Defining and transforming security rules in an MDA approach for DWs

Carlos Blanco*,
Ignacio García-Rodríguez de Guzmán
and Eduardo Fernández-Medina

Department of Information Technologies and Systems,
Escuela Superior de Informática,
Alarcos Research Group – Institute of
Information Technologies and Systems,
University of Castilla-La Mancha,
Paseo de la Universidad, 4, 13071, Ciudad Real, Spain
E-mail: Carlos.Blanco@uclm.es
E-mail: Ignacio.Grodriguez@uclm.es
E-mail: Eduardo.Fdezmedina@uclm.es
*Corresponding author

Juan Trujillo

Department of Software and Computing Systems,
LUCENTIA Research Group,
University of Alicante,
San Vicente s/n, 03690, Alicante, Spain
E-mail: jtrujillo@dlsi.ua.es

Mario Piattini

Department of Information Technologies and Systems,
Escuela Superior de Informática,
Alarcos Research Group – Institute of
Information Technologies and Systems,
University of Castilla-La Mancha,
Paseo de la Universidad, 4, 13071, Ciudad Real, Spain
E-mail: Mario.Piattini@uclm.es

Abstract: Data Warehouses (DWs) store historical information which support the decision-making process. Since this information is crucial, it has to be protected from unauthorised accesses by defining security constraints in all stages of the DW development process. In previous works, we applied the Model-Driven (MDA) philosophy to define secure DWs using several security models and transformations. Nevertheless, our conceptual model makes it possible to define complex security rules with OCL expressions which are not transformed automatically. This paper deals with this problem and completes our approach improving our conceptual metamodel and defining new transformation rules. Finally, an application example is shown.

Keywords: DWs; data warehouses; OLAP; on-line analytical processing; multidimensional modelling; security; security rules; MDA; model-driven architecture; QVT; query/view/transformation.

Reference to this paper should be made as follows: Blanco, C., García-Rodríguez de Guzmán, I., Fernández-Medina, E., Trujillo, J. and Piattini, M. (2010) 'Defining and transforming security rules in an MDA approach for DWs', *Int. J. Business Intelligence and Data Mining*, Vol. 5, No. 2, pp.116–133.

Biographical notes: Carlos Blanco has an MSc in Computer Science from the University of Castilla-La Mancha. He is currently a PhD student at the School of Computer Science at the University of Castilla-La Mancha (Spain), and his research activity is in the field of security in Data Warehouses, MDA, information systems and ontologies.

Ignacio García Rodríguez de Guzmán has an MSc in Computer Science and a PhD from the University of Castilla-La Mancha (UCLM). He is currently a member of the Alarcos Research Group in the School of Computer Science of UCLM and his research activity is in the field of MDA, Architecture-Driven Modernisation, Reverse Engineering and Reengineering, SOA and Web Services.

Eduardo Fernández-Medina holds a PhD in Computer Science from the University of Castilla-La Mancha. His research activity is in the field of security in databases, data warehouses, web services and information systems, and also in security metrics. He is co-editor of several books and book chapters on these subjects and has presented several dozens of papers at national and international conferences (DEXA, CAISE, UML, ER, etc.). He is the author of several manuscripts in national and international journals (DSS, ACM Sigmod Record, IS, IST, C&S, ISS, etc.) and belongs to various professional and research associations (AEC, ISO, IFIP WG11.3, etc.).

Juan Trujillo received a PhD in Computer Science from the University of Alicante in 2001. His research interests include database modelling, the conceptual design of data warehouses, multidimensional databases, OLAP, OO analysis and design with UML. He has had papers published in international conferences and journals (ER, UML, ADBIS, CAiSE, WAIM, JDM, IEEE Computer, DSS, IS, etc.). He has served as a Program Committee member of several workshops and conferences (ER, DOLAP, DSS, SCI, etc.) and as Program Chair (DOLAP, BP-UML, etc.). He has also spent some time as a Reviewer for several journals (JDM, DKE, KAIS, ISOFT, JODS, etc.).

Mario Piattini has an MSc and a PhD in Computer Science from the Polytechnic University of Madrid. He is a Certified Information System Auditor from the ISACA (Information System Audit and Control Association). As the 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. His research interests include advanced database design, database quality, software metrics, object-oriented metrics and software maintenance.

1 Introduction

A Data Warehouse (DW) is a repository that manages a large amount of enterprise historical information integrated from different Data Sources (DSs) (Inmon, 2002). This information is usually organised following a multidimensional approach by using facts (for instance, a product sale) and related dimensions with classifications by subject (for instance, departments, cities or product categories). A typical DW architecture is composed of several layers: heterogeneous DS, ETL (Extraction/Transformation/Load) processes, which extract and transform data from these DSs and load the information into the DW, the main part of the architecture, the DW repository where data are stored, and DataBase Management Systems (DBMS) and On-Line Analytical Processing (OLAP) tools that analyse data.

Information security is a critical issue for information systems that must be considered from the early stages of development as a strong requirement (Mouratidis and Giorgini, 2006) and must be integrated into the whole development process (Fink et al., 2006) to take into account these security constraints for design decisions. Data in DW are very sensitive because they involve vital business information, which are used to support the strategic decision-making process and can also manage personal information protected under the law. Thus, security involves all the layers and operations of the warehouse (Thuraisingham et al., 2007) and security constraints have to be defined to prevent unauthorised access.

Moreover, MDA (MDA, 2003) provides model-driven software development based on the separation of the specification of the system functionality and its implementation by defining models at different abstraction levels and the transformations between them, i.e., business models (CIM) with system requirements, conceptual models (PIM), which do not include information about specific platforms and technologies, and logical models (PSM) with information about the specific technology used. Developing information systems applying the MDA approach improves productivity, saving time and effort, and provides support for system evolution, integration, interoperability, portability, adaptability and reusability.

Our proposal (Fernández-Medina et al., 2007a), then, develops secure DWs, including security issues, in all the stages of the development process. It considers the special characteristics of DWs and security aspects on the basis of an Access Control and Audit (ACA) model (Fernández-Medina et al., 2006) specifically developed for DWs in which the security classification of subjects and objects and the definition of several kinds of security rules are supported. This proposal has also been aligned with an MDA (MDA, 2003) and defines extensions of models at different abstraction levels. MDA provides model-driven software development based on the separation of the specification of the system functionality and its implementation. It allows the model to be defined at different abstraction levels (business, conceptual and logical levels) and the automatic transformation between models through the definition of transformations rules.

Our architecture was originally focused on a relational approach towards DBMS, providing a logical relational model. However, since most DWs are managed by OLAP tools over a multidimensional approach, our most recent research efforts have been focused on a logical multidimensional model, which eventually leads to secure implementation in OLAP tools. We have defined a multidimensional logical metamodel and transformation rules from conceptual models, considering some security issues,

but our proposal did not, until now, completely support the transformation of all the types of security rules that can be defined with our ACA model by using OCL notes, but their automatic transformation has not been dealt with so far.

This paper analyses this problem and provides a solution composed of:

1 an improvement of our conceptual metamodel with new features that support these advanced security rules

2 a set of transformations, which translate these rules from the conceptual level into a secure multidimensional model at a logical level.

The remainder of this paper is organised as follows: Section 2 will present the research background on secure DW development; Section 3 will briefly introduce our complete MDA approach through which we develop secure DWs and on which this work is focused; Section 4 will present the improvements made to support security rules in conceptual metamodels and the sets of transformation rules defined to automatically obtain secure logical models; Section 5 will show, using an example, how security rules defined in conceptual models are transformed into a multidimensional logical model; finally, Section 6 will present our conclusions and future work.

2 Related work

There are interesting contributions in the field of information systems but they do not deal with DWs in the context of their specific security issues. One of the first and most relevant proposals that integrates security through the use of UML is UMLsec (Jürjens, 2004), which can be used to specify and evaluate UML security specifications using formal semantics. Furthermore, Model-Driven Security (MDS) (Basin et al., 2006) extends MDA to build secure information systems. Its designers specify the inclusion of security properties in high-level system models and use tools to automatically generate secure system architectures. Within the context of MDS, the same authors propose an extension of UML for modelling a generalised Role-Based Access Control (RBAC) called SecureUML (Lodderstedt et al., 2002).

Data Warehouses (DWs) present specific characteristics and security challenges related to all their layers and operations (Thuraisingham et al., 2007). Since DWs mainly deal with read operations, the confidentiality problem is the most important to consider with secure development.

There are some interesting works dealing with DW security. Saltor et al. (2002) use the existing parallelism between Federated Information Systems (FISs) and DWs to adopt a design architecture for FIS and to improve it with security capabilities supporting the integration of Mandatory Access Control (MAC) policies. Furthermore, there are access control models for DWs and OLAP focused on expressiveness and usability (Essmayr and Weippl, 2000).

Traditionally, security in DWs was taken into account in the final stage of development. Several works (Katic et al., 1998; Kirkgöze et al., 1997; Priebe and Pernul, 2001) exist, which deal with a secure implementation in OLAP tools using a Discretional Access Control (DAC) security policy and a simplified concept of user role defined as subject. These works do not include security in the entire development process but focus on the last stages. Although other interesting proposals for modelling

DWs at conceptual and logical levels, which consider special characteristics of DWs, also exist, they do not support security issues.

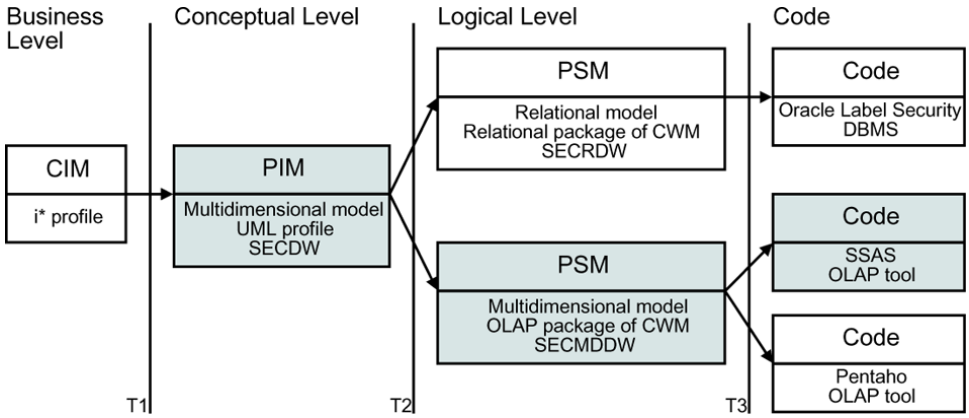
The most interesting proposal is Priebe and Pernul (2001) in which the authors define a methodology to analyse security requirements, to represent them at the conceptual level with ADAPTed UML and finally, to implement a solution in SQL Server Analysis Services (SSAS). They extend multidimensional expressions (MDX) with hide statements (for cubes, dimensions, etc.), to create a Multidimensional Security Constraint Language (MDSCL). However, they do not define the transformation between levels and focus on a DAC security policy.

Our research efforts are thus applied to the development of secure DWs considering confidentiality issues during the whole development process, from an early development stage to the final implementation. To achieve this goal, our proposal (Fernández-Medina et al., 2007a) provides security models at different abstraction levels and has been aligned with an MDA architecture in which security models are embedded and scattered throughout the high-level system models, which are transformed towards the final implementation according to the MDA strategy.

3 An MDA approach for Secure DWs

This section briefly presents our MDA approach for developing DWs taking into account security issues in all the stages of the development process (Fernández-Medina et al., 2007a). It is composed of several models at different abstraction levels (business, conceptual and logical levels) and rules to define the automatic transformation between them (see Figure 1).

Figure 1 Model-driven architecture for secure DWs (see online version for colours)



At the business level, a UML profile (Trujillo et al., 2009) was defined to include security requirements in a Computational Independent Metamodel (CIM). This profile extends *i**, which is a requirement engineering framework focused on agents and their intentional characteristics.

At the conceptual level, a Platform Independent Metamodel (PIM) defined by a UML profile, called SECDW (Fernández-Medina et al., 2007b), extends an existing proposal for conceptual modelling of DWs with security capabilities and considers fact, dimension and base classes and specific aspects of DWs such as many-to-many relations, degenerated dimensions, multiple classifications and alternative paths of hierarchies.

These security capabilities are provided by an ACA model (Fernández-Medina et al., 2006) specifically designed for DWs. This classifies subjects and objects in three ways (clearance levels 'Security Levels', hierarchical role structures 'Security Roles' and horizontal compartment or groups 'Security Compartments'), defines secure classes and properties, and allows for several kinds of security rules related to multidimensional elements, which will be analysed in the next section.

Two Platform-Specific Metamodels (PSMs) have been proposed at the logical level as extensions of Common Warehouse Metamodel (CWM) packages, and the corresponding transformations from the conceptual level have also been defined by using Query/View/Transformation (QVT) rules. On the one hand, we have defined a relational path (Soler et al., 2008) composed of:

1 A logical relational metamodel called SECRDW

2 QVT rules from PIM models

3 The final implementation into DBMS using Oracle Label Security.

This architecture has been specialised with a multidimensional path because most DWs are managed by OLAP tools within a multidimensional approach. This specialisation (Blanco et al., 2009) is composed of:

1 a multidimensional secure metamodel at the logical level, called SECMDDW

2 a set of QVT rules from PIM models

3 the final implementation in a specific OLAP tool, SSAS.

However, although our proposal considers structural aspects and security constraints, the support of more complex security rules, which can be established in our conceptual models using OCL expressions, has not, until now, been completely supported. This paper completes our proposal, improving the metamodels to include information on security rules and defining new sets of QVT rules for their automatic transformation into logical models and the final implementation.

4 Including security rules support in our MDA approach

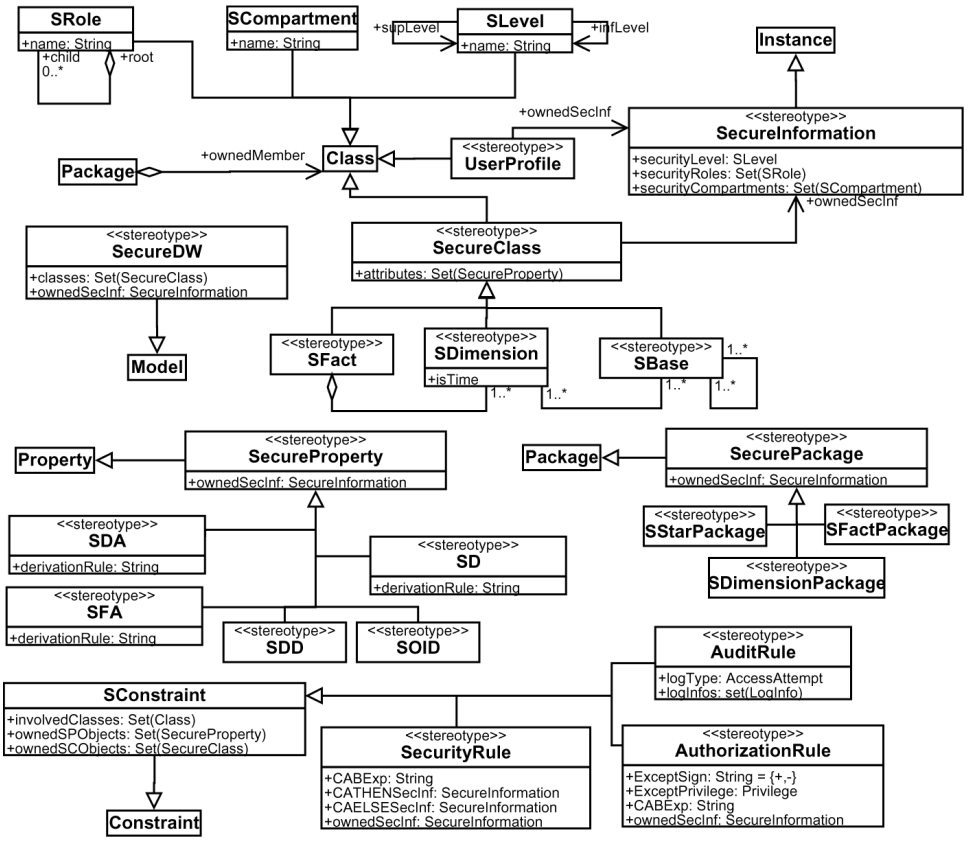
This section presents the improvements carried out to support security rules defined at the conceptual level using our ACA model (Fernández-Medina et al., 2006).

4.1 Conceptual metamodel (SECDW)

Our conceptual metamodel (SECDW) has been improved to include security rules support (see Figure 2). The security classification of objects and subjects into security roles, compartments and levels is defined by using 'Srole', 'Scompartment' and 'Slevel' metaclasses, which allow us to specify a role hierarchy, a set of compartments

and a list of levels. The 'SecureInformation' metaclass has also been added to the metamodel to clearly define the security information (security roles, compartments and level) associated with a specific multidimensional element.

Figure 2 Conceptual metamodel (SECDW)



Our ACA model allows us to specify three kinds of security rules in conceptual models using OCL expressions in notes associated with the corresponding classes. Sensitive Information Assignment Rules (SIARs) let us define sensitivity information for each element in the multidimensional model over a multilevel security policy. Authorisation Rules (AURs) permit or deny access to certain objects by defining the subject that the rule applies to, the object that the authorisation refers to, the action that the rule refers to and the sign describing whether the rule permits or denies access. Finally, Audit Rules (ARs) ensure that authorised users do not misuse their privileges.

Security rules usually include information about subjects, objects, conditions, security information, privileges, log types, etc., which is very difficult to directly analyse over the OCL expression. Therefore, our conceptual metamodel has been improved to manage this information. The three kinds of security rules manage a list of multidimensional objects (secure properties and classes) where, with the purpose of applying the rule and defining this information, we have defined sets of secure classes

and properties in a metaclass called 'Sconstraint', which is specialised into one specific metaclass for each kind of security rule. SIARs are specified by the 'SecurityRule' metaclass, which considers conditions with a Boolean expression and the secure information that will be assigned whether the condition is satisfied or not. AURs use the 'AuthorisationRule' metaclass that includes information about the security, information associated with the sign of the authorisation (positive or negative), the privilege (read, insert, update, delete, all) and a Boolean expression condition to establish a condition. Finally, ARs are specified by the 'AuditRule' metaclass, which defines the access attempt and the information (subject, object, action, time and response) that will be logged.

4.2 Logical multidimensional metamodel (SECMDDW)

The SECMDDW metamodel (Blanco et al., 2009) uses a multidimensional approach to define secure models at the logical level. This metamodel is composed of three metamodels: a security configuration metamodel that represents roles of an RBAC policy; a cube metamodel that defines structural aspects of cubes, measures and hierarchies and security constraints with cube and cell permissions; finally, a dimension metamodel with structural information on dimensions, attributes and base classes and security issues using permissions over dimensions and attributes.

4.3 Transformation SECDW to SECMDDW

Up to now, the following issues have been addressed:

1 existing Security Rules have been presented

2 involved metamodels have been improved for a suitable representation of the mentioned rules (see Figure 2).

Two new rule sets have been defined to automatically transform security rules from conceptual models into multidimensional logical models. In brief, these transformations extract the SIAR and AUR from the conceptual model and introduce their semantics into the Cubes, Dimensions and Attributes of the logic models of DW.

SECDWSecurityRules2CubePermissions processes all the SIAR and AUR included in the PIM, which are related to Fact classes. These security rules (see Table 1) are expressed as a specialisation of the SConstraint metaclass, which in turn is a specialisation of the UML 2.0 Constraint metaclass.

To reflect the SIARs in the PSM, the transformation executes its processCubeSIAR relation (see Table 2). The processCubeSIAR relation modifies the existing PSM and establishes certain values in properties of the PSM classes to implement the semantics of the SIARs included in the PIM. In the same way, the processCubeAUR (see Table 3) deals with the representation of the AuthorisationRules in the PIM and processes it to produce the required security aspects depicted by these rules in the PSM.

Table 1 SECDWSecurityRules2CubePermission

Transformation **SECDWSecurityRules2CubePermissions**
top relation processSecurityRules {…}
relation processCubeSIAR {…}
relation processCubeAUR {…}
relation denySLevelAtt2CellPermissionForSIAR {…}
relation denySRoleClass2CubePermission {…}
relation SCompartmentClass2CubePermission {…}
relation SRoleClass2CubePermission {…}
relation SLevelClass2CubePermission {…}
relation denySRoleAtt2CellPermissionForSIAR {…}
relation denySCompartmentAtt2CellPermissionForSIAR {…}
relation denySCompartmentClass2CubePermission {…}
relation denySLevelClass2CubePermission {…}
relation SCompartmentClass2CubePermissionForAUR {…}
relation SRoleClass2CubePermissionForAUR {…}
relation SLevelClass2CubePermissionForAUR {…}
relation SLevelAtt2CellPermissionForAUR {…}
relation SRoleAtt2CellPermissionForAUR {…}
relation SCompartmentAtt2CellPermissionForAUR {…}

Table 2 processCubeSIAR relation

```
relation processCubeSIAR{
…CLASSES:Sequence(Class);
SLEVS:Set(SLevel);
SROLES:Set(SRole);
SCOMPS:Set(SCompartment);
  checkonly domain pim
sr:SecurityRule{
    involvedClasses =
INVCLASS:Set(Class),
    ownedSPObjects =
OWNSPO:Set(SecureProperty),
    ownedSCObjects =
OWNSCO:Set(SecureClass),
    ownedSecInf = secinf,      CABExp =
cabexp,
    CATHENSecInf = catSecInf,
CAELSESecInf = caeSecInf }
  where{
    INVCLASS->forAll(cTmp:Class
|CLASSES.append(cTmp));
    OWNSCO->forAll(cTmp:Class |
CLASSES.append(cTmp));
    Let FACTS:Sequence(SFact) =
        SRoleClass2CubePermission(sr,sf));
    SCOMPS->forAll(sc:SCompartment |
        SCompartment2CubePermission(sc,sf)));
    PROPERTIES->forAll(sp:SecureProperty |
 SLEVS->forAll(sl:SLevel |
SLevelAtt2CellPermission (sl, sp));
    SROLES->forAll(sr:SRole |
SRoleAtt2CellPermission(sr,sp));
    SCOMPS->forAll(sc:SCompartment|
        SCompartmentAtt2CellPermission(sc,sp)));
    SLEVS =
getLowerSecurityLevels(caeSecInf.securityLevel);
     caeSecInf.securityRoles->forAll(sr:SRole |
     getNotLeafSRoles(st)->forAll(srTmp:SRole|
       SROLES.append(srTmp)))
     SCOMPS = getNotIncludedCompartments
        (caeSecInf.securityCompartments);
     FACTS->forAll( SLEVS->forAll(sl:SLevel|
denySLevelClass2CubePermission(sl, sf, cabexp));
        SROLES->forAll(sr:SRole|
denySRoleClass2CubePermission(sr,sf, cabexp));
        SCOMPS->forAll(sc:SCompartment|
```

Table 2 processCubeSIAR relation (continued)

CLASSES->select(cFact:Class |cFact.oclIsKindOf(SFact)) in Let PROPERTIES:Sequence(SecureProperty) = CLASSES->select(cProp:Class| sProp.oclIsKindOf(SecureProperty)) in SLEVS = getUpperSecurityLevels(catSecInf.securityLevel); catSecInf.securityRoles->forAll(sr:SRole | getLeafSRoles(st)->forAll(srTmp:SRole| SROLES.append(srTmp))) SCOMPS = catSecInf.securityCompartments; FACTS->forAll(sf:SFact | SLEVS->forAll(sl:SLevel| SLevelClass2CubePermission(sl, sf)); SROLES->forAll(sr:SRole |	denySCompartment2CubePermission(sc, sf, cabexp))); //**Denying SecureProperties** PROPERTIES->forAll(sp:SecureProperty | SLEVS->forAll(sl:SLevel| denySLevelAtt2CellPermissionForS (sl, sp, cabexp)); SROLES->forAll(sr:SRole| denySLevelAtt2CellPermission (sr,sp, cabexp)); SCOMPS->forAll(sc:SCompartment| denySLevelAtt2CellPermission (sc,sp, cabexp)));}

Table 3 processCubeAUR relation

relation **processCubeAUR**{
…CLASSES:Sequence(Class); SLEVS:Set(SLevel);
SROLES:Set(SRole); SCOMPS:Set(SCompartment);
checkonly domain pim ar:AuthorisationRule{
involvedClasses = INVCLASS:Set(Class),
ownedSPObjects = OWNSPO:Set(SecureProperty),
ownedSCObjects = OWNSCO:Set(SecureClass),
ownedSecInf = secinf, ExceptionSign =es,
ExceptPrivilege = ep,
involvedClasses = INVCLASS:Set(Class),
CABExp = exp }
where{
//**Firstly, prepare a set of classes**
INVCLASS->forAll(cTmp:Class| CLASSES.append(cTmp));
OWNSCO->forAll(cTmp:Class| CLASSES.append(cTmp));
//**Prepare the set of SFact Classes**
	getLeafSRoles(st)
	->forAll(srTmp:SRole | SROLES.append(srTmp)))
	SCOMPS = secInf.securityCompartments;
	if (es = "+") then
	SLEVS = getUpperSecurityLevels(catSecInf.securityLevel);
	else
	SLEVS = getLowerSecurityLevels(catSecInf.securityLevel);
	//**Process AUR for SFACTS according to the sign**
	FACTS->forAll(sf:SFact |
	SLEVS->forAll(sl:SLevel| SLevelClass2CubePermissionForAUR(sl, sf, exp,es));
	SROLES->forAll(sr:SRole| SRoleClass2CubePermissionForAUR(sr, sf, exp, ex));
	SCOMPS->forAll(sc:SCompartment| SCompartment2CubePermissionForAUR(sc,sf, exp, es)));
	PROPERTIES->forAll(sp:SecureProperty |
	SLEVS->forAll(sl:SLevel| SLevelClass2CubePermissionForAUR(sl, sf, exp,es));

Table 3 processCubeAUR relation (continued)

Let FACTS:Sequence(SFact) = CLASSES->select(cFact:Class | cFact.oclIsKindOf(SFact)) in Let PROPERTIES:Sequence(SecureProperty) = CLASSES->select(cProp:Class | sProp.oclIsKindOf(SecureProperty)) in //**Process ownedSecInf** secInf.securityRoles->forAll(sr:SRole |	SROLES->forAll(sr:SRole| SRoleClass2CubePermissionForAUR(sr, sf, exp, ex)); SCOMPS->forAll(sc:SCompartment}

The QVT relations to process the AURs produce positive and negative authorisations for the information on the levels, roles and compartments specified by the SecureInformation classes referenced by the AURs. Since both the SIAR and AUR accept Boolean expressions to be evaluated and applied over the content of the aforementioned SecureInformation, the transformations also propagate these Boolean expressions to the PSM. Thus, the positive or negative authorisation (when AUR) depends on the Boolean expressions.

When the AUR is positive, the QVT rules authorise the given level (in the SecureInformation) and the upper levels the given role and its descendants, and the specified compartments. On the other hand, when the authorisation is negative, the QVT rules deny the given level and the lower levels, the given role and its descendants, and the specified compartments.

SECDWSecurityRules2DimensionPermissions (see Table 4) includes relations such as processDimensionSIAR and processDimensionAUR, which work in a similar manner to the previous transformations for cubes, performing the same functionalities over the Dimension classes in the PSM. Thus, no further explanation will be provided. As noted, this section includes the transformations, which drive the required actions to include the semantics of the Security Rules in the logic models (processCubeSIAR, processDimensionSIAR, processCubeAUR and processDimensionAUR).

Table 4 SECDWSecurityRules2DimensionPermission

Transformation **SECDWSecurityRules2DimensionPermissions**
top relation processDimensionSecurityRules {…}
relation processDimensionSIAR {…}
relation processDimensionAUR {…}
relation authoriseSCompartment {…}
relation authoriseSRole {…}
relation authoriseSLevel {…}
relation createDimensionSIARForSCompartment {…}
relation createDimensionSIARForSRole {…}
relation createDimensionSIARForSLevel {…}
relation createNegativeSIARAttributePermissionsForSLevel {…}
relation createNegativeSIARAttributePermissionsForSRole {…}
relation createNegativeSIARAttributePermissionsForSCompartment {...}
relation createDimensionAURForSLevel {…}

Table 4 SECDWSecurityRules2DimensionPermission (continued)

relation createDimensionAURForSRole {…}
relation createDimensionAURForSCompartment {…}
relation authoriseSLevelForAUR {…}
relation authoriseSCompartmentForAUR {…}
relation createAttributePermisionForAUR {…}
relation authoriseSRoleForAUR {…}

Audit Rules (ARs) have not been considered when developing the QVT transformations. Despite the fact that these rules are also considered as Security Rules, the OLAP tools do not implement AR defining permissions for cubes, dimensions or attributes, as the other kinds of rules do. The DW administrator is now in charge of defining which elements and information must be audited. To that end, the DW administrator uses special auditing tools, which are frequently provided by the OLAP tools. Once we have obtained logical models, the next step is to generate the final implementation. In previous works (Blanco et al., 2009), we have dealt with the implementation in a specific OLAP platform, SSAS.

5 Example

In this section, an application example of a web store sales service is used to show the representation of security rules at the conceptual model and their transformation at the logical level.

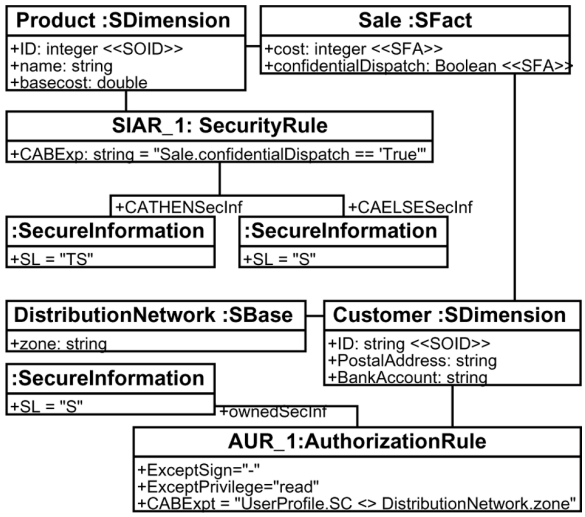
5.1 Secure conceptual model

Figure 3 shows the conceptual model of our example, defined according to the SECDW profile. The DW manages sales (secure fact class 'Sale') with information concerning their cost and whether the delivery is confidential or not. Furthermore, sales information is classified in two dimensions: 'Product' with information relating to identification, name and base cost of products, and 'Customer', which contains clients' information regarding identification, postal address and bank account. Customers are additionally aggregated by distribution zones through the use of a 'Distribution Network' base class.

The security classification used in this example is as follows: two security levels, Top Secret (TS) and Secret (S); one security compartment for each delivery zone (USA, Europe and Asia). The security roles have not been defined in this example because the DW is only queried by one user role: web store administrators.

Various security rules have also been defined. There is one SIAR that establishes a 'Top Secret' security level for information concerning products if the delivery is confidential (confidentialDispatch = true) or a 'Secret' security level if it is a normal delivery.

Figure 3 Conceptual model for the example



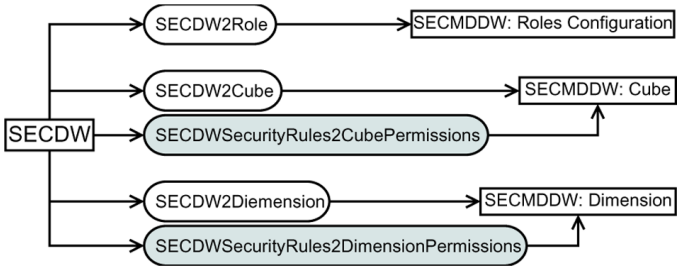
Moreover, a negative Authorisation Security Rule (AUR) hides customers' information from users with a low security level (Secret) according to the distribution areas ('Zone' attribute of 'DistributionNetwork'). That is, users with the security level 'Secret' can read customers' data from the same distribution zone (for instance, USA), but not the remaining information (for instance, Europe and Asia).

5.2 Secure multidimensional logical model

Once the conceptual model has been defined, QVT transformation rules are applied to generate a multidimensional logical model according to the SECMDDW metamodel. This metamodel is composed of three metamodels: a "security configuration metamodel", which represents the roles of an RBAC policy; a 'cube metamodel', which defines the structural aspects of cubes, measures and hierarchies and security constraints with cube and cell permissions; finally, a 'dimension metamodel' with structural information concerning dimensions, attributes and base classes, and security issues, which use permissions over dimensions and attributes.

Transformations are composed of several sets of rules (see Figure 4), which obtain a DW security configuration (SECDW2Role), structural aspects and security constraints related to cubes (SECDW2Cube) and dimensions (SECDW2Dimension).

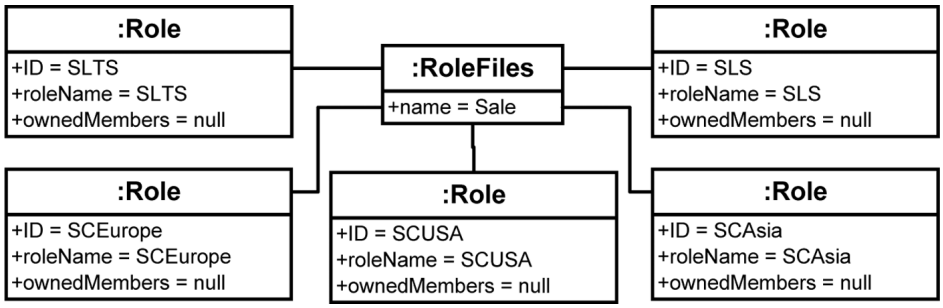
Figure 4 Transformation rules (see online version for colours)



These transformations have been improved with two new sets, which deal with security rules (SECDWSecurityRules2CubePermissions and SECDWSecurityRules2Dimension Permissions). This section shows how transformations are applied to our example, generating multidimensional logical models, focuses on security rules and briefly comments on the remaining transformations.

First, **SECDW2Role** creates a role-based security configuration at the logical level (see Figure 5). That is, it transforms each security level, compartment and role defined at the conceptual level into roles at the logical level to prepare the DW for the use of an RBAC policy. In our example, the roles 'SLTS' and 'SLS' are generated for security levels 'TS' and 'S', and 'SCUSA', 'SCEurope' and 'SCAsia' are generated for security compartments 'USA', 'Europe' and 'Asia'.

Figure 5 Logical model for the example (security configuration)



Next, structural aspects and some security issues for cubes and dimensions are generated by the **SECDW2Cube** transformation, which creates the 'Sales' cube and its associated measures and dimensions (see Figure 6), and by the **SECDW2Dimension** transformation which creates 'Product' and 'Customer' dimensions and the 'DistributionNetwork' base with their related attributes (see Figure 7).

Figure 6 Logical model for the example (cubes)

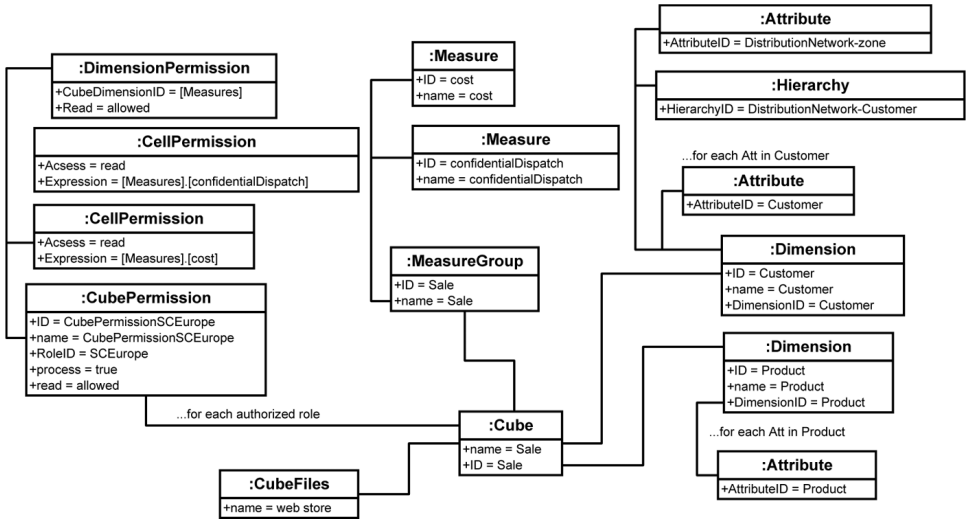
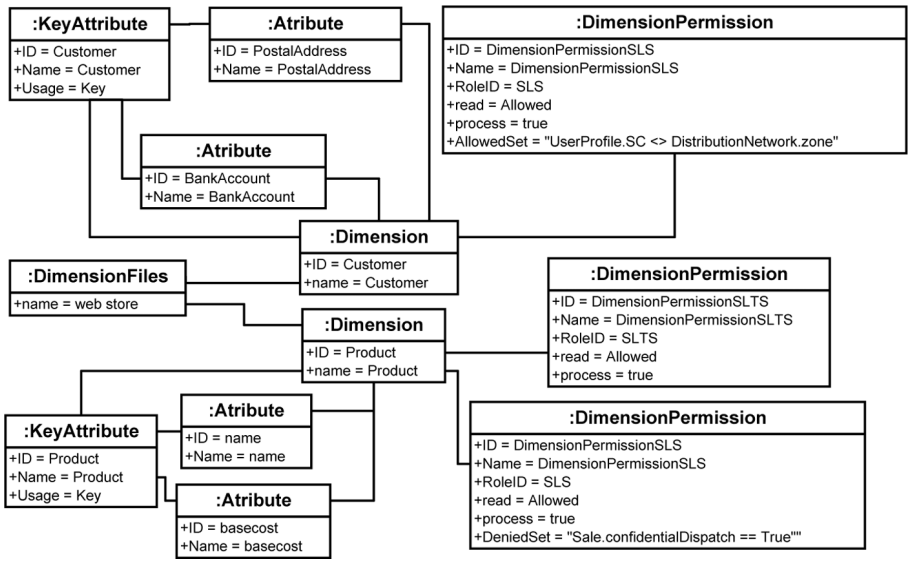


Figure 7 Logical model for the example (dimensions)



The security rules are now analysed and permissions at the cube and dimension levels are defined. First, the **SECDWSecurityRules2CubePermissions** transformation processes the SIAR and AUR security rules expressed, respectively, in the 'SecurityRule' and 'AuthorisationRule' of the PIM. Since the example includes neither SIAR nor AUR at the cube level, this transformation is not activated, and therefore the cube logical model (and thus the PSM) is not modified with these security rules.

SECDWSecurityRules2DimensionPermissions processes the aforesaid SIAR and AUR from the PIM. In the example shown, the 'Product' dimension and the 'Customer' dimension have an SIAR and an AUR, respectively. This transformation has two main relations, processDimensionSIAR and processDimensionAUR, each of which is intended to deal with the different kinds of security rules.

For a better understanding, Tables 5 and 6 show the flow of the execution for these relations. The rules in Table 5 are executed for the security rule SIAR_1 (which establishes a security level of TS to read the product data). The classes from the PIM, which throw the rule, have been included for each relation in the table. After executing the whole 'processDimensionSIAR', the model shown in Figure 7 is produced.

Table 5 Flow of relations to process SIAR rules

relation **processDimensionSIAR:** WebStore
relation **processDimensionSIAR:** SIAR_1
//authorising
relation **createDimensionSIARForSLevel:** TS, Product
relation **authoriseSLevel:** TS, Product
//denying
relation **createDimensionSIARForSLevel:** S, Product
relation **denySLevel:** S, Product

Table 6 Flow of relations to process AUR rules

relation **processDimensionAUR:** WebStore
relation **processDimensionAUR:** AUR_1
relation **createDimensionAURForSLevel:** S, Customer
relation **authoriseSLevelForAUR:** Customer, S,
"UserProfile.SC <> DistributionNetwork.zone", "-"

The AUR_1 authorisation rule denies users with the security level 'S' access to customers' data from a distribution zone, which is different from that of the user compartment. Furthermore, to process the AUR_1 security rule, the 'processDimensionAUR' relation is thrown. Table 6 depicts the relations, which are in turn executed to process the whole rule.

After executing all the transformations, the MD logic model for Dimensions includes two 'DimensionPermission' classes attached to the Product dimension (created from the SIAR_1 rule) and 1 'DimensionPermission' (created from the AUR_1 rule). These new classes place the semantic of the security rules from the PIM in the different PSMs.

6 Conclusions

Data Warehouses (DWs) manage vital historical information for the decision-making process, which have to be protected from unauthorised access. Our proposal provides several security models, which involve all the stages of the development process considering security as an important requirement. Furthermore, our models are aligned with an MDA and automatic transformations between models have been defined.

This architecture includes one relational path towards DBMS and another multidimensional path towards OLAP tools. The corresponding logical models and transformations were defined in previous works, but the support for more complex security rules (SIAR, AUR and AR) that can be established in the conceptual model using OCL expressions was not dealt with. This work deals with security rules and completes our approach improving metamodels to provide a better definition of security rules and defining a new set of QVT rules to support the automatic transformation of conceptual models.

We have applied our proposal to several case studies to validate it. We are currently applying it to a complex case study, which manages flight data, and we are also defining a formal validation. Furthermore, we are developing a tool that supports model design and their automatic transformations, providing scalability to our approach. Our further work is also focused on improving this architecture with dynamic models that deal with the inference problem related to OLAP operations, reengineering features to allow the inverse transformation from final code to conceptual models and code migration and automatic transformations for other final tools (such as Pentaho).

Acknowledgements

This research is part of the BUSINESS (PET2008-0136) Project financed by the "Ministerio de Ciencia e Innovación" (Spain), the SISTEMAS (PII2I09-0150-3135) Project financed by the "Regional Science and Technology Ministry of Castilla-La Mancha" (Spain), the QUASIMODO (PAC08-0157-0668) Project financed by the "Viceconsejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha" (Spain), the ESFINGE (TIN2006-15175-C05-05) Project granted by the "Dirección General de Investigación del Ministerio de Educación y Ciencia" (Spain), MITOS (TC20091098) Project financed by the University of Castilla-La Mancha (Spain) and the MEDUSAS (IDI-20090557) Project financed by the "Centro para el Desarrollo Tecnológico Industrial. Ministerio de Ciencia e Innovación (CDTI)" (Spain).

References

Basin, D., Doser, J. and Lodderstedt, T. (2006) 'Model driven security: from UML models to access control infrastructures', *ACM Transactions on Software Engineering and Methodology*, Vol. 15, No. 1, pp.39–91.

Blanco, C., García-Rodríguez de Guzmán, I., Rosado, D.G., Fernández-Medina, E. and Trujillo, J. (2009) 'Applying QVT in order to implement secure data warehouses in SQL server analysis services', *Journal of Research and Practice in Information Technology*, Vol. 41, No. 2, pp.119–138.

Essmayr, W. and Weippl, E. (2000) 'Identity mapping: an approach to unravel enterprise security management policies', *16th IFIP World Computer Congress (WCC 2000)*, Beijing, China, pp.79–88.

Fernández-Medina, E., Trujillo, J. and Piattini, M. (2007) 'Model driven multidimensional modeling of secure data warehouses', *European Journal of Information Systems*, Vol. 16, pp.374–389.

Fernández-Medina, E., Trujillo, J., Villarroel, R. and Piattini, M. (2006) 'Access control and audit model for the multidimensional modeling of data warehouses', *Decision Support Systems*, Vol. 42, pp.1270–1289.

Fernández-Medina, E., Trujillo, J., Villarroel, R. and Piattini, M. (2007) 'Developing secure data warehouses with a UML extension', *Information Systems*, Vol. 32, No. 6, pp.826–856.

Fink, T., Koch, M. and Pauls, K. (2006) 'An MDA approach to access control specifications using MOF and UML profiles', *Electronic Notes in Theoretical Computer Science*, Vol. 142, pp.161–179.

Inmon, H. (2002) *Building the Data Warehouse*, 3rd ed., John Wiley & Sons, USA.

Jürjens, J. (2004) *Secure Systems Development with UML*, Springer-Verlag, USA.

Katic, N., Quirchmayr, G., Schiefer, J., Stolba, M. and Min Tjoa, A. (1998) '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.

Kirkgöze, R., Katic, N., Stolda, M. and Min Tjoa, A. (1997) 'A security concept for OLAP', *8th International Workshop on Database and Expert System Applications (DEXA'97)*, IEEE Computer Society, Toulouse, France.

Lodderstedt, T., Basin, D. and Doser, J. (2002) 'SecureUML: a UML-based modeling language for model-driven security', *UML 2002, The Unified Modeling Language, Model Engineering, Languages Concepts, and Tools, 5th International Conference*, Springer, Dresden, Germany.

MDA, O.M.G. (2003) *Model Driven Architecture Guide*, Version 1.0.1. http://www.omg.org/cgi-bin/doc?omg/03-06-01

Mouratidis, H. and Giorgini, P. (2006) 'An introduction', *Integrating Security and Software Engineering: Advances and Future Visions*, Idea Group Publishing, USA.

Priebe, T. and Pernul, G. (2001) 'A pragmatic approach to conceptual modeling of OLAP security', *20th International Conference on Conceptual Modeling (ER 2001)*, Springer-Verlag, Yokohama, Japan.

Saltor, F., Oliva, M., Abelló, A. and Samos, J. (2002) 'Building secure data warehouse schemas from federated information systems', in Bestougeff, H., Dubois, J.E. and Thuraisingham, B. (Eds.): *Heterogeneous Inf. Exchange and Organizational Hubs.*, Kluwer Academic Publisher, Dordrecht, The Netherlands, pp.123–134.

Soler, E., Trujillo, J., Fernández-Medina, E. and Piattini, M. (2008) 'Building a secure star schema in data warehouses by an extension of the relational package from CWM', *Computer Standard and Interfaces*, Vol. 30, No. 6, pp.341–350.

Thuraisingham, B., Kantarcioglu, M. and Iyer, S. (2007) 'Extended RBAC-based design and implementation for a secure data warehouse', *International Journal of Business Intelligence and Data Mining (IJBIDM)*, Vol. 2, No. 4, pp.367–382.

Trujillo, J., Soler, E., Fernández-Medina, E. and Piattini, M. (2009) 'A UML 2.0 profile to define security requirements for datawarehouses', *Computer Standard and Interfaces*, Vol. 31, No. 5, pp.969–983.