Multidimensional Models: A State of Art.

Fernando Carpani

Introduction.

In the last four years, some multidimensional models were defined. Some of these models are focused on query specifications, but only a little has really a conceptual approach.

The next section is an attempt to present some fundamentals over some of “query models”. These models are summarized in [Car97] and [Sap99]. For such models, only some comparisons are made with a special emphasis on data structures.

The following sections are dedicated to some relevant works on Multidimensional Conceptual and Logical Modeling ([Cab97], [Gol98] , [Fra99a] , [Sap99a]).

Later, some conclusions about the works are presented.

To read the next sections, some considerations must be assumed:

- The basic structure on a multidimensional model is the hypercube or cube, a multidimensional array with discrete valued dimensions.
- These dimensions are hierarchically structured and some basic operations allow navigate across them.
- The navigation over dimensions can make a different view, or a transformation of the original cube. Typically, these transformations have an aggregation operation related to.

The “Query” Models.

These models are focused on describe data manipulation operations and not on specify a data structure. However, they must be based on some data structure to express these manipulations.

The data structure is, basically, the same:

> *A Multidimensional Database is a set of Cubes built on some hierarchically structured dimensions with a set of discrete values on each level of each dimension*. *A Cube is a partial function from the Cartesian Product of levels from different dimensions in other set of values called Measures*.

The major difference between these models is the way this structure is represented.

Li and Wang.

For Li and Wang ([Li96]), a cube schema is a set of pairs (D_n,R_n) where D_n is a dimension name and R_n is a set of attribute names (relation schema, similar to Relational Model).

In this model, a cube instance is a pair (F, μ) where F is a set of pairs (D_n,r_n) and μ is a function. The pairs in F are composed by a dimension name that is the first element of some pair in the cube schema, and a relation for the attribute set associated to the dimension name in the cube schema. The second element of the pair μ is a function which associate a scalar value in the set ν to each value in the set $\{\{(D_1,t_1,),...,(D_n,t_n)\} \mid \forall(i\in[1,n]).(ti \in ri)\}$. This set can be seen as the set of all n-tuples where each tuple is composed by each $t_i \in r_i$ in F.

An advantage of this point of view is that a cube is a function from dimensions into some values.

The queries in this model can be made with a language named *grouping algebra*. This language is a Relational Algebra extension with notions similar to *order by* and *group by* in SQL. This makes the model Relational dependent, i.e. it is too oriented to Relational Data warehouses.

The formalizations appear as too intricate to specify an SQL-like language. This model can be seen as a specification of *Star Schema [Kim96]* .

Gyssens & Lakshmanan.

The model presented in [Gys97] is based in one basic structure that is a representation of a cube: the table.

Like the previous one, this model is highly based in the Relational Model but the structure is softly different.

A *table schema* is a triplet <*D,R,Pair*> where:

- $D = \{d_1,\ldots,d_n\}$ is a set of dimension names.
- $R = \{A_1,\ldots,A_n\}$ is a set of attributes.
- Pair: $D \rightarrow 2^{\{A_1,\ldots,A_n\}}$ such
  1) $\forall(<i,j>\in[1,n]X[1,n])(i\neq j \rightarrow (Pair(d_i) \cap Pair(d_j) = \varnothing))$
  2) $\cup_{d\in D}Pair(d) \subseteq R$

This structure is similar to the one used by the previous proposal, but has a different way to associate a dimension name with a set of attributes. While Li and Wang use a list of pairs, this proposal uses a function (*Pair*) from dimension names into sets of attributes names with some constraints over the dimensions: the dimensions must be disjoint.

In this proposal, a cube can have a set of measures. The measures are attributes from $R - \cup_{d\in D}Pair(d)$.

An *instance* for a table is defined as n+1 finite relations where the first n relations are over the attributes from each dimension with an extra attribute for tuple identification (T_{id}). The last relation is over the attributes from M and has each one (T_{id}) from the first n relations. With a simple constraint, this can be seen as a specification for *Star Schema* too. This constraint is:

$$\pi_{Tid}(r_{d1}) \otimes \ldots \otimes \pi_{Tid}(r_{dn}) = \pi_{rd1.Tid,\ldots,rdn.Tid}(r_m) \ (\otimes \text{ natural join})$$

In this model, the operations are presented as a relational algebra extension and have some operations oriented to representation. A table has two representations: One tabular and one relational (see Fig. 1 and Fig. 2). This can be considered as an advantage because simplify the algebra definition. When an operation is inherited from Relational Algebra, is defined as the same operation in the relational representation. This example was taken from [Gys97] .

PRODUCTION			Time						
			Year	1996		...	1997		
			Month	Jan.	Feb.	...	Jan.	Feb.	
CAT.	Product	Employee	(Prod. / hour, Garbage)						
	Pistons	John Smith		(5,6)	(5,7)	...	(4,6)	(4,8)	...
		Joe Jones		(5,7)	(5,8)	...	(4,8)	(4,9)	
		...		...	...		...	...	
	Valves	John Smith		(7,8)	(7,9)	...	(6,9)	(6,8)	
		Joe Jones		(6,9)	(6,9)	...	(5,8)	(5,9)	
		...		...	...	...	...	...	
	...	...		...	...	...	...	...	

Fig. 1. A multidimensional representation of a table.

Category

Tid	***Part***	***City***
c1	Pistons	J. Smith
c1	Pistons	J. Jones
...	...	...
c6	Valves	J. Smith
c7	Valves	J. Jones

Time

Tid	***Year***	***Month***
t1	1996	Jan.
t2	1996	Feb.
...	...	...
t13	1997	Jan.
t14	1997	Feb.

rm

C.tid	***T.tid***	***Prod/hour***	***Garb.***
c1	t1	5	6
c1	t2	5	7
...	...	...	...
c6	t13	6	9
c6	t14	6	8

Fig. 2. A relation representation of a table.

The relational algebra is extended with aggregate functions and “group by” mechanisms.

There are two functions for restructuring tables: Fold to transform a dimension attributes into measures, and Unfold to transform measures into dimension attributes.

These operations are oriented to implement the symmetry between dimensions and measures. ([Cod93])

In summary, this model can be seen as a more simple specification for Star Schema than the previous model. Also has some advantage as the symmetry between dimensions and measure. However, still is Relational Dependent.

Agrawal, Gupta and Sarawagi.

The model is presented by the authors as a logic model, as an attempt to uniform and extend the multidimensional database functionalities. The model yields on some geometric notions.

The basic structure is the *Hipercube*. A hipercube has two elements:

- Dimensions. A dimension is a name d_i and an associate domain dom_i.
- Elements (or measures). This is defined as a map

$$E(C): dom_1 X \ldots X dom_n \rightarrow \text{n-tuples} \cup \{0,1\}$$

The set n-tuples are tuples of measures.

In this model, a coordinate can be mapped to a Boolean or to a tuple. If the result of such application is a tuple, then must be a tuple with the same structure for each other coordinate. If the result is a Boolean then must be a Boolean to each other coordinate. In this way, a hypercube can be seen as a Boolean function and with a process similar to functional curryfication dimensions can be transformed in measures.

An algebra is defined over the model. This algebra is not directly relational algebra. It has different operations.

Some operators are:

- *Push*. Transform a dimension in a measure.
- *Pull*. Transform a measure in a dimension.
- *Destroy Dimension*. Delete a hypercube dimension. The dimension to delete only can have one value.
- *Restriction*. Delete each value in the hypercube whose respective values in a specific dimension does not verify certain condition.
- *Join*. With two hipercubes, construct a new hypercube. Each dimension in the first hypercube is combined with only one dimension of the second hypercube. The result is a dimension with the union of the original dimensions. The measures from each cube for each coordinate are combined in a new measure, as result of an operation f_{elem} .

In summary, this model has a great generality degree, based in a functional view of multidimensional structures and operations. The major features of the model are summarized in the following items:

- There is not an explicit difference between schema and instance of a structure.

- This model has support for symmetry between measures and dimensions. To this introduces a "Boolean cube" which is used by other models
- The model can be mapped to SQL in an easily way. Therefore, this model is valid in a ROLAP environment, but it is not exclusive. The model can be used on MOLAP or HOLAP environments to.
- There is an explicit operation to relate two hipercubes.
- There is not an structural way to represent hierarchies. These are managed with auxiliary functions like f_{merge} or f_{elem}.
- This model is presented by the authors as a logical model so, it is oriented to express manipulations.

Logical and Conceptual Models.

In this section, the main models with conceptual features are sketched. These models will be presented deeper than the previous ones in order to expose in the future the differences with CMDM.

The assumptions presented in the introduction are also valid with these models and the basic multidimensional structure underlying are the same as the previous.

Cabibbo and Torlone.

Their proposal is based on a logical model named *MD*. This work can be read in three papers: Querying Multidimensional Databases ([Cab97]) , A Logical Approach to Multidimensional Databases ([Cab98]) and Data Independence in Olap Systems ([Cab99]).

The first work, presents a logical multidimensional model with a clear multidimensional structure, a calculus as query language, and some results on query languages expressive power.

The second work is oriented to logical design on OLAP and is based on the model presented in the first work. Here, some design methodology is presented. This methodology is oriented to transform an ER schema of operative data into a MD schema.

In the third work, an architecture for Olap Systems is presented introducing the Data Independence notion on these systems.

In this section, the first version of MD model is presented. This is in the assumption that this version of MD is the major advantage of their proposal.

This work can be considered as a "Query Model", but it's recognized by its authors as a logical model and, in [Cab98] its conceptual features are exposed.

Data Structures.

There are two fundamental structures in the model: the *dimension* and the *f-table*. The last one is used to represent the cube notion.

A multidimensional database is a pair with a set of dimensions and a set of f-tables constructed over those dimensions.

A set of names £ is assumed. Each name in this set is called *level*. This set has two conditions:

- For each $l \in$ £, exists a set of values DOM(l). These sets are called *level domains*.
- For each $l \in$ £ and for each $l' \in$ £, $DOM(l) \cap DOM(l') \equiv \varnothing$.

The last condition express that every pair of level domains are disjoint.

Schemas.

Formally, a *dimension schema* is a triplet **<L,≤,R-UP>** where:

- L is a set such: $L \subseteq$ £ and L is finite. This is a finite set of level names.
- ≤ is a partial order over L. When $l_1 \leq l_2$ you can say that l_1 *rolls-up to* l_2 or that l_2 drills-down to l_1. This is a partial order over the levels and represents the dimension hierarchy.
- R-UP is a family of functions: the roll-up functions. These functions describe the way the navigation over the dimension hierarchy from an element level to another one takes place.

An f-table schema has the following form:

$$\mathbf{f\ [\ A_1:\ l_1,....,A_n : l_n\ ] : l_0}$$

Each l_i is a level from any dimension. The symbol *f* is the schema name and each A_i is an attribute of *f*. The level $\mathbf{l_0}$ is called a *measure*.

As were presented above, a multidimensional schema is a pair **<D,F>** where:

- D is a set of dimensions.
- F is a set of f-tables such each l_i in each f-table in this set, is a level from some dimension of D.

Instances.

A *symbolic coordinate* over a f-table $f[\ A_1:l_1,....,A_n:l_n\ ]:l_0$, is a function which maps each attribute name A_i in an element of $DOM(l_i)$ where $1 \leq i \leq n$. This definition is congruent with the classical notion of *tuple*.

An instance of an f-table $f[\ A_1:l_1,....,A_n:l_n\ ]:l_0$ is a function coordinates over f in $DOM(l_0)$. This function is defined over a finite set of coordinates over f.

An instance of a multidimensional schema S=<D,F> is a function which maps each f-table schema in F to an f-table instance.

Other Features.

A family of query languages is defined over these structures under the form of a parametric multidimensional calculus. The generic form of a query is the following:

$$\{\ x_1,...,x_n : x\ |\psi(x,x_1,...,x_n)\}$$

In this expression, x_i are variables and x is a distinguished variable called *result variable*. The expression denoted with ψ is a first order formula involving roll-up expressions, scalar functions and aggregate functions. The scalar functions can be user or system provided. Based on this feature, the language can be oriented to different applications.

With two query examples, the application of roll-up and f-table instances can be saw.

The f-table *production* represents the diary units produced, classified by product and employee.

Production(*day:*day, *item:*product_id, *name:*employee_name): numeric

1. To represent the diary units from each product by the employee with name Smith, the following query can be used to build a new f-table:

   $SMITH\_UNITS = \{ x_1, x_2, x_3 : x \mid$

   $x = Production[day: x_1, item:x_2, name:x_3] \wedge$

   $\text{“SMITH”} = R\text{-}UP_{employee\_id}^{name}(x_3)$

   $\}$

2. To define a new f-table to represent the summary of production in a week of each item and in each factory area, the following query can be used:

   $PRODUCTION\_SUMMARY = \{ x_1, x_2, x_3 : x \mid$

   $x = sum (y_1, y_2 : y \mid$

   $y=Production[day: y_1, item:x_2, name:y_2] \wedge$

   $x_1= R\text{-}UP_{day}^{week}(y_1) \wedge$

   $x_3= R\text{-}UP_{employee\_id}^{area}(y_2))$

   $\}$

This query language allows defining a new Boolean f-table from any previous f-table in a practical way. This feature can be used to express the symmetry between dimensions and measures. These Boolean f-tables are called by the authors, *abstractions*.

Conclusions.

The model represents the basic data structures in a direct way. This is an advantage to build a conceptual model over it.

The roll-up paths are represented in a structural way. Because of this fact, an increment of precision in the analyst descriptions may be raised.

In [Cab98], the version of the model includes a specification of *descriptions* that are descriptive attributes.

The feeling of these papers is that the authors think that a level must be scalar or string. This constraint is not explained in nowhere.

CMDM is based on this model, proposing some changes and extensions.

Lehner.

In [Leh98a] presents the “Nested Multidimensional Model”.

This model is oriented to operations, but implements multidimensional structures in a direct way. It can be considered as a logical model.

Structures.

The dimensional structures are based on the fact that the attributes are classified in the following three categories:

- **Primary attributes (PA)**. It identifies the *dimensional elements*, i.e. the elements of a dimension.
- **Classification attributes (CA)**. It allows structuring the elements of the dimension in levels. These attributes describe a balanced tree called *classification hierarchy* where each node is an instance of the classification attribute for that level.
- **Dimensional attributes (DA)**. It describes features of the dimensional elements.

The classification hierarchy has a value for each classification attribute in its internal nodes and a value for the primary attribute for the dimension in its leaves. The root always is the level *top* with the node *all* that must be considered as a classification attribute. In Fig. 3 can be seeing the different attributes for a dimension.

For de root and for each internal node, are defined two domain types:

- **Classification Oriented Domain (DOM($C_{|CA}$))** contains the values of each classification or basic object from the level CA that belongs to the sub-tree rooted on the node C.
- **Feature (or Description) Oriented Domain (DOM($C_{|DA}$))** contains the values of the attribute DA for each node that belongs to the sub-tree rooted on the node C.

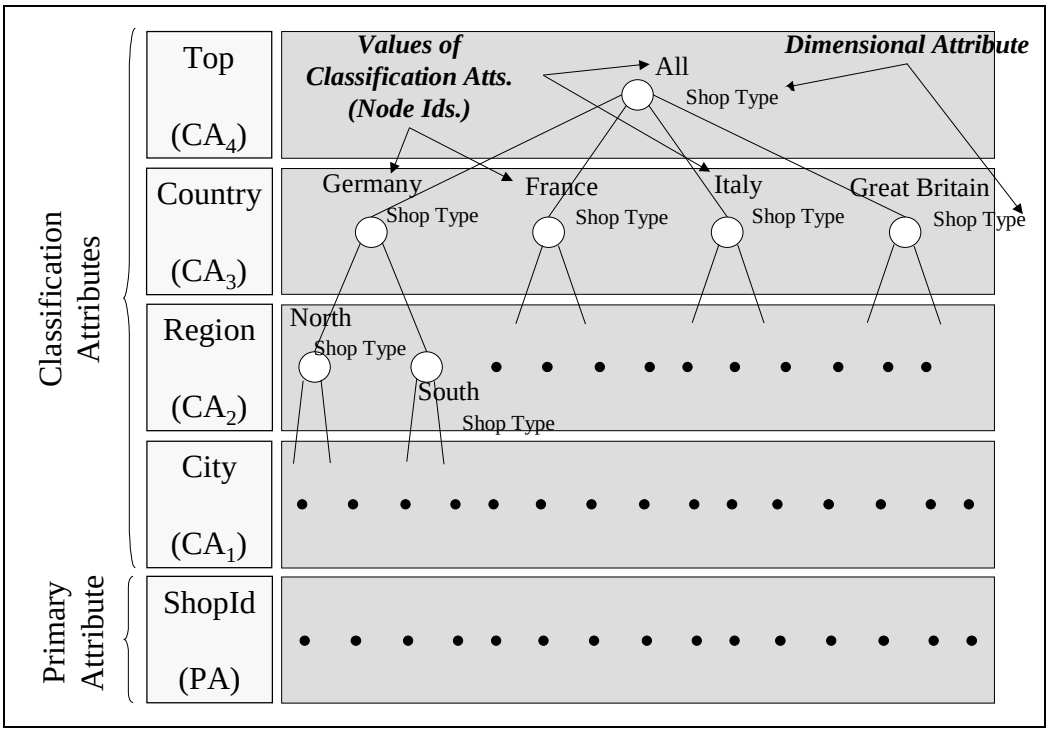

Fig. 3. Node Domain an attribute Identification for a Shop dimension.

The expression *DOM (Germany|$_{Region}$)* denotes the classification domain for the node *Germany* with respect to the level (or classification attribute) *Region*. This domain contains only the values *North* and *South*.

The expression *DOM(Germany|$_{ShopType}$)* denotes the description domain for the node *Germany* with respect to the *Shop Type* description attribute. The values contained in this domain are the values for the attribute *Shop Type* for the sub-tree rooted on this domain, so, must contain the values of *DOM(North$_{|ShopType}$)* and *DOM(South$_{|ShopType}$)*.

The attribute *Shop Type* is on all nodes but is not a rule. A dimensional attribute must be in a context (sub-tree) depending on the value of the node. In [Leh98a] there is an example where exists a Product dimension. Some products are Video Recorders, Washers and Dryers. The Video Recorders can have an attribute Video System (pal, ntsc, etc.) which no sense for Washers and Dryers. This dimensional attribute only appears in the sub-tree for Video Recorders.

Over this notions, the author defines three types of multidimensional objects:

- **Primary Multidimensional Objects (PMO)** are tuples with five items: a selection item (context descriptor), an aggregation item (context descriptor schema), an aggregation operation type, a data type and a cell identifier. The selection item is a tuple of node identifiers, i.e. a tuple of classification attributes values. The aggregation item is a tuple of classification attributes, which match with the values in the selection. The aggregation operation type can be summarization, average or a constant. The data type can be natural, integer or real. The cell identifier is only a name for the structure. Note that in any of the previous items are dimensional attributes.
- **Secondary Multidimensional Objects (SMO)** are pairs with a context descriptor and a set of dimensional attributes. This set of dimensional attributes must be a subset of the union of the dimensional attributes of each node included in the context descriptor.
- **Multidimensional Objects (MO)** are pairs with a PMO and set of dimensional attributes. This dimensional attributes are used for define a set of SMOs associated with the PMO.

The PMOs defines the dimensional structure of a cube. The SMOs adds the measures using the contextual information of the PMO. The MOs are really cubes with a SMO for each element in the PMO.

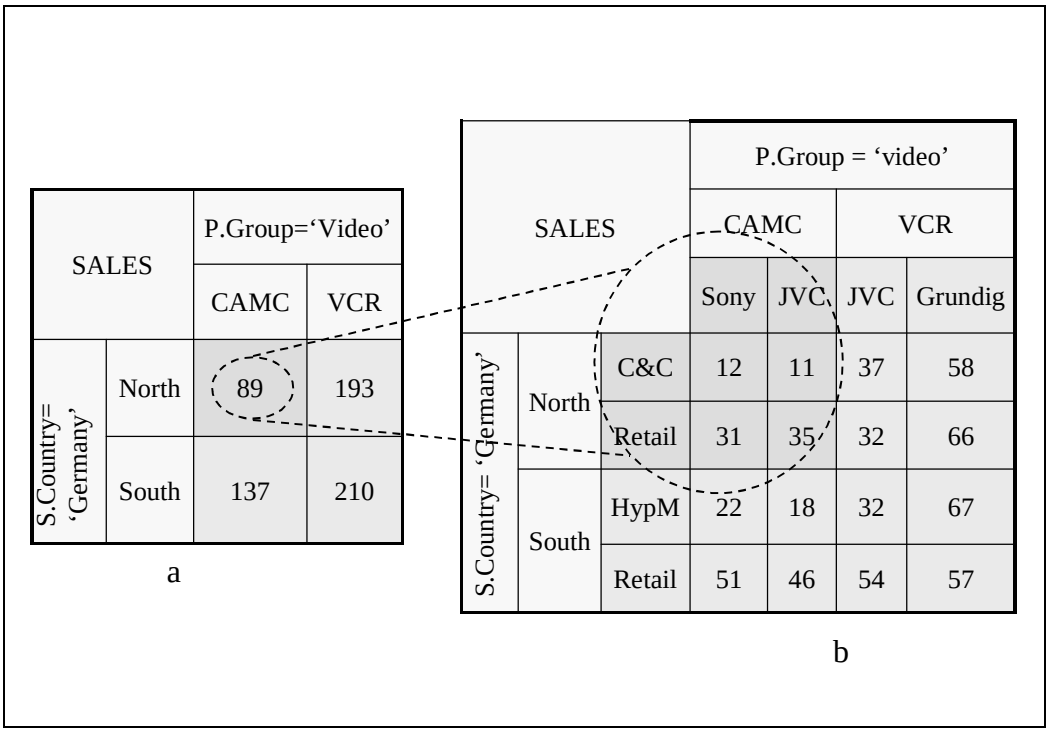

Fig. 4. Sample MOs in a tabular representation showing context-sensitive nesting.

The primary and secondary multidimensional objects have domains formally defined. The domain of a PMO is the Cartesian product of the classification-oriented domain for each element in the context descriptor.

In the table **a** in Fig. 4, the clearest cells are the PMO. The first cell is the *cell identifier*, the first column and line, which have conditions, are the context descriptor and the second column and line have the elements of the classification-oriented domain for each element from the context descriptor. The rest of the table is a representation of the cartesian product of the domains of the context descriptor. Note that the lines rises of a Shop dimension and the columns rises of a Product dimension.

The figure represents a MO so; in each element of this Cartesian product, there is a SMO. The domain of a SMO is the cartesian product of the description-oriented domain of the elements in the context descriptor with respect to the dimensional attributes in the SMO. In the table **b**, each SMO is presented as a inner table (darker shaded) for each element of the PMO.

The darkest zone in the tables shows a specific SMO. In the **a** table, this SMO is 0-dimensional, but in the **b** table is 2-dimensional.

Note that in the PMO there is an aggregation operation type not an aggregation function. If the aggregation operation type is summarization, the function can be SUM, AVG, MIN, MAX or COUNT. If the aggregation operation type is average, the function can be AVG, MAX or MIN. If the aggregation type is constant, there is no function to apply.

In the example can be assumed that the operation is count.

In the rest of the paper, an algebra is defined, with the natural multidimensional operation included (roll-up, slice, etc.)

Conclusions.

The model must be considered as a logical model in spite of being query oriented.

In spite of the model is query oriented, must be considered as a logical model because expose clearly the multidimensional structures and operations.

In the analyzed models, only this one has some notions of summarizability.

The model seems to be adequate to express dynamic cubes. This feature rises from the context definitions.

Golfarelli, Rizzi et. al.

In [Gol98a] the authors present a multidimensional data model called Dimensional Fact Model (DFM.). A method to construct a multidimensional schema in this model from an E/R schema is also presented. This model is presented as graphical notation without any formalization.

In [Gol98] and in [Gol99] there is a reformulation of the model focused on a design methodology and it formalization is presented.

Since this model is focused on conceptual design, the emphasis is on the structures and not on operations.

Structures.

Graphical Model.

The basic structure of the model is the *fact scheme*. A graphical representation can be saw Fig. 5. This example was taken from [Gol98a].

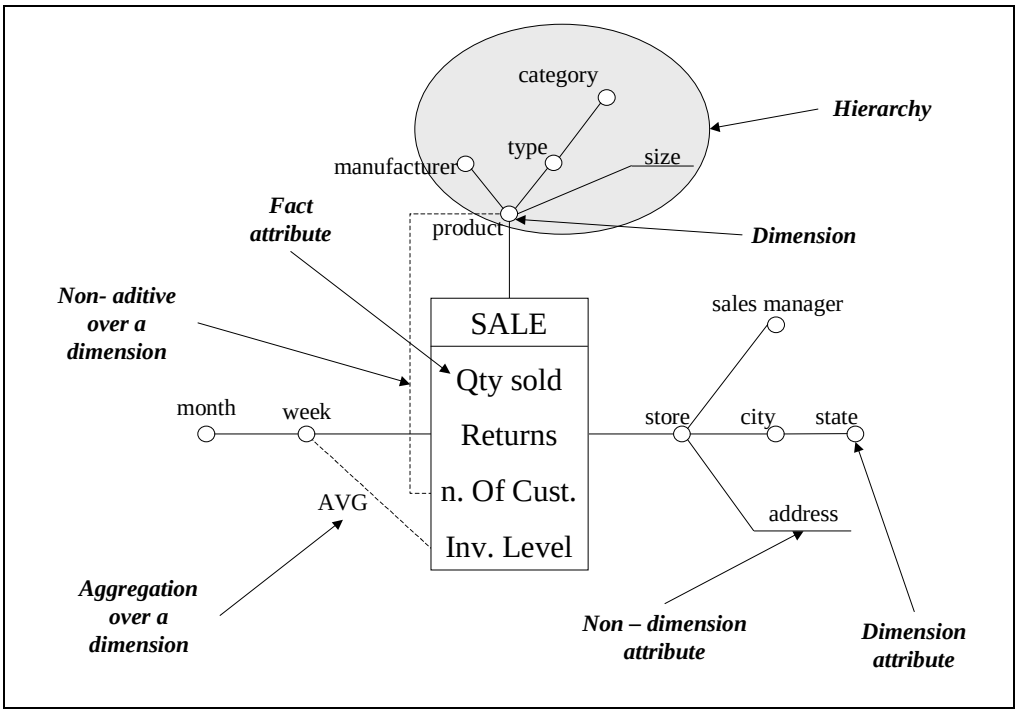

Fig. 5. A Fact Scheme.

Formal Model.

This structure is formalized using a kind of Directed Acyclic Graph (DAG) called *quasi-tree*. If g=(V,E) is a directed, acyclic and weakly connected graph, then g is a *quasi-tree* with root v_0, if all vertexes $v_j \in V$ must be reached from v_0 through at least one directed path.

A fact scheme is a tuple

f=(M,A,N,R,O,S)

where:

- M is a set of numerical or Boolean expressions involving values from the operational systems. Each $m_i \in M$ is called a *measure*.
- A is a set of identifiers called *dimension attributes*. Each $a_i \in A$ has a discrete domain *Dom(a_i)* associated with it.
- N is a set of identifiers called *non-dimension attributes*.
- R is a set of ordered pairs. Each pair (a_i, a_j) verify that $a_i \in A \cup \{a_0\}$ and $a_j \in A \cup N$ and $a_i \neq a_j$ and *qt(f)* is a quasi-tree with root a_0 where qt(f) is the following graph:

  qt(f)=($A \cup N \cup \{a_0\}$,R)

  The dummy vertex a_0 represent the facts.

- $O \subset R$ is a set of optional relationships.
- S is a set of *aggregation statements*. Each element of this set is a triplet (m_j,d_i,Ω) where m_j is a measure, $d_i \in Dim(f)$ and Ω is an aggregation operator. If $(m_j,d_i,\Omega) \in S$ then the measure m_j can be aggregated over the dimension d_i using the operator Ω. The expression Dim(f) can be defined as:

$$Dim(f)=\{a_i \in A \mid \exists(a_0,a_i) \in R\}$$

If $d \in Dim(f)$ then d is a *Dimension*. Therefore, a dimension is an attribute directly connected to the facts.

A *hierarchy* over the dimension d_i is the quasi-tree *sub(d_i)*. In this way, the hierarchy over a dimension is the graph of its attributes.

Instances

The instances are defined in two steps.

- In a first step, the *Primary Fact Instances* are defined. This is no other thing that the coordinates: a tuple with one value from each dimension domain. Each primary fact instances describe one value for each fact attribute (measures).
- In the second step, the *Secondary Fact Instances* are defined. These instances are aggregations of primary fact instances. These definitions are based on legal *v-dimensional aggregation patterns* which are sets of dimension attributes such there is no path between two of them.

The formalization of these concepts is hard to be present here. It can be found in [Gol98] and [Gol99].

More over the Model.

Drill-Across Paths.

The model is oriented to represent independent fact schemes. For that reason, there is no a clear way to represent the drill-across paths.

To solve this drawback, a mechanism for the fact schemes integration is proposed. This mechanism can be sketch out as the union of the measures and the intersection of the dimensions. In this way, if a designer needs to specify drill-across paths, then he/she must integrate the two fact schemes involved in the drill-across.

Query and Workload.

The methodology presented in [Gol98] take in account the workload in the data warehouse. So:

- A simple language to express queries is defined.
- The Workload on a dimensional scheme is defined as a set of pairs $<q_i,v_i>$ where q_i is a query and v_i is it expected frequency.

Conclusions.

The formal model is a formalization of the graphical language taking in account some additional considerations. However, it's not a direct formal representation of the multidimensional concepts.

The model doesn't support generic dimensionality and the way to represent the drill-across paths is tricky. However, this method can be applied in a more cases than an explicit drill-across because it can be used for schema integration.

In the notion of *legal aggregation pattern*, there is another underlying notion: Independence between dimensions or at least, levels. This notion is present in the meaning of the term "multidimensional", but only a few times is explicit like here.

Some lines over workload and data volume treatment are presented. Also is presented a translation method to star schema. However, there's nothing about multidimensional DBMS.

This work presents a complete design methodology for DW, which goes from a conceptual to a physical (but relational only) model.

Sapia, Blaschska et al. (System 42).

In [Sap99a] a multidimensional extension to E/R model called ME/R (Multidimensional Entity Relationship) is presented. In the paper, there is a lot of "common sense" in the reasons for extend E/R model.

The extension is a specialization of E/R model, based on the ISO/IRDS standard metadata. So, only the structures are described. The semantic is similar to E/R.

Structures.

In ISO/IRDS, a meta-model is used to describe a model. This meta-model is similar to E/R and the authors use an extended version with a generalization concept. This model is different from that whose extended version will be the ME/R. This last model is our every day E/R.

In summary, three models are in this discussion and must not be confused:

- The meta-model, which is an E/R model with generalization.
- The E/R model, where the modeler decides which constructions are allowed.
- The Multidimensional E/R, which rises from the previous E/R as a specialization.

The ME/R model has three specializations respect to E/R:

A special entity set: dimension level.

A special n-ary relationship set: the "*fact*" relationship set.

A special binary relationship set: the "*rolls-up to*" relationship set.

The meta-model of ME/R can be saw in the Fig. 6. The gray zone is the extensions added to E/R. The rest is the meta-model of the basic E/R.

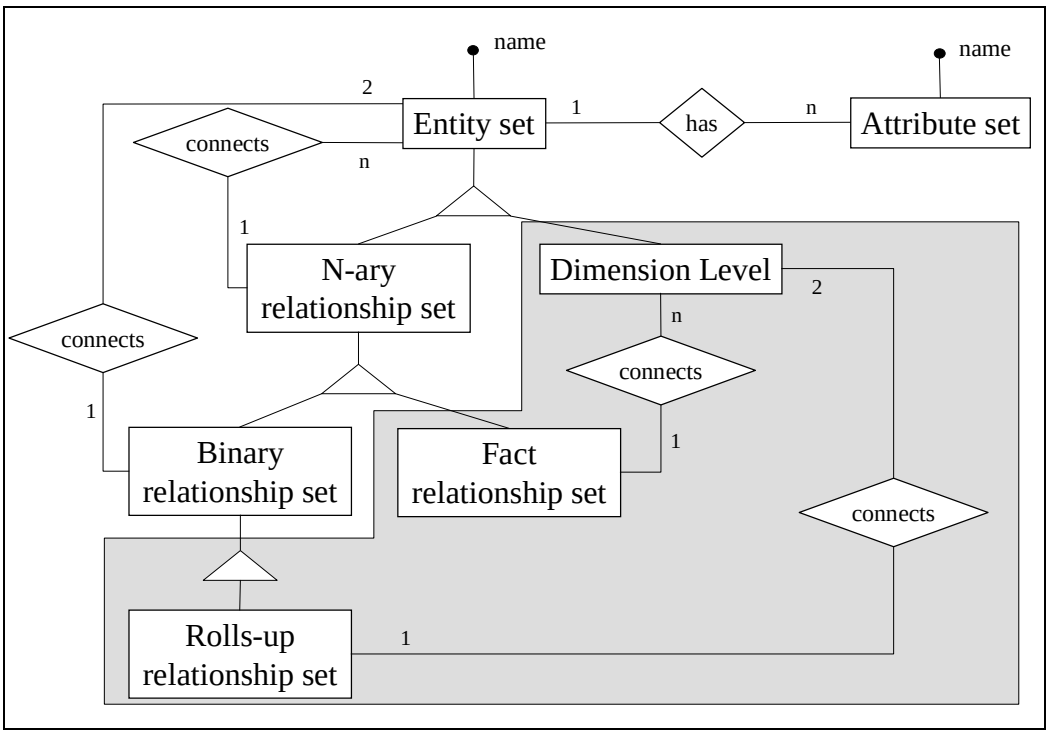

Fig. 6. The meta-model of the ME/R.

A *Dimension level set* is an entity set. A *Fact Relationship Set* is a n-ary relationship set that connects n dimension levels. Dimension attributes are modeled as elements of dimension level attribute set. Measures are modeled as elements of fact relationship attributes set. A *Rolls-up Relationship Set* is a relation between dimension levels. The relationship *connects* between a Rolls-up relationship set and a dimension level has a constraint: the graph constructed with the pairs of dimension levels connected by a Rolls-up relationship Set must be a DAG (Directed Acyclic Graph).

Graphic Notation and Example.

Each extension has a graphic notation related to. These notations can be seen in the Fig. 7.

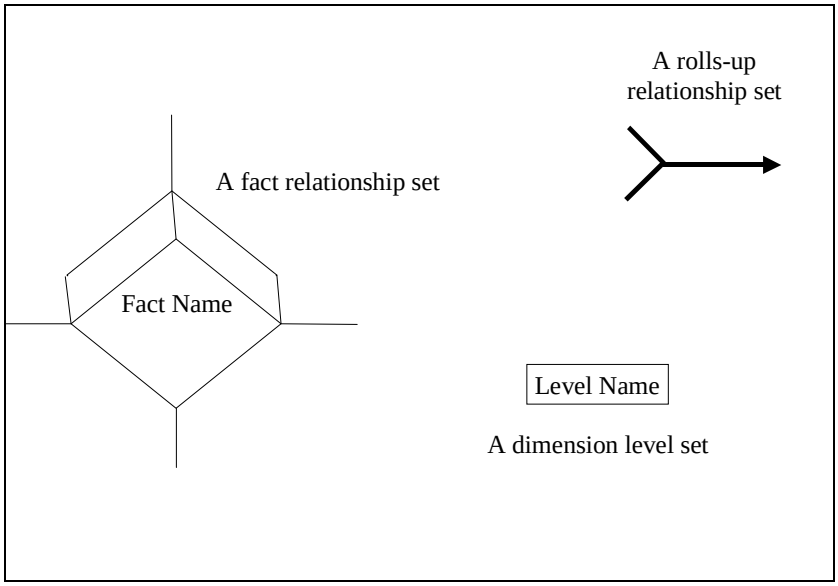

Fig. 7. ME/R graphic extensión to E/R.

These components can be combined freely as all constructions in E/R. There is only a constraint presented above: the graph of dimension levels cannot be cyclic.

The next figure presents the same example as in the previous model.

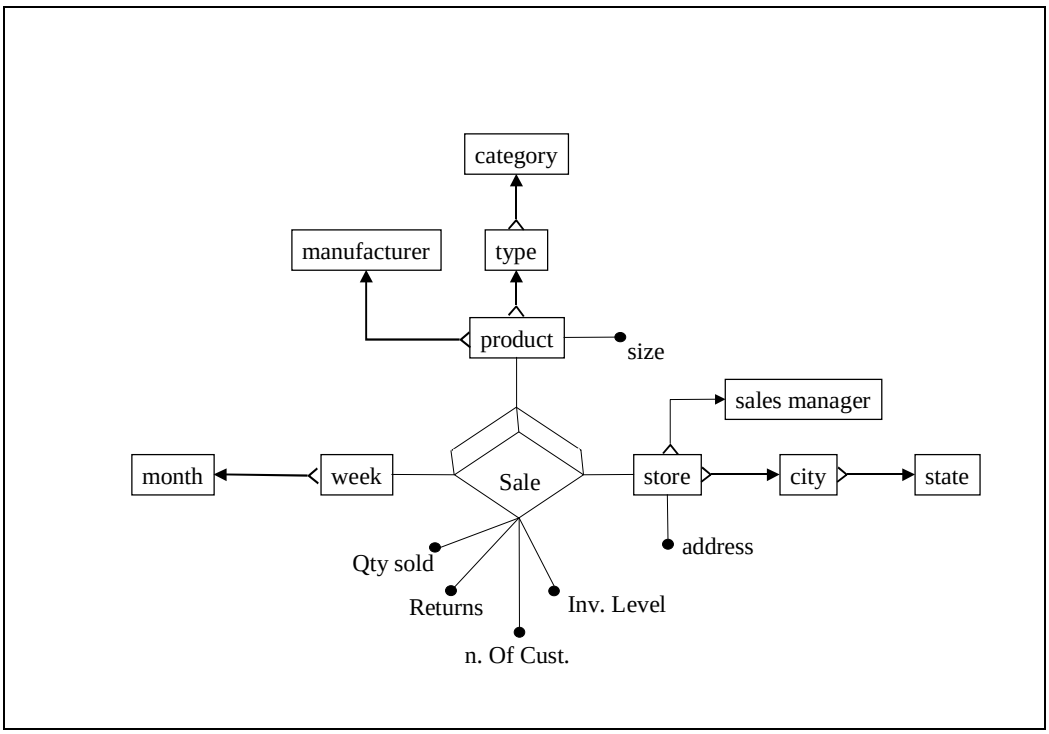

Fig. 8. The Sales Example.

The measures are attributes of a fact relationship set. Description attributes are attributes of dimension levels.

Conclusions.

The model has an advantage, with respect to the Golfarelli proposal: Is based on a standard model, in syntax and semantic. Therefore, the model should be friendly to persons that know E/R. Moreover, the experience in (semi) automatic treatment of E/R may be reused.

The drill-across paths rise in a natural way when two or more fact relationship sets are connected to levels in the same hierarchy.

However, there is no support for generic dimensionality and has the same drawbacks as E/R like no (wide accepted) modular representation.

Is not clear in the paper how the dimension level elements can be identified but an identifier for each one can be supposed.

The model of Cabibbo and Torlone can be saw as a more generic semantic for this model. To do this, it can be identified fact relationship set with fact tables dimension levels with the same name in the Cabibbo's model.

Franconi – Sattler. (DWQ)

"Foundations of Data Warehouse Quality" (DWQ) was a long-term research project funded by the European Commission under the ESPRIT program. The interest of the project goes from conceptual to physical aspects of Data Warehouse.

The Data Warehouse Conceptual Data Model (DWCDM) is its proposal for conceptual modeling.

The model has two languages:

- A graphical language based on E/R diagrams.
- A basic language based on Description Logics.

In [Fra99a], the authors present an overview of both languages and a sketch from a method to translate the graphical language in the basic language.

In [Fra99] the authors expose the fundamentals of they proposal and give a good description of the basic language.

The rest of the section is an attempt to explain this model based on these papers. It must be taking in account that there is no description of the graphical language and the basic languages in both papers are softly different.

Fundamentals.

All the assumptions made previously are valid in this model.

The first difference is the vision of a dimension: A dimension is an attribute domain potentially structured with multiple hierarchies. Therefore, *there is no structure to represent explicitly a dimension.* The dimensions are represented as relationship sets and its levels are represented as entity sets. Which relationship set is a dimension can be inferred from the aggregation notation.

The model is oriented to express the structure of aggregations. This idea involves abstract properties of aggregations, relationships between the aggregation and their components. However, the description of an aggregation not includes a specification of how its attribute values must be computed in function of the components attributes values.

When an expression like *average(age)* or *min(income)* appears in a diagram, it must be understood as an attribute name.

Graphical Language.

The graphical language is an extension of E/R diagrams[1], adapted to represent explicitly the aggregations.

An example proposed in [Fra99a] talks about a telephone company. In the Fig. 9 a diagram of the base data for a data warehouse for this company can be saw.

Which is the difference between this diagram and an E/R from the operational data? The logic of the construction is different. The Fig. 10 presents a diagram sketch that, possibly, it is closer to the operational data.

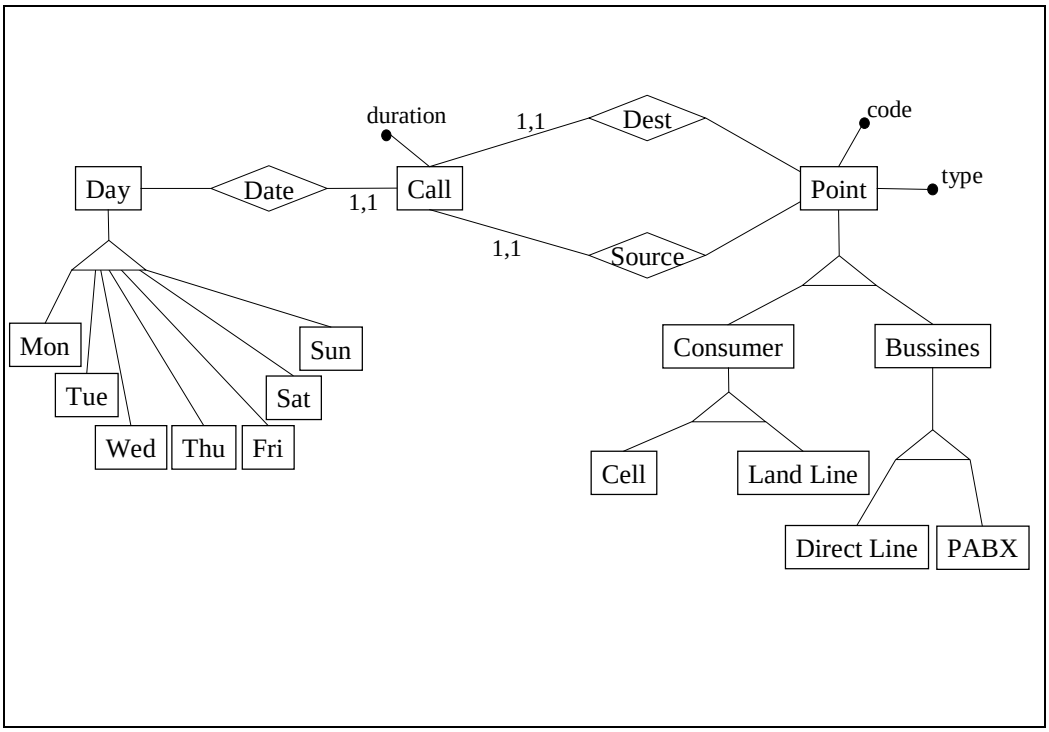

Fig. 9. Conceptual Data Warehouse Schema for base data.

[1] The syntax of the E/R diagrams was modified from the original papers to a more standard notation.

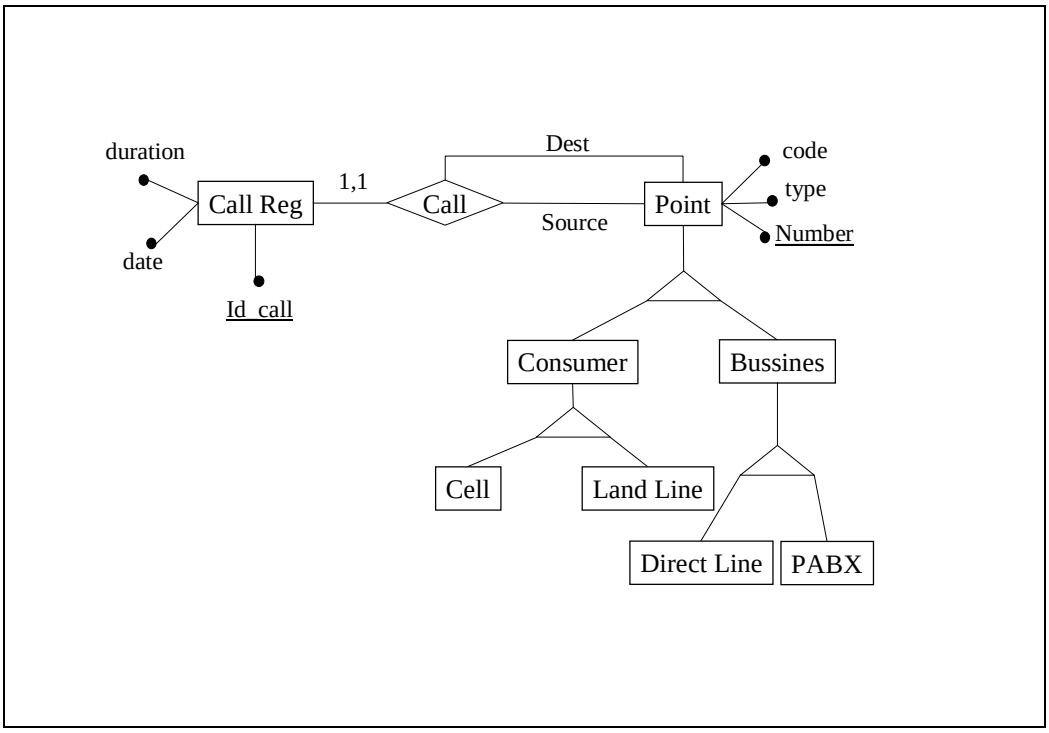

Fig. 10. A possible schema of operational data for the previous DW Schema.

There are at least two reports required by the users: one presents the average of duration by day and point type (cell, land line, etc.) and the other presents the average of duration by weekday and customer type (consumer or Business).

Average(Duration)	date			
Source (point type)	01/01/2000	01/03/2000	01/05/2000	01/07/2000
Direct Line	62,71483396	93,13603339	75,93415712	81,44172689
PABX	119,5707099	15,9396764	67,68244536	...

Fig. 11. Duration average by source point type and date

Average(Duration)	week day				
Source (type of customer)	1	2	3	4	5
Bussines	84,24889464	50,73514859	11,78495917	103,6093389	64,34583368
Consumer	71,40934886	61,32803122	115,6543695	60,18969521	...

Fig. 12. Duration average by source type of customer and web day (1=sunday).

In the Fig. 13 can be saw the conceptual data model for the first cube.

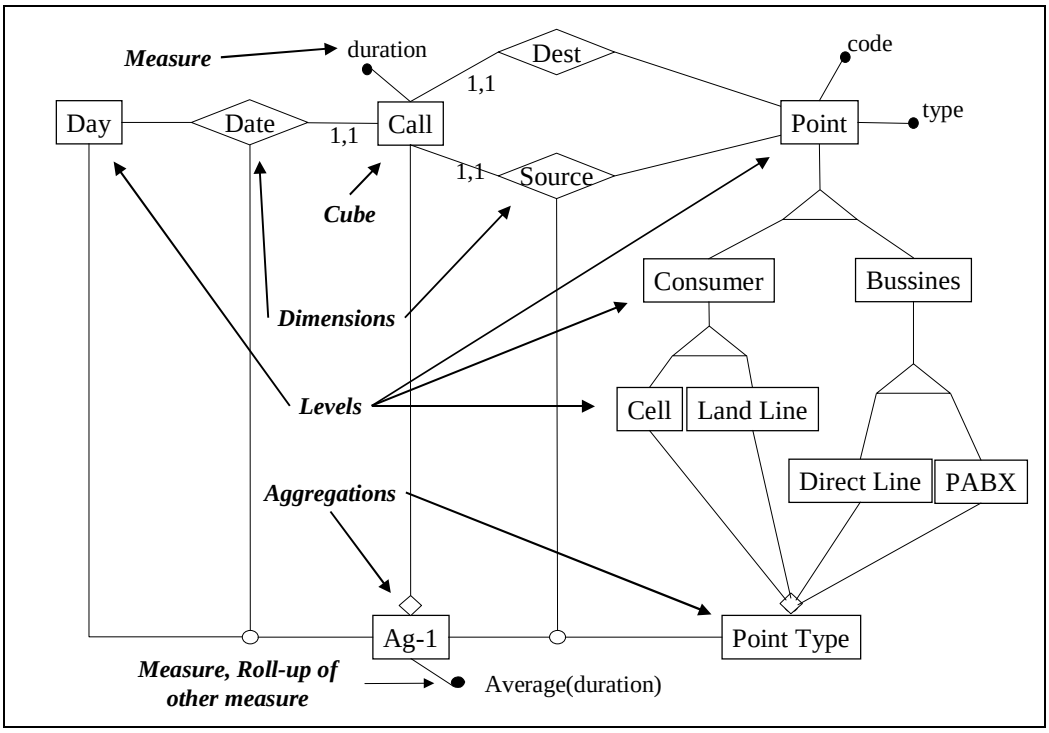

Fig. 13. Conceptual model for first cube.

The aggregation *Ag-1* aggregates *calls* depending on dimensions *Date* and *Source* at levels *Day* and *Point Type*. This last level is an aggregation of all point types. The diamond symbol can be read as a relationship set called *aggregates* with distinguished roles.

The semantic of this graphical model is defined in term of legal states of the multidimensional database. So, to interpret the diagrams, the authors choose the logical model proposed by Cabibbo and Torlone in [Cab98]. An instance of this logical model expresses a legal state for the multidimensional database. If the model is inconsistent then, there is no legal state to any database.

The Basic Language.

The graphical language can be mapped to this basic language. This language is based on Description Logics.

In order to simplify the reading, in the next section a minimal background on the theme will be exposed.

Next, some of the language will be presented.

Description Logics.

Description Logics are a family of formalisms for knowledge representation, which are based in the representation of Concepts and Roles.

A concept can be understood as a description for a set of elements or as a unary predicate.

A role can be understood as a description for a binary relation or as a binary predicate.

A description logic has a name according with its features. The Description Logic used by the authors is named ALCFI because is the basic (ALC) with “features” and “inverse roles”.

The syntax used in [Fra99a] is presented in the Fig. 14. In this syntax, C and D are concepts, R and S are roles and f and g are features.

A feature is a functional role.

C,D	→	A |	(concept name)
		☐ |	(top)
		☐ |	(bottom)
		¬C |	(complement)
		C☐D |	(conjunction)
		C☐D |	(disjunction)
		∀R.C |	(universal quantifier)
		∃R.C |	(existencial quantifier)
		f↑ |	(undefinedness)
		f:C |	(selection)
R,S	→	P |	(role name)
		f |	(feature)
		R^{-1} |	(inverse role)
		$R\|_C$ |	(restriction)
		R ° S	(role chain, composition)
f,g	→	p |	(feature name)
		f ° g	(feature chain, composition)

Fig. 14. Syntax for the ALCFI Description Logic.

$\square^{\square}$ = $\Delta^{\square}$

$\square^{\square}$	=	$\varnothing$
$(\neg C)^{\square}$	=	$\Delta^{\square} \setminus C^{\square}$
$(C \square D)^{\square}$	=	$C^{\square} \cap D^{\square}$
$(C \square D)^{\square}$	=	$C^{\square} \cup D^{\square}$
$(\forall R.C)^{\square}$	=	$\{i \in \Delta^{\square} \mid \forall j.(i,j) \in R^{\square} \Rightarrow j \in C^{\square}\}$
$(\exists R.C)^{\square}$	=	$\{i \in \Delta^{\square} \mid \exists j.(i,j) \in R^{\square} \wedge j \in C^{\square}\}$
$(f\uparrow)^{\square}$	=	$\Delta^{\square} \setminus dom(f^{\square})$
$(f:C)^{\square}$	=	$\{i \in dom(f^{\square}) \mid f^{\square}(i) \in C^{\square}\}$
$(R^{-1})^{\square}$	=	$\{(i,j) \in \Delta^{\square} \times \Delta^{\square} \mid (j,i) \in R^{\square}\}$
$(R \circ S)^{\square}$	=	$R^{\square} \circ S^{\square}$

Fig. 15. Extensional Semantic for ALCFI Description Logic

The semantic of Description Logic is defined by means of an interpretation $\square$. An interpretation $\square$ is a pair $(\Delta^{\square}, \cdot^{\square})$ where $\Delta^{\square}$ is a set of elements (domain of $\square$) and $\cdot^{\square}$ is a function which maps each concept in a subset of $\Delta^{\square}$ and each role in a subset of $\Delta^{\square} \times \Delta^{\square}$.

The semantic of ALCFI can be seen in Fig. 15.

For a better comprehension of the ALCFI, its equivalence with First Order Logic is exposed in Fig. 16. A concept C in Description Logic is equivalent to an open formula $F_C(y)$ with one free variable *y* and a Role R is equivalent to an open formula $F_R(y,x)$ with two free variables *y* and *x* ([Art99]).

$\square^{\square}$	~	**True**
$\square^{\square}$	~	**False**
$(\neg C)^{\square}$	~	$\neg F_c(y)$
$(C \square D)^{\square}$	~	$F_C(y) \wedge F_D(y)$
$(\exists R.C)^{\square}$	~	$\exists x.(F_R(y,x) \wedge F_C(x))$
$(\forall R.C)^{\square}$	~	$\forall x.(F_R(y,x) \Rightarrow F_C(x))$
$(f\uparrow)^{\square}$	~	$\neg\exists x.f(y,x)$
$(f:C)^{\square}$	~	$\exists x.(f(y,x) \wedge F_C(x))$
$(R^{-1})^{\square}$	~	$F_R(x,y)$
$(R \circ S)$	~	$\exists z.(F_R(y,z) \wedge F_S(z,x))$

Fig. 16. First Order Logic equivalence for ALCFI

The Translation of Extended E/R Diagrams.

The translation consist of constructing a knowledge base (or terminology) Σ, from a diagram D, where there is a concept name for each entity, aggregation, relationship or domain and there is a feature name for each relationship role or attribute. For each type of construction in the diagram, there are some rules that add appropriate terminological axioms to the terminology Σ.

As example, the following is part of the knowledge base for the example:

$Date \sqsubseteq \forall what : Call \sqsubseteq \forall when : Day$

$Source \sqsubseteq \forall what : Call \sqsubseteq \forall where : Point$

$Dest \sqsubseteq \forall what : Call \sqsubseteq \forall where : Point$

$Point \sqsubseteq Consumer \sqsubseteq Business$

$Consumer \sqsubseteq Point \sqsubseteq \neg Business$

$Business \sqsubseteq Point \sqsubseteq \neg Consumer$

The first axiom is the definition for the concept *Date*, which rises from the relationship Date in the diagram. The expression *what* in this axiom is a feature (functional role) and denotes the role between the entity Call and the relationship Date. The expression *when* denotes the role between the relationship Date and the entity Day. The expression *what:Call* is a concept that must be understand as the set of elements from the interpretation domain that are related with any Call by the role *what*. In terms of the extensional semantic, this can be thought as:

$(what:Call)^{\Box} = \{ i \in dom(what^{\Box}) \mid what^{\Box}(i) \in Call^{\Box} \}$

The expression *when:Day* is similar to the previous. So, the axiom say that the intersection of the concepts *what:Call* and *when:Day* subsumes the concept *Date*. This is considered as the primitive concept definition for the concept *Date*.

The rest of the axioms in the first group are similar to this.

The first axiom in the second group expresses that any instance of *Point* must be (primitive concept definition again) an instance of *Consumer* or a instance of *Business*. The second and third axioms of this group express that *Consumer* and *Business* conforms a partition of *Point*.

In [Hor99] and [Hor99a] an algorithm based on the tableau method that makes decidable the satisfaction of a terminology is proposed. This is used by Franconi and Sattler to decide the model consistency.

Conclusions.

The proposal is the most complete, at least, until today.

The graphical language does not support generic dimensionality and apparently, there is no a clear way to see the paths on dimension hierarchies. The last remark is evident in the Fig. 13. The entity *Point Type* is a level from the dimension *Source* (and *Dest* too) because is involved by the aggregation Ag-1. However, *Point Type* is a level higher in the dimension hierarchies to the levels *Cell, Land Line, Direct Line and PABX*, but independent from *Consumer and Business*.

There is not any language to express general constraints, but it is possible to suppose that these constraints can be expressed in Description Logics.

Other Related Works.

The Design Methodologies are a target in itself. However, all multidimensional models support different methodologies, in spite of some of them were designed very close of a particular methodology ([Gol99], [Gol98], [Cab98]).

Summarizability is really a topic for conceptual modeling. Are all roll-up legal in any cube? It is clear that the answer to this question is not. Some of this is explained in [Kim96] as additive measures, but a more clear and general exposition over this is [Len97].

Lehner and other expose some consideration and definitions on Multidimensional Normal Forms in [Leh98]. The authors define that in multidimensional databases the normal forms must be oriented to two targets: ensure summarizability and reduce the sparseness of the data cube.

In the Description Logics appendix, there are many references on the topic.

There is no found any work, which explicitly says something about constraints on multidimensional models. However, it is possible that Description Logics can express some constraints. In fact, in [Fra99a], the language allows specify some cardinality constraints. In [Hor99], the DL language supports more cardinality constraints, but, in anywhere there is an explanation on how express general constraints in DLs. It is possible that in a DL with this feature the satisfaction not be decidable.

A Framework for the Comparison of Multidimensional Models.

To compare conceptual multidimensional models, the following topics must be taken in account:

- **Model Type**. If the model is query oriented or logical or conceptual.
- **Relational Independence**. The model must be multidimensional, not an implementation of a multidimensional model in Relational Model or other data model with a lower conceptual level. This problem makes difficult the implementation of the structures in a real multidimensional system.
- **Definition of dimensional hierarchies**. The model must allow the designer to specify some dimensional hierarchies. Each model has different orientations in this topic.
- **Symmetry between measures and dimensions**. This is a condition for OLAP imposed by Codd in its definition in [Cod93]. A few models treat to this topic.
- **Summarizability**. Only one of the studied model has some support for it.

- **Constraint Language**. A data model must allow to define data structures and must give a precise semantic description providing a language for integrity constraints.

Many other topics can be taken in consideration but these ones seems to concentrate in Conceptual Modeling or semantic of the multidimensional modeling.

Model	Model type	Relational Independence	Definition of dim. hierarchies	Symmetry between Dim. And Measures	Summarizability	Constraint Language
Li	query	no	relations	no	No	no
Agrawal	query	yes	by operations	by operations	No	no
Gyssens	query	no	relations	by operations	No	no explicitly
Cabbibo	logical	yes	structural	by operations	No	no explicitly
Lehner	logical	yes	structural	no	Yes	no
Sapia	conceptual	yes (ER/extension)	structural	no	No	no explicitly
Golfarelli	conceptual	yes	structural	no	Yes	no explicitly
Franconi	conceptual	yes (ER/extension)	structural (but no explicitly)	no	No	no explicitly

Fig. 17. A comparison between the multidimensional models.

In the Fig. 17, in the column *constraint language*, the value *no explicitly* expresses that in spite of there is not a definition of a constraint language in the model, there are elements to think that may be possible a clear definition. When the model has a calculus as a query language, it can be used to express some constraints with a high expressiveness. For Franconi's model, some constraints may be expressed in Description Logics or similar.

In the column *definition of dimensional hierarchies* the Franconi's model has a remark: is structural but not in an explicit way. This remark appears because the model has no explicit structure for the dimensional hierarchies but it can be read from the diagrams as ISA hierarchies and simple aggregations.

Conclusions.

The most important multidimensional models were briefly described. There are only three conceptual models. However, all of the models have worth noting properties.

Li ([Li96]) and Gyssens ([Gys97]) models are valuable as Star Schema formalizations.

Agrawal's model ([Agr97]) introduces the notion of Boolean cube, which can be used to manage the symmetry between dimensions and measures.

Cabibbo´s model ([Cab97] , [Cab98]) has a clear and complete formalization of basic multidimensional structures and its calculus introduce a clear way to consult these structures.

Lehner's model ([Leh98a]) has a formal characterization of multidimensional domains (instances) introducing clear definitions of data contexts. In another paper ([Leh98]), there is a definition of Normal Forms over this model.

Sapia´s model ([Sap99a]) has a clear graphical notation based on ER diagrams.

Golfarelli´s model ([Gol99], [Gol98]) has a definition of a method to express drill-across, which seems to be adequate for multidimensional schema integration. Also has a general specification for summarizability properties.

Franconi´s model ([Fra99], [Fra99a]) is based in Description Logics, which allows the implementation of decision procedures over the consistency of the model. Moreover, its graphical language extends ER diagrams with multidimensional and simple aggregations.

A lack in all models is an explicit way to express general constraints. This feature can help the designer with the specification accuracy.

Bibliography.

[Agr97] Agrawal, R. Gupta, A. Sarawagi, S.:*"Modelling Multidimensional Databases"*, ICDE, 1997.

[Art99] Artale, A. Franconi, E:*"Introducing Temporal Descriptions Logics."*, ???? DWQ Internet Site, 1999.

[Cab97] Cabibbo,L. Torlone, R.:*"Querying Multidimensional Databases"*, SEDB, 1997.

[Cab98] Cabibbo, L. Torlone, R.:*"A Logical Approach to Multidimensional Databases"*, '', Sixt Int. Conference on Extending Database Technology (EDBT '98), Springer-Verlag, 1998.

[Cab99] Cabibbo, L. Torlone, R.:*"Data Independence in OLAP Systems"*, Dipartimento de Informatica e Automazione, Università di Roma Tre, 1999.

[Cal98] Calvanese, D. Lenzerini, M. Nardi, D.:*"Description Logics for Conceptual Data Modeling."*, Logics for Database and Information Systems., J. Chomicki and G. Saake eds., 1998.

[Car97] Carpani, F. Goyoaga, J. Fernandez, L.:*"Modelos Multidimensionales"*, IV Jornadas de Informática e Investigación Operativa. Instituto de Computación de la Facultad de Ingeniería. Monteviedo, Uruguay., 1997.

[Cod93] Codd, E.F. Codd, S.B. Salley, C.T.:*"Providing OLAP to user-analysts. An IT mandate."*, Technical Report., E.F. Codd and Associates., 1993.

[Fra99] Franconi, E. Sattler, U.*"A Data Warehouse Conceptual Data Model for Multidimensional Aggregation: A Preliminary Report."*, Technical Report., DWQ, 1999.

[Fra99a] Franconi, E. Sattler, U.:*"A Data Warehouse Conceptual Data Model for Multidimensional Aggregation."*, Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW'99), S. Gatziu, M.Jeusfeld, M.Staudt, Y. Vassiliou, Eds., Jun. 1999.

[Gol98] Golfarelli, M. Rizzi, S.:*"A Methodological Framework for Data Warehouse Design"*, First International Workshop on Data Warehousing and OLAP (DOLAP), ACM, Nov. 1998.

[Gol98a] Golfarelli, M. Maio, D. Rizzi, S.:*"Conceptual Design of Data Warehouses from E/R Schemes."*, International Conference on Systems Science, Hawaii, IEEE, Ene. 1998.

[Gol99] Golfarelli, M. Rizzi, S.:*"Designing the Data Warehouse: Key Steps and Crucial Issues."*, Journal of Computer Science and Information Management. Vol. 2. N. 3, Maximilian Press Publisher, 1999.

[Gru99] Grumback, S. Rafanelli, M. Tininini, L.:*"Querying Agregate Data"*, Proceedings Conference on Principles of Database Systems (PODS '99), Philadelphia, ACM, 1999.

[Gys97] Gyssens, M. Lakshmanan, L.:*"A Foundation for Multi-Dimensional Databases",* VLDB, Athens, Greece, 1997.

[Hor99] Horrocks, I. Sattler, U. Tobies, S.:*"Practical Reasoning for Expressive Description Logics.",* DWQ Internet Site., 1999.

[Hor99a] Horrocks, I. Sattler, U. Tobies, S.:*"Practical Reasoning for Description Logics with Functional Restrictions, Inverse Roles and Transitive Roles, and Role Hierarchies.",* DWQ Internet Site, 1999.

[Jar99] Jarke, M. Lenzerini, M. Vassiliou, Y. Vassiliadis, P.:*"Fundamentals of Data Warehouses",* , Springer-Verlag, 1999.

[Ken96a] Kenan Technologies:*"An Introduction to Multidimensional Databases.",* White Paper, Kenan Technologies, 1996.

[Kim96] Kimball, Raplh.:*"The Datawarehouse Toolkit.",* Book., John Wiley & Son, Inc., 1996.

[Leh98] Lehner, W. Albrecht, J. Wedekind, H.:*"Normal Forms for Multidimensional Databases.",* , 1998.

[Leh98a] Lehner, W.:*"Modeling Large Scale Olap Scenarios.",* Proc. EDBT '98. Valencia. España., http://www6.informatik.uni-erlangen.de/dept/staff/lehner-publications.html, 1998.

[Len97] Lenz, H. Shoshani, A.:*"Summarizability in OLAP and Statistical Databases.",* 9th Int. Conf. on Statistical and Scientific Databases , 1997.

[Li96] Li, C. Wang, X.S.:*"A Data Model for Supporting On-Line Analytical Processing",* Conf. on Information and Knowledg Management, Nov. 1996.

[Lou92] Loucopoulus, P. Zicari, R.:*"Conceptual Modeling Databases and CASE",* , John Wiley & Sons, Inc., 1992.

[Man99] Mangisengi, O. Tjoa, A M.:*"A Multidimensional Modeling Approach for OLAP within the Framework of the Relational Model based on Quotient Relations",* Proceedings Conference on Principles of Database Systems. (PODS ´99) Philadelphia, ACM, 1999.

[Sap99] Sapia, C. Blaschka, M. Höfling, G.:*"An Overview of Multidimensinal Data Models for OLAP",* Technical Report., http://www.forwiss.de/~system42/publications, Feb. 1999.

[Sap99a] Sapia, C. Blaschka, M. Höfling, G. Dinter, B.:*"Extending the E/R Model for the Multidimensional Paradigm.",* Advances in Database Technologies. LNCS Vol 1552, Springer-Verlag. 1999.

[Sap99b] Sapia, C. Blaschka, M. Höfling, G.:*"GraMMi: Using a Standard Repository Management System to Build a Generic Graphical Modeling Tool",* ???, ???, +Ju. 1999.

[Tho97] Thomsen, E.:*"OLAP Solutions. Building Multidimensional Information.",* Book, John Wiley & Sons, Inc. , 1997.