第25卷第3期 华东交通大学学报 Vol. 25 No. 3
2008年6月 Journal of East China Jiaotong University Jun. ,2008

文章编号:1005-0523(2008)03-0059-05

ASP. NET 环境下 MVC 模式的一种实现方法与应用

李 朔,李培松

(华东交通大学 信息工程学院,江西 南昌 330013)

摘要: MVC 是一种成熟优秀的设计模式,分析了 ASP. NET 平台自身提供的实现 MVC 模式机制的缺点,进一步引入前端控制器和具体页面控制器的两级控制器的概念,并以某软件企业"工作进度管理系统"为实例,介绍了 Web 开发中一种使用 Maverick. NET 作为前端控制器,以 ASP. NET 后台页面类作为页面控制器来实现 MVC 模式的方法,该方法可以有效改进 ASP. NET 原有机制实现 MVC 模式的不足.

关 键 词: ASP. NET; MVC; Maverick; 前端控制器

中图分类号: TP311 **文献标识码:** A

ASP. NET 是微软公司近年来推出的新型体系结构 Microsoft. NET 框架下的 Web 开发技术,它可以开发功能强大的 Web 应用,构建大型的 Web 程序和 Web 服务变得更加方便快捷. 越来越多的应用开始选择基于 ASP. NET 的 B/S 架构进行开发,而 Web 应用的开发较之传统 C/S 架构的桌面应用开发受限较多,对于较大规模的 B/S 应用程序,程序的调试及维护非常不便,如何在. NET 框架下构建易维护、可复用的 Web 应用程序一直是开发人员力图解决的问题. 本文以某软件企业工作进度管理应用系统的开发为例介绍 ASP. NET 平台上一种可行的 MVC 解决方案.

系统采用了 ASP. NET 下的 MVC 设计模式,实现了业务逻辑、控制逻辑及前端数据显示逻辑的分离,从而使系统具有良好的扩展性与可维护性.

1 设计模式与 MVC 设计模式

1.1 设计模式

过去的几十年中,人们在寻找问题的解决方案的过程当中,发现大量本质上相类似的问题会反复出现并不断改变面孔,它们"相类似的本质"就是模式的概念. 人们在利用面向对象技术解决这些相似问题的过程中有针对性的就某些同类问题设计总结了一些良好的解决方案,即所谓的设计模式. 面向对象技术的目的之一就是提高软件的重用性,而设计模式、设计方案的重用则从更深的层次上体现了重用的意义和本质.

1.2 MVC 设计模式

MVC 设计模式是 Model(模型) - View(视图) - Controller(控制器)的简称. 它要求应用程序在结构上按照功能的不同划分为 model,view 及 controller 三个功能模块. 作为一种软件设计模式,MVC 模块划分清晰、责任明确. 应用 MVC 模式开发的程序具有较好的伸缩性、可重用性和可维护性. MVC 架构如图 1 所示.

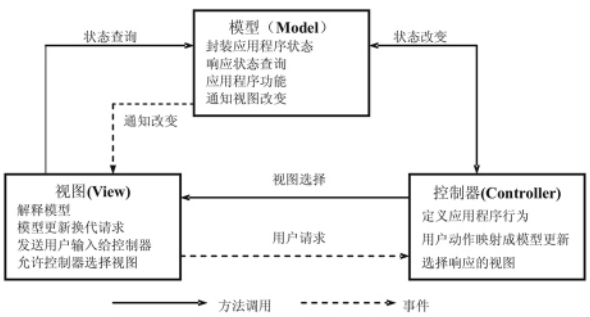

图 1 MVC 关系图

收稿日期:2007-12-12

作者简介:李朔(1977-),男,江苏南京人,华东交通大学硕士研究生,主要研究方向为电子商务,分布式计算.

模型是问题中相关数据的逻辑抽象,代表了系统的内在属性,是整个模型的核心. 它包含了应用程序的数据以及对数据进行访问、修改的业务逻辑规则,表达了程序所使用的数据和应用程式序的运行状态.

视图是模型的外部表现,一个模型可以对应一个或者多个视图. 用户看到视图界面并与之交互,它主要负责访问模型中的数据并决定如何展现这些数据. 在模型发生改变时,视图将维护与模型的一致性. 同时,视图也负责把用户请求传递给控制器.

控制器是模型与视图的联系纽带,它接收通过视图传入的外部消息,解释用户动作,并把它映射为模型执行的过程. 它控制着模型和视图之间的交互,决定对用户动作的响应流程. 它主要完成两个方面的任务: 一方面将用户与视图的交互动作映射成应用程序的标准业务事件并调用相应的模型进行处理,另一方面它将根据用户动作及模型的改变选择适合的视图来展现给用户.

2 ASP. NET 下的 MVC

2.1 ASP. NET 对 MVC 的实现及缺点

ASP. NET 本身提供了一个实现 MVC 模式的环境[1]. 在 ASP. NET 中,以页面(视图)为中心: 每一个视图(. aspx 文件)都有一个对应的控制器(页面隐藏代码. aspx. cs 文件). 页面控制器捕获视图中发生的事件和提交的数据,并调用模型来处理它们,控制器是介于视图和模型之间的一个中间层. 用户浏览器、视图、模型、控制器、数据库五者之间的应用系统模型如图 2 所示.

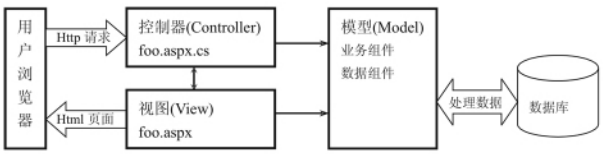

图 2 ASP. NET 下的 MVC 应用模型

在 ASP. NET 中使用基于 code - behind 技术的所谓 Web Form 开发方式,使得 Web 程序的开发接近于传统桌面程序,尽管这样可以有效的使美工人员和程序人员各自关注自己的领域,使前后台的开发同时进行,但这其中 aspx 文件和 aspx. cs 被混成了一个类,控制器只是对应页面的页面控制器(PageController),所以 ASP. NET 只是在形式上把视图和控制分开而已,视图和控制器之间有较强的耦合性. 而这种强耦合性会使得控制器选择下一视图时受到了很大限制,通常只能通过在代码中硬编码 Response. Redirect(……) 这样的语句来将请求转给下一个视图,而不是控制器. 本质上说,这种方式使 Web 应用仍是从一个页面流向另一个页面,当页面很多且页面间的这种联系复杂多变时,程序的复用性,扩展性和维护性能非常的不好.

2.2 对 ASP. NET 下 MVC 的扩展与 Maverick. NET

针对 ASP. NET 中控制器与视图的高耦合性,可以对其进行部分扩展,将控制器分为两层:

1) 前端控制器(Front Controller): 在整个应用程序中只有一个前端控制器,它负责接受所有客户端的动作请求,对请求做一定的处理后再选择转交给某一具体的页面控制器. 前端控制器决定了整个对用户的响应流程.

2) 具体页面控制器(Specific Page - Controller): 在整个系统中有多个具体的页面级控制器,它对应于页面,它负责根据用户的具体请求来调用模型功能进行具体的业务逻辑处理,然后决定数据通过视图以何种方式展现给用户. 采用两层控制器,将与视图无关的程序流转控制分离出来形成前端控制器,而与视图的数据展示有较强关系的页面控制功能则由具体页面控制器来实现.

2.3 Maverick. NET

Maverick. NET 是一个比较早的. NET 下的 MVC 实现,它是一个非常轻量级的实现,但其功能相当完备,具有较好的灵活性. 早期的 Maverick 在视图的层面上只支持纯粹的 ASP 方式或者是以 ASP 方式编写的 aspx 页面,这导致 ASP. NET 环境下 View-State、Postback 事件机制、用户控件等这些特性都不能使用,这极大限制了 ASP. NET 中的新特性的应用. 而自 1.2 版本后 Maverick. NET 对 ASP. NET 有了一个比较完全的支持,包括对服务器控件,用户控件等的支持,这样就使得可以在 Maverick 框架下充分利用 ASP. NET 的特性进行设计和开发.

Maverick. NET 作为前端控制器的主要控制流程如图 3 所示.

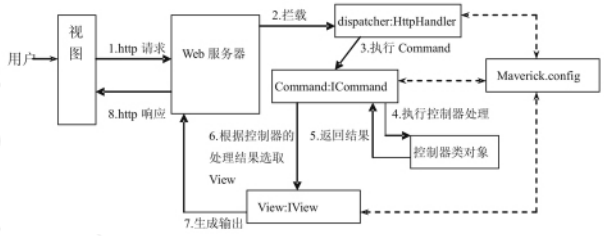

图 3 . Maverick. Net 控制流程

Maverick 首先将每一个由视图提交的 Http 请求拦截并通过 dispatcher 对象(HttpHandler 类) 映射成一个 command,接着由 Command 对象(实现了 Icommand 接口) 根据 Maverick 的核心配置文件 maverick. config 选择与该 Command 对应的控器对象并执行控制器对象中的特定方法,在控制器方法中调用模型执行业逻辑,最后 Command 对象根据控制器执行的结果决定返回给用户的视图,使用 View 对象生成输出[3].

3　某软件企业工作进度管理应用系统 MVC 模式的开发实例

3.1　系统需求

该应用系统主要包含企业工作任务的定立、委派、工作管理及工作进度查看等功能. 用户等级分为试用员工、员工部门主管,分公司经理,经理总监 5 个级别,各级别用户可用功能基本相同,但不同级别的用户对各功能的使用方式和访问权限不同. 系统还设有管理员级别用户,该级别用户可使用系统管理功能,如添加/编辑员工,添加/编辑部门等系统功能.

该工作进度管理系统的开发只是最终企业内完整信息化系统的第一阶段,是将来完整系统的一个子系统,要求它为将来的继续开发搭建好系统的基本框架,后续的其它子系统开发将以此为基础,分多个开发阶段加入进系统.

3.2　系统设计

系统基于 IIS + ASP. NET + SqlServer 平台开发,整体架构使用前述具有二级控制器的 MVC 模式. 视图部分通过. aspx 文件实现,用于向用户展现数据; 具体页面控制器通过. NET 页面的后台编码文件. aspx. cs 来实现,它对应于具体前台页面,控制前台页面的展现的数据内容和方式,这种方法可以最好的利用 ASP. NET 的 Web Form 编程方式的各种优点; 前端控制器使用 Maverick. Net 来实现,它主要用于截获用户访问动作,并根据 maverick. config 文件配置决定对用户动作的响应及程序的执行逻辑; 模型采用. NET 组件类[4] 实现,主要分为 AccessBase(基本访问) ,Proc(工作管理) ,Guage(工作进度) ,OpenFun(公共功能) ,WorkManage(工作管理) 等组件类. AccessBase 组件类作为基本的数据访问组件可以完成数据的存取更新等最基本的数据库操作,其它组件也可以使用该组件类完成一些数据访问工作. Proc 组件类完成一些较高级的数据访问,主要是用于实现对各实现业务逻辑的 SQL 存储过程的调用. Guage 组件类用于实现系统中项目、行为期、工作项三个级别工作的进度计算. OpenFun 组件类用于实现整个系统的一些公共功能,如登录功能,日志功能等. WorkManage 组件类用于实现工作设定、委托、进度调整的业务逻辑.

系统按照三层 Web 结构划分为表示层(Web) 、业务层(Biz) 和数据访问层(DAO) [2]. 结合 MVC 模式,表示层由视图与前端控制器和页面控制器组成,实现 View 和 Controller 的功能; 业务层由业务组件 Guage、OpenFun、WorkManage 等组成; 数据层由数据组件 AccessBase、Proc 等组成,由业务层和数据层共同实现 Model 的功能. 如图 4 所示

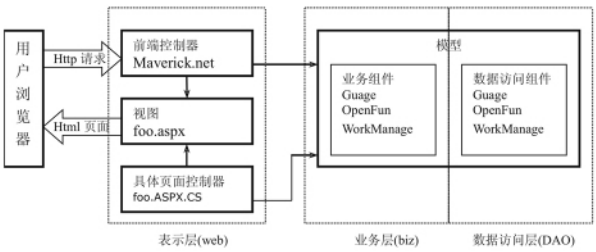

图 4　MVC 与三层 Web 结构示意图

4　部分实现代码

4.1　前端控制器 Maverick. NET

1) Maverick. NET 控制类

maverick 的中定义的多个控制类,具体完成对用户 URL 请求的前端处理控制. Maverick 中,根据控制类中是不是直接包含数据模型(比如一个用户定义的 DTO) 分为 Throwaway,FormUser,Throwaway-Form,ThrowawayFormUser 四个基类,用户编写的控制类时,应根据实际继承其中的一种类型,并重载特定方法完成实际功能. 系统中每一个用户 URL 请求都将映射为一个 command,而每一个 command 对应一个实际控制类,由该控制类完成前端总控功能. 控制类程类 mav. ctrl. cs 片断如下:

```
namespace OA _ WGW. mav. ctrl
{
    ……
public class addwork: Maverick. Ctl. Throwaway
    {
      public addwork( ) { }
      public override string Go( )    {
        OpenFun opf = new OpenFun( ) ;
```

```
            usrCls = opf. getCurrentUserClass( )
              switch( usrCls)
              {
                 case "1": return "GM";
                 case "2": return "LM";
                 case "3": return "M";
                 case "4":
                 case "5": return "E";
                 default: return "error";
              }
          }
        }
……
}
```

类 addwork 继承了基类 Throwaway，重载基类的 Go( ) 方法，该方法的作用是返回一个字符串，该字符串将被用于决定程序怎样流转. addWork 类所做的工作就是调用模型中方法取得当前用户等级，然后返回相应的字串“GM”(经理总监) /“LM”(分公司经理) /“M”(部门经理) /“E”(试用员工/员工) /“error”(非法使用) .

2) maverick. config

maverick. config

```
<? xml version = "1. 0"? >
< maverick version = "2. 0" default - view - type = "
document" default - trasform - type = "document" >
< commands >
……
< command name = "addwork" >
   < controller class = “OA _ WGW. mav. ctrl. ad-
dwork，OA _ WGW" / >
   < view name = "E"path = ". /workprocess/add/Ead-
dWork. aspx" / >
   < view name = "M"path = ". /workprocess/add/Mad-
dWork. aspx" / >
   < view name = "LM" path = ". /workprocess/add/
LMaddWork. aspx" / >
   < view name = "GM" path = ". /workprocess/add/
GMaddWork. aspx" / >
   < view name = "error"path = "LogIn _ error. aspx" / >
   < /command >
……
< /commands >
< /maverick >
```

maverick. config 是 maverick 的核心配置文件，它定义了所有有效的命令，并指定了处理每个命令的控制类和程序流转路线，如以上文本所示，它定义了一个名称为 addwork 的命令，指定了对应的控制类为 OA _ WGW. mav. ctrl. addwork，并且还指定了控制类返回不同字串时应转向的视图. 如果上述 addwork 控制类的 GO( ) 方法返回的是“LM”(表示用户是分公司经理) 时，则会转向 LMaddWork. aspx.

4.2 具体页面控制器

在本系统中后台文件的作用是具体页面控制器，它负责调用模型的功能进行具体的业务逻辑处理，然后决定视图中的展现内容和方式.

```
namespace OA _ WGW. WorkProcess. add
{……
public class LMaddWork: System. Web. UI. Page
   {
      ……
     public void showWorkItem( )
      {
      AccessBase ab = new AccessBase( ) ;
      DataSet ds = ab. getWorkItem( _ KEYS)
      LBWorkItems. DataSource = ds;
      LBWorkItems. DataTextField = "WITitle";
      LBWorkItems. DataValueField = "WIID";
      LBWorkItems. DataBind( ) ;
     }
        ……
   }
   ……
}
```

以上是对应页面 LMaddWork. aspx 的页面近代控制器代码片断，在页面类中定义了一个显示工作项的方法，它调用模型部分的数据访问组件 AccessBase 类的 getWorkItem 来获取一个数据集，并将数据绑定到页面上的列表控件 LBWorkItems 上，实现对数据的显示.

5 结语

采用 MVC 设计模式，可以很好的实现 WEB 程序中控制逻辑、业务逻辑、数据逻辑和显示之间的分离. ASP. NET 本身提供了一个实现这种模式的类似环境，开发者通过在 ASPX 页面中开发用户接口来实现 View，Controller 功能则在页面后台代码文件中

实现,而通过服务器组件实现 Model 功能. 虽然这种两层的系统较经典的 ASP 结构来说有明显的优点,它将用户显示(View) 从动作(Controller) 中分离出来提高了代码的重用性,但是视图(view) 和控制器(controller) 实质上还是牢牢的绑在一起的,这不能不说是 ASP. NET 本身实现 MVC 模式的一个缺憾,通过引入前端控制器完成程序的流程控制,可以有效的弥补这一缺点. 在实际系统开发中,由于引入了 Maverick. NET 作为整个结构中的前端控制器,使得整个系统各部分独立程度较好,整个系统易于修改与扩充,具有很好的可维护性与可扩展性.

参考文献:

[1] MARCO Bellinaso,KEVIN Hoffman 著,康博译. ASP. NET Web 站点高级编程 [M]. 北京: 清华大学出版社,2002.

[2] 启明工作室. ASP. NET + SQL Server 网络应用开发与实例 [M]. 北京: 人民邮电出版社,2005.

[3] Jim Moore. . Maverick. NET Manual [EB/OL]. http: //maverick. sourceforge. net/maverick - manual. html,2005 - 1 - 28.

[4] Dino Esposito. Programming Microsoft ASP. NET 2. 0 Core Reference [M]. 北京: 清华大学出版社,2006.

Implementation and Application of MVC under ASP. NET

LI Shuo,LI Pei - song

(School of Information Engineering,East China Jiaotong University,Nanchang 330013,China)

Abstract: MVC is an excellent design pattern. The paper analyzes the problems in the current MVC implementation of ASP. NET and introduces the concept about front controller and specific page - controller. By taking a real Enterprise Application as an example,the paper introduces a method which implements model MVC by means of employing Maverick. NET as a front controller and ASP. NET back page type as a page controller. This method can improve the original ASP. NET mechanism.

Key words: ASP. NET; MVC; maverick; front - controller

(责任编辑: 李　萍)

第25卷第3期 华东交通大学学报 Vol. 25 No. 3
2008年6月 Journal of East China Jiaotong University Jun., 2008

文章编号: 1005 - 0523(2008)03 - 0064 - 05

基于 OLE 自动化技术的 SPSS 二次开发原理及应用

胡 辉

(华东交通大学 经济管理学院应用统计研究所,江西 南昌 330013)

摘要: 随着计算机技术的迅速发展,建立在各种统计方法模型基础上的统计分析软件在实际中的应用成为可能. 但如何灵活使用统计软件,深度挖掘软件功能去解决实际中的复杂问题,往往需要对软件进行二次开发. 首先从软件开发原理的角度,以常见的统计分析软件 SPSS 为例,介绍了 SPSS 软件体系结构,并深入分析了其支持二次开发的原理: 基于组件的 OLE 自动化技术,最后以 Visual C + + 为例说明了如何在其他应用程序中使用 SPSS 提供的功能进行二次开发.

关 键 词: SPSS; 二次开发; 组件; 对象链接与嵌入; 自动化; Visual C + +

中图分类号: TP312 文献标识码: A

随着计算机技术的迅速发展,传统的很多统计方法在实际上的应用成为可能,各种建立在统计方法模型上的统计分析软件不断涌现. 如国外的 SAS, SPSS, STATISTICA 等,国内如 DPS 数据处理系统, NOSA(非典型数据统计分析系统),马克威 Markway 统计分析与数据挖掘软件等. 然而,由于实际工作中问题的复杂性,往往难以做到原始数据恰好符合软件中的统计模型和方法的适合条件(这需要对统计方法模型有正确的理解),或是在其他应用系统(如 Excel 甚至是一些决策支持系统)中希望利用统计软件得出的数据进行再加工(这需要对统计软件开发原理本身有深入的了解). 因此,灵活使用统计软件、深度挖掘软件功能就变得必要,而建立在这些统计分析软件基础上的二次开发便是解决这些问题的主要手段.

事实上, SAS, SPSS, STATISTICA 这些具有国际影响力的统计分析软件在支持二次开发方面是非常强大的. 如 SAS 本身就主要面向具有专业编程能力的统计人员; SPSS 一方面依靠菜单、对话框的方式赢得了非统计专业人士的厚爱,另一方面它内置的语法 Syntax、脚本 Script、SaxBasic 语言对于欲灵活使用它的人带来了很大的便利; STATISTICA 中的 STATISTICA Visual Basic 语言也同样是如此.

鉴于 SPSS 适用面的广泛性(既针对统计专业人员,又面向非统计专业人士),在此以 SPSS 为例探讨二次开发的技术原理及其应用. 首先从软件构建角度分析 SPSS 软件体系结构,然后着重在技术开发原理上深入探讨软件支持二次开发的原理,最后以 Visual C + + 为平台说明如何应用软件中的对象模型进行二次开发.

1 SPSS 软件的体系结构[1]

基于对象、基于组件的软件体系目前是主流的软件架构, SPSS 软件体系也莫能外. 它本身就是建立在面向对象、组件的基础之上的,其中 SPSS 类库和对象是支撑 SPSS 整个软件体系的基础.

1.1 SPSS 类库

类库是文件或文件中的组件,主要有两种类型(.tlb、.olb)的类库. 扩展名为.tlb 的类库可以作为单独的文件进行安装、扩展名为.olb 的类库可以嵌入到对象库文件内部. SPSS 提供了 SPSS 类库(spss-win.tlb)、SPSS 转轴表类库(spssvt.tlb)、SPSS RTF 类库(spssrtf.tlb)和 SPSS 图形编辑器 OLD 控制库(spssgctl.tlb).

安装完 SPSS 软件以后,第一次运行它时 SPSS 类库会自动注册到 Windows 注册数据库中,以后若有其他应用程序使用 SPSS 类库中的对象便可以直

收稿日期: 2008 - 03 - 19

作者简介: 胡 辉(1973 -),男,江西萍乡人,华东交通大学经济管理学院统计系教师,研究方向为统计分析与数据挖掘.