From UML to ROLAP multidimensional databases using a pivot model

Nicolas Prat* — Jacky Akoka**

* ESSEC
Avenue Bernard Hirsch, BP 105, F-95021 Cergy cedex
prat@essec.fr

** CEDRIC-CNAM & INT
292, rue Saint-Martin, F-75141 Paris cedex 03
akoka@cnam.fr

ABSTRACT. Effective data warehouse design requires a conceptual modeling phase. This paper describes a data warehouse design method based on the conceptual, logical and physical levels. The conceptual phase creates an UML model. To this end, UML is enriched with multidimensional concepts. The logical phase maps the enriched UML model into a multidimensional one. The physical phase maps the multidimensional model into a database schema, depending on the target ROLAP or MOLAP tool. Our method consists in (1) the metamodels used at each design step, including the unified multidimensional metamodel (2) a set of transformations defined on the concepts of these metamodels, facilitating the design process. We present the metamodels and transformations, and illustrate the formal specification of the transformations with OMG’s Object Constraint Language (OCL). We apply our method to a case study and compare it to the state-of-the-art.

RÉSUMÉ. Pour être efficace, la conception d’entrepôts de données nécessite une phase de modélisation conceptuelle. Cet article décrit une méthode de conception d’entrepôts de données fondée sur les niveaux conceptuel, logique et physique. La phase conceptuelle crée un modèle UML. A cette fin, UML est enrichi de concepts multidimensionnels. La phase logique traduit le modèle UML enrichi en modèle multidimensionnel. La phase physique traduit le modèle multidimensionnel en un schéma de base de données, en fonction de l’outil ROLAP ou MOLAP cible. Notre méthode est constituée (1) des métamodèles utilisés à chaque étape de conception, dont le métamodèle multidimensionnel unifié (2) d’un ensemble de transformations définies sur les concepts de ces métamodèles pour faciliter le processus de conception. Nous présentons les métamodèles et les transformations, et illustrons la spécification formelle des transformations avec OCL (Object Constraint Language) de l’OMG. Nous illustrons notre méthode par une étude de cas et la comparons à l’état de l’art.

KEYWORDS: data warehouse, multidimensional database, OLAP, design method, UML, OCL.

MOTS-CLÉS : entrepôt de données, base de données multidimensionnelle, OLAP, méthode de conception, UML, OCL.
1. Introduction

The data warehousing and OLAP market is growing rapidly. [OLA 02] estimates that the worldwide OLAP total market is likely to exceed 5 billion dollars by 2004, compared with 1 billion dollars in 1996. Like the relational database market at its beginning, the OLAP market has no dominant players.

The OLAP design process is described by tool vendors as much easier than classical database design. We claim that, since data warehouses are developed to provide managers with data on which they will build queries depending on their constantly evolving needs, the design process is crucial.

Each OLAP tool is based on a specific underlying physical metamodel. We claim that a standardization of the multidimensional metamodel will allow both the users to better understand the underlying concepts and the tool editors to adopt a unified view.

Unlike the transactional database world which has taken some time to adopt the three levels of abstraction recommended by ANSI/X3/SPARC, the data warehouse actors should rapidly divide the modeling task according to the three conceptual, logical and physical levels. The conceptual level allows the data warehouse designer to build a high level abstraction of data for decision making, independently from implementation issues. The logical level maps this abstraction into a standard multidimensional representation. Finally, taking into account the target tool, the physical phase aims at building a database schema to be implemented on a specific platform.

The main objective of this paper is to propose a four-step design method for data warehouse development, including transformations for systematic mapping between each step. In order to address several target tools, we define a unified multidimensional metamodel at the logical level and build our process on it. The paper is organized as follows. Section 2 describes the unified multidimensional metamodel. Section 3 presents the four-step design method. The transformations applied at each step are described and formalized with OCL, the Object Constraint Language associated with UML [OMG 01a]. A case study illustrating the process is presented in Section 4. Section 5 compares our approach with the state-of-the-art. Finally, Section 6 concludes and describes further research.
2. A unified multidimensional metamodel

In contrast with the relational metamodel\(^1\), there is no standard multidimensional database metamodel. More precisely, there is no commonly accepted formal multidimensional data metamodel. As a consequence, many multidimensional metamodels have been proposed in the literature [AGR 97; BLA 98; CAB 98; CHA 97; GOL 98a; GYS 97; KIM 96; LIW 96; OMG 01b; PED 99; VAS 99]. The concepts used vary depending on the authors and some concepts, e.g. the notion of “fact”, are employed with various meanings. Furthermore, there is no consensus concerning the level of the multidimensional metamodel (physical, logical or conceptual). The star and snowflake models presented in [KIM 96] have often been considered to be at the physical level, since the choice between stars and snowflakes is based on performance considerations (trade-off between query performance and optimization of disk space). More recent publications have placed the multidimensional metamodel at the logical level [VAS 00] or even at the conceptual level [GOL 98a; HÜS 00].

Our strong belief is that the multidimensional metamodel belongs to the logical level. Even though there is no consensus on this metamodel, it clearly exists independently of physical implementations. However, the multidimensional metamodel should not be situated at the conceptual level since the concepts of this metamodel (e.g. the concept of dimension) are not as close to reality as concepts like the object (used in conceptual object modeling languages like UML [OMG 01a] for example). There is indeed a strong parallel between the relational and the multidimensional metamodels - e.g. the definitions or attempts to define a (standard) associated query language and algebra. This is the reason why we argue that both metamodels should be considered as belonging to the same level, i.e. the logical level.

In the multidimensional metamodel, data are organized in (hyper)cubes. Although the detailed concepts of this model vary depending on the authors, we can describe multidimensional semantics using four concepts which appear recurrently, namely the notions of measure, dimension, hierarchy and attribute. Our metamodel is composed of these four concepts, which are illustrated in Figure 1.

\(^1\) To be consistent with the conventions of the Object Management Group [OMG 01a; OMG 01b], we will refer to modeling formalisms (e.g. relational, UML…) as metamodels.
The key concept of the multidimensional metamodel is the notion of measure. A measure is typically a quantitative data, a numeric value of interest for the analysis. A measure needs not be of numeric type, as long as its values are totally ordered. For example, it can be an enumeration type. Thus, the satisfaction of customers with a product may be measured on a four-value scale (unsatisfied – mitigated – satisfied – enthusiastic). In a cube, the measures correspond to the content of the cells.

The dimensions form the edges of the cube. Each measure is associated with one or several dimensions, which specify the context of the measure. In the example of Figure 1, the quantity sold is dimensioned by the dimensions product, day and city, in other words, by “quantity sold”, we mean the quantity sold for a particular product at a particular date in a particular city. Note that a dimension is represented by its identifier (e.g. the values of the dimension product are product codes).

Sometimes, we need to represent an event linking several dimensions without having any measure associated with this event [KIM 96]. For this purpose, we use a specific type of measure, called dummy measure. Consider for example the relationship “reservation” linking a borrower, a book and a reservation date, without any specific attributes characterizing the reservation. The dummy measure “reservation” will serve to indicate which books have been reserved by which borrowers and at which date. Note that dummy measures need to be distinguished
from other measures, since they will require specific implementation depending on the target ROLAP or MOLAP tool.

The dimensions are organized in **hierarchies**. A hierarchy is an aggregation path between dimensions. In Figure 1, “city->region” and “day->month->quarter->year” are examples of hierarchies. A hierarchy is oriented from the lower to the upper abstraction levels (here, from city to region and from day to year). The arrow between two successive dimensions (called **dimension link**) may be interpreted as a functional dependency. Hierarchies are of paramount importance in the multidimensional metamodel since they are used to aggregate (“rollup”) or detail (“drill-down”) measures.

Dimensions may be described by **attributes**. For example, the dimension product (i.e. product code) is described by the product name and its unit price. Attributes are not the object of multidimensional analysis, as opposed to measures. In other words, if a dimension is described by a feature that is a measure of interest, this feature should be defined as a one-dimensional measure associated with this dimension.

A multidimensional model is specified textually as illustrated for the multidimensional model of Figure 1:

```
dimension day
dimension month
dimension quarter
dimension year
dimension product code
dimension category
dimension city
dimension region

measure quantity sold [product code, day, city]
hierarchy time day->month->quarter->year
hierarchy type product code->category
hierarchy location city->region
attribute product name [product code]
attribute unit price [product code]
```

Our multidimensional metamodel unifies the concepts of the main multidimensional metamodels found in the literature. The metamodel is generic in that it can be mapped into many OLAP tools, as the present article will illustrate for the ROLAP star case. For the sake of this article, we present a simplified version of the metamodel. A more complete version can be found in [AKO 01].

Figure 2 summarizes the main concepts of our multidimensional metamodel, represented with UML. The link between a measure and a dimension dimensioning this measure is called **dimensioning** and is strong if the dimension is necessary to functionally determine the measure (if the dimensioning is not strong, the dimension shall be indicated between parentheses in the textual specification of the measure). Every dimensioning has at least one **aggregate function** associated with it. The name of an aggregate function is either Sum, Avg, Min, Max or Count. Aggregate functions are characterized by a restriction level, as suggested by [HÜS 00]. A Sum aggregate function is less restrictive than a Count aggregate function in that it allows more analysis.
The unified multidimensional metamodel is used as a pivot metamodel in our design method, as described in the next section.

3. The design method

Starting from user requirements, our method is based on the three usual abstraction levels: conceptual, logical and physical (Figure 3). It is therefore decomposed into four phases:

- In the conceptual phase, the designer represents the universe of discourse using the UML notation [OMG 01a] along with the associated approach of development [JAC 99] (step 1); the UML model is then enriched and transformed to take into account the specific features of multidimensional modeling (step 2).

- In the logical phase, the enriched and transformed UML model is mapped into a unified multidimensional model, using mapping rules.

- The physical phase allows the designer to map the multidimensional model into a physical database schema, depending on the target OLAP tool. Due to space limitations, in this paper, we focus on ROLAP tools using the star model. A specific set of mapping rules from the logical to the physical model is defined for each type of tool.

![Figure 2. Unified multidimensional metamodel](image-url)
The data confrontation phase consists in mapping the physical schema data elements with the data sources. It leads to the definition of queries for extracting the data corresponding to each component in the physical schema. This is a very complex problem, going beyond the scope of this paper. However, it is important to mention this crucial phase in the data warehousing process.

Figure 3. The four steps of the design method

3.1. Conceptual design

OLAP systems are emerging as the dominant approach in data warehousing. OLAP allows designers to model data in a multidimensional way as hypercubes. ROLAP snowflakes and stars as well as MOLAP cubes do not offer a visualization
of data structures independently from implementation issues. Therefore, they do not ensure a sound data warehouse conceptual design.

Our design method uses the Unified Modeling Language (UML) at the conceptual level. The reason for this is threefold: UML is now a well-known language for software engineers; it provides simple yet powerful constructs to describe at a high level of abstraction the important concepts of the application domain. Finally, it can be easily mapped to relational as well as to multidimensional logical models.

Due to these considerations, many authors use the UML notation at a first step of transactional database design. It is now natural to apply UML to data warehousing as well, as illustrated by the Object Management Group’s recent Common Warehouse Metamodel [OMG 01b].

Our design method consists of a two-step conceptual design:

– Step 1 leads to a UML model, more precisely to a class diagram without operations.

– Step 2 enriches and transforms this model to facilitate its automatic mapping to a unified multidimensional model. Four types of operations are conducted: the determination of identifying attributes, the determination of measures, the migration of association attributes and the transformation of generalizations.

For the second step, we need to specialize and enrich OMG’s UML metamodel with concepts specific to multidimensional modeling (Figure 4).

![Figure 4. Enriched UML metamodel](image-url)
In Figure 4, classes which are not association classes are called ordinary classes.
Similarly, associations which are not association classes are called ordinary associations. The main extensions are the attribute measure of class Attribute (which indicates if the attribute corresponds to a measure), and the attribute identifyingAttribute, which indicates if an attribute identifies its owner class. The attribute identifyingAttribute is only necessary for the attributes of ordinary classes.

The four transformations performed during step 2 are presented below and formalized with OCL [OMG 01a]. A transformation is represented in OCL as an operation of the UML model considered. The formal specification of the transformation consists in pre- and post-conditions on the operation. Due to space limitations, some of the transformations are expressed in natural language only.

3.1.1. Determination of identifying attributes

Since the notion of identifying attribute is not defined in the standard UML notation, we need to determine explicitly the identifying attributes of classes in order to define the dimensions of the multidimensional model at the logical level. Note that since an association class is identified by the n-uple of identifying attributes of the participating classes, the determination of identifying attributes is necessary only for the ordinary classes. For each ordinary class of the UML model, the user and the data warehouse designer have to decide which attribute identifies the class (for simplicity, we assume that only one such attribute may exist for each ordinary class). If necessary, a specific attribute is created in order to identify the class. Identifying attributes are specified using the UML construct of tagged value: the suffix \{id\} is added to each identifying attribute, as in [MOR 00]. This process can be synthesized by the following transformation (expressed in natural language and OCL):

\[\text{Transformation Tcc1: Each attribute of an ordinary class is either an identifying attribute or not.}\]

\[
\text{context UMLModel::Tcc1(ordinaryClass: OrdinaryClass)}
\]
\[
\text{post: (ordinaryClass.attribute ->size() = ordinaryClass@pre.attribute ->size()) or (ordinaryClass.attribute ->size() = ordinaryClass@pre.attribute ->size()+1)}
\]
\[
\text{post: (ordinaryClass.attribute ->forall(a1: Attribute | (a1.identifyingAttribute = true) or (a1.identifyingAttribute = false)))}
\]
\[
\text{post: (ordinaryClass.attribute ->select(a1: Attribute | a1.identifyingAttribute = true) ->size = 1)}
\]
3.1.2. Determination of attributes representing measures

We differentiate between attributes representing measures, and attributes which can be defined as qualitative values. As described in the previous section, this distinction is not based on data types even if, generally, measures are numerical and qualitative attributes are not. Therefore this differentiation cannot be performed automatically. The user and the data warehouse designer have to decide which attributes must be considered as measures. Note that this does not deal with identifying attributes determined previously, since an identifying attribute cannot be a measure. In the UML model, attributes representing measures are specified by the tagged value \{meas\}. This process can be synthesized as follows:

Transformation Tcc2: Each attribute is either a measure or not.

3.1.3. Migration of association attributes

This step is concerned with 1-1 and 1-N associations having specific attributes (these associations are actually association classes, since an ordinary association cannot bear attributes in UML). Let us mention that this case is rarely encountered. If specific attributes are present in these associations, the designer has first to check the validity of this representation. Even if their presence cannot be questioned, they cannot be mapped into multidimensional models by using hierarchies. The reason is that, in multidimensional models, these hierarchies do not contain information. Therefore, they must migrate from the association to the participating class on the N side. In case of 1-1 association, they can indifferently migrate into one of the two classes. The transformations for migrating association attributes are expressed as follows:

Transformation Tcc3: Each attribute belonging to a 1-1 association is transferred to one of the classes involved in the association.

Context UMLModel::Tcc3(associationClass: AssociationClass)

pre: (associationClass.connection->size()=2) and
    ((associationClass.connection->at(1)).multiplicity='zero or one'
    or
    (associationClass.connection->at(1)).multiplicity='exactly one') and
    ((associationClass.connection->at(2)).multiplicity='zero or one'
    or
    (associationClass.connection->at(2)).multiplicity='exactly one') and
    associationClass.attribute->isNotEmpty()
post: associationClass@pre.attribute->
  forall(a1:Attribute| associationClass.attribute->excludes(a1) and
  (associationClass.connection
   ->at(1)).participant.attribute
   ->includes(a1))
or
  associationClass@pre.attribute->
  forall(a1:Attribute| associationClass.attribute->excludes(a1) and
  (associationClass.connection
   ->at(2)).participant.attribute
   ->includes(a1))

Transformation Tcc4: Each attribute belonging to a 1-N association is transferred to the N-class, i.e. the class involved several times in the association.

3.1.4. Transformation of generalizations

The generalizations of the UML notation cannot be mapped directly to hierarchies in the multidimensional model, since the semantics of hierarchies in object-oriented models and multidimensional models differ. However, we want to preserve the information contained in UML generalizations and transform these hierarchies to enable their correct mapping to multidimensional hierarchies in the logical phase. To this end, we transform the generalizations into aggregations and classes following the proposal of [MOO 00] for ER models. We have adapted this rule to UML and extended it to consider the different cases of incomplete specialization and/or overlapping specialization. The corresponding transformation is informally described below:

Transformation Tcc5: For each level i of specialization of a class C, a class named Type-C-i is created. The occurrences of these classes define all the specializations of C. In case of overlapping between specializations, a special value is created for each overlapping between two or more sub-classes of C. In case of incomplete specialization, the special value “others” is created. A N-1 aggregation is created between the classes C and Type-C-i.

This transformation is illustrated with the case study in section 4.1. Note that the mapping of UML generalizations into multidimensional models is not trivial. This issue is dealt with in more depth in [AKO 01].

Thanks to the four transformations described above, the resulting UML model can then be automatically mapped into a logical multidimensional model, as described in the following section.
3.2. Logical design

The aim of the logical design phase is to map the enriched UML conceptual model into a logical one expressed with the concepts of our unified multidimensional metamodel. This model is generated using specific mapping transformations. The transformations map first the ordinary classes and their attributes (transformations Tcl1 to Tcl3) and then the associations – association classes or ordinary associations – and their attributes (transformations Tcl4 to Tcl6). A transformation is represented in OCL as an operation of the UML model, whose parameters are the target multidimensional model and the concepts mapped in the UML model; the result consists in the concepts obtained in the target multidimensional model.

The ordinary classes of the conceptual UML model are mapped by transformation Tcl1:

Transformation Tcl1: The identifying attribute of each ordinary class is mapped into a dimension in the multidimensional model.

Context UMLModel::Tcl1(identifier:Attribute, multidimensionalModel:MultidimensionalModel):Dimension
pre: identifier.owner.oclIsTypeOf(OrdinaryClass)=true and identifier.identifyingAttribute=true
post: result.name=identifier.name
post: multidimensionalModel->includes(result)

The attributes of ordinary classes are mapped by transformations Tcl2 and Tcl3:

Transformation Tcl2: The non-identifying attributes of each ordinary class are mapped into dimension attributes in the multidimensional model if these non-identifying attributes are not measures of interest.

Context UMLModel::Tcl3(nonIdentifier:Attribute, multidimensionalModel:MultidimensionalModel):Measure
pre: nonIdentifier.owner.oclIsTypeOf(OrdinaryClass)=true and nonIdentifier.identifyingAttribute=false and nonIdentifier.measure=true
post: result.name=nonIdentifier.name

The multidimensional attributes are associated with the dimension obtained by mapping the identifying attribute of the ordinary class (transformation Tcl1).

Transformation Tcl3: The non-identifying attributes of each ordinary class are mapped into measures in the multidimensional model if these non-identifying attributes are measures of interest.

Context UMLModel::Tcl3(nonIdentifier:Attribute, multidimensionalModel:MultidimensionalModel):Measure
pre: nonIdentifier.owner.oclIsTypeOf(OrdinaryClass)=true and nonIdentifier.identifyingAttribute=false and nonIdentifier.measure=true
post: result.name=nonIdentifier.name
The measures are associated with the dimension obtained by mapping the identifying attribute of the ordinary class (transformation Tcl1).

The attributes of association classes are mapped using transformation Tcl4:

**Transformation Tcl4:** The attributes of each association class are mapped into measures, associated with dimensions obtained by mapping the identifying attributes of the ordinary classes directly or indirectly participating in the association class (transformation Tcl1).

If the association class bearing the attributes has one (or several) participating class(es) with a maximal cardinality of 1, the dimension(s) obtained by mapping the identifying attribute(s) of this (these) class(es) should be indicated between parentheses in the specification of the measures, to express the fact that the dimension(s) are not necessary to functionally determine the measures.

Finally, associations are mapped using transformations Tcl5 (for binary N-1 associations) and Tcl6 (for other associations):

**Transformation Tcl5:** A path formed by N-1 associations is mapped into a hierarchy in the multidimensional model.

This is a simple transformation. However, to improve the definition of multidimensional hierarchies, the transformation could be refined, e.g. by considering the different kinds of UML associations.

**Transformation Tcl6:** Every N-M or N-ary association without at least one attribute that is always defined is mapped into a dummy measure, associated with dimensions obtained by mapping the identifying attributes of the ordinary classes directly or indirectly participating in the association (transformation Tcl1).

Note that if an N-M association or an N-ary association has attributes, these attributes have already been mapped into measures (transformation Tcl4). If one of these attributes is always defined, the corresponding measure is also always defined, making the definition of a “dummy measure” unnecessary.

At the end of the logical design phase, the universe of discourse is described through a unified multidimensional model. Depending on the OLAP tool to be used, this model must be implemented, i.e. mapped into physical concepts.
3.3. Physical design

The physical design phase depends heavily on the target system. ROLAP systems implement the multidimensional model in a relational database system (RDBMS). This category of systems may be subdivided depending on the model used to implement the multidimensional model in the RDBMS. The models used are typically the star model, the snowflake model, or any combination or extension of these two models.

For each type of target system, a specific set of mapping transformations from the logical multidimensional model has to be defined. In this paper, we focus on the ROLAP star metamodel. This metamodel is represented in Figure 5. The metamodel is an extension of OMG’s relational metamodel [OMG 01b]. The main extension consists in the distinction between dimension tables and fact tables. Foreign keys are defined for fact tables only.

![Figure 5. ROLAP star metamodel](image)

The transformations described below map a logical multidimensional model into a ROLAP star schema. Similarly to the previous phases, OCL can be used to formalise these transformations.

The dimensions of the logical multidimensional model are mapped by transformation Tls1:
Transformation Tls1: Every dimension dimensioning at least one measure is mapped into a dimension table and an associated primary key.

Note that the logical dimensions that do not dimension any measure, i.e. the ones that only participate in hierarchies, will be taken care of by transformation Tls4.

Measures are mapped using transformations Tls2 and Tls3:

Transformation Tls2: Every non-dummy measure is mapped into a fact table column (i.e. a fact) in table T, whose foreign keys correspond to the logical dimensions of the measure and whose primary key corresponds to the subset of these dimensions which are not indicated in parentheses. If table T does not exist, it is defined when mapping the measure.

Note that the logical dimensions indicated in parentheses in the specification of a measure are the ones that are functionally determined by the others. Therefore, they are not used in the definition of the primary key of table T.

Transformation Tls3: Every dummy measure is mapped into a fact table whose foreign keys correspond to the logical dimensions of the measure and whose primary key corresponds to the subset of these dimensions which are not indicated in parentheses.

The fact tables generated by transformation Tls3 are thus factless fact tables [KIM 96].

The hierarchies of the logical model are mapped using transformation Tls4:

Transformation Tls4: Every hierarchy $D_{1} \rightarrow D_{2} \rightarrow \ldots \rightarrow D_{n}$ of the logical model is mapped by considering all the sub-hierarchies $D_{j} \rightarrow D_{j+1} \rightarrow \ldots \rightarrow D_{n}$ where $1 \leq j \leq n$ and $D_{j}$ dimensions at least one measure. A sub-hierarchy $D_{j} \rightarrow D_{j+1} \rightarrow \ldots \rightarrow D_{n}$ is mapped in the physical model by defining in the dimension table identified by $D_{j}$ a column corresponding to each of the $D_{i}$ (where $j < i \leq n$).

**Context** MultidimensionalModel::

Tls4(hierarchy:DimensionHierarchy, starSchema:StarSchema):Set(Column)

post:Sequence{1..(hierarchy.dimensionLink->size())} ->forall(J:Integer|
  if (hierarchy.dimensionLink->at(J)).source.measure ->isNotEmpty()
  then (Sequence(J..(hierarchy.dimensionLink
  ->size())))
  ->forall(I:Integer|
  starSchema ->forall(dimensionTable:DimensionTable|
    if dimensionTable.primaryKey.column.name=(hierarchy.dimensionLink
    ->at(J)).source.name
then (dimensionTable->select(cl:Column|
c1.name=(hierarchy.dimensionLink
->at(I)).source.name)->size()=1)
result->includes((dimensionTable
->select(cl:Column|
c1.name=(hierarchy.dimensionLink
->at(I)).source.name))

endif)

post:starSchema->includes(result)

Dimension attributes are mapped with transformation Tls5:

Transformation Tls5: Every attribute of every dimension Di of the logical model is mapped into a dimension table column, in all the dimension tables which possess an (identifying or non-identifying) column corresponding to Di.

The next section illustrates our design method through an example, phase after phase.

4. Case study

A firm is faced with the definition of an optimal media-planning system. The company wishes to launch advertising campaigns for its products using several types of media (radio, TV, newspapers, magazines, etc.). Its objective is to maximize the number of consumers being exposed to the advertising campaign. To support the decision-making process, we need to define a multidimensional model with all the data relevant to the media-planning problem.

4.1. Conceptual design

The conceptual model is represented in Figure 6. It contains data related to the products concerned by the advertising campaigns. The consumers are represented as targets located at different regions. The consumers are defined according to their purchasing behavior over time which is strongly influenced by the advertising campaigns. The consumers are exposed to several types of media. This exposure is measured over time. The model includes all the key information about the media shareholding. The real model has been simplified in order to be more readable.
The UML conceptual model for the media-planning example

The UML model is then enriched by determination of identifying attributes \{id\}, attributes representing measures \{meas\}, migration of association attributes and transformation of generalizations. Two classes Type-shareholder-1 and Type-shareholder-2 are created and renamed as Shareholder_type and Private_shareholder_type. The set of occurrences of Shareholder_type is \{private, public, both\}. The set of occurrences of Private_shareholder_type is \{person, company, others\}. The attribute percentage_of_region is transferred to the class Target (transformation Tcc4). The result is represented in Figure 7.
4.2. Logical design

We give below the unified multidimensional representation resulting from the application of the mapping transformations (Table 1).

Figure 7. Enriched/transformed UML model
### Table 1. Media-planning multidimensional model

<table>
<thead>
<tr>
<th>dimension</th>
<th>hierarchy</th>
<th>attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>product_code</td>
<td>campaign_product</td>
<td>product_unit [product_type]</td>
</tr>
<tr>
<td>product_type</td>
<td>campaign_product</td>
<td>product_name [product_code]</td>
</tr>
<tr>
<td>campaign_code</td>
<td>campaign_product</td>
<td>status [target_code]</td>
</tr>
<tr>
<td>dd_mm_yy</td>
<td>campaign_product</td>
<td>minimum_age [target_code]</td>
</tr>
<tr>
<td>quarter</td>
<td>campaign_product</td>
<td>maximum_age [target_code]</td>
</tr>
<tr>
<td>year</td>
<td>campaign_product</td>
<td>sex [target_code]</td>
</tr>
<tr>
<td>shareholder_name</td>
<td>shareholder_type</td>
<td>insertion [media_type]</td>
</tr>
<tr>
<td>shareholder_type</td>
<td>shareholder_type</td>
<td>advertising_price [media_name]</td>
</tr>
<tr>
<td>region</td>
<td>shareholder_type</td>
<td>public_shareholder_level [media_name]</td>
</tr>
<tr>
<td>target_code</td>
<td>location target_code-&gt;region</td>
<td>manager_name [shareholder_name]</td>
</tr>
<tr>
<td>region</td>
<td>location target_code-&gt;region</td>
<td>public_shareholder_level [media_name]</td>
</tr>
<tr>
<td>media_name</td>
<td>location target_code-&gt;region</td>
<td>manager_name [shareholder_name]</td>
</tr>
<tr>
<td>media_type</td>
<td>location target_code-&gt;region</td>
<td>manager_name [shareholder_name]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>measure</th>
<th>dummy measure in</th>
<th>attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>percentage_of_region [target_code]</td>
<td>campaign_code-&gt;product_code-&gt;product_type</td>
<td></td>
</tr>
<tr>
<td>number_of_inhabitants [region]</td>
<td>campaign_code-&gt;quarter-&gt;year</td>
<td></td>
</tr>
<tr>
<td>product_consumption [product_code,target_code,quarter]</td>
<td>campaign_code-&gt;product_code-&gt;product_type</td>
<td></td>
</tr>
<tr>
<td>strongly_influenced_by [product_code,target_code,campaign_code]</td>
<td>campaign_code-&gt;product_code-&gt;product_type</td>
<td></td>
</tr>
<tr>
<td>may_be_advertised_in [campaign_code,media_name]</td>
<td>campaign_code-&gt;product_code-&gt;product_type</td>
<td></td>
</tr>
<tr>
<td>gets [region,media_name]</td>
<td>main_shareholder [shareholder_name]</td>
<td></td>
</tr>
<tr>
<td>gets [region,media_name]</td>
<td>media_exposure [media_name,target_code,quarter]</td>
<td></td>
</tr>
</tbody>
</table>

#### 4.3. Physical design

Figure 8 presents the result of the application of the transformations described in Section 3.3, assuming that ROLAP star is used for implementation. Note that since many dimension tables are shared by different fact tables, Figure 8 actually represents a constellation of stars.
Figure 8. Media-planning ROLAP star physical schema
After the definition of the physical database schema, the data confrontation phase is performed in order to map the physical schema data elements with the data sources. Since it is beyond the scope of this paper, this confrontation phase is not illustrated here.

5. State-of-the-Art

The deficit in data warehouse design is real. Very few methods have been proposed until now. Let us mention [AKO 97 ; AKO 01 ; GOL 98a ; CAB 98 ; MOO 00].

Unlike papers describing design methods, as stated by [SAP 98], a fair number of publications is available concerning multidimensional data modeling but with very few recognizing the importance of the separation of conceptual, logical and physical issues. Moreover, even if the three levels are considered, some confusion exists between, on the one hand, conceptual and logical models and, on the other hand, logical and physical models. A real confusion also seems to exist between the conceptual and physical aspects. As an example, the multidimensional modeling manifesto of Kimball is inadequate for conceptual modeling [KIM 97]. His approach tends to include physical design issues, especially with his propositions of star and snowflake schemas which appear not to be independent from implementation issues.

5.1. Conceptual-logical models

[CAB 98] focus on logical design issues and propose a logical model for OLAP systems. They assume that there exists an integrated ER schema of the operational data sources. They provided a methodology to transform the ER schema into a dimensional graph. [GOL 98b] proposed a conceptual model called Dimensional-Fact schema. They provided a methodology to transform the ER model of the data sources into a Dimensional-Fact model. Note that their approach is not based on a formal data model. [LEH 98b] proposed a conceptual multidimensional model which includes some mechanism to structure qualifying information. Note that no formal graphical notation is provided. [SAP 98] presented a specialization of the ER model called Multidimensional Entity-Relationship Model (MER) expressing the multidimensional structure of the data by means of two specialized relationship sets and a specialized entity set. Their approach models user requirements independently from the structure of the data sources. [GYS 97] proposed a multidimensional database model providing the functionalities necessary for OLAP-based applications. They made a clear separation between the structural aspects and the contents, allowing them to define data manipulation languages in a transparent way. They defined an algebra and a calculus.
5.2. Logical-physical models

A survey of logical models for OLAP databases can be found in [VAS 99]. The main features of these models are that they systematically offer a logical view of data to be queried by a set of operators and, usually, a set of implementation mechanisms. Among these models, let us mention [AGR 97] who provide a logical model in which dimensions and measures are treated in a symmetric way and where multiple hierarchies of dimensions allow ad hoc aggregates. [PED 99] propose a multidimensional logical data model for complex data, justifying nine requirements to be satisfied in order to support complex data. They showed that their model covers the nine requirements in a better way than previous models [RAF 90; AGR 97; GRA 96; KIM 96; LIW 96; GYS 97; DAT 97; LEH 98a]. [HAR 96] investigated mainly physical design and implementation issues.

Finally, our method is an attempt to define a generic framework based on the following principles:

- it makes a clear distinction between the classical steps of database design (conceptual, logical, physical),
- it unifies the different multidimensional concepts into a single and generic model,
- it can be used for implementation with ROLAP as well as MOLAP tools, although this article stressed the frequent case of ROLAP star
- it capitalizes on the existing UML schemas,
- it is based on well-established UML concepts.

6. Conclusion and further research

We have described a method for designing and developing data warehouses. Capitalizing on database design techniques, we proposed a conceptual design phase based on the UML notation, followed by an enrichment/transformation of this model. This enrichment/transformation allows the designer to automatically convert this conceptual representation into a logical multidimensional model. At this step, we proposed a generic multidimensional metamodel independent from implementation issues and unifying ROLAP and MOLAP concepts. Using mapping rules, this generic logical model can be mapped to any physical multidimensional platform. A case study was described to illustrate the main features of the method.

Similarly to software engineering which has come to maturity with the advent of CASE tools, the data warehousing community needs extensive tool support to achieve significant productivity improvements. In this respect, our method, which can be semi-automated, is a contribution. A first prototype has been developed [BAR 02] and the results we got so far confirm the relevance an usefulness of our
approach. We plan to extend the tool and use it to further test our approach on more extensive, real life case studies.

Further questions still remain open. The different mapping transformation sets have to be extended. A reverse engineering approach taking into account existing data warehouses and/or data marts must be developed [AKO 99]. We are currently working on these issues.

Acknowledgments

This paper has benefited from the helpful advice and comments of Isabelle COMYN-WATTIAU. Her contribution is gratefully acknowledged. The authors also thank the RESEARCH CENTER of ESSEC for its financial support.

7. References


