Available online at www.sciencedirect.com
ScienceDirect
Decision Support Systems 42 (2006) 1270–1289
Decision Support Systems
www.elsevier.com/locate/dss

Access control and audit model for the multidimensional modeling of data warehouses

Eduardo Fernández-Medina [a,*], Juan Trujillo [b], Rodolfo Villarroel [c], Mario Piattini [a]

[a] *Departamento de Informática, Universidad de Castilla-La Mancha, Spain*
[b] *Departamento de Lenguajes y Sistemas Informáticos, Universidad de Alicante, Spain*
[c] *Departamento de Computación e Informática, Universidad Católica del Maule, Chile*

Received 24 November 2004; received in revised form 19 October 2005; accepted 24 October 2005
Available online 6 December 2005

Abstract

Due to the sensitive data contained in Data Warehouses (DW), it is essential to specify security measures from the early stages of the DW design and enforce them. Traditional access control models for transactional (relational) databases, based on *tables*, *columns* and *rows*, are not appropriate for DWs. Instead, security and audit rules defined for DWs must be specified based on the multidimensional (MD) modeling used to design data warehouses. Current approaches for the conceptual modeling of DWs do not allow us to specify security and confidentiality constraints in the conceptual modeling phase. In this paper, we propose an Access Control and Audit (ACA) model for DWs by specifying security rules in the conceptual MD modeling. Thus, we define authorization rules for users and objects and we assign sensitive information rules and authorization rules to the main elements of a MD model (e.g., facts or dimensions). Moreover, we also specify certain audit rules allowing us to analyze user behaviors. To be able to include and use our ACA model in the conceptual MD modeling, we extend the Unified Modeling Language (UML) with our ACA model, thereby allowing us to design secure MD models. Finally, to show the benefit of our approach, we apply our approach to a health care case study.

Keywords: Data warehouses; Secure multidimensional modeling; Access control; Audit; UML

1. Introduction

Data Warehouses (DW), Multidimensional (MD) Databases, and On-Line Analytical Processing (OLAP) applications are used in conjunction to form a highly powerful mechanism for discovering crucial business information in strategic decision-making processes. From the early nineties, data warehouses have been considered to be the key aspect of success for any Decision Support System (DSS) [21]. Even now, a decade later, data warehouses are still seen as being the most important issue for the next generation of Decision Support Systems [6,35,44]. Breaches in security and confidentiality, however, continue to pose a threat, due to the high sensitivity of the information that can be discovered in a data warehouse.

MD modeling is the foundation of DWs, MD databases and OLAP applications. Sometimes MD models also store information regarding private or personal aspects of individuals, such as identification

* Corresponding author. Universidad de Castilla-La Mancha, Escuela Superior de Informática, Paseo de la Universidad 4, 13071 Ciudad Real, Spain.
E-mail address: eduardo.fdezmedina@uclm.es (E. Fernández-Medina).

0167-9236/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.dss.2005.10.008

data, medical data or even religious beliefs or ideologies. Many governments are decidedly concerned about privacy (due also to the fact that this matter is a pressing concern for many businesses today). Hence they promulgate laws for the protection of individual privacy—take, for example, the United States' HIPAA-Health Insurance Portability and Accountability Act, which regulates the privacy of personal health care information, the GLBA (Gramm–Leach–Bliley Act, also known as the Financial Modernization Act), the Sarbanes–Oxley Act, and the EU's Safe Harbour Law. These laws usually require strict technical security measures for guaranteeing privacy, and failure to comply with them is normally strictly sanctioned, with severe penalties being imposed. Although some corporations incorporate support designed to meet some of these security requirements [34], there are not any models or methods to support them from the very first stages of the data warehouse development life cycle.

Therefore, taking into account that the very survival of organizations frequently depends on the correct management, security and confidentiality of information [12], and bearing in mind also the extreme significance of the information that users can discover by using these kinds of applications, it is crucial to specify confidentiality measures in the MD modeling process, and to enforce them. As some authors have remarked [11,16,47], information security is a serious requirement which must be carefully considered, not as an isolated aspect, but as an element which manifests itself at all stages of the development lifecycle, from the requirement analysis to implementation and maintenance.

We can consider three technologies that have been widely used to protect information against improper disclosure or modifications. Authentication, access control and audit jointly provide the foundation for information security [42]. Authentication establishes the identity of one party to another. Access control determines what one party will allow another one to do with respect to resources and objects mediated by the former. Access control usually requires authentication as a prerequisite. The Audit process gathers data about activities in the system and analyzes it in order to discover security violations or to diagnose their cause. Authentication is a mechanism that is design-independent and relies more on the company policies, and therefore, it is beyond the scope of this paper. Nevertheless, access control and audit have an important component of design. In fact, access control and audit considerations should be taken into account throughout the design process, beginning at its early stages, and not just when the system is completely developed [19]. Consequently, we claim that these aspects should be included in the conceptual modeling of a system, as the sooner we deal with security aspects, the more willing we are to consider all the major security aspects in the final system implementation.

In data warehouses, this needed is even much harder as the MD modeling used to design the data warehouse is the same one that the final user will use to query the data warehouse by front-end tools such as OLAP tools. However, this does not mean that a final user may have access to all the MD elements. Therefore, we believe that the data warehouse designer, in addition to design its structure by means of a MD modeling, should also specify access control and audit rules together with the conceptual MD modeling at the same in order to protect the data warehouse information against further improper disclosure.

As other authors point out [21,26–28,36], even though most data warehouses are implemented on top of relational DBMS, security measures and access control models specified for transactional (relational) databases are not appropriate for data warehouses. The main reason is that these models specify security measures in terms of database tables, rows and columns, which do not allow us to specify security measures for the wide variant of users accessing a DW. Instead, security measures for data warehouses must be defined on a MD basis, since DW users query the DW in terms of facts, dimensions, classification hierarchy levels and so on. Furthermore, DWs usually imply having different levels of aggregation for the same facts depending on the various classification hierarchy levels defined along dimensions. The flexibility of OLAP tools for DWs make it easy and flexible for a final user to access an aggregation level by navigating from different dimension hierarchy levels. Moreover, the peculiarities of OLAP operations (e.g., roll-up, drill-down) based on these classification hierarchy levels differ so much from the classical database operations (e.g., insert, delete, update, select), that other access mechanisms must be used. Finally, other specific tools for the final user or DWs such as data mining tools need to access a vast amount of data for a proper forecasting, thus threatening sensitive information. Therefore, these and other particular situations for data warehouses (covered throughout the paper) can only be correctly covered by defining security measures on the conceptual MD model and enforce them.

We focus this paper on the conceptual MD modeling, thus their logical modeling and implementation are

beyond the scope of this work. To build a secure conceptual MD model, the designer should first develop the conceptual MD model, *and then based on this model, a security model has to be defined, and finally, the security details of that security model should be integrated into the whole conceptual MD model*.

In this paper, we set out an Access Control and Audit (ACA) model that allows us to specify access control and audit considerations when carrying out the conceptual MD modeling of data warehouses. Since there are some proposals of MD models and modeling processes for DWs, the ACA model should be independent, but easily adaptable to any of these proposals. We have chosen one, based on UML [31,48], that easily allows us to model all main MD properties at the conceptual level. The next step was to define a UML extension that allows us to specify all concepts that are previously defined in the ACA model following the previously mentioned UML approach. To the best of our knowledge, this is the first approach for the conceptual design of DWs to consider security and audit aspects as part of the conceptual MD model. With this proposed model we are able to restrict the further access of users to the data that they must not have access to.

The remainder of this paper is structured as follows: Section 2 introduces related work. Section 3 gives a brief summary of the conceptual approach for MD modeling in which our study is based. Section 4 presents our ACA model. Section 5 gives a short introduction to the UML extension for the conceptual modeling of secure DWs which we use. Section 6 presents a case study and applies our ACA model and UML extension for secure MD modeling. Finally, Section 7 presents the main conclusions and sketches the immediate future work.

2. Related work

As this paper treats different research topics, the related work is organized in four subsections. We first deal with the MD modeling approaches proposed for data warehouse design. Then we study work related to the integration of security into the design process of Information Systems or databases in general. In the third subsection, we analyze current access control models, and finally we study specific issues regarding security in data warehouses and OLAP. This related work concludes with the relevant aspect that there is not either any approach for designing secure conceptual MD models and specific access control and audit models for data warehouses.

2.1. Multidimensional modeling

Lately, several MD data models have been proposed. Some of them fall into the logical level (such as the well-known star-schema by R. Kimball [27]. Others may be considered as formal models as they provide a formalism for the consideration of the main MD properties. A review of the most relevant logical and formal models can be found in [5] and [1].

In this section, we will only make brief reference to the most relevant models that we consider "pure" conceptual MD models. These models provide a high level of abstraction for the main MD modeling properties at the conceptual level and are totally independent from implementation issues. One outstanding feature provided by these models is that they provide a set of graphical notations (such as the classical and well-known Extended Entity-Relationship model) which facilitates their use and reading. These are as follows: *The Dimensional-Fact (DF) Model* by Golfarelli et al. [18], *The Multidimensional/ER (M/ER) Model* by Sapia et al. [43], *The starER Model* by Tryfona *et al.* [49], the Model proposed by Hüseman et al. [20], and *The Yet Another Multidimensional Model (YAM^2)* by Abelló et al. [2]. Unfortunately, none of them has been accepted as a standard for the conceptual modeling of Data Warehouses. Recently, another approach [31,48] has been proposed as an object-oriented (OO) conceptual MD modeling approach. This proposal is a profile of the UML, which uses its standard extension mechanisms (stereotypes, tagged values and constraints).

However, none of these approaches for MD modeling considers security as an important issue in their conceptual models, and consequently they do not solve the problem of modeling security from the early stages of a DW project.

2.2. Security integration into the design process

There are a few proposals which attempt to integrate security into conceptual modeling, such as the Semantic Data Model for Security [46] and the Multilevel Object Modeling Technique [33], but they are partial (since they do not cover the complete development process). More recent proposals are UMLSec [25] and SecureUML [30] where UML is extended to develop secure systems. These approaches are interesting, but they only deal with information systems (IS) in general, whilst conceptual database and DW design are not considered. Moreover, a methodology and a set of models have been proposed [13] in order to design

secure databases for implementation with Oracle9i Label Security (OLS) [29]. This approach, based on the UML, is relevant because it considers security aspects in all stages of the database development process, from requirement-gathering to implementation. Together with the previous methodology, the proposed Object Security Constraint Language (OSCL) [14], based on the Object Constraint Language (OCL) [51] of UML, allows us to specify security constraints in the conceptual and logical database design process, and to implement them in a specific database management system (DBMS), OLS. Nevertheless, the previous methodology and models do not consider the design of secure MD models for DWs, and therefore, are not appropriate for the representation of the peculiarities of DWs.

2.3. Access control models

Many proposals have been developed with the aim of protecting information against improper disclosure. All of them make use of the particularities of the systems they deal with, such as the types of objects, subjects, privileges, signs, etc. For instance, there are authorization models for data archives [7], database systems [4,37], XML documents [10], distributed hypertext systems [40], web services [45], and even for multimedia documents [9] and workflows [3]. Nevertheless, although many authorization models that allow us a flexible and simple specification of authorizations have been proposed, they rely on the particular properties of the underlying data model [24]. Thus, due to the peculiarities of data warehouses and the underlying MD modeling and its terms facts, dimensions, classification hierarchies and so on, these authorization models cannot be easily extended to include security measures into the conceptual MD modeling of data warehouses.

2.4. Security and access control models for data warehouses

As previously described, the peculiarity of DWs and the MD model and its terms (facts, dimensions, classification hierarchies and so on) used for both designing and querying DWs, makes it necessary to deal with specific access control and audit models for DWs.

In the literature, we can find several initiatives for the inclusion of security in DWs [26,38]. Many of them are focused on aspects related to access control, multilevel security, the applications of these to federated databases, applications using commercial tools and so on. These initiatives refer to specific aspects that allow us to improve DW security in acquisition, storage, and access aspects. However, none of them considers the security aspects as an incorporation into all stages of the DW development cycle, nor does any examine the introduction of security in the conceptual MD design.

On the other hand, there are some interesting proposals which define authorization models for data warehouses [28,36,50,52], but they only deal with OLAP operations (e.g., roll-up or drill-down) accomplished with OLAP tools. So these are not conceived for their integration in MD modeling as part of the DW design process, and as a consequence, inconsistent security measures might well be defined. We believe that we should consider basic security measures for business DWs with a conceptual model from the early stages of a DW project. Then more specific security rules can be defined for particular groups of users in terms of data marts, views, and OLAP tools, or any other analysis tools, but which are consistent with the main security rules defined for the DW.

Finally, Priebe and Penul propose the definition of basic security measures for main OLAP operations when querying a DW [36]. To this end, authors extend the ADAPTed UML approach for specifying security measures on a UML class diagram which authors use for the MD modeling of OLAP tools at the conceptual level. This is a very interesting proposal and has some similarities to ours as authors provide a set of authorization rules based on the roles of the different users and they also focus on the read operation. However, we extend their work in some sense and provide other features. Firstly, the conceptual modeling approach we based on allows us to model complex data warehouse scenarios considering non-strict and complete hierarchies, degenerate dimensions, degenerate facts or many-to-many relationships between facts and a particular dimension (with the corresponding attributes). Then, we also try to provide a wider approach for modeling an enterprise data warehouse, rather than an approach for modeling small data marts. Finally, we cover some relevant situations such as providing role hierarchies and compartment groups for users, not only the typical security levels of multilevel security. This allows us to cover more real DWs as in real world projects, since we found that users normally are classified by different criteria.

Finally, all the above-presented proposals only examine access control, but not audit; so the approach proposed in this paper is the first one to consider both

access model and audit in the MD modeling of DWs at the conceptual level.

3. Object-oriented multidimensional modeling

In this section, we outline the UML we use for DW conceptual modeling [31,48]. This approach has been specified by means of a UML profile which contains the necessary stereotypes for carrying out the MD modeling at the conceptual level successfully [17]. The main features of MD modeling considered here are the relationships *many-to-many* between facts and one specific dimension, degenerated dimensions, multiple classification and alternative path hierarchies, and non-strict and complete hierarchies. In this approach, structural properties of MD modeling are represented by means of a UML class diagram in which the information is clearly organized into facts (items of interest for a given business) and dimensions (context in which facts have to be analyzed).

Facts and dimensions are represented by means of fact classes (stereotype Fact) and dimension classes (stereotype Dimension), respectively. Fact classes are defined as composite classes in shared aggregation relationships of n dimension classes. The minimum multiplicity in the role of the dimension classes is 1 (all facts are always related to all dimensions). Relations *many-to-many* between a fact and a specific dimension are specified by the multiplicity 1..* in the role of the corresponding dimension class.

Let us introduce the case study we will use throughout the paper to illustrate our approach. In this case

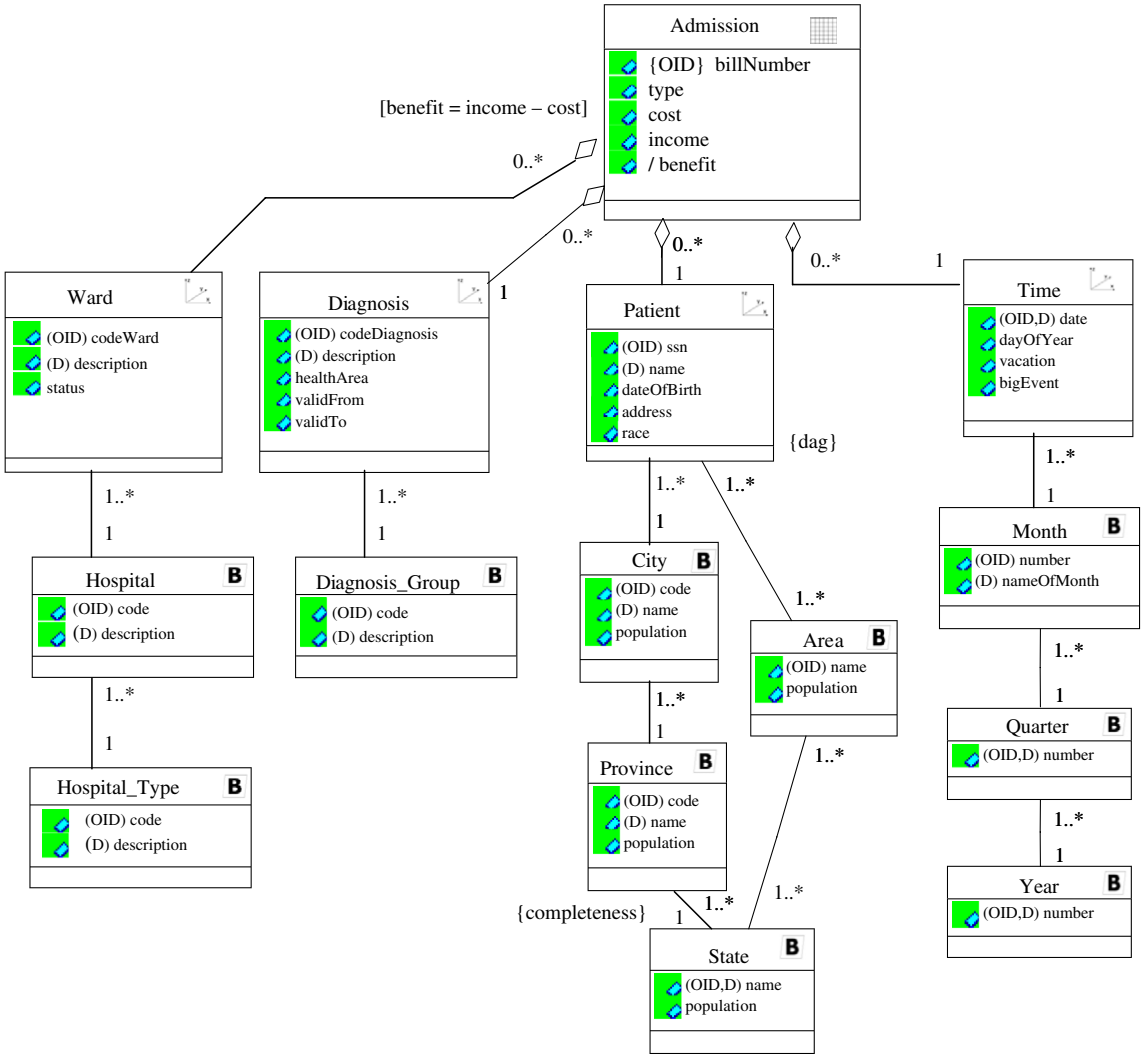

Fig. 1. MD modeling using the UML.

study, we are interested in finding out the profitability of a patient through the different admissions in terms of the diagnosis that was accomplished, the patient to whom it was made, the ward the patient was assigned to, and the date the diagnosis was made. In the MD model (see Fig. 1), the fact is represented by the *Admission* fact class and the dimensions by the *Diagnosis*, *Patient*, *Ward* and *Time* dimension classes. In Fig. 1 we can also see how the *Admission* fact class has a many-to-one relationship with all dimension classes (*diagnosis*, *patient, ward* and *time*).

A fact is composed of measures or fact attributes. By default, all measures in the fact class are considered to be additive. For non-additive measures, additive rules are defined as constraints and are included in the fact class. Furthermore, derived measures can also be explicitly represented (indicated by /) and their derivation rules are placed between square brackets near the fact class. See the *benefit* attribute and its corresponding derived rule in the *Admission* fact class in Fig. 1.

With this approach, we can also define identifying attributes in the fact class (stereotype OID). In this way *degenerated dimensions* can be considered [27], thereby representing other fact features in addition to the measures for analysis. For instance, we could store the bill number (*bill_number*) as a degenerate dimension (see *Admission fact* in Fig. 1); allowing us to consider another interesting fact feature different from the usual measures.

With respect to dimensions, each level of a classification hierarchy is specified by a base class (stereotype Base B). An association of base classes specifies the relationship between two levels of a classification hierarchy. The only prerequisite is that these classes must define a Directed Acyclic Graph (DAG) rooted in the dimension class (DAG constraint is defined in the stereotype Dimension). The DAG structure can represent both multiple and alternative path hierarchies. Every base class must also contain an identifying attribute (OID) and a descriptor attribute[1] (stereotype D). These attributes are necessary for an automatic generation process into commercial OLAP tools, as these tools store this information on their metadata.

Due to the flexibility of UML, we can also consider non-strict hierarchies (an object at a hierarchy's lower level belongs to more than one higher-level object) and complete hierarchies (all members belong to one higher-class object and that object consists of those members only). These features are specified by means of the multiplicity of the roles of the associations and defining the constraint {completeness} in the target associated class role respectively. See *Patient* dimension in Fig. 1 for an example of all kinds of classification hierarchies. Lastly, the categorization of dimensions is considered by means of the generalization / specialization relationships of UML.

[1] A descriptor attribute will be used as the default label in the data analysis in OLAP tools.

4. Access control and audit (ACA) model

Access control is not a complete solution for securing a system [42] as it must be coupled with auditing. Auditing requires the recording of all user requests and activities for their later analysis. Therefore, in our approach, we take both concepts into consideration for their integration in the conceptual MD modeling design.

Access control models are typically composed of a set of authorization rules that regulate accesses to objects. Each authorization rule usually specifies the *subject* to which the rule applies, the *object* to which the authorization refers, the *action* to which the rule refers, and the *sign* describing whether the rule permits or denies the access.

In order to regulate access to objects in a MD model, we have considered the Mandatory Access Control model (in the form of multilevel security policies), which is based on the classification of subjects and objects in the system. Therefore, our access control and audit model allows us to specify sensitivity information assignment rules for all elements of MD models (facts, dimensions, etc.), which define static and dynamic object classification. Moreover, our model allows us to define authorization rules that represent exceptions to the general multilevel rules, where the designer can specify different situations in which the multilevel rules are not sufficient. Finally, a set of audit rules, which represent the corresponding audit requirements, can be included in the model.

Once the data warehouse designer has developed the conceptual MD model, he or she should discover all the security issues associated to this model, and specify them through our ACA model.

In the following section we introduce the complete set of details of the ACA model: the access control model that has been considered, authorization subjects, authorization objects, actions, sensitivity information assignment rules, authorization rules, audit rules, and conflict resolution.

4.1. Access control model

We have considered the Mandatory Access Control model (in the form of multilevel security policies) as a basis for the MD modeling. A brief summary of this access control policy is described below, but for a more detailed discussion, we refer the reader to [39].

In multilevel policies, an access class is assigned to each object and subject. The most common access class is defined as a security level and a set of categories. The security level is an element of a hierarchically ordered set, such as Top Secret (TS), Secret (S), Confidential (C), and Unclassified (U), where, TS > S > C > U. The set of categories is a subset of an unordered set, whose elements reflect functional, or competence, areas. The access class is one element of a partially ordered set of classes, where an access class c_1 dominates an access class c_2 iff the security level of c_1 is greater than or equal to that of c_2 and the categories of c_1 include those of c_2 [39]. We have considered a secrecy-based mandatory policy, so the two principles that must be satisfied to protect information confidentiality are: i) no-read-up (a subject is allowed a read-access to an object only if the access class of the subject dominates the access class of the object), and ii) no-write-down (a subject is allowed a write-access to an object only if the access class of the subject is dominated by the access class of the object).

This access control model has been widely studied [22,39,41], and many vulnerabilities have been detected, such as its lack of flexibility, the polyinstantiation [23], the existence of covert channels, etc. Nevertheless, most of these troubles come from the necessity of taking into consideration both read and write operations in the system. Fortunately, we consider that the sole operation to be used by the final users in decision-support systems is *read*, so Mandatory Access Control is absolutely suitable. Moreover, some of the most important DBMS, such as Oracle [29] and DB2 Universal Database (UDB) [8] are considering commercial products that incorporate an adapted version of Mandatory Access Control. This is important because MD models can be implemented by some of these DBMSs.

In our model, we define an access class on the basis of three different but compatible ways of classifying users; by their security level, the role they play and by the compartments they belong to:

- *Security user roles*: Used by a company to organize users in a hierarchical role structure, according to the responsibilities of each type of work. Each user can play more than one role.
- *Security levels*: This indicates the clearance level of the user.
- *Security user compartments*: Also used by an organization to classify users into a set of horizontal compartments or groups, such as geographical location, area of work, etc. Each user can belong to one or more compartments.

Therefore, in our model, the access class is one element of a partially ordered set of classes, where an access class c_1 dominates an access class c_2 iff the security level of c_1 is greater than or equal to that of c_2, the compartments of c_1 include those of c_2, and at least one of the user roles of c_1 (or one of its ascendant) is defined for c_2.

Thus, for each object in the model, the user access requirements (security level, user roles, and user compartments) can be defined, thereby specifying with high accuracy which users can access each particular object.

4.2. Authorization subjects

The specification of subjects in access control rules often has two apparently contrasting requirements [39]. On the one hand, a subject reference must be simple, to allow for efficient access control and for making use of possible relationships between subjects in the resolving of conflicts between the authorizations (e.g., most specific relationships between roles and sub-roles). On the other hand, one would like to see more specific expressions rather than the simple reference to user identities or user roles, compartments or security levels, providing support of profile-dependent authorizations whose validity depends on properties associated with users (e.g., age, citizenships, or field-of-specification). Our solution encounters both requirements by supporting both user classification concepts (security levels, roles and compartments) and user profiles. The profile can be modeled as a class (with stereotype *UserProfile*) in a UML class diagram, where all the relevant properties of each user for a system are modeled as its attributes. Fig. 2 illustrates an example of user profile definition, and three instances of that class, defining the profiles for three users. The language OCL [51] can be used as a query language in a class diagram. For instance, OCL expression "UserProfile.type = 'hospitalEmployee' and UserProfile.securityLevel ≥ 'Confidential' and UserProfile.securityRoles->includes('Doctor')" returns in-

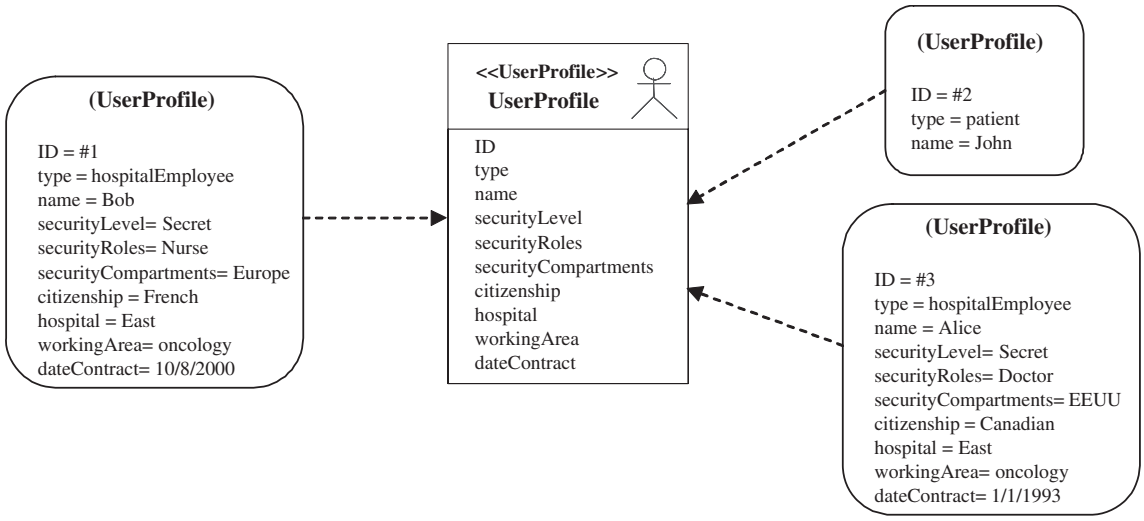

Fig. 2. Example of UML user profiles.

stances of UserProfile who are hospital employees, have a security level equal to or greater than confidential, and who play the role Doctor. In particular, such an expression, evaluated on the profiles of Fig. 2 would return the profile of *Alice*.

Therefore, the *subject* component of our ACA model includes two parts:

- An *identity*, which can be composed of one or more of the following sub-attributes:
  - *UserID*: The user identifier.
  - *RoleID*: The identifier of one user role. It identifies all users who play this role.
  - *CompartmentID*: The identifier of one user compartment. It identifies all users who belong to this compartment.
  - *LevelID*: The name of one security level. It identifies all users with this security level or higher.
- A *subject expression* which is an OCL expression on users' profiles.

Note that the purpose of using OCL expressions in our model is not to retrieve instances of the userProfile class that satisfy certain criteria, but to determine whether a given profile (that of the requestor) satisfies the criteria. Intuitively, security constraints should be applied only if the profile satisfies the constraints specified in OCL.

The grammar we consider for the definition of elements in the ACA model uses Extended Backus Naur Form syntax, in which | means a choice, ? means optionally, * means zero or more times, and+means one or more times. Authorization subjects are then defined as follows:

```
Subjects := subjectIdentification
subjectExpression
subjectIdentification :
= subjectIdentifier (logicalOperator
subjectIdentifier)*
subjectIdentifier :
= "ALLSUBJECTS" | ("ID" userID)
| ("RID" roleID) |
("CID" compartmentID) | ("LID" levelID)
logicalOperator := "AND" | "OR"
subjectExpression :
= "COND" OCLExpression
```

Note that in this grammar, some reserved words have been defined in order to express the concepts clearly. For instance, 'ALLSUBJECTS' is a reserved word that will be used to refer to all users of the system, 'ID' will be used to identify a particular user, 'RID' will be used to refer to all users who play a particular role, and so on.

For instance, *subject* element "RID nurse AND CID Europe AND LID Secret COND profile.workingArea=oncology" denotes all subjects who play the role *nurse*, who belong to the compartment *Europe* and who have the security level *Secret or Top Secret*, and who also satisfy the condition. In this example, this subject expression will be evaluated to true for *Bob*, and therefore an authorization rule using this subject

will be applicable to him. By contrast, the authorization rule will not be applicable to *Alice* (who does not play the role *nurse*) or to John (who fulfills none of those conditions).

4.3. Authorization objects

In this sub-section, we need to identify and specify the main objects we deal with for the conceptual modeling of DWs such as facts, dimension, classification hierarchy levels, measures, dimension attributes and instances. From now on, all the required rules for specifying security measures will be defined based on the UML stereotypes of our approach for the conceptual MD model at the conceptual level [31,32]. Thus, the object component of our ACA model includes two parts:

- An *identity*, that can be one of the following sub-attributes:
- *MDClassName*: The MD class identifier. It can be related to a fact class or a dimension class, referring to a fact or dimension; or base class referring to a classification hierarchy level.
- *MDAttributeName*: The MD attribute identifier. In object-oriented notation we have to indicate the *class name*, and then the *attribute name*. Thus, we treat both measures and dimension attributes as just attributes; since the preceding class name will tell us if we refer to a measure or a dimension attribute. Moreover, treating both measures and dimension attributes in the same way will allow us to execute security measures on attributes correctly, if in a further analysis carried out by OLAP tools, users interchange measures and dimension attributes.
- An *object expression* which is an OCL expression on the class model that represents the MD model. This OCL expression can be necessary when the identity is a MD class, and when it selects some instances of that class depending on a condition based on the value of the attributes of its instances (or instances of other classes that are reachable by navigating from the former class) or perhaps, on any property of the contextual user profile. The contextual user profile is the profile of the user who is accessing the data, and can be referred to in an OCL expression by using the stereotyped class *UserProfile*. Notice that this object expression represents a *DICE* action in OLAP terminology, since it selects some elements from the data cube of the data warehouse, according the evaluation of a condition.

Authorization objects are defined with this grammar:

```
Objects := objectIdentifier
objectExpression
objectIdentifier :
= (“MDCL” MDclassName+) | (“MDATT”
(MDclassName”.”MDattributeName))+
MDclassName := factClass
| dimensionClass | baseClass
MDattributeName := factAttribute
| dimensionAttribute
objectExpression :
= “COND” OCLExpression
```

Considering the conceptual diagram of Fig. 1, the object expression “MDCL diagnosis COND diagnosis. type=AIDS” denotes all instances of the dimension class *Diagnosis* whose attribute *type* has the value “*AIDS*”. Suppose now Bob (see Fig. 2) executes a query, and that the following object expression “MDCL diagnosis COND diagnosis.healthArea=UserProfile.workingArea” has been defined. The objects that are affected by this object expression are all instances of the dimension class *Diagnosis* whose attribute *healthArea* has the value “oncology”, which is the working area of Bob.

4.4. Authorization actions

For the sake of simplicity, in this ACA model we have only considered the *read* action for accessing the DW repository structures. Other database-oriented actions relevant for ETL (Extraction-Transformation-Loading) processes such as *insert*, *delete*, *update*, and other possible OLAP-oriented actions such as *drilling-through*, *drilling-down*, *rolling-up, slice*, *dice*, and so on, are based on the *read* action. In addition, these operations finally execute the *read* operation on the database structure that hosts the data, and therefore, from the conceptual point of view, the most relevant operation at this point is the *read* one. However, specific navigating and OLAP operations are beyond the scope of this work, as in this paper we present the basis for the conceptual modeling of secure business data warehouses, and then more specific security measures for final users or specific OLAP tools may be defined from our secure enterprise data warehouse. Therefore, the action component of our model has only one element that identifies the type of action, and is defined with this part of the model grammar:

```
Actions := action (logicalOperator action)*
action := ”READ”
```

4.5. Sensitivity information assignment rules

Our access control and audit model uses Sensitive Information Assignment Rules (SIARs) in order to specify multilevel security policies. These rules allow us to define the corresponding sensitivity information (access class in multilevel security terminology) for each element of the MD model. The information we need to specify for each of these rules is as follows:

- *Objects* to which the rule is applicable.
- *Sensitivity information* that is assigned. It represents the access class for the object, which, as previously mentioned, can be composed of security levels, user roles and user compartments.
- *Involved multidimensional classes* in the query. In MD models, the sensitivity of the information may depend on the classes that are involved in the query. Some information may not be particularly confidential if it is consulted in isolation, but this same information might be highly confidential if it is associated with other data. For instance, a list of illnesses is not very confidential, but if this information is associated with patients, it is transformed into confidential information. By default, there is always a MD involved class to which the rule refers, and which it is not necessary to specify. This element is also necessary as not all dimensions are *sliced* or *diced*: instead we may need to specify a security rule on a dimension that is not explicitly a constraint.
- *A condition* specified with OCL. Different instances of a class in our MD model can have different access classes depending on the value of some attributes. If a condition is defined in one of these rules, the specific access class of each instance of this object will depend on the evaluation of this condition. If we do not define a condition in a rule, all specified objects will have the same access class.

We can consider a generic MD model (see Fig. 3), composed by a fact class (F) and four dimension classes (D_i), each dimension with a classification hierarchy with three base classes (B_{ij}). We are interested in controlling access to individual class instances, which is fulfilled by defining SIARs associated with the fact class, or with the dimension or base classes (e.g., F, D_2 or B_{32} in Fig. 3).

Moreover, as previously commented in Section 2.4., we are also interested in controlling access to combination of data (e.g., fact with dimensions or base classes, or even dimensions with base classes, etc.), which is one of the most dangerous situations from the security point of view, when a DW is queried. This is why we specify the list of involved MD classes in this type of rules. A SIAR can be defined, for instance, for the fact class *F*, considering some of its dimensions (at a particular level in the classification hierarchy) as involved MD classes (e.g., B_{21} and B_{43}), and as defining an access class (e.g., security level *SECRET*). This rule means that users who try to combine F with B_{21} and B_{43} should have a security level equal to or greater than Secret. A particularity of the classification hierarchies of dimensions is that the closer to the fact class the base class is, the more specific is the information the base class contains, and therefore the greater the depth to which confidentiality measures should be defined. This fact makes it necessary to propagate up the SIARs. That is to say, if a user has no access to a query combining F with B_{21} and B_{43}, she does not have access to a query combining F with ascendants of B_{21} or B_{43}, since the information that can be obtained is more specific and therefore more confidential. On the other side, the user will have access to a query combining F with descendants of these base classes. Note that with the use of involved MD classes in SIARs, multilevel security is being implicitly defined for typical OLAP operations (e.g., drill-down, drill-across, slice, etc.), since all of them are based on queries reading a combination of facts with one or more dimensions.

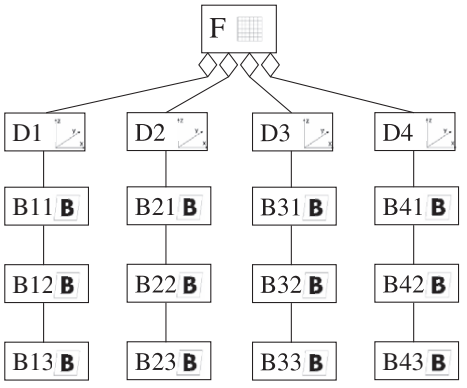

Fig. 3. Generic MD model.

The grammar we have defined for representing sensitivity information assignment rules is defined below:

```
SIAR :="OBJECTS" Objects
("INVMDCLASSES" MDClassName+)?
("SECINF" securityInformation) |
("COND" conditionAssignment)
securityInformation :=
("SL" securityLevel)?
```

```
("SR" userRole+)?
("SC" userCompartment+)?
conditionAssignment :=
"IF" booleanExpression
"THEN"(securityInformation |
conditionAssignment)
("ELSE"(securityInformation |
conditionAssignment) )?
"ENDIF"
```

Elements we define in this grammar (except the objects) are expressed as tagged values associated with classes or attributes (the rule objects) in the MD class diagram. For the sake of the diagram readability, we have considered two general situations with two different graphical representations: i) if the rule just defines the access class of a class or attribute (without specifying neither involved classes nor condition), we directly insert the tagged values (SL for security levels, SC for security compartments and SR for security roles) into the class square (see for example how SIAR 1 in Table 6 is modeled in Fig. 4), and ii) if the rule also specifies complex objects, involved MD classes and/or conditions, we insert the corresponding tagged values (*Objects*, *InvolvedMDClasses*, and *Condition*, respectively) into a note that is associated with the corresponding class (see for example how SIAR 6 in Table 6 is modeled in Fig. 4). The values of those tagged values are simply literals, or expressions specified by using OCL, thereby allowing us an easy understanding of UML users and tools.

4.6. Authorization rules

Sensitivity Information Assignment Rules allow us to specify multilevel security policies, but unfortunately, these rules are not enough to specify other situations in which the access class of objects or subjects is not relevant. As we aim to define a general and expressive enough access control and audit model, we also define Authorization Rules (AURs), which coexist with SIARs, thus permitting us to specify security models which are much more elaborate.

In the literature, we can find different authorization variants, some of the most important being implicit, explicit, positive, negative, weak and strong [16]. Moreover, access control models can be open (allowing access by default, unless a negative authorization exists) or closed (denying access by default, unless a positive authorization exists) [39]. In a multilevel system, a user needs to fulfill the multilevel security imposed by the object she or he wants access to, so we can consider that our system is closed. In our system both positive and negative AURs can coexist (by defining the sign + or − in the rule). Some types of implicit and explicit AURs can also be defined, but we will not consider weak and strong authorizations explicitly (we provide rules for resolving conflicts between AURs and SIARs). For each AUR, we will specify the following concepts:

- *Subjects* to which the rule is applicable.
- *Objects* to which the rule is applicable.
- *Actions* considered. As we have hitherto remarked, in terms of basic database operations, all OLAP operations execute the *read* action. Thus, we only consider this operation.
- *Sign*, that defines if the authorization is positive (allowing access) or negative (denying access).
- *Involved multidimensional classes in* the query. It specifies the fact, dimension or base classes that have to be involved in the query in order for this AUR to be applicable. By default, there is always a MD involved class which is that to which the rule refers, and which it is not necessary to specify.

An AUR, therefore, specifies who can (or cannot) access whichever element, performing whichever action. Once again, we have to interpret the rule depending on what the MD classes are that are involved in a query. To illustrate the semantics of these rules, we can consider again the general MD model on Fig. 3. The definition of an AUR for the dimension D_i without specifying involved MD classes protects the instances of D_i against direct queries. Nevertheless, the definition of an AUR for the fact class F, specifying the involved MD classes B_{ij} and B_{kl}, protects the instances of F but only in the case of a user trying to access a combination of F, B_{ij} and B_{kl}. The consequences of specifying an AUR are greater than we can see intuitively. To analyze these consequences, we deal with positive authorizations first, and then with negative authorizations:

- Let us suppose we define a positive AUR, where the considered Subject is Bob, the Object is the fact class F, and the involved classes are the base classes B_{21} and B_{42}. This authorization allows Bob to access instances of any possible combination of these MD classes (in this case, they can be F, B_{21}, B_{42}, F with B_{21}, F with B_{42}, and F with B_{21} and with B_{42}). Moreover, the AUR is propagated down. For instance a combination of F with B_{23} and B_{43} would be accessible to Bob, since the information that can be obtained in this case is more general (and as a

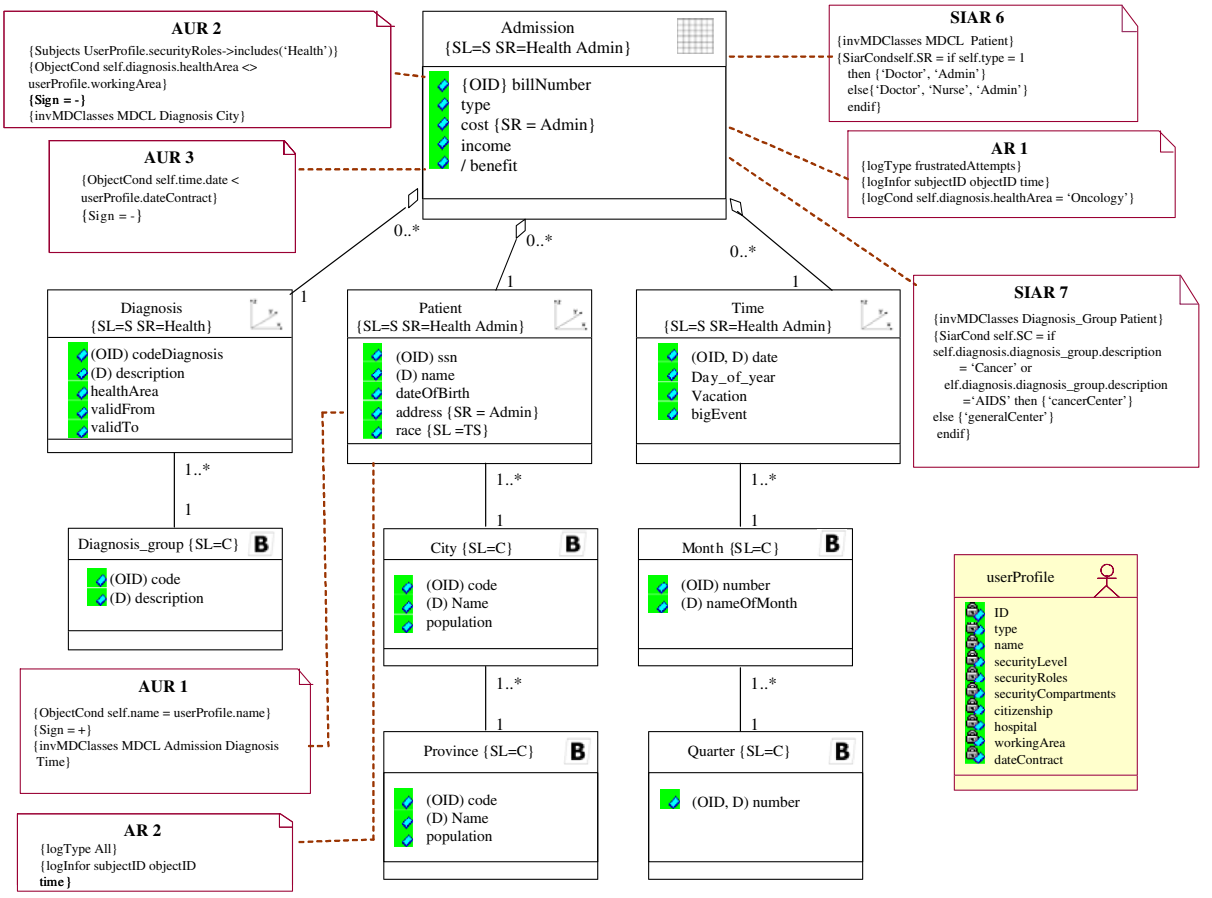

Fig. 4. Example of MD model with security information and constraints.

result less sensitive) than considering the former base classes (see Fig. 5 left).

- Suppose now we define a negative AUR with the same elements as in the previous case. This authorization denies Bob the access to instances of the combination of F with B_{21} and B_{42}. It also denies access in the case of accessing F with B_{21} and also in the case of F with B_{42}. Moreover, the AUR is propagated up (see Fig. 5 right). The case of a combination of F with D_2 and B_{41} is also denied to Bob, therefore.

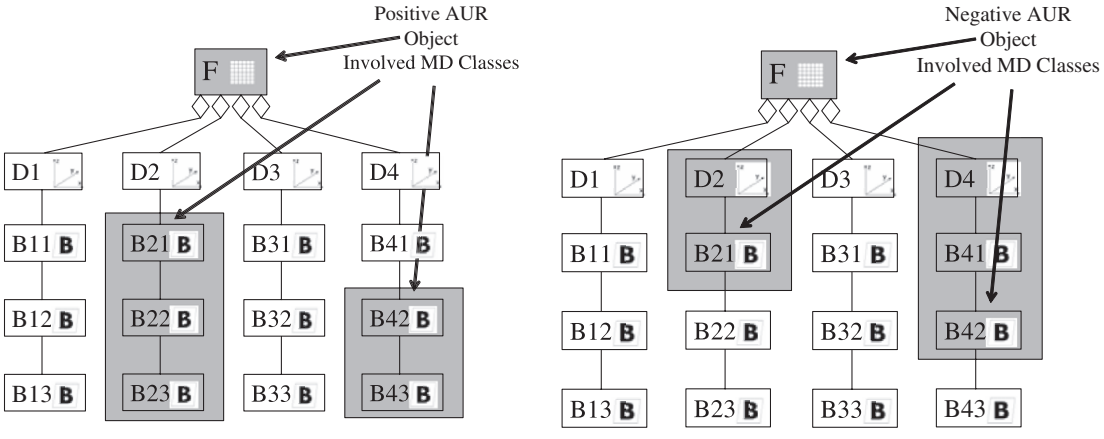

Fig. 5. Positive and negative AUR propagation.

Of course, the combination of F with B_{21} and B_{42} (or any of its ascendants) with any other dimension or base class is also denied.

Once again, the use of involved MD classes in AURs, can be interpreted and applied in OLAP environments, since all its typical operations are based on queries combining the facts with one or more dimensions.

The definition of authorization rules in our model must be according to this grammar:

```
AUR := "SUBJECTS" Subjects "OBJECTS"
Objects "ACTIONS" Actions "SIGN" Sign
("INVMDCLASSES" MDclassName+)?
Sign := "+" | "−"
```

Elements of authorization rules (except the objects) are represented in the form of tagged values associated with classes (the rule objects) in the class diagram. For the sake of the diagram readability, all tagged values are grouped in a note that is associated to the corresponding class in the diagram (see for example how AUR 1 in Table 6 is modeled in Fig. 4).

4.7. Audit rules

Audit controls are useful both as a deterrent against misbehavior as well as a means for analyzing the user behavior by employing the system to find out possible attempted or actual violations. Additionally, auditing is essential to ensure that authorized

Table 1
Stereotypes of the new data types defined for classes

Tagged values of the class				
Name	Type	M	Description	Default value
SecurityLevels	Levels	1	It specifies the interval of possible security level values. If the upper and lower security levels are different, the specific level will be defined with a security constraint.	The lower level should be 'Unclassified'
SecurityRoles	Set (Role)	1	It specifies a set of user roles. Each role is the root of a sub-tree of the general user role hierarchy defined for the organization. A security constraint can decide the user roles according to the value of some attributes.	The set composed by the root role defined for the model
Security-compartments	Set(Compartment)	1	It specifies a set of compartments. All instances of this class can have the same user compartments, or a subset of them. A security constraint can decide the user compartments according to the value of some attributes.	Empty set of compartments
LogType	Attempt	0..*	It specifies whether the access has to be recorded: none, all accesses, only frustrated accesses, or successful accesses.	None
LogInfo	Set(LOGInfo)	0..*	It specifies the elements we want to record.	Empty
LogCond	OCLExpression	0..*	It specifies whether the access has to be recorded.	Empty
InvMDClasses	Set (OclType)	0..*	It specifies the classes involved in an exception.	Empty
SIAR cond	OCL expression	0..*	It specifies the condition that defines the sensitivity information.	Empty
Sign	{+,−}	0..*	An exception permits (+) or denies (−) access to users.	+
Privileges	Set(Privilege)	0..*	It specifies the privileges the user can receive or remove.	Read
ObjectCond	OCLExpression	0..*	It selects the objects that are affected by the rule.	Empty
Subjects	OCLExpression	0..*	It specifies the subjects that are affected by this rule.	All subjects

Table 2
Stereotype *UserProfile* of our extension

Name	UserProfile
Base class	Class
Description	Classes of this stereotype contain all the properties that the systems manage from users
Constraints	- This class has no associations to other classes. Self.AssociationsEnd.size()=0 - There is no more than one class of this type Context Model Inv self.classes->forAll(oclisTypeOf(UserProfile))->size()<=1 - The name of a class of this stereotype will be *UserProfile.* Self.className=*UserProfile*
Tagged values	None
Icon	

users do not misuse their privileges [42]. Audit Rules (ARs) can be specified, in line with the following concepts:

- *Objects* to which the rule is applicable.
- *Log type*. It specifies whether the access has to be recorded. Values can be none, all access, only frustrated accesses, or only successful accesses.
- *Log condition*. It defines a condition based on the object model.
- *Information* to be logged. Depending on the situation, we can record information such as the subject requesting the access, the object to be accessed, the operation requested, the time of the request, and the response of the access control model.

Audit rules are defined as illustrated below:

```
AR := "OBJECTS" Objects "LOGTYPE"
logType "LOGCOND" OCLExpression "
LOGINFO" logInformation
logType := "none" | "all"
| "frustratedAttempts" | "
successfulAttempts"
logInformation := subjectID?
objectID? action? Time? response?
```

Once again, elements of audit rules (except the objects) are represented in the form of tagged values of classes (the rule objects) in the class diagram, which are grouped in a note, associated to the corresponding class in the diagram (see for example how AR 1 in Table 6 is modeled in Fig. 4).

4.8. Conflict resolution

Some conflicts can appear between several AURs, several SIARs, and even between AURs and SIARs. We solve these conflicts according to the following rules: i) If there is a conflict between a positive AUR and a SIAR, the set of users that will be able to access the information will be composed of users who fulfill the SIAR and users who do not fulfill the SIAR but who fulfill the subject condition of the AUR, ii) If there is a conflict between a negative AUR and a SIAR, the set of users that will not be able to access the information will be composed of users who do not fulfill the SIAR and the users who fulfill the SIAR but who fulfill the subject condition of the AUR, iii) If there is a conflict between two SIARs, the security information for each instance will be the result of applying the most restrictive SIAR, iv) An AUR that refers to an individual user has a higher level of preference than any other AUR that refers to a set of users, v) An AUR that refers to a particular user role *r* has greater preference than any other AUR that refers to an ascendant of r in the user-roles tree, vi) Two AURs that refer to different sets of users (for instance to a compartment and a role) have the same preference, and vii) If two AURs have the same preference, we will select the negative one. If they have the same sign there is no conflict.

5. UML extension for secure multidimensional modeling

In this section, we sketch our UML extension for applying our ACA model to the conceptual MD modeling of secure DWs. We have re-used the previous profile defined in [31], which allows us to design DWs from a conceptual perspective as described in Section 3, and we have added the required elements whose security aspects we need to specify (Subjects, Objects, Actions, Sensitive Information Assignment Rules, Authorization Rules, and Audit Rules) defined in our ACA model in this paper.

Table 3
An example of well-formedness rules of our UML profile

The set of user compartments of each class and attribute has to be a subset of the compartments defined for the model
Context Model inv self.classes->forAll(c | c.Compartments->forAll(comp | self.Compartments->includes(comp))) inv self.classes->forAll(c | c.attributes->forAll(a | a.Compartments->forAll (comp |self.Compartments->includes(comp))))

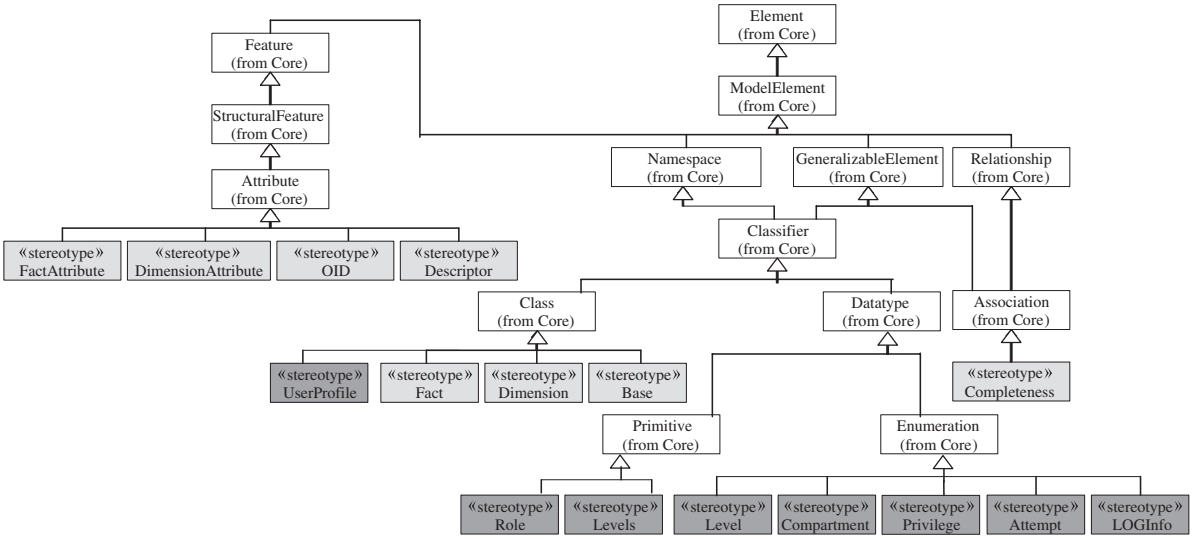

Fig. 6. Extension of the UML with stereotypes.

According to the UML specification, our UML extension is defined as follows: *description* (a short description of the extension in natural language), *prerequisite extensions* (indicates whether the current extension needs the existence of previous extensions), the definition of *stereotypes* and / or *tagged values*, *well-formedness rules* (the static semantics of the metaclasses are defined both in natural language and in OCL expressions), and *comments* needed for the correct understanding of the extension.

Next, we will present only a summary of all these elements. We refer the reader to [15] for a complete description of the UML profile, as this paper focuses on the ACA model.

5.1. Description

This UML extension re-uses a set of stereotypes, defined previously in [31], and establishes a set of tagged values, stereotypes, and constraints, which enables us to create secure MD models. The 25 new tagged values are applied to certain objects that are especially peculiar to MD modeling, allowing us to represent these in the same model and on the same schemas that describe the rest of the system. These tagged values will represent the sensitivity information of the different objects of the MD modeling (fact classes, dimension classes, attributes, etc.) and they will allow us to specify security constraints depending on this security information and on the value of attributes of the model. A set of inherent constraints is specified, in order to define rules of well-formedness. These rules are defined in both natural language and OSCL [14], assuring a correct use of them.

Thus, we have defined 8 new stereotypes: one specializes in the Class model element, two specialize in the Primitive model element and five specialize in the Enumeration model element. In Fig. 6, we have represented portions of the UML metamodel[2] to show where our stereotypes fit. We have only represented the specialization hierarchies, as the most important fact about a stereotype is the base class a new stereotype specializes. In Fig. 6, new stereotypes are colored in dark grey, the stereotypes we re-use from the previous profile [31] in light grey and classes from the UML metamodel remain white.

5.2. Prerequisite extension

This UML profile re-uses stereotypes previously defined in another UML profile [31]. The main elements of this profile are *fact classes, dimension classes, base classes*; *OID, descriptor, dimension* and *fact attributes*; and, *completeness* for complete hierarchies. See Section 3 for further detail on how to use these elements.

5.3. Data types

We need the definition of some new data types to be used in our tagged values definitions. These types are as

[2] All the metaclasses come from the Core Package, a subpackage of the Foundation Package. We based our extension on the UML 1.5 as this is the current accepted standard. To the best of our knowledge, the current UML 2.0 is not the final accepted standard yet.

Table 4
Example of SIAR rules with our ACA model

Rule 1	For each instance of the fact class *Admission* and the dimension classes *Patient* and *Time*, the security level will be at least *Secret*, and the security roles will be *Health* and *Admin*.
SIAR 1:	OBJECTS MDCL Admission Patient Time SECINF SL Secret SR Health Admin
	Tagged values SL and SR of the fact class *Admission* and the dimension classes *Patient* and *Time* in Fig. 4.
Rule 2	For each instance of the dimension class *Diagnosis*, the security level will be at least *Secret*, and the security role will be *Health*.
SIAR 2:	OBJECTS MDCL Diagnosis SECINF SL Secret SR Health
	Tagged values SL and SR of the dimension class *Diagnosis* in Fig. 4.
Rule 3	For each instance of the base class *Diagnosis_group*, *City*, *Province*, *Month* and *Quarter*, the security level will be *Confidential*.
SIAR 3:	OBJECTS MDCL Diagnosis_Group City Province Month Quarter SECINF SL Confidential
	Tagged values SL of the base class *Diagnosis_Group, City, Province, Month* and *Quarter* in Fig. 4.
Rule 4	The security roles for the attribute *cost* of the fact class *Admission* and for the attribute *address* of the dimension class *Patient* will be only *Admin.*
SIAR 4:	OBJECTS MDATT Admission.Cost Patient.Address SECINF SR Admin
	Tagged value SR of the attribute *cost* in the fact class *Admission* and the attribute *address* in the dimension class *Patient* in Fig. 4.
Rule 5	The security level for the attribute *race* of the dimension class *Patient* will be *TopSecret.*
SIAR 5:	OBJECTS MDATT Patient.Race SECINF SL TopSecret
	Tagged values SL of the attribute *race* in the dimension class *Patient* in Fig. 4.
Rule 6	For each instance of the fact class *Admission*, if it is queried together with the dimension class *Patient*, if the admission type is *primary diagnosis*, then the security role will be Doctor and Admin, otherwise it will be Doctor, Nurse, and Admin. A primary diagnosis is considered to the most important reason for a treatment, while secondary diagnosis completes the view of the patient's condition. In the *Admission* fact class, the 'type' attribute is '1' for primary diagnosis and '2' for secondary diagnosis.
SIAR 6:	OBJECTS MDCL Admission invMDClasses Patient Cond if self.type=1 then SR Doctor Admin else SR Doctor Nurse Admin endif
	Note titled 'SIAR6', associated with the fact class *Admission* in Fig. 4.
Rule 7	For each instance of the fact class *Admission*, if the description of the *diagnosis_group* of its *diagnosis* is especially sensitive (cancer or AIDS) when it is related with the dimension class *Patient*, then it only will be accessible to a member of the compartment *cancerCenter.* Note that the rest of security information (security levels and user roles) remain and must be fulfilled in order to access this information. Note also that this rule is propagated up. Therefore, if a user has no access to query combining the fact class *Admission* with classes *Diagnosis_group* and *Patient*, she will not have access to a query combining *Admission* with *Patient* and an ascendant of *Diagnosis_group* (*Diagnosis* in this example). Furthermore, note that the model in Fig. 6 does not contain security compartments, because by default, all users are not restricted by compartments, except in this very special case because this information must well-known and be used only by personnel of health related to very serious diseases.
SIAR 7:	OBJECTS MDCL Admission INVMDCLASSES Diagnosis_Group Patient COND IF self.Diagnosis. Diagnosis_Group.description='Cancer' of self.Diagnosis. Diagnosis_Group.description='AIDS' then SC cancerCenter ENDIF
	Note titled "SIAR7", associated with the fact class *Admission* in Fig. 4.

follows. *Level* (typical values: *unclassified, confidential, secret* and *top secret*, but they could be different). *Levels* as an interval of levels composed by a lower level and an upper level. *Role* and *Compartment* will represent the hierarchy of user roles and the defined

Table 5
Example of authorization rules with our ACA model

Rule 8	Patients will be special users of the system as we would like them to have access to their own data. Each patient will therefore have permission to access his/her own data (not the data of the other patients) when querying classes *Patient, Admission, Diagnosis* and *Time*. As this authorization rule is positive, it is propagated down.
AUR 1:	SUBJECTS AllSubjects OBJECTS MDCL Patient COND self.Name=userProfile.name ACTION Read SIGN+INVMD CLASSES Admission Diagnosis Time
	Note titled "AUR1", associated with the dimension class *Patient* in Fig. 4.
Rule 9	Queries involving *Admission, Diagnosis* and *City* corresponding to a particular health area will be accessible to members of the user role *Health* only if their working area is the same as that health area. Note that this authorization rule is propagated up.
AUR 2:	SUBJECTS RID Health OBJECTS MDCL Admission COND self.Diagnosis.healthArea<>userProfile.workingArea ACTION Read SIGN-INVMDCLASSES Diagnosis City
	Note titled "AUR2", associated with the fact class *Admission* in Fig. 6.
Rule 10	All members of the user role *HospitalEmployee* should not have access to admissions that are prior to the date of the start of their contract.
AUR 3:	SUBJECTS RID HospitalEmployee OBJECTS MDCL Admission COND self.Time.date<userProfile.dateContract ACTION Read SIGN-
	Note titled "AUR3", associated with the fact class *Admission* in Fig. 4.

user compartments, respectively. *Compartments* is a set of user compartments. *Privilege* as all different privileges that have been considered (values: *read, inserte, delete, update*, and *all*). *Attempt* as all different considered access attempts (normal values: *none, all, frustratedAttempt, sucessfulAccess*). Finally, *LOGInfo* as all elements that we could record (values: *subjectID, objectID, Action, time, response*).

5.4. Tagged values

Here we define several tagged values for the *model, classes, attributes, instances* and *constraints*. For reasons of space, in Table 1 we only show the tagged values defined for *classes*, as these are the most significant ones and are widely used in our case study. See [15] for further details on the rest of the defined tagged values. By using these tagged values on a class, we mean that the instances of the corresponding class may have any of the defined values according to its data type.

5.5. Stereotypes

We need one stereotype to specify other types of security constraints (Table 2). The stereotype *UserProfile* can be necessary in order to specify constraints depending on a particular piece of information about a user or a group of users, e.g., depending on citizenship, age, etc. Thus the previously defined data types and tagged values will then be used on the *fact*, *dimension* and *base* stereotypes, with the objective of examining other security aspects.

5.6. Well-formedness rules

We can identify and specify in both natural language and OCL constraints some well-formedness rules for the proper use of the newly defined UML elements. An example of these well-formedness rules is shown in Table 3 (see [15] for further detail).

6. A case study applying our extension for secure MD modeling

In this section, we apply our ACA model and UML extension for the conceptual design of a secure MD model in the context of a typical healthcare system. Regarding the example of Fig. 1 (in Section 3), we have only considered an example in shortened form, so as to focus our attention on the main security specifications. We have looked at a simplified hierarchy of user roles that is typical for a hospital (the most general is *hospitalEmployee*, which is then specialized in the roles *health* and *nonHealth*, and which are in turn specialized in the roles *doctor* and *nurse* in the former case, and in *maintenance* and *administrative*) in the second. As security levels, we have considered in this case those typical in military environments, and as compartments,[3] we consider *cancerCenter*, which groups users who research in the cancer center, and *generalCenter*, which groups the rest of the users.

Table 6
Example of audit rules with our ACA model

Rule 11	We wish to record the subject, object and time for every frustrated access attempt to admissions of the oncology health area.
AR 1: OBJECTS MDCL Admission LOGTYPE frustratedAttempts LOGINFO SubjectID ObjectID time LOGCOND self.diagnosis.healthArea=oncology	
Note titled "AR1", associated with the fact class *Admission* in Fig. 4.	
Rule 12	We wish to record the subject, object and time for all accesses to a patient's data.
AR 2: OBJECTS MDCL Patient LOGTYPE All LOGINFO SubjectID ObjectID time	
Note titled "AR2", associated with the dimension class Patient in Fig. 4.	

Studying a submodel of the MD model shown in Fig. 1, which is composed of the fact class *Admission*, the dimension classes *Diagnosis, Patient* and *Time*, and the base classes *Diagnosis_group*, *City, Province, Month* and *Quarter*, we could define our ACA model by defining the SIARs, AURs and ARs. Once our security rules have been defined, we can enrich our MD conceptual model with the corresponding tagged values and stereotypes that contain the security specifications.

Fig. 4 shows an MD model which includes all classes described above, the security specifications (which we explain below), and an additional class (*UserProfile*). The class *UserProfile* (stereotype *UserProfile*) is a special class which models all properties of users that the system stores in order to manage the authorizations. We can observe in Fig. 4 that we use several tagged values to allow us to model all our rules of the ACA model. In Tables 4–6 we specify the security specifications. Each security rule is first defined in natural language, then we specify the corresponding ACA rule, and finally we indicate the place where this rule is specified in the secure MD conceptual model.

[3] As can be seen in Fig. 6, Level and Compartment are new data types inherited from the *UML enumeration data* type.

7. Conclusions and future work

In this paper, we have presented an Access Control and Audit (ACA) model designed to represent major confidentiality and audit aspects in the conceptual modeling of DWs. Our ACA model allows us to define rules in order to specify the security information of each element in the MD (MD) model (SIARs), rules for representing authorization rules (AURs), which work together with SIARs, and rules which allow us to specify audit requirements (ARs). In addition, we have outlined the UML extension specified by us, and which allows us to integrate our ACA model with our previous UML extension for the conceptual modeling of DWs, thereby permitting us to obtain secure MD models at the conceptual level. Thus, we are able to specify security aspects such as security levels on data, compartments and user roles on the main elements of a MD model such as facts, dimensions, classification hierarchies and so on. We have used the OSCL to establish the ACA rules and the constraints attached to these newly defined elements, thereby avoiding an arbitrary use of these rules and constraints. The main relevant advantage of this approach is that it gives us the possibility of taking security aspects into account from the early stages of a DW project, i.e., at the conceptual level. Furthermore, the UML extension we used for the conceptual modeling of DWs and the ACA model in conjunction allow us to represent all main relevant MD properties at the conceptual level.

Our work for the immediate future is to improve the ACA model, extending the set of privileges considered in this paper (i.e., read), so as to permit us to specify security aspects in the crucial ETL processes for DWs, thereby considering other operations such as delete, insert and update. As well as this, we plan to consider implementation issues for the use of the security aspects considered when querying a MD model from OLAP tools. This includes an in-depth study on the different combinations of dimensions, measures, hierarchies and so on, as involved in queries.

Acknowledgements

This research is part of the RETISTIC (TIC2002-12487-E) and the METASIGN (TIN2004-00779) projects, supported by the Dirección General de Investigación of the Ministerio de Ciencia y Tecnología, the MESSENGER project, supported by the Consejería de Ciencia y Tecnlogía of the Junta de Comunidades de Castilla-La Mancha (PCC-03-003-1), and the DADAMESCA project (GV 05/220) supported by the Consellería de Empresa, Universidad y Ciencia de la Generalitat Valenciana. We would also like to thank the reviewers for their valuable comments, which have helped us improve this paper.

References

[1] A. Abelló, J. Samos, F. Saltor, A framework for the classification and description of multidimensional data models, 12th International Conference on Database and Expert Systems Applications (DEXA'01), Springer-Verlag, Munich, Germany, 2001, LNCS 2113.
[2] A. Abelló, J. Samos, F. Saltor, YAM2 (Yet another multidimensional model): an extension of UML, International Database Engineering and Applications Symposium (IDEAS 2002), IEEE Computer Society, Edmonton, Canada, 2002.
[3] V. Atluri, W.-K. Huang, An authorization model for workflows, 5th European Symposium on Research in Computer Security, Springer Verlag Rome, Italy, 1996.
[4] E. Bertino, S. Jajodia, P. Samarati, A flexible authorization mechanism for relational data management systems, ACM Transactions on Information Systems 17 (April 1999).
[5] M. Blaschka, C. Sapia, G. Höfling, B. Dinter, Finding your way through multidimensional data models, 9th Intl. Conference on Database and Expert Systems Applications (DEXA'98), Springer-Verlag, Vienna, Austria, 1998.
[6] N. Bolloju, M. Khalifa, E. Turban, Integrating knowledge management into enterprise environments for the next generation decision support, Decision Support Systems 33 (2) (June 2002).
[7] P. Bonatti, E. Damiani, S. De Capitani di Vimercati, P. Samarati, An access control model for data archives, IFIP-TC11 International Conference on Information Security, (Paris, France), 2001.
[8] S. Cota, For Certain Eyes Only, DB2 Magazine 9 (1) (2004).
[9] E. Damiani, S. De Capitani di Vimercati, E. Fernández-Medina, P. Samarati, An access control system for SVG documents, in: E. Gudes, S. Shenoi (Eds.), Research Directions in Data and Applications Security, Kluwer Academic Publisher, Boston, 2002.
[10] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, P. Samarati, A fined-grained access control system for XML documents, ACM Transactions on Information and Systems Security 5 (May 2002).
[11] P. Devanbu, S. Stubblebine, Software engineering for security: a roadmap, in: A. Finkelstein (Ed.), The Future of Software Engineering, ACM Press, 2000.
[12] G. Dhillon, J. Backhouse, Information system security management in the new millennium, Communications of the ACM 43 (7) (2000).
[13] E. Fernández-Medina, M. Piattini, Designing secure database for OLS, Database and Expert Systems Applications: 14th International Conference (DEXA 2003), Springer, Prague, Czech Republic, 2003, LNCS 2736.
[14] E. Fernández-Medina, M. Piattini, Extending OCl for secure database design, International Conference on the Unified Modeling Language (UML 2004), Springer-Verlag, Lisbon, Portugal, 2004, LNCS 3273.
[15] E. Fernández-Medina, J. Trujillo, R. Villarroel, M. Piattini, Extending the uML for designing secure data warehouses, In-

ternational Conference on Conceptual Modeling (ER 2004), Springer-Verlag, Shangai, China, 2004, LNCS 3288.
[16] E. Ferrari, B. Thuraisingham, Secure database systems, in: M. Piattini, O. Díaz (Eds.), Advanced Databases: Technology Design, Artech House, London, 2000.
[17] M. Gogolla, B. Henderson-Sellers, Analysis of uML stereotypes within the UML metamodel, 5th International Conference on the Unified Modeling Language—The Language and its Applications, Springer, Dresden, Germany, 2002, LNCS 2460.
[18] M. Golfarelli, D. Maio, S. Rizzi, The dimensional fact model: a conceptual model for data warehouses, International Journal of Cooperative Information Systems 7 (2–3) (1998).
[19] A. Hall, R. Chapman, Correctness by construction: developing a commercial secure system, IEEE Software 19 (1) (January/February 2002).
[20] B. Husemann, J. Lechtenborger, G. Vossen, Conceptual data warehouse design, Proceedings of the 2nd. International Workshop on Design and Management of Data Warehouses, Technical University of Aachen (RWTH), (Stockholm, Sweden), 2000.
[21] H. Inmon, Building the Data Warehouse, 3rd ed., John Wiley & Sons, USA, 2002.
[22] S. Jajodia, R. Sandhu, Toward a multilevel secure relational data model, ACM SIGMOD International Conference on Management Data, (Denver, Colorado), 1991.
[23] S. Jajodia, R. Sandhu, Polyinstantiation for cover stories, Second European Symposium on Research in Computer Security, (Toulouse, France), 1992.
[24] S. Jajodia, P. Samarati, M.L. Sapino, V.S. Subrahmanian, Flexible support for multiple access control policies, ACM Transactions on Database Systems 26 (June 2001).
[25] J. Jürjens, UMLsec: extending uML for secure systems development, in: J. Jézéquel, H. Hussmann, S. Cook (Eds.), UML 2002—The Unified Modeling Language, Model Engineering, Concepts and Tools, Springer, Dresden, Germany, 2002.
[26] N. Katic, G. Quirchmayr, J. Schiefer, M. Stolba, A. Min Tjoa, A Prototype Model for Data Warehouse Security Based on Metadata, 9th International Workshop on Database and Expert Systems Applications (DEXA'98), IEEE Computer Society, Vienna, Austria, 1998.
[27] R. Kimball, M. Ross, The Data Warehousing Toolkit, 2nd ed., John Wiley, 2002.
[28] R. Kirkgöze, N. Katic, M. Stolda, A. Min Tjoa, A security concept for OLAP, 8th International Workshop on Database and Expert System Applications (DEXA'97), IEEE Computer Society, Toulouse, France, 1997.
[29] J. Levinger, Oracle label security. Administrator's guide, Release 2 (9.2) (2002) (http://www.csis.gvsu.edu/GeneralInfo/Oracle/network.920/a96578.pdf).
[30] T. Lodderstedt, D. Basin, J. Doser, SecureUML: a UML-based modeling language for model-driven security, The Unified Modeling Language Conference, Springer, Dresden, Germany, 2002, LNCS 2460.
[31] S. Luján-Mora, J. Trujillo, I.Y. Song, Extending the UML for multidimensional modeling, 5th International Conference on the Unified Modeling Language (UML 2002), Springer-Verlag, Dresden, Germany, 2002, LNCS 2460.
[32] S. Luján-Mora, J. Trujillo, I.Y. Song, Multidimensional modeling with UML package diagrams, International Conference on Conceptual Modeling-ER 2002, Springer, Tampere, Finland, 2002, LNCS 2503.
[33] D. Marks, P. Sell, B. Thuraisingham, MOMT: a multi-level object modeling technique for designing secure database applications, Journal of Object-Oriented Programming 9 (4) (1996).
[34] A. Nanda, Keeping Information Private with VPD, Oracle, XVIII(2) (March/April, 2004).
[35] H. Nemati, D. Steiger, L. Iyer, R. Herschel, Knowledge warehouse: an architectural integration of knowledge management, decision support, artificial intelligence and data warehousing, Decision Support Systems 33 (2) (June 2002).
[36] T. Priebe, G. Pernul, A pragmatic approach to conceptual modeling of OLAP security, 20th Int. Conference on Conceptual Modeling, Springer-Verlag, Yokohama, Japan, 2001.
[37] F. Rabitti, E. Bertino, W. Kim, D. Woelk, A model of authorization for next-generation database systems, ACM Transactions on Database Systems 16 (1) (1991).
[38] A. Rosenthal, E. Sciore, View security as the basic for data warehouse security, 2nd International Workshop on Design and Management of Data Warehouse, (Sweden), 2000.
[39] P. Samarati, S. De Capitani di Vimercati, Access control: policies, models, and mechanisms, in: R. Focardi, R. Gorrieri (Eds.), Foundations of Security Analysis and Design, Springer, Bertinoro, Italy, 2000.
[40] P. Samarati, E. Bertino, S. Jajodia, An authorization model for a distributed hypertext system, IEEE Transactions on Knowledge and Data Engineering 8 (4) (August 1996).
[41] R. Sandhu, F. Chen, The multilevel relational data model, ACM Transactions on Information and Systems Security (TISSEC) 1 (1) (June 1998).
[42] R. Sandhu, P. Samarati, Authentication, access control, and intrusion detection, in: A. Tucker (Ed.), CRC Handbook of Computer Science and Engineering, CRC Press Inc, 1997.
[43] C. Sapia, M. Blaschka, G. Höfling, B. Dinter, Extending the E/R model for the multidimensional paradigm, 1st International Workshop on Data Warehouse and Data Mining (DWDM'98), Springer-Verlag, Singapore, 1998.
[44] P. Shim, M. Warkentin, J.F. Courtney, D.J. Power, R. Sharda, C. Carlsson, Past, present, and future of decision support technology, Decision Support Systems 33 (2) (June 2002).
[45] E.G. Sirer, K. Wang, An access control language for web services, 7th ACM Symposium on Access Control Models and Technologies, ACM Press, 2002.
[46] G.W. Smith, Modeling security-relevant data semantics, IEEE Transactions on Software Engineering 17 (11) (1991).
[47] A. Toval, J. Nicolás, B. Moros, F. García, Requirement reuse for improving information systems security: a practitioner's approach, Requirement Engineering Journal 6 (4) (2002).
[48] J. Trujillo, M. Palomar, J. Gómez, I.Y. Song, Designing Data Warehouses with OO Conceptual Models, IEEE Computer, special issue on Data Warehouses, 34 (2001).
[49] N. Tryfona, F. Busborg, J. Christiansen, starER: a conceptual model for data warehouse design, ACM 2nd International Workshop on Data Warehousing and OLAP (DOLAP'99), ACM, Missouri, USA, 1999.
[50] L. Wang, S. Jajodia, D. Wijesekera, Securing OLAP data cubes against privacy breaches, IEEE Symposium on Security and Privacy, (Berkeley, California), 2004.
[51] J. Warmer, A. Kleppe, The object constraint language second edition, Getting Your Models Ready for MDA, Addison Wesley, 2003.
[52] E. Weippl, O. Mangisengi, W. Essmayr, F. Lichtenberger, W. Winiwarter, An authorization model for data warehouses and OLAP, Workshop on Security in Distributed Data Warehousing, (New Orleans, Louisiana, USA), 2001.

Eduardo Fernández-Medina holds a PhD. and an MSc. in Computer Science from the University of Sevilla. He is Assistant Professor at the Escuela Superior de Informática of the University of Castilla-La Mancha at Ciudad Real (Spain), his research activity being in the field of security in databases, data-warehouses, web services and information systems, and also in security metrics. Fernández-Medina is co-editor of several books and chapter books on these subjects, and has several dozens of papers in national and international conferences (DEXA, CAISE, UML, ER, etc.). Author of several manuscripts in national and international journals (Information Software Technology, Computers And Security, Information Systems Security, etc.), he is a member of the ALARCOS research group of the Department of Computer Science at the University of Castilla-La Mancha, in Ciudad Real, Spain. He belongs to various professional and research associations (ATI, AEC, ISO, IFIP WG11.3 etc.). Eduardo's e-mail is eduardo.fdezmedina@uclm.es.

Juan Trujillo is an associated professor at the Computer Science School at the University of Alicante, Spain. Trujillo received a PhD in Computer Science from the University of Alicante (Spain) in 2001. His research interests include database modeling, conceptual design of data warehouses, multidimensional databases, OLAP, as well as object-oriented analysis and design with UML. With papers published in international conferences and journals such as ER, UML, ADBIS, CAiSE, WAIM, Journal of Database Management (JDM) and IEEE Computer, Trujillo has served as Program Committee member of several workshops and conferences such as ER, DOLAP, DSS, and SCI and has also spent some time as a reviewer of several journals such as JDM, KAIS, ISOFT and JODS. His e-mail is jtrujillo@dlsi.ua.es.

Rodolfo Villarroel has an MSc in Computer Science from the Universidad Técnica Federico Santa María (Chile), and is currently a PhD student at the Escuela Superior de Informática of the University of Castilla-La Mancha in Ciudad Real (Spain). Assistant Professor at the Computer Science Department of the Universidad Católica del Maule (Chile), his research activity is in the field of security in data warehouses and information systems, and of software process improvement. Author of several papers on data warehouse security and improvement of software configuration management process, Villarroel belongs to the Chilean Computer Science Society (SCCC) and the Software Process Improvement Network (SPIN-Chile). His e-mail is rvillarr@spock.ucm.cl.

Mario Piattini has an MSc and a PhD in Computer Science from the Politechnical University of Madrid. He is a Certified Information System Auditor from the ISACA (Information System Audit and Control Association). Full Professor at the Escuela Superior de Informática of the Castilla-La Mancha University (Spain) and author of several books and papers on databases, software engineering and information systems, Piattini leads the ALARCOS research group of the Department of Computer Science at the University of Castilla-La Mancha, in Ciudad Real, Spain. His research interests are: advanced database design, database quality, software metrics, object-oriented metrics and software maintenance. His e-mail address is Mario.Piattini@uclm.es.