

# 区域电子健康档案系统的分布式应用程序 框架模型研究 \*

张远鹏 董建成 耿兴云 吴辉群

(南通大学医学信息学系 南通 226001)

[摘要] 将 .NET 分布式技术应用到区域电子健康档案系统中, 介绍 .NET 平台 4 层分布式应用程序框架, 包括表示层、业务逻辑层、数据访问层和数据实体层, 并具体阐述电子健康档案系统 4 层架构设计与实现。  
[关键词] .NET; 区域电子健康档案; 分布式技术

**Research on Distributed Application Program Framework Model in Regional Electronic Health Records System** ZHANG Yuan - peng, DONG Jian - cheng, GENG Xing - yun, WU Hui - qun, Department of Medical Informatics of Nantong University, Nantong 226001, China

[Abstract] The paper applies .NET distributed technology into regional electronic health records system, introduces four - level application program framework in .NET platform, including presentation layer, business logical layer, data access layer and data entity layer, furthermore concretely elaborates four - level structure design and realization of electronic health records system.  
[Keywords] .NET; Regional electronic health records; Distributed technologies

随着 Internet 和 Web 技术的迅速发展, 基于 Web 的分层构架设计进入了快速发展、应用阶段。本文在研究电子健康档案系统架构时, 采用基于 .NET 分布式技术的 4 层 Web 架构, 这种架构相对于传统的 3 层架构更加合理、高效<sup>[1]</sup>。

## 1 .NET 平台分布式应用程序框架模型

### 1.1 4 层分布式框架模型

[收稿日期] 2010 - 06 - 29  
[作者简介] 张远鹏, 硕士, 助教, 主要研究方向为 .NET 分布式技术, 发表论文 1 篇; 通讯作者: 董建成, 教授, 硕士, 主要研究方向为数字医学。  
[基金项目] 国家社会科学基金“基于云计算理念与技术的文献资源服务研究”(09CTQ011)。

基于 .NET 的分布式应用程序框架模型可分为表示层 (Presentation Layer)、业务逻辑层 (Business Logic Layer)、数据访问层 (Data Access Layer) 和数据实体层<sup>[2]</sup> (Data Entity Layer), 见图 1。

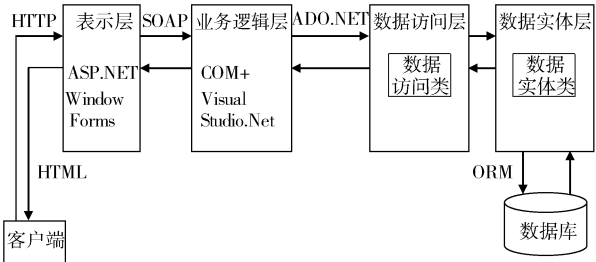


图 1 4 层分布式应用程序框架

表示层包括客户端的软件组件、应用程序、Web 浏览器等, 负责与客户端相关的工作, 接受来自浏览器的 HTTP 请求, 然后返回一个浏览器可以

显示的 HTML 页面。业务逻辑层是分布式应用系统的关键所在,大量的商务逻辑在该层内实现,表示层通过 SOAP 协议与业务逻辑层进行通信。业务逻辑层被打包成组件,通过详细定义的接口进行交互,一般来说主要包括数据管理器、构件库和构架库。数据访问层是分布式应用框架的数据最终提供者,实际数据存储在该层中,一般采用 SQL Server2000, Sybase, Oracle 等,逻辑和数据库层之间的通信使用 ADO.NET 提供的 API 进行。数据实体层是整个分层模型的最下层,用来封装数据,通过对象关系映射 (Object/Relation Mapping, ORM) 框架实现与数据库的映射<sup>[3]</sup>。

## 1.2 4 层分布式结构的优势

与 3 层结构相比,在 4 层结构中将业务逻辑和数据访问完全分离。可扩展性:因为业务逻辑与数据访问处于不同层次结构中,当系统数据对象发生变化时,只要系统的业务没有发生变化,只需要修改或维护数据访问层的访问逻辑即可。当系统的业务有所变化时只需要局部增加、修改或维护业务逻辑层的部分业务逻辑即可,通过每层架构来清晰地分隔代码,允许扩展解决方案,而不需要修改现在的代码。可维护性:区分业务和数据逻辑单元,将它们放在不同的组件中,降低层与层之间的耦合度,使得系统功能更易于扩充,系统的可维护性提高,增强了代码和资源的复用、重用能力,对提高软件开发效率和软件质量具有非常重要的作用。

# 2 电子健康档案系统 4 层架构设计

## 2.1 功能模块

本系统主要划分为系统管理、健康档案管理、预防管理、保健管理、基本医疗管理、康复管理、健康教育管理、计划生育管理 8 个主要功能模块。在设计开发系统过程中,利用 .NET 技术,采用面向对象的设计方法, PnP 即插即用的设计理念,认真分析,精心组织,采用如图 1 所示的 4 层开发模型实现对电子健康档案系统的开发。

## 2.2 表示层

表示层负责与用户进行交互,接受用户的输入并将服务器端传来的数据呈现给用户,该层通过代码隐藏 (code-behind) 方法以 .aspx.cs 文件存放。通过代码隐藏技术可以实现显示逻辑和处理逻辑的分离,对于代码重用、程序调试和维护都具有革命性的意义,同时利用该技术还可以有效保护代码,提高系统的安全性。

## 2.3 业务逻辑层

是整个电子健康档案系统的核心,各个子系统的加工和信息处理都由该层组织实施。该层处于数据访问层与表示层中间,在数据交换中起着承上启下的作用,根据表示层传来的参数,制定业务规则,实现业务流程等。该层的类和数据访问层中的类对应,当业务逻辑层调用数据访问层时,需验证数据的有效性,再调用数据访问层。

## 2.4 数据访问层

负责数据库的访问和数据存取,向业务逻辑层提供一致性的数据服务,是与数据操作有关的接口,其核心技术是 ADO.NET。数据存储和访问模式采用 DataAdapter 和 DataSet 方法<sup>[4]</sup>。将查询结果放入 DataSet 中,采用离线方式进行数据操作,在适当时机连接数据源,将 DataSet 中发生变化的数据更新到数据源,以减轻服务器端连接的资源负担,提高数据库使用率。同时对多应用方式可用 DataSet 产生多个离线的 DataTable,对应多个查询结果提供系统使用,且便于数据成批维护。由于无需在线连接内部数据,可确保内部数据安全。

## 2.5 数据实体层

用于封装实体类数据结构,一般用于映射数据库的数据表或视图,用以描述业务中客观存在的对象。实现过程中,通过 .NET 提供的 System.Data.OleDb 或 System.Data.SqlClient 命名空间中的数据访问与控制类,构造出与具体数据库相应的类,从而实现数据存储层与数据访问层的通信。

### 3 电子健康档案系统 4 层架构的实现

#### 3.1 数据实体层的实现

以用户登录模块为例，采用 C#语言，阐述实现方法和过程。在 SQL2005 数据库中建立一个名为 EHR 的数据库，在 EHR 中建表 Comm\_Worker，表的结构，见图 2。

| 列名               | 数据类型          | 允许空                                 |
|------------------|---------------|-------------------------------------|
| id               | varchar(50)   | <input type="checkbox"/>            |
| user_name        | varchar(50)   | <input type="checkbox"/>            |
| password         | varchar(50)   | <input type="checkbox"/>            |
| real_name        | nvarchar(50)  | <input type="checkbox"/>            |
| department       | nvarchar(500) | <input checked="" type="checkbox"/> |
| region_code      | varchar(50)   | <input checked="" type="checkbox"/> |
| power            | text          | <input checked="" type="checkbox"/> |
| create_time      | datetime      | <input checked="" type="checkbox"/> |
| transfer_in_time | datetime      | <input checked="" type="checkbox"/> |
| sex              | varchar(50)   | <input checked="" type="checkbox"/> |
| nation           | varchar(50)   | <input checked="" type="checkbox"/> |
| native_place     | varchar(50)   | <input checked="" type="checkbox"/> |
| political_status | varchar(50)   | <input checked="" type="checkbox"/> |
| birthday         | datetime      | <input checked="" type="checkbox"/> |

图 2 Comm\_Worker 表结构

在该层中利用 ORM 框架，将数据表转换为类，将表以对象的形式呈现在程序员面前。对该类的操作就要通过数据访问层中封装的方法进行。将表映射为数据实体类的代码如下：

```
namespace ENTITY//数据实现层命名空间
{
    [Serializable]
    public class Worker//数据实体类
    {
        .....

        private string user_name; //将字段映射为私有成员
        public string User_name//用公有方法对私有成员进行读写
        {
            get { return user_name; }
            set { user_name = value; }
        }
        .....
    }
}
```

#### 3.2 数据访问层的实现

数据访问层为业务逻辑层提供数据访问服务，可以把数据库访问操作归纳为以下几种类型：创建记录、删除记录、读取记录并返回实体数据给调用程序、根据调用程序传过来的实体数据更新记录、调用存储过程。图 3 是数据访问层的设计类图，整个层由 DataAccess 类组成。DataAccess 接口定义了访问数据库的通用方法，包括建立数据库连接、关闭数据库连接、释放数据库连接资源、执行非查询语句、执行查询语句并返回数据实体、执行查询语句并返回数据集、获取数据库错误信息等。

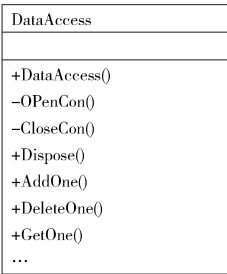


图 3 数据访问层类图

#### 3.3 业务逻辑层的实现

在工程中点击右键，选择“添加 -> 新建项”，在弹出的对话框中选择“类”，并取名为 WorkerBLLDB.cs，在该文件中添加如下代码：

```
namespace BLLDB
{
    public class WorkerBLLDB: DBBase < Worker >
    {
        public WorkerBLLDB ( ) : base ( " Comm_ Worker", " id", null) { }

        public Worker GetWorkerByUserNameAndPassword ( string user_name, string password)
        {
            return GetOne ( " select * from Comm_Worker where user_name = @user_name and password = @password", false, new XDbParameter ( "@user_name", XDbType. VarChar, user_name, SupportDatabase. MSSQL), new XDbParameter ( "@password", XDbType. VarChar, password, SupportDatabase.
```

```
MSSQL)); //调用数据访问层的方法
```

```
    }
}
}
```

在实际应用中业务逻辑层是至关重要的, 承载着整个系统最核心的部分, 在登录模块中业务逻辑层主要承担了以下职责: 对数据访问层进行封装, 使得表示层可以不关心具体的数据访问层; 业务逻辑数据的填充与转换; 核心业务的实现, 在登录模块中一个业务方法对应一个数据访问方法。GetWorkerByUserNameAndPassword 方法就调用了数据访问层的 GetOne 方法。

### 3.4 表示层的实现

在“Web”工程下点击右键, 选择“添加新项→Web 窗体”, 并取名为“Login.aspx”, 见图 4。

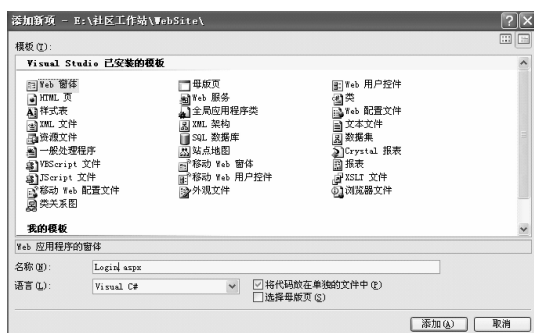


图 4 添加 Web 窗体对话框

在 Login.aspx 文件中, 通过放置 textbox 文本框、button 按钮和 label 控件实现登录表单。在后台的 Login.aspx.cs 文件中, 写入以下代码:

```
protected void Button1_Click ( object sender, ImageClickEventArgs e )
{
```

```
    Worker entity = new Worker (); //创建一个数据实体类
    entity. User_ name = txtUserName. Text. Trim (); //用数据
    实体类的属性来存储从客户端取来的数据
```

```
    entity. Password = txtPassword. Text. Trim ();
```

```
    .....//省略有效性验证代码
```

```
    WorkerBLDDB blldb = new WorkerBLDDB (); //利用表示
    层传来的数据, 制定业务规则
```

```
    Worker worker = new Worker (); //创建一个数据实体
```

类, 用来保存数据库操作的结果

```
    worker = blldb. GetWorkerByUserNameAndPassword ( entity. User_ name, entity. Password ); //通过业务逻辑层调用数
    据访问层
```

```
    if ( worker == null )
```

```
    {
```

```
        ShowMessageBox ( " 账号或密码有误, 请重新输
        入", " Login. asp" );
```

```
        return;
```

```
    }
```

可以看出表示层只是获取客户端传入的数据, 调用业务逻辑层提供的方法。具体的逻辑处理完全由逻辑层的组件负责。在 Visual Studio 2005 中表示层可以通过其提供的可视化编辑工具制作, 而逻辑层可以在任何机器上存在, 两者互不干扰。

## 4 结语

随着 .NET 分布式技术的发展, 基于 4 层架构的系统开发已经成为当今研究的热点, 利用 .NET 4 层架构开发模式, 开发人员可以相对简单地创建灵活性更好、可伸缩性更高、可重用性更强的多层结构的应用程序, 更好地解决软件系统中负载平衡、异地数据访问、容错处理和可维护性等方面的问题<sup>[5]</sup>, 从而真正地提高软件的开发效率和质量。

## 参考文献

- 1 张海藩. 软件工程导论 [M]. 北京: 清华大学出版社, 1992: 59-62.
- 2 Spenser K, Eberhard T, Alexander J. 著, 杜志秀译. Microsoft Visual Basic. NET 面向对象可重用组件开发 [M]. 北京: 清华大学出版社, 2003.
- 3 郭小峰, 姚世军, 尹祖伟. 基于 .NET 的 Web 应用框架的设计与应用 [J]. 计算机工程与设计, 2008, 20 (2), 454-456.
- 4 邱云飞, 邵良彬. 基于 ADO.NET 的通用数据访问层设计方法研究 [J]. 计算机系统应用, 2007, (7), 54-56.
- 5 葛瑶, 李晓风, 王辉. 基于 .NET 分布式系统的模式研究 [J]. 计算机工程与设计, 2008, 29 (5), 1049-1051.