第35卷 第14期 计 算 机 工 程 2009年7月
Vol.35 No.14 Computer Engineering July 2009
·安全技术· 文章编号：1000—3428(2009)14—0145—03 文献标识码：A 中图分类号：TP309

可扩展系统中基于RBAC模型的访问控制

周锦程[1,2]，张佳强[1]，冷文浩[3]
(1. 江南大学信息工程学院，无锡 214122；2. 黔南民族师范学院数学系，都匀 558000；3. 中国船舶科学研究中心，无锡 214082)

摘 要：可扩展信息系统的建设可运用领域工程需求方法，将领域对象从系统中分离出来形成领域对象向导，系统管理员可根据向导完成领域对象的建立，从而实现系统的动态可扩展功能，而扩展的系统应能实现对用户的安全访问控制。通过分析基于角色的访问控制(RBAC)模型，在Struts, Spring和Hibernate集成框架下，给出一种全方位可扩展的动态自定制信息系统中基于RBAC模型的访问控制模块的实现方法，论述模块的后台数据库设计和模块实现等关键技术。

关键词：基于角色的访问控制；构件；J2EE框架；可扩展

Access Control Based on RBAC Model in Extensible System

ZHOU Jin-cheng[1,2], ZHANG Jia-qiang[1], LENG Wen-hao[3]
(1. School of Information Technology, Jiangnan University, Wuxi 214122; 2. Department of Mathematics, Qiannan Normal College for Nationalities, Duyun 558000; 3. China Ship Scientific Research Center, Wuxi 214082)

【Abstract】 Extensible information system can use the domain project demand analysis method to separate the domain object from the system to produce the domain object guide. System manager can establish the domain object according to this guide to implement the system's dynamic expansion, and the expansion system can implement the security access control. Through analyzing the Role-Based Access Control(RBAC) model, based on the integration framework of Struts, Spring and Hibernate, this paper proposes an access control scheme based on RBAC model, suitable for those dynamic information system, which can be Omni-directional expansion. And some relative key technologies for database design and module implementation are presented.

【Key words】 Role-Based Access Control(RBAC); component; J2EE frame; extensible

企业在使用信息系统的过程中，往往因环境或服务的变化而要求系统能为其提供动态扩展支持的功能，因此，开发适合企业自身需求、全方位可扩展的动态自定制信息系统，能使企业方便地根据自身的业务特点，迅速创建或修改所需的信息子库，为相关企业、单位、部门或者人员提供信息服务。而扩展的每一子库对各类用户提供的功能可能不尽相同，因此，须在扩展的系统中实现对用户的安全访问控制。

1 Struts+Spring+Hibernate框架

选择一个优秀的软件架构不仅能使软件开发有章可循、结构清晰、缩短开发周期，还能提高软件的可扩展性和可维护性。J2EE构件集成了先进的软件体系架构思想，具有采用多层分布式应用模型、基于组件并能重用组件、统一完善模型和灵活的事务处理控制等特点。

在J2EE中，Struts+Spring+Hibernate(以下简称SSH框架)进行整合开发是当前最受欢迎的框架。其基本业务流程如下：通过JSP页面实现交互界面，负责传递请求(request)和接收响应(response)；在服务器端表示层引入Struts，根据Struts ActionServlet接收到的request，委派相应的Action；在业务逻辑层中，管理服务组件的Spring IoC容器负责向Action提供业务模型(module)组件和该组件的协作对象数据处理(DAO)组件来完成业务逻辑，并提供事务处理、缓冲池等容器组件提升系统性能和保证数据完整性；而持久层依赖于Hibernate的对象化映射和数据库交互以处理DAO组件请求的数据，并返回处理结果[1]。

2 基于角色访问控制的RBAC模型

2.1 访问控制

访问控制是通过某种途径显示的准许或限制访问能力及范围，从而限制对目标资源的访问，防止非法用户的侵入或合法用户的不慎操作所造成的破坏[2]。目前流行的访问控制模型有自主访问控制模型(Discretionary Access Control, DAC)、强制访问控制模型(Mandatory Access Control, MAC)和基于角色的访问控制模型(Role-Based Access Control, RBAC)。自主访问控制是访问控制技术中最常见的一种方法，允许资源的所有者自主地在系统中决定可存取其资源客体的主体，此模型灵活性很高，但安全级别相对较低；强制访问控制是主体的权限和客体的安全属性都是固定的，由管理员通过授权决定一个主体对某个客体能否进行访问。无论是DAC还是MAC都是主体和访问权限直接发生关系，根据主体/客体的所属关系或主体/客体的安全级别来决定主体对客体的访问权，它的优点是管理集中，但其实现工作量大、不便于管理，不适用于主体或客体经常更新的应用环境。RBAC是一种可扩展的访问控制模型，通过引入角色来对用户和权限进行解耦，简化了授权操作和安全管理，它是目前公认的

作者简介：周锦程(1981－)，男，助教、硕士研究生，主研方向：企业应用集成，数据库技术；张佳强，硕士研究生；冷文浩，研究员、博士生导师、博士

收稿日期：2009-01-15 **E-mail：**guideaaa@126.com

解决大型企业的统一资源访问控制的有效访问方法，其 2 个特征是：(1)减小授权管理的复杂性，降低管理开销；(2)灵活地支持企业的安全策略，并对企业变化有很大的伸缩性。

2.2 RBAC 模型的基本思想

在 RBAC 模型中，角色是实现访问控制策略的基本语义实体。系统管理员可以根据职能或机构的需求策略来创建角色、给角色分配权限并给用户分配角色，用户能够访问的权限由该用户拥有的角色权限集合决定，即把整个访问控制过程分成 2 步：访问权限与角色相关联，角色再与用户关联，从而实现用户与访问权限的逻辑分离。

2.3 RBAC 基本模型

RBAC 基本模型包含了 RBAC 标准最基本的内容，该模型的定义如图 1 所示。

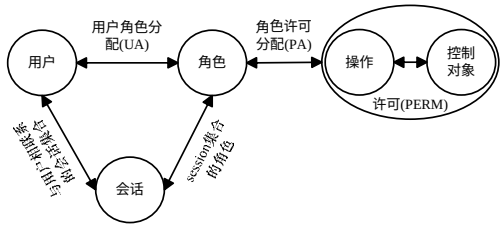

图 1 Core RBAC 模型

RBAC 基本模型包括 5 个基本数据元素：用户，角色，资源，控制，授权。它们之间的关系如下：用户被分配一定角色，角色被分配一定许可权，会话是用户与激活的角色集合之间的映射，用户与角色间的关系定义和角色与权限间的关系定义无关。

3 访问控制实现

采用 RBAC 模型最大的好处是分离了用户和权限，使管理员可以分别处理用户的授权和权限的划分。这种面向对象的权限分离思想使开发出来的访问控制模块的通用性和可复用性更强，最大限度地避免了开发人员的重复性工作[3]。在开发系统时，可以将访问控制作为一个独立的模块进行开发。

3.1 访问控制模块系统设计

上面的分析表明 RBAC 基本模型模型能满足可扩展的动态自定制信息系统中的访问控制，由此可将访问控制模块分为以下 2 个模块：

(1)系统管理模块：包括用户管理模块和角色及权限管理模块 2 个部分，主要完成用户增减、角色增减及其权限分配。

(2)身份认证模块：通过用户名、密码确认用户身份并对其访问某一资源的某一功能进行认证。

3.2 访问控制模块体系结构设计

访问控制模块的体系结构设计采用前面介绍的 J2EE 架构下的 SSH 框架，划分为表示层，业务逻辑层，数据持久层和数据源层，结构如图 2 所示。

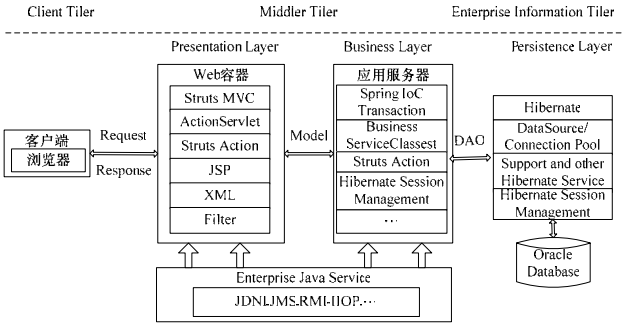

图 2 访问控制模块体系结构

3.3 访问控制模块数据库设计

为了实现基于 RBAC 的访问控制模型，将角色、用户、角色和用户的关系、角色和权限的关系以及可控资源信息等保存到数据库中，数据库设计上采用下面 4 个权限控制相关的表，如表 1~表 4 所示，各表说明如下：(1)角色信息表：存储系统角色集，随角色的添加与删除动态变化。(2)用户信息表：存储系统中的个体用户集及用户与角色的关系，随用户的添加与删除动态变化。(3)角色权限信息表：存储角色对应的权限信息，随角色及对应权限的变化而动态变化。(4)资源表：系统可控制的资源，即由系统管理员创建的动态子库。在表 3 中，若对某子库的操作具有对应的权限，则存储该子库编号，否则存储 0。

表 1 角色表(USER_ROLE)

字段名	数据类型	中文名	说明
ID	数字	角色编号	自动编号
RoleName	文本	角色名称	唯一
Description	文本	角色描述	

表 2 用户信息表(USER_INFO)

字段名	数据类型	中文名	说明
ID	数字	用户编号	自动编号
EnterpriseCode	文本	所属企业编号	外键
UserName	文本	用户姓名	
UserAccount	文本	用户账号	
Password	文本	用户密码	
UserValidity	文本	授权有效期	
UserRoleId	数字	角色 ID	外键

表 3 权限表(USERROLE_RIGHT)

字段名	数据类型	中文名	说明
ID	数字	编号	自动编号
ViewData	数字	子库信息浏览权限	见说明
AddRecord	数字	子库信息修改权限	同上
ModifyRecord	数字	子库信息录入权限	同上
DelRecord	数字	子库信息删除权限	同上
UploadData	数字	子库信息导入权限	同上
DownloadData	数字	子库信息导出权限	同上
UserRoleId	数字	角色 ID	外键

表 4 资源表(DB_LIST)

字段名	数据类型	中文名	说明
ID	数字	编号	自动编号
DbName	文本	自定义库名称	
DbSort	数字	自定义库类别	外键
Description	文本	自定义库介绍	

3.4 访问控制模块实现的关键技术

3.4.1 角色管理模块

本模块将介绍角色的定义及授权的实现，其操作界面如图 3 所示。

图 3 角色定义及角色授权界面

系统对每种资源设置了 6 种功能：浏览(含查询)，新增，修改，删除，批量导入和批量导出数据，如果还有额外的功

能，也可很方便地扩展。

(1)角色管理模块表示层设计

表示层基于 Struts 框架，在表示层的实现中，主要注意以下 2 个方面：

1)由于所控资源是动态变化的，因此各功能的动态复选框组采用 Sturts 的<html:multibox>标签来实现，每个<html:multibox>标签在 ActionForm 中定义 2 种状态的数组并分别对应该标签的全部选项值和选中值，使用时须为每一功能的 2 个数组初始化。在本模块实现中，将所有复选框的全部选项值初始为该动态子库的编号，而选中值初始为 0，数组的大小为资源表中子库的个数。在 JSP 页面中用<logic:iterate>标签迭代与子库数目相等的所有权限并结合<html:multibox>标签创建相应选项，同时用<bean:write>标签创建<html:multibox>标签对应处要显示的文本信息。

2)授权浏览以外的权限时须先选中浏览权限，因此，授权前只有浏览权限为可选状态，当勾选浏览权限时同时将该资源的其他权限置为可选；取消浏览权限时去除该资源的其他已选中的相关权限并将该行资源除浏览以外的权限置为不可用，此功能在 JSP 页面采用 JavaScript 控制。

(2)角色管理模块业务逻辑层设计

业务逻辑层基于 Spring 框架，该框架可将服务器端表示层与业务逻辑层乃至持久层之间的耦合度降到最低，业务逻辑层是应用实现的关键，需要完成的工作包括提供 DAO 与数据库交互，提供方法完成业务逻辑校验、使用 DAO、完成其他业务应用等。在角色的定义及授权实现中，通过各业务逻辑接口提供的相应方法调用数据访问类实现角色及其对应权限信息的保存，其业务逻辑流程如图 4 所示。

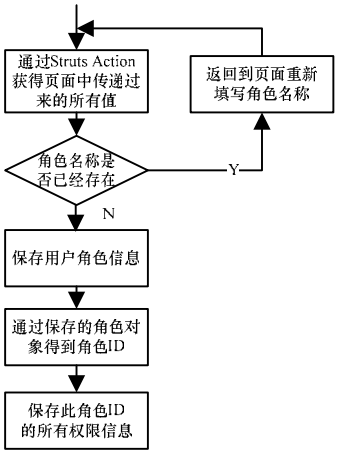

图 4　实现保存角色及其权限信息的业务逻辑

(3)角色管理模块持久层设计

持久层基于 Hibernate 框架，为业务逻辑层提供服务。在持久层需要做的是建立数据库表与域对象的映射关系，书写持久化类接口(DAO)及其实现类(DaoImpl)。数据库访问使用 DAO 模式，抽象和封装对数据源的操作，此外，Hibernate 的配置能实现数据库的高移植性，只需要改变 Hibernate 配置文件，不用修改程序就可以完成不同数据库之间的更换。

3.4.2　用户管理模块

在新增用户时，只须为其选中相应的角色，该用户就具有了此类角色所配置的权限。

3.4.3　身份认证模块

用户在客户端提交账号和密码等信息，进行身份认证，认证通过后，若为普通用户，则将该用户对应角色所具有的权限保存在 Session 中，触发某子库的某功能时，将 Session 中当前普通用户的所有权限取出，如果此用户的权限信息中该功能的权限值与所要访问的数据库编号相同，则用户对此库具有此功能的操作权，否则拒绝其操作。

算法 1 判断用户是否有某数据库的某操作权限

```
输入：long no(要访问的子库编号)
String operator(要访问的子库的功能名称)
输出：若此子库对应功能的操作权，返回 true 否则返回 false
boolean CheckUserRight (long no, String operator) {
List userRoleRightList = new ArrayList();
//获得当前普通用户保存在 Session 中的所有权限
UserRoleRightList= (List) request.getSession ().
getAttribute ("USERROLERIGHTLIST");
boolean flag = false;
for (int i = 0; i < userRoleRightList.size(); i++) {
//子库编号与 Session 中存放的权限中 Operator
//项的值相同，则用户有此库的 Operator 权
if(no==(UserRoleRight)userRoleRightList.get(i).getOperator())
{flag = true; break;}
}
return (flag);
}
```

相关 JSP 页面上通过 Struts 的<logic:notEqual>标签中的变量值(变量定义为对应的功能)是否等于 0 来实现是否显示执行某个功能的图标或者按钮。

3.4.4　资源控制模块

该模块主要负责用户自扩展的动态子库的实现及管理，这里假设把领域建模的任务交给系统管理员，根据系统领域对象生成向导让其完成系统资源的扩展，如动态扩展子数据库和子数据库类别。在子数据库的实现中，可以通过如图 5 所示的方法添加某子库的数据库表的字段信息，在执行保存此信息的过程中创建数据库表，同时结合 SSH 框架为此数据库表自动生成相应的实体域类文件、O-R 映射文件，表单域类文件并自动添加相应的配置到 Spring 容器中以统一进行业务对象的管理。此外，还需为其编写相应的通用 Action 方法和 JSP 页面以实现系统的各种功能。

图 5　船舶术语库表的部分界面

4　结束语

本文提出的访问控制模块采用基于 J2EE 框架下的 SSH 架构实现，该方法已成功应用于某可扩展的信息管理平台和某可扩展的中小企业局项目服务平台中，效果良好，从应用结果看出，在该方法在不同的可扩展动态信息系统中，只需做简单的修改就能很方便地使用，具有一定的实用价值。

(下转第 150 页)