基于 Jet 引擎的文件数据合并

Data Merging based on Jet Engine

胡修林　唐信忠　(武汉华中科技大学 430074)

摘要:介绍在 Delphi 中利用 Microsoft MDAC Jet 4.0 引擎实现 Excel 文件和文本文件两种不同格式的文件的数据存取的技术,并通过一个实例程序展示该技术在实际工程中的应用。

关键词:数据存取技术 Jet 引擎 Excel 文本文件 数据合并 Delphi

1 引言

数据库的访问一直是计算机应用中的一个重点。随着技术的发展,各大软件厂商均提出了基于数据库独立性的数据访问解决方案。如早期的 Microsoft 的 ODBC 和 Borland 的 IDAPI(即 BDE)。随着 COM 技术的成功,Microsoft 开始使用 OLE DB 代替 ODBC。OLE DB 是一个系统级的接口,为了方便程序员,Microsoft 在其上提供了在 Win32 平台广泛应用的适合编程的 ADO 接口。所有 Microsoft 的这些数据访问技术,包括 ADO(以及.Net 平台的 ADO.Net)、OLE DB、ODBC 和 RDS,被统称为 MDAC。Jet OLE DB 属于 MDAC 的一部分,它主要提供对 MS Access 以及其它一些基于文件的桌面型数据库的直接访问支持。通过 Jet 引擎,除了能访问 MS Access 数据库而外,还可以访问基于文件的桌面型数据库甚至是单个文件,如:Paradox、dBase、Excel 文件、文本文件,Lotus1 -2 -3 或 HTML。本文会将重点放在 Excel 文件和文本文件上。通常编程访问 Excel 时使用的是基于 OLE 的方法,不过笔者准备从基于数据库应用开发的思路,利用 Jet 引擎对这两种文件进行数据访问与合并。

2 在 Delphi 中利用 Jet 引擎访问 Excel 文件和文本文件中的数据

下面在 Delphi7 中以 ADODataSet 组件为例介绍该技术。

2.1 利用 Jet 引擎访问 Excel 文件

打开 ADODataSet 组件的 ConnectionString 属性编辑器,点击 Build 按钮将弹出 Microsoft 的数据连接属性编辑器。在"提供程序"页中选中 Microsoft Jet 4.0 OLE DB Provider;点击下一步按钮,在"连接"页中的"选择或输入数据库名称"输入框中输入要访问的 Excel 文件的全路径(或相对路径)文件名;然后选中"所有"页,双击其中的"Extended Properties"项,在"属性值"输入框中输入 Excel 8.0(这是 Jet 4.0 引擎能处理的 Excel 的最高版本,对应的外部版本号为 Excel 97 - Excel 2000。具体可以查看注册表 HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Jet\4.0\ISAM Formats\Excel 8.0 键的详细内容),点击"确定"按钮关闭所有编辑窗口后,ConnectionString 属性便设置好了;接着将 ADODataSet 控件的 CommandType 设置为 cmdTableDirect;然后在 CommandText 属性的下拉框中选中需要访问的 Excel 工作表,至此配置已全部完成,接着便可以像利用 ADODataSet 访问数据库一样轻松访问 Excel 文件了。

2.2 利用 Jet 引擎访问文本文件

打开 ADODataSet 控件的 ConnectionString 属性编辑器,点击 Build 按钮将弹出 Microsoft 的数据连接属性编辑器。在"提供程序"页中选中 Microsoft Jet 4.0 OLE DB Provider;点击下一步按钮,在"连接"页中的"选择或输入数据库名称"输入框中输入要访问的文本文件所在的文件夹的全路径或相对路径(注意,不同于 Excel 文件,Jet 引擎将文本文件所在的文件夹当作数据库看待,所以这里填入的是文件夹的路径而非文件本身的路径);然后选中"所有"页,双击其中的"Extended Properties"项,在"属性值"输入框中输入 Text,点击确定关闭所有编辑窗口后,ConnectionString 属性便设置好了;接着将 ADODataSet 控件的 CommandType 设置

为 cmdTable；然后在 CommandText 属性的下拉框中选中需要访问的文本文件名称（注意文件名中的'.'已被'#'取代），至此配置已全部完成，接着便可以像利用 ADODataSet 访问数据库一样轻松访问文本文件了。

菜单项后，程序根据用户在打开对话框中选择的文件，首先将文件拷贝到程序所在目录下（拷贝文件的原因是因为利用 Jet 引擎访问 Excel 表是以独占方式打开的，拷贝文件可以避免程序运行期原来的 Excel 文件不

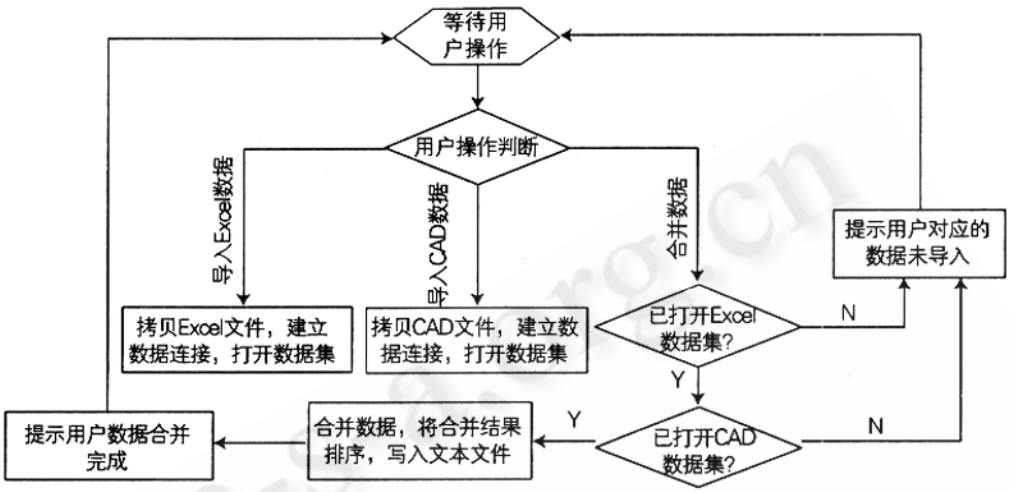

图 1

3　利用 Jet 引擎实现文件数据合并

这里讨论的实例程序需要完成下述功能：用户每次提供两个文件，一个 Excel 文件，一个 CAD 文件（因为其内部格式为 csv 的文本格式，所以可以将其当作文本文件处理）。Excel 文件包含 7 列：序号、名称、位号、规格型号、数量、物料编码和备注。CAD 文件包含 7 列：序号、规格型号、横坐标、纵坐标、角度、空（该列无数据）、位号，各列以逗号隔开。程序检索 Excel 文件所有"位号"列中的单元格数据，在 CAD 文件中找到"位号"相同的行记录，然后从 Excel 文件中取出"位号"和"规格型号"数据，从 CAD 文件中取"横坐标"、"纵坐标"和"角度"数据，将合并后的数据输出到新生成的文本文件中。其中 Excel 表格每个"位号"列单元格可能有多个数据，每个数据可能表示位号本身（如 C50），也有可能表示范围（如 C1 – C50 表示 50 个位号）。

3.1　程序设计

程序分别使用两个 ADODataSet 连接对应的 Excel 文件和文本文件（为便于管理，数据集组件和数据源组件被放置在一个单独的 DataModule 中）。当用户分别点击界面中的"导入 Excel 数据"和"导入 CAD 数据"的

能被其它程序再次打开的问题，另外的好处是整个过程中其它程序对数据的修改不影响当前数据），然后再打开两个 ADODataSet 与这两个文件的连接，读入数据到数据集中。当用户单击"合并数据"的菜单项后，程序首先判断两个数据集是否都已打开，若是，则遍历数据集，找出"位号"相同的项，将数据一一打印到文本文件中；若有数据集未打开，则对用户做相应的提示。程序主要部分的流程图如图 1 所示。

程序主要包括主窗体（命名为 FrmMain）和数据模块（命名为 DM），数据模块中有两个 ADODataSet 组件（分别命名为 ADOExcel 和 ADOCAD），通过两个 DataSource 组件与主窗体中的数据敏感控件关联。ADODataSet 组件与对应文件的连接采用设计时设置的方式。

ADOExcel 的设置为：

ConnectionString：Provider = Microsoft. Jet. OLEDB. 4. 0；Data Source = . \Excel. xls；

Extended Properties = Excel 8. 0；Persist Security Info = False

CommandType：cmdTableDirect

CommandText：Sheet1 $（笔者注：Excel 工作簿需要后缀一个 $）

ADOCAD 的设置为：

ConnectionString：Provider = Microsoft. Jet. OLEDB. 4.0;Data Source = .;

Extended Properties = Text;Persist Security Info = False

CommandType：cmdTable

CommandText：cad#txt（笔者注：'.' 被 '#' 所替换）

3.2 程序实现

导入数据（以导入 Excel 数据为例）：

```
if DlgExcel. Execute then//打开文件对话框。
begin
 DM. ADOExcel. Close;//必须先关闭才能导入新的数据。
 StatusBar. SimpleText := '正在导入 Excel 文件...';
 Screen. Cursor := crHourGlass;
 try
  CopyFile ( PChar ( DlgExcel. FileName ), PChar ( AppPath +
  'Excel. xls'), False);//拷贝 Excel 文件到程序目录。
  DM. ADOExcel. Open;//打开数据集。
  PageControl. ActivePage := TabExcel;//显示数据。
      finally
  StatusBar. SimpleText := '完毕';
  Screen. Cursor := crDefault;
 end;
end;
```

合并数据：

```
if DM. ADOExcel. State = dsInactive then
begin
 MessageBox( Self. Handle, '未导入 Excel 数据', 'Warning',
  MB_ICONWARNING);
 Exit;
end;
if DM. ADOCAD. State = dsInactive then
begin
 MessageBox( Self. Handle, '未导入 CAD 数据', 'Warning',
  MB_ICONWARNING);
 Exit;
end;
DM. ADOExcel. DisableControls;//避免较长时间的后台操作对界面造成影响
DM. ADOCAD. DisableControls;
StatusBar. SimpleText := '正在合并数据...';
Screen. Cursor := crHourGlass;
try
 SLResult. Clear;//清空 SLResult，避免多次合并结果的累积。
```

笔者注：SLResult 为一 TStringList 对象，用于存储合并后的导出数据。

```
 DM. ADOCAD. First;
 while not DM. ADOCAD. Eof do//遍历数据集。
 begin
  Application. ProcessMessages;//防止 GUI 响应缓慢。
  MergeData;//合并数据。
```

笔者注：MergeData 是笔者自定义的实现数据合并算法的主要例程。由于该算法稍显复杂（由于要考虑到 Excel 表格每个"位号"列单元格可能有多个数据，每个数据可能表示位号本身（如 C50），也有可能表示范围（如 C1 – C50 表示 50 个位号）。），因此该函数代码比较长，限于篇幅原因，这里没有给出。

```
  DM. ADOCAD. Next;
 end;
 SLResult. Sort;//对结果进行排序。
 WriteFile;//写文件。
```

笔者注：WriteFile 同样是笔者自定义的写文件的例程。限于篇幅原因，这里没有给出具体代码。

```
finally
 DM. ADOExcel. EnableControls;//恢复界面数据显示。
 DM. ADOCAD. EnableControls;
 StatusBar. SimpleText := '完毕';
 Screen. Cursor := crDefault;
end;
```

程序运行界面和合并结果的数据示例见图 2、图 3 和图 4。

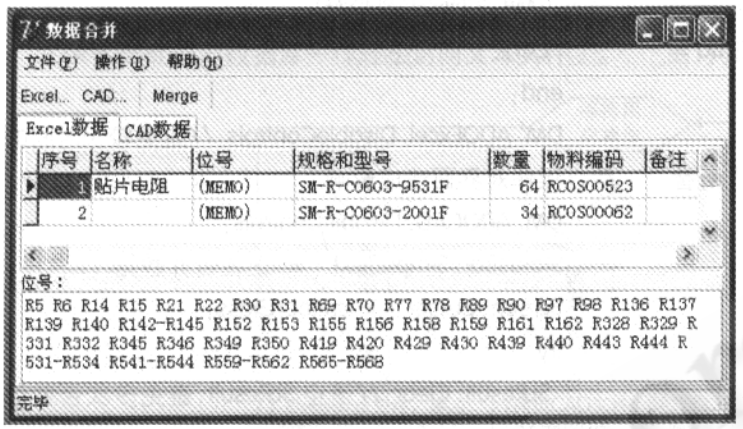

图 2

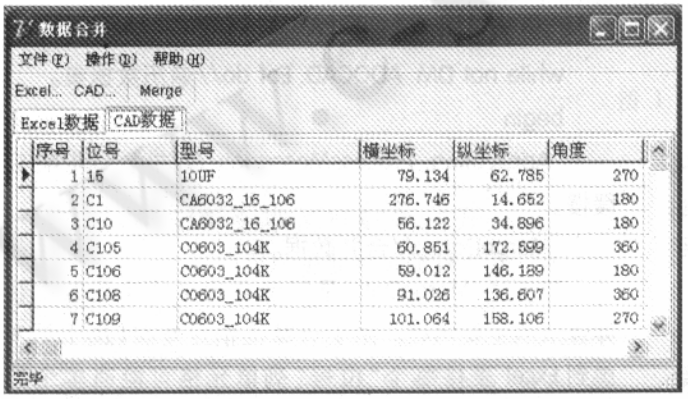

图 3

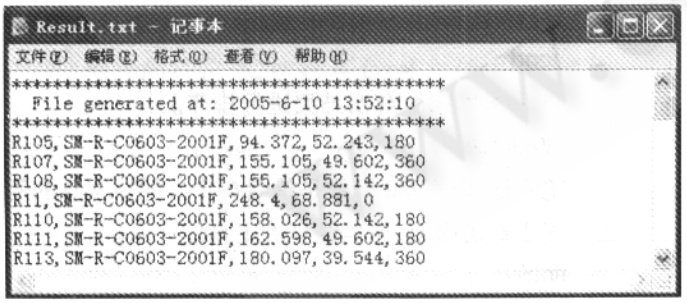

图 4

4 结束语

利用 Jet 引擎可以很方便的访问 Excel 文件和文本文件，而且使用该技术最大的好处就是运行该程序的电脑上可以不必安装 Excel，这也是该技术与基于 OLE 的 Excel 文件访问技术的一个不同之处。不过使用该技术也是有限制的。首先，Jet 引擎对 Excel 文件是以独占方式打开的；其次，使用该引擎操作 Excel 时可以添加新的行并编辑现有的行，但不允许删除行。而在使用该引擎操作文本文件时也有两个限制：不能删除行和编辑行。尽管该引擎有一些限制，但通过前面的工程实例可以发现，在类似的工作中，通过 Jet 引擎程序员便可以用访问数据库的方式来思考，既可以利用数据集组件的强大功能又简化了工作，这样便可以将更多的精力投入到算法的研究中去。

参考文献

1 Marco Cantu. Mastering Delphi7，电子工业出版社，2003－10。
2 李维，Delphi 5.X ADO/MTS/COM＋高级程序设计篇，机械工业出版社，2000。
3 Delphi7 Online Help.