BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 61, No. 3, 2013
DOI: 10.2478/bpasts-2013-0058

Modeling and design of role engineering in development of access control for dynamic information systems

A. PONISZEWSKA-MARAŃDA*

Institute of Information Technology, Technical University of Lodz, 215 Wólczańska St., 90-924 Łódź, Poland

Abstract. Nowadays, the growth and complexity of functionalities of current information systems, especially dynamic, distributed and heterogeneous information systems, makes the design and creation of such systems a difficult task and at the same time, strategic for businesses. A very important stage of data protection in an information system is the creation of a high level model, independent of the software, satisfying the needs of system protection and security. The process of role engineering, i.e. the identification of roles and setting up in an organization is a complex task.

The paper presents the modeling and design stages in the process of role engineering in the aspect of security schema development for information systems, in particular for dynamic, distributed information systems, based on the role concept and the usage concept. Such a schema is created first of all during the design phase of a system. Two actors should cooperate with each other in this creation process, the application developer and the security administrator, to determine the minimal set of user's roles in agreement with the security constraints that guarantee the global security coherence of the system.

Key words: access control of information systems, access control models, role engineering, usage control.

1. Introduction

Nowadays, the growth and complexity of functionalities of the current information systems, especially dynamic, distributed and heterogeneous information systems, makes the design and creation of such systems a difficult task and at the same time, strategic for businesses [1,2]. Data protection against improper disclosure or modification in the information system is an important issue of each security policy realized in the institution. Distributed information systems or federation of information systems provide the access of many different users to huge amount of data, sometimes stored in different locations and secured by different strategies, security policies and models or inner enterprise rules. These users have different rights to the data according to their business or security profiles that depend on their organization positions, current locations and many others conditions. Therefore, it is important for an enterprise to develop the security system that secures the information system against external and internal threats.

A very important stage of data protection in an information system is the creation of high level model, independent of the software, satisfying the needs of system protection and security. Role engineering in a domain of access control is a process that consists of determination and definition of roles, permissions, security constraints and their relations. The company's roles can be regarded in two aspects – functional (reflects the main business functions of a company) and organizational (reflects the organization hierarchy of a company). Before the implementation of concrete access control model, the role engineering process has to take place. The issue of role engineering is connected close to the issues of modeling and design of information systems. During our previous studies, we examined the design methods of the information systems and the design methods of information system security [3]. However, it is difficult to find the global method that takes into account both the design of system and its associated security scheme.

The process of role engineering, i.e. identification of roles and setting up in an organization is a complex task because very often the responsibilities of actors, e.g. users, in an organization and their functions are few or badly formalized. Moreover, the role concept is an abstract approach. It does not correspond to particular physical being and therefore it is very difficult to give the definition that comprise the whole world. Furthermore, numerous security constraints should be formulated in order to define the security policy in a proper way. They are defined both at developer level and administration level of security schema. The main difficulty is to assure the global coherence of all elements (applicative and organizational) during the whole system lifecycle, especially during the system exploitation, e.g. when a new application with the new elements is added to the existing information system. To conclude, the research of roles in security schema needs a real engineering approach that provides a solution to identify, specify and maintain such schema and their components.

Many different works concerning the problem of role engineering were presented in the literature [4–12]. First of all, most of the papers treat the static aspects of the access control domain. They do not take into consideration the problems of dynamic changes affecting the access control rules in modern information systems, particularly distributed information sys-

*e-mail: anetap@ics.p.lodz.pl

tems. Moreover, whereas everyone agrees that safety must be taken into account as quickly as possible, the proposed role engineering approaches are often disconnected from the design and the evolution of information system for which they are provided.

Coyne in [5] proposes to collect different user activities to define the candidate roles. Fernandez et al. in [6] propose to use the use cases to determines the role rights of system users but they do not describe the constraints aspect and the role-hierarchies. Epstein et al. [8] propose to express RBAC elements with UML diagrams but do not define the role engineering process. Roeckle et al. [9] propose a process-oriented approach for role engineering but only at meta level without details about the role hierarchy, derivation of permissions and relation between some elements. Epstein and Sandhu [10] describe the role engineering of role-permission assignment but not go into detail about constrains and role hierarchies in the process. Neumann and Strembeck propose in [13, 14] very interesting scenario-driven role engineering process for RBAC model. However this proposal does not take into consideration the dynamic aspects of access control.

The paper presents the modeling and design stages in the process of role engineering in the aspect of security schema development for information systems, in particular for dynamic, distributed information systems, based on the role concept and the usage concept. Such a schema is created first of all during the design phase of a system. Two actors should cooperate with each other in this creation process, the application developer and the security administrator, to determine the minimal set of user's roles in agreement with the security constraints that guarantee the global security coherence of the system.

The purpose of the paper is to show the methodology for analysis and design of access control based on the role concept and the usage concept during the development process of information systems. The proposed solution is based on the authoring access control model with the use of UML concepts in order to obtain a harmonized approach in the matter of roles production for dynamic information systems. Consequently, the purpose of the role engineering is to determine the security profiles for authorized users of a system application.

The paper is composed as follows: section 2 presents the role concept and usage concept from the point of view of access control. It gives the short description of current state of the art in domain of logical security. The next sections presents the results of research carried out by the author of the paper. Section 3 gives the outline of the approach URBAC (Usage Role Based Access Control) based on the concepts presented in the previous part with its formal definition and its representation with the use of UML concepts. Section 4 deals with the cooperation of two actors in role creation process of information system security, while section 5 shows the process of roles production based on URBAC approach presenting the rules and stages for creation of security profiles for system's users and describing the possible incoherences during the creation of the security scheme.

2. Role concept and usage concept in access control

Currently, the development in the area of information systems, especially modern, distributed information systems causes that traditional access control models are not sufficient and adequate for such systems in many cases. Some imperfections were found in the domain of these security models:

- traditional access control models provide only the mechanisms for definition of authorizations but do not give the possibility to define the obligations or conditions in the access control,
- developers or security administrators can only pre-define the access rights that will be next granted to the subjects,
- decision about the access can be only made before the required access but not during the access,
- mutable attributes cannot be defined for subjects and objects of a system.

These disadvantages and the needs of present information systems caused the creation of a unified model that can encompass the use of traditional access control models and allow to define the dynamic rules of access control. Two access control concepts are chosen in our studies to develop the new approach for dynamic information systems: the role concept and the usage concept. The first one allows to represent the whole system organization in the complete, precise way while the second one allows to describe the usage control with authorizations, obligations, conditions, continuity (ongoing control) and mutability attributes.

Role-Based Access Control (RBAC) [15, 16] or its extensions [17] require the identification of roles in a system. The role is viewed as a structure around which the access control policy is formulated. Each role realizes a specific task in the enterprise process and it contains many functions that the user can take. A role can be presented as a set of functions that this role can take and realize. Specific access rights are necessary to realize a role or particular function of this role.

The *Usage Control (UCON)* [18,19] is based on three sets of decision factors: authorizations, obligations and conditions that have to be evaluated for the usage decision. It consists of eight main components. The UCON strategy has two characteristic features: mutability and continuity. Mutability represents the mutability of subject and object attributes that can be either mutable or immutable. The mutable attributes can be modified by the actions of subjects and the immutable attributes can be modified only by the administrative actions. The continuity means that a decision can be made before, during or after an access [18, 19].

3. Approach of role based access control with concept of usage control

The complexity of actual, current organizations (i.e. matrix based organizational structure) has guided our proposal to extend the standard RBAC model by the role management facilities that allow a finer distribution of responsibilities in an

enterprise. On the other hand, the dynamism of current information systems, especially distributed information systems or federation of information systems was the reason to use the concepts of Usage Control to develop the new implementation of access control to support the management of information system security.

In order to meet these requirements and problems in modern access control, a new access control approach, named Usage Role-based Access Control (*URBAC*) was proposed. This approach was based on two access control models: extended RBAC model [17] and UCON model [18, 19].

The traditional access control models utilize only the authorizations for a decision process. Authorization decision is made based on the subject attributes, object attributes and the requested rights. In discretionary access control, definition of access control list is based on the properties of objects and subjects (i.e object attributes and subject attributes). In mandatory access control, the clearance labels of subjects are based on subject properties (i.e. subject attributes) and the classification labels of objects are based on object properties (i.e.object attributes). Role-based access control model assign the users to roles based on the users' properties, responsibilities and privileges (i.e. subject attributes). Definition of the permissions and the relationships between roles and permissions is given based on the objects' attributes for which these permissions were defined.

However, a usage decision is not only based on the existence of subject attributes and object attributes. It is based also on the fulfillment of required actions by the users while access to the data or simply when connecting to the system and is based also on certain environmental or system status that can be static and constant during some period of time or can change dynamically. Such decision factors were named obligations and conditions and they represent the dynamic aspects of access control in information systems. They represent the type of constrains that determine the possibility to allow an access to data based on the factors that can change dynamically – they can be evaluated before or during the access request. Moreover, the usage of an object can demand some modifications in the subject attributes or in the object attributes before an access, during an access or even after the access.

The existence of obligations and conditions has been recognized in modern information systems, particularly dynamic, distributed information systems, that realize many business transactions each day, hour or minute such as enterprise information systems of type B2B (i.e. transactions and interactions between business partners) or B2C (i.e. systems of mass distribution of services or products).

3.1. Usage Role-based Access Control. Two types of users were distinguished in the URBAC model: a single user (*User*) and a group of users (*Group*). These two elements are represented by the element *Subject* that is the superclass of User and Group. *Subjects* can be regarded as individual human beings. They hold and execute indirectly certain rights on the objects. Subject permits to formalize the assignment of users and groups of users to the roles. *User* is a human being, a person or a process in a system, so it represents the system entity, that can obtain some access rights in a system. *Group* represents a group of users that have the same rights. Subjects can be assigned to the groups by the aggregation relation *SubjectGroup* that represents an ordering relation in the set of all system subjects. Groups of users are often created in enterprise management information systems as PLM systems or ERP systems to provide the possibility to assemble a set of people in a group with the same obligations, responsibilities or privileges.

Role is a job function or a job title within the organization and can represent a competency to do a specific task, and can embody the authority and responsibility. The roles are created for various job functions in an organization. The users are assigned indirectly to the roles, based on their responsibilities and qualifications. The user can take different roles on different occasions and also several users can play the same role (*Group* element). It is also possible to define the hierarchy of roles, represented by relation *RoleHierarchy*, which represents the inheritance relations between the roles.

The association relation between the roles and subjects is described by the association class *SubjectAttributes* which represents the additional subject attributes (i.e. subject properties), for example an identity, enterprise role, credit, membership, security level. A role can contain many functions *Function* that a user can apply. Consequently, a role can be viewed as a set of functions that this role can take to realize a specific job. It is also possible to define the hierarchy of functions, represented by the relation *FunctionHierarchy*. Hierarchy of functions, just like hierarchy of roles, represents the inheritance relations between the functions.

Because each function can perform one or more operations, a function needs to be associated with a set of related permissions *Permission*. To perform an operation one has the access to required object, so necessary permissions have to be assigned to the corresponding function. Therefore, all the tasks and required permissions are identified and they can be assigned to the users to give them the possibility to perform the responsibilities involved when they play a particular role. Due to the cardinality constraints, each permission must be assigned to at least one function.

The permission determines the execution right for a particular method on the particular object. In order to access the data, stored in an object, a message has to be sent to this object. This message causes an execution of particular method *Method* on this object *Object*. It can be said that a *permission* is a possibility of method execution on an object in order to access the data stored in this object. Very often the constraints have to be defined in assignment process of permissions to the objects. Such constraints are represented by the authorizations and also in same cases by the obligations and/or conditions.

Authorization (A) is a logical predicate attached to a permission that determines its validity depending on the access rules, object attributes and subject attributes. *Obligation (B)* is a functional predicate that verifies the mandatory requirements, i.e. a function that a user has to perform before or

during an access. *Conditions (C)* evaluate the current environmental or system status for the usage decision concerning the permission constraint.

A security constraint determines that some permission is valid only for a part of the object instances. Therefore, the *permission* can be presented as a function

$$p(o, m, Cst),$$

where *o* is an object, *m* is a method which can be executed on this object and *Cst* is a set of constraints which determine this permission. Taking into consideration the concept of authorization, obligation and condition, the set of constraints can take the following form *Cst* = {*A, B, C*} and the permission can be presented as a function

$$p(o, m, \{A, B, C\}).$$

According to this, the permission is given to all instances of the object class except the contrary specification.

The *objects* are the entities that can be accessed or used by the users. The objects can be either privacy sensitive or privacy non-sensitive. The relation between objects and their permissions are additionally described by an association class *ObjectAttributes* that represents the additional object attributes (i.e. object properties) that can not be specified in the object's class and they can be used for the usage decision process. The examples of object attributes are security labels, ownerships or security classes. They can be also mutable or immutable as subject attributes do.

The *security constraints* can be defined for each main element of the model presented above (i.e user, group, subject, session, role, function, permission, object and method), and also for the relationships among these elements. The concept of constraints was described widely in the literature [20–22]. It is possible to distinguish different types of constraints, static and dynamic, that can be attached to different model elements.

The URBAC model distinguishes the following general types of security constraints:

- *Authorizations* – constraints defined for the permissions, based on the access rules defined by enterprise security policy but also based on the objects' attributes and subjects' attributes. The authorizations evaluate the subject attributes, object attributes and the requested permissions together with a set of authorization rules for the usage decision.
- *Obligations* – constraints defined for the permissions but concerning also the subjects – *Subject* can be associated with the obligations which represent different access control predicates that describe the mandatory requirements performed by a subject before (*pre*) or during (*ongoing*) an access. They can represent the security constraints that are defined on the subjects (i.e. users or groups) and they can be static or dynamic.
- *Conditions* – constraints defined also for the permissions but they concern the session – *Session* can be connected with the set of conditions that represent the features of a system or an application. They can describe the current environmental or system status and states during the user session that are used for the usage decision.
- Constraints on roles and on functions. The most popular type in this group of constraints are *Separation of Duty (SoD)* constraints [20, 22].
- Constraints on relationships between the model elements [22, 23].

The meta-model of URBAC approach with the set of elements and relationships presented above is shown in Fig. 1.

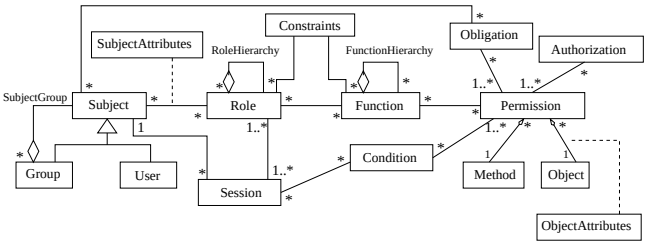

Fig. 1. Meta-model of URBAC approach

Formal definition of URBAC approach. The URBAC can be defined with the use of following components:

- main sets of elements: *U* – users, *G* – groups, *S* – subjects, *R* – roles, *F* – functions, *P* – permissions, *M* - methods, *O* – objects and *Sn* – sessions,
- additional sets of elements: *Cl* – classes, *Op* – operations, *ATT(S)* – subject attributes, *ATT(O)* – object attributes,
- functional decision predicates: *A* – authorizations, *B* – obligations and *C* – conditions,
- $UA \subseteq U \times R$ is many-to-many user-to-role assignment relation,
- $GA \subseteq G \times R$ is many-to-many group-to-role assignment relation,
- $subAttribute : U \rightarrow 2^{SAtt}$ is a function mapping each user u_i to a set of its attributes $sAtt(u_i)$,
- $subAttribute : G \rightarrow 2^{SAtt}$ is a function mapping each group of users g_i to a set of its attributes $sAtt(g_i)$,
- $RH \subseteq R \times R$ – a partial order on R set, called the role hierarchy or role dominance relation,
- $FA \subseteq F \times R$ – many-to-many function-to-role assignment relation,
- $FH \subseteq F \times F$ – a partial order on F set, called the function hierarchy or function dominance relation,
- $PA \subseteq P \times F$ – many-to-many permission-to-function assignment relation,
- $user : Sn \rightarrow U$ is a function mapping each session sn_i to a single user $user(sn_i)$,
- $group : 2^{Sn} \rightarrow G$ is a function mapping a set of sessions Sn to the group of users $group(users(sn_i))$,
- $roles : Sn \rightarrow 2^R$ is a function mapping each session sn_i to a set of roles $roles(sn_i) \subseteq \{r|(user(sn_i), r) \subseteq UA\}$ and session sn_i has the functions $\bigcup_{r \in roles(sn_i)} \{f|(f, r') \subseteq FA\}$,
- taking into consideration the hierarchy of roles: $roles : Sn \rightarrow 2^R$ is a function mapping each session sn_i to a set of roles $roles(sn_i) \subseteq \{r|(\exists r' \geq r)[(user(sn_i), r')] \subseteq UA\}$ and session sn_i has the functions $\bigcup_{r \in roles(sn_i)} \{f|(\exists r'' \leq r)[(f, r'') \subseteq FA]\}$,

- $functions : R \rightarrow 2^F$ is a function mapping each role r_i to a set of functions $functions(r_i) \subseteq \{f|(f, r_i) \subseteq FA\}$ and role r_i has the permissions $\bigcup_{f \in functions(r_i)} \{p|(p, f) \subseteq PA]\}$,
- taking into consideration the hierarchy of functions: $functions : R \rightarrow 2^F$ is a function mapping each role r_i to a set of functions $functions(r_i) \subseteq \{f|(\exists f' \geq f)[(f, r_i)] \subseteq FA\}$ and role r_i has the permissions $\bigcup_{f \in functions(r_i)} \{p|(\exists f'' \leq f)[(p, f'') \subseteq PA]\}$,
- $permission : M \times O \rightarrow P$ is a function mapping a couple: method m_i and object o_j to the permission $p(m_i, o_j)$,
- $objAttribute : O \rightarrow 2^{OAtt}$ is a function mapping each object o_i to a set of its attributes $oAtt(o_i)$,
- $AUTH \subseteq 2^A \times P$, many-to-many assignment relation of authorizations to a permission,
- $OBLIG \subseteq 2^B \times S$, many-to-many assignment relation of obligations to a subject,
- $COND \subseteq 2^C \times Sn$, many-to-many assignment relation of conditions to a session.

3.2. Representation of URBAC model using the UML concepts. Currently, the Unified Modeling Language (UML) is a standard language for analysis and design of information systems. It is widely known and used in the software engineering field to support the object oriented approach. UML gives the possibility to present the system using different models or points of view. Therefore, UML can be used in the role engineering process to implement and realize the URBAC approach. To accomplish this, the concepts of UML and URBAC model should be first joined. Two types of UML diagrams have been chosen to provide the URBAC approach: use case diagram and interaction diagram. The use case diagram presents the system functions from the user point of view. It define the system's behavior without functioning details. According to the UML meta-model, each use case from a use case diagram should be described by a scenario and in consequence by at least one interaction diagram (i.e. sequence diagram or communication diagram). The interaction diagram describes the behavior of one use case with objects and messages exchanged during the use case processing [24, 25].

The relationships between UML concepts and the concepts of usage role-based access control model are as follows (Fig. 2) [26]:

- role (R) from URBAC is presented as an UML actor,
- function (F) from access control model is represented by an UML use case,
- each actor from the use case diagram is in the interaction with a set of use cases and these relations specify the relations of R-F type (relations between roles and functions),
- methods executed in the sequence diagrams and also in the other UML diagrams represent the methods of URBAC,
- objects that occur in the UML diagrams, e.g. sequence diagrams, are attached to the object concept of access control model,
- analysis of the sequence diagram(s) describing the particular use case allows to find the permissions (P) of URBAC,
- use case diagram offers four types of relations between its elements:
  - communication relation between an actor and a use case that represents the relation between role and a function, i.e. R-F relation,
  - generalization relation between actors, representing the inheritance relation between roles (R-R relation),
  - two types of relations between use cases represent the inheritance relations between functions of URBAC, i.e. F-F relations,
- subject attributes (e.g. user attributes) from URBAC are represented by the set of attributes defined for an instance of an actor class on UML class diagram,
- concept of the object attributes from URBAC are attached to set of attributes defined for the object in its class specification.

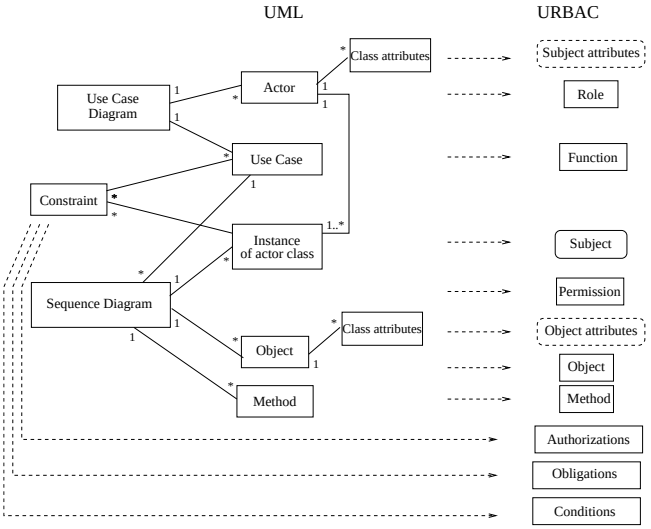

Fig. 2. Relationships between URBAC concepts and UML concepts

The concept of constraints in the URBAC approach corresponds directly to the constraint concept existing in UML. The security constraints of URBAC can be defined for different elements and for the relations between these elements. These constraints are presented on UML diagrams corresponding to the types and locations of elements for which these constraints are defined. The authorization is a constraint attached strictly to a permission and it is represented by the UML constraint defined for the method's execution in a sequence diagram. The obligation is a constraint defined on a permission but it concerns also the subject (e.g. a user). This type of constraints is presented as UML constraint in a sequence diagram (as a pre-condition or an invariant), especially from version 2.0 of UML that provides the combinator fragments in the sequence diagrams (e.g. "alt" or "opt") allowing the definition of constraint conditions. The condition is a constraint also defined on a permission but it concerns the session element. The conditions are also represented by the UML constraints defined in the sequence diagrams (mainly as an invariant). Remaining types of constraints represent the constraints defined for

roles, functions and their relations. Such constraints are represented by UML constraints defined for actors, use cases and their relations on the use case diagrams or sometimes on the sequence diagrams.

4. Creation of roles from the point of view of two actors

Two types of actors cooperate in the design and realization of security schema of an information system [22]: application/system developer who knows its specification that should be realized and on the other hand the security administrator who knows the general security rules and constraints that have to be taken into consideration at the whole company level. The partition of responsibilities between these two types of actors in the process of definition and implementation of the security schema is proposed and this is the basis for determining their cooperation for the global access control schema that fulfill the concepts of URBAC (Fig. 3).

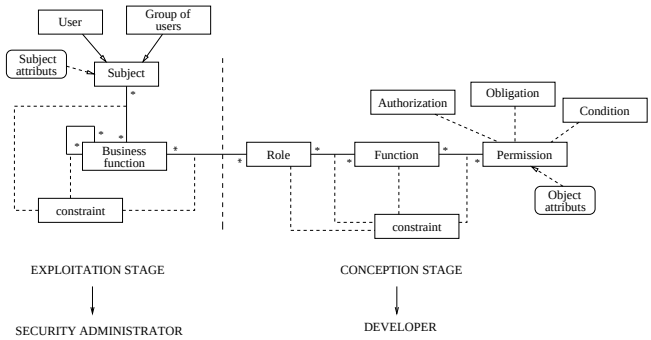

Fig. 3. Two actors in creation of security schema for information system

Conception stage. The conception stage is realized by the application developer at the local level of an information system. The realization process of an information system or a simple application is provoked by the client's request to create a new information system or a new application (i.e. to add a new component to existing information system). Basing on the client's needs and requirements the application/system developer creates the logical model of this application/system and next designs the project of this system that will be the base for its implementation. This model contains all the elements expressing the client's needs (i.e. the needs of future users).

The application developer defines the elements and constraints of this application corresponding to the client's specifications. These elements can be presented in a form adequate to the access control concepts, i.e. URBAC model. The developer generates the sets of the following elements: roles, functions, permissions and security constraints. These sets of elements have to be presented to the security administrator in a useful and legible form. The duties of the application developer on the basis of the URBAC approach are:

- definition of permissions – identification of the methods and objects on which these methods can be executed,
- definition of object attributes associated to certain objects according with the access control rules,
- assignment of elements: permissions to functions and functions to roles,
- definition of security constraints associated to the elements of the application, i.e. authorizations, obligations and conditions on the permissions and the standard security constraints on roles, functions and their relationships.

Exploitation stage. The exploitation stage is realized by the security administrator at the global level of an information system. The security administrator defines the administration rules and company constraints according to the global security policy and the application/system rules received from the developer. He receives from the developer the sets of elements in the form adequate to URBAC: set of roles, set of functions, set of permissions and set of security constraints of the application. He uses these sets to mange the global security of the information system. First of all he defines the users' rights to use the particular applications. Two sets of elements are important to define the users' rights on company level: persons working for the enterprise (i.e. users) and functions realized in the enterprise (i.e. business functions).

Business function is a task or a set of tasks that can be realized in a enterprise by one or more persons. From the access control point of view, the realization of such functions has to take into consideration the security constraints that specify the additional conditions for their performance.

Therefore, the duties of security administrator are as follows:

- definition of users' rights based on their responsibilities and their business functions in an organization – assignment of users to the roles of information system,
- organizing of users in groups and definition of access control rights for the groups of users that realize for example the same business functions – assignment of groups to the roles of information system,
- definition of subject (i.e. user or group of users) attributes associated to certain users or groups of users that allow to determine the dynamic aspects of the security constraints,
- definition of security constraints for the relationships between users and roles or group of users and roles at the global level with respect to defined security rules.

Another important task of the security administrator is the management of set of applications assuring the global coherence of the whole system at access control level. This assurance is exceptionally important in a case of addition of new application to the information system when new access control elements appear both at local and global level.

5. Production of roles based on URBAC approach

System security policies generally express the fundamental choices taken by an organization to protect the data. They define the principles on which an access is granted or denied. Role-based access control models are highly effective models,

especially in the large companies, that allow the administrator to simply place the new employees into the roles of the system. However, access control mechanisms still demand a clear definition of a set of activities and operations for each system's user that he will be allowed to execute. Consequently, a set of permissions should be defined for the user. As it was shown above, a set of permissions composes indirectly the system's roles. Therefore, the production of roles using the URBAC approach consists in the determination of permissions and next functions for the application/system roles with the consideration of defined security constraints.

The process of role production is based on the connections between UML and URBAC described in Subsec. 3.2. This process is realized with the use of use case diagrams, where the system roles and functions are defined and with the use of sequence diagrams, where the permissions are assigned to the rights of execution of methods realized in each use case. These two types of diagrams should be examined to identify the roles of URBAC, the functions that are used by these roles to interact with the information system and the permissions needed to realize these functions and the security constraints that determine the possible rights. To obtain these elements of the URBAC model, first the rules for creation of the set of roles have to be defined.

5.1. Rules for creation of security profiles. Each subject (i.e. user or group of users) of an information system is assigned to a security profile (i.e. user profile) which is defined by the set of roles that can be realized by him. Security profile is defined by a couple *(s, listRoles(s))*, where *s* is a subject, *listRoles(s)* is a set of roles assigned to this subject. Taking into consideration the concept of a user, such profile can be defined as follows: *(u, listRoles(u))*, where *u* is a user, *listRoles(u)* is a set of roles assigned to this user.

The proposal of the rules of the role creation and the assignments of these roles to users or groups of users in order to create the security profiles for the subjects of an information system is as follows:

1. An access control profile has to be defined for each subject (i.e. user) who interacts with the system

   $$s_i \vdash securityProfile_{s_i}$$

   in particular for a user:

   $$u_i \vdash securityProfile_{u_i}.$$

2. This profile is defined by a set of roles which can be assigned to the subject (i.e. user) with the respect to subject attributes defined mainly at the level of security administrator

   $$securityProfile_{s_i} \vdash setRoles_{s_i}, subjectAtt_{s_i}$$

   in particular for a user:

   $$securityProfile_{u_i} \vdash setRoles_{u_i}, subjectAtt_{u_i}$$

   To receive a significant profile, each subject (i.e. user) has to be assigned at least to one role.

3. A role is defined by a set of functions assigned to this role with the respect to potential security constraints cst_{r_j} defined for them

   $$r_j \vdash setFunctions_{r_j}, cst_{r_j}$$

   To receive a significant role, each role should have at least one function assigned.
   In UML meta-model, each actor has to be assigned at least to one use case in the use case diagram

   $$a_j \vdash setUseCases_{a_j}, setConstraints_{a_j}.$$

4. A function is defined by a set of permissions necessary to perform such function in accordance with the potential security constraints cst_{f_k} defined for such functions and/or the security constraints defined on the permissions (i.e. authorizations, obligations, conditions) $cstPermission_p$

   $$f_k \vdash setPermissions_{f_k}, cst_{f_k}, cstPermission_p,$$

   where

   $$cstPermission_{p_l} \vdash A_{p_l}, B_{p_l}, C_{p_l}.$$

   To receive a significant function, each function should have at least one permission assigned. In UML meta-model, each use case has to be defined by detail description, i.e. represented by a set of methods executed on the objects using the interaction diagram (in our case the sequence diagram)

   $$uc_k \vdash setPrivileges_{uc_k}, setConstrains_{uc_k}.$$

5.2. Creation of user profiles. Creation process of the user profiles, i.e. production of a set of roles, in an information system with the use of UML diagrams contains two stages:

- determination of *set of privileges* (i.e. permissions) for an *use case* in order to define a function and
- determination of *set of use cases* (i.e. functions) for an *actor* in order to define a role.

Determination of function and assigned permissions. As it was shown in Subsec. 3.2, a function of the URBAC model corresponds to an use case of UML meta-model. Use cases define the system functionality or in other words the needs of system's users that cooperate with this system to fulfill these needs. Each use case has to be defined by its scenario that determines the specification of use case interaction in form of sequence of actions performed on the system's objects. It allows to specify the set of privileges for execution of different actions on the objects. Therefore, in order to identify the permissions assigned to a function it is necessary to start from the sequence diagram corresponding to this function.

The interaction is defined as a set of messages and each message is realized by an action that can change the state of the system. Execution permission has to be assigned to each message of the sequence diagram.

Each message $msg\,(o_1, o_2, m)$ in the interaction sent by object o_1 to object o_2 to execute method m on object o_2

must have a suitable permission assigned to it. This permission corresponds to the execution of method m on object o_2. Taking into consideration the security constraints (Subsec. 3.1), the proposal of the message definition is as follows: $msg\,(o_1, o_2, m, cst)$, where cst represents a set of constraints - authorizations, obligations and conditions:

$$cst(p) = A\,(p) \cup B\,(p) \cup C\,(p)\,.$$

Consequently, the set of permissions for interaction i is defined as follows:

$$P\,(i) = \{p \mid \varphi\,(msg\,(o_1, o_2, m, cst)) = p\,(m, o_2) \wedge cst\,(p) = true\}\,,$$

where φ is a function that assigns a permission p to message msg, o_1 is an actor or class instance that can execute method m on object o_2 and cst is a set of constraints

$$cst(p) = A\,(p_{m,o_2}) \cup B\,(p_{m,o_2}) \cup C\,(p_{m,o_2})\,.$$

Sequence diagram in UML meta-model is defined by the set of interactions. Therefore, the set of permissions determined for sequence diagram Sqd and described by the set of interactions I_{Sqd} is as follows:

$$P\,(Sqd) = \bigcup_{i \,\in\, I_{Sqd}} P\,(i)\,.$$

The use case μ described by set D_I of interaction diagrams, e.g. sequence diagrams Sqd has the following set of permissions assigned to it:

$$P\,(\mu) = \bigcup_{Sqd_j \,\in\, D_I} P\,(Sqd_j)\,.$$

The set $P\,(\mu)$ represents the set of permissions assigned to the function f_j specified by the use case μ:

$$P\,(f_j) = \bigcup_{Sqd_j \,\in\, D_I} P\left(f_{Sqd_j}\right).$$

Determination of role and assigned functions. The use case diagram presents the system functionality from the point of view of its actors. The set of use cases can be found (i.e. URBAC functions) for each actor (i.e. URBAC role) examining this type of UML diagrams. Therefore, the determination of a role and its set of assigned functions is realized by examining the relationships between the actors and the use cases in the use case diagram.

The use case diagram UCd contains the use cases (i.e. functions) attached to the chosen actors (i.e. roles). The set of functions assigned to role r_j, described by one use case diagram UCd_i, is defined by the functions that are in direct or indirect relations with this role (i.e. by the inheritance relations between the functions) on this diagram:

$$F\,(r_{UCd_i}) = \{f \mid f = uc\,,\ uc \in UCd_i \wedge (r_{UCd_i}, f) \in R - F\} \cup \{f' \mid f' = uc\,,\ uc \in UCd_i \wedge ((f, f') \in F - F \wedge (r_{UCd_i}, f) \in R - F)\}\,.$$

The set of functions of the role r_j is defined by the union of use cases assigned to this role in all use case diagrams describing the whole application of the information system D_{UC}:

$$F\,(r_j) = \bigcup_{UCd_i \,\in\, D_{UC}} F\,(r_{UCd_i})\,.$$

In order to define the security profiles for the system's users or groups of users, the set of roles has to be assigned to the subject profiles (i.e. user profiles), taking into consideration the security constraints defined at the global level and the subject attributes defined for the subjects (i.e. users or groups of users) that determine the access control rights of particular system's users. This task is realized by the security administrator during the exploitation stage, described in Sec. 4.

5.3. Example of role production with the use of URBAC approach. The example illustrating the proposed approach from the point of view of the distribution of access control elements and the production of roles represents the applications of typical university information system. Such applications have different functionalities, range and different users. One of their users, for example the user "Professor" can have two roles: *Teacher* and *Researcher*, to perform his teaching and researching activities. The teaching role has a number of functions, for example prepare lectures, give lectures, prepare exams, record results, modify results, etc. The functions assigned to the researching role are for example: create a theory, test the theory, document the results, etc. Next, the sets of permissions have to be mapped to these functions to grant the access to perform the works required by each function (Fig. 4).

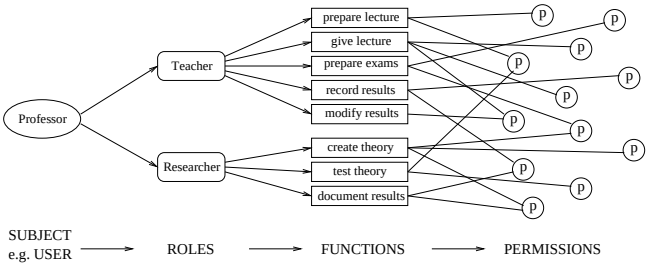

Fig. 4. Example of roles-functions-permissions mapping

In order to describe the particular user of an application or a system the following elements of URBAC model can be determined:

$subject\ s := \{Professor\}$
$user\ u := \{Prof.\ Tomas\ Smith\}$
$group\ g := \{professors\ at\ IT\ department\}$
$SubjectAttribute(subject\ s) :=$
$\{position\ of\ "Professor"\ at\ the\ department\}$

The development of an information system needs first the analysis and design of its components. The applications of the mentioned university system must be first described, for example with the use of UML concepts, i.e. the UML diagrams.

The analysis phase begins with the creation of use case diagrams for the applications. As it was presented in this section, it is possible to obtain from the use case diagram(s) the list of roles and the list of functions for an application or a whole system, for example:

$listRoles(ucd) = Roles(user\ u) = R_u :=$
$\{Teacher,\ Researcher\}$

$listFunctions(ucd) = Functions(Roles(user\ u)) = Functions(R_u) :=$
$\{prepare\ lectures, give\ lectures, prepare\ exams, record\ results, modify\ results, create\ theory, test\ the\ theory, document\ the\ results\}$

Next, each use case from the use case diagram has to be presented by means of at least one interaction diagram (i.e. sequence or communication diagram). This interaction diagram allows to find the permissions of the URBAC model that are assigned to a function represented by this use case but also the other elements of URBAC approach such as methods (represented by the messages sent from an actor or object to another objects), objects and security constraints (i.e. authorizations, obligations and conditions). Continuing the presented case, the examples of these elements, i.e. permissions, methods, objects and security constraints, for a selected function (i.e. use case) are the following:

$listMethods(uc) = Methods(Permissions\ p) :=$
$\{active(), content(), chose(), validTeacher(),$
$getLecture(Teacher), setExam(Lecture, Teacher),$
$setGrade(Student, Lecture, Exam), setValue(), ...\}$

$listObject(uc) = Objects(Permissions\ p) := \{: listStudents, : listLectures, : listExams, : Exam, : Grade, ...\}$

$listPermissions(uc)$
$= Permissions(Function\ f) :=$
$\{(content(), : listStudents), (getLecture(Teacher), : listLecture), (setExam(Lecture, Teacher), : listExam), (setGrade(Student, Lecture, Exam), : Exam), ...\}$

$ObjectAttribute(Object\ o) :=$
$\{weight\ of\ particular\ grades\ for\ the\ total\ grade\}$

$listAuthorizations(uc)$
$= Authorization(Permissions\ p)$
$:= \{"Teacher"\ can\ visualize\ only\ his\ own\ Lectures\}$

$listObligations(uc) = Obligation(Permissions\ p)$
$:= \{permission\ "(setGrade(Student, Lecture, Exam)"$
$can\ be\ executed\ only\ after\ execution\ of\ permission$
$"(setExam(Lecture, Teacher), : listExam)"\}$

$listConditions(uc) = Condition(Permissions\ p)$
$:= \{role\ "Professor"\ can\ realize\ the\ permission$
$"(setExam(Lecture, Teacher), :$
$listExam)"\ only\ during\ the\ business\ hours\}$

The presented example shows the main ideas of the role engineering process given in the previous sections. It demonstrates the possibilities of developed URBAC approach and the role production process based on it in domain of modeling and design of an access control for dynamic information systems. The URBAC assures the well-defined organization of the access control policy, as in the role-based models, and the usage control in data accessing that is especially very important in dynamic, distributed information systems.

5.4. Two viewpoints in creation of security schema and possible incoherences. The responsibilities of two actors mentioned in Sec. 4, i.e. the application developer and the security administrator are different during the definition process of logical security schema for an information system. The developer has to fulfill the needs of all future users of the information system and ensure that the defined roles, functions and permissions are in agreement with determined application constraints. On the other hand, the security administrator is responsible for the proper definition of security profiles for the system users and for the global coherence of system security.

The confrontation of these two points of view is based on the verification whether the developer's job is not in conflict with the work of security administrator. Such confrontation allows to find out the incoherences that can be caused by:

- addition of a new application that uses the elements (i.e. data) of another application existing in the system,
- addition of a new application with the new roles, functions and permissions and these elements will be next included in the role hierarchy of the whole system,
- addition of the new elements, e.g. roles or functions, and new relations between these elements and the elements already existing in the system,
- addition of a new business function to the system or removal of a business function, realized by the security administrator,
- removal of the elements from a system, e.g. an application, roles, by the security administrator.

These situations have to be resolved either at the conception and/or administration level. The confrontation of two points of view can reveal the possible incoherences between these two levels and then enable their elimination. The global coherence of dynamic, distributed information system has to guarantee that the concepts shared between different applications respect the global security schema. Such situation generally provokes the necessity of different modifications in defined sets of application elements, e.g. in the role hierarchy or in the function hierarchy.

The purpose of the verification of coherence is to guarantee that the addition of new elements, e.g. a new application, to an information system will not cause the incoherences and this verification should be performed before the integration of the new elements with the existing security schema. Important component of this process is the set of security constraints, defined by the developer and security administrator, that determine the coherence preserving.

The proposal distribution of the integration process of new elements with the logical security schema of an information system into two stages is as follows (Fig. 5):

1. Transformation of application/system UML Model that was defined by the developer to the model that will be understandable for the security administrator.

2. Verification of the new elements upon the fulfillment of the security constraints defined by the developer at application level and by the security administrator at global level.

Whereas, the transformation stage is composed of the following steps:

- creation of UML Model for an application by the developer, that contains the application elements and security constraints defined in OCL (Object Constraint Language) [25],
- translation of UML Model, containing the OCL constraints, into model consistent with the URBAC approach, containing the constraints defined in extended RCL (Role Constraint Language) – uRCL,
- integration of the new elements in model of existing system by the security administrator – addition of the roles and translation of the constraints into security schema.

The first step is the analysis and design of an application model based on the project specification containing the client's (i.e. future users') needs and functional requirements. Application developer with the use of UML defines the model of the new system component (i.e. application) containing all the elements and constraints from the point of view of URBAC approach. Next, he establishes the relations between these elements and generates the UML Model that has to be then translated for the security administrator.

The second step is the automatic use of developed parser that analyzes the application UML Model and preserves the concepts that will be used for the coherence verification. The parser with the use of rules defined in Sec. 5 translates the Model from UML into URBAC concepts returning the lists of roles, functions, permissions (objects and methods) and security constraints from the local security level.

During the third step the security administrator receives the application model or system model and defines or modifies the security profiles for the systems users (i.e. definitions for the new users, modifications for the existing users) with the use of URBAC concepts. He integrates the new application with an existing information system.

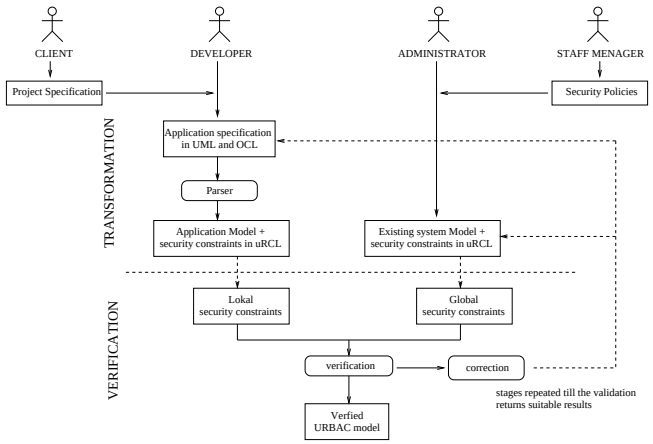

Fig. 5. Integration of logical security schema on two levels

The second stage of the integration process is the verification of global system coherence at the access control level and in case of possible incoherences the correction of elements and security constraints defined by the developer and by the security administrator. In a such case it is necessary to compare two groups of access control elements and constraints and eventually apply the changes if they are not coherent. This stage can be repeated till the verifications returns the correct, suitable system state and the new application can be properly integrated in the existing distributed information system.

6. Conclusions

The usage Role-based Access Control approach presented in the paper allows to define the access control policy based on an access request, as traditional access control models, and the access decision can be evaluated while the access to information which you want to control the usage. The described model takes into consideration the provisional aspects of access security. All the elements of the URBAC approach form a fairly complex model to present the whole organization of each enterprise at the access control level. It allows the complex structure of security policy to make possible the decomposition of this policy and make easy its definition. On the other hand, it expresses more than simple authorizations but also the interdictions or obligations that have to be fulfilled in order to obtain the access to a dynamic information system.

It seems the URBAC model can also support the security of dynamic information systems. It can be done thanks to the mutable concept came from the usage control where the dynamic change of security policy can be translated by the change of values of the subject attributes or object attributes.

The concepts of the presented access control approach were used to define the process of role engineering in the creation of security profiles for the users of an information system. The paper presents the representation of the URBAC approach with the use of UML concepts, the rules and stages of creation of the user profiles. The process of role production that is very important in the definition of logical security policy of an information system, is realized by two actors: application/system developer and security administrator who cooperate with each other to guarantee the global coherence at the access control level.

The presented approach was applied in practice. It was implemented on the software platform to provide the management of logical security for an information system from the point of view of the application developer and from the point of view of the security administrator – the project MACiDiS (*Management tool for Access Control in Dynamic information Systems*) in the framework of "TEWI" platform (the grant of European Union, *www.tewi.ics.p.lodz.pl*). This tool provides the implementation of role engineering process based on the URBAC approach and the management of information system security schema at the access control level.

The URBAC approach presented in the paper, has been also used and implemented with success as an access control model in a few prototypes of the information systems, both small and large, with both the static and dynamic char-

acteristics, for example it has been implemented in the web applications that currently have many dynamic features. This shows that the model works well in current applications and its rich structure provides the ability to define the complex properties and organizational dependencies of the particular application/system elements while determining the aspects of dynamic access control.

Our next research is concentrated on the aspects of security constraints for the URBAC approach and the algorithms to maintain the coherence of the URBAC schema during the modifications of an existing information system.

REFERENCES

[1] M. Lauder, M. Schlereth, S. Rose, and A. Schurr, "Model-driven systems engineering: state-of-the-art and research challenges", *Bull. Pol. Ac.: Tech.* 58 (3), 357–366 (2010).

[2] M. Kamola and P. Arabas, "Dynamically established transmission paths in the future Internet – proposal of a framework", *Bull. Pol. Ac.: Tech.* 59 (3), 357–366 (2011).

[3] G. Goncalves and A. Poniszewska-Maranda, "Role engineering: from design to evaluation of security schemas", *J. Systems and Software* 81 (8), 1306–1326 (2008).

[4] E. Bertino, C. Bettini, and P. Samarati, "A temporal access control mechanism for database systems", *IEEE Trans. on Knowledge and Data Engineering* 8 (1), 67–80 (1996).

[5] E. Coyne, "Role engineering", *Proc. 1st ACM Workshop on Role-Based Access Control* 1, CD-ROM (1996).

[6] E. Fernandez and J. Hawkins, "Determining role rights from use cases", *Proc. 2nd ACM Workshop on Role-Based Access Control* 1, 121–125 (1997).

[7] E. Bertino, E. Ferrari, and V. Atlurii, "The specification and enforcement of authorization constraints in workflow management systems", *ACM Trans. on Information and System Security* 2 (1), 65–104 (1999).

[8] P. Epstein and R. Sandhu, "Towards a UML based approach to role engineering", *Proc. 4th ACM Workshop on Role- Based Access Control* 1, 135–143 (1999).

[9] H. Roeckle, G. Schimpf, and R. Weidinger, "Process-oriented approach for role-finding to implement role-based security administration in a large industrial organization", *Proc. 5th ACM Workshop on Role-Based Access Control* 1, 103–110 (2000).

[10] P. Epstein and R. Sandhu, "Engineering of role permission assignment to role engineering", *Proc. 17th Annual Computer Security Applications Conference* 1, 127–136 (2001).

[11] D. Basin, J. Doser, and T. Lodderstedt, "Model driven security: From UML models to access control infrastructures", *ACM Trans. on Software Engineering Methodology* 15, 39–91 (2006).

[12] E. Coyne and J. Davis, *Role Engineering for Enterprise Security Management*, Artech House, London, 2008.

[13] G. Neumann and M. Strembeck, "A scenario-driven role engineering process for functional RBAC roles", *Proc. 7th ACM Symposium on Access Control Models and Technologies* 1, 33–42 (2002).

[14] M. Strembeck and G. Neumann, "An integrated approach to engineer and enforce context constraints in RBAC environments", *ACM Trans. on Information and System Security* 7 (3), 392–427 (2004).

[15] R.S. Sandhu and P. Samarati, "Access control: principles and practice", *IEEE Communication* 32 (9), 40–48 (1994).

[16] D. Ferraiolo, R. Sandhu, S. Gavrila, D. Kuhn, and R. Chandramouli, "Proposed NIST standard for role-based access control", *ACM Trans. on Information and Systems Security* 4 (3), 224–274 (2001).

[17] A. Poniszewska-Maranda, G. Goncalves, and F. Hemery, "Representation of extended rbac model using UML language", *LNCS* 3381, 413–417 (2005).

[18] J. Park and R. Sandhu, "The UCONABC usage control model", *ACM Trans. on Information and System Security* 7 (1), 128–174 (2004).

[19] J. Park, X. Zhang, and R. Sandhu, "Attribute mutability in usage control", *Proc. 18th Annual IFIP WG 11.3 Working Conference on Data and Applications Security* 1, 15–29 (2004).

[20] G.-J. Ahn, "The RCL 2000 language for specifying role-based authorization constraints", *Ph.D. Thesis*, George Mason University, Fairfax, 1999.

[21] G.-J. Ahn and R.S. Sandhu, "Role-based authorization constraints specification", *ACM Trans. on Information and Systems Security* 3 (4), 207–226 (2000).

[22] A. Poniszewska-Maranda, "Access control coherence of information systems based on security constraints", *LNCS* 4166, 412–425 (2006).

[23] A. Poniszewska-Marada, "Conception approach of access control in heterogeneous information systems using UML", *J. Telecommunication Systems* 45 (2–3), 177–190 (2010).

[24] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modelling Language User Guide*, Addison Wesley, London, 1998.

[25] OMG, *OMG Unified Modeling Language Specification*, 2011.

[26] A. Poniszewska-Maranda, "UML representation of extended role-based access control model with the use of usage control concept", *LNCS* 7465, 131–146 (2012).