
     Exit;
   End;
   Try
    GSDM.cdsDICMG.Post;
   Except
     On E : Exception Do
      GSDM.ShowErrorMessage(E);//调用数据模块中通用的异常处理函数
   End;
  End;

Procedure TGSDM.ShowErrorMessage(E : Exception); //通用“post异常”的处理过程
  Begin
   If E Is EDBClient Then
   Begin
    Case (E as EDBClient).Errorcode Of
      9729 : ShowMessage('数据保存错误：主键重复！');
      9732 : ShowMessage('数据保存错误：必录字段没有录入！');
      9733 : ShowMessage('数据保存错误：主细表的主表中相关记录不存在！');
      9734 : ShowMessage('数据保存错误：主细表的细表存在记录，不能删除！');
     End;
     Exit;
   End;
   ShowMessage('数据库保存错误:' + E.Message);
   Exit;
  End;
```

3.4.3 ApplyUpdates异常

“ApplyUpdate异常”是在将客户端数据集提交到中心数据库时可能发生的异常。为了简化这类异常的处理，Delphi给出一个通用的对话框组件“Reconcile Error Dialog”，程序员只需要在项目中加入一个新的实例，并在数据集的“OnReconcileError”事件中书写相应的事件代码，则可以通过相应的对话框实例组件对提交时所可能发生的各种异常进行处理。事件处理代码只需要简单的一行，如下所示：

```
Procedure TGSDM.cdsDICMGReconcileError(DataSet : TCustomClientDataSet;
  E : EReconcileError; UpdateKind : TUpdateKind;
  Var Action : TReconcileAction);
Begin
  Action := HandleReconcileError(DataSet, UpdateKind, E);
End;
```

而在相应的程序处理流程中，可以通过获得“ApplyUpdates(0)”或“ApplyUpdates(-1)”的结果来判断是否发生“ApplyUpdate异常”，从而控制相应的程序流程。如在窗体的“OnCloseQuery”使用下面的代码来决定窗体是否真的关闭。

```
If (ChangeCount > 0) Then
  If (ApplyUpdates(0) <> 0) Then
  Begin
    CanClose := False;
    Exit;
  End;
```

4 结论

本文讨论了基于dbExpress与ClientData组件的数据库应用的体系结构，定义了该体系结构的三个层次的异常，并给出了相应的异常处理解决方案。所使用的方法已应用于多个大型管理信息系统的开发中，规范了相应的异常处理方法，大大提供了相应系统的可靠性与可用性。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 王涛，多层分布式数据库实践，清华大学出版社，2000年。

4 李维，Delphi 7 高效数据库程序设计，机械工业出版社，2003年。