第 32 卷　第 17 期　　计　算　机　工　程　　2006 年 9 月
Vol.32　№ 17　　Computer Engineering　　September 2006
·开发研究与设计技术·　文章编号：1000—3428(2006)17—0262—03　文献标识码：A　中图分类号：TP311

基于 STRUTS 框架的 SRM 系统设计

梅登华，闵华清
(华南理工大学计算机科学与工程学院，五山 510640)

摘　要：介绍了构建基于 Struts 框架的 SRM 系统，给出了系统的结构和总体设计方案，进行了系统功能的划分，给出了系统总体流程和总体框架设计；分析了 J2EE 设计模式下的 SRM 系统实现以及数据与表现的分离问题、SRM 系统 Web 层的权限管理、SRM 系统客户端设计等问题。

关键词：供应商关系管理；J2EE；MVC；Struts 框架

Design of SRM System Based on STRUTS

MEI Denghua, MIN Huaqing
(Computer Science and Engineering College, South China University of Technology, Wushan 510640)

【Abstract】 The thesis sets up the system of SRM that based on the frame of Struts. And it gives a system construction and the total design project, processes the demarcation of the system function, gives the total system process and the total frame designs. It analyzes the system realization and the demarcation between data and presentation, performances the legal power management of SRM system Web layer, client system design of SRM system based on J2EE.

【Key words】 SRM; J2EE; MVC; Struts structure

公司在生产经营活动中，虽然在产能方面并没有发生瓶颈现象，但常常由于计划员、采购员、供应商之间的信息沟通不畅，物料供应协调困难，导致缺料状况时有发生，生产计划也无法得到及时的调整和更新。因而需要解决好计划员、采购员、供应商之间的信息沟通问题，使物料能够得以顺畅的供应。

1 系统的结构与总体设计

1.1 系统功能划分和主体流程

我们将 SRM 系统分为 11 个功能模块，这些模块之间并不是独立的，模块之间的联系如图 1 所示。SRM 系统必须整合企业的关键数据，使得可以利用这些数据实现分析、统计、预测等功能，使得数据在各个模块之间“流动”。系统的流程主要是采购员对由 BAAN 和 SRM 来的订单进行 Approval 和 Release，供应商 Accept 订单，并进行一系列的后续处理。系统运作流程如图 2 所示。

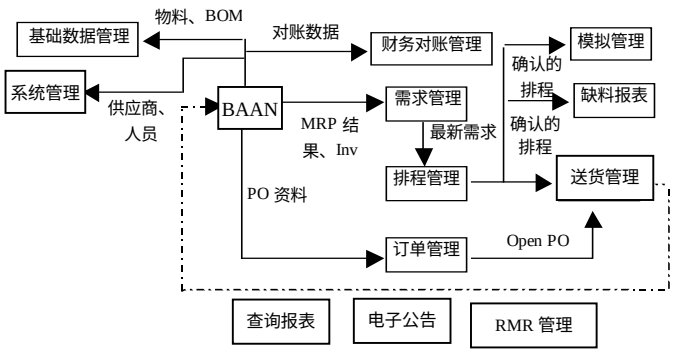


图 1 功能模块图

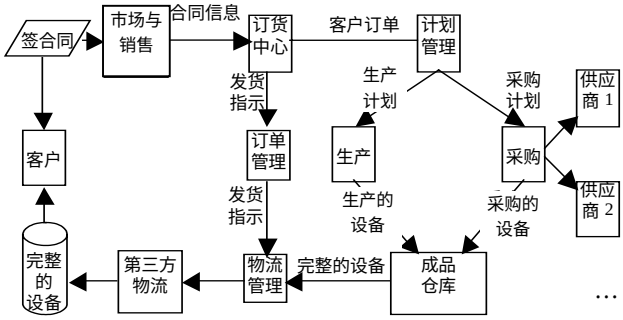


图 2 系统运作流程

1.2 系统的框架设计

从系统功能划分、系统模块之间的关系，以及系统的主体流程，可以知道 SRM 系统的需求复杂，数据流之间关联度很大，如果某一需求发生变化后，会引发其他模块的需求同时变动。因此设计 SRM 系统的程序框架时候，使程序能够最大限度保持稳定性是关键。SRM 作为企业应用，数据是企业的机密，数据安全是企业必须考虑的问题。

Struts 框架具有强大的功能，Struts 技术实现了 MVC 模式的思想，强制分离了界面显示和业务逻辑，可以使在界面显示需求变化时候，业务逻辑的改变大大减少；模型和控制器的划分使得可以通过控制器方便地分发业务逻辑，根据不同的用户显示不同的内容；此外从 JSP 继承的自定义标签可以很容易实现用户权限和数据权限的控制，由此保障了企业的数据安全。

作者简介：梅登华(1967—)，男，副教授、博士后，主研方向：软件可靠性分析，J2EE 架构下软件系统开发；闵华清，教授

收稿日期：2005-12-12　**E-mail：**dhmei@scut.edu.cn

SRM 系统的主体架构如图 3 所示。

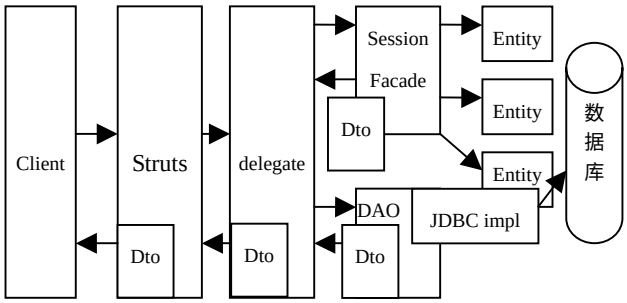


图 3 主体架构

其中 Client 层是 HTML, javascript 代码，用 javascript 仿照 Windows 菜单做出容易上手、容易操作的菜单功能。

而 Web 层是用 Struts 开源框架作为控制器，页面代码上用了大量的 struts 标签和用自定义标签作为权限的控制。这样提供了可复用的代码，在各个页面提供方便的使用。Struts 框架控制器方面也提供很方便复用机制。

虽然 Delegate 是属于 Web 服务器的，但是它与业务层结合比较紧密，一般把它划分到业务层。业务层有两种实现，EJB 和 Without EJB 这两种都在 SRM 系统中有体现。EJB 架构是 Session Bean 作为 Façade(门面)，调用持久层的 Entity Bean。而另外一种架构是直接 Delegate 调用 DAO(数据访问对象)实现业务逻辑。这 DAO 层是由 JDBC 实现的。

以上各层之间传送数据是通过 DTO(数据传送对象)进行数据传输的。采用基于 Struts 的 B/S 模式系统结构：浏览器端，服务器端，请求接收层，请求处理层以及数据存储层。设计 SRM 系统的程序框架如 4 图所示。

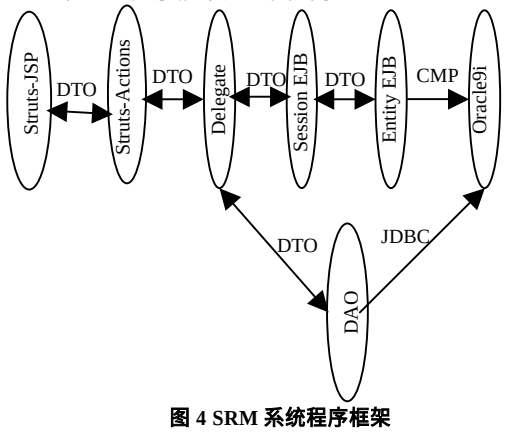


图 4 SRM 系统程序框架

2 J2EE 设计模式下的 SRM 系统实现

SRM 系统从源代码的组织结构看需求的符合型，主要考虑针对用户需求可能的变化的软件代码及构架的最小冗余（同时又要使得系统具有一定的可扩展性）。

模型-视图-控制器 (Model-View-Controller，MVC)设计模式是一种基于请求－响应(Request－Response)模式的应用框架。MVC 模式是 J2EE Web 层的主要实现。

2.1 表现层模式

在 SRM 系统中使用 Struts 框架作为表现层的实现技术，并利用自定义标签作为系统的权限控制部分。下面介绍这两种技术的 J2EE 设计模式。

前端控制器（FrontController），在 SRM 的 Web.xml 配置文件中：

```
<servlet-name>action</servlet-name>
<servlet-class>org.apache.struts.action.ActionServlet
</servlet-class>
        <servlet-mapping >
          <servlet-name>action</servlet-name>
          <url-pattern>*.do</url-pattern>
        </servlet-mapping>
```

用 Struts 的 ActionServlet 类作为前端控制器，所有 url 最后面为.do 的请求都会给 ActionServlet 处理，而它又把请求任务委派给 RequestProcessor(请求处理器)这个应用控制器。

Context 对象（Context Object），使用 Context 对象，按照协议无关的方式封装状态，然后在整个应用系统中使用这种封装后的对象。Struts 里面的 ActionForm 类就是 Context Object。它既提供了普通 java 对象（POJO）的优点，又具有基于表单的策略具有的开发和维护的便利性。并且 Context 对象支持数据验证。利用 Jakarta Commons Validator 框架很容易对 Context 对象进行数据验证。

应用控制器(ApplicationController)，这个模式已经被实现为 Struts 框架的一部分了，Struts 框架通过配置文件中声明来实现映射，从而实现相应的操作并把请求分发到客户端视图。其中 RequestProcessor 就是一个应用控制器。我们只需要开发的 action 调用相应的业务逻辑。而其它的操作由应用控制器完成。

视图助手(View Helper)，把视图和相关的处理逻辑分离开。在 SRM 中采用自定义标签助手策略，来实现权限的控制。这样就可以减少在页面中直接暴露处理逻辑的实现细节。

2.2 业务层模式

业务代表（Business Delegate）：从 SRM 系统的目录结构有一个目录名为 delegate 的目录，这个目录就是 Bussiness Delegate 模式的实现。业务模式借助设计模式、代理(Proxy)和适配器(Adapter)，来实现简单而强大的抽象层。隐藏了寻址(JNDI 查找远程的 SessionBean 或 Entity Bean)，缓存 EJBObject，提高性能。隐藏了远程的访问，为表现层提供一个简单的接口。在 Struts 的 action 中，只是调用业务代表这层的接口就可以完成复杂的业务逻辑。

服务定位器(ServiceLocator)：在 SRM 系统中，ServiceLocator 是一个工具类，在 UtilEJB 的 com.gdnt.util 目录下。它使用了设计模式中的单例(Singleton)模式实现。它提供 EJB 的定位策略和 JDBC 数据源的定位策略。还用缓存（Cache）来保留着以前寻址操作的引用（EJBHome，连接工厂），提高了性能。它封装服务寻址，创建过程的复杂性，在项目中只需要一个 ServiceLocator，就可以为所有的客户(Delegate,DAO)提供服务。这种统一性减少了开发和维护工作量。

会话门面（Session Facade）：SRM 系统中，用无状态 SessionBean 充当 Session Façade 的功能，即存放在 ejb/session 目录下的都是会话门面的实现。它使用了设计模式中的门面(Facade)模式，之所以使用会话门面，是出于两个考虑：一是要控制客户端对业务对象的访问，二是要降低远程客户端和细粒度的业务组件、业务服务交互造成的网络负载。因为 SessionFacade 封装了低层业务组件交互的复杂性，提供暴露统一的粗粒度的接口。它的客户是 Business Delegate，而 Business delegate 是运行在 Web 容器的。SessionFacade 运行在 EJB 容器中。二者通过 RMI-IIOP 进行交互，为提高性能，它们的通信越少越好，因此 SRM 中 SessionFacade 都是粗粒度的，基本都是一个模块一个 Façade。

传输对象(TransferObject)，也叫数据传输对象(Data TransferObject)：SRM 系统中，我们用 DTO 作为各层的数据传送方式。DTO 是一个可串行化（Serialization）的普通 java 对象(POJO)，其中包含若干成员，能够通过一次的方法调用就聚合、传递所有的数据。

传输对象组装器(DTOAssembler)：在 SRM 中用 POJO 来实现传输对象主装器，将 EntityBean 对象组装成为一个 Dto，或者把一系列的 EntityBean 对象组装成一个 Dto 数组。

传输对象(DTO)和传输对象组装器都在 dto 的目录之下。

2.3 集成层模式

借助设计模式中抽象工厂(Abstract Factory)和工厂方法(Factory Method)，可使数据访问对象的创建极具灵活性。

因为 SRM 系统只使用一种持久化存储介质（Oracle9i RDBMS），一般情况下，JDBC 作为 DAO 的实现放在 dao 的子目录 jdbcimpl，DAO 工厂策略就可以。但是如果要切换到另一种底层的存储实现（比如 hibernate 实现的 DAO 放在 dao 的实现 hibernateimpl）抽象工厂策略就带来了很大的灵活性。上层只是针对接口编程，而不是实现。这种 DAO 实现对于上层是透明的。

2.4 数据与表现的分离

真正实现数据和表现分离的是 XML＋XSLT。在这种机制中，XML 和样式表一起送进转化引擎，其中样式描述了 XML 模型中的每个部分在视图中应该如何显示。转化引擎按照样式表中的格式规则匹配 XML 中的相应部分，最后把输入的 XML 转化为另一个适合于显示的 XML 文档。这样也就减少了这些格式和 XML 内容之间的耦合。

3 SRM 系统 Web 层的权限管理

采用 J2EE 设计模式中表现层模式－视图助手。助手采用自定义标签的形式，视图把处理逻辑交给这些助手完成。助手在模型视图和模型间充当设计模式中的适配器。

采用这种策略的好处，即把权限处理逻辑封装在助手（而不是视图）中，能够使应用系统更加模块化，使组件更易于重用。如果把权限处理逻辑放在基于模板的视图中，那么常见的重用办法就只能是把这个权限处理逻辑的代码复制，粘贴到其它地方。这样的复制，将会给系统维护带来困难，因为如果要修改一个 bug，就可能要改动多个文件。

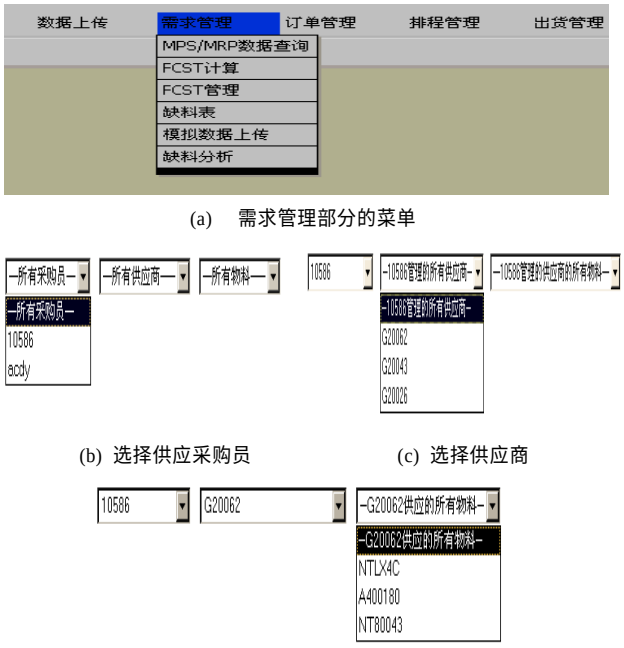
在 SRM 权限处理的标记文件中定义了两个标签 page 和 button。page 标签用在页面的前面，Right 和 function 两个参数是访问该页面所需要的数据权限和功能权限，它的实现在 RightPageTag.java 中，根据 Session 中的用户信息查询该用户是否有足够的权限进入当前页面；如果没有，页面的内容就不会显示出来，而只显示出用户没有足够的权限的提示页面。

Button 标签用在页面按钮的前面，也是提供两个参数来访问该按钮所需的权限，它的实现在 RightButtonTag.java，也是根据 Session 中的用户信息查询该用户是否有足够的权限访问该按钮；如果没有，按钮就不会显示出来。这样用户就不能做相应的操作。

4 SRM 系统客户端设计

传统的 J2EE Patterns 中将表示层完全交给 JSP/Servlet 来做，忽略了浏览器 JavaScript 的能力，这是一个不足的地方。下面介绍在 SRM 系统中所用到的客户端方面的应用。

SRM 的部分界面如图 5 所示。这种网页形式的菜单，跟 C/S 架构的菜单类似，且比一般的导航链接美观。把 javacript 和 css 完美的结合起来。通过利用 JavaScript 和 CSS 建立互联网应用程序或网站的标准化的客户端界面组件，可以使用户一眼就看出来可以进行的操作以及如何完成自己的任务。用户就会对自己的操作更有信心，也不会轻易出现误操作。



(a) 需求管理部分的菜单

(b) 选择供应采购员

(c) 选择供应商

(d) 选择物料

图 5 SRM 的部分界面

对于需求来说，一个采购员(Buyer)管理一个多个供应商(Supplier)，一个 supplier 管理供应多个物料(Item)。因而需要查询任何一个 Buyer 对应哪些供应商，选择这些供应商中的一个，就可以显示该 supplier 所供应的 item。

在用户登录的时候，根据用户代码(UserCode)查询数据库，查出该用户所对应的 Buyer。这个是一对一(OneToOne)关联。根据 Buyer 查对应的 Supplier 和根据 Supplier 查对应的 Item 都是一对多的关系(OneToMany)关联。把所有的数据查出保存在 Session 里面。

实现的方法是设计 3 个数组，第 1 个是一维数组，第 2 个是二维数组，第 3 个是一个三维数组，第 1 个数组保存所有 Buyer，第 2 个数组第一维保存 Buyer 下标，第二维 Supplier 下标，也就是说第 2 个数组保存 Buyer 与 Supplier 的关联，第 3 个数组保存 Buyer、Supplier 和 Item 之间的关联，这样就实现了连动功能。这只是提及 javascript 如何创建级联，没有考虑到这些数据是在 jsp 中的，要获得数据就要进行 javascript 与 jsp 数据的交互。

参考文献

1 Allen P, Bambara J. Sun Certified Enterprise Architect for J2EE Study Guide[M]. McGraw-Hill/Osborne, 2003.

2 Ambler S W. Mapping Objects to Relational Databases[Z]. O/R Mapping In Detail, http://www.isys.uni-klu.ac.at/ISYS/Courses/03SS/eva/vounterlagen/5-XAmbler2000, 2002-07.

3 Fowler M. Analysis Patterns Reusable Object Models[M]. Addison Wesley, 1997.

4 刘 宇, 胡建平. 基于 J2EE 技术的政府采购系统的设计与实现[J]. 计算机工程, 2005, 31(8): 195-198.

5 Allamaraju S. J2EE 编程指南(1.3 版)[M]. 北京：电子工业出版社, 2002.