文章编号:1001 - 9081(2005)01 - 0238 - 03

一种基于双 Servlet 控制器的 MVC 模式的设计与研究

刘运龙[1],黄烟波[2]
(1. 湖南师范大学 继续教育学院,湖南 长沙 410012; 2. 中南大学 网络中心,湖南 长沙 410012)
(net@hnsdyc.net)

摘 要:MVC 模式是开发 Web 应用程序的一种重要方法。文中首先简要介绍了通常的 MVC 模式及其运行过程,然后在此基础上提出了一种改进的 MVC 模式,即一种含有两个 Servlet 控制器的混合型 MVC 模式,并详细介绍了这种模式的设计和实现,最后分析了这种改进后的新模式的性能和特点。

关键词:MVC; 敏感表单; 事件处理
中图分类号: TP311.1 **文献标识码**:A

Design and study of a kind of MVC model based on double Servlet controller

LIU Yun-long[1], HUANG Yan-bo[2]
(1. *College of Continuing Education, Hunan Normal University, Changsha Hunan* 410012, *China;*
2. *Network Center, Central South University, Changsha Hunan* 410012, *China*)

Abstract: MVC model is an important method in developing Web application program. First of all, the common MVC model and its running procession were introduced briefly. Then a kind of improved MVC model based on double Servlet controller was put forward. With great emphasis put on the design and implementation of this improved MVC model, its feature and performance were analyzed.

Key words: MVC; sensitive forms; event handling

1 基于双 Servlet 控制器的 MVC 模式的提出

模型-视图-控制器(Model-View-Controller, MVC)是分布式应用软件(如 Web 应用程序)开发过程中广为采用的一种设计方法。在设计和构建任何稍复杂一点的 Web 应用程序时,开发者都应该考虑使用 Model2(MVC)架构[1]。

1.1 通常的 MVC 模式及其运行过程

MVC 模式包括 3 类对象:模型 Model 是应用对象,用于存储状态及更新视图;视图 View 是模型数据的表现形式;控制器 Controller 抽象用户的交互和应用的语义映射,传递用户输入给应用程序,根据用户的输入和上下文信息选择适当的视图显示数据[2]。

在 MVC 模式的一种基本应用形式中其控制器的运行过程(通过 service 方法实现)比较简洁:从操作库中获取操作,然后调用它的 perform 方法,操作的 perform 方法返回一个操作路径选择器,为请求确定路由[3]。另一种形式是在此基础上添加了事件处理机制,即触发应用程序事件的 MVC 框架,这种形式更好用,因为应用程序通过对事件做出反应,无需修改框架就可以扩展该框架的功能。这种扩展了事件处理机制的 MVC 模式其控制器服务方法的运行过程包括获取操作、产生事件、触发事件、执行操作、重新设置事件类型、再次触发事件、转向新的路由等多个步骤[3]。为便于区别,本文将这两种形式分别简称为基本型 MVC 和扩展型 MVC。

1.2 一种改进的 MVC 模式的提出

扩展型 MVC 架构比基本型 MVC 架构更好用,具有更强的功能。但在其运行过程中每个操作执行前和执行后都要进行一次触发事件的额外处理,这就大大增加了额外时间开销。这种额外处理对需要进行事件处理的操作来说是必要的,但事实上大量的操作是不必进行事件处理的。例如对于数据处理的常见操作插入、删除、修改和查询,一般情况下只有插入数据时必需进行事件处理。因此可以对扩展型 MVC 架构进一步进行改进,使之具有更好的综合性能。改进的一条途径就是在控制器中另添加一个基本型 MVC 架构的 Servlet 控制器,专门用来处理无需事件处理的操作。这种在扩展型 MVC 架构基础上进一步改进的、含有两个 Servlet 控制器的 MVC 架构就是本文提出的基于双 Servlet 控制器的 MVC 架构。为便于与前两种形式区分,本文中简称这种形式为混合型 MVC。

2 基于双 Servlet 控制器的 MVC 模式的结构设计

基于双 Servlet 控制器的混合型 MVC 模式对扩展型 MVC 架构的主要改进之处是将直接构筑控制器的 Servlet 由原来的一个改进为两个,且为不完全相同的两个(其中一个是基本型 MVC 框架中的控制器 Servlet,另一个是扩展型 MVC 框架中的控制器 Servlet)。

基于双 Servlet 控制器的 MVC 模式的系统结构图如图 1 所示。

其结构特点是:

(1) 同时含有基本型 MVC 框架中的 Servlet 控制器(图 1 中取名为 ActionServlet1)和扩展型 MVC 框架中的 Servlet 控

收稿日期:2004 - 07 - 14;**修订日期**:2004 - 12 - 08

作者简介:刘运龙(1971 -),男,湖南衡阳人,助理研究员,硕士,主要研究方向:计算机应用技术、Web 应用程序开发; 黄烟波(1959 -),男,湖南邵阳人,教授,主要研究方向:计算机应用技术、网络技术.

制器(图 1 中取名为 ActionServlet);(2) 所有操作分为两类:需要进行事件处理的操作和不需要进行事件处理的操作;(3) 两个 Servlet 控制器分别处理两类不同操作。ActionServlet 控制器处理需要进行事件处理的操作,ActionServlet1 控制器处理不需要进行事件处理的操作;(4) 两个 Servlet 控制器可并发执行。

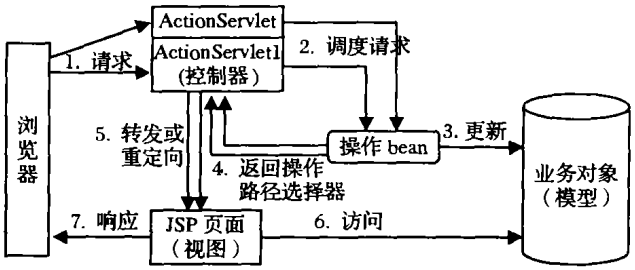

图 1　基于双 Servlet 控制器的 MVC 架构图

其工作过程是:如果在请求的处理过程中不需要进行事件处理,系统的运行过程将按基本型 MVC 的步骤进行;如果在请求的处理过程中需要进行事件处理,系统的运行过程将按扩展型 MVC 的步骤进行。

3　基于双 Servlet 控制器的 MVC 模式的具体实现

笔者在开发一个现代远程教育综合管理信息系统时,设计并实现了一个基于双 Servlet 控制器的混合型 MVC 模型。这种混合型 MVC 结构不是前述基本型和扩展型两种 MVC 模式的简单混合,而是两者的有机结合。两者共用一个初始部署描述文件(web. xml),共用一个属性文件(actions. properties)。其实现过程的要点如下:

3.1　基本构成类分成 2 个目录

框架的各种基本构成类和各种业务操作类分别放在 actions 和 actions1 目录下。其中 actions1 目录下存放了基本型 MVC 架构的构成类和不需事件处理的业务操作类;actions 目录下存放了扩展型 MVC 架构的构成类和需要进行事件处理的业务操作类。

3.2　Web 应用程序部署描述文件 web. xml 内容的关键设置

```
<web-app>
  <servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>ActionServlet</servlet-class>
    <init-param>
      <param-name>action-mappings</param-name>
      <param-value>actions</param-value>
    </init-param>
  </servlet>
  <servlet>
    <servlet-name>action1</servlet-name>
    <servlet-class>ActionServlet1</servlet-class>
    <init-param>
      <param-name>action1-mappings</param-name>
      <param-value>actions</param-value>
    </inti-param>
  </servlet>
  <servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
    <servlet-name>action1</servlet-name>
    <url-pattern>*.do1</url-pattern>
  </servlet-mapping>
  ……
</web-app>
```

3.3　相应的 ActionServlet、ActionRouter 类及 JSP 文件中请求信息的设置

ActionServlet 和 actions 目录里 ActionRouter 中的初始化参数均设为 action-mappings,由 ActionServlet 处理的 JSP 页面中的请求形式均设为 *.do。ActionServlet1 和 actions1 目录里 ActionRouter 中的初始化参数均设为 action1-mappings,由 ActionServlet1 处理的 JSP 页面中的请求形式均设为 *.do1。

3.4　敏感操作与非敏感操作的分类

重复提交敏感操作会导致出现错误或者不想要的状态,系统设计时应该考虑捕获重复提敏感操作的机制,区分哪些操作是敏感操作,哪些表单是敏感表单,哪些操作是带有敏感表单的操作。对于数据处理的常见操作插入、删除、修改和查询,插入操作是敏感操作,后三种可归为非敏感操作,因此将敏感操作类存放在由 ActionServlet 控制器处理的目录(actions)中而将非敏感操作类存放在由 ActionServlet1 控制器处理的目录(actions1)中。

4　基于双 Servlet 控制器的 MVC 架构的特性分析

基于双 Servlet 控制器的混合型 MVC 框架能同时兼顾系统结构、功能和速度三个方面,实现三者的有机统一,优化系统的整体性能。

4.1　结构方面

MVC 框架最大的优点就是结构清晰,可重用性、可扩展性、可维护性、灵活性和有效性都比较好。它不仅实现了将业务对象与 JSP 页面分离开,而且实现了将内容的产生与内容的显示分离开。正是这种分离,使得可插入部件的应用成为可能,进而可以开发出灵活的,可重用的和适应性强的软件。

双 Servlet 控制器结构中的两个 Servlet 保持了原来结构的特点,因而具有原结构所具有的优点。

4.2　功能方面

双 Servlet 控制器中的用来执行事件处理的 Servlet,除了捕获敏感表单的重复提交外,还可以进一步扩展功能。例如,如果该 Servlet 在执行每个操作之前和刚刚执行之后触发事件,应用程序可以处理那些事件来完成多种任务,例如为用户进行身份验证或为国际化设置地区[3]。这是基本型 MVC 所不及的。

捕获重复提交敏感表单尽管还可以用定制标记的技术来实现,但这种方法失去了整体结构的完美性。

4.3　性能方面

到目前为止,还没有评价 B/S 应用系统性能的统一标准,但可以从系统响应用户的请求时间、系统支持用户并行访问的能力以及系统的稳定性等方面进行考虑[4]。

本文从上述三个方面对既含有基本型 MVC 结构的控制器(在本文后续部分简称 I 型控制器)又含有扩展型 MVC 结构的控制器(在本文后续部分简称 II 型控制器)的双 Servlet 控制器 MVC 模式与只含有单个 II 型 Servlet 控制器的 MVC 模式作比较:

(1) 服务速度加快。

服务器响应同一请求在基于双 Servlet 控制器的 MVC 架构上的运行时间小于在只由单个 Ⅱ 型 Servlet 控制器构建的扩展型 MVC 框架上的运行时间。

时间比较分初次运行某个操作和会话期间再次运行某个操作两种情况。

初次运行某个操作由于系统要为操作类实例化,两种情况下的时间差比较明显。现以一个登录身份验证的操作为实验,表 1 记录了两种框架下首次运行该操作所需要的响应时间:

表 1 同一操作在两种模型下的时间比较(单位:ms)

次序	1	2	3	4	5	6	7	8
改进后的混合型	531	551	530	531	541	561	541	882
原扩展型	611	591	571	631	581	581	561	982
时间差	80	40	41	100	40	20	20	100

经计算,前者的平均响应时间比后者的平均响应时间小 55ms,即前者的响应速度快。这是因为登录操作是非敏感操作,在混合型 MVC 框架中是由双 Servlet 控制器中的 Ⅰ 型控制器完成的,完成一次操作只需三个主要步骤。后者则没有选择的余地,只能由 Ⅱ 型控制器完成,完成一次操作需要七个以上步骤。表 2 和表 3 进一步记录了两种情况下的各步骤的详细时间:

表 2 混合型中的 Ⅰ 型控制器各步执行时间(单位:ms)

步骤	1	2	3
平均时间	10	422	154

表 3 扩展型控制器各步执行时间(单位:ms)

步骤	1	2	3	4	5	6	7
平均时间	30	10	0	428	0	0	174

在其他扩展型操作中,第 3,5,6 步也出现过 10ms 的现象,由实验数据可得出,Ⅱ 型 Servlet 比 Ⅰ 型 Servlet 多运行 4 个步骤,时间至少多 30ms 以上,最多可能多 60ms 以上(当第 3、5、6 步分别也为 10ms 时)。

一般情况,由 Ⅰ 型、Ⅱ 型 Servlet 混合构建的双 Servlet 控制器 MVC 模型与通常的扩展型 MVC 模型首次运行同一个操作的时间比较有:

设在基于双 Servlet 控制器的 MVC 架构中共有 n 个不同操作的请求,其中需运行 Ⅰ 型 Servlet 的概率为 P,则此双 Servlet MVC 的运行时间比扩展型 MVC 中 Servlet 的运行时间最小快 nP△T(△T 表示在 nP 次两种情况下的最小时间差)。

会话期间运行某个操作尽管没有操作类实例化的过程,但运行一次 Ⅱ 型控制器的代码段比 Ⅰ 型控制器的代码段长,理论上运行时间肯定要长。相应地对于数以万次的运行其累计时间差就更明显了。

(2) 单位时间内系统支持并行访问的用户数增加。

设基本型 MVC 系统支持并行访问的用户数为 n1,扩展型 MVC 系统支持并行访问的用户数为 n2,则双 Servlet 混合型 MVC 系统支持并行访问的用户数 m 有关系式 $n1 < m \leqslant n1 + n2$ 且 $n2 < m \leqslant n1 + n2$。

(3) 双 Servlet 控制器的两个 Servlet 同时出现异常抛出的概率减少,系统的稳定性增强。

设双 Servlet 控制器 Ⅰ 型 Servlet 出现异常抛出的概率为 P1,Ⅱ 型 Servlet 发生异常抛出的概率为 P2,则两个 Servlet 同时出现异常抛出的概率是 $P1 \cdot P2$,显然 $P1 \cdot P2 < P2$。

5 结语

本文提出的基于双 Servlet 控制器(一个是基本型 MVC 的控制器,另一个是扩展型 MVC 的控制器)的混合型 MVC 框架结构,能实现结构、功能和速度三者的优化组合,取得较好的整体性能。与基本型 MVC 架构比较,具有更强的功能,与扩展型 MVC 架构比较,具有更快的速度,更强的支持并行访问的能力,更少的异常抛出概率和更强的稳定性。

参考文献:

[1] BROWN S. JSP 编程指南[M]. 第 2 版. 王军, 等译. 北京: 电子工业出版社, 2002. 719.

[2] 李杰, 李建华, 胡韧. 设计模式在电子商务中的应用[J]. 计算机工程, 2003, 29(8): 100 - 102, 139.

[3] GEARY DM. JSP 高级开发与应用[M]. 贺民, 译. 北京: 科学出版社, 2002. 120 - 161.

[4] 谭骏珊, 吴昌盛. 基于 B/S 模式应用系统性能优化的研究[J]. 计算机应用, 2003, 23(1): 70 - 72.

(上接第 237 页)

```
QueryManager qm = new QueryManager();
AdhocQueryRequest req = new AdhocQueryRequest();
/*
设置查询条件
*/
RegistryResponse regres;
regres = qm. submitAdhocQuery(req);
/*
客户端处理查询结果
*/
```

最后,将 Web 应用程序部署在 IIS 6.0 服务器中。

5.3 Web Browser 层

只要用户使用支持 ASP. NET 的网络浏览器即可。

6 结语

ebXML 注册中心在整个 ebXML 框架中处于极其重要的核心地位。通过对 ebXML 2.0 规范中注册中心体系结构规范的研究,分析了现有的基于 C/S 结构的 ebXML 注册中心的在安全等方面的缺点,提出了基于 B/S 结构的 ebXML 注册中心的框架,并介绍了在. NET 平台下的实现方法。除了本文中提到的 C/S 结构和 B/S 结构,ebXML 注册中心还有其他的实现模式,这将是我们进一步研究的方向。

参考文献:

[1] CARNAHAN JL. ebXML Registry Services Specification[EB/OL]. http://www.oasis-open.org/committees/regrep/documents/2.0/specs/ebrs.pdf, 2002.

[2] 段智华. 浅谈 SOAP[EB/OL]. http://www-900.ibm.com/developerWorks/cn/xml/x-sisoap/index.shtml, 2001.

[3] FREEMAN A, JONES A. Microsoft .NET XML Web Services Step by Step[M]. Microsoft Press, 2002.

[4] ULLMAN C, GOODE C. BEGINNING ASP. NET using C#[M]. Wrox Press, 2001.