A Methodological Framework for Conceptual Data Warehouse Design

Leopoldo Zepeda
Depto. de Sistemas y Computación
Inst. Tecnológico de Culiacán
Juan de Dios Bátiz s/n Col. Gpe.
667-7133804, Cln. Sin. México 80020
lezepeda@dsic.upv.es

Matilde Celma
Depto. De Sistemas y Computación
Universidad Politécnica de Valencía
Camino de Vera s/n
346-3877736, Val. España. 46022
celma@dsic.upv.es

Ramón Zatarain
Depto. de Sistemas y Computación
Inst.Tecnológico de Culiacán Dios
Juan de Dios Bátiz s/n Col. Gpe
667-7133804, Cln. Sin. México 80020
rzatarain@itc.ccs.com

ABSTRACT

A Data Warehouse (DW) has been an approach adopted for giving support to the process of taking decisions in an organization. This paper is concerned with the data warehouse conceptual schema design starting from the conceptual operational schemas and user requirements. We propose and illustrate an algorithm for automatic conceptual schema development. Our algorithm uses an enterprise schema represented with UML as a starting point for source driven data warehouse schema design and produces a set of multidimensional candidates schemas. The candidates schemas are created using an UML profile for data warehouse. Next the automatic generation of multidimensional schemas we use user requirements to guide the selection of the candidates schemas most likely to meet users needs.

Categories and Subject Descriptors

H.2.3 [**Database Management**]: Database Applications.

General Terms

Design

Keywords

Multidimensional design, UML

1. INTRODUCTION

A Data Warehouse (DW) has been an approach adopted for giving support to the process of taking decisions in an organization. The data warehouse component is a database built for analytical processing whose primary goal is to maintain and analyze historical data. The data organization of a data warehouse, called a multidimensional schema, is very simple: the data being analyzed (facts), constitute the star's center with their most important descriptors (measures); around the center, other data describe the dimensions along which data analysis can be performed [1].

In simple warehouses, multidimensional schema may extract their content directly from operational databases; in complex situations, the multidimensional schemas content may be loaded from heterogeneous data sources [2].

This paper is concerned with the data warehouse conceptual schema design starting from the conceptual operational schemas and user's requirements. The method proposed in this paper consists of two basic steps: 1) Obtaining a set of candidate multidimensional schemas from UML schemas. 2) Refine a candidate schema from user's requirements. The main contribution of this work is a new design methodology, which is not only initialized from user's requirements, but also from the operational database schema.

The paper is organized in four sections. Section 2 relates the proposed approach to the state of the art. Section 3 provides a presentation of the methodology. Conclusions and future work are presented in section 4.

2. RELATED WORK

We present one of the most important contributions concerned with the conceptual data warehouse design. The approaches presented in Cabibbo and Torlone [3] and Golfarelli [4,5] are the closest to our methodology. Cabibbo and Torlone [3] illustrate a method for developing multidimensional schemas from the conceptual operational schemas. The design method starts from an existing E/R schema, derives a multidimensional schema, and provides implementations in terms of relational tables as well as multidimensional arrays. The derivation of the multidimensional schema (**MD**) is structured into the following steps (1) identification of facts and dimensions, (2) restructuring of the E/R schema, (3) derivation of a dimensional graph, and (4) translation into the multidimensional model. The multidimensional schemas are based on graphs entailing F-tables (tables that serve as abstract implementation of both relational databases and multidimensional arrays). The proposed design method builds an **MD** schema starting from an underlying operational database, where an **MD** schema consists of a finite set of dimensions, a finite set of F-tables, and a finite set of level descriptions of dimensions.

The work of [4] presents a data warehouse design method, which consists of 5 steps: (1) analysis of the information system, (2) conceptual design (3) schema validation, (4) logical design, (5) physical design. The design of a conceptual schema is carried out by producing a fact schema for each fact, which, can be derived

43rd ACM Southeast Conference, March 18-20, 2005, Kennesaw, GA, USA. Copyright 2005 ACM 1-59593-059-0/05/0003…$5.00.

from an E/R schema using an algorithmic procedure. The procedure transforms the E/R schema in a tree-structured fact schema. This model represents the fact as the tree's root and the dimension attributes as the tree's descendents. The above contributions are concerned with datawarehouse conceptual design starting only from conceptual operational schemas, but they do not integrate this design phase with the user's requirements yet.

2. THE NEW METHODOLOGY

The main problem in DW design is to obtain a set of multidimensional schemas that allow capturing the user requirements and be maintained for the operational database.

The main goal of this work consists in defining a new methodology, which allows to make a design based on the fact established before. In order to tackle the problem we divide the solution in two phases, which are described in this section.

2.1 Phase 1

The methodology to get a set of multidimensional schemas starting from an UML schema, consists in performs an exhaustive analysis to it. The goal is identifying the entities that are candidates to be facts. Once the entities of facts are identified a search for dimensions must be done. The goal is to add dimensions so we can produce a multidimensional schema for each candidate fact identified. In order to get a correct work, we assume the following:

- The generalization was transformed into relations one to one.
- The generalization was transformed into relations one to one.

Then we follow the next three steps: 1) Identifying facts, 2) Defining Dimensions, 3) Defining Hierarchies.

2.1.1 Identifying facts

This step performs a detailed analysis of the UML diagram in order to discover the entities that are candidates to be facts in the multidimensional schema. In an UML diagram context, an entity **H** will be classify as a fact entity, if it has the next feature: **H** has at least one non primary numeric attribute and it is related at least with one entity **E_1**, with many to one cardinality. The identified entities will be stored in an array **F**.

2.1.2 Defining dimensions and hierarchies

A dimension is an UML subschema, and represents a viewpoint used to analyze a fact. During this process we consider the following:

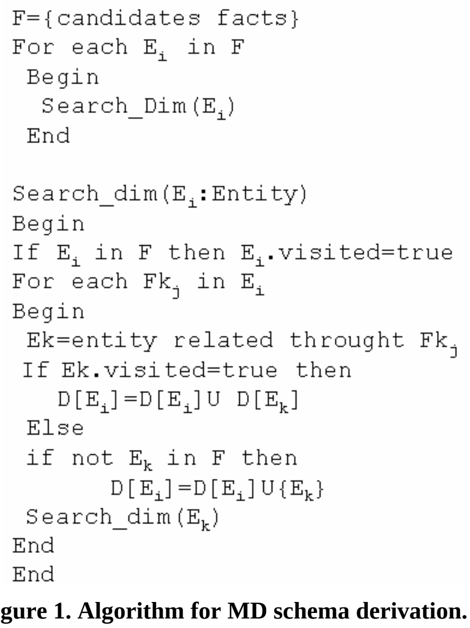
- All the entities of the UML diagram that are not candidates to be facts are candidates to be dimensions.
- A dimension can be connected with more than one entity of facts.
- Whenever we find a relation between two facts entities, the child entity will inherit every dimension of the parent entity.
- The time dimension will always be part of the multidimensional schema, so it will be added to the multidimensional schema.

In Figure 1, an algorithm that finds all the candidate multidimensional schema is given.

The algorithm receives as input the UML schema and the array **F**, and produces as output an array **D** of snowflake schemas (one snowflake schema for each entity in F).

The core of the algorithm is the recursive procedure **Search_Dim().** This procedure receives a potential entity of facts (**E_i**), and mark it as visited. Next the algorithm look for dimensions reachable from the entity of facts through many to one relationships or one to one relationships and assigns it to **E_k**. If **E_k** has already been visited, the elements of **D** associated to it are added to the elements of **D** associated to the entity currently visited (**E_i**); otherwise the search continues adding **E_k** to the array **D**.

The algorithm automatically produces all the possible snowflakes schemas using an UML profile for multidimensional models [6]. In order to fully explain the algorithm, we will use the schema of Figure 2. This schema is for a farmer company. The set of identified facts F in the diagram are**: F={*manifiest, manifest detail, pallet, product, presentations*}.** The first iteration selects an element Ei from the list F, for example: ***E_1* = *manifiest*.** The recursive procedure **Search_Dim()** receives as input parameter the entity manifest and set it to ***visited***.



```
F={candidates facts}
For each $E_i$ in F
 Begin
  Search_Dim($E_i$)
 End

Search_dim($E_i$:Entity)
Begin
If $E_i$ in F then $E_i$.visited=true
For each $Fk_j$ in $E_i$
Begin
 Ek=entity related throught $Fk_j$
 If Ek.visited=true then
   D[$E_i$]=D[$E_i$]U D[$E_k$]
 Else
 if not $E_k$ in F then
       D[$E_i$]=D[$E_i$]U{$E_k$}
 Search_dim($E_k$)
End
End
```

Figure 1. Algorithm for MD schema derivation.

A search in the diagram is done for each entity related with **E_1 (*manifest*).** Those entities will be considered dimension's levels of the snowflake schema associated to **E_1.** So**, D[Manifiest] =** {Agricolcycle, Client, City}. Figure 3, shows the snowflake schema for Manifiest.

After the recursive procedure **Search_Dim() is** done, the algorithm continues the search of dimensions for the second element of the array **F.** So **E_2={***manifest detail***}.** Because there is a relation between ***Manifest detail*** and ***Manifest*** and because ***Manifest*** is set as ***visited,*** the entities in **D[M*anifest*]** will be part of **D[*Manifest detail*].** So **D[*Manifest detail*]= D[*Manifest detail*] U D[*Manifest*].**

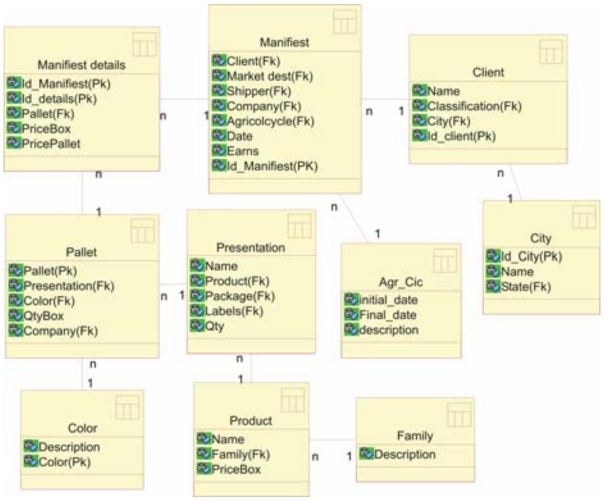


Figure 2. Operational UML Schema.

In Figure 4, we show the snowflake schema for ***Manifest detail.*** The algorithm follows the same step for every element in **F**. (we only show two possible schemas). This Phase, returns all the possible multidimensional schemas extracted from the operational UML schema.

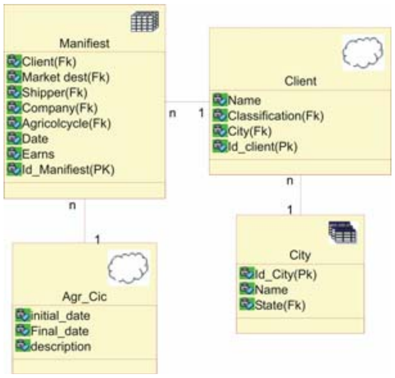


Figure 3. Snowflake schema for Manifiest.

2.2 Phase 2

This phase captures the warehouse end-user requirements, and is considered only to select and refine exiting candidates snowflake schemas. In order to select a candidate schema, it is necessary evaluate the properties of each snowflake schema against user requirements. This evaluation is based on the number of elements. We require mainly a correspondence between the attributes of facts and the number of dimensions.

The goal is to select the candidate schema which best reflect user requirements. In the development of a DW requirements specification is an iterative process, which involve two steps: Identification and process analysis.

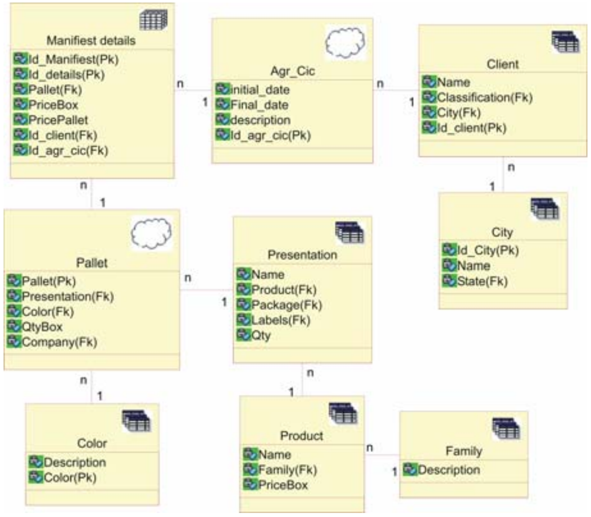


Figure 4. Snowflake schema for Manifiest detail.

The first step is the identification and definition of process. For this, it is necessary having a set of interview or workgroup [7]. During this phase we get a list of processes for each one of the interviewed users. Beside that, an analysis of each process must be done with the aim of identifying the information that can be used to refine a candidate snowflake schema. Continuing with the example (farmer company). The interviews directed to the workgroup allow us identify two processes: *sales* and *delivers*. We show a summary list of analysis needs identified during the interviews**:**

Sales Information: Sales information includes presentations, products, agricocycle, destination, clients, company, market (national or foreign).

Delivers Information: Delivers analysis requires market information (national or foreign), clients, etc.

Summarizing, the main needs of the company are to study and analyze the behavior of the market with the aim of improving the production and marketing. Another aim is to improve the service of delivering for the different products. The description of the sale process is shown in Table 1. Due to paper size limitations we omit the delivery process.

Whenever we want to select a multidimensional schema, it is necessary to evaluate the multidimensional schema against the user requirements. This evaluation is based on the number of elements of the multidimensional schema. We require mainly a correspondence between the attributes of facts and dimensions. The goal is to select the snowflake schema which best reflect user requirements. The integration is carried out in two steps:

Table 1. Summary of requirements

Process	No. Dimensiones	No. Attributes
Sales	**8**	**3**
Attributes	Dimensions	
Num_clients	**Client, City, Color**	
PriceBox	**Presentation,Product Family**	
PricePallet	**Company, Agr_cicle**	

Comparison. - It is necessary to do a comparison between the multidimensional elements of each schema produced by the algorithm and the elements produced from user requirements. The metrics to decide if a snowflake schema acquires them are:

Corresponding Attributes. We must count the number of attributes from the table of facts of each candidate schema with the attributes identified from user requirements.

Corresponding dimensions. We must count the number of dimensions from each multidimensional schema that are corresponding with the dimensions obtained from user requirements.

This metrics are the most important ones. The reason is that they indicated how a schema is capable of giving information required by the user. From this information we can select the multidimensional schema ***Manifiest detail*** because it captures better the user requirements and is supported by the operational data.

Manual refinement. - Once the schema was obtained it must be manually modified with the goal of represent the most important multidimensional aspects like: aggregation constraints about measures and special dimensions properties like covering, strictness, and completeness. The modifications must be done from the collected data during the analysis phase. Figure 5 shows the modified snowflake schema. The changes are:

- Was added the attribute NumClients to the fact table.
- The aggregation constraint about measure NumClients was also added.

3. CONCLUSIONS

This work presents a new methodology for DW conceptual design. This methodology is based on the UML model and user requirements. Our proposal is divided in two phases. The first phase starts with facts identification based on a set of requirements. Next, using the entities of facts and an UML schema, we get a set of multidimensional schemas. We can achieve that by using a recursive algorithm.

The second phase gets a set of metrics from user requirements. These metrics allow us to select a multidimensional schema from a set of candidate schemas and refine it. Our future work will extend this proposal hoping to increase the set of metrics. The new metrics will allow selecting the multidimensional schema from the candidate schemas. We can achieve that by defining or adopting a new methodology that allow analyze the requirements of potential users.

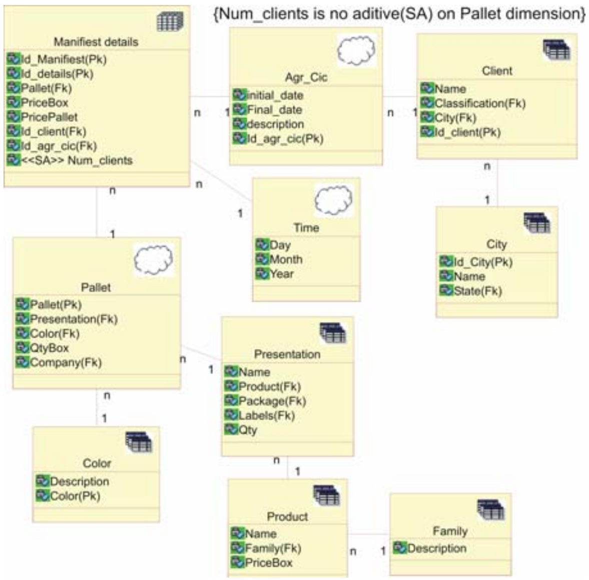


Figure 5. Ideal Snowflake Schema.

4. REFERENCES

[1] Kimball, R, “The Data Warehouse Toolkit: Practical Techniques for building Dimensional Data Warehouses”, John Wiley and Sons, Inc., New York, NY, 1998**.**

[2] Vicky Nassis1, R. Rajugan2, Tharam S. Dillon2, and Wenny Rahayu1. Discovering Data Warehousing and Knowledge Discovery: 6th International Conference, DaWaK 2004, Zaragoza, Spain, September 1-3, 2004. Proceedings ISBN: 3-540-22937-X.

[3] Cabibbo, L. and Torlone. A logical approach to multidimensional databases. In Proceedings of the International Conference on Extending Data Base Technology (EDBT ’98,Valencia, Spain, Mar.). 183–197.

[4] Golfarelli, M, and Rizzi, S. 1998. Conceptual design of data warehouses from E/R schemas. In Proceedings of the 31st Hawaii International Conference on System Sciences (HICSS ’98, Kona, Hawai).

[5] Zepeda, L. Matilde C. An UML Profile for Data warehouse Design. In proceedings of CIC 2004., México.ISBN:970-36-0194-4

[6] Beate List, Josef Schiefer, A Min Tjoa. Process-Oriented Requirement Analysis Supporting the Data Warehouse Design Process Use Case Driven Approach.