

Chapter 10

A Multidimensional Data Warehouse Development Methodology

José María Cavero and Esperanza Marcos
Escuela Superior de Ciencias Experimentales e Ingeniería
Universidad Rey Juan Carlos, Madrid, Spain

Carmen Costilla
Dpto. de Ingeniería de Sistemas Telemáticos
ETS de Ingenieros de Telecomunicación
Universidad Politécnica de Madrid, Spain

Mario G. Piattini
Escuela Superior de Informática
Universidad de Castilla-La Mancha, Spain

Adolfo Sánchez
Cronos Ibérica, S.A., Madrid, Spain

Data warehousing and On-Line Analytical Processing (OLAP) technologies have become growing interest areas in latest years. Specific issues, such as conceptual modeling, schemes translation from operational systems, physical design, etc... have been widely treated. A few methodologies covering the entire development cycle have also been proposed, but there is still not a general accepted complete methodology for data warehouse design. In this work we present a multidimensional data warehouse development methodology integrated within a traditional software development methodology.

1 INTRODUCTION

The term *data warehouse* first appeared in [9], defined as a “*subject oriented, integrated, non-volatile, and time variant collection of data in support of management’s decisions*”. Data warehouses are closely related to OLAP technology, first introduced by Dr. E.F. Codd in 1993 [3] to characterize the requirements of aggregation, consolidation, view production, formulae application and data synthesis in many dimensions. A data warehouse is a repository of information mainly coming from On-Line Transactional Processing (OLTP) systems that provides data for analytical processing and decision support.

Multidimensional view of data is not anything new (in fact, is a very old concept): managers observe the evolution of interesting data organized in dimensions, such as products, clients, promotions, sell points, and, of course, time. The need of having simply and rapidly every historical information of operational systems has pushed to companies to look for new ways of structuring and accessing their data, for having advantage over their competitors. There is an agreement in that traditional database systems are not appropriate for multidimensional data analysis. Traditional OLTP systems are optimized for providing high performance in processing a lot of concurrent transactions. These transactions usually affect to a very few records. Meanwhile, multidimensional systems have to answer to complex queries (sometimes unpredictable) that need a huge number of records [1]. In fact, as Ralph Kimball points [10], OLTP is profoundly different from dimensional data warehousing in their users, their data content and structures, their hardware and software, their administration and management, and their daily rhythms.

Due that OLTP and OLAP environments are profoundly different, the techniques used for operational database design are inappropriate for data warehouse design [10, 11].

The development of a data warehouse needs the integration of data mainly from legacy systems. The process of developing a data warehouse is, like any other task that implies some kind of preexisting resources integration, profoundly complex. This process is “*labor-intensive, error-prone, and generally frustrating, leading a number of warehousing projects to be abandoned mid-way through development*” [14].

To this respect, in latest years, there have been many proposals for some particular aspects involved in the data warehouse design process. However, although many solutions have been developed “*for interesting subproblems like handling multidimensional data as typical requirement for data warehouses, view maintenance for aggregated data, data integration etc., combining these partial and often very abstract and formal solutions to an overall design methodology and warehousing strategy is still left over to the practitioners*” [6].

Despite the obvious importance of having a methodological support for the development of OLAP systems, the design process has received very little attention of the scientific community and product providers. Models usually utilized for operational data base design (like E/R model) shouldn't be used without further ado for analytical environment design. Attending only to technical reasons, databases obtained from E/R models are inappropriate for decision support systems, in which query performance and data loading (including incremental loading) are important [10]. Multidimensional paradigm should be used not only in data base queries, but also during its design and maintenance. *“To use the multidimensional paradigm during all development phases it is necessary to define dedicated conceptual, logical and physical data models for the paradigm and to develop a sound methodology which gives guidelines how to create and transform these models during the development process”* [5]. In [15] authors claim for data warehouse design methodologies and tools *“with the appropriate support for aggregation hierarchies, mapping between the multidimensional and the relational models, and cost models for partitioning and aggregation that can be used from the early design stages”*.

There are a few proposals for data warehouse design [9, 4, 1, 7, 8]. They are usually very focused from its initial phases on the relational model. Most of them are incomplete, or starts from ideal assumptions, such as that every information is included in a corporate E/R scheme. There are also many other partial proposals, focused on issues such as model translations, view materialization, indexes, etc.

The problem with all these works is that they propose to use a new different methodology for data warehouse design, so organizations must use at least two totally different methodologies: one for OLTP environments and one for OLAP environments. We think it is better to integrate data warehouse design in existing methodologies, modifying and adding new activities, so that training and learning curve for data warehouse design was less difficult.

In this chapter we present a methodology for the development of multidimensional data warehouses. The methodology is based on an existing traditional methodology, and differs from the traditional one in three of its processes: Analysis, Design and Construction. It allows the creation of multidimensional or relational databases, and it is supported by a CASE tool.

The rest of the chapter is organized as follows: in section 2 we briefly present IDEA, the multidimensional model used in the methodology. Section 3 outlines a methodology for the development of multidimensional data warehouses. In Section 4 a modeling example is developed. Finally, in Section 5 we present some conclusions.

2 MULTIDIMENSIONAL CONCEPTUAL MODEL

IDEA [13] is the multidimensional conceptual data model used for conceptual modeling of multidimensional data warehouses. As every data model, it consists of a static part, which deals with data structures, and a dynamic part, which deals with data manipulation.

The elements that define the storage structures are described in the static part of IDEA. Due to IDEA is an analytical multidimensional data model, the main purpose is to serve as a basis for data analysis. Next, static part of IDEA is briefly and informally described.

Idea establishes a classification of non-exclusive kinds of *Domains*:

- *Dimension Domains*: a Dimension Domain is used to represent dimension values. *Dimension Attributes* are defined on a Dimension Domain. There are two kinds of Dimension Domains:
 - *OID Domains*: this kind of domain is used to represent objects (products, employees...). By means of this domain, operational original data can be included in a multidimensional schema, and a link between elementary and multidimensional databases can be established.
 - *Category Domain*: Values of Category Domains are qualitative, usually extensionally defined. Not only Dimension Attributes can be of Category Domain. Also *Description Attributes* (attributes used to describe a Dimension Attribute) can be defined on a Category Domain.
- *Synthesis Domains*: a Synthesis Domain is used to represent *Synthesis Attributes* of *Fact Schemas* (see below). There are two kinds of Synthesis Domains:
 - *Quantity Domains*: are the most common Domains used in Synthesis Attributes. They are intensionally defined, and mathematical operations can be applied on them.
 - *Boolean Domains*: Boolean Domains are used to indicate existence or not of information for a given Sub-cell of the multidimensional space.
- *Description Domains*: a Description Domain is used to represent Description Attributes, which represent complementary information about Dimension Attributes. Description Domains can be the already defined Category or Quantity Domains.

Aggregations, Hierarchies and Sub-hierarchies can be defined on Domains. An Aggregation consists of an *Aggregation Function* and two Dimension Domains, being one of them the origin and the other the destination. The Aggregation Function is a mathematical function that makes a correspondence between both domains. A Hierarchy is a set of Domain Aggregations. It is graphically represented by a graph in which each node represents a Dimension Domain, and each arc represents an Aggregation Function. A Sub-hierarchy is a set of Domain Aggrega-

tions contained in a Hierarchy. That is, a Domain Sub-hierarchy is a subgraph of a Hierarchy graph. On a Domain Sub-Hierarchy can be defined an *Attribute Sub-Hierarchy*, which can be the basis of a *Dimension*, as we will see later. Figure 1 shows an example of Domain Hierarchy.

A *Fact Schema* describes a n-dimensional space related to a fact of interest for analytical processing. A Fact Schema consists of a set of *Dimensions*, the *Dimension Attributes* associated to each Dimension, a *Cell Structure* and, optionally, a *Predicate*.

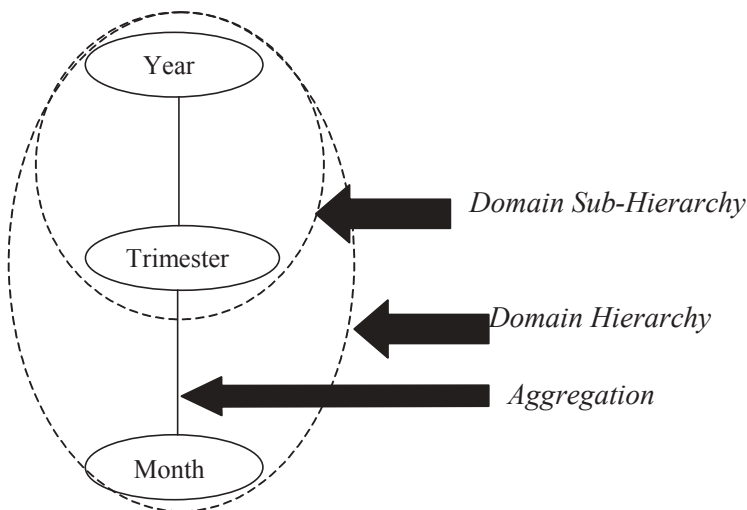
A Dimension is defined on a Dimension Domain, and is defined by a Dimension Attribute that could be (or not) the root of an Attribute Sub-Hierarchy.

Every Cell Structure is composed of sub-structures named *Sub-cell Structures* and *Methods* applied to them. Each Sub-cell Structure consists of one *Synthesis Attribute* (defined on a Synthesis Domain), and a set of *Synthesis Functions* that represents how operational data have been processed to obtain summarized data (for example, sum, frequency, average, maximum, minimum, ...). Synthesis Functions and Methods can return more than one value.

Figure 2 shows an example (graphical notation is based on [8]) that represents *Sum* and *Average of Units made*, *Sum of Income* and *Average price* along time (year), country and product.

Till now, we have just described the static part of the IDEA Conceptual Model, that is, its structural part at intensional level. The extensional level of this model, that is, the *Cube*, concerns to content of the n-dimensional space defined on the Fact Schema in a certain moment (n is the number of dimensions).

Figure 1. Example of Domain Aggregation, Hierarchy and Sub-hierarchy



For each Sub-cell: if the Synthesis Attribute is defined on a Boolean Domain, then it must not exist any Synthesis Function, so the content of the Sub-cell should be “True” or “False”. If the Synthesis Attribute is defined on a Quantity Domain and it does not exist any Synthesis Function, the Sub-cell should contain only one data, coming from operational original source, so no synthesis has been applied on them. If there are Synthesis Functions, each Sub-cell should contain one value for each function (or more than one in the case of functions that return more than one value, such as maximum(n), minimum(n), and so on).

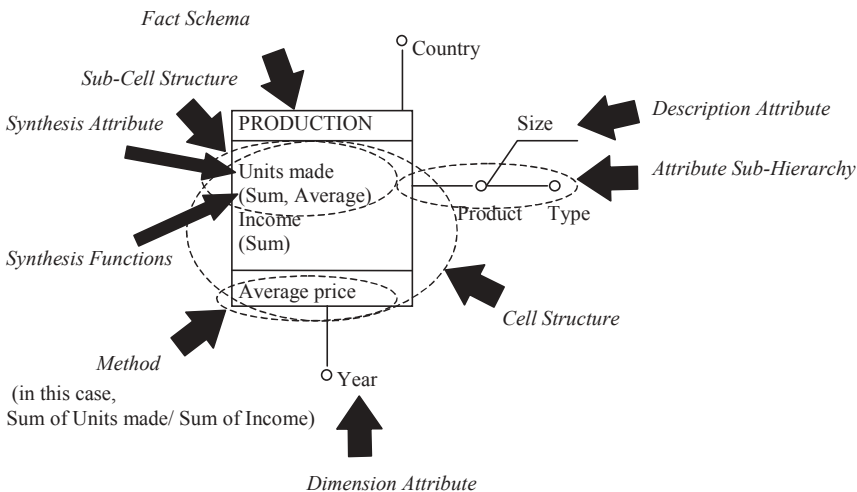
Figure 3 shows a Cube of the example of Figure 2. A Cell should be identified by its Dimensions, and should contain Values.

3 METHODOLOGY

MIDEA, a multidimensional database development methodology [2] consists of conceptual, logical and physical design. In conceptual design, IDEA is used as multidimensional data model. In the logical and physical design, multidimensional or relational logical models can be used. The methodology is supported by a CASE tool [12] that allows the translation of conceptual IDEA schemes into logical schemes based on models supported by multidimensional or relational products. Figure 4 shows a tool prototype window.

IDEA multidimensional conceptual model is used to understand and represent analytical users requirements in a similar manner than ER model is used to interact with microdata (operational) users. Preexistent OLTP system data scheme and requirements obtained from analytical data users are the main inputs to the construction of IDEA multidimensional conceptual scheme.

Figure 2. Graphical representation of a Fact Schema



Next step is to transform, using a set of methodological rules, each conceptual schema previously defined into a logical scheme based on the model of each product (pure multidimensional or relational with multidimensional issues). The most usual procedure in current projects is to translate directly relational scheme into multidimensional schemes supported by OLAP tools.

The MIDEA approach allows reverse engineering of existing specific multidimensional schemes into IDEA conceptual schemes. These schemes could be checked against OLAP users requirements to verify that current data warehouse satisfies them.

In the same way, is possible to create and/or modify elementary ER conceptual schemes using a set of rules proposed in the methodology to satisfy analytical users requirements.

The methodology uses as reference framework the Spanish Public Methodology METRICA version 3 proposal (MV3), which is similar to British SSADM or French Merise. MV3 processes considered are those on which the data warehouse development has more influence, that is, Information System Analysis, Design and Construction (ASI, DSI and CSI). The new processes, modified from the MV3 proposal, have been named as ASI-MD (MultiDimensional), DSI-MD and CSI-MD. Of course, considering only these three processes doesn't mean that the others processes shouldn't be taken into account on a data warehouse development, but we have considered that the differences shouldn't be significant enough with respect to any other information system development.

Every process is divided into activities and every activity is divided into tasks. The methodology is fundamentally focused on data modeling, and does not take into consideration the functional aspects of the development. Therefore, extraction, translation and load functions are not fully considered.

Figure 3 Cube corresponding to Figure 2

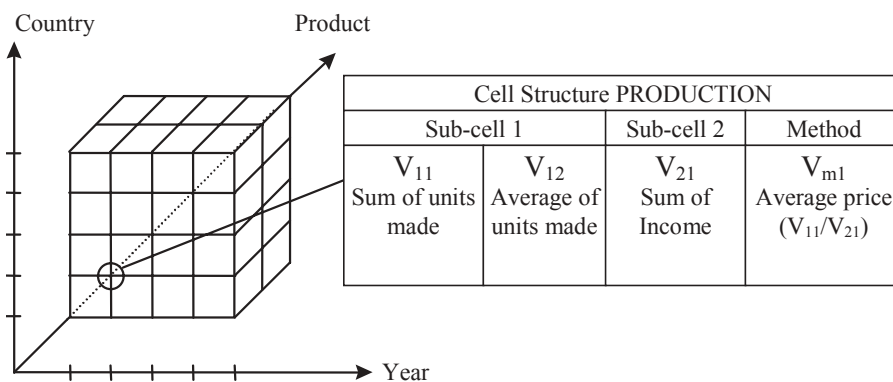


Figure 5 shows an overview of the methodology, showing the scope of its three processes, ASI-MD, DSI-MD and CSI-MD.

Bellow we offer a general overview of the three processes of the MIDEA.

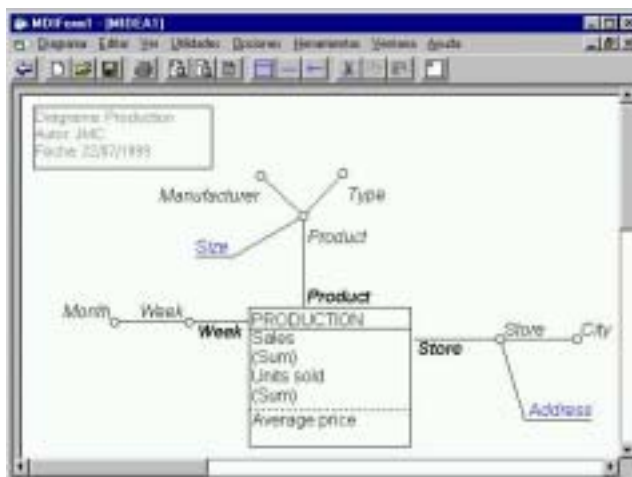
The main purpose of ASI-MD process is to obtain a detailed specification of the data warehouse. This specification has to satisfy the information needs of users (business analysts, specialists...) and serve as a basis for the design.

Information gathering is mainly done in ASI-MD 2 activity, “*Obtaining Detailed Requirements*”. We use as starting point the General Requirements Catalogue and high level schemes. Such Catalogue consists of a set of generic and user oriented requirements. These products should be refined with users by means of work sessions. In this way, data warehouse requirements are in more detail specified. In addition, data warehouse non-functional requirements must be identified (constraints that have to be accomplished related to performance, security, etc). The purpose of activity ASI-MD 2 is to define a detailed and validated Requirements Catalogue, which serve as a basis to test correctness of schemes obtained in activity ASI-MD 3, “*Data Warehouse Conceptual Modeling*”. This activity contains a verification and validation task in which the schema must be revised to guarantee that it is complete, complied with the Requirements Catalogue, and met some predetermined quality criteria.

Participation of users is essential to this process, because it constitutes a warranty that requirements initially identified have been understood and incorporated into the system and, therefore, that it will be accepted.

In DSI-MD process are described the necessary activities to obtain the data warehouse design. The process start from the Software Requirements Specifica-

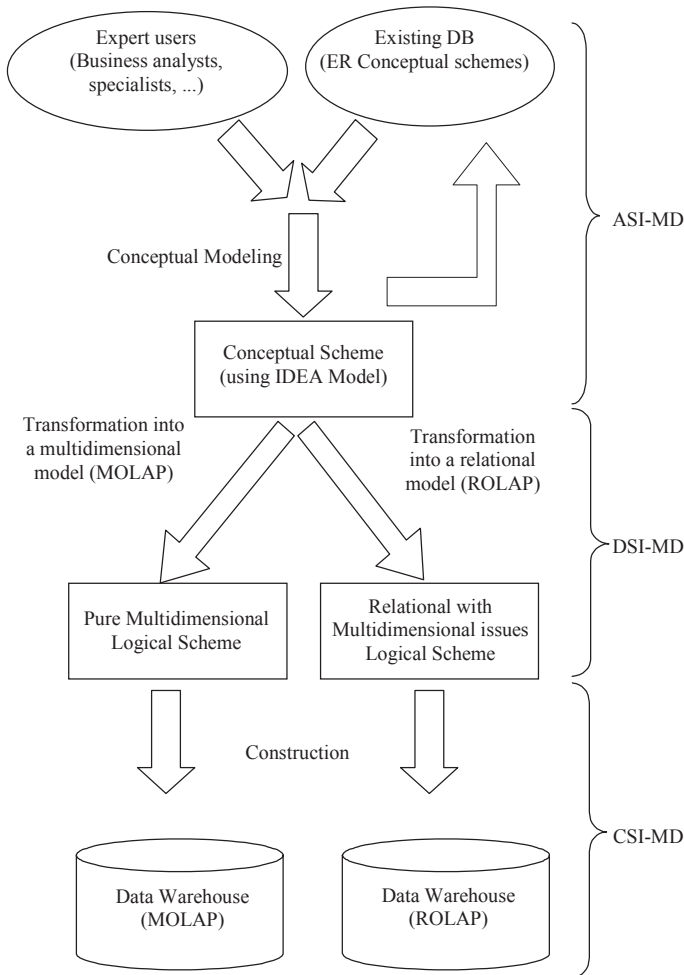
Figure 4 IDEA-DWCASE Tool



tion obtained in ASI-MD process. The design process describes “how” to implement the elements detected in the Analysis process. In this process the following tests are designed: query tests, query consistency tests and data warehouse acceptance tests.

Due to the non-existence of a standard nor commonly accepted multidimensional logical model, the data design process should be done in one step, from conceptual to logical specific model (i.e., product dependent). This “one-step” logical design is carried out in activity DSI-MD 2 in case of MOLAP (Multidimensional OLAP) systems, or in DSI-MD 3 for ROLAP (Relational OLAP) systems. Previously (in the activity DSI-MD 1), the appropriate technology (ROLAP or MOLAP) and product must be chosen. In this process there are three activities

Figure 5 Methodology overview



focused on the Physical Design. The purpose of them is to obtain and tune the physical design starting from the logical design obtained in previous activities (DSI-MD 2 and DSI-MD 3).

Finally, the main purposes of CSI-MD process are codification and test of data warehouse starting from the design specification obtained in DSI-MD process. Tests made during this process are focused on queries and queries consistency. Acceptance tests will be carried out during system implantation.

4 EXAMPLE

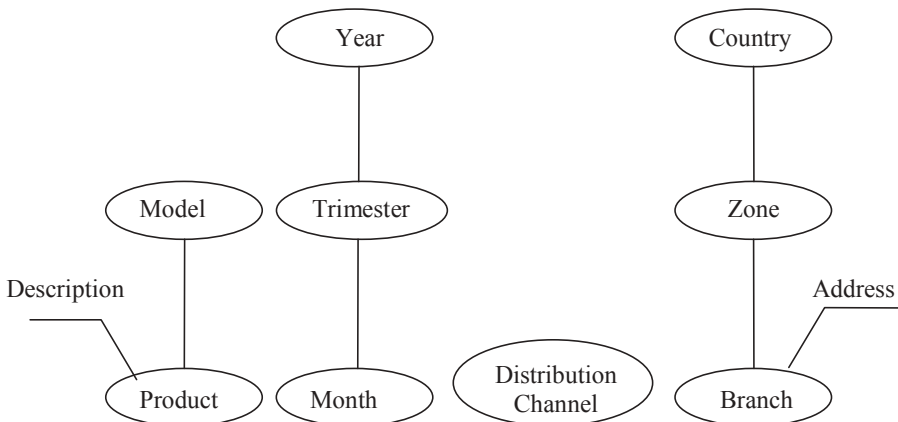
In the following sections, we will see a small example. First, we will obtain the IDEA conceptual schema. Next, we will translate it to specific relational and multidimensional schemes. Conceptual schema should be obtained in ASI-MD 3 activity, “*Data warehouse Conceptual Modeling*”. Translation into logical schemes should be done in activities DSI-MD 2, “*Multidimensional Logical Design*” (in the case of MOLAP implementation) or DSI-MD 3, “*Multidimensional relational logical design*” (in the case of ROLAP implementation).

4.1 Conceptual Modeling

Let’s suppose that we are modeling the results obtained by the sales of some products by a company. We are interested in analyzing the products sold, the date, the branch office, and the distribution channel used for the sale.

The purpose of the first tasks of ASI-MD 3 activity is to obtain a preliminary structure of the conceptual schema. So, in those tasks we should detect the four dimensions and a general overview of the dimensioned facts (incomes, expenditures, and benefits). Next, is important to make a detailed study of the dimensions

Figure 6. Dimensions detected



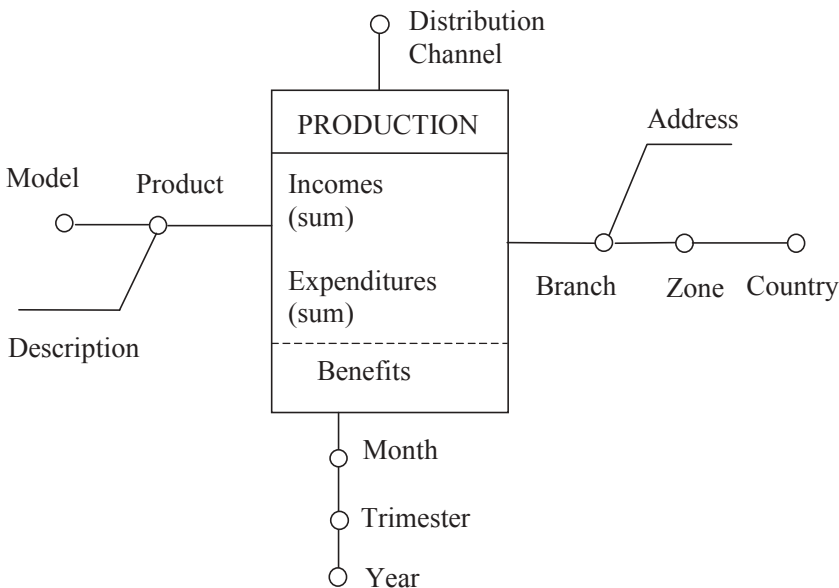
(perhaps new dimensions could be detected). We obtain then a detailed description of Hierarchies, Dimension Attributes, and Description Attributes. Aggregation Functions between Dimension Domains should also be defined. Figure 6 shows the Dimensions detected. Two of them are very common: time (in this case, Month-Trimester-Year) and geography (Branch-Zone-Country). There are, also, two Description Attributes, one of them associated to the Branch Dimension Attribute (which should contain the Branch Address), and the other associated to the Product Dimension Attribute (containing a Description of such product). As we can see, one of the Dimensions (Distribution Channel) has only one Dimension Attribute.

The next step is to detail the Fact Schemas. We must specify Synthesis Attributes, Synthesis Functions applied on them, and Methods. In this case, we have two Synthesis Attributes (Incomes and Expenditures). Aggregation Function in both cases is Sum. Besides, we have a Method (Benefits), defined by Sum of Incomes minus Sum of Expenditures. Figure 7 shows the resulting conceptual schema.

4.2 Logical Design: Relational

In DSI-MD process transformation from conceptual into logical multidimensional or relational schema is carried out. In the relational case, conceptual schema is translated into a star schema [10, 11]. Every Dimension is transformed into a Dimension Table. Physical Design should be done taking into account the charac-

Figure 7. Conceptual Schema



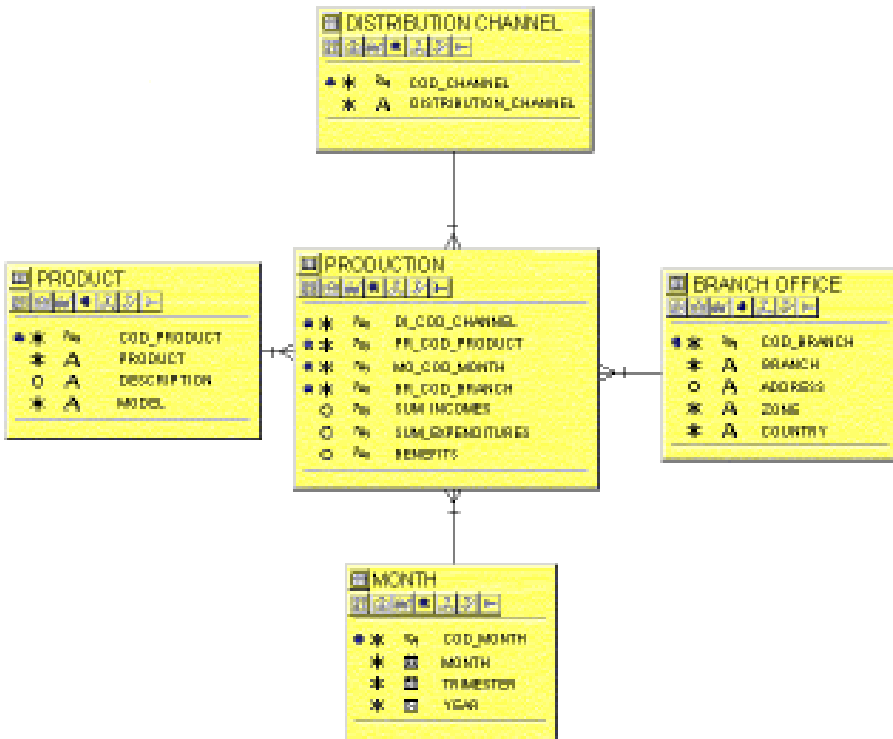
teristics of the specific relational product, such as bitmap indices, table partitions, materialized views, etc.

So, every Fact Schema is transformed into a Fact Table, and every Dimension into a Dimension Table. Fact Table Primary Key is composed of the Foreign Keys of the Dimension Tables. Every Synthesis Attribute without synthesis function is transformed into columns of the Fact Table (in this case, data have not been summarized from the operational databases). If Synthesis Functions exist, every function is transformed into a column, which will store the result of applying such function to operational data. A Method is also transformed into Fact Table columns, or views (perhaps materialized views) obtained from Synthesis Attributes and Functions. In the example, the Benefits Method should be transformed into a column. Figure 8 shows the relational schema corresponding to the conceptual schema of Figure 7 in Oracle Designer/2000 format.

4.3 Logical Design: Multidimensional

Because there is not any standard logical multidimensional model, transformation rules in this case have to be based on specific products. Next, we show a transformation into Oracle Express, one of the most used multidimensional DBMS.

Figure 8. Star Schema



Dimension Attributes of IDEA model can be directly transformed into Express Dimensions, by means of the command *Define name dimension type [width number]*. Description Attributes must also be transformed into Dimensions. In Express we can not distinguish between Description and Dimension Attributes. Aggregation Functions are transformed into Express Relations. We have to make correspondences between Dimension Values affected. Synthesis Attributes with no associated Function are transformed into Express Variables. If there is Synthesis Function associated with the Attribute, every Function is transformed into a Variable, using the command *Define name variable type <dimensions>*. IDEA methods can be transformed into Express Formulas of Variables, depending on efficiency issues, storage space, and other physical design criteria. If we choose a Variable, pre-calculated data are physically stored (so, response times are reduced). If we choose a Formula, data are calculated “on-the-fly” (so, response times are worst, and storage space is reduced). The command for defining a formula is *Define name formula expression type <dimensions>*. Finally, a multidimensional schema should be transformed into a Express Database, by means of the command *Database create name*.

5 SUMMARY AND CONCLUSIONS

Datawarehouse development has turned into a critical success factor for many companies. In this chapter we have presented a methodology for the development of multidimensional data warehouses. Such methodology is based on an existing traditional methodology, and it is supported by a CASE tool.

The methodology allows the verification and construction of analytical schemes starting from analytical data user requirements, using existent operational scheme as support for the creation of the conceptual multidimensional scheme.

REFERENCES

- [1] Cabibbo, L. and Torlone, R. (1998) “A Logical Approach to Multidimensional Databases” In Sixth International Conference on Extending Database Technology (EDBT’98), Valencia, Spain, Lecture Notes in Computer Science 1377, Springer-Verlag, 183-197, 1998.
- [2] Cavero, J.M. (2001) “MIDEA: A methodology for data warehouses development”. Ph. D. Thesis, 2001 (in spanish)
- [3] Codd, E. F., Codd, S. B. and Salley, C. T. (1993) “Providing OLAP (On-Line Analytical Processing) to User-Analyst: An IT Mandate”. Technical Report, E. F. Codd and Associates
- [4] Debevoise, N. T. (1999) “The Data Warehouse Method”. Prentice Hall PTR.

- [5] Dinter, B., Sapia, C., Blaschka, M., Höfling, G. (1999) "OLAP Market and Research: Initiating the Cooperation". *Journal of Computer Science and Information Management*, Vol 2, N. 3.
- [6] Gatziau, S., Jeusfeld, M.A., Staudt, M. and Vassiliou, Y. (1999) "Design and Management of Data Warehouses - Report on the DMDW'99 Workshop". *SIGMOD Record* 28(4), Dec.
- [7] Giovinazzo, W.A. (2000), *Object Oriented Data Warehouse Design. Building a Star Schema*. Prentice Hall International
- [8] Golfarelli, M. and Rizzi, S. (1999) "Designing The Data warehouse: Key Steps And Crucial Issues". *Journal Of Computer Science And Information Management*, Vol 2, N. 3.
- [9] Inmon, W.H. (1993) *Building the Data Warehouse*, John Wiley & Sons, New York
- [10] Kimball, R. (1996) *The Data Warehouse Toolkit: Practical techniques for building dimensional data warehouses*. John Wiley & Sons
- [11] Kimball, R., Reeves, L., Ross, M. and Thornthwaite, W. (1998). *The Data Warehouse Lifecycle Toolkit.*, John Wiley & Sons
- [12] Miguel, A. de, Cavero, J.M., Sánchez, A. and Canela, J. (2000) "IDEA-DWCASE: Modeling multidimensional databases", in *EDBT 2000 Software Demonstrations Track*. Konstanz, Germany, March
- [13] Sánchez, A. (2001) "Multidimensional conceptual model IDEA". Ph. D. Thesis (in spanish)
- [14] Srivastava, J. and Chen, P-Y. (1999) "Warehouse Creation - A Potential Roadblock to Data Warehousing" *IEEE Transactions on Knowledge and Data Engineering*. Vol 11, Num. 1, Jan/Feb
- [15] Wu, M.C. and Buchmann, A.P. (1997) "Research Issues in Data Warehousing". *BTW'97*, Ulm, March