

# The GOLD model: An Object Oriented multidimensional data model for multidimensional databases

Juan Trujillo  
Research Group of Logic Programming and Information Systems  
Dept. of Financial Economics  
University of Alicante. E-03071. Alicante. Spain.  
E-mail: [juan.trujillo@ua.es](mailto:juan.trujillo@ua.es)

## Abstract

There has recently been an increased interest in multidimensional databases (MDB) and On-line Analytical Processing (OLAP) scenarios. OLAP systems impose different requirements than On-line Transactional Processing (OLTP) systems, and therefore, different data models and implementation methods are required for each type of system. There have been several different multidimensional data models proposed recently. However, there are certain key issues in multidimensional modeling, such as derived measures, derived dimension attributes and the additivity on fact attributes along dimensions, that are not considered by these proposals. To the best of our knowledge, our Object Oriented multidimensional data model (OOMD, renamed GOLD) is the only one based on the Object Oriented (OO) paradigm. The aim of this paper, therefore, is to introduce the GOLD model ([7], and extended in [8]), as an OO multidimensional data model in which all the above-mentioned issues are taken into consideration. Since the GOLD model is based on the OO paradigm, data functionality and behavior are easily considered.

**Keywords:** analysis/design, databases, data warehouses, conceptual modeling, OLAP, multidimensional data models

## 1 Introduction

Companies may adopt strategic decisions that could imply a competitive advantage over their competitors. In this context, the concept of Data Warehouses (DW) emerged during the nineties ([20], [18]) as an integrated data collection system for companies oriented to decision making. The analysis of this historical data (DW), however, is carried out through user final tools that are based on OLAP technology [4]. OLTP systems impose different requirements than OLAP systems, and therefore, different data models and implementation methods are required for each type of system. The Entity-Relationship (E/R) model is commonly used to represent an OLTP application at the conceptual level. This model, however, is not capable of sufficiently representing the multidimensionality of data [18] nor can it express data functionality and behavior. The multidimensional view of data (cube or hypercube) [19] is, therefore, a popular alternative conceptual model that is used to conceptualize the data in a DW. Consequently, multidimensional databases are based on multidimensional modeling.

However, there is no formal multidimensional (MD) data model that is commonly accepted for OLAP applications. Several multidimensional data models have been proposed recently, such as [1], [2], [11], [13], [16] and [17] (a comprehensive review of these is presented in [3]). Most of them, however, focus either on an OLAP query language such as [2], [13] and [17] or on a presumption about a subsequent implementation such as [13] or [17]. Furthermore, they all overlook certain key issues in multidimensional modeling, such as derived measures or the additivity on fact attributes. On the other hand, W. Lehner in [21] introduces, for first time in this field, an OO approximation for the design of MDB in the proposed Nested Multidimensional Data. This model introduces the concept of a multidimensional object to define the multidimensional cube on which a group of operations are defined, in order to permit subsequent data analysis. This approach, however, does not consider the whole database schema and, as a consequence, the additivity on fact attributes along dimensions cannot be considered. Nor are derived measures considered either.

Our GOLD model ([7] and extended in [8]) has, nevertheless, demonstrated that the application of the OO paradigm allows us to consider key issues in multidimensional modeling that are scarcely considered by other models, such as derived measures, derived dimension attributes, multiple classification hierarchies and the additivity on fact attributes along dimensions.

This paper is organized as follows: In section 2 we introduce the classical multidimensional model by means of an example. Section 3 summarizes the basic concepts of the GOLD model with the above-mentioned extensions. In section 4 we provide the latest achievements of our approach as well as a summary of the goals of the Ph.D. that is currently being carried out. In section 5 we draw some conclusions about the use of the model. Finally, the appendix at the end of the paper provides a formal specification of the basic components of the GOLD model.

## 2 The classical multidimensional model by means of an example.

In this model, a fact is an item of interest for a given enterprise, and is described through a set of attributes called measures or fact attributes, that are contained in cells or points in the data cube. Moreover, this set of measures is based on a set of dimensions that are the granularity adopted for representing facts (i.e., the context in which facts are to be analyzed). Thus, dimensions are also characterized by attributes that are called dimension attributes. Another relevant feature of the model is the classification hierarchy defined on attributes along dimensions, which determines how fact instances may be aggregated and significantly selected for the decision making process. Furthermore, multiple classification hierarchy and alternative path hierarchy are also relevant in multidimensional modeling. A final relevant feature of the multidimensional model is the concept of the additivity on fact attributes along dimensions. A fact attribute is additive along one dimension if the sum operator can be used to aggregate attribute values along all the hierarchies defined in that dimension.

In figure 1, we show the classical multidimensional cube by means of an example that will be handled throughout the paper. Specifically, the fact is the *sales of products* in a great store chain, and measures are *product\_price*, *quantity\_sold* and *total\_price* (*total\_price* is a derived measure obtained by the formula  $total\_price = product\_price * quantity\_sold$ ). Dimensions, with their attributes, are as follows: *store* with *name*, *country*, *area*, *city*, *sales\_area*, *address* and phone number, *product* with *colour*, *quantity*, *weight*, *group*, *family*, *type*, *supplier*, *brand*, *department*, *manager* and *transport\_cte* (*transport\_cte* is a derived dimension attribute obtained by the

formula  $transport\_cte = product.quantity * product.weight$ ) and finally, *date of sales* with *year*, *semester*, *month*, *date* and *day of the week*.

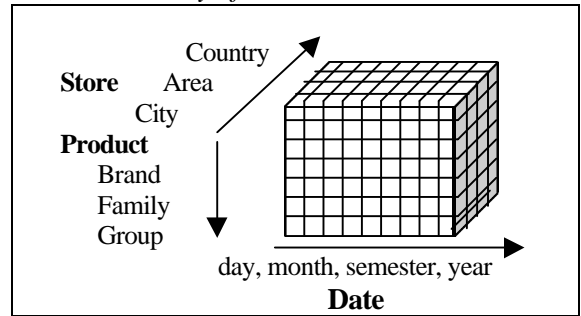


Figure 1. Multidimensional cube

Moreover, for the sake of maintaining the simplicity of this paper, we only define the following attribute classification hierarchies: along the *store* dimension  $city \rightarrow area \rightarrow country$ , along the *product* dimension,  $product \rightarrow type \rightarrow family \rightarrow department$ ,  $product \rightarrow type \rightarrow group$  and  $product \rightarrow brand$ , and along the *date of sales* dimension,  $date \rightarrow month \rightarrow semester \rightarrow year$ .

On the other hand, according to S. Chaudhuri and U. Dayal (1997) in [19], the set of operations generally applied to this cube is as follows: *roll-up* (increasing the level of aggregation) and *drill-down* (decreasing the level of aggregation) along one or more dimension hierarchies, *slice/dice* (selection and projection) and *pivoting* (re-orienting the multidimensional view of data).

## 3 The GOLD model

In this section we summarize the basic concepts of the GOLD model presented in [7] (extended in [8]) and provide a brief outline of the latest achievements of previous versions. A more detailed summary definition of the model can be found in the appendix at the end of the paper. The GOLD model is based on the OO paradigm and a MDB schema is defined by *dimension classes* (DC), *fact classes* (FC), *cube classes* (CC) and *views* (V). DC contain *dimension objects* (DO) that provide characteristics of the factual data, while FC contain *fact objects* (FO) that represent the factual data itself. *Fact classes* are specified as *composite classes* in an aggregation relation where *dimension classes* are the components. *Cube classes*, to which OLAP operations are applied to allow us to accomplish a subsequent data analysis, are then defined from these DC and FC, based on user requirements.

As we have already mentioned in the introduction, there is no multidimensional data model that considers derived measures and derived dimension attributes, since they only consider static properties of data, and

therefore, data functionality and behavior cannot be considered. With our model, however, both data functionality and behavior can be considered, as it uses the OO paradigm. The GOLD model, therefore, defines complex predicates to build derived attributes, both in dimensions and in facts, by applying both arithmetic operations and relational grouping functions.

On the other hand, apart from the conceptual model presented in [12], none of the multidimensional data models presented so far consider the additivity on fact attributes along dimensions. The GOLD model, however, introduces a definition called Aggregation Patterns (AP) on fact attributes to represent this additivity. Fact attributes can therefore be additive if aggregation operations can be applied along all dimensions, semi-additive if it is not additive along all dimensions, and non-additive if it is not additive along any of the dimensions.

With reference to the attribute classification hierarchy, the GOLD model defines dimension attributes as a directed, acyclic and weakly connected graph in which each edge represents a to-one relationship between attributes. We can therefore distinguish between Attribute Roll-up Relation Paths (ARRP) and Attribute Classification Paths (ACP), depending on whether there is a classification hierarchy defined on dimension attributes along the path or not.

For another thing, users can query the database basic schema formed by *dimension classes* and *fact classes*. We define a *cube class* for each basic requirement that the user wishes to execute on the database basic schema. These *cube classes* will encapsulate not only data but also operations allowed on objects in the given class. Thus, OLAP operations (*roll-up*, *drill-down*, *slice-dice* and *pivoting*) that allow us to accomplish a subsequent data analysis are considered as public methods of *cube classes*. By applying these OLAP operations, new *views* are defined on *cube classes* that share the same basic properties, i.e. OLAP operations can also be applied to these *views*. In [8] we defined *roll-up*, *drill-down* and *slice-dice* operations to be applied each time on one dimension of the *cube class*.

As previous concepts, we denote  $A$  as a set of attributes ( $a_1, a_2, \dots, a_n$ ). This set of attributes is defined on domains or primitive classes. Domains consist of a set of values and a set of operations allowed on these values. The domain type is the set of values of the corresponding Data Abstract Type (DAT). Instances of these domains are always in the system, they are neither created nor destroyed and they never change. As basic examples we have the well-known integer and Boolean with their trivial types  $\{0, 1, 2, \dots\}$  and  $\{\text{true}, \text{false}\}$  respectively.

We now present the definition of the *product dimension class* for the example in section 2, as an example of the GOLD model. We also represent all paths within the graph to clarify the distinction between ARRP and ACP<sup>1</sup>.

**Example** *Product dimension class* (DC) is a tuple  $(A_d, ag, E)$ , where

- $A_d = KA \cup DA \cup DDA$ , where
  - $KA = \text{product\_code}$ ,
  - $DA = \{\text{manager, group, department, group, family, quantity, weight, brand, supplier, type, colour}\}$
  - $DDA = \{\text{trans\_cost}\}$ , where  $\text{trans\_cost}$  is the following predicate:  

$$\text{trans\_cost} = \text{product.weight} * \text{product.quantity}$$
- $ag$  is the directed acyclic graph defined on  $A_d$ . Examples of paths defined within  $ag$  are as follows: ARRP paths representing attribute classification hierarchies are  $\text{ARRP}_{\text{cod\_product,brand}}(\text{cod\_product}, \text{brand})$ ,  $\text{ARRP}_{\text{cod\_product,department}}(\text{cod\_product}, \text{type}, \text{family}, \text{department})$  and  $\text{ARRP}_{\text{cod\_product,group}}(\text{cod\_product}, \text{type}, \text{group})$ . Examples of ACP are  $\text{ACP}_{\text{department,manager}}(\text{department}, \text{manager})$ ,  $\text{ACP}_{\text{cod\_product,weight}}(\text{cod\_product}, \text{weight})$  and  $\text{ACP}_{\text{type,supplier}}(\text{type}, \text{supplier})$ ,  $\text{ACP}_{\text{cod\_product,quantity}}(\text{cod\_product}, \text{quantity})$ ,  $\text{ACP}_{\text{cod\_product,colour}}(\text{cod\_product}, \text{colour})$ ,  $\text{ACP}_{\text{cod\_product,trans\_cost}}(\text{cod\_product}, \text{trans\_cost})$ .
- $E$  is the set of events allowed on the class objects, i.e. “new” and “delete”

In Figure 2, the directed, acyclic and weakly connected graph ( $ag$ ) with ARRP and ACP paths defined in  $ag$  may be observed.

<sup>1</sup> Further application of the GOLD model, based on a project developed with a partner, can be found in [9]

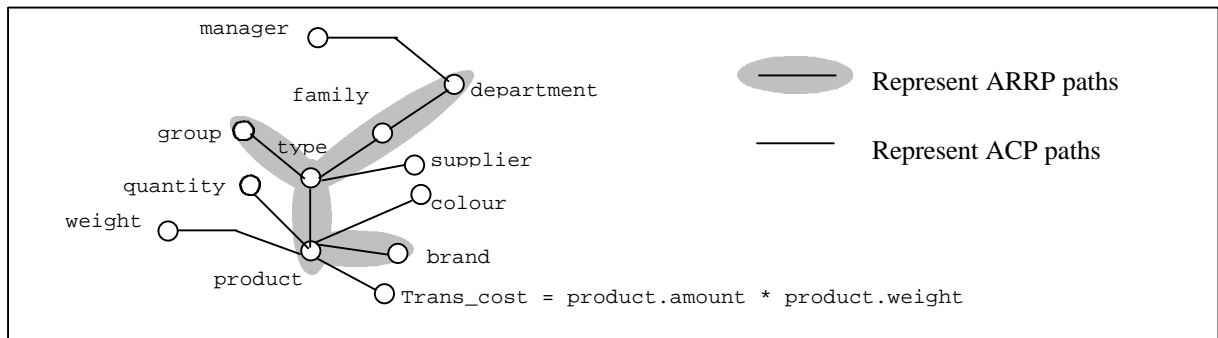


Figure 2. A graphical representation of ARR and ACP paths to represent relationships between attributes

## 4 Discussion

As a future study, soon to be published, we extend the OLAP operations defined in [8], so that they may be applied on more than one dimension at the same time. Furthermore, as the *pivoting* operation is to re-orientate the multidimensional view of data, a formal definition of the cube is previously required. It is for this reason that the definition of the *pivoting* operation, together with the formal definition of the cube structure, are to be extended in a future paper. Moreover, all OLAP operations that a user can execute are public methods in *cube classes*. Furthermore, the *pivoting* operation needs to call other private methods within *cube classes* to re-orientate the multidimensional view of data.

OLAP operations that the user can execute depend on the structure of the attributes within the dimensions. For that reason, we identify, in [10], relationships between dimension attributes for each dimension included in the *cube class* definition, and define behavioral patterns (BP) in which we associate operations with data to accomplish a navigational process. We then define the set of OLAP operations that can be applied to *cube classes*, and therefore, their particular *views*. Furthermore, we extend the set of OLAP operations, defined in [8], with two more operations, (*Combine* and *Divide*), for navigational process when there is no classification hierarchy defined on attributes.

There is research being focused on the linking of our GOLD model with an appropriate Object Definition Language (ODL) to define multidimensional conceptual schemas. This is why we have selected a subset of the OASIS formal specification language ([15], [14]). One of the main features of this language is that it provides an ODL to specify conceptual schemas based on the OO paradigm. Although we need not use all the potential that a formal OO specification language provides, we will extend OASIS to deal with the peculiarities of multidimensional databases. In this way, GOLD classes will be considered as OASIS classes with multidimensional capabilities. Another advantage is that the same language can be used as an Object Query Language (OQL) ([6]). This feature has an important advantage: we could represent concrete queries on *classes* using the OQL language. A further aim of the GOLD model is for it to be implemented by an OO language to enable us to test the GOLD model by querying databases, however; this step has not yet been developed.

## 5 Conclusions

In this paper we have presented the GOLD model, an Object Oriented multidimensional data model, and have demonstrated that the GOLD model can achieve better standards than the models proposed up to now. Examples of these achievements are the consideration of derived measures, derived dimension attributes, the additivity on fact attributes along dimensions and the encapsulation of data with its operations in classes. We are convinced that the application of the OO paradigm allows us to consider all the key issues in multidimensional modeling that other approaches hardly consider, as previous studies presented cannot consider either data functionality or behavior.

## Acknowledgements

I wish express my gratitude for the helpful comments of the anonymous referees of my previous papers presented on the GOLD model, which have helped me to improve on the first version of the model. I would also like to acknowledge my Ph.D. advisor, Manuel Palomar, and my colleagues, Il-yeol Song and Jaime Gomez, for their invaluable help in the organization of my research.

## Appendix

**Definition 1** Let  $A$  be a set of attributes, we define the key attribute (KA) as an attribute  $a_i \subseteq A$  that defines univocally every object of a particular class

**Definition 2** Let  $A$  be a set of attributes of a particular class, we define  $ag = (V, E)$  as a directed, acyclic and weakly connected graph with its root in  $v_0$  (KA) where each vertex ( $v_i \in V$ ) is an element of  $A$ . We say that  $ag$  is a *quasi-tree* with root in  $v_0$  if each other vertex  $v_j \in V$  can be reached from  $v_0$  through at least one directed path. We will denote  $path_{ij}(ag) \subseteq ag$  as a directed path (if exists) starting in  $v_i$  and ending in  $v_j$  and then we will denote with  $sub(v_i) \subset ag$  the quasi-tree rooted in  $v_i \neq v_0$ . Then, we denote  $ARRP_{ij}(ag) \subseteq path_{ij}(ag) \subseteq ag$  as an Attribute Roll-up Relation Path that defines a classification hierarchy on attributes. On the other hand, we denote  $ACP_{ij}(ag) \subseteq path_{ij} \subseteq ag$  as an Attribute Classification Path where there is not any classification hierarchy defined on attributes.

**Note.** We should take into consideration that the distinction between ARR and ACP is to differentiate attribute classification hierarchies, although arcs within both kinds of paths represent to-one relationships between a pair of attributes. Finally, for the sake of comprehension in the definition of ARR and ACP we provide the list of all vertices within the path (see example at the end of section 3).

**Definition 3** Let ARR be an Attribute Roll-up Relation Path within  $ag$ , we define the roll-up domain function as  $domroll-up: v_i, a_j \rightarrow V_j$  to obtain the attribute values according to the classification hierarchy. Conversely, we define the drill-down domain function as  $domdrill-down: V_j, a_i \rightarrow V_i$

**Definition 4** We define a *dimension class* (DC) as a 3-tuple  $(A_d, ag, E)$ , where

- $A_d = KA \cup DA \cup DDA$  is the set of attributes that characterizes *dimension objects* where,
  - KA is the *key attribute*,
  - DA is the set of *dimension attributes*, i.e. attributes that provide more characteristics about the objects contained in the *dimension class* and,
  - DDA is the set of *derived dimension attributes* i.e. attributes obtained from DA or DDA. Each element in DDA is a predicate  $p$  of the form:

$$p = [<op_1>] a_1 <op_2> a_2 <op_3> a_3 <op_4> \dots <op_n> a_n$$

where  $a_i \in DA \mid DDA$  and  $<op_i>$  represents an arithmetic operator or a relational grouping function (e.g. SUM, COUNT or AVG).

**Note** that the first operator can be omitted as well as it may be necessary for expressions like  $p_1 = \text{SUM}(a_i)$ .

- $ag$  is the directed acyclic graph defined on  $A_d$ .
- $E$  is the set of events allowed on the class objects, i.e. “new” and “delete” to create and destroy objects respectively.

Before making the fact class definition, we must define the kind of aggregation operations that can be applied to *fact attributes*. To achieve this, we classify *fact attributes* into three groups, since this defines the kind of aggregation operations that can be applied to them. We follow the classification done by W. Lehner in [25], and therefore, fact attributes can be either  $\Sigma$  (data that can be summarized),  $\phi$  (data that may be used for average calculations) or  $c$  (constant data which implies no application of aggregation operators). Thus, if the aggregation type is  $\Sigma$ , the aggregation operations that can be applied to the data are SUM, AVG, MIN, MAX and COUNT. On the other hand, if the aggregation type is  $\phi$ , the aggregated operations allowed on data are AVG, MIN and MAX.

**Definition 5** Let  $DC_1, DC_2, \dots, DC_n$  be  $n$  *dimension classes*, we define a *fact class* (FC) as an *composite class* in an aggregation relation where  $n$  *dimension classes* are the components. FC is a 4-tuple  $(DC, A_f, AP, E)$ , where

- DC is the set of components in the aggregation, i.e. the set of *dimension classes*.
- $A_f = KA \cup FA \cup DFA$ , where,

- KA is *key attribute*,
- FA is the set of *fact attributes (measures)* that provide more characteristics about the objects contained in the *fact class* and,
- DFA is the set of *derived fact attributes (derived measures)*, i.e. attributes obtained from FA, DA or DDA. Each element in DFA is a predicate ( $p$ ) of the form:

$$p = [<op_1>] a_1 <op_2> a_2 <op_3> a_3 <op_4> \dots <op_n> a_n$$

where  $a_i \in FA \mid DA \mid DDA \mid DFA$  and  $<op_i>$  represents an arithmetic operator or a relational grouping function (e.g. SUM, COUNT or AVG).

**Note** that the first operator may sometimes be omitted, or may be necessary for expressions like  $p_i = \text{SUM}(a_i)$ . We will use the format *class\_name.attribute\_name* to allow us to know to which dimension class each attribute belongs.

- AP (Aggregation Pattern) is a triple of the form:

$$AP = (a_i, d_j, \text{agt})$$

where  $a_i \in FA \mid DFA$ ,  $d_j \in DC \mid \{ALL\}$  and  $\text{agt} \in \{\Sigma, \phi, c\}$ . This triple states that the *fact attribute (atomic or derived)* can be aggregated along the dimension  $d_j$  by applying the corresponding aggregation operations (depending on the attribute aggregation type,  $\text{agt}$ ). As most fact attributes are additive along all dimensions, we can express it by  $AP = (a_i, ALL, \text{agt})$ . We can also state that a fact attribute can be aggregated along no dimension (non-additive attribute) with  $AP = (a_i, ALL, c)$ . Semi-additive attributes are also represented, expressing along which dimension they are additive and the type of aggregation operator to be applied.

- E is the set of events allowed on FO, i.e. “new” and “delete”

**Definition 6** Let  $DC_1, DC_2, \dots, DC_n$  be  $n$  *dimension classes* and FC be a *fact class* built from these  $n$  dimension classes, we define a *cube class (CC)* as a 6-tuple  $(DC, FC, A_c, C, E, CE)$ , where

- DC is the set of *dimension classes* that have been used for constructing the *cube class*
- FC is the *fact class* from which the cube class has been built
- $A_c = KA \cup CFA \cup CDFA \cup CDA \cup CDDA$  where,
  - KA is the *key attribute*,
  - CFA is a subset of FA from the FC,
  - CDFA is a subset of DFA from the FC,
  - CDA is a subset of the DA from the  $DC_i$  and,
  - CDDA is a subset of the DDA from the  $DC_i$ .
- C is a condition  $n$ -tuple  $(a_1=v_1, a_2=v_2, \dots, a_n=v_n)$  where  $a_i$  are dimension attributes and  $v_i$  the values that the attribute  $a_i$  of each object must fulfill to include that object in the CC.
- E is the set of operations allowed on the objects of the CC, i.e. “new” and “delete”.
- CE is the set of events (OLAP operations) permitted on the *cube class* ([7] and [8]).

**Note 1.** By defining both *dimension attributes* and *fact attributes* while constructing the CC, we use the following format: *class\_name.attribute\_name* to allow us to know to which class each attribute belongs.

**Note 2.** If  $a_i = a_j$  with  $v_i \neq v_j$  both kinds of objects will be selected. Moreover, attributes used in the condition tuple will not be included in the CDA or CDDA list, to avoid redundancy. Moreover, if no condition is desired on this CC, we denote it with the tuple  $(KA=ALL)$  in which ‘ALL’ indicates that all the objects in the *fact class* will be included in the *cube class*.

## References

- [1] A. Datta and H. Thomas, “A conceptual Model and an algebra for On-Line Analytical Processing in Data Warehouse”, Workshop on Information Technologies and Systems, Atlanta, 1997.
- [2] C. Li and X. Wang, “A Data Model for Supporting On-Line Analytical Processing”. Proc. in Intl. Conf. on Information and Knowledge Management, (CIKM’96), Rockville (Maryland) USA Nov. 1996, pp. 81-88.
- [3] C. Sapia, M. Blaschka, G. Höfling, and B. Dinter, “An Overview of Multidimensional Data Models for OLAP”. Technical Report, <http://www.forwiss.tu-muenchen.de/~system42/>.
- [4] E.F. Codd, S.B. Codd, C.T. Salley, “Providing OLAP (On-Line Analytical Processing) to User Analyst: An IT Mandate”. Available from Arbor Software’s web site <http://www.arborsoft.com/OLAP.html>.

- [5] J. Gray, A. Bosworth, A. Layman and H. Pirahesh, "Data Cube: A Relational Aggregation Operator Generalizing Group-by, Cross-Tab and Sub Totals". *Data Mining and Knowledge Discovery Journal*, vol. 1, no 1, 1997.
- [6] J. H. Canós. "OASIS: un lenguaje único para bases de datos orientadas a objetos". Ph.D. Thesis, Universidad Politécnica de Valencia, 1996.
- [7] J. Trujillo and M. Palomar, "An Object Oriented Approach to Multidimensional Database Conceptual Modeling (OOMD)". In Proc. of the ACM 1<sup>st</sup> International Workshop on Data warehousing and OLAP (DOLAP'98), Washington D.C., The USA, Nov. 1998, pp. 16-21.
- [8] J. Trujillo and M. Palomar, "An Object Oriented Approach to Multidimensional Databases & OLAP operations". *Journal of Computer Science and Information Management*, 1999. (To appear).
- [9] J. Trujillo, "Specification of a multidimensional model based on the Object Paradigm (OOMD)". Research Project (Spanish), February, 1999. University of Alicante.
- [10] J. Trujillo, M. Palomar, J. Gomez and I. Song. "Detecting behavioral patterns for OLAP operations in the GOLD model". Submitted for publication to the First International Conference on Data Warehouses and Knowledge discovery (DaWaK'99), Florence, Italy, August, 1999.
- [11] L. Cabibbo and R. Torlone, "A Logical Approach to Multidimensional Databases". *Lecture Notes in Computer Science*, number 1377, in proc. of the 6th Int. Conf. On Extending Database Technology, (EDBT'98), Valencia, Spain, March.1998, pp. 183-197.
- [12] M. Golfarelli and S. Rizzi, "A methodological Framework for Data Warehouse Design". In Proc. of the ACM 1<sup>st</sup> International Workshop on Data warehousing and OLAP (DOLAP'98), Washington D.C., The USA, Nov. 1998, pp. 3-9.
- [13] M. Gyssens and L. Lakshmanan "A Foundation for Multi-Dimensional Databases". Proc. in the 33rd Intl. Conf. On Very Large Database Conference (VLDB'97), Athens, Greece, August, 1997, pp.106-115.
- [14] O. Pastor and I. Ramos. "OASIS 2.1.1: A Class-Definition Language to Model Information Systems Using and Object-Oriented Approach". Servicio de Publicaciones. Universidad Politécnica de Valencia, 3rd edition, 1995.
- [15] O. Pastor, F. Hayes, and S. Bear. OASIS: An Object Oriented Specification Language". In P. Loucopoulos, editor, *Advanced Information Systems Engineering* , volume 593 of *Lecture Notes in Computer Science*, pages 348-363. Springer-Verlag, 1992.
- [16] P. Vassiliadis. "Modeling Multidimensional Databases, cube and cube operations". In proc. of 10th Intl. Conf. on Statistical and Scientific Databases (SSDBM'98), Capri, 1998.
- [17] R. Agrawal, A. Gupta and S. Sarawagi, "Modeling Multidimensional Databases". Proc. 13th Intl. Conf. On Data Engineering, (ICDE'97), Birmingham, U.K., April 1997, pp. 232-243.
- [18] R. Kimball, "The data warehousing toolkit". John Wiley, 1996.
- [19] S. Chaudhuri, and U. Dayal, "An Overview of Data Warehousing and OLAP technology". *ACM Sigmod Record*, vol. 26, no 1, March 1997.
- [20] W. H. Inmon, "Building the Data Warehouse". John Wiley, 1992.
- [21] W. Lehner "Modelling Large Scale OLAP Scenarios". *Lecture Notes in Computer Science*, number 1377 in proc. of the 6th Int. Conf. On Extending Database Technology, (EDBT'98), Valencia, Spain. March 1998, pp. 153-167.