面向对象技术与可视化开发环境

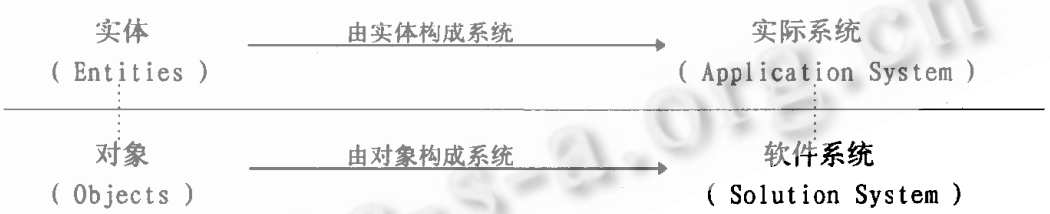
俞集辉　贾代平　（重庆大学电气工程系　400044）

摘要：本文首先阐述了面向对象的思想，并充分说明了这种思想产生的根源和它对软件开发过程的影响，接着就可视化开发和面向对象技术在 Delphi 集成环境中的应用作了比较详细的解说。

关键词：对象　类　封装　继承　面向对象　可视化　Delphi

一、“面向对象”的思想

与传统的结构化思想相比，面向对象是一种全新的设计思想。下面试图在两者的比较过程中阐述它们的区别与优劣。结构化方法在设计时是根据用户的要求从应用系统的功能着手，首先把应用系统按功能分为不同的子系统，再把各个子系统分成许多相对独立的模块，最后把模块分成一系列的函数和过程。这是一种自上而下、由大到小的设计方法。这种设计方法虽然曾经给软件设计带来过一定的生机，但随着软件规模的扩大、软件复杂性的迅速增加，越来越显示出它固有的弊端，究其原因是这种以“功能”为中心的软件设计方式有背于人们认识真实世界的思维方式。当我们要深入分析一个实际系统的时候，第一步要做的就是着眼于组成这个系统的客观实体，在认清了这个系统的各个实体后，我们才有可能从整体上把握系统并认识在这个系统中所发生的变化。这是一个自上而下、由小到大的过程。软件设计是一项非常严谨的工作，任何只从宏观上把握而不从微观上深入考察的做法都不会创造出符合实际的软件。事实上，任何一个软件系统都是实际问题（系统）在计算机中的表述形式。显然，如果软件系统的设计遵循人们认识实际问题的思维方式，则软件不仅易于被人们理解，而且易于维护和修改，从而提高软件的可靠性和可维护性。“面向对象”的思想正是基于上述相法而诞生的，即按人们通常的思维方式建立实际系统的计算机模型，设计出尽可能自然地表现求解问题的方法。下图反映了实际系统和解决问题的软件系统两者之间的对应关系：



面向对象技术在解决实际问题时是从一个个具体的实体着手的。首先要寻找存在于问题系统的实体，然后去研究每个实体的属性、特征和功能，这些实体在设计的模型中被抽象为对象类，而在最终的应用软件系统中这些实体被转化为确定的对象（类实例）。面向对象的软件设计正是从分析实体开始，继而确定各个对象类的属性，定义每个对象类的功能，建立对象类之间的层次结构，最终形成完整的软件系统。分析问题中的实体总是很具体的，与此相对照，结构化方法则把注意力集中在对问题的解决方案上，这就导致了面向对象技术在解决问题的一开始就和结构化方法完全相反。面向对象是从小到大，结构化是从大到小；面向对象是自下而上，结构化是自上而下。由此我们不难看出面向对象更接近问题的本来面目、更符合人们对它的认识发展过程。应用面向对象技术在设计中从实际系统到软件系统具有极强的对应性，因而在我们和客户讨论需求时，关心的是存在于问题系统中的各个实体以及每个实体的功能、特征及属性。这样即使我们的客户对计算机知识一无所知，也能很容易与之沟通。不仅如此，在面向对象的软件设计中，我们把客观实体转化为软件对象，把必要的数据及其操作封装

© 中国科学院软件研究所 http://www.c-s-a.org.cn

到对象里去，由对象构成系统，这样的应用软件具有易改进、易维护、便于扩充等突出优点。结构化方法具有两个明显的缺点：一是对于一个复杂的问题难于进行分解，二是不同的设计人员对问题有着不同的分解方案。在对问题系统的细节不甚了解的情况下，这种分解难免存在或多或少的弊端，随着开发的进行软件向问题的底层深入，一旦在设计的后期发现这种弊端，此时已难于对分解方案作出更改，也就是说结构化的设计思想和方法没有一个很好的规范。运用面向对象技术设计的软件系统和实际问题有着极强的对应关系，结构化方法的不规范在此得到了很好的限制，越到开发的底层，这种限制越明显，因为软件的基本构件——对象是从客观存在的实体中产生的。

把面向对象的思想赋之于编程实践，即是我们所说的面向对象编程(即 OOP)，以“对象”为中心来构建系统正是面向对象技术的实质所在，它比传统的以“功能”为中心的结构化方法显得更为自然，也更易于被人们所理解，因为它更接近于真实的客观世界。面向对象的程序设计就是试图以拟人的思维方式去理解现实世界对象的实质，运用这种方式进行计算机的程序设计。在面向对象的软件开发方法中，软件系统是由对象组成的，而对象又是现实世界中实体的抽象。面向对象的程序设计的对象、类以及对象类的封装和继承对产生可重用软件的设计与实现具有极大的优越性，面向对象的程序设计方法使软件开发人员摆脱了具体的数据格式与过程，集中精力去研究所处理的对象。

总之在软件领域，面向对象技术不仅仅是一种方法，更是一种思想，应用的好坏在很大程度上取决于我们对这种思想的理解和掌握的程度。如果说 80 年代是结构化程序设计的年代，那么 90 年代则是面向对象程序设计年代。可以预言未来的软件系统是建立在“对象”概念之上，面向对象技术将成为 90 年代软件开发的主流技术。

二、可视化开发环境中的面向对象技术

1. Delphi 的编程环境

为了更好地解说面向对象技术在 Delphi 中的应用，先简要介绍一下开发环境是必要的。启动 Delphi 后屏幕显示主要有四部分组成：菜单条、组件板、对象观察器和一个空白窗口。菜单条为开发者提供了一组功能完善的菜单项，为编写代码、调试以及保存应用程序服务；组件板上存放了用以编制 Windows 程序所需的常用元件如按纽、编辑框等，为可视化开发提供方便，它是面向对象技术在 Delphi 中的直观体现；默认的空白窗口有两页，上层是一个标准的 Windows 窗口，它象我们作画时的一张白纸，供我们在设计过程中去加工描绘；下层是一个代码编辑窗口，我们在编程过程中的任何代码都放在这里；对象观察器是开发过程中一个方便的工具，软件系统中任何可见的构件(对象)都可在这里观察、设置其属性，同时它又是开发者为构件编写响应事件的入口。一个基本的编程过程可分为两个阶段：可视化设计阶段和代码设计阶段。可视化设计阶段主要任务是设计应用软件的界面，可将 Delphi 中内建的组件(对象)直接放置到窗口中，因此这一阶段实际上是设置对象属性和布置窗口；接下来就是代码设计阶段，其主要任务是为对象编写需要响应的事件代码，这部分代码就是应用程序响应用户操作的执行代码。用传统的开发技术开发 Windows 应用软件要比开发 DOS 应用软件困难得多，即使是编制一个简单的对话框也要耗费大量的代码，面向对象技术很快解决了这类问题，它把视窗软件常用的组件诸如按钮、文本框，甚至是窗口本身封装为通用的对象，供开发者调用。可视化开发则更进一步，它把常用的对象具体化、可视化，于是对象不再是初学者难以琢磨的概念，而是呈现在屏幕上实实在在的实体。Delphi 开发环境中的组件板上就存放着大量的这样的实体。

2. 对象的属性、方法和事件

在 Delphi 中，任何一个对象(类)具有下列三个特性，即属性(Properties)、方法(Methods)和可以响应的事件(Events)，这些特性是一个对象与外界交换信息、相互连接的标准接口。下面分别作一简要叙述。

任何一个客观实体都具有属于它本身的一些特性，如一台电视机，有它的规格、品牌、尺寸、性能指标等，对象的属性与此类似，它描述对象外在的或内在的一些特征，改变对象的属性可以改变对象的外观以及对象的一些非可视因素，如对象的行为等。本质上对象的属性可以看作被封装程序中的局部变量，并被用于其内部处理。与局部变量不同的是，对象的属性可以被外部查看和利用，因而对象的属性可以被重新设置或读取，它成为对象内部和外部的接口。

方法是指处理对象类或类实例的函数和过程的总称，任何一个对象(类)内在地拥有属于它自己的一些函数和过程，并能够被它所调用。方法反映了一个对象所拥有的“能力”，描述这些函数和过程的代码则被封装于对象内部，根据需要对外提供一些必要的参数，调用这些方法是发挥对象功能的基本手段。要使应用软件中的对象去执行某一特定的任务，最常用的方法就是在程序中

© 中国科学院软件研究所 http://www.c-s-a.org.cn

灵活运用 Delphi 为每个对象(类)所内建的方法。

事件是软件开发中一个比较重要的概念,事件驱动编程的思想早已有之,但直到图形用户界面出现后,才得到人们的足够重视。事件可以看作用户或系统所发生的任何操作行为,如用户单击鼠标、一个窗口关闭等。Windows 应用程序普遍采用事件驱动方法管理程序和用户间的交互行为,事件能够触发相应程序段的执行,即程序能够对不同的事件作出不同的反应,这种反应则需要开发者为之编写代码。事件与代码之间通过事件句柄相联系,由事件触发的过程称之为事件句柄。每一类的对象都有响应特定事件的能力,可以这么说,应用程序的执行过程就是系统不断地检测事件、执行事件代码的过程。

由此可见,要想在 Delphi 中游刃有余,必须熟练掌握对象的属性、方法,并深刻理解事件驱动的内涵,这也是 OOP 的特征所在。

3.无处不在的对象

在一个彻底的面向对象的开发环境中,对象是构建一个应用软件的基石。打个比喻,想一想我们如何组装一台计算机?首先要从市场上购买组件如主板、电源、硬盘等,然后把它们组装起来。组装即把这些现成的组件连接起来,只要连接正确,一台计算机(硬件部分)就完成了。我们为什么能够这样?就是因为我们购买了"封装好的功能",买来一台电源,它就已经具备了我们需要的性能,如提供稳定的电压、输出一定的功率等。面向对象的软件开发与此类似,我们用"封装好的功能"——对象来构建应用程序,因此我们始终要与各种各样的对象打交道,对象在整个开发过程中无处不在。如果我们抛弃对象,一切从头开始,那无异于生产计算机要从生产每个元器件开始。果真这样的话,当今世界上任何一个厂家也很难生产出一台完整的计算机。

请看我们在 Delphi 中是怎样做的,其基本步骤是这样的:首先在组件板上选取适当的组件(对象)置于窗口中,然后通过对象观察器调整它们的属性以满足我们的要求,这相当于我们买回来一台电源要把它的输出电压等参数调整到适当位置相仿,接着就要确定对象需要响应的事件,并为事件编写执行代码,这实际上为对象发挥必要的功能以及建立不同对象之间的正确连接关系。从中可以明显地看出这其中的每一步我们都在与对象打交道。一般来说可以直接使用 Delphi 中内建的对象,不仅如此,也可以根据需要构造具有特定功能的对象。当然构造新的对象类也没有必要完全从头开始,我们可以在已有对象的基础上通过必要的修改来构造满足我们需要的特殊对象,这就是利用对象的继承关系。在 Delphi 中有一个较庞大的继承体系,从 TObject(基类)开始,逐级延伸、变异、细化,最终呈现在我们面前一系列实用的具体的对象类。

值得注意的是并不是所有的对象都是可见的,在 Delphi 内建的对象体系中存在着大量不可见的、抽象的对象,有的虽然在设计期间可见,在运行期间却是不可见的,这些不可见的对象同样是我们构建应用软件所不可缺少的组成部分。例如当需要在屏幕上绘图时,我们就需要一张无形的画布 Canvas,即画布对象,如何绘图呢?有不可见的对象类 TBrush 和 TPen,即刷子和画笔。因此不可见的对象与可见的对象一样,为程序开发提供了极大的便利。对象实质上就是封装好的功能。那些可见的对象我们比较好理解,事实上任何特定的功能都可以将其封装为对象。例如在数学运算中积分是比较常见的运算,我们可以预先设计封装一个积分对象类,它对外提供一些标准的接口比如积分的上下限、被积函数、积分类型、积分结果等。在面向对象的软件设计中我们不仅要熟练掌握可见对象的应用,灵活设计和运用那些不可见对象也是我们从事面向对象的软件开发必须具备的一项基本功。

4.对象的可视化与可视化编程

Delphi 是一个优秀的可视化开发环境,它将面向对象技术与可视化开发技术结合于 Object Pascal 的编程语言中,代表着软件开发的最新水平。随着面向对象技术的发展,各种基于面向对象的软件开发技术逐渐成熟,其中可视化开发也是在此基础上发展而来的一项新技术。可视化开发环境与 GUI 有着不可分割的联系,Delphi 将视窗软件中典型的界面和功能分别进行细化分解,并根据现实世界对软件系统的一般需求,有针对性的将分解后的界面单元和功能单元相结合,构成一系列相对独立的软件单元,然后将这些软件单元尽可能地用图形化的方式来表示。这样构成的图形化的软件单元具有一般对象的所有特征,因而称之为可视化的对象。

可视化的对象显得直观、形象、便于理解,它大大缩短了编程人员的学习周期,降低了软件开发人员的门槛,使得一般的非专业人员经过较短时期的训练就可以着手进行开发工作。Delphi 的可视化元件库(VCL)中就提供了大量可视化的对象类(俗称为组件),它能够基本满足一般视窗软件所需要的界面单元和功能单元。例如,VCL 中就提供了足够的用于开发数据库应用程序的组件,灵活运用这些已有的数据库组件可以高效地开发出完美的数据库应用程序,既方便又快捷,毫不逊色于专用的数据库开发工具。应用传统的编程技术开发数据库应

© 中国科学院软件研究所 http://www.c-s-a.org.cn

用软件往往需要熟悉数据库的专业人员才能胜任，对于非专业人员，那些复杂的数据库连接会使人一筹莫展，然而应用面向对象技术，Delphi 使这一切变得非常简单，它将数据库连接、显示过程中的不可见的中间环节封装为几个相对独立的可视化对象，借助于 Borland 数据库机(Borland Database Engine)可直观地完成这些复杂的幕后操作，如下图所示：

数据库中的数据 ←→ TTable ←→ TDataSource ←→ 数据感知对象

其中 TTable 和 TDataSource 封装了从数据库中提取数据的全过程，数据感知对象完成数据显示的任务，即使在设计阶段它也能将数据感应(Data - aware)出来。类似这样的可视化功能大大地方便了开发者。当然前面提到的 VCL 也完全是动态的、开放的体系，这是对象技术的核心思想，它在 Delphi 中得到充分的体现。

可视化编程环境为创建应用程序增加了一个新的内容，在程序执行以前，可以把对象画在屏幕上。没有可视化的环境时，“画”的过程需要写出实际的源代码，创建并自定义这些对象。只有在程序执行阶段才有可能看到所编写的对象。在这种环境下，要使对象的外观和行为如所需要的那样，是一件繁琐的过程，需要调整程序代码，再次运行程序观察最终结果。由于有了象 Delphi 这样的可视化开发工具，现在可以用对象可视地工作，立即在屏蔽上观察到结果，所见的对象如同在运行时出现的一样。这种性能解除了非可视环境中所必须的大量反复步骤。更为可喜的是在 Delphi 中，一旦把某个对象放入窗体后，对象的特征便被记录在程序的源代码中，这些代码最后被编译结合于程序的可执行文件中，也就是说可视化设计的结果会被自动转化为程序代码。类似这样的优点对编程人员具有极大的吸引力。可视化的观念是从对象概念发展而来，可视化编程的一切好处都是得益于面向对象的思想。事实上，当今流行的每一类开发工具都具有一个共同的特征，那就是面向对象的思想，与此同时可视化开发也是他们竞相引入的一个热点。

(来稿时间：1997 年 9 月)

© 中国科学院软件研究所 http://www.c-s-a.org.cn