CincoSecurity: A Reusable Security Module Based on Fine Grained Roles and Security Profiles for Java EE Applications

María Consuelo Franky
Department of Systems Engineering
Pontificia Universidad Javeriana
Bogotá, Colombia
lfranky@javeriana.edu.co

Victor Manuel Toro C.
Department of Systems and Computing Engineering
Universidad de los Andes
Bogotá, Colombia
vm.toro815@uniandes.edu.co

***Abstract*— Almost every software system must include a security module to authenticate users and to authorize which elements of the system can be accessed by each user. This paper describes a reusable security software module that follows the Role Based Access Control model (RBAC), but implementing fine grained roles and grouping them into "security profiles". This leads to a great flexibility to configure the security of an application by selecting the operations allowed to each profile, and later, by registering the users in one or several of these profiles. The security module has been designed and developed to be the initial code baseline for the development of any Use Cases oriented Java EE system, offering from the beginning a flexible, extensible and administrable access control to the elements of the application.**

Keywords-Security; Access control; RBAC; Framework; Java EE; Seam.

I. INTRODUCTION

This paper summarizes the experience of the authors designing and developing a reusable security module, called CincoSecurity, that has been used for several years to control access to the elements of web applications written in Java Enterprise Edition (J2EE initially [8] and later Java EE 5 [9]). Currently, the module CincoSecurity is available [17] under the GPL license, and is used by some software houses in Colombia.

CincoSecurity implements a RBAC (Role-Based Access Control) [6], providing high flexibility to control access to the various elements of a Web application, such as the invocation of an operation of a business component, the access to a web page, and the access to elements within that page. The innovation of CincoSecurity is the use of very fine grained roles, each role having a single permission associated with the invocation of an operation of a business component. From these fine roles —which can be directly controlled by the application server— CincoSecurity allows to define "security profiles" as sets of fine grained roles. This facility of security profiles gives a great flexibility for configuring the security of an application by selecting the operations allowed to each profile, and later, by registering the users in one or several of these profiles.

Any Java EE web application that is to be constructed with the Seam framework [11] gets the following benefits by integrating the CincoSecurity module: user authentication, registration in the application server of the fine roles derived from the security profiles the user belongs to, and dynamic construction of a personalized menu containing only the entries leading to the use cases allowed for the user. Additionally, CincoSecurity contributes to the application being constructed with use cases to administer the security profiles, to manage user registration in these security profiles and to administer passwords. Additionally, CincoSecurity comes with use cases to register new modules, new use cases and new services, as they become available during a development project, for their security to be administrable.

In the following section, this paper presents the RBAC security model on which the CincoSecurity module is based, and more specifically, the RBAC model applied to the context of Java Application Servers. Then, additional concepts provided by the CincoSecurity module are introduced, as well as its entities model. Later, there is a description of the use cases coming with the CincoSecurity module (e.g., create a new user, create/edit a security profile, add/delete users from a security profile, etc.). At the end, the paper provides a short summary and references to the detailed guidelines [18] for integrating the CincoSecurity module to a Java EE application built with the Seam framework [11]. Finally, there is a comparison with other works, followed by the conclusion and a short description of our future work.

II. EVOLUTION OF THE RBAC SECURITY MODEL

The RBAC model introduced the concept of "role" to control the access to computing resources. The RBAC term was first proposed by Ferraiolo and Kuhn [2], based on previous works of Baldwin [1]. The initial proposal of this model creates a role for each type of job within an organization (cashier, customer service person, office director, ...). Then, each role is assigned with the set of access permissions that are required for this type of job. Finally, each user is enrolled into one or more roles (rather than to specific permissions). This model simplifies security management because the roles (with their associated permissions) tend to be stable, and users can be added or retired easily from roles. The RBAC model allows reinforcing the "least privilege" principle by giving each user

 ISBN: 978-1-61208-124-3

the minimum set of permissions required to perform his work, by enrolling him only in the appropriate roles [6].

From the initial RBAC model (called Core RBAC) the work of Sandhu and colleagues [3] defined extended models, such as the hierarchical RBAC (to include role hierarchy with inheritance of permissions), and constrained RBAC (to prevent, for example, to assign a user to 2 conflicting roles, or to restrict the time slot in which a user can use the permissions of one of its roles).

The main applications of the RBAC model have been in Data Base management Systems, enterprise security management systems, and Web applications that run on application servers [5][6][7].

The wide spread of RBAC models, implemented in numerous products from many providers, led to define an ANSI standard [4] in 2004, aiming to standardize terminology, promote its adoption and improve productivity. However, the current RBAC ANSI standard (consisting of a reference model and a functional specification) has some limitations and gaps as indicated in the work of Bertino and colleagues [7].

III. THE RBAC MODEL APPLIED TO JAVA EE APPLICATION SERVERS

Since the late 90's, the emergence of Application Servers brought a new way to build web applications (both in the enterprise Java platform and in Microsoft .NET), with business components managed by containers that provides added services for security, transaction management, parallelism, pool of connections, logging, etc. [8]

Regarding security, Java EE Application Servers [9] implement the Core RBAC model [6] to control the access to resources based on the roles the user belongs to. In order to take advantage of these security services (and not to write code in the application to internally control access to resources), it is necessary to specify the roles of the application, the association of resources to roles, and the association of users to roles.

A. Enrolling users in roles

In a Java EE application that uses a database to store the authentication and authorization information, the following entities EJB3 (Enterprise Java Beans - version 3) are required [10]:

- An entity "User" shall be implemented (with its corresponding support table in the database), to store users and passwords.
- Entities shall be implemented (with its support tables in the database) to specify the association of each user with one or more roles.
- A "User management" use case shall be implemented to enroll a user in one or more roles.

These facilities are included in CincoSecurity. Similarly, it is also possible to manage users, passwords and roles in a LDAP (Lightweight Directory Access Protocol) server.

B. Controlling access to resources

Seam is a framework to develop Java EE applications, that is being developed by JBoss since 2005, whose principal author is Gavin King [11][13]. Seam allows to directly expose and use in the Web layer the entities and business components of the application. This simplifies enormously the development by eliminating the intermediaries and conversions between the layers of the application. Seam has been widely accepted and has been incorporated in the recent Java EE 6 standard, under the name of "Web Beans".

To control access to resources in a Java EE application that uses the Seam framework, the following strategies are required [12]:

- An annotation is used to protect each method of the session EJB3. This annotation indicates which roles are authorized to invoke the method.
- The url of each JSF (JavaServer Faces) web page can be protected in the navigation flow descriptor (pages.xml) so that it can be accessed only by users belonging to one of the specified roles.
- Each button or element of a JSF web page can be protected so that it is rendered only to users belonging to one of the specified roles.

C. User authentication

In a Java EE application that uses Seam, the authentication service must be specified in the descriptor components.xml. This service shall be a method of a class of the application, and must implement a query in JPQL (Java Persistence Query Language) [10] to verify the user's password (alternatively this process can also be performed with a LDAP server).

Additionally, the authentication service must also obtain the roles of the authenticated user. With the Seam component called "Identity" these roles can be added to the session and informed to the application server.

D. Controlling access to a JSF page

The application server verifies if the user belongs to a role allowed to access the requested JSF page, and if so, the page is displayed. Similarly, inside the page only the elements that the user is authorized to see are shown (elements such as buttons and text boxes can specify with the attribute "rendered" what roles can see them).

E. Authorizing an action from a JSF page

In a JSF page a button's action is typically associated with the invocation of a method of an session EJB3. The server verifies that the user roles allow him to invoke the associated method (assuming that the method is protected by an annotation indicating the roles that can invoke it).

IV. ADDITIONAL CONCEPTS IMPLEMENTED BY THE CINCOSECURITY MODULE

In addition to the security concepts of Java EE applications that use the Seam framework [11][12], the CincoSecurity module implements additional concepts to provide greater flexibility to define the permissions for user.

A. Use case and services

Definition: A use case is a system's capacity to deliver a useful and indivisible result to the user.

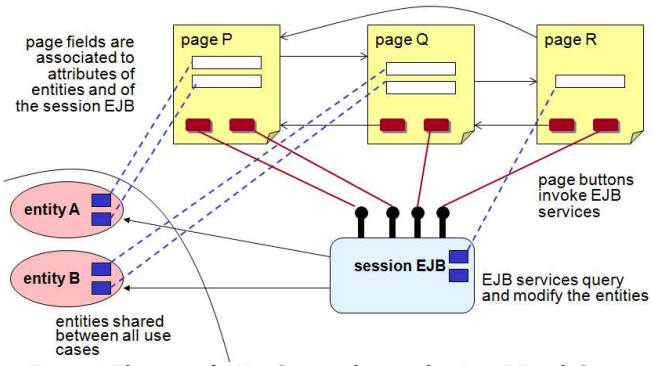

Figure 1: Elements of a Use Case implemented in Java EE with Seam

Definition in terms of its implementation in Java EE with Seam (see Figure 1): a use case consists of one (or more) business entities and a group of services (actions) that act upon them. These actions are implemented as methods of a session EJB3. One or more JSF pages display attributes of the entities involved in the use case, and attributes of the session EJB3 controlling it. In those JSF pages there are actions that invoke the services of the session EJB3. These EJB3 methods are programmed in terms of queries and modifications to the persistent business entities. The navigation flow descriptor contains rules to decide the next page to display.

B. *Module*

Definition: A module is a set of related use cases. The CincoSecurity module comes with the following use cases, that will be explained below: security profiles management, users management, change of password, basic security reports, registration of menu entries, and registration of modules, use cases and services.

C. *Fine grained roles*

The CincoSecurity module works with fine grained security roles:

- A role for entering to each use case. The name assigned to this role is the same name of the use case (which is also the Seam name of the session EJB3 that supports the use case).
- A role to invoke each service within a use case (i.e., each of the methods of the session EJB3 that supports the use case). The name assigned to this role is "use case name"_"name of the method".

D. *Protection of resources*

- Each session EJB3 that supports a use case is protected with an annotation indicating the role for entering to the use case. For example, the profileGestion use case is supported by the session EJB3 ProfileGestionAction.java (which implements the interface ProfileGestion.java); this EJB3 has the Seam name "profileGestion". Consequently, the role for entering to this use case is called "**profileGestion**" and the EJB3 class will have the following annotation:
  @Restrict("#{s:hasRole('profileGestion')}")
- Each service (method) of the session EJB3 is protected by an annotation indicating the role associated with the service. The name of this role is the concatenation of the use case name with the name of the service (with "_" between). For example, the **update** method of the session EJB3 that supports the use case **profileGestion** will have the following annotation:
  @Restrict("#{s:hasRole('profileGestion_update')}")
- Access to each JSF page of a use case is protected in the navigation flow descriptor by the role for entering to the use case. For example, the page **profiles.xhtml** of the use case **profileGestion** has a navigation flow descriptor called **profiles.page.xml** which contains the following restriction:
  <restrict>#{s:hasRole('profileGestion')} </restrict>
- Each button in a JSF page should be displayed only to users with the role associated to invoke the action of the button, which corresponds to an EJB3 service (method). For example, the **profiles.xhtml** page of **profileGestion** use case contains a button whose associated action is to invoke the **update** method of the session EJB3 that supports the use case. Consequently the button tag indicates that only it is showed to the role **profileGestion_update**:

```
<h:commandButton id="update"
  value="Update"  styleClass="button"
  action="#{profileGestion.update}"
  rendered="#{s:hasRole('profileGestion_update')}" />
```

E. *Security profile*

A security profile is a set of fine roles, each fine role expressing the right to invoke a service belonging to a use case. Unlike the role, the concept of security profile is not supported directly by application servers and must be built with additional entities.

The use cases of the CincoSecurity module allow the association of users to roles via security profiles:

- A user can be enrolled in one or more security profiles, so he/she will have the set of fine roles allowed by the union of these profiles.
- There is a *many-to-many* relationship between users and security profiles.
- There is a *many-to-many* relationship between security profiles and roles.

F. *Actions after a user authentication*

After a user is authenticated, CincoSecurity calculates all the fine grained roles from the security profiles the user belongs to, and informs them to the application server (by assigning these roles to a Seam component called "Identity"). Additionally, the following actions are performed by a EJB3 Login:

- The session timeout is set, according to the parameters stored in the database.
- The user's menu is built, containing only the entries leading to use cases allowed to the user.
- The security information of the user is added to the session context, should the application logic needs it.

 ISBN: 978-1-61208-124-3

It is important to remark that the access to use cases not authorized to a user by any profile is prevented in two ways. From one side, not authorized use cases do not appear in the user's menu. From the other side –even if the user types in the url of a not authorized use case– the application server throws a security exception because the fine role for entering to this use case was not included in the list of fine roles that was informed to the application server.

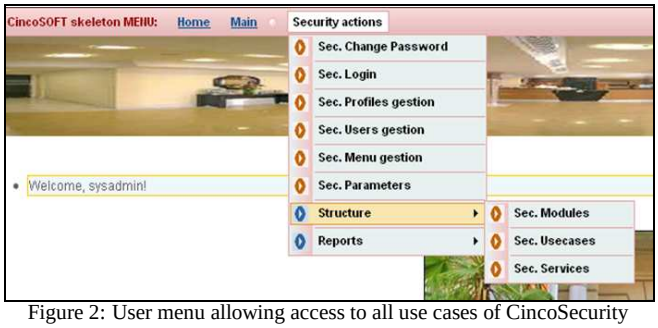

Figure 2: User menu allowing access to all use cases of CincoSecurity

The screen snapshot of Figure 2 shows the menu of a user that is enrolled in security profiles allowing access to all the use cases of the CincoSecurity module.

Figure 3: User menu allowing access to fewer use cases of CincoSecurity

The screen snapshot of Figure 3 shows the menu of another user that is enrolled in security profiles allowing access to just a few use cases of the CincoSecurity module.

V. ENTITIES MODEL OF THE CINCOSECURITY MODULE

The entities model shown in Figure 4 illustrate the relationship one-to-many from Module to Usecase, and from Usecase to Service.

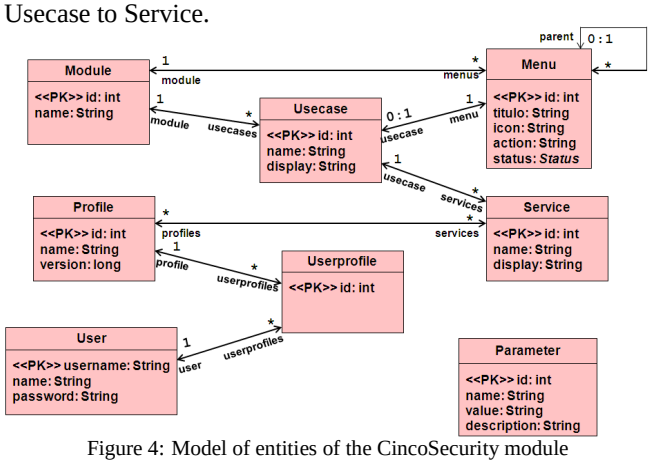

Figure 4: Model of entities of the CincoSecurity module

Figure 4 also illustrates the relationship many-to-many between Profile (security profile) and Service, as well as between Profile and User (via the intermediate entity Userprofile).Each menu entry may have submenus (only terminal menu entries have an action for going to the entry page of a use case).

The system parameters are arbitrary. They can be used, for example, to record the session timeout, the path of the directory to store reports, the address of printers, etc.

VI. USE CASES OFFERED BY THE CINCOSECURITY MODULE

The following are the use cases offered by the CincoSecurity module:

A. *CRUD Use cases*

The CincoSecurity module offers:

- A use case to list/add/edit and remove **parameters** of the application.
- A use case to list/add/edit and remove **modules** of the application.
- A use case to list/add/edit and remove the **use cases** of a module.
- A use case to list/add/edit and remove the **services** of an application's use case.
- A use case to list/add/edit and remove **menu entries**.

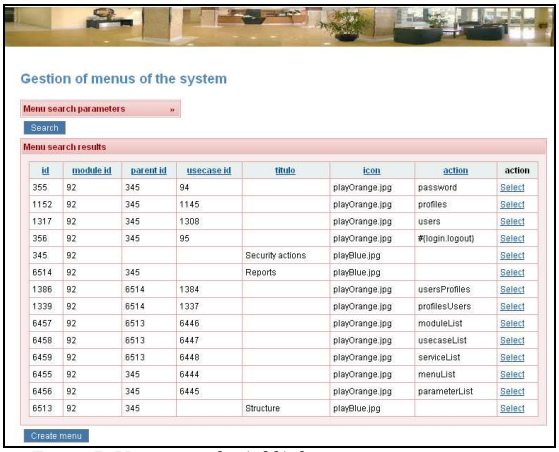

Figure 5: Use case to list/add/edit or remove menu entries.

It can be easily specified which menu entries have a submenu, as well as the use case associated with a terminal entry (see Figure 5).

B. *Management of security profiles*

This use case allows to add/edit/remove security profiles. Initially, the existing security profiles are listed. When a security profile is selected, the modules, use cases and allowed services are shown, so that the user can check or uncheck services (see Figure 6 in the next page).

Similarly, the user can create a new security profile. In this case, the system displays all modules, and within it, the use cases and services, for the user to select those allowed by the new profile.

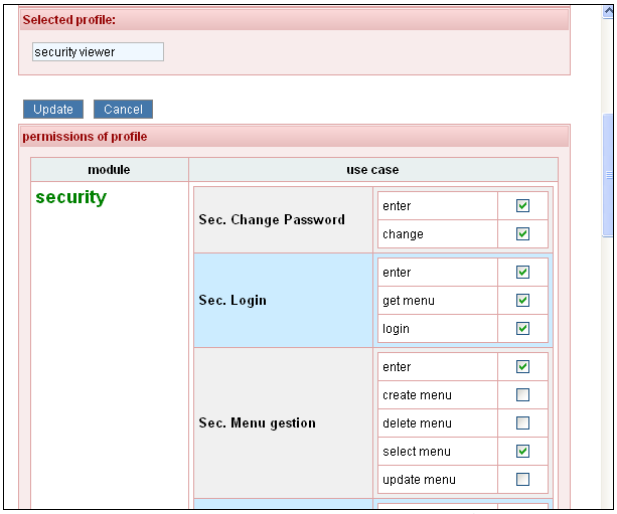

Figure 6: Use case to manage security profiles

C. *Management of users*

This use case allows to add users of the application, indicating its name, login and password. It also allows to enroll the new user in one or more security profiles.

D. *Password change*

This use case allows a user to change his password. Passwords are stored encrypted.

E. *Report of security profiles vs users*

This use case reports, for each security profile, which users are enrolled.

F. *Report of users vs security profiles*

This use case reports, for each user, in which security profiles is enrolled.

VII. HOW TO INTEGRATE THE CINCOSECURITY MODULE TO A JAVA EE SEAM APPLICATION

The CincoSecurity module is open source with GPL License [16]. It can be downloaded from SourceForge [17] in the form of an Eclipse project [15]. It comes ready to be deployed on the application server JBoss [14], but can be installed in any other Java EE application server by following the guidelines of the Seam manual [13].

The documentation accompanying the CincoSecurity module explains in detail how to deploy and execute the module, and how to integrate it with a Java EE application built with Seam. In particular, detailed explanations are included for registering application's use cases and services in the security module. An earlier publication about the CincoSecurity module, oriented to programmers, focused on these technical details [18].

It is important to emphasize that to incorporate and manage the security of an application, the modules of the application, the use cases contained in such modules, and the services offered by these use cases must be registered into the CincoSecurity module (by using CincoSecurity's use cases provided for this). This way, the fine roles associated with these services can be included in the security profiles, and the access to these use cases will appears in the menu of authorized users.

VIII. COMPARISON WITH OTHER WORKS

There are other security modules proposed for the Java EE technology. Currently, in the SourceForge portal there is a dozen of software projects related with security for Java EE 5 [9] applications, but most of them are proposals without implementation (i.e., they are in planning status). Two relevant projects with implementation and good acceptance from users are the following:

- *JPA Security* [19] is an Access Control Solution for the Java Persistence API (JPA) [10] with support for role-based access control, access control lists (ACLs) and domain-driven access control. Compared with CincoSecurity, this project does not offer access control to web pages as CincoSecurity does. JPA Security uses access rules in terms of database operations, which provides a different type of flexibility from CincoSecurity, where security profiles are defined in terms of the services offered by the use cases of the application.
- *[fleXive]* [20] is a Java EE 5 open-source framework for the development of complex and evolving web applications. It offers an administration module that manages users and security. It implements an access control list based approach, combined with roles. Compared with CincoSecurity, this project uses 10 coarse roles with predefined permissions related to the administration module, while CincoSecurity lets to define any number of security profiles, each one as a set of fine roles related to the services of the application (not only to the security services). We believe it is more intuitive to associate the users to these security profiles and not to the [fleXive] Access control lists (ACL), that define lists of permissions attached to arbitrary objects. [fleXive] does not offer access control to elements of web pages, as CincoSecurity does.

With respect to recent proposals for extending the RBAC model, some research works as [21][22] try to statically validate the correctness of roles usage in an application, for solving what they call the fragility of traditional dynamic checks. CincoSecurity does not implement static checks, but its strategy of fine grained roles enables to automate the correct incorporation of security in a web application. In effect, by following the names discipline explained in this paper it is possible to automatically add annotations to each method of the session EJB3 controlling a use case, in order to permit its access only to users having the associated fine role; it is also possible to automatically modify button tags of JSF pages for rendering it only to users having the corresponding fine role.

On the other hand, it is important to note that Seam offers a complete security module [13], that is based on (coarse) roles, permissions and rules, that achieves a very flexible control of resources. The CincoSecurity module takes advantage of the Seam security by using some of its facilities

 ISBN: 978-1-61208-124-3

related with authentication, restriction annotations for roles, and tags of JSF pages for rendering only for the appropriate role. However, we believe that for an administrator it is more difficult to write rules for granting fine permissions to roles (as is done in the Seam module), than to configure security profiles by checking the services of the application to be granted (as is done in the CincoSecurity module). Also, given that CincoSecurity does not use permissions nor rules (only fine grained roles), the incorporation of security to a web application can be automated, as it was explained above; with the Seam module it seems more difficult to automate the incorporation of security.

IX. CONCLUSION AND FUTURE WORK

Needless to say that developing a secure Java EE application is a difficult job, where dozens of subtle details must be handled coherently. The CincoSecurity module provides a complete code baseline to develop a Java EE application with the Seam framework, incorporating from the beginning full and flexible access control to the use cases and services of the application being developed. The CincoSecurity module also provides the use cases required to administer the users and their access permissions to the use cases and services of the application being developed.

With respect to the Core RBAC model [6], an access permission is materialized in CincoSecurity as the right to invoke a method (service) of a business component. Fine grained roles are defined and implemented, each one having just one permission to invoke a single method (service) of a business component (session EJB3). There is also a fine grained role to allow entrance to each application's use case, as well as a fine grained role to grant access to each one of the services provided by the use case. The concept of "security profile" is defined and implemented as a set of fine grained roles.

The CincoSecurity module takes advantage of the low level access control principle implemented by any application server, by feeding the application server with the fine grained roles included in the security profiles the authenticated user belongs to. Additionally, the CincoSecurity module dynamically builds a customized menu containing only the entries leading to the application's use cases authorized for the user.

The CincoSecurity module is a Java EE 5 application built using the Seam framework [11][12]. It is distributed under the GPL license and can be freely downloaded from http://sourceforge.net/projects/cincosecurity. CincoSecurity is used by several software houses in Colombia.

As future work, CincoSecurity will be extended to automate the incorporation of security to a web application (business components and web pages), as well as to include other capabilities of the Seam security module, like Identity management, in a compatible way with our approach.

REFERENCES

[1] R. L. Baldwin, "Naming and Grouping Privileges to Simplify Security Management in Large Databases", Proceedings of the 1990 IEEE Symposium on Research in Security and Privacy (Oakland, CA), IEEE Computer Society Press, pp. 116-132, 1990.

[2] D. F. Ferraiolo and D. R. Kuhn, "Role-Based Access Control". Proc. 15th Nat'l Information Systems Security Conf., Diane Publishing Company, pp. 554–563, 1992.

[3] R. Sandhu, C. L. Feinstein, and C. E. Youman, "Role-Based Access Control Models". IEEE Computer Magazine, pp. 38-47, 1996.

[4] American National Standard for Information Technology – Role Based Access Control, ANSI INCITS 359-2004, 2004.

[5] B. Messaoud, "Access Control Systems: Security, Identity Management and Trust Models", Springer Science+Business Media, Inc., 2006.

[6] D. F. Ferraiolo, D. R. Kuhn, and R. Chandramouli, "Role Based Access Control", Artech House 2003, 2nd Edition 2007.

[7] N. Li, J. W. Byun, and E. Bertino, "A Critique of the ANSI Standard on Role-Based Access Control", IEEE Security and Privacy, Volume 5, Issue 6, pp. 41-49, 2007.

[8] D. Alur, J. Crupi, and D. Malks, "Core J2EE Patterns: best practices and Design Strategies", Sun Microsystems - Prentice Hall, 2001.

[9] Sun Microsystems, "The Java EE 5 Tutorial", http://java.sun.com/javaee/5/docs/tutorial/doc 01.18.2011

[10] M. Keith and M. Schincariol, "Pro EJB 3: Java Persistence API", Apress, 2006.

[11] M. Yuan and T. Heute, "JBoss Seam: Simplicity and Power Beyond Java EE", Prentice Hall, 2007.

[12] D. Allen, "Seam in Action", Manning Publications Co., 2009.

[13] JBoss Seam Group, "Reference manuals of JBoss Seam", http://seamframework.org 01.18.2011

[14] JBoss Community, "JBoss Application Server", http://www.jboss.org/jbossas 01.18.2011

[15] Eclipse Open source development platform comprised of extensible frameworks, tools and runtimes, http://www.eclipse.org 01.18.2011

[16] General Public License, http://www.gnu.org/licenses/gpl.html 01.18.2011

[17] M. C. Franky, V. M. Toro, and R. López, "CincoSecurity Module", http://sourceforge.net/projects/cincosecurity 01.18.2011

[18] M. C. Franky, V. M. Toro, and R. López, "CincoModule: Módulo de seguridad basado en roles finos y en perfiles de seguridad para aplicaciones Java EE 5". Quinto Congreso Colombiano de Computación (5CCC), Cartagena-Colombia, Abril 2010. ISBN: 978-958-8387-40-6.

[19] "JPA Security", http://jpasecurity.sourceforge.net 01.18.2011

[20] D. Lichtenberger, M. Plesser, G. Glos, J. Wernig-Pichler, H. Bacher, A. Zrzavy, and C. Blasnik, "[fleXive]™ 3.1 Reference Documentation", Copyright © 1999-2010 UCS - unique computing solutions gmbh, http://www.flexive.org/docs/3.1/xhtml/index.xhtml 01.18.2011

[21] P. Centonze, G. Naumovich, S. J. Fink, and Marco Pistoia, "Role-Based access control consistency validation". In Proceedings of the 2006 international symposium on Software testing and analysis (ISSTA'06). ACM, New York, NY, USA, pp. 121-132. , 2006

[22] J. Fischer, D. Marino, R. Majumdar, and T. Millstein, "Fine-Grained Access Control with Object-Sensitive Roles", ECOOP 2009 – OBJECT-ORIENTED PROGRAMMING, Lecture Notes in Computer Science, Volume 5653/2009, pp. 173-194, 2009.

 ISBN: 978-1-61208-124-3