JOURNAL OF APPLIED
COMPUTER SCIENCE
Vol. 22 No. 1 (2014), pp. 183-199

Model Driven Architecture for Modeling of Logical Security Based on RBAC Approach

Aneta Poniszewska-Marańda

Institute of Information Technology
Lodz University of Technology
Lodz, Wolczanska 215
anetap@ics.p.lodz.pl

Abstract. *This paper presents an approach of role-based access control (RBAC) for information systems with the use of MDA (Model Driven Architecture). The main purpose is to join the concepts of MDA approach with the concepts of access control models, in particular with the concepts of access control based on roles and on usage concept. To reach this objectives the appropriate solution was created to model the extended RBAC model and URBAC model with the use of concepts and tools of software engineering, in particular MDA methodology and UML (Unified Modeling Language). The presented approach was developed for role engineering in the aspects of logical security of information systems.*
Keywords: *access control, role-based access control, usage control, modeling of access control, model driven architecture.*

1. Introduction

Recently, rapid development in different technologies of information systems have caused the computerizing of many applications in various business areas. Data has become very important and critical resource in many organizations and therefore efficient access to data, sharing the data, extracting information from the data and making use of the information has become an important and urgent necessity.

Access control is a significant part of each information system. It is concerned with determining allowed activities of the system users and mediating every attempt by a user to access a resource in the system. The evident objective of access control is protecting the system resources from any undesired user access. Nowadays, the access control is connected with the great development of information technologies and methodologies that allow to create more complex, dynamic information systems.

Data protection against improper disclosure or modification in the information system is the important issue of each security policy realized in the institution. Access control policies, strategies or models in some cases should be improved or adapted to manage the security requirements of modern information systems, such as dispersion of data and resources, dynamic changes both in data proprieties and users' proprieties, responsibilities and abilities from the point of view of access control rules, dispersion of users, complex system organization with different users, rules and security principles that are sometimes mutually exclusive. It is also important to protect the information against non-controlled utilization and control the usage and diffusion of the information. It gives the possibility to specify how it can be used and specify the utilization constraints.

This paper presents an approach of role-based access control (RBAC) for information systems with the use of MDA (Model Driven Architecture). The main purpose is to join the concepts of MDA approach with the concepts of access control models, in particular with the concepts of access control based on roles and on usage concept. To reach this objectives the appropriate solution was created to model the extended RBAC model and URBAC model with the use of concepts and tools of software engineering, in particular MDA methodology and UML (Unified Modeling Language). The presented approach was developed for role engineering in the aspects of logical security of information systems.

The method developed in the paper was supported by the object-oriented design described in the UML (Unified Modeling Language). Owing to its high level diagrams, the UML [1, 2] allows to specify clearly the information system needs and facilitates the dialog between different actors (i.e. system user, application/system developer, security administrator) that interact in its design.

The paper is structured as follows: section 2 presents the access control policies, models and related works on access control concepts, section 3 deals with Model Driven Architecture and software engineering techniques for access control. Section 4 describes the MDA approach for modeling of access control in information system based on role concept and on usage concept.

2. Access control policies and models

Data protection against improper disclosure or modification is an important requirement of each information system. Access control allow to define the user's responsibilities and possibilities in a system. It can define what a user can do directly and also what programs executing on behalf of the user are allowed to do. Access control limits the activities of successfully authenticated users basing on the security constraints defined on the conception level and on the administration level.

The system administrators have to implement the access control mechanisms to protect the confidentiality and integrity of applications and its data. In the past, the user access was granted by adding necessary permissions to each individual application, which made the administration, involving many users and several different applications, complicated and ineffective.

The development of access control policies and models has a long history. It is difficult to mention all the access control models that were specified in literature and this is not the objective of this paper. It is possibly to distinguish two main approaches. The first one represents the group of traditional access control models. Discretionary Access Control (DAC) model [3, 4], the first model in these group, manages the users' access to information based on user identification and on the rules defined for each user (i.e. subject) and each object in the system using the access control matrix. However, the DAC model has the inherit weakness that information can be copied from one object to another and it is difficult for DAC to enforce the safety policy and protect the data against some security attacks.

In order to prevent the shortcomings of DAC model, the Mandatory Access Control (MAC) model was created to enforce the lattice-based policies [3]. In this model each subject has their own authorization level that permits them the access to objects starting from the classification level, which has lower or equal range. MAC does not consider the covert channels but they are expensive to eliminate. Next, Sandhu et al. proposed the Role-Based Access Control (RBAC) model [4, 5] that has been considered as an alternative to DAC and MAC models. This model requires the identification of roles in a system. The RBAC model was a progress in access control but it is still centered around the access control matrix and has static character, particularly from the point of view of distributed information systems [5].

The second approach of access control models corresponds to the temporal models that introduce the temporal features into traditional access control. The temporal authorization model was proposed by Bertino and al. in [6] that is based on the temporal intervals of validity for authorization and temporal dependencies among authorizations. Next, the Temporal-RBAC (TRBAC) model was proposed in [7]. This model introduces the temporal dependencies among roles. Other model - Temporal Data Authorization model (TDAM) was presented in [8] and extends the basic authorization model by temporal attributes associated to the data such as transition time or valid time. Recently, the TRBAC model was extended to Generalized Temporal RBAC (GTRBAC) model in [9] to express the wider range of temporal constraints.

Unfortunately, all these models propose still the static authorization of access control decisions that are based on the subject's permits and possibilities which can be performed on the target objects. If the access to the object is permitted, the subject can access it repeatedly at the valid time intervals, i.e. during the logging session. However, the appearance of new trends in global markets, trades and economy joined with the Internet development (i.e. e-business, e-markets, supply chain management) caused changes in information technologies, also concerning the development of information systems and security of these systems. In consequence, it provoked the necessity of new access protocols and solutions for flow of objects, informations stored in information systems, that are more complex and more distributed.

Park and Sandhu defined Usage Control (UCON) model [10, 11] as an access control generalization used for security policy evaluation before the access (as usual) and during the access to information because security policy can dynamically vary. The UCON considers the the temporal attributes as the mutable attributes of subjects or objects and permits to evaluate the usage decision also during the access to the information to which we want to control the usage. It is realized based on set of authorizations, obligations and conditions to satisfy.

However, the RBAC model seems to be still the most popular and most stable and for these reasons it was chosen at the beginning of our study. The definition of roles in the RBAC model implies the determination of organizational roles of each person in a given organization. It gives the possibility to present the organization structure from the point of view of access control. This model provides the support

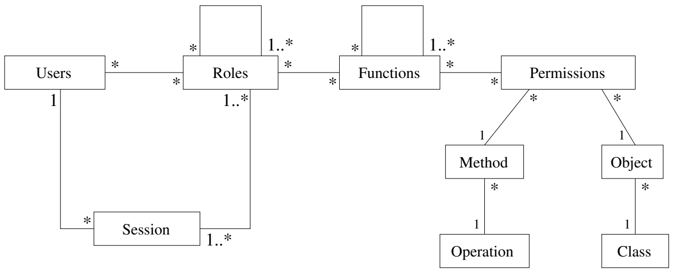


Figure 1. Extended RBAC model

for several important security principles (i.e. least privilege, privilege abstraction and separation of duties) but does not dictate how they should be put into practice. The same can be obtained with the UCON model. It gives the possibility to manage the dynamism of information systems security but the model it abstract and does not specify how to use them in practice. How to define the role engineering for information systems?

Role-Based Access Control (RBAC) [12, 13] requires the identification of roles in a system. The role is properly viewed as a semantic structure around which the access control policy is formulated.

In *extended RBAC (eRBAC)* model [14, 15] each role realizes a specific task in the enterprise process and it contains many functions that the user can take. For each role it is possible to choose the necessary system functions. Thus, a role can be presented as a set of functions that this role can take and realize. Each function can have one or more permissions, and a function can be defined as a set or sequence of permissions. Specific access rights are necessary to realize a role or a particular function of this role. Therefore, in extended RBAC model some elements were specified in comparison to classical RBAC model, i.e. function, object, method, class, operation, to express more complex the elements of information system that are secured by this model (Fig. 1).

Usage Role-based Access Control (URBAC) [16] is based on a role concept from extended Role-Based Access Control and usage concept from Usage Control. It takes best features from both models in order to combine them into an even more efficient model of an access control. It incorporates the control of usage in data access with authorizations, obligations and conditions that can be applied both before and during access. URBAC also uses a complete and precise way to

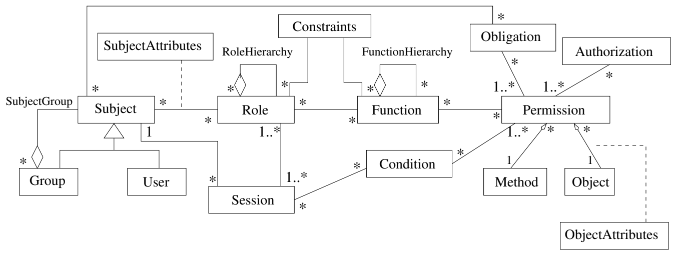


Figure 2. Meta-model of URBAC approach

represent the entire system organization with the use of roles and functions. The main elements of the model are presented in figure 2.

3. Model Driven Architecture and software engineering techniques for access control

MDA (Model Driven Architecture) is an approach that separates the specification of system operations from the implementation details of system functionality on the given platform. The business model in MDA is leaved aside as *Platform Independent Model* (PIM) and a system for concrete platform is defined as *Platform Specific Model* (PSM) that is created from PIM by application of translation rules specific for given platform [17].

In MDA everything is treated as a model of model element. In consequence, well defined model language, for example Unified Modelling language (UML), plays an important role because it is used to describe precisely this model. MDA gives the possibility to define the languages for concrete domains that can precisely formalize the specific areas such as business domains, system aspects or special technologies. Then, the users can describe PIM and PSM basing on defined meta-models, specific fo given domain.

MDA was to proposed as an approach determining the creation method of applications where systems are presented as models and as an approach for automatic generation of systems basing on these models. Of course, complete automatic synthesis of complex system from high-level description is unattainable in general. It is not possible to generate automatically the functions implementing system functionality, i.e. business logic, but we can automatize the generation of different no-functional system elements, specified for given platform, such as login or system aspects. Security and in particular access control are one of such aspects that can be generated automatically.

Model driven security means the reduction of gap between system models and security models and reduction of gap between system project and its implementation. The security, in particular the access control has to be openly integrated with modeling language and supported on transformation stage.

At the beginning the developer designs the RBAC XACML (respectively URBAC XACML) specification of PIM models basing on access control points together with application models. Next, created RBAC XACML (URBAC XACML) PIM model is transformed to RBAC/URBAC XACML specification characteristic for the platform that is used for generation of infrastructure of security files. After automatic generation of security infrastructure, the developers implement the missing parts basing on generated framework, create the system and test it.

Currently, Unified Modeling Language (UML) is a standard language for analysis and design of information systems. It is widely known and used in software engineering field to support the object oriented approach. Specification of the language differs between abstract syntax and notation. Abstract syntax defines the language of primitives used to build the models and the notation defines the graphical representation of these primitives. UML supports the description of system structure and behavior, gives the possibility to present the system using different models or points of view. It has a set of diagrams to represent the elements of whole information systems or one of its components. Some chosen features of UML can be used to implement the eRBAC model, especially during the design of information system and its associated security schema based on eRBAC model [1, 2].

UML describes the processes that occur in a system or application and such processes are use case oriented. Use case is the main concept of UML used in analysis and design of software. It is used for specifying required usages of a system, to capture the requirements of a system, describes what a system is supposed to do.

Two types of UML diagrams have been chosen to provide the RBAC/URBAC model: use case diagram and interaction diagram. The use case diagram presents the system's functions from the user point of view. It define the system's behavior without functioning details. According to UML meta-model, each use case from 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 [1, 2]. It represents the objects and messages exchanged during the use case processing.

eXtensible Access Control Markup Language (**XACML**) defines declarative language of access control realized in XML and model for evaluation processing of access requests according to rules specified in XACML. It is a security standard created to use the security rules and also to configure, develop and modify these rules with minimal costs [18].

XACML is first of all the access control system based on attributes (*Attribute Based Access Control*, *ABAC*), where attributes are joined with an user, an activity or resource and the making decision about granting the user access is based on these attributes. The role-based access control and its extensions can be also realized with the use of XACML.

The XACML allows to separate the decision of access permission from the concrete use case. When the decisions of access permission (i.e. authorizations) are placed in client applications it is very difficult to actualize the decision criteria especially when the changes in their granting occur. When the client (i.e. user) is separated from the access decision then authorization principles can be actualized on a current basis and they will concern all clients at once [18].

XACML based system contains four following elements:

- *PAP - Policy Administration Point* - management point of access control policy. Any modifications of access control policy and security rules are realized in this place.
- *PEP - Policy Enforcement Point* - this component verifies the access requests in respect of validity, user identity and return the access to required data or access refusal
- *PIP - Policy Information Point* - information point about access policy. It collects the information for decision point that will be used for decision making and has to ensure all the necessary information. PIP is usually the "front-end" component for many other systems tah can be used for security policy decision.
- *PDP - Policy Decision Point* - this point makes the decision for access request with the use of rule engine or another mechanism.

Access control policy in XACML standard is divided into three levels of elements (Fig. 3):

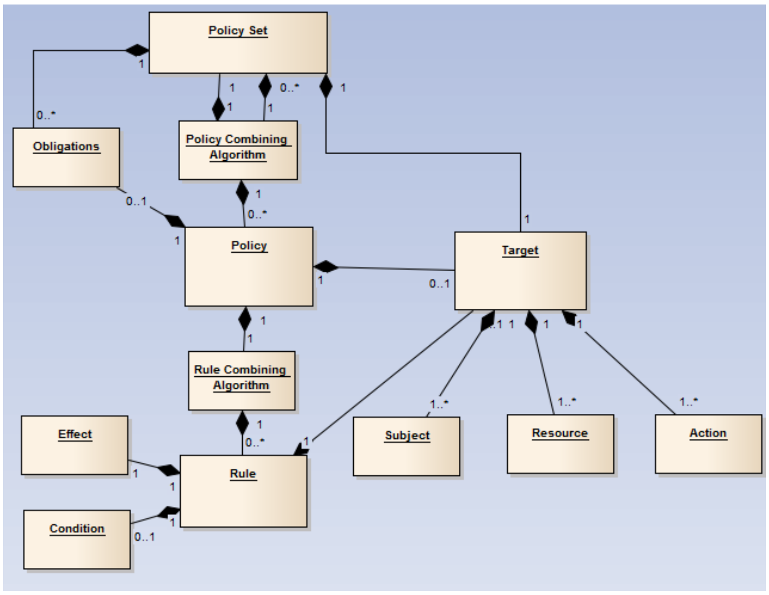
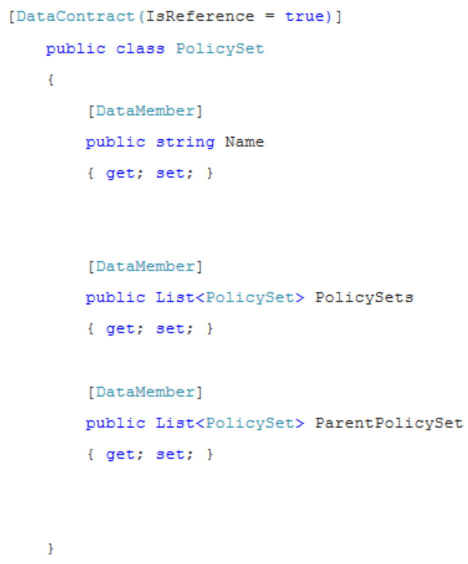


Figure 3. PolicySet, Policy and their joined components in XACML

- PolicySet - set of security policies (Fig. 4),
- Policy - set of access control rules (Fig. 5) and
- Rule (Fig. 6).

Subject is a system entity that requires an access. It can have one or more attributes. *Resource element* is a service of system component. Resources has only one attribute. *Action element* defines the access mode that is required for certain resource. Action has one or more attributes. *Environment element* can contain eventually additional information. Example of access control policy defined in XACML can be as follows:

```
<Policy PolicyId=" " RuleCombiningAlgId=" ">
    <Target>
       <Subjects> ...</Subjects>
       <Resources> ...</Resources>
       <Actions> ...</Actions>
    </Target>
    <Rule RuleId=" " Effect=" ">
```



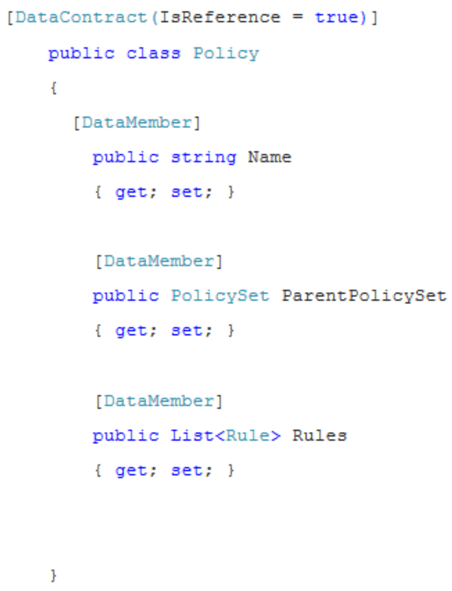
```
[DataContract(IsReference = true)]
    public class PolicySet
    {
        [DataMember]
        public string Name
        { get; set; }


        [DataMember]
        public List<PolicySet> PolicySets
        { get; set; }

        [DataMember]
        public List<PolicySet> ParentPolicySet
        { get; set; }


    }
```

Figure 4. PolicySet class implemented in .NET technology [19]

```
       <Target> ...</Target>
       <Condition FunctionId=" "> ...</Condition>
    </Rule>
    <Obligations>
       <Obligation ObligationId=" " FulFillOn=" "> </Obligation>
    </Obligation>
</Policy>
```

4. MDA for modeling of role based access control approach

The problem in MDA approach for modeling of role based access control and usage control is the expression transparency of access control method connected with minimal knowledge that should have the person who will model the rules of dynamic access control to resources. In such case, a resources can be consider in abstract way in order to find the universal method, independent on the secured resources.

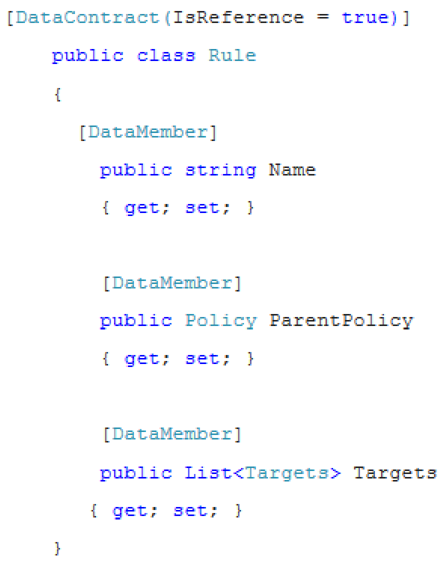


```
[DataContract(IsReference = true)]
    public class Policy
    {
      [DataMember]
        public string Name
        { get; set; }

        [DataMember]
        public PolicySet ParentPolicySet
        { get; set; }

        [DataMember]
        public List<Rule> Rules
        { get; set; }


    }
```

Figure 5. Policy class implemented in .NET technology [19]

The proposed solution of this problem is to join the intuitive approach of UML with XACML that allows to define the security policy. UML is a standard tool of object-oriented analysis and design of information systems that is known everywhere. From the other side, XACML ensures ready principles for resources security and uses XML that is standard language for record and exchange of data files. Moreover, there are exist programming libraries of XACML security standard for two the most popular programming platforms, i.e. .NET and Java (XACML.NET for C# language and Sun XACML for Java language).

Proposed solution allows to create a system basing on two independent applications. First application enables modeling of access policy in transparent way with the use of UML notation, in particular graphical notation of class/object diagrams. It is used to serve only the files created in XACML standard based on XML. For that reason it allows the creation of appropriate hierarchy of different objects, such as PolicySet, Policy, Rule, Subject, Target. Next, it allows to export the created object diagram to XML file or to load such file to the program (for example to show the graphical representation of set of policies and rules). Therefore,



```
[DataContract(IsReference = true)]
    public class Rule
    {
      [DataMember]
        public string Name
        { get; set; }

        [DataMember]
        public Policy ParentPolicy
        { get; set; }

        [DataMember]
        public List<Targets> Targets
       { get; set; }
    }
```

Figure 6. Rule class implemented in .NET technology [19]

the first part of a system is used to determine concrete permissions according to *RBAC* (*eRBAC* or *URBAC*) with the use of UML.

The second part of a system (i.e. second application) is responsible for login into system using API XACML, named Sun XACML. It grants users the access basing on the events and knowledge about the user that wants to access the resources. Event that can be used for such operations is for example timing event. This system's part has to verify if all policies modeled in the first part function in desirable way.

The solution used to join the modeling language independent on the platform, i.e. UML, with a standard for description of security rules functions through the possibility of creation and deletion of objects from certain hierarchy. Single object, represented by abstract class *SingleXACMLObject* allows to nest in it other inferior objects came from the same abstract class. Number of children of parent object is unlimited.

Application to model the access rules

The application was created to guarantee the functioning of set of rules. Therefore, the correct functioning can be based on single object of type *Policy* as a hierarchy root. This object can contain other objects of XACML standard, it can contain directly such objects as *Target*, *Description* or many objects of *Rule* type.

Role based access control policy based on eRBAC model defined in XACML standard is following:

```
<Policy PolicyId=" " RuleCombiningAlgId=" ">
    <Target>
       <User> ...</User>
       <Role> ...</Role>
       <Function> ...</Function>
       <Permission> ...</Permission>
       <Object> ...</Object>
       <Method> ...</Method>
       <Class> ...</Class>
       <Operation> ...</Operation>
    </Target>
    <Rule RuleId=" " Effect=" ">
       <Target> ...</Target>
       <Condition PermissionId=" "> ...</Condition>
    </Rule>
    <Constraint>
       <Constraint ConstraintId=" " FulFillOn=" "> </Constraint>
    </Constraint>
</Policy>
```

Access control policy based on URBAC approach defined in XACML standard is as follows:

```
<Policy PolicyId=" " RuleCombiningAlgId=" ">
    <Target>
       <Subject> ...</Subject>
       <User> ...</User>
       <GroupUsers> ...</GroupUsers>
       <Role> ...</Role>
       <Function> ...</Function>
       <Permission> ...</Permission>
       <Object> ...</Object>
       <Method> ...</Method>
       <Class> ...</Class>
       <Operation> ...</Operation>
    </Target>
    <Rule RuleId=" " Effect=" ">
       <Target> ...</Target>
       <Condition UserId=" "> ...</Condition>
    </Rule>
    <Rule RuleId=" " Effect=" ">
       <Target> ...</Target>
       <Condition GroupUsersId=" "> ...</Condition>
    </Rule>
    <Rule RuleId=" " Effect=" ">
       <Target> ...</Target>
       <Condition PermissionId=" "> ...</Condition>
    </Rule>
    <Constraint>
       <Constraint ConstraintId=" " FulFillOn=" "> </Constraint>
    </Constraint>
    <Authorization>
       <Authorization AuthorizationId=" " FulFillOn=" "> </Authorization>
    </Authorization>
    <Obligation>
       <Obligation ObligationId=" " FulFillOn=" "> </Obligation>
    </Obligation>
    <Condition>
       <Condition ConditionId=" " FulFillOn=" "> </Condition>
    </Condition>
</Policy>
```

Simple example of application functioning can be as follows. Single *Policy* contains one inferior element *Target* and two inferior rules *Rule*: first rule permits an access to a resources with some conditions and final rule that denies an access in other cases. File is recorded in XML standard and transfered to the second part of the system, responsible for an access to resources. Code generated in XACML for such example is as follows:

```
<Policy PolicyId="ExamplePolicy"
    RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining
    -algorithm:permit-overrides">
    <Rule RuleId="PermitRule" Effect="Permit"/>
    <Target/>
    <Rule RuleId="DenyRule" Effect="Deny"/>
</Policy>
```

5. Conclusion

Proposed conception and implemented solution can be used for real problems connected with access control to system resources. It is realized basing not only on users properties but also on additional context of access realization. The role based access control (in particular extended role based access control model) and usage control can be modeled with the use of MDA approach that contains UML, XML and XACML technologies. Moreover, the presented solution is independent on the platform and the only condition is installed Java Virtual Machine.

Important issue is the form and place for storage of generated set of access rules for different system users. These data are very important for proper functionality of a system from security point of view and they have to be accessible for system designers and security administrators.

References

[1] Booch, G., Rumbaugh, J., and Jacobson, I., *The Unified Modeling Language User Guide*, Addison-Wesley, 2004.

[2] Group, T. O. M., *OMG Unified Modeling Language (OMG UML): Superstructure*, http://www.omg.org/technology/documents/formal/uml.htm, 2009.

[3] Castaro, S., Fugini, M., Martella, G., and Samarati, P., *Database Security*, Addison-Wesley, 1994.

[4] Dows, D., Rub, J., Kung, K., and Jordan, C., *Issues in discretionary access control*, In: Proc. of IEEE Symposium on Research in Security and Privacy, 1985.

[5] Sandhu, R. S. and Samarati, P., *Access Control: Principles and Practice*, IEEE Communication, Vol. 32, No. 9, 1994, pp. 40–48.

[6] Bertino, E., Bettini, C., and Samarati, P., *A Temporal Access Control Mechanism for Database Systems*, IEEE Transitions on Knowledge and Data Engineering, , No. 8(1), 1996.

[7] Bertino, E., Bonatti, P., and Ferrari, E., *A Temporal Role-based Access Control Model*, ACM Transaction on Information and System Security, , No. 4(3), 2001, pp. 191–233.

[8] Gal, A. and Atluri, V., *An Authorization Model for Temporal Data*, ACM Transaction on Information and System Security, , No. 5(1), 2002.

[9] James, B., Joshi, E., Bertino, U., Latif, A., and Ghafoo, A., *A Generalized Temporal Role-Based Access Control Model*, IEEE Transitions on Knowledge and Data Engineering, , No. 17(1), 2005, pp. 4–23.

[10] Park, J. and Sandhu, R., *The UCON ABC Usage Control Model*, ACM Transactions on Information and System Security, , No. 7, 2004.

[11] Park, J., Zhang, X., and Sandhu, R., *Attribute Mutability in Usage Control*, In: 18th IFIP WG 11.3 Working Conference on Data and Applications Security, 2004.

[12] Sandhu, R. S., Coyne, E. J., Feinstein, H. L., and Youman, C. E., *Role-Based Access Control Models*, IEEE Computer, Vol. 29, No. 2, 1996, pp. 38–47.

[13] Ferraiolo, D., Sandhu, R. S., Gavrila, S., Kuhn, D. R., and Chandramouli, R., *Proposed NIST Role-Based Access Control*, ACM, Transactions on Information and Systems Security (TISSEC), Vol. 4, No. 3, 2001.

[14] Poniszewska-Maranda, A., Goncalves, G., and Hemery, F., *Representation of extended RBAC model using UML language*, In: SOFSEM 2005, LNCS 3381, Publisher: Springer-Verlag Heidelberg, 2005.

[15] Goncalves, G. and Poniszewska-Maranda, A., *Role engineering: from design to evaluation of security schemas*, Journal of Systems and Software, Elsevier, Vol. 81, No. 8, 2008, pp. 1306–1326.

[16] Poniszewska-Maranda, A., *Modeling and design of role engineering in development of access control for dynamic information systems*, Bulletin of the Polish Academy of Sciences, Technical Science, Vol. 61, No. 3, 2013.

[17] Jin, X., *Applying Model Driven Architecture approach to Model Role Based Access Control System*, Ph.D. thesis, Canada, 2006.

[18] Krause, L., *eXtensible Access Control Markup Language (XACML) what is it and why is it important?* http://codingbliss.com/?p=161.

[19] Niemiec, J., Morawiec, M., Ber, J., Drozyński, D., and Wcislo, M., *MDA (Model Driven Architecture) for modeling of access control of information system based on eRBAC model*, Tech. rep., Supervisor: A. Poniszewska-Marańda, Institute of Information Technology, Lodz University of Technology, 2012.