基于移动 Web 应用技术的 FoxERP 企业资源计划系统的研究与实现

李成大　张　京　刘甫迎
（成都电子机械高等专科学校计算机工程系　成都 610031）

摘　要　论述了基于移动 Web 应用技术开发的企业资源计划系统 FoxERP 的功能、特点，讨论了其关键技术并给出了实现的关键程序段。

关键词　FoxERP，企业资源计划，移动 Web 应用，ASP. NET，C#

Research and Implementation of FoxERP Based on Movable Web Application Technology

LI Cheng-Da　ZHANG Jing　LIU Fu-Ying
(Department of Computer Engineering, Chengdu Electromechanical College, Chengdu 610031)

Abstract　This paper introduces FoxERP system based on Movable Web Application Technology and discusses it's key programs.

Keywords　ERP, Movable Web application technology, ASP. NET, C#

1　FoxERP 企业资源计划系统的组成、架构及功能特点

1.1　FoxERP 的组成、开发环境及功能特点

企业资源计划 ERP(Enterprise Resources Planning)系统是对企业的信息流、资金流、物流进行全面信息化管理的系统，目前在国内外正方兴未艾。

FoxERP 企业资源计划系统是笔者的课题组为成都小狐狸软件公司历经数年开发的今年推出的新产品。它是关于制造业的企业资源计划系统，由主生产计划子系统、库存管理子系统、销售管理子系统、采购管理子系统、供应链管理子系统、工作流管理子系统、制造标准及维护子系统、在制品管理子系统、制令管理子系统、外包管理子系统、JIT 管理子系统、总账管理子系统、成本管理子系统、应收应付管理子系统、费用管理子系统、固定资产管理子系统、人事管理子系统、考勤管理子系统、工资管理子系统、质量管理子系统、绩效管理子系统、设备管理子系统等 22 个子系统组成。从接单、制造到出货对整体企业流程进行了全面的数字化管理(其中 JIT 管理子系统对“拉动式”的生产模式进行了管理)。其供应链管理子系统和销售管理子系统实际上是 SCM(供应链管理系统)和 CRM(客户关系管理系统)的雏形，为今后的协同商务系统打下了基础。

FoxERP 企业资源计划系统使用了 ASP. NET、C# 主流语言和 SQL Sever2000 数据库技术，很好地支持了 B/S 模式结构，其人机交互能力强、界面友好、速度快，且还具有系统的总帮助和每一页的页帮助信息，便于广大用户使用。设置了多级用户权限(例如，工作流管理系统的签发等)，保证了系统安全。

本系统既能用于实际的企业管理，又能用于课堂教学和实验，也可以在 Internet 环境中用于不同地区的学生作为远程教育软件使用。

1.2　FoxERP 的系统架构

FoxERP 在架构方面分为两个部分：普通 Web 应用和专门为移动用户开发的移动 Web 应用。两部分应用程序实际上是系统的两部分不同的表征，用户可以使用 PC 访问普通 Web 应用站点进行管理工作进程等操作，也可以使用移动设备访问移动 Web 应用站点。两部分应用在功能上并不完全一样，移动 Web 应用提供的所有功能仅仅是普通 Web 应用提供的一小部分，因此两部分应用在逻辑上有着共同性，在设计上两部分应用可以共用逻辑层，如图 1 所示。

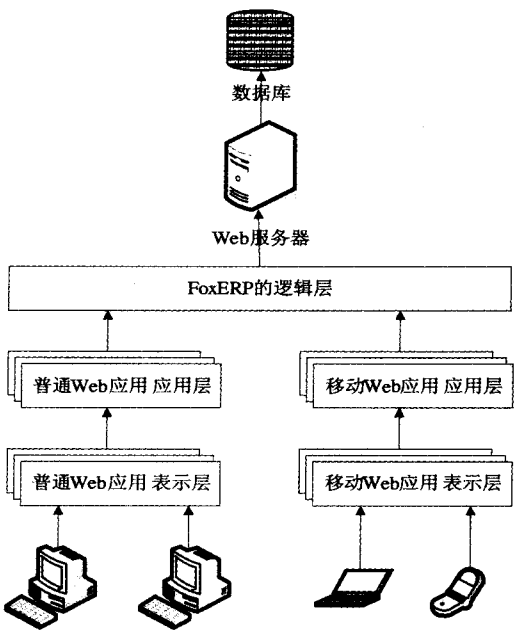

图 1　FoxERP 的系统架构

2　总体设计思想和主要关键技术

本软件系统总体设计思想是：“方便、规范、安全、速度和用户广”。为了规范，本系统开发时使用 PowerDesigner CASE 工具进行了基于 UML 的 CDM、PDM 和 OOM(用例

李成大　副教授，硕士，主要研究方向：计算机网络、信息安全；张　京　副教授，硕士，主要研究方向：软件、数据库；刘甫迎　教授，主要研究方向：数据库、网络技术。

图、时序图、类图)建模,此外还使用了下述的移动 Web 应用技术等。

2.1 使用了移动 Web 应用技术

现今移动设备已成为人类日常生活中的一部分。当这些移动设备连接到 Internet 时,移动设备的力量将无穷无尽,移动设备的使用者可以在任何事件、任何地点使用移动设备接收和发送数据。当然,移动设备的使用离不开移动设备应用程序的服务器,典型的移动应用程序是在服务器上使用 WML、WMLScript 和 WBMP 开发的。

对于动态 WML 应用程序,开发者可以使用 ASP、JSP、PHP 等等。移动设备包括蜂窝电话、寻呼机、掌中浏览器、袖珍 PC 和车载 PC。这些设备中少数支持 WML,少数支持 HTML,还有少数同时支持 WML 和 HTML。如果想确保应用程序能在大多数的移动设备中使用,必须以 WML 和有限的 HTML 创建应用程序。

.NET 框架包括用于移动 Web 开发的 ASP. NET,基于 ASP. NET 的移动 Web 应用程序既支持传统的 Web 客户端如 IE 和 Netscape,又支持移动客户端如 Smartphone、PDA 等。ASP. NET 移动 Web 应用程序可以在. NET 支持的如 C # 等语言环境下进行开发 。

2.1.1 ASP. NET 移动 Web 应用程序

要开发 ASP. NET 移动 Web 应用程序,必须引用由. NET Mobile Web SDK 提供(通过 MobileUI. DLL 文件)的命名空间 System. Mobile. UI。. NET Mobile Web SDK 提供了 3 个容器对象:MobilePage、Form 和 Panel。MobilePage 控件是移动应用程序的重要容器,相当于普通 ASP. NET 中 Page 对象,一个单独的 MolilePage 可以有一个或多个 Form 控件。移动 Form 控件和普通 ASP. NET 中的 Form 控件的概念基本一致,一个移动 Form 控件可以有 0 个或多个 Panel 控件。Panel 控件用于给各种 Mobile 控件分组,Molile 控件可以被分为 3 个组。用户界面(UI)控件:是如 Lable 控件一样允许用户控制用户界面的一组控件。该组控件基本上能和普通的 ASP. NET 中的控件一一对应。验证(Validatiln)控件:允许验证用户输入值的正确性,如 RequiredFieldValidator 控件。这些控件在向服务器发送数据之前验证用户输入的数据。这组控件和普通的 ASP. NET 中的验证控件十分类似。功能(Utility)控件:是诸如日历控件一类的控件。

2.1.2 FoxERP 移动 Web 应用程序设计

从理论上讲,FoxERP 在普通 Web 应用程序中所提供的功能都可以在移动 Web 应用下提供,但在实际的情况中并没有这个必要,移动 Web 应用只需要提供部分系统中的核心功能即可。例如在 FoxERP 的关于工程进度管理方面的子系统中,移动 Web 应用主要提供以下功能。

·用户登录

和普通 Web 应用一样,移动 Web 应用也需要对用户的身份进行识别和验证。因为系统所有的操作都是基于用户身份和用户角色的,如果用户不能提供有效的身份信息,则无论是在普通 Web 应用还是移动 Web 应用下都不能登录系统。图 2 就是用户登录系统的页面。

·按日期查看工作进程

移动 Web 应用可以按照日期查看工作进程,日期以星期为单位,用户可以选择查看某个星期内所有的工作进程。系统将以列表的形式展示在该周内所有存在工作进程的日期,单击日期查看该日期下的所有工作进程情况。

·管理工作进程

用户可以新增、修改、删除个人的工作进程,其中工作进程的信息包括工作进程的工作分类、发生日期、工作量统计以及工作进程描述。图 3 是修改工作进程的页面。

在前面已经介绍了 FoxERP 的系统架构,其中重点提到移动 Web 应用和普通 Web 应用共用同一个逻辑层。因此,下面重点介绍移动 Web 应用层和表示层。

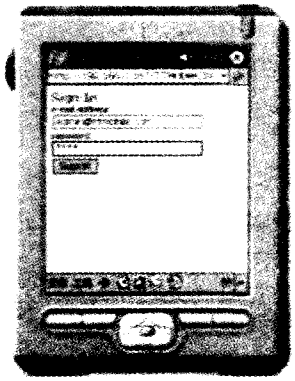

图 2 用户登录系统的页面

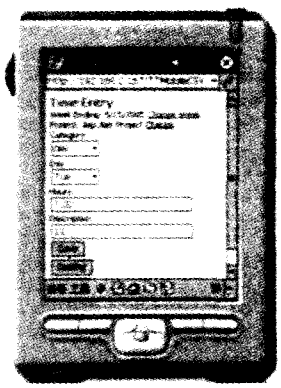

图 3 修改工作进程的页面

(1)FoxERP 的移动 Web 应用层和表示层

FoxERP 的移动 Web 应用层例如其用户登录:SignIn. aspx 是用户登录页面,用户输入用户名、密码进行身份验证,如果用户名和密码匹配,则登录成功转向选择日期页面;如果用户名和密码不匹配,则显示登录失败的信息。下面是 Signin. aspx 的后台编码类 SignIn. cs 中最为重要的一个方法 Submit_Click,该方法是页面提交按钮的单击事件所触发的方法,该方法用于完成对用户的身份验证。

```
Private void Svbmit_Click(object sender,
System. EventArgs e)
{
//实例化一个用户对象
TTUser accountSystem =new TTUser();
//调用用户对象的 Login 方法对用户进行身份验证并返回 usrId
  string userId=accountSystem. Login(eMailText.
Text,TTSecurity. Encrypt(passwordText. Text));
//判断用户的 userId 是否为空
if(userId ! ="''&& useId ! =string. Empty){
//如果 userId 不为空,则表明用户通过身份验证
FormsAuthentication. SetAuthCookie(MailText. Text,false);
//转向工作进程页面
RedirectToMobilePage("TimeSheet. aspx");}
else
    //如果 userId 为空,则说明身份验证失败
    signInMsg. Text="Login Failed!";}
```

FoxERP 的移动 Web 表示层例如其用户登录页面:在 SignIn. aspx 的前台页面中呈现的是 ASP. NET 的移动 Wed 应用控件,其代码清单如下。

```
〈%@Register Tagprefix="mobile"
  Namespace="System. Web. UI. MobileControls" Assembly="
System. Web. Mobile, Version=1. 0. 33. 0, Culture=neutral, Pub-
licKeyToken=b0bf5f7flld50a3a"%〉
```

```
〈%@Page language="c#" Codebehind=
"SigeIn. aspx. cs" Inherits="TTMobileCSVS. SignIn" AutoEven-
tWireup="false"%〉
〈meta name="CODE_LANGUAGE"
content="C#"〉
〈meta conetent="Microsoft Visual Studil. NET 7. 0" name=
"GENERATOR"〉 〈meta
content=http://schemas. microsoft. com/Mobile/Page"name="vs-
targetSchema"〉
〈body
Xmlns:mobile=http://schemas. microsoft. com/Mobile/WebForm〉
〈! --使用 mobile:stylesheet 标签引入页面的样式表文件-- 〉
〈mobile:stylesheet id="Stylesheetl"
ReferencePath="Styles. ascx" runat="server"〉
〈/mobile:stylesheet〉
〈! --使用 mobile:Form 表单标签定义表单--〉
〈mobile:Label id="FormSignIn" runat="server" StyleRefer-
ence="Form"〉
〈! --使用 mobile:Label 标签定义 Label 控件--〉
〈mobile:Label id="LabelSignInTitle"
StyleReference="title" Runat="server"〉
Sign-In〈/mobile:Label〉
〈mobile:Label id="Label5" runat="server"〉
e-mail address:〈/mobile:Label〉
〈! --使用 mobile:TextBox 标签定义 TextBox 控件--〉
〈mobile:TextBox id="eMailText"
runat="server"〉〈/mobile:TexBox〉
…
〈! --使用 mobile:Command 标签定义按钮控件--〉
〈mobile:Command id="Submit"runat="server"〉
Submit〈/mobile:Command〉
〈mobile:Label id="signInMsg"
runat="server"〉〈/mobile:Label〉
〈! --结束表单定义--〉
〈/mobile:Form〉
〈/body〉
```

(2)FoxERP 移动 Web 应用的页面多表单机制和状态保持对象的使用

TimeSheet. aspx 是一个含有 4 个表单的移动 Web 页面,在该页面中可以完成日期浏览、查看某一星期内的工作进程安排、修改和删除现有工作的功能。多个表单同一页面内的这种设计模式是在普通 ASP. NET 页面中是没有的,在普通 ASP. NET 页面中数据的传递是通过表单的提交完成的,但是在移动 ASP. NET 页面中多个表单在同一页面内,表单之间的数据交换则可以使用状态保持的对象来完成。在每一次表单的跳转过程中,系统将需要传递和保存的数据暂时存放在状态保持对象中。以下是 TimeSheet. aspx. cs 的代码清单。

```
//声明状态保持对象_entryHoldrer
private TimeEntry _entryHolder
=new TimeEntry();
private TTUser _user;
…
//定义表单名的枚举
private enum FormName{
FormMain,FormDetail}
//页面载入方法
private void page_Load(object sender,
System. EventArgs e){
//首先获得用户对象
_user=new TTUser(TTSecurity. GetUserID(),
User. Identity. Name,TTSecurity. GetName(),
Ttsecuriyt. GetUserRole());
If (! IsPostBack){
//如果页面首次载入则将状态保持对象中的日期设置为当前日
期
_entryHolder. EntryDate=DateTime. Today;
//将隐藏域 HiddenDate 的值也设置成当前日期
HiddenDate. Text=DateTime. Today. ToshortDateString(); }
else {
//如果页面被提交则从隐藏域中获得值
-entryHolder. EntryDate = Convert. ToDateTime ( HiddenDate.
Text);
if(CategoryList. Selection! = null)-entryHolder. CategoryID =
Convert. ToInt32(CategoryList. Selection. Value);
_entryHolder. Description=TTSecurity. CleanStringRegex(En-
tryDescriptionText. Text);
…}}
```

TimeEntryGrid_Select 方法是用户选择查看 objectlist 控件中某一条工作进程的详细信息事件所触发的方法,该方法获得用户所要查看的工作进程的 ID,并将状态保持对象实例化成用户所选择工作进程的对象,最终跳转表单到 FormDetail。

```
Private void TimeEntryGrid_Select
〈object sender,System. Web. UI. MobileControls.
ObjectLixtSelectEventArge e){
//获得用户所要查看的工作进程的 ID
int currentEntryLogID=Convert. ToInt32
〈e. ListItem["EntryLogID"];
//将状态保持对象实例化成用户所选择工作进程的对象
_entryHolder=new TimeEntry
(currentEntryLogID);_entryHolder. Load();
//跳转表单到 FormDetail
Activeform=FormDetail;}
```

2.2 使用 Session 确保系统的安全性

Session 简单来说就是服务器给客户端的一个编号。当一台 WWW 服务器运行时,可能有若干个用户浏览正在运行在这台服务器上的网站。当每个用户首次与这台 WWW 服务器建立连接时,他就与这个服务器建立了一个 Session,同时服务器会自动为其分配一个 SessionID,用以标识这个用户的唯一身份。这个 SessionID 是由 WWW 服务器随机产生的一个由 24 个字符组成的字符串,这个唯一的 SessionID 是有很大的实际意义的。当一个用户提交了表单时,浏览器会将用户的 SessionID 自动附加在 HTTP 头信息中(这是浏览器的自动功能,用户不会察觉到),当服务器处理完这个表单后,将结果返回给 SessionID 所对应的用户。试想,如果没有 SessionID,当有两个用户同时进行注册时,服务器怎样才能知道到底是哪个用户提交了哪个表单。

当 IIS 关闭、重起后,这些信息都会丢失。但是这种模式也有自己最大好处,就是性能最高。因为所有的 Session 信息都存储在了 IIS 的进程中,所以 IIS 能够很快的访问到这些信息,这种模式的性能比进程外存储 Session 信息或是在 SQL Server 中存储 Session 信息都要快上很多。

一般来说,在网站上某一个页面中的变量(指服务器端变量,下同)是不能在下一页中用的,有了 session 就好办了。session 中注册的变量可以作为全局变量使用。这样我们就可以将 session 用于用户身份认证,程序状态记录,页面之间参数传递。

2.3 尽量减少表单回送

每当点击 Web 网页上的 Button、LinkButton 或 ImageButton 控件时,表单就会被发送到服务器上。如果控件的 AutoPostBack 属性被设置为 true,如果 CheckBox、CheckBoxList 等控件的状态被改变后,也会使表单会发送回服务器。

每次当表单被发送回服务器,就会被重新加载,启动 Page_Load 事件,执行 Page_Load 事件处理程序中的所有代码。把网页的初始化代码放在这里是最合适不过的了。我们经常会希望在每次加载网页时执行一些代码,而希望只有在网页第一次加载时执行另一些代码,甚至希望一些代码在除首次加载外的每次加载时执行。

可以利用 IsPostBack 特性来完成这一功能。在网页第一次加载时,该属性的值是 false。如果网页因回送而被重新加载,IsPostBack 属性的值就会被设置为 true。通过测试,可以在任意时候执行指定的代码。

结束语 我们开发的 FoxERP 企业资源计划系统已由成都小狐狸软件公司成功地使用和运行,实践证明其功能不亚于台湾宝盛公司使用的 ERP。它适用于中小型的制造业企业,必将得到进一步的推广,为我国企业信息化做贡献。

参 考 文 献

1 刘甫迎,等. C#程序设计教程[M]. 北京:电子工业出版社,2005

2 叶宏模. 企业资源计划 ERP[M]. 北京:电子工业出版社,2002