A Model Driven Approach for Secure XML Database Development

Belén Vela[1], Eduardo Fernández-Medina[2], Esperanza Marcos[1] and Mario Piattini[2]

[1] Kybele Research Group. Languages and Information Systems Department
Rey Juan Carlos University
C/ Tulipán, s/n - 28933 Móstoles, Madrid, Spain

[2] Alarcos Research Group. Information Systems and Technologies Department
UCLM-Soluziona Research and Development Institute
University of Castilla-La Mancha
Paseo de la Universidad, 4 – 13071 Ciudad Real, Spain

Abstract. In this paper, we propose a methodological approach for the model driven development of secure XML Databases (DB). This proposal is under the framework of MIDAS, a model driven methodology for the development of Web Information Systems (WIS) based on the Model Driven Architecture (MDA) proposed by the Object Management Group (OMG). The XML DB development process in MIDAS proposes to use as Platform Independent Model (PIM) the data conceptual model and as Platform Specific Model (PSM) the XML Schema model, both of them represented in UML. In this work, such models will be modified to be able to add security aspects if the stored information is considered as critical. On the one hand, it is proposed the use of a UML extension to incorporate security aspects at the conceptual secure DB development (PIM) level and on the other hand, the previously-defined XML schema profile will be modified with the purpose of incorporating security aspects in the logical secure XML DB development (PSM) level. In addition, the semi-automatic mappings to pass from PIM to PSM for secure XML DB will be defined. The development process of a secure XML DB will be shown through a case study: a WIS for the management of hospital information in an XML DB.

1 Introduction

XML is the current standard for information interchange and data transportation between heterogeneous applications. Traditionally, the XML documents' information was stored directly in XML files or in conventional database (DB) systems, by mapping the XML data to relational data stored in relational tables or by using the data types supplied for supporting file management, for example the CLOB (Character Large OBject) type. Now, the XML DBs are emerging as the best alternative to store and manage XML documents. Nowadays, there are different solutions for XML documents' storage, which could be roughly categorized according to [18] into two main groups: native XML DBs like Tamino [16] or eXcelon XIS [3];

Vela B., Fernández-Medina E., Marcos E. and Piattini M. (2006).
A Model Driven Approach for Secure XML Database Development.
In *Proceedings of the 4th International Workshop on Security in Information Systems*, pages 33-46

and XML DB extensions enabling the storage of XML documents within conventional, usually relational or Object-Relational (OR) Database Management Systems (DBMSs) like Oracle which includes, since version 9*i* release 2, new features for the storage of XML (Oracle's XML DB) [15]. Besides, other products such as IBM DB2 XML Extender [9] or Microsoft SQLXML [13] also include extensions for XML storage. In [18] a study of different XML DB solutions is performed.

For most organizations, management, security and confidentiality of information are critical topics [2]. Moreover, as some authors remarked, information security is a serious requirement which must be carefully considered, not as an isolated aspect, but as an element present in all stages of the development life cycle [1,6,8]. Even the Information Systems Audit and Control Foundation insists on the fact that security should be considered in an integral and explicit way in all the development stages of an information system [10]. In the case of the XML DBs, security is also a key aspect that must be explicitly considered and that has to be taken into account in an orthogonal way for the complete development process of this kind of DB [7].

Although there are different ideas for integrating security into the information systems development process, within the scope of DBs, information security is used to be considered only from a cryptographic point of view. Recently, there have appeared works, in which a methodology for relational DB is proposed including security aspects in all stages of the development process [4]. However, to the best of our knowledge, there are no works that deal with security when developing an XML DB.

In this paper, we will integrate the security aspect into the methodological approach for XML DB development [17] framed in MIDAS [11], a model driven methodology for the development of Web Information Systems (WIS). MIDAS proposes the use of standards in the development process as well as the use of UML to model the WIS with independence of the abstraction level and the aspect of the system to model. As UML does not allow us to represent all the necessary models, MIDAS incorporates some existing UML extensions [5] and defines or adapts some new ones, whenever it is necessary [12].

MIDAS proposes a model driven architecture based on the Model Driven Architecture (MDA) proposed by the Object Management Group (OMG) [14] and it considers, when modelling the system, the aspects of *content*, *hypertext* and *behaviour* at the levels of Computation Independent Models (CIMs), Platform Independent Models (PIMs) and Platform Specific Models (PSMs). In Figure 1 we can see the simplified MIDAS model driven architecture, where the CIMs, common to all the system, as well as the different PIMs and PSMs to represent the aspects of content, hypertext and behaviour are proposed.

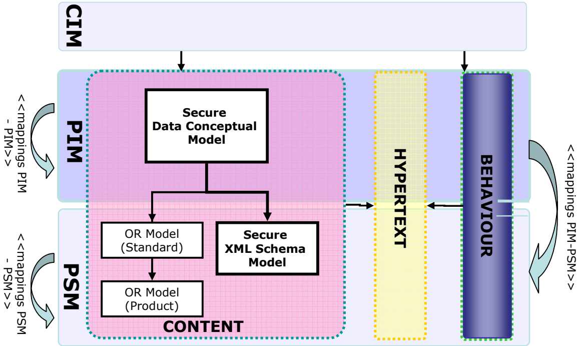


Fig. 1. Simplified MIDAS architecture.

Moreover, a third dimension is considered in MIDAS, which includes all aspects to be taken into account when developing a WIS, as the system architecture or security, which are orthogonal to the ones presented in Fig. 1. Fig. 2 shows the MIDAS architecture with the three mentioned dimensions.

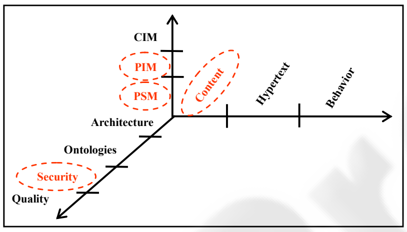


Fig. 2. Dimensions to be considered in the WIS development.

In this work, we will focus on the ***content*** aspect, which corresponds to the traditional concept of a DB and the orthogonal aspect of ***security*** for the ***PIM*** and ***PSM*** levels (see Fig. 2). In the next section, we will focus on the XML DB development process in the frame of MIDAS, where the used data PIM is the data conceptual model and it will be represented with an extended UML class diagram, including the security aspect at this level. This profile will be summed up in section 0. As data PSM in MIDAS, it is proposed to use the OR model or the XML Schema model, depending on the technology that should be used. In this paper, we will show the part corresponding to the secure XML DB development and therefore, the used PSM will be the XML Schema model, using the previously defined profile for XML DB. In section 0, we will present an adaptation of this profile to incorporate specific security aspects into this kind of secure XML DB. Moreover, in section 0, we will show the mappings to pass from the secure data PIM to the secure data PSM that will be the schema of the secure XML DB. These mappings are based on those defined in [17], where the transformation rules to obtain the data PSM are described, but without taking into consideration security aspects. In this paper, we will adapt such rules to obtain the schema of an XML DB including the necessary constraints for security. In section 0, we will present a case study of a WIS for the management and analysis of hospital information, in which our proposal for the development of a secure XML DB has been applied and validated. Finally, in section 0 we will put forward our main conclusions and future works.

2 Development of Secure XML Databases in MIDAS

As we have already mentioned in the previous section, in this work, we focus on the ***content*** aspect of MIDAS that corresponds to the traditional concept of a DB. The development of a DB depends on several aspects; First of all, on the fact that whether there is already a DB within the organization or not, and on the other hand, on the technology to be used, in other words, if we aim at using an OR DB [12] or an XML DB [17]. In addition, it is necessary to take into account the fact that if the DB that we want to develop includes information to be protected, it will be necessary to consider security aspects from the earliest stages of the DB development.

Now, we will describe in a detailed way the XML DB development process from the beginning, including the necessary tasks, models and notations:

- At the ***PIM*** level, the data conceptual design is carried out. To do so, the data conceptual model is used without considering the selected technology since this model is independent of the platform. This data PIM is represented through a UML class diagram. In our proposal, we will use, as we have mentioned before, an extended UML class diagram to be able to represent security aspects together with a set of security constraints that have been expressed through OSCL language [5], as we will see in the subsection 0.
- At the ***PSM*** level, the data logical design is performed. Here, it is necessary to take into account the selected technology that, in our case, is an XML DB. We will start from the secure data PIM obtained at the previous level and we will apply the mappings summarized in subsection 0. The secure data PSM will be represented through an XML schema in extended UML (see subsection 0). In this case, the DB schema will be the obtained XML schema that will take into account the necessary security aspects.

Table 1 summarizes the tasks, models and notations to be performed when developing a secure XML DB.

Table 1. Development process of a secure XML DB.

	MIDAS: Secure XML DB Development		
Level	**Tasks**	**Models**	**Notation**
PIM	Secure Data Conceptual Design	Secure Data Conceptual Model	Class Diagram (Extended UML)
PSM	Secure Data Logical Design	Secure Data Logical Model	XML Schemas (MIDAS-UML)

2.1 Secure Data PIM

To develop a secure data PIM, a secure UML profile has been developed (for more details, see [5]). The defined UML profile allows us to classify both data and users according to different classification criteria with the purpose of performing the mandatory access control and a simplified role based access control. These criteria are as follows:

- **Security levels**: They allow us to define a hierarchy of levels such as the traditional in military contexts: unclassified, confidential, secret and top secret.
- **User roles**: They allow us to define a hierarchical set of user roles that represent the hierarchical functions within an enterprise.
- **User categories**: They allow us to define a horizontal organization or classification (non hierarchical) of groups of users.

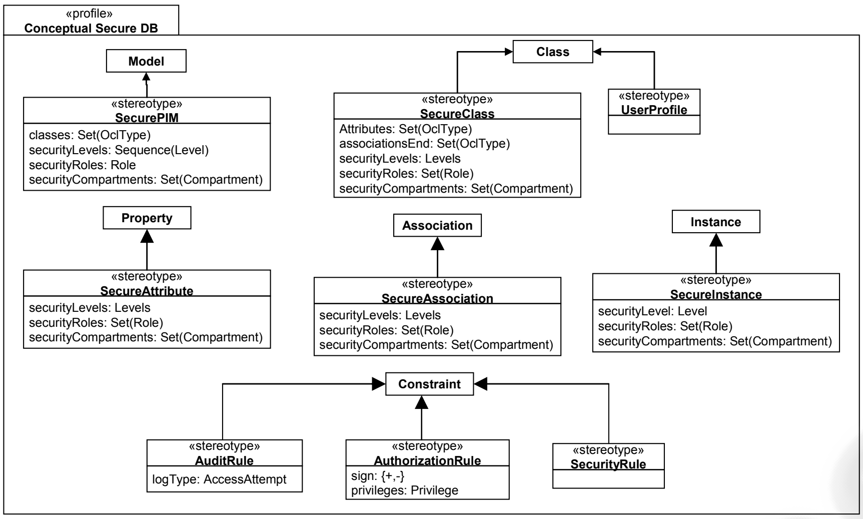
In addition to this classification information, this profile allows us to define three kinds of constraints:

- **Data dynamic classification rules**: They allow us to define the classification data of different instances depending on the value of one or several attributes of the instances.
- **Authorization rules**: They allow us to define which users will be allowed to access to which data and to perform which actions depending on a condition expressed in OCL.
- **Audit rules**: They specify situations in which it is interesting to register an audit trail to analyze which users have accessed (or have tried to access) to information. To do so, conditions expressed in OCL are defined. For the definition of all these elements, we consider the UML profile known as *Conceptual Secure DB* (extension of UML and OCL to design secure DBs), that is composed of a set of data types, tagged values and stereotypes together with the definition of a set of well-formedness rules.

The package containing all the stereotypes defined within this UML profile can be analyzed in Fig. 3. These stereotypes can be classified into three categories:

- The necessary stereotypes to represent security information in the *model elements* (the model itself, the classes, the attributes, associations and instances).
- The necessary stereotypes to model the *security constraints* to a) define the dynamic classification of any element of the model, b) define authorization rules and c) define audit rules depending on the access types, perhaps of any condition expressed in OCL.
- The *UserProfile* stereotype that is necessary to specify constraints depending on any property of a user or a group of users, for instance, depending on the citizenship, age, etc.

A detailed description of all these stereotypes as well as the tagged values that have been defined for these stereotypes can be found in [5].


«profile»
Conceptual Secure DB
Model
«stereotype»
SecurePIM
classes: Set(OclType)
securityLevels: Sequence(Level)
securityRoles: Role
securityCompartments: Set(Compartment)
Class
«stereotype»
SecureClass
Attributes: Set(OclType)
associationsEnd: Set(OclType)
securityLevels: Levels
securityRoles: Set(Role)
securityCompartments: Set(Compartment)
«stereotype»
UserProfile
Property
«stereotype»
SecureAttribute
securityLevels: Levels
securityRoles: Set(Role)
securityCompartments: Set(Compartment)
Association
«stereotype»
SecureAssociation
securityLevels: Levels
securityRoles: Set(Role)
securityCompartments: Set(Compartment)
Instance
«stereotype»
SecureInstance
securityLevel: Level
securityRoles: Set(Role)
securityCompartments: Set(Compartment)
Constraint
«stereotype»
AuditRule
logType: AccessAttempt
«stereotype»
AuthorizationRule
sign: {+,-}
privileges: Privilege
«stereotype»
SecurityRule

2.3 Mappings to Pass from Secure Data PIM to Secure Data PSM

In the same way that methodologies for relational or OR DBs propose some rules for the transformation of a conceptual schema into a standard logical one, in MIDAS, mappings to pass from the data PIM to the data PSM are proposed. In this work, we have defined the necessary transformation rules to obtain a secure data PSM from the secure data PIM. Now, we will show these rules to collect the characteristics of security taking as a basis the work of [17], where the different mappings to obtain the schema of an XML DB were defined.

- **Transformation of the secure data PIM:** The data conceptual model, that is, the *secure data PIM*, is transformed, at the PSM level, into an XML schema called '*Secure Data PSM*'. This will be represented with a UML package stereotyped with <<Secure XML SCHEMA>> including all components of the secure data PSM. Furthermore, it will contain the security attributes (*securityLevel*, *securityRoles* and *securityCompartments*) of the secure data PIM. These attributes will be defined within the XML schema as global elements. These security attributes could have been included as schema attributes but if they were represented in such a way, they would not be considered first order elements and the fact that they could have a multiple maximum cardinality could not be collected either.
- **Transformation of the *User Profile* class:** This class includes the information that we want to record about one or several users. It will be transformed by including a global element stereotyped with <<User Profile ELEMENT>>, that will contain a sequence complexType with all class attributes as subelements.
- **Transformation of secure classes:** In a generic way, a UML class is transformed into an element of the XML schema with the same name as the class it comes from [17]. To transform secure UML classes, stereotyped with <<SecureClass>>, we have to include the secure characteristics that they have too. Secure classes can have three specific attributes: *securityLevel*, *securityRoles* and *SecurityCompartments*. They will be transformed into secure elements stereotyped with <<Secure ELEMENT>>. Each secure element will contain a complex type of sequence type, that will contain as subelements, among others, the secure attributes, indicating with the subelements attribute *maxOccurs* the number of possible instances of the security attributes.
- **Transformation of secure attributes:** Due to the fact that the attributes of a class, according to the proposal of [17], are transformed as subelements of the element that represents the UML class to which those attributes belong, if an attribute has its own security attributes associated with it, these attributes will be represented as subelements of the element that represents the corresponding attribute. Thus, the security attributes defined within an attribute will be transformed into <<Secure ELEMENT>> subelements.
- **Transformation of secure associations:** Regarding the transformation of associations, in [17] it was carried out a detailed study of the most appropriate way to map these associations at the PSM level. The associations between two classes are transformed, in a generic way, by including a subelement in one of the elements, corresponding to one of the classes implied in the relationship with one or several references to the other element implicated in the association. If it was a

secure association, this subelement would have subelements to represent the corresponding security attributes (*securityLevel*, *securityRoles*, *securityCompartment*) stereotyped as <<Secure ELEMENT>>.

- **Transformation of security constraints:** When transforming the security constraints that had been defined at the PIM level, these can be defined for any element (model or class) although it is normal to define them at the class level. If they are defined at the model level, global elements to collect this fact will be created. In the rest of the cases, there will be created subelements of the elements they depend on. There are different types of security constraints:

a) **Audit Rules**: They will be transformed by creating a subelement stereotyped with <<AuditRule>> with the name of “AuditRule_” plus the number of the rule. This element will be of the complexType and it will contain a sequence formed by two elements: One *AuditRuleType* element of simple Type of the *string* base type with a constraint of enumeration type with the values *all*, *frustratedAttempt*, *successfullAccess*; and another element *AuditRuleCondition* that will be an element of *string* type, that will contain the XPath expression associated with the expression in OCL.

```
<complexType>
     <sequence>
        <element name= “AuditRuleType”>
           <simpleType>
                  <restriction base= ”string”>
                     <enumeration value= ”all”/>
                     <enumeration value= ”frustatedAttempt”/>
                     <enumeration value= ”successfullAccess”/>
                  </restriction>
            </simpleType>
        </element>
        <element name= “AuditRuleCondition” type=”string”/>
  </sequence>
</complexType>
```

b) **Authorization Rules:** They will be transformed by creating a subelement stereotyped with <<AuthorizationRule>> with the name “AuthorizationRule_” plus the number of the rule. This element will be of complexType and it will contain a sequence formed by three elements: An *AuthorizationRuleSign* element of simpleType of *string* base type with a constraint of enumeration type with the values: + or - ; another *AuthorizationRulePrivileges* element of simpleType of *string* base type with a constraint of enumeration type with the values: *read*, *insert*, *delete*, *update* and *all*; and an *AuthorizationRuleCondition* element of string type that will contain the XPath expression associated with the expression in OCL.

```
<complexType>
      <sequence>
         <element name= “AuthorizationRuleSign”>
               <simpleType>
                     <restriction base=”string”>
                        <enumeration      value=”+”/>      <enumeration
                      value=”-”/>
                     </restriction>
               </simpleType>
         </element>
         <element name= “AuthorizationRulePrivileges”>
               <simpleType>
                     <restriction base=”string”>
                        <enumeration value=”read”/>
                        <enumeration value=”insert”/>
                        <enumeration value=”delete”/>
```

```
                                <enumeration value=”update”/>
                                <enumeration value=”all”/>
                             </restriction>
                       </simpleType>
                 </element>
                 <element name= “AuthorizationRuleCondition” type=”string”/>
           </sequence>
         </complexType>
```

c) **Security Rule:** The dynamic classification of any PIM element will be transformed by creating a subelement stereotyped with <SecurityRule>>, with the name “SecurityRule_” plus the number of the rule. This element will be of complexType and it will contain one element of string type with the XPath expression associated with the OCL expression.

```
<complexType>
     <sequence>
      <element name= “SecurityRuleCondition” type=”string”/>
     </sequence>
</complexType>
```

In Table 2, we will summarize the transformation rules to pass from the data PIM to the corresponding PSM using XML DB technology. In [17], these rules are detailed, but without including the security aspect.

Table 2. Transformation rules to pass from the secure data PIM into the secure data PSM.

Data PIM	Data PSM
Secure Data PIM	**Secure Data PSM <<Secure XML Schema>>**
securityLevels attribute	Global element of the schema (maxOccurs=unbounded)
securityRoles attribute	Global element of the schema
securityCompartments attribute	Global element of the schema (maxOccurs=unbounded)
Secure Class	**Secure XML Element <<Secure Element>>**
securityLevels attribute	Subelement (maxOccurs=unbounded)
securityRoles attribute	Sublement
securityCompartments attribute	Sublement (maxOccurs=unbounded)
User Profile Secure Class	**Global XML Element <<User Profile Element>>**
Attribute	**Subelement**
securityLevels atributte	Subelement (maxOccurs=unbounded)
securityRoles atributte	Sublement
securityCompartments atributte	Sublement (maxOccurs=unbounded)
Association	
securityLevels atributte	Subelement of the association element (maxOccurs=unbounded)
securityRoles atributte	Subelement of the association element
securityCompartments atributte	Subelement of the association element (maxOccurs=unbounded)
Constraint	Subelememt with complexType of sequence type with subelements
AuditRule	- AuditRuleType of simpleType with enumeration constraint - AuditRuleCondition of string type containing the XPath expression associated with the OCL expression
AuthorizationRule	- AuthorizationRuleSign of simpleType with enumeration constraint - Privileges of simpleType with enumeration constraint - AuthorizationRuleCondition of string type that will contain the XPath expression associated with the OCL expression
SecurityRule	- Subelement of string type with XPath expression

3 Case Study

In this section, we will apply our UML extension to develop the secure data PIM in a case study in the context of hospital information systems.

Later, through the transformation rules that we have defined in section 0, we will create a secure XML DB by using the other UML extension for secure data PSM.

For this example, we have defined a simplified users' hierarchy (see Fig. 5, left-hand side) composed of a generic role "Hospital Employee" that is separated into the roles "Sanitary" personnel and "Non Sanitary" personnel. The first of these roles is divided into "Doctor" and "Nurse", while the second one is specialized into "Maintenance" and "Administrative". Additionally, for this case study, we have considered three security levels: "Unclassified", "Secret" and "Top Secret" (see Fig. 5, right-hand side). For the sake of simplicity, we have not defined users' categories, but we could have considered regional categories (region Spain, region Argentina, etc), professional criteria (Paediatrics, Surgery, etc.) and so on.

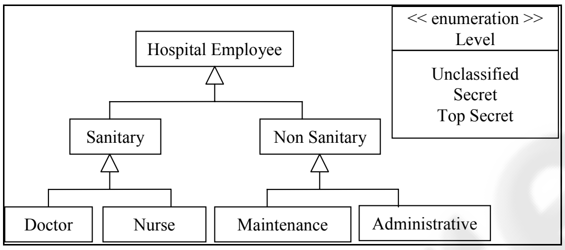


Fig. 5. Users' roles hierarchy and security levels.

Fig. 6 shows us the secure data PIM, represented through a class diagram containing many details. However, we will only explain some classes to be able to focus our attention on security aspects. The classes of the diagram that we will mention are (the rest are obvious and we will not study them in depth) the following: *UserProfile*, *Patient* and *Admission*. The *UserProfile* class contains the different information fields that are registered for all users that will have access to the DB. The *Patient* class contains information of all hospital patients and can be accessed by the users that have at least the *secret* security level and perform administrative or sanitary roles. The *Admission* class contains information of all hospital admissions and can be accessed by users that perform sanitary or administration tasks and that have also a *secret* security level. In this class, we specify that the attributes *diagnosis, result and treatment* can only be accessed by sanitary personnel (and not by administrative one) and that the attribute *cost* can be seen by administrative personnel and not by sanitary personnel. There is an association between the classes *Patient* and *Admission* and besides, we can see that there are many security constraints associated with these classes.

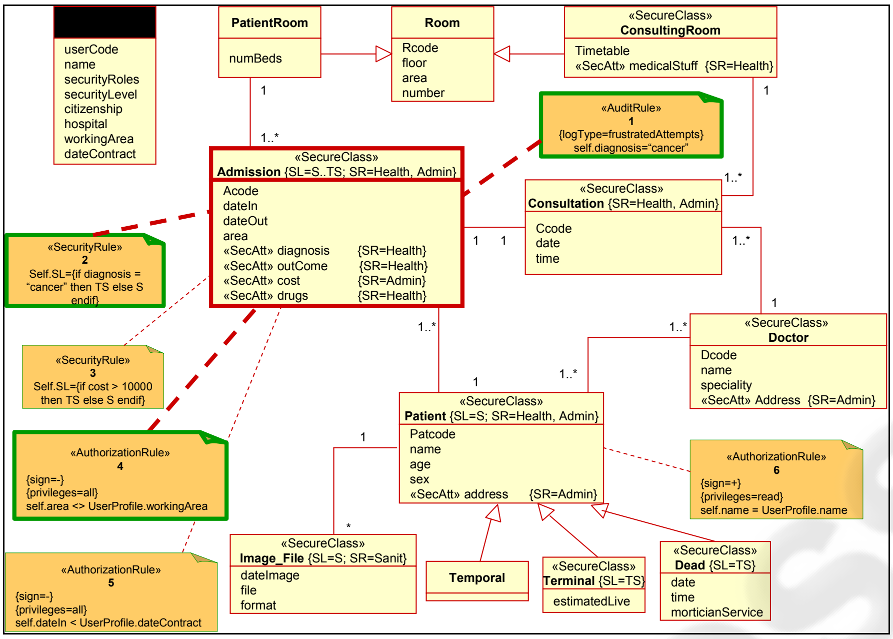


Fig. 6. Data PIM with security constraints.

We have identified each security constraint with a number. The detail of each of them is as follows:

1. It is a stereotyped constraint that represents an audit rule. This rule specifies that all accesses denied by the access control mechanism (tagged value equal to *frustratedAttempts*), corresponding to instances of the *Admission* class whose value is "cancer" (OCL expression self.diagnosis='cancer'), should be registered for a future audit. This audit rule will help us identify possible attackers that try to access to confidential information without having the necessary permissions.
2. This constraint defines a dynamic security rule that specifies the security level of each instance of the *Admission* class. If the value of the diagnosis attribute is "cancer" then the security level will be top secret. On the contrary, it will be only secret.
3. This is another dynamic security rule for the Admission class. In this case, the security level will depend on the value of the cost attribute that will indicate the value of the hospital service.
4. The concept modelling the fourth constraint is an authorization rule. We could deny the access (symbol = -) to the admission information to users whose work area is different from the sanitary area of a particular admission (self.area <> UserProfile.workingArea).
5. For confidentiality reasons, we could deny the access (symbol = -) to the admission information to all doctors whose date of contract with the hospital is later than the date of admission of patients (self.date of admission < UserProfile.date of contract). This constraint specifies that authorization rule.

6. Finally, we could consider patients as special users of the system in the sense that they could have access only to their own personal data. In this case, it is necessary to specify a positive authorization rule (symbol = +) indicating as a condition that the user's name has to be equal to the patient's name (self.name = UserProfile.name). We can see that using the UML extension, it is possible to specify a wide range of confidentiality constraints in the secure data PIM.

When transforming the secure data PIM of Fig. 6, we will apply the rules defined in subsection 0 and we will obtain data PSM. In Fig. 7, we can see part of the secure data PSM that we have obtained.

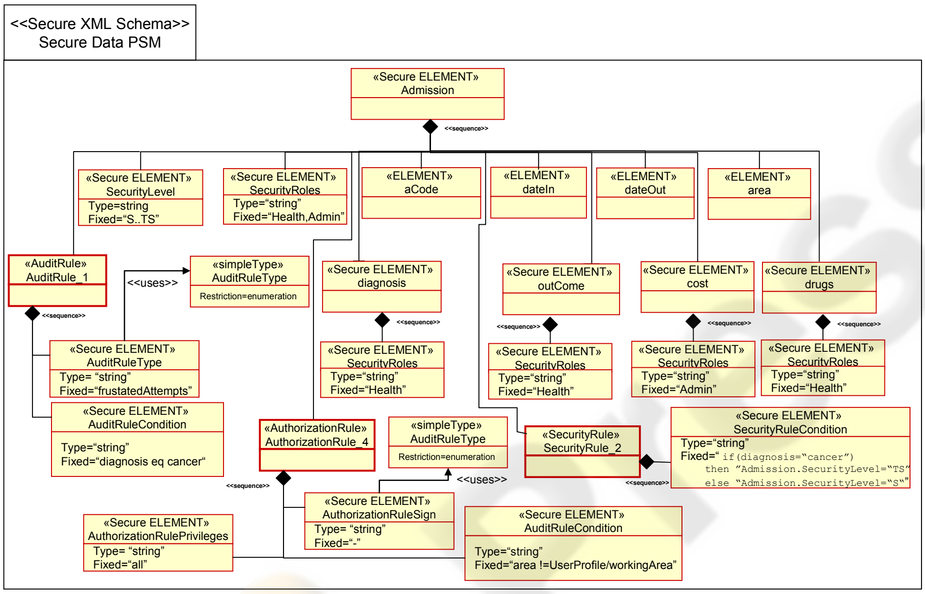

<<Secure XML Schema>>
Secure Data PSM
«Secure ELEMENT» Admission
<<sequence>>
«Secure ELEMENT» SecurityLevel
Type=string
Fixed="S..TS"
«Secure ELEMENT» SecurityRoles
Type="string"
Fixed="Health,Admin"
«ELEMENT» aCode
«ELEMENT» dateIn
«ELEMENT» dateOut
«ELEMENT» area
«AuditRule» AuditRule_1
<<uses>>
«simpleType» AuditRuleType
Restriction=enumeration
«Secure ELEMENT» diagnosis
«Secure ELEMENT» outCome
«Secure ELEMENT» cost
«Secure ELEMENT» drugs
<<sequence>>
<<sequence>>
<<sequence>>
<<sequence>>
<<sequence>>
«Secure ELEMENT» AuditRuleType
Type= "string"
Fixed="frustatedAttempts"
«Secure ELEMENT» SecurityRoles
Type="string"
Fixed="Health"
«Secure ELEMENT» SecurityRoles
Type="string"
Fixed="Health"
«Secure ELEMENT» SecurityRoles
Type="string"
Fixed="Admin"
«Secure ELEMENT» SecurityRoles
Type="string"
Fixed="Health"
«Secure ELEMENT» AuditRuleCondition
Type="string"
Fixed="diagnosis eq cancer"
«AuthorizationRule» AuthorizationRule_4
«simpleType» AuditRuleType
Restriction=enumeration
«SecurityRule» SecurityRule_2
«Secure ELEMENT» SecurityRuleCondition
Type="string"
Fixed=" if(diagnosis="cancer")
then "Admission.SecurityLevel="TS"
else "Admission.SecurityLevel="S""
<<sequence>>
<<uses>>
<<sequence>>
«Secure ELEMENT» AuthorizationRuleSign
Type= "string"
Fixed="-"
«Secure ELEMENT» AuthorizationRulePrivileges
Type= "string"
Fixed="all"
«Secure ELEMENT» AuditRuleCondition
Type="string"
Fixed="area !=UserProfile/workingArea"

Fig. 7. Part of the secure data PSM.

First of all, we have created a UML package stereotyped with <<Secure XML Schema>> known as Secure Data PSM, that will include all the obtained elements when transforming the secure data PIM. In our case, due to space constraints, we have focused on the transformation of the *Admission* class as well as that of some constraints, in concrete, the following rules: *AuditRule1*, *AuthorizationRule4* and *SecurityRule2,* following the steps indicated in section 0.

When transforming the *Admission* secure class, firstly, it will be created an element stereotyped with <<Secure ELEMENT>> that will contain a complexType with the subelements that represent the attributes of the class (*aCode*, *dateIn*, *dateOut*, *area*). In addition, it will contain the subelements that correspond to the security attributes (*Security Level* and *Security Roles*) and stereotyped with <<Secure ELEMENT>>. The value of these attributes will be collected with the attribute *fixed* of these elements. The attributes *diagnosis, result, cost and treatment* of such class are secure attributes and therefore, they will be represented as secure elements. For this reason, they have their own subelements that represent secure attributes.

4 Conclusions and Future Work

Nowadays, there are different solutions for the storage of XML data but there is not a methodology for the systematic design of XML DB that incorporates security in all the development process if the information to be stored is considered as critical.

In this work, we have integrated the security aspect into the methodological approach for the development of an XML DB in the framework of MIDAS, a model-driven methodology for the development of WIS based on MDA. According to the specified development process for secure XML DB, for the secure data PIM, a UML extension to incorporate security aspects at the conceptual level is used, while for the secure data PSM we have modified the previously-defined XML DB profile with the goal of incorporating security aspects. Moreover, we have defined transformation rules to pass from secure data PIM to secure data PSM that will be the secure XML DB schema. From this logical secure XML DB (PSM), we will obtain in a semi-automatic way the code for the specific XML DB product that we want to use. In future works, we will study in detail different XML DB products, in order to analyze which of them take into account security aspects and how.

We have developed a case study for the management of hospital information to validate our proposal; in this paper, we have shown part of it, in which the secure XML DB schema is defined for a reduced part of the developed secure data PIM.

Now we are working in several lines to extend the proposal of this paper. One of them, in which we have already started to work, is the automatization of the transformations of the constraints expressed in OCL at the PIM level to convert them into XPath language. Furthermore, we have the purpose of performing the implementation of several case studies to detect new needs as well as to analyze the advantages of incorporating security aspects provided by the different XML DB administrators, not only native but also the XML extensions that DB systems have. On the other hand, we are going to include the security aspect in the module for the semi-automatic development of XML DB of the tool CASE that we are developing.

Acknowledgements

This research has been carried out in the framework of the following projects: GOLD financed by the Spanish Ministry of Education and Science (TIN2005-00010/), DIMENSIONS (PBC-05-012-2) financed by the FEDER and by the “Consejería de Ciencia y Tecnología of the Junta de Comunidades de Castilla-La Mancha” and the RETISTIC network (TIC2002-12487-E) financed by the “Dirección General de Investigación” of the Spanish Ministry of Science and Technology.

References

1. Devanbu, P. and Stubblebine, S. *Software engineering for security: a roadmap*, in: A. Finkelstein (Ed.), The Future of Software Engineering, ACM Press pp. 227-239, 2000.
2. Dhillon, G. and Backhouse, J. *Information System Security Management in the new Millennium.* Communications of the ACM. 43, 7. pp. 125-128, 2000.

3. eXcelon Corporation. *Managing DXE. System Documentation Release 3.5.* eXcelon Corporation. Burlington. Retrieved from: www.excelon.corp.com, 2003.
4. Fernández-Medina, E. and Piattini M. *Designing secure databases.* Information & Software Technology 47(7), pp. 463-477. 2005
5. Fernández-Medina, E. and Piattini, M. *Extending OCL for Secure Database Design*. In Int. Conference on the Unified Modeling Language (UML 2004). Lisbon (Portugal), October, 2004. Springer-Verlag, LNCS 3273, pp. 380-394. 2004.
6. Ferrari E. and Thuraisingham B., *Secure Database Systems*, in: M. Piattini, O. Díaz (Ed.), Advanced Databases: Technology Design. Artech House, 2000.
7. Ferrari, E. *Secure DataBase Systems.* Second Meeting. RETISBD. Murcia (Spain), June 2001.
8. Ghosh, A., Howell C., Whittaker J., *Building software securely from the ground up*, IEEE Software 19 (1) (2002), pp. 14-17, 2002.
9. IBM Corportation. *IBM DB2 Universal Database -XML Extender Administration and Programming*, Product Documentation Version 7. IBM Corporation, 2000.
10. ISACF, Information Security Governance. *Guidance for Boards of Directors and Executive Management*, Information Systems Audit and Control Foundation, USA, 2001.
11. Marcos, E., Vela, B., Cáceres, P. and Cavero, J.M. *MIDAS/DB: a Methodological Framework for Web Database Design.* DASWIS 2001. Yokohama (Japan), November, 2001. Springer-Verlag, LNCS 2465, pp. 227-238, 2002.
12. Marcos, E., Vela, B. and Cavero J.M. *Methodological Approach for Object-Relational Database Design using UML.* Journal on Software and Systems Modeling (SoSyM). Springer-Verlag. Ed.: R. France and B. Rumpe. Vol. SoSyM 2, pp.59-72, 2003.
13. Microsoft Corporation. *Microsoft SQL Server - SQLXML 2.0*, System Documentation. 2000.
14. OMG. *MDA Guide Version 1.0.* Document number omg/2003-05-01. Ed.: Miller, J. and Mukerji, J. Retrieved from: http://www.omg.com/mda, 2003.
15. Oracle Corporation. *Oracle XML DB. Technical White Paper.* Retrieved from: www.otn.com, 2003.
16. Software AG. *Tamino X-Query. System Documentation Version 3.1.1*. Software AG, Darmstadt, Germany. Retrieved from: www.softwareag.com, 2001.
17. Vela, B., Acuña, C. and Marcos, E. *A Model Driven Approach for XML Database Development*, 23rd. International Conference on Conceptual Modelling (ER2004). Shanghai (China), November, 2004. Springer Verlag, LNCS 3288, pp. 780-794. 2004.
18. Westermann, U. and Klas W. *An Analysis of XML Database Solutions for the Management of MPEG-7 Media Descriptions*. ACM Computing Surveys, Vol. 35 (4), pp. 331-373, December, 2003.