工控组态软件中图形组态子系统的设计与实现

吴修国　贾智平　山东大学计算机学院(250061)

Abstract

Graph configuration subsystem is an important part in industrial configuration software. On the basis of introducing the design and implementation of the graph configuration subsystem using the technology of object-oriented, this paper gives the way to implement the procedure of zoom, rotation and save etc.

Keywords: configuration software, object-oriented, delphi

摘　要

图形组态子系统是工控组态软件的重要组成部分。本文在介绍了采用面向对象技术进行图形组态子系统的设计与实现的基础上，阐述了在设计图形组态子系统过程中图形缩放、旋转、保存等的实现方法。

关键词：组态软件，面向对象，Delphi

1　引言

图形组态子系统是任何一个工控组态软件都应该具备的图形处理系统，图形组态在整个工控组态软件系统中占有非常重要的地位。它是设计人机交互界面的主要工具，可以通过一些背景、跟踪模块等提供丰富逼真的图文形式，工业控制中的各种显示仪表，控制表盘，回路调节图等都可以通过它来实现。如果图形组态子系统设计的优良，对于用户来说，可以非常方便地模拟现场的生产状况，使得组态软件的开发具有操作简单，显示画面丰富等优点。图 1 是我们开发的工控组态软件的图形组态子系统的设计界面。

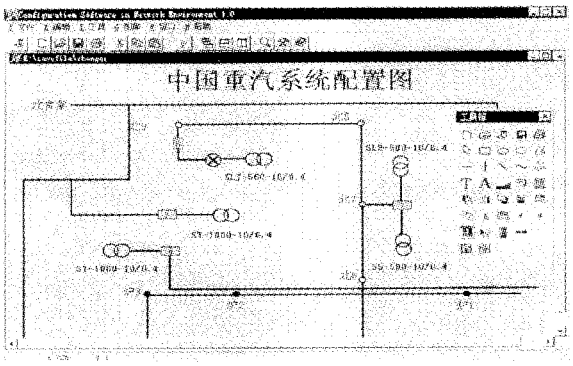

图 1　工控组态软件图形组态子系统设计界面

2　面向对象的图形组态子系统的设计

概括地说，面向对象方法的基本思想是，从现实世界中客观存在的事物（即对象）出发来构造软件系统，并在系统构造中尽可能运用人类的自然思维方式。与传统的方法不同的是，面向对象方法在系统分析与设计阶段没有明显的区别，并对系统的描述能力十分强大，开发者在建模阶段花费的时间也是最多的。面向对象的编程的主要特点是类、继承、多态。运用 Delphi 开发应用程序的过程实际上是一个设计相应的类生成所需要的对象，然后通过消息机制把相关的对象联系起来的过程。

类是一些享有共同特征的对象的集合。设计类的第一步是确定类层次结构。每一个层次结构从一般的简单类开始以支持非常特殊操作的子孙类结束。下一层类与上一层类的关系是继承关系。需要指出的是，Delphi 只支持线性继承，即每个子类只有一个父类。

在图形组态过程中，需要定义一系列的图形对象，包括管道、电器符号、反应罐等。这些对象可以分为两类：一类是静态对象，它们的数据、颜色等不需要实时改变，这类对象包括文本、几何图形等；另一类是动态对象，需要根据实时数据，动态改变它们的显示形状、颜色等属性，这类对象包括数码管显示、流动跟踪等。

在设计过程中，我们设计了一个基类 TPic ，定义各种图形对象的基本属性与方法，如位置、大小、

颜色、设备号等属性以及刷新、移动、保存到文件、从文件中读取等方法。

```
Unit UserClass ()        //用户类库单元
Interface
Type  TPic = Class          //图形类
  Private
    x1,y1,x2,y2:Integer; //定义坐标
    AnsiString  DevNo;        //设备号
    ColorREF  FColor β };     //颜色
    . . .
  Public
    Load(TFilestream);    //从文件中加载
    Save(TFilestream);    //保存到文件
    . . .
  End;
```

它派生出两个子类：TStaticPic、TDynamicPic，分别定义为静态类与动态类。静态对象的类由TStaticPic 派生，动态对象的类由 TDynamicPic 派生，在各类中又定义了各自的属性、方法。类的层次结构如图 2 所示。

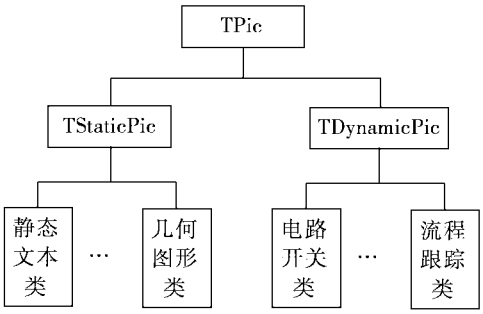

图 2　图形类层次结构图

3　图形组态的实现方法

3.1 图形缩放的实现

在图形组态的过程中，为了更好地处理图形，应当使图形具有缩放功能。这样既能观看图形组态的总体结构，又能观看某些细节。可以有两种方法实现图形的缩放。

＊ 定义全局变量存放缩放比例。当窗口刷新时，根据各个对象的原始坐标、大小与缩放比例相乘，就得到各个对象在新的缩放比例下的坐标、大小。但是这种方法执行速度慢、操作复杂。

＊ 利用 Windows 提供的 API 函数，实现图形地缩放。当在 Windows 下输出文本或图形时，使用的 GDI 函数需要坐标值或大小作为参数。大部分的 GDI 函数中这些坐标值或大小是“逻辑单位”。Windows 必须将逻辑单位转换为“设备单位”，即象素。这种转换是由映射方式、窗口、视口的原点以及窗口和视口范围控制的。映射方式还隐含给出了 X 轴和 Y 轴的方向。视口是基于“设备坐标”的。通常，视口和客户区相同，但是如果已经用 GetWindowDc 或 CreateWindowDC 获取了一个设备环境，则视口也可以指整个窗口坐标或屏幕坐标。“窗口”是基于逻辑坐标的。逻辑坐标可以是象素、毫米、英寸或者任何其它形式。GDI 函数使用逻辑窗口坐标。

在所有的映射模式中，Windows 都用下面两个公式来将窗口(逻辑)坐标转化为视口(设备)坐标：

其中，(xWindow, yWindow)是待转换的逻辑点，(xViewPort, yViewPort) 是转换后的设备坐标点。(xWinOrg, yWinOrg) 是逻辑坐标的窗口原点，(xViewOrg, yViewOrg) 是设备坐标的视口原点，(xWinExt, yWinExt) 是逻辑坐标的窗口范围，(xViewExt, yViewExt)是设备坐标的视口范围。

当窗口需要刷新时，根据当前缩放比例，设置映射方式和视口、窗口坐标。然后调用各个对象的显示函数。在对象的显示函数中，不再需要考虑显示比例。例如，如果要设置缩放比例为 2(放大 2 倍)，可以如下编写程序。

```
SetMapMode (dc, MM_ISOTROPIC )
SetWinodwExtEx (dc, 1, 1, NIL )
SetWiePortExtEx (dc, 2, 2, NIL )
```

3.2 图形旋转的实现

在图形组态过程中，为了更好地模拟现场控制，有时需要将将图形进行旋转。对于图形的旋转算法，一般的实现方法是对图像的每一像素点依次作平移、旋转、再平移变换。下面是图片旋转的一个简单的算法。

(1) 顺时针旋转 90 度

```
DestinationImg. Picture. Bitmap. Height: = Sour-
ceImg. Picture. Width;
DestinationImg. Picture. Bitmap. Width: = Sour-
ceImg. Picture. Height;
For  i: = 0  To  SourceImg. Height  Do
     For  j: = 0  To  SourceImg. Width  Do
DestinamtionImg. Canvas. Pixels [(- i + Sour-
ceImg. Height )j ] =
          SourceImg. Canvas. Pixels [i ];
```

(2) 水平旋转 180 度

```
DestinationImg.Picture.Bitmap.Height:= SourceImg.Picture.Height;
DestinationImg.Picture.Bitmap.Width:= SourceImg.Picture.Width;
For i:=0 To SourceImg.Height Do
    For j:=0 To SourceImg.Width Do
        DestinationImg.Canvas.Pixels[(SourceImg.Width-j),(SourceImg.Height-i)]:=SourceImg.canvas.Pixels[j,i];
```

(3) 顺时针旋转 270 度

代码和步骤(2)相似,只要用以下的语句替换步骤 3 中 For 循环中的原有的语句就可以了。

```
DestinationImg .Canvas.Pixels[(SourceImg.Width-j),]:=
    SourceImg.Canvas.Pixels[j,i];
```

3.3 图形保存的实现

在图形设计完成后,如何对图形进行保存呢? 我们通过分析 Delphi 对文件的保存格式, 采用对每一个对象进行保存的方法。具体地实现方法是:首先保存对象的类名,然后保存类的属性,比如对象的 Left, Top, Width, Height 等。当打开文件进行加载的时候,首先要根据类名进行创建新的对象,然后根据保存在文件中的属性来设置对象的属性。例如,我们在保存一个静态文本的时候, 首先要保存静态文本的类名,然后将它的属性保存下来。下面是 TvreLabel 的保存过程。

```
Procedure TVRELabel.save(var F:Textfile);
Begin
  Writeln(F,classname);     // Classname 表示类名称
  Writeln(F,ffont.color);   // 保存静态文本的字体颜色
  Writeln(F,ffont.size);    // 保存静态文本的字体宽度
  …
End;
```

同样地, 在文件加载的过程中, 我们要根据读取的值设定对象的属性。

```
Procedure TVRELabel.Load(var F:Textfile);
  …
Begin
  …
Readln(F,fcolor);
Ffont.Color:=Strtoint(color); //设置静态文本的字体颜色
  …
End;
```

通过这种方法就可以正确的将原来创建的对象重新加载到当前的文件中。

4 结束语

图形组态子系统是工控组态软件中必不可少的组成部分,它为模拟工业现场生产,丰富系统界面起了重要的作用,所以对它的可靠性要求特别高。由于本系统采用了面向对象的处理方法, 此外 Delphi 5.0 开发环境提供了强大的自定义组件的支持,因此程序从开发,调试到程序运行,效果良好。当然,本系统还有待完善的地方,比如说优化图形的旋转算法等等。本文旨在为工业组态软件中图形组态子系统的设计提供一种思路,为组态软件的开发做一些铺垫。

参考文献

1. 邵维忠,杨芙清编,面向对象的系统分析,清华大学出版社,广西科学技术出版社,2000.1
2. 黄伟,叶朝晖,面向对象的工控组态软件的开发与应用,工业控制计算机,1996.4
3. 伊宏卫,工业控制组态软件的设计,电子技术应用,1996.2

(上接第 30 页)

操作人员在任何时刻按一下打印键,软件生成出此时刻全部被测量的实际值和系统运行的状态表单,打印机立即打印。

(7)事故预报、故障诊断

对生产过程进行持续监视并对不正常环节报警,进入故障诊断系统,采取相应措施。

(8)控制功能

实现检测过程的控制和输入、输出板卡的控制。

(9)操作的安全性

设计多级安全控制和访问权限,操作人员必须登录自己的身份才能获得一定的操作权。当操作者的权限小于软件运行的访问权限,操作者启动不了系统软件。

6 结束语

本测试系统经多次运行,性能可靠,操作简单,测量数据真实地反映了被测对象的实际性能指标,达到了预期的功能要求, 是飞机发电机检修时的理想测试设备。

参考文献

1. 徐用懋,颜纶亮,微机在过程控制中的应用,清华大学出版社,1990
2. 组态王 5.0 用户手册,北京亚控自动化软件科技有限公司