

## 基于 MVC 模式的 WAP 开发与设计

吴轶婷, 姚琳

(北京科技大学 信息工程学院, 北京 100083)

(wu\_yiting@126.com)

**摘 要:**针对时下流行的 WAP 开发技术,对 WAP 协议在移动互联网中的应用进行研究,并运用成熟的 MVC 开发模式进行设计与应用。首先简要介绍了 WAP 协议与 WAP 网络以及 MVC 模式,并将 MVC 模式引入到 WAP 系统中来,使 WAP 应用程序得到优化,最后给出一个具体的应用实例。

**关键词:**无线应用协议;MVC(Model-View-Controller);JavaBean

**中图分类号:**TP311.52 **文献标识码:**A

## Development and design of WAP based on MVC pattern

WU Yi-ting, YAO Lin

(Department of Information and Technology School of Beijing University of Science and Technology, Beijing 100083, China)

**Abstract:** Focusing on WAP technology and its application in mobile Internet, which is popular at present, there was a research about using mature MVC pattern on designing such applications. It gave a brief introduction of WAP, WAP network and MVC pattern, and also introduced MVC pattern to WAP system, which made WAP program optimized. A concrete application was given as an example lastly.

**Key words:** WAP(Wireless Application Protocol); MVC(Model-View-Controller); JavaBean

### 0 引言

随着移动通信技术以及 Internet 技术的发展, WAP (Wireless Application Protocol, 无线应用协议) 技术已经成为移动终端访问无线信息服务的全球主要标准,也是实现移动数据以及增值业务的技术基础。WAP 使无线用户通过移动电话和微浏览器接入因特网上的信息和业务,为人们提供了更为便捷迅速的信息获取渠道。

无线数据网络和有线网络相比,表现在更苛刻的通信环境。与台式机相比,手机受到内存空间、处理能力等因素的制约,因此在 WAP 应用上程序的执行效率显得尤为重要。随着 WAP 应用的发展与普及, WAP 应用程序模块的功能复杂性也日益加大,这就要求提高软件的重用性。Model-View-Controller (MVC) 开发模式可以分离数据访问和数据表现,让开发人员可以开发一个可伸缩性强、便于扩展的控制器,来维护整个流程。MVC 的目的是增加代码的重用率,减少数据表达,数据描述和应用操作的耦合度。同时也使得软件的可维护性,可修复性,可扩展性,灵活性以及封装性大大提高。因此 WAP 应用程序开发适合使用 MVC 架构。

### 1 WAP 概述

WAP 是一组向移动终端提供互联网应用和服务的开放式协议,是一个用于解决手机上网浏览问题,并适用于不同无线网络技术的全球性无线网络规范。它由一系列协议组成,涵盖了从 WAP 设备到用户代理以及传输协议与 GSM 信道的各个方面,用来标准化无线通信设备,访问 WAP 网站上的页面,收发电子邮件等。WAP 将移动网络和 Internet 以及公司的局域网紧密地联系起来,提供一种与网络类型、运行商和终端设备都独立的移动增值业务。WAP 是为了解决无线终端

与 Internet 相连时的一种协议,它是为了把 Internet 网上结构复杂、内容丰富的信息进行提炼,简化以便显示在比 PC 机尺寸小得多的无线终端上而提出来的。

全球 WAP Forum 最早提出一种新的标记语言——WML (无线标记语言),并用它的第一个版本 WML1.0 完成了 WAP1.0 标准。经不断改进出现了 WAP1.1 和 WAP1.2 等标准。WAP2.0 采用了 XHTML (Extensible Hypertext Markup Language, 扩展超文本标记语言)使其描述精炼而严密。它克服了 WML1.X 的繁琐和冗余。WAP2.0 在 WAP1.x 的基础上做了很大的改进,对 WAP 协议的结构作了重大变革,采用了一些最新的标准和协议,以适应无线环境的变化和预期的市场需求。同时还定义了很多新的业务和应用。

WAP 网络框架由 WAP 移动终端、WAP 网关和 WAP 内容服务器三部分组成。其中, WAP 网关起着协议“翻译”的作用; WAP 内容服务器用于存储网络信息。当移动终端发出要访问 WAP 内容服务器的 URL 请求后,信号经过无线网络,以 WAP 协议方式发送请求至 WAP 网关,然后网关进行解析、翻译,再以 HTTP 协议方式与 WAP 内容服务器之间进行交互,最后 WAP 网关将返回的内容进行解码和压缩,并把结果送回给移动终端。这样,就完成了整个的会话过程。

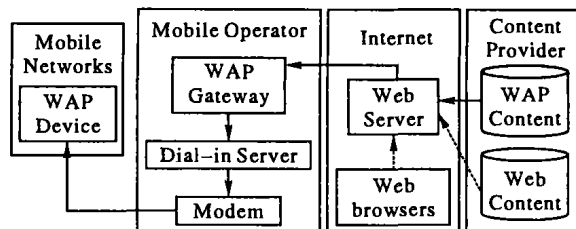


图1 WAP应用模型

在图1中,移动网络(GSM、CDMA等)中的移动WAP设

备,通过拨号连接到接入服务器(RAS,或者远程控制服务器)的 Modem 上。这个服务器使得 WAP 设备可以使用协议来工作。有一些如同 Internet Service Provider 将提供给用户底层协议,PPP(Point-to-Point)协议。这个协议用在整个 WAP 链中的下一个环节设备——由移动运营商提供的 WAP 网关。网关连接无线和 Web 世界,使得 WAP 设备能够操作普通的 Internet。

## 2 MVC 模式概述

MVC 最初是在 Smalltalk-80 中被用来构建用户界面的,是许多交互和界面系统的构成基础。M 代表模型 Model, V 代表视图 View, C 代表控制器 Controller。

模型部件是软件所处理问题逻辑在独立于外在显示内容和形式情况下的内在抽象,封装了问题的核心数据、逻辑和功能的计算关系,它独立于具体的界面表达和 I/O 操作。

视图部件把表示模型数据及逻辑关系和状态的信息及特定形式展示给用户。它从模型获得显示信息,对于相同的信息可以有多个不同的显示形式或视图。

控制部件是处理用户与软件的交互操作的,其职责是控制提供模型中任何变化的传播,确保用户界面于模型间的对应联系;它接受用户的输入,将输入反馈给模型,进而实现对模型的计算控制,是使模型和视图协调工作的部件。通常一个视图具有一个控制器。

模型、视图与控制器的分离,使得一个模型可以具有多个显示视图。如果用户通过某个视图的控制器改变了模型的数据,所有其他依赖于这些数据的视图都应反映到这些变化。因此,无论何时发生了何种数据变化,控制器都会将变化通知所有的视图,导致显示的更新。这实际上是一种模型的变化—传播机制。

MVC 模型结构如图 2 所示。MVC 模式把模型、视图、控制器实行分离,使设计和使用有了很大灵活性。模型独立于视图,使模型有了

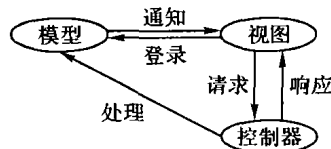


图 2 MVC 模型结构

更大的可移植性。基于该编程模型进行开发,各个模块之间的接口一旦制定,开发人员可根据模块的主要功能选用适当的技术对该模块进行有针对性的开发,且无须熟悉实现其他模块的具体技术细节,最后根据所制定的接口进行系统集成即可。因此,将 MVC 编程模式引入 WAP 应用程序的设计开发中,可以有效地弥补目前 WAP 应用程序开发上的不足。

## 3 MVC 在 WAP 系统中的应用

传统的以 JSP 语言为基础的 WAP 应用程序将程序的表示逻辑和控制逻辑都放在 JSP 程序中,这就影响了程序的易读性,并给今后修改页面带来了难度。解决这一问题的最佳方法就是将表示逻辑和控制逻辑分离,即采用 MVC 模式。在此模式中,视图 View 表示页面显示部分如 WAP 页面的风格、色调、显示的内容等,这部分经常需要变化;模型 Model 表示商业规则和数据,要相对稳定;而表示控制的控制器 Controller 则最稳定。其结构如图 3。

MVC 模式使 JSP 更好地将代码与 WML 或 XHTML 部分分开,尤其是在 WAP2.0 采用 JSP/XHTML 语言编写,页面设计更为复杂的情况下,有利于页面设计人员和代码开发人员的分离,提高效率。同时也提高了程序模块的重用性、可维护性及封装性。

### 3.1 视图

视图 View 代表系统的显示,是 WAP 应用程序的外在表现。JSP 通过 Java Bean 来读取模型中的数据,模型和控制器则负责对 Java Bean 的数据更新。视图动态地展现出这些数据并指明是通过何种方式来展现这些数据的。但视图又不会对数据库中的数据造成破坏与更改。WAP 页面因手机内存、浏览器尺寸及网络速度等因素的制约,对视图的展现有其特殊的要求,如文字应尽量简炼、图片不宜过多过大、页面不宜过长等。WAP2.0 具有更强的页面表现力,它与 WAP1.2 的区别就主要表现在视图部分。

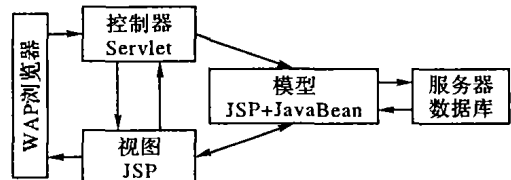


图 3 基于 JSP 的 MVC 模型

### 3.2 模型

模型 Model 包含了 WAP 应用程序功能的核心,负责存储与应用程序相关的数据,如商业规则和商业数据。模型模块主要功能为事务处理与逻辑判断,通常将其封装为 JavaBean,JavaBean 从数据库中取出客户所需数据,如用户的账号信息,公司的数据信息等,或者读取相关文件中的内容,如文本等。

### 3.3 控制器

控制器 Controller 对象协调模型与视图,把用户请求翻译成系统识别的事件。控制器主要提供以下功能:首先接收用户端发送的与网络协议相关的请求,然后解析请求并转换为事务逻辑模块 Model 的输入参数,调用相应的事务逻辑模块进行处理,最后根据事务逻辑模块的处理结果调用相应的用户视图模块生成结果页面,返回至浏览器。Controller 还有一个重要的功能就是同步 View 和 Model 的数据。在 ModelManger 中包含一个 ModelUpdateManger,它把系统事件转换为一个 Model 的集合,即所有需要同步的 Model,然后通知 Listeners 去做同步操作。

## 4 实例:在线注册功能的设计

随着 WAP 网站的内容和形式越来越丰富,手机用户在线注册功能在许多网站中都要用到,如 WAP 游戏网站、WAP 博客网站等。由于运营商的 WAP 平台可以提供获取用户手机号等用户唯一标识的功能,使得手机用户的注册信息更具可靠性与可操纵性。下面就介绍一下在线注册系统的设计模式与开发方法。

本设计的基本页面与功能如下:

用户注册页面:用户输入姓名、密码信息,页面提供格式控制功能(由于手机设备屏幕较小,不易被手机用户以外的人误看到,因此不必将密码以星号形式输出)。

注册信息处理页面:调用信息处理模块对用户输入的信息进行处理:根据取到的用户手机号判断该用户是否已注册,如已注册则提示进入登录页面。信息验证通过则将新用户的姓名、密码、手机号信息存入数据库,并跳转到注册成功页面。

结果显示页面:注册成功或不成功分别设置一个结果显示页面,并提示用户进入主页面(注册成功时)或登录页面(重复注册时)或重新注册页面(注册失败时)。

在图 4 所示的注册系统 MVC 模型中:

(1)将用户在注册页面填写的姓名和密码信息提交给 ControllerServlet。

(2)表单信息填写不当则返回注册页面让用户重新填写。

(3)注册信息处理页 `regist_do.jsp` 创建 `UserBean` 实例。

```
UserBean user = new UserBean();
```

调用 `UserBean` 的 `getPhone` 方法获取用户的手机号。

```
String phone = user.getPhone();
```

`getPhone` 方法中手机号的取得:

```
String phone = request.getHeader("x-up-calling-line-id");
```

若无法取到公开的手机号码,可以取加密的手机号码 `deviceid` (`deviceid` 为 256 位的加密手机号):

```
String phone = request.getHeader("deviceid");
```

调用 `UserBean` 的 `isRegist` 方法判断该用户是否已注册。

```
boolean isReg = user.isRegist(phone);
```

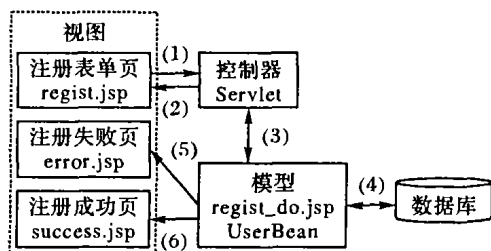


图4 注册系统的MVC模型

(4)在获取用户信息和新建用户信息的过程中通过连接池与数据库相连,并读取和插入信息。

(5)如果 `isReg` 为 `true` 则表示该用户之前已注册,跳转到注册失败页面 `error.jsp` 并提示其进入登录页面凭密码进主页面。

(6)如果 `isReg` 为 `false` 则表示该用户还未注册,调用 `UserBean` 的 `NewUser` 方法注册该用户,在数据库内新建一条

用户数据,存入该用户的姓名、密码、手机号,并返回其 `userid`。

```
int userid = user.NewUser(name, pwd, phone);
```

跳转到注册成功页面 `success.jsp`,并传递 `userid` 参数。提示用户进入主页面。

## 5 结语

从上述简单应用实例可以看到,基于MVC模式的WAP应用程序结构清晰、便于管理、便于模块重用,也为系统的维护与改进提供了方便。近年来WAP技术的发展我们有目共睹,而MVC模式的应用为WAP开发注入了新的生命力。

### 参考文献:

- [1] (美) Wireless Application Protocol Forum Ltd. WAP 无线应用协议 [M]. 侯春萍, 宋梅, 蔡涛, 等译. 北京: 机械工业出版社, 2000.
- [2] 王宏锦, 杨明极, 纪合宝. WAP 增值业务网站建设研究[J]. 哈尔滨理工大学学报, 2004, (9) 6: 128.
- [3] 张为, 李绍滋. 基于 MVC 模式的 Web 应用研究[J]. 长沙电力学院学报, 2004, (5): 33.
- [4] 刘学英. 基于 J2EE 的 WAP 增值应用软件的设计与实现[J]. 计算机工程与应用, 2002, 23: 144.
- [5] 郭梅, 江红. Struts 在实现 MVC 架构中的应用[J]. 计算机与现代化, 2004, (1): 106.
- [6] BATNI RP, LEE CC, VARNEY DW. Enhanced services in WAP-enabled networks[J]. Bell Labs Technical Journal, 2000, 5(3): 145 - 152.

(上接第 1886 页)

Hibernate 架构实现了 `InvestmentDAO` 给出的方法,例如 `InvestmentDaoImpl` 的 `newInvestment` 方法实现如下:

```
public void newInvestment(Investment investment) throws Exception
{
    Session session = HibernateUtil.currentSession();
    Transaction tx = session.beginTransaction();
    session.save(investment);
    tx.commit();
    HibernateUtil.closeSession();
}
```

可见在采用了 Hibernate 之后,访问数据库操作的 SQL 语句会大量减少,像这种插入数据的操作甚至可以不用 SQL 语句。这样使系统不仅开发起来更快,也使系统更容易更新和维护,例如假设 `Investment` 类多了几个属性或者少了几个属性我们都可以不用修改业务逻辑,只需要修改映射文件即可。

### 4.4 采用多架构实现的系统与传统开发系统的对比

按照以前的开发模式(例如 Model1)来开发这个系统,所有的视图和业务逻辑都会混杂在一起,很难维护,例如可能用大量的 JSP 来写页面,里面嵌套很多的 Java 业务代码,也可能用很多的 `JavaBean` 来做业务,但是这些业务和页面是紧紧连接在一起的,往往修改视图或者业务逻辑都会造成另外一个部分的更改。在采用了 Struts 之后,视图、模型和控制器三个部分得到了彻底的分离,特别是控制器部分,通过采用了配置文件来配置业务流程,使得修改和维护业务流程的工作减轻。

在采用 O/R 技术以前,如果直接用 JDBC 来写更新和管理信息系统的代码,会有很大的工作量,而且可移植性也不是太好。在采用 Hibernate 之后,需要的 SQL 语句就很少了,所以无论是开发还是维护都轻松了很多。

最后,由于利用 DAO 模式来把 Struts 和 Hibernate 架构无缝集成起来,让它们优势互补,使得系统的业务流程控制与业务逻辑开发都得到了最好的解决方案,使得系统的开发工作量减小,系统的可维护性、灵活性和健壮性都得到了提高。

## 5 结语

本文详细解释了 MVC 模式,论述了 Struts 架构的运行原理,并详细探讨了 O/R 映射工作机制,阐述了 Hibernate 的实现方式,归纳了它们各自的优缺点,最后将这两个架构有机集成起来。然后将它们实际应用于开发移动项目管理信息系统,说明了用这个集成起来的架构开发应用系统的方法,以简化应用系统的开发,并提高应用系统的健壮性、可维护性和灵活性。

### 参考文献:

- [1] 阎宏. Java 与模式[M]. 北京: 电子工业出版社, 2002.
- [2] (美) GAMMA E, HELM R, VUSSIDES RJ. 设计模式: 可复用面向对象软件的基础[M]. 李英军, 等译. 北京: 机械工业出版社, 2000.
- [3] (美) GRAND M. Java 企业设计模式[M]. 张威, 卢庆龄, 等译. 北京: 电子工业出版社, 2003.
- [4] 孙卫琴. 精通 Struts: 基于 MVC 的 Java Web 设计与开发[M]. 北京: 电子工业出版社, 2004.
- [5] 赵晨希. 用 Struts 建立 MVC 应用的介绍[EB/OL]. <http://www-900.ibm.com/developerWorks/cn/java/1-struts-mvc/index.shtml>, 2003 - 12.
- [6] Apache Software. Struts Project[EB/OL]. <http://jakata.apache.org/struts/>, 2005.
- [7] Hibernate org. Hibernate Project[EB/OL]. <http://www.hibernate.org>, 2005.