

充分运用面向对象思想提高 BCB 编程的效率

Taking Full Advantage of the Object-oriented

Idea to Improve the Efficiency

of Programming in Borland C++ Builder

摘要: 面向对象技术经过 20 多年的发展, 日趋成熟, OOA (面向对象分析 Object-Oriented Analysis)、OOD (面向对象设计 Object-Oriented Design) 和 OOP (面向对象编程 Object-Oriented Programming) 已成为业界主流。在分析与设计领域, 对面向对象方式的研究比较深入; 然而, 在编写代码的过程中, 由于种种原因, 人们往往更关注应用软件功能的实现, 从而忽略了面向对象思想的运用。本文介绍了如何借助 UML 在编程的全过程中运用面向对象的思想, 并总结了运用面向对象思想指导软件开发对提高软件质量和开发效率的意义。

关键词: 面向对象 BCB UML OOP

1 引言

自从 1981 年面向对象(Object-Oriented 简记为 OO)的程序设计语言 Smalltalk-80 问世以来, 面向对象技术迅速发展, 而面向对象的思想也日趋成熟。在面向对象分析 (Object-Oriented Analysis 简记为 OOA)、面向对象设计 (Object-Oriented Design 简记为 OOD) 的过程中运用得十分成功, 已成为软件工程学的主流。然而在编程过程中, 许多程序员仍然习惯于过程式的编程方式。这与人们的思维定势有关系, 也有开发软件本身的问题, 比如 VC 这样的开发环境, 容易使程序员将注意力主要集中于开发实现软件的功能细节, 反而忽略了对 C++ 面向对象的设计思想的运用。

而 Borland 公司的产品 C++ Builder, 本身就是一个以面向对象技术设计开发的复杂软件系统的典范, 所以, 它充分提供了运用面向对象思想进行高效编程的广阔天地。目前, 程序员在编写代码时, 主要关注的是功能的实现, 界面的美观, 其实更该注重提高编程效率, 这就要充分运用面向对象技术的思想指导具体的代码编写。下面以 BCB5 为基础, 探讨如何利用面向对象的思想来指导程序员编写程序。

2 用 UML 对所要做程序进行描述

UML (联合建模语言 The Unified Modeling Language) 是由 Grady Booch 等三位著名的软件工程学家经过两年多的努力, 联手推出的一种编制软件蓝图的标准化语言, 于 1997 年被美国工业化组织 OGM(Object Management Group)接受, 并发布了 UML 的标准版本。现在已成为业界标准。对于从事具体编码的程序员来说, 采用 UML 来描述自己所设计的模块, 不仅是采用了一种规范的开发方法, 易于和他人交流合作, 而且便于自己对所负责开发部分的理解, 同时使自己向面向对象的开发方式迈出了第一步——以面向对象的思想将程序分解成若干组彼此关联的类。

目前最为流行的 UML 工具软件就是美国 Rational 软件公司的 Rational Rose。它是图形化的 OO CASE 工具, 集中体现了当代软件开发的先进思想, 把面向对象的建模与螺旋上升的开发过程相结合, 支持团队开发, 并且综合了其他的开发技术, 为运用面向对象的思想和技术, 控制系统的复杂性, 提高软件开发效率创造了必要的条件。利用 Rational Rose 给系统建模后, 可以直接生成 C++ 的框架代码。

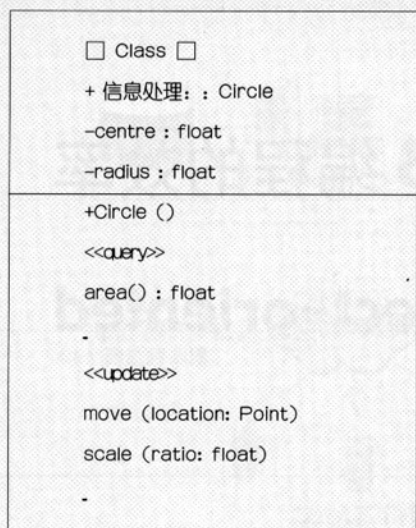


图1 类的UML表示

UML运用于编程,用得最多的是他的类图,如图1所示。

类图由三部分组成:第一部分是类名(Circle)和此类所属的模块名(信息处理),第二部分是类的属性即数据成员,第三部分是类的操作(Operation),实际上就是类的函数成员列表。可以通过给类图附加各种注释或文档,来说明类的有关信息。通过将程序的功能封装成一个一个的类,就可以从图中清晰地知道类的所有成员,而不用打开.h文件去找,也不用依赖开发环

境所提供的自动列表功能。

UML最方便之处在于它规范了一整套类之间的关联关系,从而可以很容易地画出整个系统的模块和类之间的关联关系,通过用UML表示出类之间的联系,可以很方便地知道如果要修改了某一类,可能会影响到哪些相关类,从而调试程序,捕获程序错误打下一个良好的基础。

图2所示为窗体类的调用图。从图中可以清楚地看见窗体之间的调用关系和信息传递的内容,通过给类加上不同的颜色,可以按调用方式将类区分为不同的种类,只被别人调用的类和调用别人的类。

对一个程序员来说,画出类和模块的调用图是非常关键和有用的,通过仔细分析这张图,程序员可以从中得到许多启发,有助于今后的开发工作。

(1) 会发现有些模块被反复的调用,因而如果改进这些模块,增强它的健壮性和效率会对程序本身产生很好的影响;

(2) 会发现不同的类之间功能有相同的地方,这时,可以将功能相同之处抽取出来,成为一个抽象类,原来的类改为继承关系,这就由面

向过程的开始方式转向了面向对象的开发方式。

(3) 会发现整个程序间有一些操作(比如文件操作功能,对某一控件的操作功能)被许多模块多次使用,就可以将所有这些公共操作抽取出来,然后分类形成若干个类,这样公共操作转化为类的函数成员,当需要用某一功能时,只需new一个相应对象,然后调用它的成员函数,用完马上delete掉,不再需要将功能代码分散在不同的模块之间;

(4) 可以安排开发次序,被别人调用、不调用别人的类和模块必须首先完成,实现主要功能的模块要先于实现次要功能的模块开发;

(5) 当需要扩充功能时,从这张图可以很容易地定位应如何修改程序,定下工作方案:是扩充现有类,还是新建一个类,新建的类要插入到什么地方,可能会对什么模块产生影响。

除了类图,UML还提供了状态图、交互图、合作图以及用例图等静态和动态建模的手段,不论采用什么形式,只要能让人容易地把握整个软件系统,就更有利于提高软件的开发效率。所以,软件界将UML的推出当作面向对象领域开发的一次重大进步,具有里程碑的意义。

3 用户界面的设计

设计界面可能是最有成就感的工作之一,尤其是使用过VB之类快速开发环境(RAD)的程序员,开发一个程序的第一步可能就是先弄个界面出来,然后再向其中的按钮之类控件的各种事件写代码。

这种最自然最直观的开发方法符合人的思维习惯。没什么不好,毕竟一个窗体比一个抽象的类要好理解得多,但也同样需要从面向对象的角度来做这个工作。

3.1 设计窗口框架

在弄清楚了程序要干什么之后,可以先设计一套窗体框架,决定按下某个按钮后会弹出

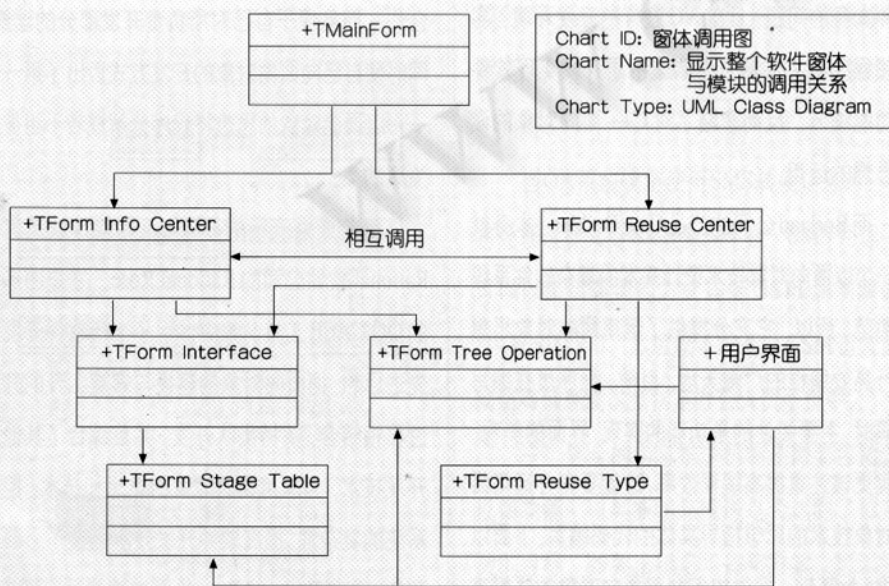


图2 类和模块调用图

哪个窗体,即定下窗体之间的调用关系,这样,可以快速利用BCB这类快速界面设计工具开发出一个可以运行的软件模型,可以给用户演示,然后做出相应调整,这样做可以避免项目后期对用户界做大的变动,同时,在设计整个界面框架的过程中,对软件要解决的问题也有了进一步的了解。

窗体框架设计好之后,就可以考虑非可视化的部分设计了,这一部分又可以分为两小部分:一是完成软件实际处理功能的模块群,用软件工程的术语来说这些模块同属于“业务逻辑层”,这一层是与用户界面无关的;另一部分是用户界面功能实现模块群,主要用来对用户的输入进行各种处理,与界面密切相关。

一般而言,针对较大的项目,业务逻辑层类的设计可能在进行分析阶段就开始,到设计用户界面时可能在进行分析阶段就开始,到设计用户界面时可能已经完成,但对于规模较小的项目,或者是程序员自行开发的软件,这两部分的设计常常是在设计界面完成之后同时完成的。

3.2 关于窗体的设计

用户界面设计完成,先别急着编代码,应先用UML画出窗体模块调用图(参见上图2),窗体的成员可以先空着,关键是要画出所有窗体之间的调用关系,同时明确每个窗体要实现什么功能。对着这张图,同样可以做不少工作,窗体和控件本就是类(VCL)。

第一,首先要弄清窗体之间传递什么参数,将其加入到窗体间关联线上。

第二,如果两个窗体之间的功能基本相同,仅有少量区别,比如一个多了几个按钮,那么,可以将公共的部分抽取出来,作为一个窗体基类,在这个基类窗体中书写实现基本功能的代码,然后将其Add To Repository(在BCB中),然后在IDE中新一个这个基类窗体的

Chart ID: 类继承图
Chart Name: 展示了类的继承关系
Chart Type: UML Class Diagram

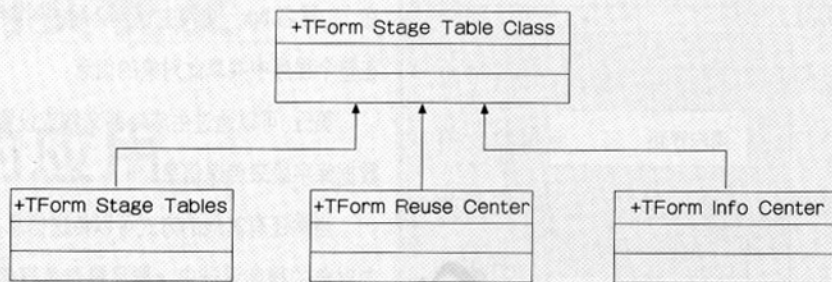


图3 继承的窗体类

对象,可以直接修改其界面,重载其事件;用UML表示的方法请参见图3,其中省去了窗体中的各种成员;

第三,如果两个窗体界面可以设计成一样的,仅处理的数据信息不同,可以利用多态构造函数(两个或两个以上的同名的构造函数,但参数不一样)的方法将两个窗体合为一个。

BCB采用的是VCL组件,为了更有效地使用它们,可以采用下面对对象的设计方式:

(1) 继承现有的组件,生成自己的VCL组件,从而扩充或定制它的功能,满足程序的特殊要求;

(2) 利用BCB5提供的组件聚合功能,实现组件重用,主要有两种方法:组件模板(Component Template)和组件框架(Tframe),其操作方法简单明了(可以参考BCB5的随机手册);“代码重用”是提高软件开发效率的关键之一,通过窗体的继承和VCL组件的重用,可以极大地提高软件的内聚性,减少模块间的耦合度。

4 层次结构

每一个软件都有其最合适的软件体系结构,这要依据具体项目的情况来决定。目前,分层体系结构用得较多,在这种体系结构中,每一层中由若干组有关联的对象组成,各层之间

的服务可以是单向或双向的。在具体开发时,将相关联的对象(类)合并放在一个模块中,每一层可能有多个模块,用UML方法,可以用包(Package)来表示软件的这种层次结构,并可以定义模块(包)之间的依赖关系。图4为一个软件的分层结构。

5 模块之间信息传递

BCB5采用了VB的开发思想,用面向对象的思想将各种Windows消息封装起来,从而使程序员可以不去处理诸如WM_CLICK之类的消息(与VB不同的是,BCB5同样可以方便地直接处理Windows消息,或定义自己的消息),而能够集中精力考虑如何以面向对象的思想来解决各模块间的信息传递问题。

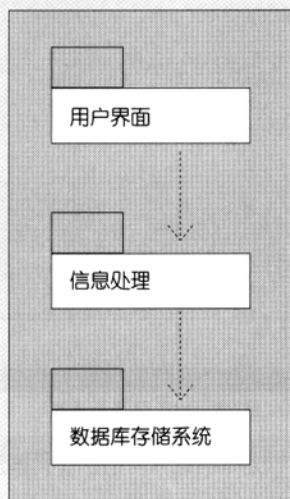
总结起来,各个对象(类)之间可以用以下方式传递信息:

第一,在类的构造函数中设置形参,从而在生成类的实例时住处可由实参携带,这种类往往有多个构造函数(比如前面介绍的窗体类);

第二,在类中设计属性,当生成此对象后,由调用者设置其某一属性,从而将相应的控制信息传给它,调用者也可以从属性中获取处理后的信息;

第三,在类中设置静态成员,从而在类的所

Chart ID: 软件分层图
Chart Name: NewPackageDiagram
Chart Type: UML Package Diagram
Chart Stereotype: ☐Package Diagram



说明:
整个软件由三大组件模块组成,其中数据存储部分采用现成的桌面型数据库,不需要开发。用户界面是整个软件开发的核心部分,信息处理部分主要完成一些公用的模块,比如访问数据库,树操作,目录操作等等。

图4 一个软件的分层结构

有实例中共享信息;

第四,生成全局的对象(在BCB5中,可以在“工程名.bpr”文件中创建全局的对象),从而在整个程序中共享此对象的信息;

第五,可以通过生成外部文件或设置专用的数据库字段来传递信息;

当然还有其他方式可以传送信息,但在面向对象的程序设计中,要尽量避免直接利用的Windows消息机制来传递信息,除非为了特殊的需要。

6 结束语

采用面向对象的思想指导开发可以很好地提高软件的质量和开发效率:

(1) 把握整个系统变得容易些了,UML提供了一种各个层次的开发人员交流思想的手段;

(2) 对系统功能的扩充也容易了,通过增加

新类和扩充已有的类的接口,可以实现对现有系统的功能扩展;

(3) 对系统的维护方便了,一个程序如果提供了UML描述的详尽而全面的文档,要捕获程序的错误和修正这些错误比直接去读源代码容易得多;

(4) 容易实现“代码复用”,如果说在设计时就考虑到代码的复用问题,精心设计其类的接口,同时降低其对外部环境的依赖性,就有可能实现“一个功能只需开发一次,便可在以后的新程序中反复使用”的梦想,当然这需要程序员建立自己的代码仓库。

总之,从面向对象开发的角度而言,用BCB可以以较高的效率开发出可重用性好、易于维护和易于扩充的软件产品,充分体现出面向对象开发的特点,这对当前急需解决的提高软件产业生产效率的问题而言,是非常有意义的。■

参 考 文 献

- 1 程序员大本营《Borland版》
- 2 吴炜煜,面向对象分析与设计(OOA/OOD/OOP),清华大学出版社,2000。
- 3 张龙祥,UML与系统分析设计,人民邮电出版社,2001.8。
- 4 Ronald J.Norman.Object-Oriented Systems Analysis and Design.Prentice-Hall International, Inc., 1998。
- 5 Edward Yourdon & Carl Argila. Object Oriented Analysis & Design. Prentice-Hall International, Inc., 1998。
- 6 郑人杰等,实用软件工程(第二版),清华大学出版社,1997.4。
- 7 Borland/Inprise公司著,梁志刚、汪浩、康向东、刘存根等译,C++ Builder5开发人员指南,机械工业出版社,2000。

(上接39页)

(4) 对于系统操作员和管理员来说,要及时地为有漏洞的程序打上补丁,如果还没有补丁,可以停掉该应用程序或者使用其它的替代品。特别对于UNIX/Linux系统来说,应该去掉特权程序的SUID或SGID。如果某个程序存在着这个漏洞,属于root,而这个文件被设了SUID,那么如攻击者就可以利用它来以root的身份执行我们准备好的代码。

4 小结

缓冲区溢出漏洞是安全漏洞中最为常见地一种形式。在本文中,我们详细分析了缓冲区溢出攻击的原理和方法。最常见地缓冲区溢出形式是攻击活动记录,然后在堆栈中植入代码,把堆栈设置为不可执行虽然可以防范大部分地攻击,但是要确保系统地安全,必须将上面的几种措施组合使用,才能提高防范的效率。

参考文献

- 1 十年来攻击和防卫的弱点, <http://go4.163.com/luohailiang/frame/ycqyc.htm>。
- 2 Jon C.Snader著,刘江林译,Effective TCP/IP Programming,中国电力出版社,2001:101-106。
- 3 王育舒,缓冲区溢出攻击分析[J],天津商学院学报,2000.3,20-23。