

基于 ASP.NET MVC 框架的干教系统的设计与实现

黄胜根¹, 陈蜀宇²

(1. 重庆大学 计算机学院, 重庆 400030; 2. 重庆大学 软件学院, 重庆 400030)

摘 要:ASP.NET MVC 体系架构是微软最近推出的基于 .NET 平台的开发框架。为了介绍 ASP.NET MVC 框架的优越性,以及采用该框架所带来的优势,首先介绍了 ASP.NET MVC 框架的体系结构、原理以及生命周期。然后通过在干部教育培训系统中的应用为例,展示了如何使用 ASP.NET MVC 框架搭建层次清晰的程序架构,从而使系统具有良好的灵活性、可扩展性以及易维护性。并且结合 C# 和 linq,给出了 ASP.NET MVC 框架下系统实现的技术细节和关键代码。

关键词:ASP.NET MVC; 设计模式; linq; C#

中图分类号:TP302.1

文献标识码:A

文章编号:1673-629X(2010)02-0190-04

Design and Implementation of Education Management System Based on ASP.NET MVC

HUANG Sheng-gen¹, CHEN Shu-yu²

(1. College of Computer, Chongqing University, Chongqing 400030, China;

2. College of Software Engineering, Chongqing University, Chongqing 400030, China)

Abstract:ASP.NET MVC is the new program framework issued by Microsoft company. To introduce the virtue of ASP.NET MVC and the advantage use the framework, introduce the system structure, principles and lifecycle of ASP.NET MVC. Then taking education management system as an example, it shows how to use ASP.NET MVC to construct application system and how it contributes to make the architecture more flexible, expandable and easy to maintain. It presents the code of the implementation method and procedure based on C# and linq.

Key words:ASP.NET MVC; design patterns; linq; C#

0 引言

MVC 模式是于 20 世纪 70 年代在 smaltalk80 的 GUI 设计中提出,它包括 3 个部分:模型(Model)、视图(View)和控制器(Controller),分别对应于内部数据、数据表示和输入输出控制部分^[1]。模型是与问题相关数据的逻辑抽象、没有用户界面。视图是模型的外在表现,是模型信息的界面展示,一个模型可以对应一个或者多个视图^[2]。控制器提取通过视图传输进来的外部信息,并将用户与视图的交互转换为基于应用程序行为的标准任务事件,再将标准任务事件解析为模型应执行的动作,包括激活任务逻辑或改变模型的状态。同时,模型的更新与修改也将通过控制器来通知视图,从而保持各个视图与模型的一致性。

模型通过更新视图的数据来反映其自身数据的变化^[3]。

MVC3 个组成部分的关系如图 1 所示^[4]。

.NET 作为现今的主流开发平台,其 Web 开发一直采用 WebForm 模式,从 .NET1.0 到 .NET2.0,ASP.NET 都采用的 WebForm 模式,开发人员在体验着 WebForm 模式带来的便利的同时,也体会到了视图与业务耦合带来的代码的混乱。Asp.net MVC 是微软官方提供的以 MVC 模式编写 Asp.net Web 应用程序的一个框架,它由 Castle 的 MonoRail 而来^[5]。

1 ASP.NET MVC 的原理与实现

ASP.NET MVC 应用的默认目录结构有三个顶层目录:/Controllers, /Models, /Views。其中控制器类应置于/Controllers 目录之中,数据模型类置于/Models 目录之中,视图模板置于/Views 目录之中。但 ASP.NET MVC 框架并不是必须使用这个结构,这只是一个默认的模式。

收稿日期:2009-05-26;修回日期:2009-08-22

基金项目:新世纪优秀人才支持计划(教技函[2005]35号 NCET-04~0843)

作者简介:黄胜根(1985-),男,硕士研究生,研究方向为网络终端及嵌入式 Linux;陈蜀宇,博导,研究方向为移动网络与嵌入式系统。

ASP.NET MVC 框架中,控制器类命名必须以 Controller 结尾,例如一个名为 Home 的 Controller 则要命名为 HomeController。并且每一个控制器类都在 /Views 目录中对应一个子目录,而且子目录的命名必须和控制器类的命名一致,例如控制器类 HomeController 的 View 就应该放到 /Views/Home 子目录中。

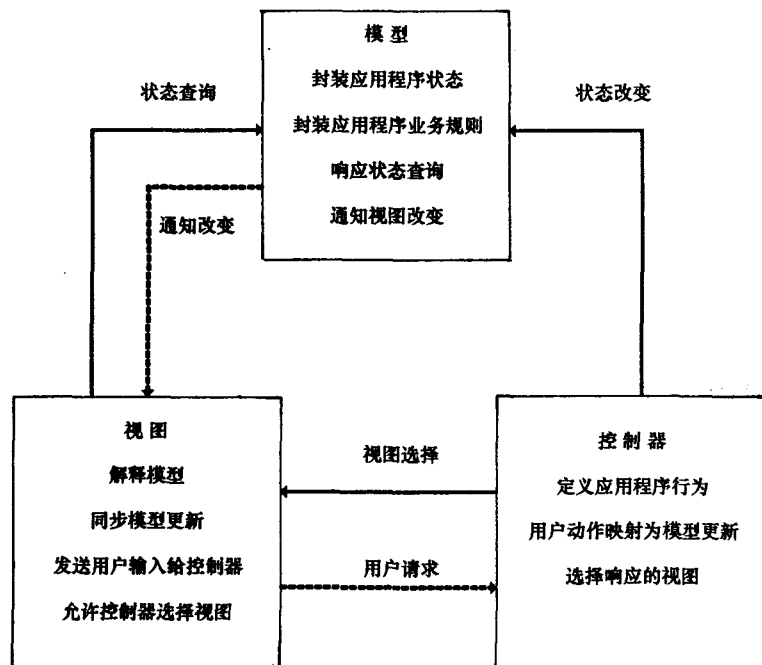


图1 MVC组成结构图

1.1 控制器

控制器是 MVC 中比较重要的部分,它主要负责从模型中取出数据,进行业务逻辑的处理。在 ASP.NET MVC 框架中,所有的控制器类都继承 Controller 基类,而 Controller 基类又实现了 IController 接口。Controller 基类的部分代码如下所示:

```
public class Controller : IController
{
    public Controller();
    .....
    protected internal virtual void Execute(ControllerContext controllerContext);
    protected virtual void RenderView(string viewName, string masterName, object viewData);
}
```

Controller 实现了接口中的 Execute 方法,在 Route 匹配到 Controller 之后,就会调用 Execute 方法来进入 Controller 的处理。Controller 基类中还定义了 TempData 和 ViewData,用来将数据从控制器传递到视图。在控制器类中,每个 Public 方法都作为一个 Action,在 ASP.NET MVC 中 URL 都是映射到某个 Action 中,然

后再由匹配的 Action 来处理业务逻辑,并返回视图。

1.2 模型

模型主要负责数据的存取等基本行为,同时也包括数据的持久化,也可以采用相应的 ORM 模型。ASP.NET MVC 框架允许使用任何数据访问模式或者框架来获取和管理模型。在 .NET 3.5 中,可以采用

linq to SQL, 或者 ADO.NET Entity Framework 等来搭建模型层。在干部教育培训系统中,采用了 ADO.NET Entity Framework + linq to Entity 来管理模型。

1.3 视图

视图是 MVC 模式中与用户直接接触的部分,它负责数据的呈现。ASP.NET MVC 框架中,默认是使用 Web 窗体来作为 View 的,所有的视图窗体都继承自 ViewPage 或者 ViewPage<T>,这里 T 指代模型中为 View 中数据显示所定义的数据类型,而 ViewPage<T> 也继承自 ViewPage。所以如果使用 ASPX 页面作为 ASP.NET MVC 的视图引擎,则所有的 ASPX 页面都必须继承自 ViewPage。ViewPage 类的部分代码如下所示:

```
public class ViewPage : Page, IViewDataContainer
```

```
{
    public ViewPage();
    public TempDataDictionary TempData { get; }
    public UrlHelper Url { get; set; }
    public ViewContext ViewContext { get; }
    public ViewData ViewData { get; }
}
```

ViewPage 继承自 Page 类,并且实现了 IViewDataContainer 接口,同时还提供了一些 Helper 类的实例。ViewPage 有一个 ViewData 属性,通过该属性,可以访问到 Controller 作为参数传递给 RenderView() 方法的特定于视图的数据对象。在视图里,可以通过后期绑定或者强类型的方式访问 ViewData。如果视图是从 ViewPage 继承而来,那么 ViewData 属性是个后期绑定的类型;如果视图是从泛型的 ViewPage<T> 继承而来,那么 ViewData 属性就是强类型的,匹配 Controller 传入的数据的类型。

例如,干部教育培训系统中,在 ArticleAdd 视图中,后台类是从 ViewPage<T> 继承而来,其中 CQG-JArticleAddViewData 是在 Module 中定义的数据类型,视图类如下:

```
public partial class ArticleAdd : ViewPage< CQG-  
JArticleAdd ViewData>  
{  
}
```

为了实现在视图中数据的显示,可以采用在 ASPX 文件里使用行内代码,或者使用服务器控件的方式。同时,在视图中可以使用在 ViewPage 中定义的 Helper 方法组来实现很多灵活的功能,从而更好地将界面与逻辑分离。因为 ASP.NET MVC 框架并不只支持 ASPX 一种视图,所以,在 ViewPage 中定义的方法组将提高程序的扩展性和移植性。

2 ASP.NET MVC 的生命周期

当请求一个普通的 ASP.NET 应用程序页面的时候,对于每一个页面请求都会在磁盘上有相同的一个页面,这个页面执行 ProcessRequest() 方法,并把内容发回浏览器。但在 ASP.NET MVC 框架中并不是这样工作的。当一个 URL 请求到达时,该请求被路由到控制器类上,并由控制器类生成内容并发回浏览器。

在 ASP.NET MVC 框架中,当应用程序第一次启动时,首先会创建路由表(RouteTable),一个应用程序有且只有一个路由表。路由表在 Global.asax 文件中创建,路由表由 RouteTable.Routes 的静态属性表示,这个属性表示了路由对象的集合^[6]。

当对 ASP.NET MVC 应用程序发起请求的时候,请求会被 URLRoutingModule Http Module 拦截,并封装当前的 HttpContext,传递给之前创建的路由表。HttpContext 中包含了 URL、表单参数、查询字符串以及与当前请求关联的 Cookie^[7]。如果当前请求和路由表中的路由对象匹配,就返回路由对象(RouteData),并创建表示当前 HttpContext 和 RouteData 的 RequestContext 对象。最后,Module 实例化带有当前 RequestContext 的 MvcHandler。

Module 实例化 MvcHandler 后,便执行 MvcHandler,调用 MvcHandler 对象中的 ProcessRequest() 方法,并创建一个新的控制器,同时构造 ControllerContext 对象,最后调用控制器类的 Execute() 方法,并传入 ControllerContext。

Execute() 方法首先创建 TempData 对象,用以保存下次请求必须的临时数据,并构建请求的参数列表。Execute() 方法通过反射来查找并执行控制器类中的

方法,而这些方法也正是在控制器类中写的 Action。最后控制器类方法会调用 RenderView() 或 RedirectToAction() 方法,将视图呈现给浏览器^[8]。

3 MVC 框架在干部教育系统中的应用

重庆市干部教育培训规模大、培训对象和班次多、培训内容丰富,需要管理的工作程序、信息项目非常复杂,重庆市干部教育培训系统的建立,利用先进的信息技术为干部教育培训工作提供系统、高效、便捷、可靠、安全的信息管理服务。通过有效地存储、利用工作数据和经验,使这项工作纵向保持连续性、横向保持完整性,并为培训工作的创新提供有力的信息支持和决策辅助。

3.1 系统总体架构

重庆市干部教育培训工作,包括培训任务的规划、分解、培训教材的编写、培训基地的建设、培训师资的管理、培训结果的统计、培训效果的评价等。

该培训系统总体功能设计分五大部分:培训子系统、电子档案子系统、培训资源管理子系统、信息交流管理子系统、干部管理子系统。系统总体功能结构图如图 2 所示。

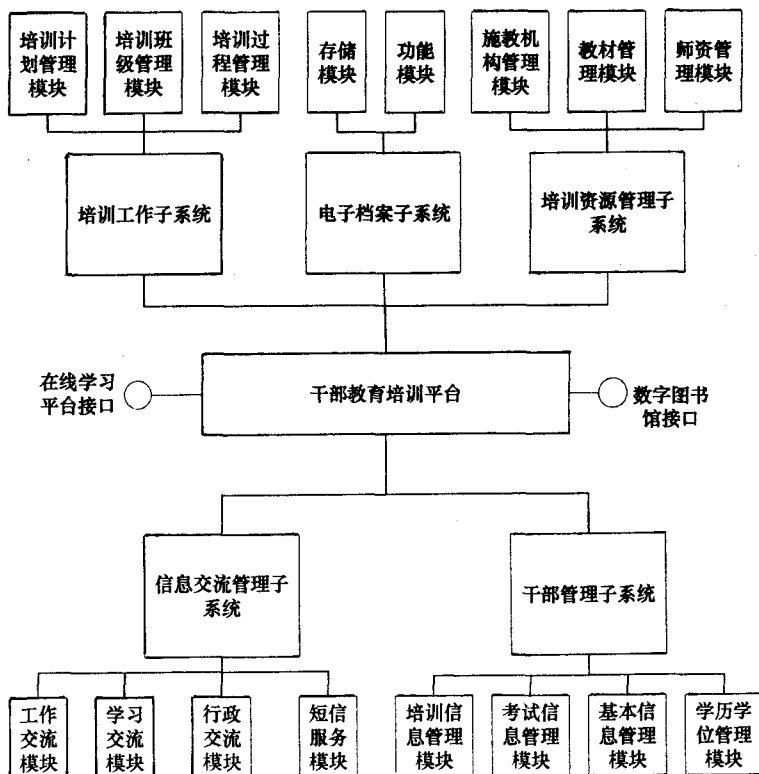


图 2 系统总体功能结构图

系统采用 ASP.NET MVC 框架,并结合 ADO.NET Entity Framework,linq to Entity 等技术构建系统总体程序框架。

3.2 系统实现关键代码

3.2.1 控制器类

在系统中,根据实际业务的需求,设计了 15 个控制器类,在控制器类中进行事件的响应、业务逻辑的处理以及视图的访问。

部分代码如下所示:

```
public class SchoolController : ControllerBase
{
    public void SchoolList(int? id)
    {
        SchoolSchoolListViewData viewData = new
        SchoolSchoolListViewData();
        var Schools = cqgip.GetSchoolOrglist();
        viewData.Schools = Schools.Skip(20 * (id.Value
        - 1)).Take(20).ToList();
        RenderView("SchoolList", viewData);
    }
}
```

3.2.2 模型类

在系统的设计过程中,采用了 ADO.NET Entity Framework + linq to Entity 来管理模型,并针对每一个 View,设计一个相应的 ViewData 类,这样可以轻松地实现数据库的基本操作。部分代码如下:

```
public partial class CQGJEntities : global::System.Data.
Objects.ObjectContext
{
    public CQGJEntities(): base("name = CQGJEntities", "CQGJEntities")
    {
    }
}

namespace CQGJ. Models
{
    public class SchoolSchoolListViewData: BaseView-
    Data
    {
        public List< CQGJ. passport. b01 > Schools {
        get; set; }
    }
}
```

3.2.3 视图类

在系统中,根据各功能模块的划分,设计了 15 块视图类,每一块视图类对应一个相对独立的子功能,同

时也对应/Controllers 中的一个控制器类文件。采用了行内代码和 Helper 方法组的方式实现视图的显示。

部分代码如下:

```
<div class="infotitle">
    编辑施教机构信息
</div>
<form id="form_ school" action="< % = Url.
Action("SchoolUpdate ") % >" method="post" >
    <input type="hidden" name="OrgID" value =
    "< % = ViewData.Org. b0111 % >" />
</form>
```

4 结束语

ASP.NET MVC 框架改变了传统的 ASP.NET 中视图与业务逻辑耦合的弊病,有助于将应用程序分割成若干逻辑部件,使程序设计变得更加容易。在重庆干部教育培训系统中采用 ASP.NET MVC 框架,可以形成清晰的程序框架,便于系统编码阶段的任务分工。同时,清晰的框架也为后期系统的维护以及功能扩展提供了良好的基础和有利的条件。

参考文献:

- [1] 黎永良,崔杜武. MVC 设计模式的改进与应用[J]. 计算机工程,2005,31(9):96-97.
- [2] 甘早斌,彭 彬,李志欣. 基于集中控制的 MVC 模型[J]. 计算机工程与设计,2005,26(2):454-455.
- [3] 许劲松,石 磊. 基于递归 MVC 结构的 Web 应用软件分析模式[J]. 计算机工程与设计,2005,26(12):3417-3419.
- [4] Sauter P, V? gler G, Specht G, et al. Extending the MVC design pattern towards a task - oriented development approach for pervasive computing applications[C]//Proc Int Conf on Architecture of Computing Systems - Organic and Pervasive Computing(ARCS 2004). [s. l.]: Springer - Verlag, 2004:309-321.
- [5] 姚慧广,赵岳松. Web 编程中 MVC 模型的应用[J]. 微机发展(现更名:计算机技术与发展),2002,12(3):9-10.
- [6] 刘 胜,王 诚,郭 亮,等. 基于虚拟分公司模式的销售管理系统开发[J]. 重庆大学学报:自然科学版,2005,28(6):15-18.
- [7] Alexander C, Ishikawa S. A Pattern Language[M]. New York:Oxford Univ Press,1977.
- [8] Gamma E, Helm R. Design Patterns Elements of Reusable Object - Oriented Software[M]. New York: Addison Wesley Longman,1998.