A Layered Multidimensional Model of Complex Objects

Doulkifli Boukraâ[1], Omar Boussaïd[2],
Fadila Bentayeb[2], and Djamel-Eddine Zegour[3]

[1] LAMEL Laboratory, University of Jijel, Po Box 98 Ouled Aissa, Jijel, Algeria
d_boukraa@esi.dz
[2] Lumière University - Lyon 2, 5 avenue Pierre Mendès-France, 69676 Bron Cedex
{omar.boussaid,fadila.bentayeb}@univ-lyon2.fr
[3] Ecole Nationale Supérieure d'Informatique, (ESI), Oued-Smar, Algeria
d_zegour@esi.dz

Abstract. Multidimensional modeling is nowadays recognized to best reflect the decision makers' analytical view on data. In this paper, we address some modeling features that we believe existing multidimensional models do not fully cover, such as considering real life entities that are meant to be analyzed as complex objects, allowing for simple and complex measures, treating facts and dimension members equally and observing hierarchies within and between complex entities. We propose a layered multidimensional model based on the concept of *complex object* which encapsulates data and structure complexity and eases the creation and manipulation of complex data cubes. We need to define our model at three layers. The first layer *class diagram* describes complex objects and captures the hierarchical organization of their attributes. The second layer *package of classes* describes the multidimensional model as a set of complex objects that are connected by relationships and some of which are organized in hierarchies. The third layer *package of packages* describes complex cubes which are derived from the multidimensional model. We show the benefits and feasibility of our proposals through their implementation in a real-life case study.

Keywords: complex object, complex relationship, hierarchy, multidimensional model, cube, projection, layer.

1 Introduction

Multidimensional modeling is nowadays widely adopted for decision support as it dedicates data organization to meet the analysts' needs. A traditional multidimensional model (MDM) organizes data around one or more facts, described by a set of measures, which are analyzed along analysis axes [6]. The star schema of Kimball is one well-known and commonly referenced multidimensional model [7]. However, Kimball's schema is targeted to a relational implementation within a ROLAP system and thus, can be viewed as a relational-dedicated logical schema.

C. Salinesi, M.C. Norrie, and O. Pastor (Eds.): CAiSE 2013, LNCS 7908, pp. 498–513, 2013.

Thus, in recent years, there has been an increasing amount of literature on multidimensional modeling at the conceptual level. These models have been reviewed by [4], [1], [13]. Following a date of publication order, in [4], the authors set a list of modeling requirements, such as complex-structured dimensions and complex measures, then compared some existing models based on those requirements. The review showed that they are partially covered in each model. A more comprehensive classification of models was proposed by Abello et al. [1]. The authors defined a framework composed of two orthogonal axes: (1) whether the model is conceptual, logical, physical or formal and (2) whether it represents the multidimensional concepts of *fact*, *dimension* and *relationships* between dimension members. In [13], the authors motivated the need for modeling data warehouses at the conceptual level. The authors then presented a set of basic and advanced features that a conceptual multidimensional model should fulfil and they proposed a formal model based on two concepts *dimension* and *datacube* (fact). The reviewed models in [4], [1], [13] made the first step of bringing the multidimensional modeling to the conceptual level. However, although important, these models remain suitable for modeling structured data relative to both dimensions and facts. Therefore, other models based on the object paradigm have emerged to handle more data structures. Among these models, we focus in the related work on the object-oriented multidimensional models then we motivate the need for a new object-oriented multidimensional model. The remainder of this paper is organized as follows. In section 5, we outline the general principle of our multidimensional approach. Then, in section 6, we present our MDM. In section 7, we present the cubic projection operator and the resulting cube model. We provide implementation details in section 8. Finally, we conclude and give future work directions.

2 Related Work

Trujillo et al. provided one of the earliest multidimensional models based on the object paradigm [15]. They motivated the use of the object orientation by the need to capture not only the data within a multidimensional model but the operations that manipulate the data as well. A dimension is modeled as an object class; a fact class is represented in a similar way to a dimension class but the set of non-key attributes of the fact are qualified as *measure* and some fact attributes may be partially ordered, thus defining a hierarchy. In [14], Trujillo proposed the *Gold* model where he considered other multidimensional aspects such as *derived measures*, *derived dimension attributes* and *fact additivity*. In 2000, Trujillo et al. proposed a UML-based multidimensional model [16]. Lujan-Mora enhanced the expressiveness of the previously described multidimensional models by using eight stereotypes that relate to different multidimensional concepts at different levels [8]. In addition, the possibility of graphically represent the stereotyped elements increased the intelligibility of the model by the analysts. Further, in [9], the authors made use of UML packages to bring their model to a more understandable level.

The work of Abello et al. dealt with multidimensional modeling using the object paradigm. In [1], the authors advocated the benefits of the object orientation with respect to multidimensional modeling and discussed six object oriented aspects, namely *Classification/Instantiation*, *Generalization/Specialization*, *Aggregation/Decomposition*, *Derivability*, *Caller/Called* and *Dynamicity*. Later, the authors further detailed the problems underlying multidimensional modeling of dimensions [2] and showed how the object paradigm can help solve these problems. In [3], Abello et al. proposed a multidimensional model called YAM^2 that extends UML to multidimensionality. In their model, they considered different levels of modeling, to obtain a schema that is easily understandable by analysts. The work of Pedersen and Jensen et al. [11] dealt with multidimensional modeling of complex data.The model proposed by the authors is composed of a fact, of multiple dimensions and of as many relationships as the dimensions. A dimension is modelled as a set of category types forming a lattice, on which an order is defined to represent hierarchies. The model supports correct aggregation by associating aggregation functions to relevant categories. In [10], Nassis et al. were motivated by warehousing XML data in order to exploit huge amounts of XML data for analysis purposes. The authors built on the idea that the XML schemata that represent the XML data sources are not suitable for modeling the XML document warehouses and therefore, they proposed to conceptually design the warehouse model using the object paradigm and in particular by means of UML class and package diagrams.

3 Discussion

The approach of Nassis et al. [10] has the advantage of representing facts with rich semantics and that are more meaningful than simple object classes. However, the authors' approach does not allow for a symmetric treatment of facts and dimensions which has been outlined as one modeling requirement for complex data by [11]. As a matter of fact, because dimensions are modelled virtually using conceptual views, they have less semantics. Thus, for a symmetric treatment of facts and dimensions, dimensions also need to be as *meaningful* as the facts and both can be viewed as *complex entities*. Therefore, we claim for modeling facts and dimensions as complex entities that can be represented by whole class diagrams, which, to the best of our knowledge has not been considered in existing models. The second feature of our approach is to use different layers to represent the multidimensional concepts in a similar way to [3] and [9]. The need for using different layers is to provide abstract levels that help understand the multidimensional model components. However, because we intend to model semantically rich facts and dimensions, the layers defined by [3], [9] are not sufficient and an additional layer is needed. Thus, at the lowest layer, we propose to represent facts and dimensions as full class diagrams which provide details about the structure of real life entities. We call such a layer *class diagram layer*. Then, at a second layer, we represent each fact and each dimension level as a package of classes of the layer below. This layer provides a macro view of the

layer below. In addition, our novelty is to treat the packages of the second layer equally in the sense that each one can play the role of fact or dimension level. The multidimensional model is neutral with respect to analysis contexts: the fact and dimensions need not be predefined in the model but designated at analysis time. The designation can be performed by means of a projection operation at the package layer, similarly to the *Focus* operator defined by Ravat et al. [12]. Then, we represent the resulting structure, called *complex cube* at a third layer using packages as well. A package at the third layer corresponds either to the fact or to a dimension. In the next section, we present a motivating example from which we derive a set of new modeling requirements. Then, we briefly show how these requirements will be answered by our multidimensional model.

4 Motivating Example

The case study is adapted from the XMark benchmark project[1]. Figure 1 pictures a UML class diagram representing auctions. An *auction* corresponds to one *item* that falls into one or many *categories* and that is being or has been *sold* by one *person*. An auction can be open or closed: an open auction may be *watched* by many people and may be subject to many *bids* whereas a closed auction has one *buyer* and it is *annotated* once. Data about auctions may be analyzed for decision support purposes. For instance, we would need to know (i) the most item categories being auctioned, (ii) the effect of distance between the item locations and people's addresses on the sales (iii) the price evolution of auctions from the initial to the final price, etc. From this case study, we have drawn up the following features that a MDM should support.

1. *Complex facts and complex dimension members.* The data model should allow facts or dimension members to have a complex structure. In our example, Auction[2] is described by one class and two sub-classes whereas Item is described by four classes, linked by eight relationships.
2. *Hierarchies within complex entities.* The data model should allow observation of hierarchies within complex entities while treating these entities as a whole. Furthermore, the hierarchy members need not be simple class attributes but may be a mixture of classes and attributes. In our example, we can observe a hierarchy in *Address* composed of *city* and *region*.
3. *Hierarchies of complex entities.* In this case, some complex entities may be organized as hierarchies. In our example, Item and Category are complex entities and they form a hierarchy.
4. *Symmetric treatment of complex entities as facts or dimension members.* Although this feature has already been recommended in related work [11], it applies here to complex entities.

Considering the above features, we propose a MDM as well as an operator to derive complex object cubes as detailed in the next section.

[1] `http://www.xml-benchmark.org/`

[2] We use capital letters to differentiate the complex entity names from the class names.

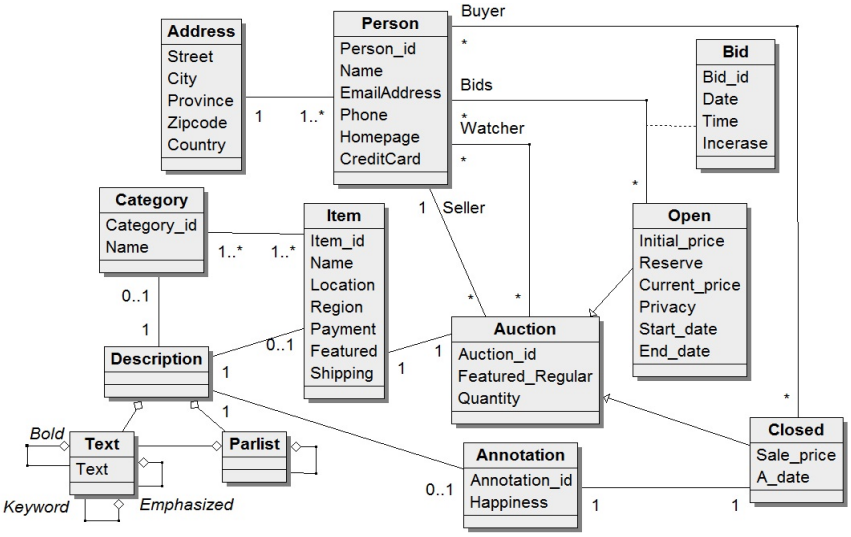

Fig. 1. Class Diagram of auction case study

5 General Principle of the Layered Multidimensional Modeling Approach

The general principle of our approach is depicted in figure 2. We present the multidimensional schema at two layers to the right side of the figure at two layers: the *package of classes* layer and the *class diagram* layer. We present the data cube schema at three layers: the *package of packages* layer, the *package of classes* layer and the c*lass diagram* layer. We show the correspondences between the layers in each side using double-arrowed lines to which we assign numbers. Let us note that the structures of the multidimensional schema and the data cube are similar at levels *class diagram* and *package of classes*. However, we have chosen to separate the presentation of the multidimensional schema from that of the data cube in order to show the roles played by the complex objects in the data cube. For the sake of clarity, we first explain the layers of the multidimensional schema then we explain the layers of the data cube.

5.1 Layers of the Multidimensional Schema

The *package of classes* layer abstracts the real world as a set of complex objects. The complex object are linked via high level relationships which we qualify as *complex relationships*. It is at this layer that we observe the hierarchical organization of some complex objects, which we qualify as *object hierarchies*. The *class diagram* layer of the multidimensional schema provides details about the classes composing each complex object (double arrow 1) and about the origin of each complex relationships, which is a relationship linking two classes of different

complex objects (double arrow 2). Finally, the *class diagram* layer is also used to design the hierarchical organization of some attributes of complex objects, which we qualify as *attribute hierarchies*.

5.2 Layers of the Data Cube

The *package of packages* layer represents the fact-dimension duality of a data cube. It allows for designing data cube constellations that share the same dimensions. The *package of classes* layer provides details of the previous layer in terms of the complex objects composing a dimension and of the complex object that plays the role of the fact (double arrow 7). The *class diagram* layer provides details about the structure of each dimension member (double arrow 3), the origin of the link between the fact and each dimension (double arrow 4) and the structure of the fact (double arrow 5). The data cube is obtained from the multidimensional schema using a projection operation, called *cubic projection*. The cubic projections consists in assigning two roles to some complex objects of the *packages of classes layer* of the multidimensional schema: one complex object is assigned the role of fact whereas the other connected complex objects are assigned the role of dimensions. We obtain the *package of classes* layer of the data cube. Moreover, the cubic projection consists in designing the measures of the data cube at the *class diagram* layer of the multidimensional schema.

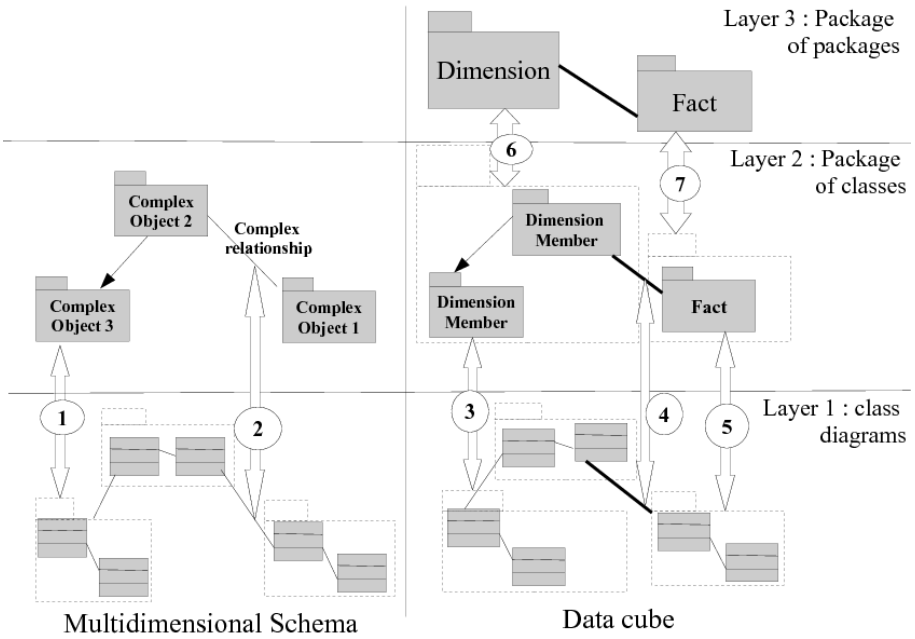

Fig. 2. General principle of our approach

6 Complex Object-Based Multidimensional Model

One of the key features of a model is its expressiveness. In this respect, the O-O paradigm provides powerful mechanisms to describe entities as objects from static and dynamic standpoints. In our work, we take advantage of the power of the object data modeling in order to model real world entities for analysis purposes. However, unlike existing O-O MDMs, we perceive the multidimensional space as a set of complex entities (CE)s that are meant to be analyzed, as a whole, or to serve as analysis axes. Multidimensionality is captured through the links between the CEs. In the context of online analytical processing (OLAP), these links are combinations of analysis axes representing the fact to be observed. Finally, we distinguish two kind of hierarchies: those defined within the complex entity structures and those that organize the complex entities themselves. In the following paragraphs, we present the four main concepts of our model.

6.1 Complex Object

A complex object (CO) is a set of object classes that form one semantic entity. A CO can be represented by a class diagram at a first modeling level or by a package at a second modeling level. Because the CO forms a whole, it has one *representative class* after which it is named. A CO is characterized by *simple attributes* (SOAs), which are the attributes of the representative class and by *complex attributes* (COAs) which are the remaining classes of the CO. Treating the classes that compose a CO as complex attributes provides is uniform way to manipulate the CO and to ease defining the concepts of our model, especially to allow observing complex measures. Figure 3 depicts the meta-model of a CO. The class *Object_Class* represents an object class of the CO. The class *Class_Attribute* represents the simple attributes of a class. The association *Class_to_attribute_link* connects an object class to its simple attributes. The association *Class_to_class_link* represents the links between classes that compose the CO, e.g. association, inheritance. The association *Attribute_to_attribute_link* represents the links between simple attributes of a CO. For instance, in the class diagram of auctions (figure 1), we can identify six COs `Person_CO`, `Auction_CO`, `Item_CO`, `Category_CO`, `Annotation_CO` and `Bid_CO` represented respectively by classes `Person`, `Auction`, `Item`, `Category`, `Annotation` and `Bid`. Figure 4(a) presents `Item_CO` at the class diagram layer. Besides, in figure 4(b), we encapsulate the complex structure of `Item_CO` at a package diagram layer. We stereotype the package as $\ll ComplexObject \gg$.

6.2 Complex Relationship

The concept of complex relationship (CR) captures the links between complex objects. These links can be observed between the representative classes of the COs or between their remaining classes. A CR is said to be complex to distinguish it from the relationships that we have defined within the COs and because it links tow complex objects at a higher level than the relationships that link

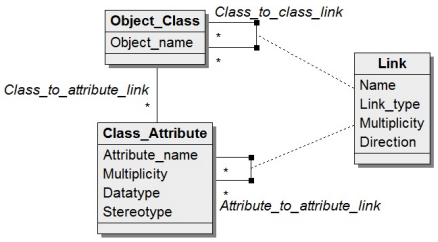

Fig. 3. Complex Object Meta-Model

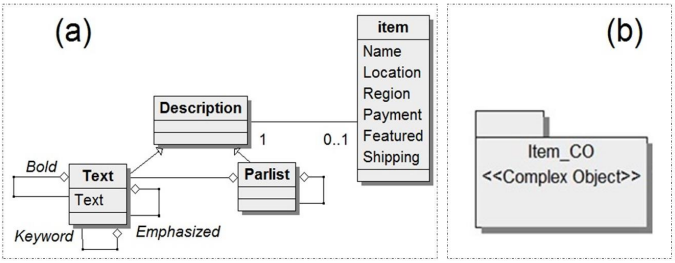

Fig. 4. Example of a Complex Object representing items

the simple classes. Added to the concept of CO, the concept of CR is the second important modeling concept that defines the multidimensionality in our model. In this respect, a CR represents a possible analysis axis along which a CO is analyzed. Besides, we define a CR at two levels: a class level and a CO level. At the class level, a CR links two classes, each one belonging to one CO. At the CO level, the CR links two COs. Let us note that in the case two COs are linked via more than one CR, each CR is represented separately. An example of a CR is shown in Figure 5(a). The CR links `Auction_CO` to `Item_CO`. The CR is modeled as a dependency between packages, stereotyped as $\ll ComplexRelationship \gg$. We use the stereotype to distinguish the CR from the other dependencies between packages, as will be seen later. We also add a stereotype attribute `name` to differentiate multiple CRs between the same pair of COs. The class diagram in figure 5(b) shows that the origin of the CR is an association between the classes `Item` and `Auction`.

6.3 Attribute Hierarchy

In section 6.1, we defined a relationship as part of the CO definition. In this section, we focus on groups of relationships that organize the simple and/or complex attributes of a CO into hierarchies. We call such an organization an Attribute Hierarchy (AH). Furthermore, because not all the attributes of a CO are members of hierarchies, an AH defines a partial order amongst the set of the

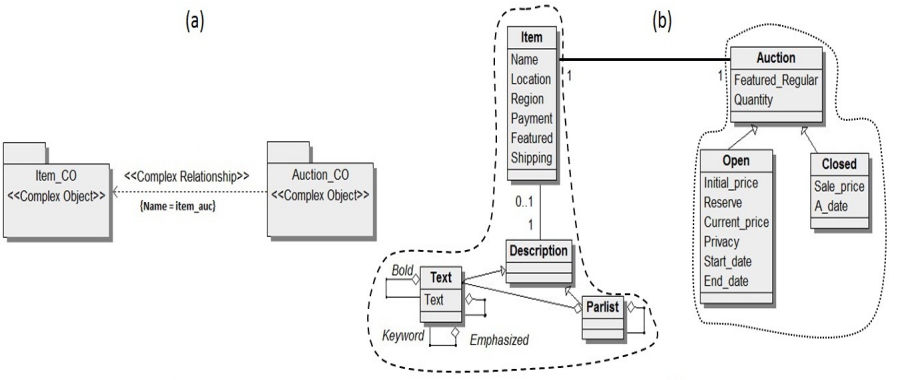

Fig. 5. Example of a relationship between complex objects

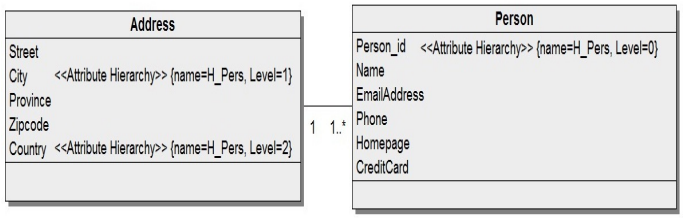

Fig. 6. Example of an Attribute Hierarchy associated to Person_ID

attributes of a CO. An AH can be represented only at the class diagram. For example, figure 6 shows an example of an AH associated with the `Person_CO`. The AH is composed of attributes `person_id`, `city`, `country` and All^A with respectively levels 0, 1, 2 and 3. We stereotype the AH by $\ll AttributeHierarchy \gg$ to differentiate the hierarchy members from the other descriptive COAs of the CO. We also use the stereotype attributes *name* and *level* to refer respectively to the hierarchy name that the attribute belongs to and to its level within the hierarchy. Doing so, we support multiple hierarchies in our model. It is worth noting that in the case a member of an AH is complex, i.e. modeled as a class, the whole class is stereotyped as $\ll AttributeHierarchy \gg$.

6.4 Object Hierarchy

An Object Hierarchy (OH) is analogous to an AH: the latter organizes the attributes of a CO whereas the former organizes the COs themselves. Moreover, similarly to an AH, an OH defines a partial order amongst the set of COs. On the other hand, since an OH organizes the COs, we define it only at the package diagram layer. For example, figure 7(a) shows an example of an OH composed of `Item_CO` and `Category_CO` at the package diagram layer. We stereotype the OH members by $\ll ObjectHierarchy \gg$ to distinguish the hierarchical organization

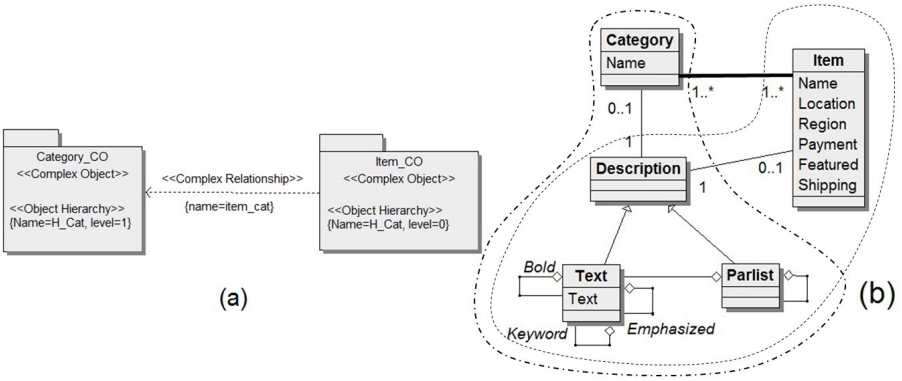

Fig. 7. Example of an Object Hierarchy

of COs from the other CRs. We also use the stereotype attributes *name* and *level* in the same way as an AH. Figure 7(b) shows the origin of the OH which is an association between the classes `Item` and `Category`.

6.5 Multidimensional Schema

Now that we have defined the four modeling concepts, we define the Complex Object-based Multidimensional Model (COMM) as composed of a set of COs that are linked by a set of CRs, where some of the COs attributes may be organized as AHs and where some COs may be organized as OHs. For example, figure 8 depicts the COMM of *auctions* at the package diagram layer. Due to space limitations, we do not present the class diagram layer, since most of the class diagrams have been detailed previously. Also, we do not present the attribute hierarchies since they appear in the class diagrams. Notice that `Item_CO`, `Category_ID` and `Annotation_CO` import classes from a common package `Description`, which in this case does not represent a CO for its own. Notice also the multiple dependencies between packages `Auction_CO` and `Person_CO` to represent different CRs. Finally, the association class `Bid` is modeled as a CO called `Bid_CO`, which gives rise to two binary CRs, one between `Auction_CO` and `Bid_CO` and the other between `Bis_CO` and `Person_CO`. We denote by *Auction_COMM* such a MDM.

7 Complex Cube Derivation

The concepts of CO, CR, AH and OH prepare the MDM for analysis: each CO is a possible fact or a dimension member, each CR is a possible analysis axis while the AHs and OHs are used to aggregate data. Thus, in order to meet specific analysis needs, we derive the analysis model from the MDM. The obtained analysis model is known as *cube* [6]. We call the derived cube *complex object cube* (COC) as it is based on complex objects and we derive it using the *cubic projection* operation. The cubic projection is a shift from the concepts of

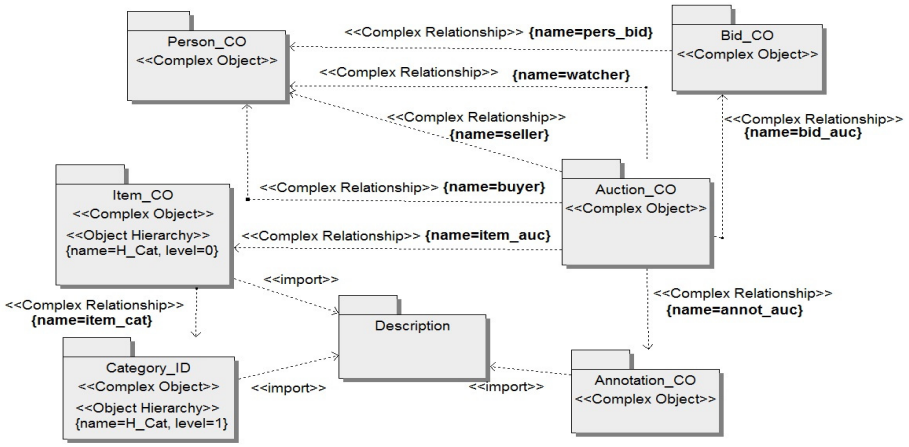

Fig. 8. Example of a COMM of auctions, (Auction_COMM)

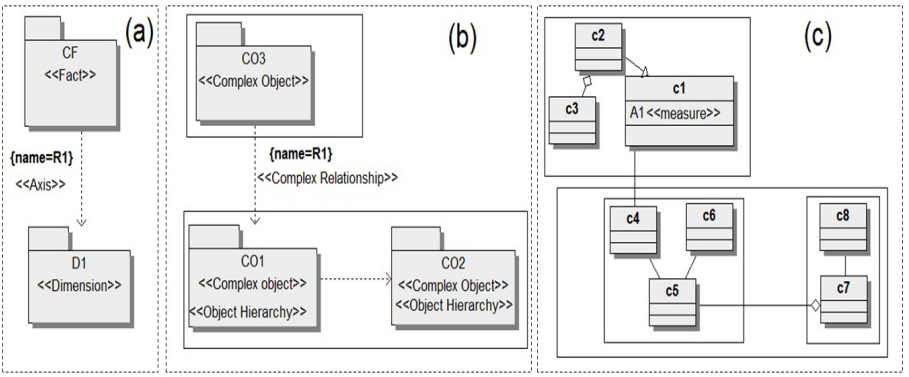

Fig. 9. Three-layer representation of a complex object cube

complex object and *complex relationship* to the concepts of *fact* and *dimension* and it is conducted at the two layers of the COMM. At the package diagram layer, the cubic projection is performed by projecting the COMM on one CO, to which we assign the role of the *fact*. The fact is said to be complex (CF) since it corresponds to a CO. A dimension is formed by one CO or by a set of hierarchically organized COs. The analysis axes of the COCs are obtained from the CRs that directly link the CF to the other COs. At the class diagram layer, we assign the role of *measure* to one attribute of the CF or more and we define a corresponding *aggregation function*. A COC can be represented by two layers as a COMM. However, since in the COC model, the fact and dimensions are explicitly named, an additional layer is needed to enhance the COC understandability. The three layers are depicted in figure 9 and described as follows.

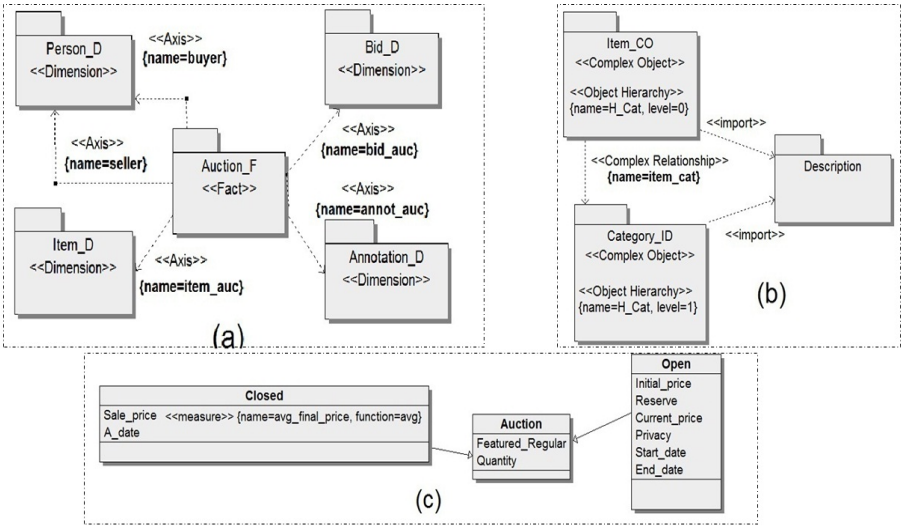

Fig. 10. Example of a complex cube of auctions

The first layer (a) is equivalent to a star schema. We represent it with a package diagram. A package corresponds either to the fact, stererotyped $\ll Fact \gg$, or to one dimension, stereotyped $\ll Dimension \gg$. The second layer (b) gives details about the content of each dimension in terms of COs as well as about the CF. This layer is similar to the diagram package layer of the COMM, but is here limited to the relevant COs and CRs of the COC. The CF is obtained by projecting the COMM on CO_3 whereas the dimension D_1 is organized into a hierarchy of CO_1 and CO_2. The third layer (c) gives details of each CO and presents its corresponding class diagram. The class diagram of a CO is the same as in the COMM class diagram layer except for the CF where we stereotype the attribute that corresponds to the measure by $\ll measure \gg$. For example, let us suppose that we want to know the average final prices of closed auctions. To carry out such an analysis need, we first project the *Auction_COMM* on `Auction_CO`, then we zoom into its class diagram. Figure 10 depicts the three layer representation of the resulting complex cube. The first layer (a) shows the star schema composed of the CF `Auction_F` and of four dimensions. In (b), we zoom into dimension `Item_D`, whereas in (c), we present the class diagram of the `Auction_CO`. The measure *avg_final_price* is associated with the attribute `Sale_price` whose values are aggregated using function `avg` .

8 Implementation

In order to validate our proposals, we have implemented *Auction_COMM* as part of a warehousing architecture. In addition, we implemented a graphical tool that is meant to design a multidimensional schema of complex object from

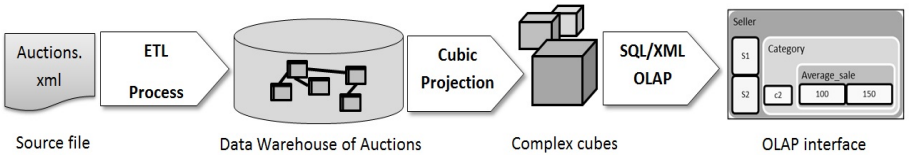

Fig. 11. Implementation of the multidimensional model and of the complex cube model

a UML-based integration schema. In what follows, we detail the warehousing architecture and the design tool.

8.1 Warehousing Architecture

The warehousing architecture is depicted in figure 11. We built an XML warehouse of auctions by transforming a single source XML document, through an Extraction, Transformation and Loading (ETL) process. The warehouse model is the same as in figure 8. At the logical level, we produced the MDM using XML Schema using a *conceptual to logical* mapping. At the physical level, we used a native storage of XML as part of Oracle 11g2 DB. Each CO and CR is modeled as a table of XML-typed objects. Each row in a table corresponds to a CO instance or a CR instance. The AHs and OHs are implemented within their corresponding COs. A complex cube is stored in a similar way to the COMM: the CF and each dimension member is stored as one object table. We implemented the cubic projection and derived a complex cube with two measures `Avg_sale_price` and `Avg_current_price`. To do that, we encoded an XML meta data file corresponding to the formal definition of the `auction_COMM`. The meta data file is input to a set of PL/SQL stored procedures which read the COMM data and populate the complex cube tables. Lets us note that we improved the quality of a complex cube by optimising the structure of the fact. Furthermore, to show the feasibility of OLAP on the complex cube, we wrote a set of queries in SQL/XML where the XML part accesses the attributes of the COs through XML paths and the SQL part wraps the XML paths and groups data along the AHs and OHs.

8.2 UML-Based Design Tool

The graphical design tool is depicted in figure 12. The design of a multidimensional schema consists in the following steps. First, the user is prompted to load a UML class diagram representing an integration schema. A complex object can be designed simply by selecting many UML classes, then by assigning a name. At this stage of our work, we provide no method to correctly package the classes in order to form a semantically correct multidimensional schema. In other words, the designer is responsible of correctly assembling a set of classes if they form a whole conceptual entity. The UML classes composing one complex objects have the same color, which helps distinguish the different complex objects from each other. In other words, colouring the original UML diagrams consists in adding

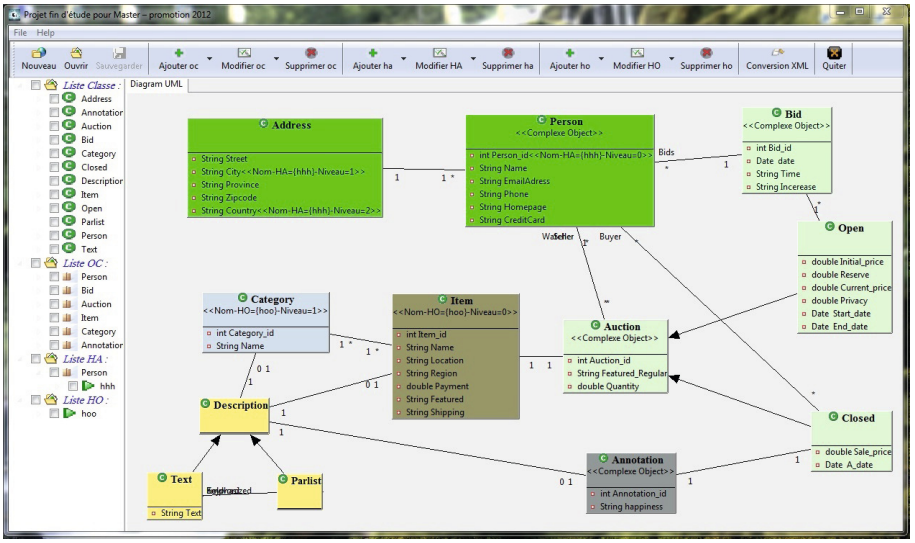

Fig. 12. Graphical Design Tool of a Multidimensional Schema

the second layer, i.e. the package diagram, to the first layer, i.e. the class diagram. Once the complex objects designed, the complex relationships are obtained automatically. Next, the designer can create an attribute hierarchy for a given complex object by selecting the hierarchy levels among the UML classes and/or the class attributes within the complex object. Similarly, an object hierarchy can be created by selecting the main class of each complex object composing the hierarchy. The design tool provides a high degree of flexibility as it allows for deleting existing complex objects and hierarchies or by modifying the hierarchies (adding, deleting, reordering levels). Besides, we implemented several controls to prevent any schema inconsistency that may occur due to deletions or changes. Besides, the design tool allows for a re-entering mode, that is by loading an existing multidimensional schema, then by performing changes as needed either to design a new schema or to complete the design of a previous one. Finally, once a multidimensional schema design is completed, the designer can translate the conceptual schema into a logical schema then into a physical schema. The conceptual/logical schema mapping implements a translation algorithm that we proposed in [5]. It generates for each complex object or complex relationship, a corresponding XML Schema schema. Similarly, the logical/physical mapping generates for each XML Schema schema a storage table meant to store the multidimensional data.

9 Conclusion

In this paper, we tackled the problem of multidimensional modeling of complex data. To this purpose, we have proposed a new model that extends existing

models by considering a whole class diagram as a fact or as a hierarchy member of a dimension. We presented our model at two layers, from which we derive a cube model at three layers. The package diagram models the universe as a set of complex objects that are linked through complex relationships and where some complex objects are organized into hierarchies. The class diagram layer provides details about the structure of each complex object, the origin of the complex relationships and models the attribute hierarchies. The two-layer modeling of the multidimensional model allows the users to design complex cubes using the cubic projection. A third layer is used as a meta model for complex cubes to explicitly represent the facts and dimensions. The main novelty or our work is that it takes full advantage of the O-O paradigm to capture the multidimensional concepts by symmetrically treating the subjects and dimension members as complex entities. Besides, the cube derivation operator allows to create different data cubes to fit different analysis contexts. For future work, we plan to cross-check the multidimensional model data and the source data to ensure a two-way validation of both the multidimensional model and the sources. We will also work on providing more analytical functions, depending on the nature of measures.

Acknowledgements. The authors would like to thank Lakhdar Belgherbi and Aimad Belilet from the University of Jijel for the co-implementation.

References

1. Abelló, A., Samos, J., Saltor, F.: A framework for the classification and description of multidimensional data models. In: Mayr, H.C., Lazanský, J., Quirchmayr, G., Vogel, P. (eds.) DEXA 2001. LNCS, vol. 2113, pp. 668–677. Springer, Heidelberg (2001)
2. Abelló, A., Samos, J., Saltor, F.: Understanding analysis dimensions in a multidimensional object-oriented model. In: Proceedings of the 3rd Intl. Workshop on Design and Management of Data Warehouses, DMDW 2001, Interlaken, Switzerland, June 4, p. 4 (2001)
3. Abelló, A., Samos, J., Saltor, F.: Yam^2: a multidimensional conceptual model extending uml. Inf. Syst. 31(6), 541–567 (2006)
4. Blaschka, M., Sapia, C., Höfling, G., Dinter, B.: Finding your way through multidimensional data models. In: DEXA Workshop, pp. 198–203 (1998)
5. Boukraâ, D., Boussaid, O., Bentayeb, F., Zegour, D.E.: Modèle multidimensionnel d'objets complexes. du modèle d'objets aux cubes d'objets complexes. Ingénierie des Systèmes d'Information 16(6), 41–65 (2011)
6. Chaudhuri, S., Dayal, U.: An overview of data warehousing and olap technology. SIGMOD Record 26(1), 65–74 (1997)
7. Kimball, R.: The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses. John Wiley (1996)
8. Pérez, M.A., Clemente, P.J.: 12th Workshop for PhD Students in Object-Oriented Systems. In: Hernández, J., Moreira, A. (eds.) ECOOP-WS 2002. LNCS, vol. 2548, pp. 44–54. Springer, Heidelberg (2002)
9. Luján-Mora, S., Trujillo, J., Song, I.-Y.: Multidimensional modeling with UML package diagrams. In: Spaccapietra, S., March, S.T., Kambayashi, Y. (eds.) ER 2002. LNCS, vol. 2503, pp. 199–213. Springer, Heidelberg (2002)

10. Nassis, V., Rajugan, R., Dillon, T.S., Rahayu, J.W.J.: Conceptual design of XML document warehouses. In: Kambayashi, Y., Mohania, M., Wöß, W. (eds.) DaWaK 2004. LNCS, vol. 3181, pp. 1–14. Springer, Heidelberg (2004)
11. Pedersen, T.B., Jensen, C.S.: Multidimensional data modeling for complex data. In: Proceedings of the 15th International Conference on Data Engineering, Sydney, Austrialia, March 23-26, pp. 336–345. IEEE Computer Society (1999)
12. Ravat, F., Teste, O., Tournier, R., Zurfluh, G.: Algebraic and graphic languages for olap manipulations. International Journal of Data Warehousing and Mining 4(1), 17–46 (2008)
13. Torlone, R.: Conceptual multidimensional models. In: Multidimensional Databases, pp. 69–90. IGI Publishing, Hershey (2003)
14. Trujillo, J.: The gold model: An oo multidimensional data model for multidimensional databases. In: Yu, H.-J., Demeyer, S. (eds.) ECOOP 1999 Workshops. LNCS, vol. 1743, pp. 24–30. Springer, Heidelberg (1999)
15. Trujillo, J., Palomar, M.: An object-oriented approach to multidimensional database conceptual modeling. In: ACM First International Workshop on Data Warehousing and OLAP (DOLAP 1998), Bethesda, Maryland, USA, pp. 16–21. ACM (1998)
16. Trujillo, J., Palomar, M.S., Gómez, J.: Applying object-oriented conceptual modeling techniques to the design of multidimensional databases and OLAP applications. In: Lu, H., Zhou, A. (eds.) WAIM 2000. LNCS, vol. 1846, pp. 83–94. Springer, Heidelberg (2000)