Automatic Generation of Secure Multidimensional Code for Data Warehouses: An MDA Approach

Carlos Blanco[1], Ignacio García-Rodríguez de Guzmán[1], Eduardo Fernández-Medina[1], Juan Trujillo[2], and Mario Piattini[1]

[1] Dep. of Information Technologies and Systems. Escuela Superior de Informática
Alarcos Research Group – Institute of Information Technologies and Systems
Univ. of Castilla-La Mancha. Paseo de la Universidad, 4. 13071. Ciudad Real. Spain
{Carlos.Blanco,Ignacio.GRodriguez,Eduardo.Fdezmedina,
Mario.Piattini}@uclm.es
[2] Department of Information Languages and Systems. Facultad de Informática
University of Alicante. San Vicente s/n. 03690. Alicante. Spain
jtrujillo@dlsi.ua.es

Abstract. Data Warehouses (DW) manage enterprise information for the decision making process, and the establishment of security measures at all stages of the DW development process is also highly important as unauthorized users may discover vital business information. Model Driven Architecture (MDA) based approaches allow us to define models at different abstraction levels, along with the automatic transformations between them. This has thus led to the definition of an MDA architecture for the development of secure DWs. This paper uses an example of a hospital to show the benefits of applying the MDA approach to the development of secure DWs. The paper is focused on transforming secure multidimensional Platform Independent Models (PIM) at the conceptual level into Platform Specific Models (PSM) at the logical level by defining the necessary set of Query/Views/Transformations (QVT) rules. This PSM model is therefore used to obtain the corresponding secure multidimensional code for a specific On-Line Analytical Processing (OLAP) platform such as SQL Server Analysis Services (SSAS).

Keywords: Data Warehouses, Security, MDA, QVT, OLAP, SQL Server Analysis Services.

1 Introduction

The survival of organizations depends on the correct management of information security and confidentiality [1], and DWs manage enterprises' historical information which is used to support the decision making process and must be ensured by establishing security measures from the early stages of the development lifecycle [2]. Therefore, it is necessary to consider security constraints in models at all abstraction levels and to ultimately take these security issues into account in the final tools in order to avoid the situation of users being able to access unauthorized information by using operations.

R. Meersman and Z. Tari (Eds.): OTM 2008, Part II, LNCS 5332, pp. 1052–1068, 2008.
© Springer-Verlag Berlin Heidelberg 2008

Furthermore, MDA [3] is the Object Management Group (OMG) standard approach for model driven software development based on the separation of the specification of the system functionality and its implementation. MDA allows us to define models at different abstraction levels: computer-independent models (CIM) at business level and platform-independent models (PIM) at conceptual level which do not include information about specific platforms and technologies, and platform-specific models at logical level (PSM) with information about the specific technology used. Moreover, MDA proposes the use of *model transformations* as a mechanism with which to move from one level of abstraction to another, by transforming input models into new models or searching for matchings, among the other models involved. Many languages for model transformations exist. Nonetheless, the OMG proposes Query / Views / Transformations (QVT) [4] as a new standard for model transformation based on the Meta-Object Facility (MOF) standard [5] through which to define model transformation in an intuitive manner. Supporting the development of DWs with an MDA approach provides many advantages such a better separation of models including security requirements from the first stages of the DWs lifecycle and automatic translations through which to obtain other models and final code for different target platforms.

An architecture for developing secure DWs by using MDA and QVT transformations has been proposed in [6]. This architecture supports the modeling of secure DWs at different abstraction levels: CIM (specifying goals and subgoals), PIM (a multidimensional model), PSM (a relational model) and the final implementation in a database management system (DBMS). However, these aforementioned works are focused on a relational approach and the greatest part of DW is managed by OLAP tools over a multidimensional approach. We have therefore deployed a specialization of this architecture which defines a secure multidimensional PSM and implements secure DWs in SQL Server Analysis Services (SSAS) as a specific OLAP platform. In this architecture, we have decided to specify those QVT rules which directly transform our secure multidimensional PIM into a secure multidimensional PSM, and to then use this PSM to obtain secure multidimensional code for this OLAP platform (SSAS). In this paper we show the benefits of our approach through its application to an example. We show the steps involved in a conceptual model (PIM), a logical model (PSM), multidimensional secure code and the application of a set of QVT rules which transform the concepts of our secure multidimensional model at the conceptual level (PIM) into a logical model (PSM) which is used to obtain pieces of the code of our target platform (both the structural and security aspects of the final DW).

The remainder of the paper is organised as follows: Section 2 will present related work. Section 3 will introduce our MDA architecture for the development of secure DWs, and will be focused both on our source and target metamodels (PIM, PSM and code), and on the transformations which are necessary to obtain PSM from PIM and code from PSM. Section 4 will introduce our example regarding the admission system of a hospital. We will present models at the conceptual (PIM), logical (PSM) and code levels and will show how the proposed QVT transformations have been applied to obtain PSM from PIM. We will then demonstrate how to obtain the final code from PSM. Finally, Section 5 will present our conclusions and future work.

2 Related Work

OLAP systems are mechanisms with which to discover business information and use a multidimensional analysis of data to make strategic decisions. This information is organized according to the business parameters, and users can discover unauthorized data by applying a set of OLAP operations to the multidimensional view. Therefore, it is of vital importance for the organization to protect its data from unauthorized accesses. Several works attempting to include security issues in OLAP tools by implementing the previously defined security rules at a conceptual level have been proposed, but these works focus solely upon Discretional Access Control (DAC) policy and use a simplified role concept implemented as a subject. For instance, Katic et al. [7] proposed a DWs security model based on metamodels which provides us with views for each user group and uses DAC with classification and access rules for security objects and subjects. However, this model does not allow us to define complex confidentiality constraints. Kirkgöze et al. [8] defined a role-based security concept for OLAP by using a "constraints list" for each role, and this concept is implemented through the use of a discretional system in which roles are defined as subjects.

Priebe and Pernul later proposed a security design methodology, analyzed security requirements, classifying them into basic and advanced, and dealt with their implementation in commercial tools. Firstly, in [9] they used ADAPTed UML to define a DAC system with roles defined as subjects at a conceptual level. They then went on to implement this in SQL Server Analysis Servicies 2000 by using Multidimensional Expressions (MDX). They created a Multidimensional Security Constraint Language (MDSCL) based on MDX and put forward HIDE statements with which to represent negative authorization constraints on certain multidimensional elements: cube, measure, slice and level.

Our proposal uses an access control and audit model specifically designed for DWs to define security constraints in early stages of the development lifecycle. By using an MDA approach we consider security issues in all stages of the development process and automatically transform models at upper abstraction level towards logical models over a relational or multidimensional approach and finally obtain from these models secure code for DBMS or OLAP tools.

3 An MDA Approach for Developing Secure DWs

Our MDA architecture [6] is an adaptation of an MDA architecture for developing DWs [10] which has been improved with security capabilities. Our approach is made up of several models which allow us to model the DW at different abstraction levels (see Figure 1): at the business level (CIM) with a UML profile [11] based on the i* framework [12], which is an agent orientated approach towards requirement engineering centering on the intentional characteristics of the agent; at the conceptual level (PIM) with a UML profile called SECDW [13]; and at the logical level (PSM) with an extension of the relational package of Common Warehouse Metamodel (CWM) called SECRDW [14]. As has previously been mentioned, this paper considers a specialization of this architecture (represented in grey in Figure 1) focused on defining a PSM metamodel over a multidimensional approach and transforming structural and

security issues from PIM into this multidimensional PSM which allows us to obtain final multidimensional code for OLAP tools. The transformation from PSM models into secure multidimensional code for a specific OLAP platform (SSAS) is also treated in this paper. The source (PIM) and target (PSM) metamodels are briefly described in the following subsections, and an overview of the QVT rules defined to support this transformation will be presented.

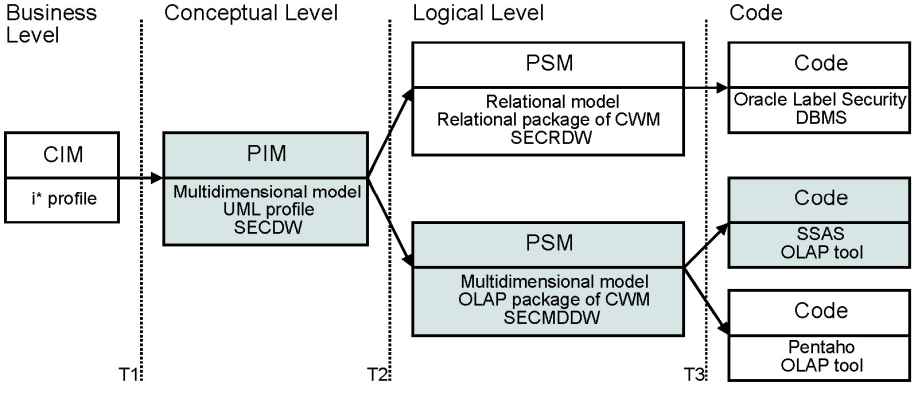

Fig. 1. MDA architecture for developing secure DWs

3.1 Secure Multidimensional PIM

A secure multidimensional conceptual metamodel called SECDW, has been defined in [13] by using a UML profile. This metamodel is shown in Figure 2 and is based on a UML profile for the conceptual design of DWs [15] which allows us to define fact, dimension and base classes, and considers specific aspects of DWs such as many-to-many relations, degenerated dimensions, multiple classifications or alternative paths of hierarchies. SECDW is enriched with security capabilities through the use of an access control and audit model (ACA) [16], which was specifically designed to consider security in DWs.

ACA allows us to define secure classes (SecureClass) and properties (SecureProperty), to classify authorization subjects and objects into security roles (SecurityRole) which organize users into a hierarchical role structure according to the responsibilities of each type of work, levels (SecurityLevel) which indicate the clearance level of the user, and compartments (SecurityCompartment) which classify users into a set of horizontal compartments or groups.

ACA also considers the definition of security rules over multidimensional elements of DWs by using stereotypes and Object Constraints Language (OCL) notes (Constraint). Three kinds of security rules are permitted: sensitive information assignment rules (SIAR) which specify multilevel security policies and allow us to define sensitivity information for each element in the multidimensional model; authorization rules (AUR) which permit or deny access to certain objects by defining the subject that the rule applies to, the object that the authorization refers to, the action that the rule refers to and the sign describing whether the rule permits or denies access; and audit rules (AR) to ensure that authorized users do not misuse their privileges.

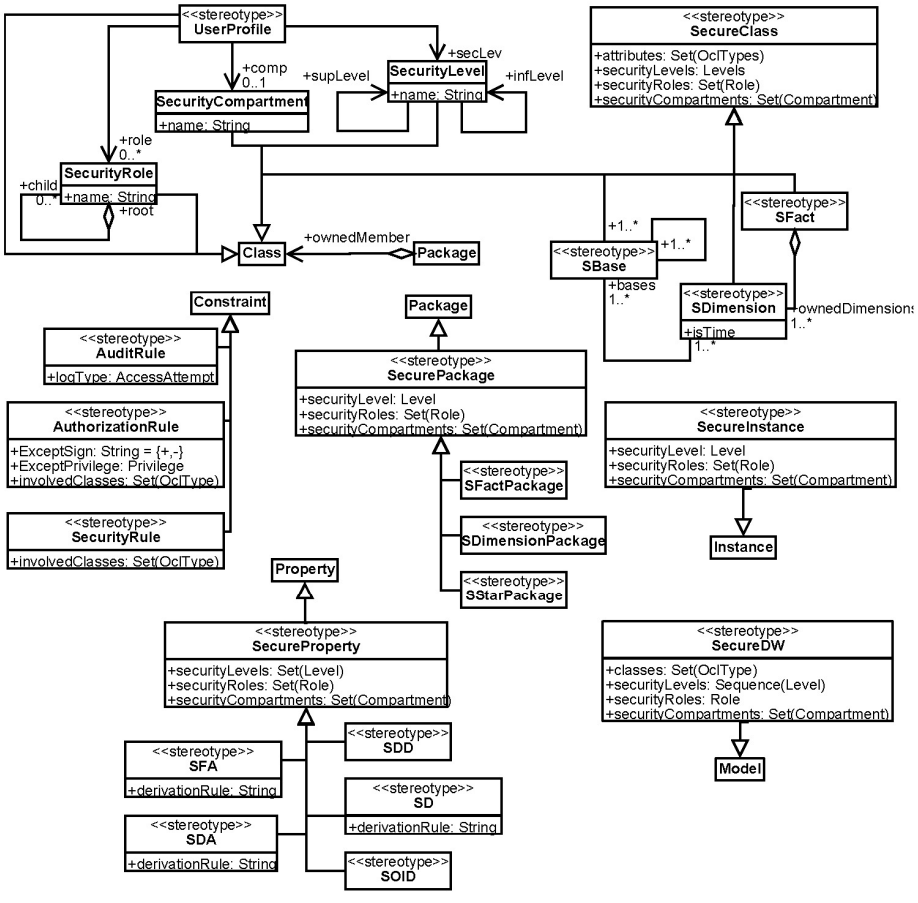

Fig. 2. Secure multidimensional PIM (SECDW)

3.2 Secure Multidimensional PSM

A secure multidimensional metamodel at the logical level (PSM), called SECMDDW, has been defined in this work by extending the OLAP package of CWM. This metamodel represents the intermediate step between conceptual and code levels, that is, our multidimensional PSM is obtained from PIM and can be used to obtain code towards different OLAP tools. Our PSM uses a multidimensional approach and considers the *security configuration* of the system, *structural* elements of the DW's and *security* constraints defined at class (fact, dimension or base) or attribute levels. Figure 3 shows the *security configuration* metamodel obtained. Our logical metamodel (PSM) only considers a role-based access control policy (RBAC) because the vast majority of OLAP tools use this policy and the PSM metamodel is closer to the final platform than the PIM metamodel. However, at the conceptual level we have considered security roles, levels and compartments defined by using our ACA model which have to be translated into roles. To accomplish this process we will follow the

methodology to implement secure DWs into OLAP tools presented in [17]. Furthermore, we have defined two metamodels to support the definition of the *structural* and *security* issues of cubes and dimensions.

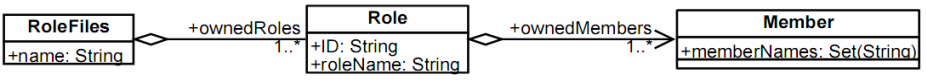

Fig. 3. Secure multidimensional PSM (SECMDDW): security configuration

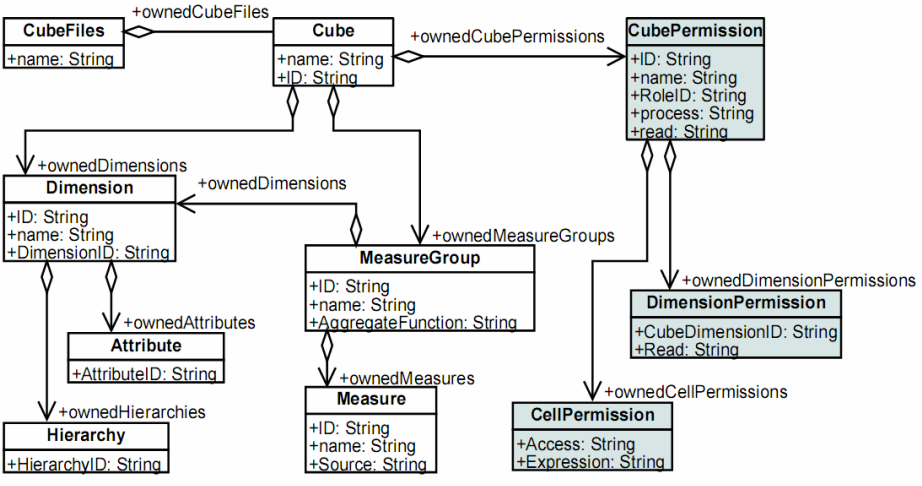

Fig. 4. Secure multidimensional PSM (SECMDDW): cubes

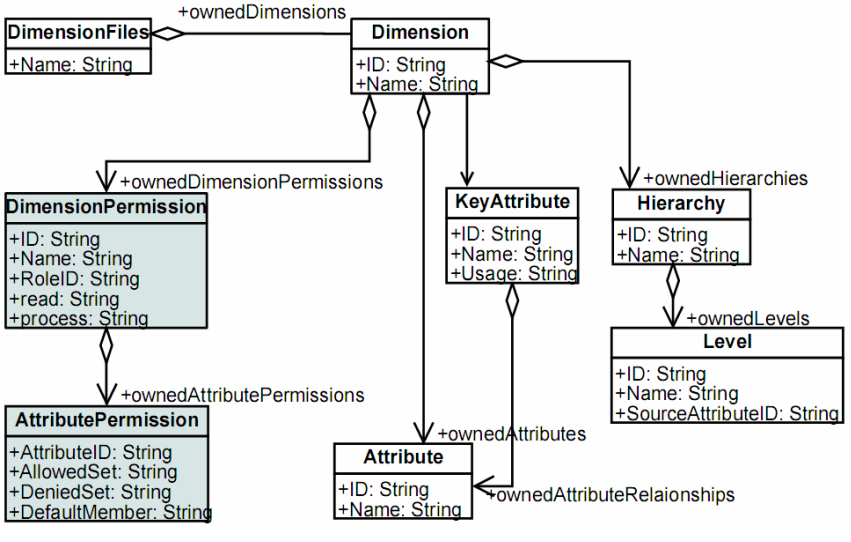

Fig. 5. Secure multidimensional PSM (SECMDDW): dimensions

Figures 4 and 5 show metamodels for cubes and dimensions which allow us to define facts classes (Cube), measures (MeasureGroup, Measure), dimension classes (Dimension), attributes (Attribute), hierarchies (Hierarchy) and base classes as attributes of the related dimension, and which also allow us to define security constraints over these multidimensional elements by using permissions over cubes (CubePermission), dimensions (DimensionPermission), cells (CellPermission) or attributes (AttributePermission). In these figures, the security-related aspects are represented in grey.

PIM to PSM transformation. A set of QVT transformations has been developed to automatically obtain secure multidimensional logical models, defined according to our PSM metamodel (SECMDDW), from conceptual models defined according to our PIM metamodel (SECDW). In order to develop these rules we have followed a methodology to implement multidimensional security in OLAP tools presented in [17]. These proposed transformations are made up of three main transformations which obtain our various kinds of target models from source conceptual models: *SECDW2Role*, *SECDW2Cube* and *SECDW2Dimension*.

SECDW2Role deals with the security configuration of the system. As our ACA model is richer than our PSM, which only considers roles, this transformation generates a new role for each security role (SR), level (SL) and compartment (SC) defined in our source model (PIM); *SECDW2Cube* generates the cube files that represent cubes, measures, dimensions, and cube permissions from the SECDW model; and *SECDW2Dimension* generates the dimension files that represent dimensions, bases, attributes, hierarchies and security permissions defined over dimensions and attributes. The security measures defined at the conceptual level (PIM) over classes or attributes by using SC, SR and SL, are translated into a set of permissions for involved roles (SC, SR and SL are translated into roles). As the transformations are quite verbose and the available space is limited, Section 4 will show some examples of complete rules and this section only presents the signatures of the developed rules.

SECDW2Role is composed of a top relation "*Package2RoleFiles* {…}" and relations "*SCompartment2Role* {…}", "*SRole2Role* {…}" and "*SLevel2Role* {…}". The signatures for the remainder of the developed rules are shown in Table 1.

Table 1. PIM to PSM transformations (signatures)

SECDW2Cube	**SECDW2Dimension**
top relation *Package2CubeFiles* {…}	top relation *Package2DimensionFiles* {…}
relation *SFact2Cube* {...} relation *CreateMeasureGroups* {...} relation *SProperty2Measure* {...} relation *SDimension2Dimension* {...} relation *ProcessSBase* {...} relation *CreateOwnedHierarchies* {...} relation *SProperty2Property* {...}	relation *SDimension2Dimension* {...} relation *KeyProperty2KeyAttribute* {...} relation *NonKeyProperty2Attribute* {...} relation *SBase2Attribute* {...}
relation *SCompartmentClass2CubePermission*{} relation *SRoleClass2CubePermission* {...} relation *SLevelClass2CubePermission* {...} relation *SCompartmentAtt2CellPermission* {...} relation *SRoleAtt2CellPermission* {...} relation *SLevelAtt2CellPermission* {...}	relation *createDimensionSIARForSCompartment* {...} relation *createDimensionSIARForSRole* {...} relation *createDimensionSIARForSLevel* {...} relation *authorizeSCompartment* {...} relation *authorizeSRole* {...} relation *authorizeSLevel* {...} relation *processSecureProperty* {...} relation *createPositiveAttributePermission* {...} relation *createNegativeAttributePermission* {...}

3.3 Secure Multidimensional Code

As a target OLAP platform we have selected SQL Server Analysis Services (SSAS), which deals with multidimensional elements and allows us to establish security measures over them. Furthermore, SSAS uses several kinds of XML files to manage this information, the most important of which are role, cubes and dimension files. We have analyzed this OLAP tool by studying how structural and security information could be implemented in this platform, in order to obtain secure multidimensional code from PSM.

PSM to Code Transformation. Obtaining secure multidimensional code from our secure multidimensional PSM is a simple task since both consider structural and security issues by using a multidimensional approach and the vast majority of the destination concepts are defined in our source metamodel. This paper is focused on obtaining PSM from PIM, but also deals with the transformation of PSM into a specific OLAP platform, SSAS. In order to obtain code for the security measures defined in the conceptual models we have followed the methodology to implement multidimensional security in SSAS which is presented in [17]. The presentation of our example will show a portion of code in SSAS and screenshots of the final implementation in SSAS.

4 Applying MDA for Developing Secure DWs

This section presents the application of our MDA architecture to an example of a hospital that wishes to automate its admission process and requires confidentiality for the information involved. This example will be used to show the application of the transformations to obtain a logical model (PSM) according to our target metamodel and to obtain secure multidimensional code in SSAS from PSM.

4.1 Secure Multidimensional PIM

Figure 6 shows the conceptual model (defined as an instance of the SECDW model) for our hospital which is required to resolve the aforementioned problem. The security configuration of the hospital uses a classification of users and objects in security roles (SR) and security levels (SL). Security compartments have not been defined since they depend on organization policies. The user roles (SR) might be “HospitalEmployee”, “Health” (including “Doctor” and “Nurse” roles) and “NonHealth” (including “Admin” and “Maintance” roles). The levels of security (SL) used are top secret (TS), secret (S), confidential (C) and undefined (U). The secure fact class “Admission” contains two secure dimension classes (“Diagnosis” and “Patient”). The “UserProfile” metaclass contains information about all the users who will have access to this secure multidimensional model. This information can also define characteristics of users such us age, citizenship, etc. which can be used to establish complex security constraints. We have also defined a set of sensitive information assignment rules (SIAR) over classes and attributes: instances of “Admission” fact class or “Patient” dimension can be accessed by the “Admin” or “Health” roles and the Secret (or upper) security level; the “Diagnosis” dimension can be accessed by the “Health” role and the Secret (or upper) security level; the bases “City” and “DiagnosisGroup” can be accessed by the Confidential (or upper) security level; and attributes “Admission.cost” and “Patient.address” can be accessed by the “Admin” role.

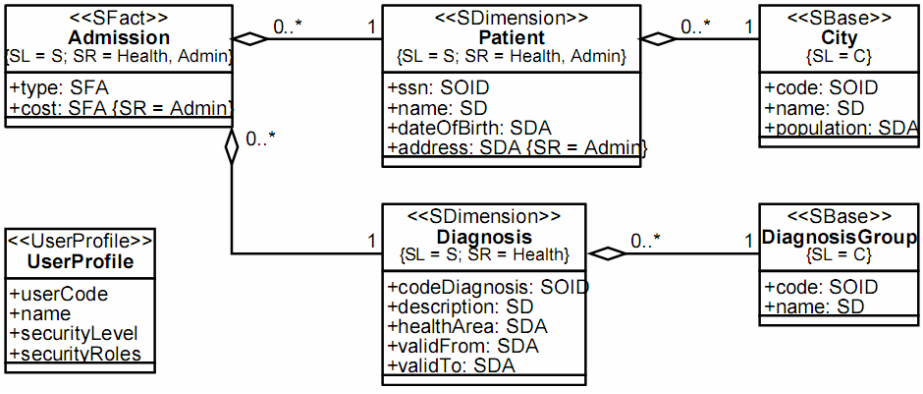

Fig. 6. Secure multidimensional PIM for hospital

4.2 Secure Multidimensional PSM

In this section we obtain a logical model (PSM) from the conceptual model defined above for a hospital according to the SECDW metamodel by using a set of QVT transformations (see Table 1). We have applied our three main defined transformations (SECDW2Role, SECDW2Cube and SECDW2Dimension) and we present the resulting logical models according to our PSM metamodel. These were obtained from conceptual models according to our PIM metamodel.

SECDW2Role. Table 2 shows the application of the *SECDW2Role* transformation to the hospital. The *SRole2Role* rule creates for each security role "r" detected in the source model a new role called "SRr". The QVT code for the rule SLevel2Role is shown in Table 3 and also creates a new role called “SLn” for each security level “n” defined at conceptual level, that is, in this example creates “SLTS”, “SLS”, “SLC” and “SLU” roles. *SCompartment2Role* has not been shown because security compartments have not been defined in the hospital.

Table 2. SECDW2Role transformation for hospital

top relation **Package2RoleFiles:** Hospital
relation **SRole2Role:** HospitalEmployee, Health, Doctor, Nurse, NonHealth, Admin, Maintenance relation **SLevel2Role:** TS, S, C, U relation **SCompartment2Role:** not thrown

Table 3. Relation SLevel2Role

relation ***SLevel2Role***

checkonly domain psm sl:SRole{
name = n; }
enforce domain pim r:Role{
fileName = "SL"+n+".role";
ID = "SL"+n;
roleName = "SL"+n;
ownedMembers = OWNEDMEMBS:Set(Member); }

The target model with the security configuration for the hospital has been represented in Figure 7 according to PSM metamodel (SECMDDW). This model defines roles at logical level for each security role and security level detected at conceptual level.

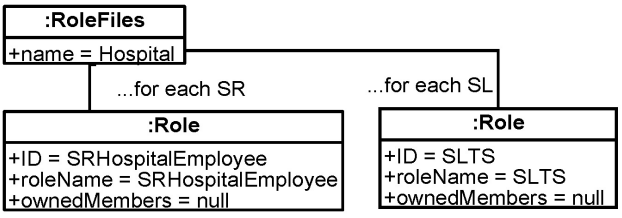

Fig. 7. Secure multidimensional PSM for hospital: security configuration

SECDW2Cube. Next, the *SECDW2Cube* transformation obtains the structure and security of cubes defined in the hospital. Table 4 shows the process: *SFact2Cube* rule creates the "Admission" cube; then the *CreateMeasuresGroups* and *SProperty2Measure* rules create measures and the remainder of the structural rules create dimensions, attributes and hierarchies for dimensions and bases related to the "Admission" cube. Finally, security rules analyze the security constraints defined over the fact class and its attributes, and define cube and cell permissions for involved security roles (which represent the SR, SL and SC of the source model).

Table 4. SECDW2Cube transformation for hospital

top relation **Package2RoleFiles:** Hospital
relation **SFact2Cube:** Admission relation **CreateMeasureGroups:** Admission relation **SProperty2Measure:** type, cost relation **SDimension2Dimension:** Patient, Diagnosis relation **ProcessSBase:** City, DiagnosisGroup relation **CreateOwnedHierarchies:** City-Patient, DiagnosisGroup-Diagnosis relation **SProperty2Attribute:** (for Patient) ssn, name, dateOfBirth, address (for Diagnosis) codeDiagnosis, description, healthArea, validFrom, validTo (for City) code, name, population (for DiagnosisGroup) code, name
relation **SCompartmentClass2CubePermission:** Not thrown relation **SRoleClass2CubePermission:** (for Admission) Health, Admin relation **SLevelClass2CubePermission:** (for Admission) S relation **SCompartmentAtt2CellPermission:** Not thrown relation **SRoleAtt2CellPermission:** (for Admission.address) Admin relation **SLevelAtt2CellPermission:** Not thrown

Table 5 shows the code for “SLevelClass2CubePermission” rule which permits accesses to cube by creating cube permissions for each authorized role that represent allowed security levels. In this example, cube permissions allowing access to security level secret “S” (SLS role) and its upper security levels (SLTS role) are defined in “Admission” class.

Figure 8 shows the model obtained from SECDW2Cube transformation in which has been created an “Admission” cube with its measure groups (including “type” and “cost” measures), related dimensions (“Patient” and “Diagnosis”), cube permissions

Table 5. Relation SLevelClass2CubePermission

relation ***SLevelClass2CubePermission***

checkonly domain psm sl:SLevel {
name = n; }
enforce domain pim c:Cube {
name = cubeName;
ID = cubeName;
ownedCubePermissions = OWNCUBEPERMS:Set(CubePermission); }
enforce domain pim cp:CubePermission {
ID = "CubePermission"+n;
name = "CubePermission"+n;
RoleID = n;
Process = "true";
Read = "Allowed"; }
where{ OWNCUBEPERMS->including(cp); }

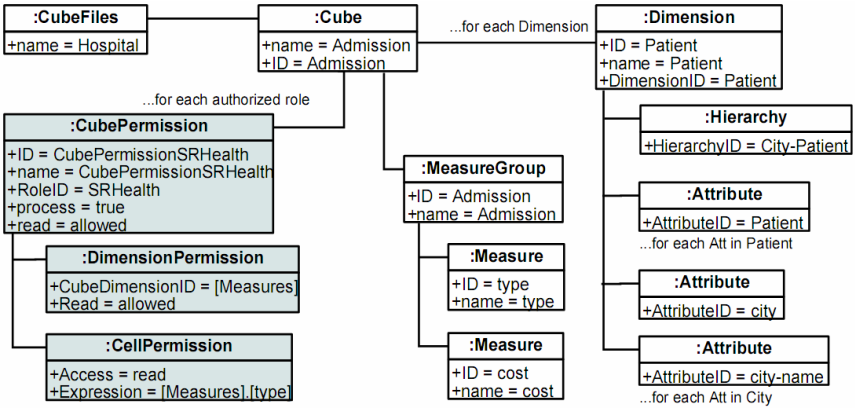

Fig. 8. Secure multidimensional PSM for hospital: cube

over "Admission" cube allowing accesses to authorized roles ("SRHealth", "SRAdmin" and its descendants, and "SLS" and upper levels) and cell permissions allowing accesses to each allowed measure ("SRAdmin" cannot access "cost" measure) .

SECDW2Dimension. Finally, the *SECDW2Dimension* transformation obtains the structure and security of dimensions and bases by following the process shown in Table 6. Firstly, structural rules obtain dimensions, hierarchies and attributes for the dimensions and bases defined in our SECDW model. Each attribute of the base classes is added as an attribute of its related dimension in our target model. Next, the security rules are applied. These are composed of several rules which obtain security constraints defined over classes (dimension or base classes) and their attributes, and they then analyze the security roles which are involved to obtain dimension and attribute permissions. Security constraints detected at the class level generate dimension permissions for each authorized role, allowing access to this class. Attribute

Table 6. SECDW2Dimension transformation for hospital

top relation **Package2Dimension:** Hospital
relation **SDimension2Dimension:** Patient, Diagnosis relation **KeyProperty2KeyAttribute:** (for Patient) ssn (for Diagnosis) codediagnosis relation **NonKeyProperty2Attribute:** (for Patient) name, dateOfBirth, address (for Diagnosis) description, healthArea, validFrom, validTo relation **SBase2Attributes:** City, DiagnosisGroup
relation **createDimensionSIARForSCompartment:** Not thrown relation **createDimensionSIARForSRole:** (for Patient) Health, Admin (for Diagnosis) Health relation **createDimensionSIARForLevel:** (for Patient) S (for Diagnosis) S (for City) C (for DiagnosisGroup) C relation **authorizeSCompartment:** Not thrown relation **authorizeSRole:** (for Patient) Health, Admin and their descendants (for Diagnosis) Health and its descendants relation **authorizeSLevel:** (for Patient) S, TS (for Diagnosis) S, TS (for City) C, S, TS (for DiagnosisGroup) C, S, TS relation **processSecureProperty:** (for Patient) address relation **createPossitiveAttributePermission:** allowed roles (Admin and its descendants) relation **createNegativeAttributePermission:** denied roles (distinct to allowed roles)

permissions are also created for the security constraints defined at the attribute level, defining positive attribute permissions for each authorized role and negative attribute permissions to avoid access to unauthorized users.

Table 7 shows a piece of source code for one rule of our developed set of QVT transformation which obtains security constraints defined at the conceptual level over attributes and establishes this constraint by using an attribute permission with an explicit denial over this attribute for the involved roles. In this example, a security constraint over "address" attribute of "Patient" dimension allowing access to security role "Admin" was defined. This rule creates attribute permissions for each unauthorized roles (roles distinct to "SRAdmin" and its descendants) with a denied set over "address" attribute of "Patient" dimension.

Table 7. Relation createNegativeAttributePermissions

relation ***createNegativeAttributePermissions***

checkonly domain pim sp:SecureProperty {
  name = spName; }
enforce domain psm dp:DimensionPermission {
  ID = "DimensionPermission"+ID;
  Name = "DimensionPermission"+ID;
  ownedAttributePermissions= OWNATTPERMS:Set(AttributePermission);}
enforce domain psm at:AttributePermission {
  AttributeID = spName;
  DeniedSet = "["+sp.class.name+"].["+sp.name+"]"; }

Figure 9 shows the model obtained from SECDW2Dimension transformation in which are defined each dimension ("Patient" and "Diagnosis"), attributes (key attributes, non key attributes and attributes derived from its related bases), hierarchies and security permissions over dimensions and attributes (positive and negative permissions).

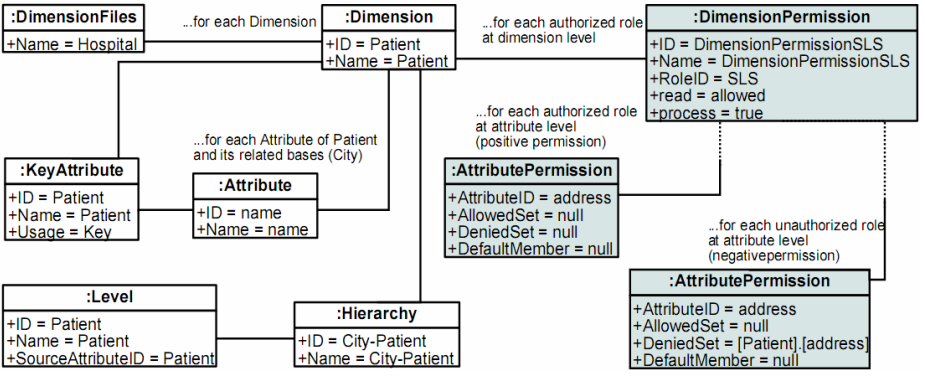

Fig. 9. Secure multidimensional PSM for hospital: dimensions

4.3 Secure Multidimensional Code

Finally, the security multidimensional code for SSAS is obtained from the logical model (PSM). Although SSAS has some particularities, this model, with which to text transformation, is easy to obtain. SSAS manages multidimensional elements (such as cubes, dimensions, hierarchies or measures) and also considers the establishment of security measures over these multidimensional elements with security permissions over cubes, dimensions, cells and attributes.

Table 8. Secure multidimensional Code for hospital: Admission cube

```
<Cube>
  <ID>Admission</ID>
  <Name>Admission</Name>

  <Dimensions>…</Dimensions>
  <MeasureGroups>…</MeasureGroups>

  <CubePermissions>
    <CubePermission>
      <ID>CubePermissionSLS</ID>
      <Name>CubePermissionSLS</Name>
      <RoleID>SLS</RoleID>
      <Process>true</Process>
      <Read>Allowed</Read>
      <CellPermissions>
        <CellPermission>
          <Access>Read</Access>
          <Expression>[Measures].[type]</Expression>
        </CellPermission>
      </CellPermissions>
    </CubePermission>
     …(cube permissions for each authorized role)

  </CubePermissions>
</Cube>
```

The first example correspond to a security rule defined at conceptual level that allows accesses to "Admission" measures for security level secret or upper and security roles "Health", "Admin" and their descendants. Table 8 shows a piece of the final code for SSAS with a cell permission over attribute "type" that allows access to security level secret (role "SLS"). In this example we have used a positive permission to allow access to "type" and we have thus denied access to the remaining cube measures (attributes of the "Measures" dimension).

At conceptual level we have defined a rule that hides the "Diagnosis" dimension from users with a security level which is lower than "Secret" ("SLC" and "SLU" at the logical level) and with a security role which is not "Health" or "Admin" or their descendents ("SRMaintance" at the logical level). Table 9 shows secure multidimensional code obtained from PSM for the "Diagnosis" dimension which hides all its attributes from unauthorized roles. Due to space constraints, this table only shows a piece of the code in which the "DiagnosisGroup" attribute is hidden from users with the "Confidential" security level ("SLC" role). The rest of the code similarly defines attribute permissions for each attribute of the "Diagnosis" dimension and dimension permission for each unauthorized role.

Table 9. Secure multidimensional Code for hospital: Diagnosis dimension

```
<Dimension>
  <ID>Diagnosis</ID>
  <Name>Diagnosis</Name>
  <Attributes>…</Attributes>
  <DimensionPermissions>
    <DimensionPermission>
      <ID>DimensionPermissionSLC</ID>
      <Name>DimensionPermissionSLC</Name>
      <RoleID>SLC</RoleID>
      <Read>Allowed</Read>
      <AttributePermissions>
        <AttributePermission>
          <AttributeID>DiagnosisGroup</AttributeID>
          <DeniedSet>[Diagnosis].[DiagnosisGroup]</DeniedSet>
        </AttributePermission>
        …(attribute permissions for each attribute of Diagnosis)
      </AttributePermissions>
    </DimensionPermission>
    …(dimension permissions for each unauthorized role)
  </DimensionPermissions>
</Dimension>
```

Figures 10 and 11 show screenshots of the code generated for this example by working with SSAS in which we can see the result of executing two queries that check a security rule defined at conceptual level that has been automatically translated at logical level by using the set of QVT rules defined and has been finally implemented in SSAS by obtaining the corresponding secure multidimensional code from this logical model. At conceptual level, in our PIM model (Figure 6), we have define a security constraint over "Patient" dimension that permits accesses to security level secret and upper and security roles "Health", "Admin" and their descendants. When

logical models are obtained from this conceptual model by applying our set of QVT rules (see Figure 9), this security constraint is transformed into dimension permissions that deny accesses to unauthorized roles (security levels “SLC” and “SLU”, and security roles distinct to authorized roles).

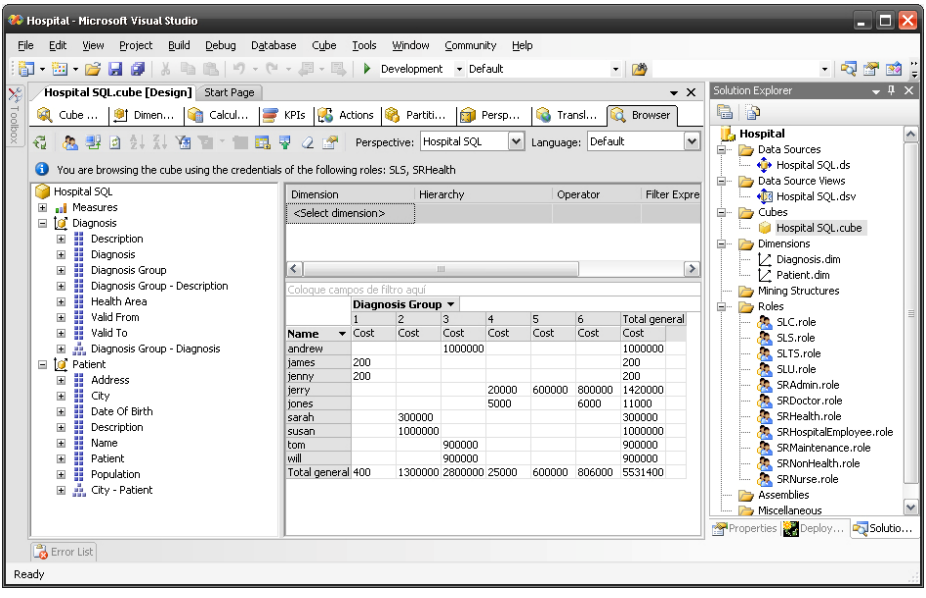

Fig. 10. SSAS implementation for hospital: authorized query

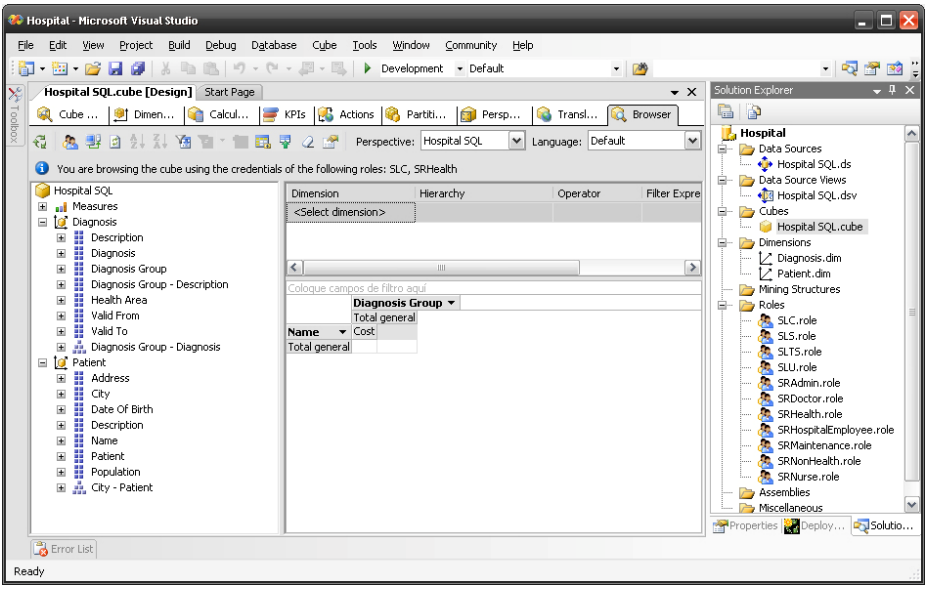

Fig. 11. SSAS implementation for hospital: unauthorized query

Firstly, an authorized user with the security level "Secret" ("SLS" role at the logical level) and the security role "Health" ("SRHealth" role at the logical level) makes a query involving attributes from the "Diagnosis" and "Patient" dimensions. Figure 10 shows the result of this query. Next, an unauthorized user with a lower security level, "Confidential" ("SLC" role at the logical level) and the same security role, "Health", makes the same query, but in this case it is an unauthorized query and the requested information is hidden. Figure 11 shows the result of this unauthorized query.

5 Conclusions

This work shows the advantages of applying an MDA approach to the development of DWs by analyzing PIM to PSM and PSM to code transformations, which is then applied to an example. This approach allows us to automatically develop DWs, thus saving time and money and obtaining better quality and security by translating the requirements identified at early stages of development into the final implementation.

We have defined the necessary metamodels at the logical level (PSM), multidimensional secure code for a specific OLAP platform (SSAS) and the transformations to obtain PSM from conceptual models defined according to our SECDW metamodel and secure multidimensional code in SSAS from PSM. Furthermore, we have analyzed an example in which we have obtained secure multidimensional PSM and code from a conceptual model of a hospital.

In future works, we intend to improve our MDA architecture for the development of secure DWs in several ways. New security rules and constraints will be included in our ACA model in order to consider the security threats related to specific OLAP operations such as navigations or inferences. These transformations from PIM with which to include the advanced security rules defined in SECDW will be extended by using OCL notes, and we shall also define the transformation from PSM to code for other OLAP tools such as Pentaho and Oracle, and the inverse transformations from code to PSM and PIM.

Acknowledgments. This research is part of the ESFINGE (TIN2006-15175-C05-05) and METASIGN (TIN2004-00779) Projects financed by the Spanish Ministry of Education and Science, and of the MISTICO (PBC-06-0082) Project financed by the FEDER and the Regional Science and Technology Ministry of Castilla-La Mancha (Spain).

References

1. Dhillon, G., Backhouse, y.J.: Information system security management in the new millennium. Communications of the ACM 43(7), 125–128 (2000)
2. Mouratidis, H., Giorgini, y.P.: An Introduction. In: Integrating Security and Software Engineering: Advances and Future Visions. Idea Group Publishing (2006)
3. MDA, O.M.G., Model Driven Architecture Guide (2003)
4. OMG, MOF QVT final adopted specification (2005)
5. OMG, Meta Object Facility (MOF) specification (2002)
6. Fernández-Medina, E., Trujillo, J., Piattini, y.M.: Model Driven Multidimensional Modeling of Secure Data Warehouses. European Journal of Information Systems 16, 374–389 (2007)

7. Katic, N., Quirchmayr, G., Schiefer, J., Stolba, M., Tjoa, y.A.: A Prototype Model for DW Security Based on Metadata. In: en 9th Int. Workshop on DB and Expert Systems Applications, Vienna, Austria (1998)
8. Kirkgöze, R., Katic, N., Stolda, M., Tjoa, y.A.: A Security Concept for OLAP. In: en 8th Int. Workshop on Database and Expert System Applications, Toulouse, France (1997)
9. Priebe, T., Pernul, y.G.: A Pragmatic Approach to Conceptual Modeling of OLAP Security. In: en 20th Int. Conference on Conceptual Modeling, Yokohama, Japan (2001)
10. Mazón, J.-N., Trujillo, y.J.: An MDA approach for the development of data warehouses. Decision Support Systems 45(1), 41–58 (2008)
11. Soler, E., Stefanov, V., Mazón, J.-N., Trujillo, J., Fernández-Medina, E., Piattini, y.M.: Towards Comprehensive Requirement Analysis for Data Warehouses: Considering Security Requirements. In: en Proccedings of The Third International Conference on Availability, Reliability and Security (ARES). IEEE Computer Society, Barcelona (2008)
12. Yu, E.: Towards modelling and reasoning support for early-phase requirements engineering. In: en 3rd IEEE International Symposium on Requirements Engineering (RE 1997), Washington, DC (1997)
13. Fernández-Medina, E., Trujillo, J., Villarroel, R., Piattini, y.M.: Developing secure data warehouses with a UML extension. Information Systems 32(6), 826–856 (2007)
14. Soler, E., Trujillo, J., Fernández-Medina, E., Piattini, y.M.: SECRDW: An Extension of the Relational Package from CWM for Representing Secure Data Warehouses at the Logical Level. In: en International Workshop on Security in Information Systems, Funchal, Madeira, Portugal (2007)
15. Luján-Mora, S., Trujillo, J., Song, y.I.-Y.: A UML profile for multidimensional modeling in data warehouses. Data & Knowledge Engineering 59(3), 725–769 (2006)
16. Fernández-Medina, E., Trujillo, J., Villarroel, R., Piattini, y.M.: Access control and audit model for the multidimensional modeling of data warehouses. Decision Support Systems 42(3), 1270–1289 (2006)
17. Blanco, C., Fernández-Medina, E., Trujillo, J., Piattini, y.M.: Implementing Multidimensional Security into OLAP Tools. In: en Third International Workshop Dependability Aspects on Data WArehousing and Mining applications (DAWAM 2008). IEEE Computer Society, Barcelona (2008)

Appendix A: Acronyms

ACA: Access Control and Audit model
AR: Audit Rule
AUR: Authorization Rule
C: Confidential
CIM: Computer Independent Model
CWM: Common Warehouse Metamodel
DAC: Discretional Access Control
DBMS: Database Management System
DW: Data Warehouse
MDA: Model Driven Architecture
MDSCL: Multidimensional Security Constraint Language
MDX: Multidimensional Expressions
MOF: Meta-Object Facility
OCL: Object Constraints Language
OLAP: On-Line Analytical Processing
OMG: Object Management Group
PIM: Platform Independent Model
PSM: Platform Specific Model
QVT: Query / Views / Transformations
RBAC: Role-Based Access Control
S: Secret
SC: Security Compartment
SIAR: Sensitive Information Assignment Rule
SL: Security Level
SR: Security Role
SSAS: SQL Server Analysis Services
TS: Top Secret
U: Undefined