WAND: A CASE Tool for Data Warehouse Design

Matteo Golfarelli
DEIS - University of Bologna
Viale Risorgimento, 2
40136 Bologna - Italy
mgolfarelli@deis.unibo.it

Stefano Rizzi
DEIS - University of Bologna
Viale Risorgimento, 2
40136 Bologna - Italy
srizzi@deis.unibo.it

Abstract

The statistic reports about data warehouse project failures state that a major cause lies in the absence of a structured design methodology. In this direction, our research is aimed at defining the basic steps required for a correct design. The goal of this demonstration is to present the main features of WAND, the prototype CASE tool we have implemented to support our methodology. WAND assists the designer in structuring a data mart, carries out conceptual design in a semi-automatic fashion, allows for a core workload to be defined on the conceptual scheme and carries out logical design to produce the data mart scheme.

1. Motivation and overview

Building a *data warehouse* (DW) for an enterprise is a huge and complex task, which requires an accurate planning aimed at devising satisfactory answers to organizational and architectural questions. There is substantial agreement on the fact that, when planning a DW, a bottom-up approach should be followed: the *data mart* playing the most strategic role for the enterprise should be identified and prototyped first in order to convince the final users of the potential benefits; other data marts are built progressively, to be finally integrated bottom-up into the global warehouse.

Within this process, the phase of data mart design sometimes seems to be given relatively small importance. On the other hand, the statistic reports related to DW project failures state that a major cause lies in the absence of a global view of the design process: in other terms, in the absence of a structured design methodology [6].

We believe that a methodological framework for design is an essential requirement to ensure the success of complex projects. In this direction, our research is aimed at defining the basic steps required for a correct design. In particular, we considered the problem of conceptual design from the operational database as well as some relevant issues related to logical design.

It is well-known among software engineers that devising a design methodology is almost useless, if no CASE tool to support it is provided. The goal of this demonstration is to present the main features of WAND (Warehouse Integrated Designer), the prototype CASE tool we have implemented to support our methodology. WAND assists the designer in structuring a data mart; it carries out conceptual design in a semi-automatic fashion starting from the logical scheme of the operational database (read via ODBC), allows for a core workload to be defined on the conceptual scheme and carries out logical design to produce the data mart scheme. Both the

Table I. The six phases in our DW design methodology.

Step	*Input*	*Output*	*Involves*
Analysis of the operational database	existing documentation	(reconciled) database scheme	designer; managers of the information system
Requirement specification	database scheme	facts; preliminary workload	designer; final users
Conceptual design	database scheme; facts; preliminary workload	conceptual scheme	designer; final users
Workload refinement, conc. scheme validation	conceptual scheme; preliminary workload	validated conceptual scheme; workload	designer; final users
Logical design	conceptual scheme; data volume; workload	logical scheme	designer
Physical design	logical scheme; target DBMS; workload	physical scheme	designer

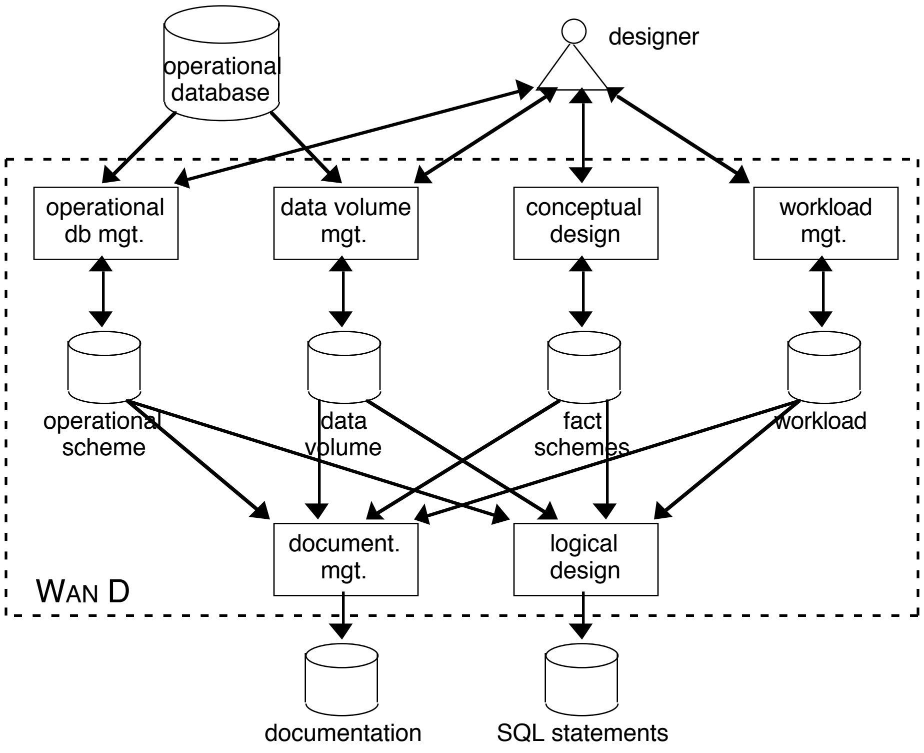

Figure 1. The functional architecture of WAND

operational source and the target DBMS are relational. The conceptual model adopted is the *Dimensional Fact Model* (DFM), described in [2].

The demonstration will be based on a sample operational database describing a supply chain, and on the standard TPC-D database.

2. The methodological framework

According to our methodology, outlined in [4], each data mart is mainly created in a top-down fashion by building a fact scheme for each fact of interest. The design steps are summarized in Table I; those actively supported by WAND (steps 3, 4, 5) are shaded in gray. As a matter of fact, our on-the-field experience showed that also step 2 (requirement specification) can be more fruitfully carried out by relying on WAND to quickly sketch different conceptual scenarios to be submitted to the users and interactively tuned to meet their expectations.

3. The WanD architecture

The functional architecture of WanD is sketched in Figure 1; its main functions are:

- *Acquisition of the relational scheme describing the operational database.* The scheme is acquired by querying, via ODBC, the DBMS hosting the database. If necessary, the designer can then edit the scheme (for instance, by defining primary and foreign keys if they are not defined in the source scheme or if the ODBC driver used does not support them).
- *Conceptual design*. Conceptual design is carried out semi-automatically from the scheme of the operational database, according to the algorithm proposed in [2]. The output is a set of fact schemes, one for each fact of interest (see Figure 2).

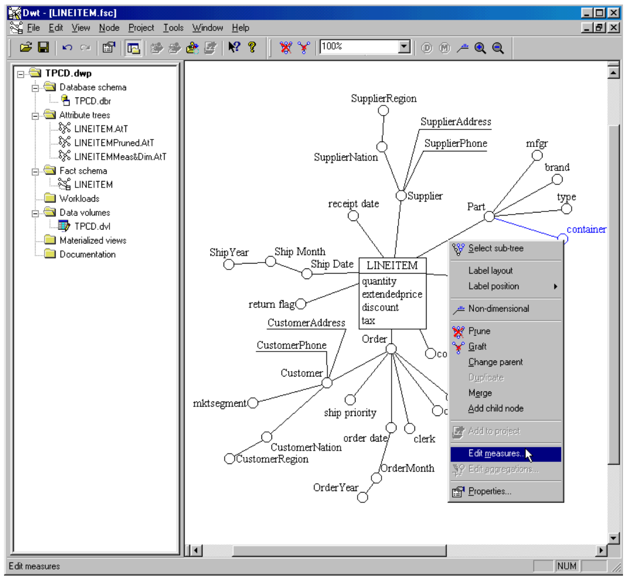

Figure 2. Editing the fact scheme

- *Workload definition*. The designer defines the most frequent/interesting queries on fact schemes through a simple query language.
- *Data volume acquisition*. WAND estimates the current data volume by querying the operational database via ODBC.
- *Logical design*. For each fact scheme defined, and taking both the workload and the data volume into account, WAND carries out view materialization as in [1] and [5], then applies the vertical fragmentation algorithm proposed in [3]; in output, WAND automatically generates the SQL statements defining fact and dimension tables according to the star/snowflake scheme, as well as the SQL queries allowing these tables to be populated from the operational database.
- *Production of the documentation*. The documentation produced includes the fact schemes designed, the attribute and measure glossaries, the workload and the data volume.

References

[1] E. Baralis, S. Paraboschi, and E. Teniente, "Materialized view selection in multidimensional database", *Proc. 23rd Int. Conf. on Very Large Data Bases*, Athens, Greece, 1997, pp. 156-165.

[2] M. Golfarelli, D. Maio, and S. Rizzi, "The Dimensional Fact Model: a conceptual model for data warehouses", *Int. Jour. of Cooperative Inf. Syst.*, vol. 7, n. 2&3, 1998, pp. 215-247.

[3] M. Golfarelli, D. Maio, and S. Rizzi, "Applying Vertical Fragmentation Techniques in Logical Design of Multidimensional Databases", *Proc. DaWaK 2000*, Greenwich, 2000, pp. 11-23.

[4] M. Golfarelli, and S. Rizzi, "Designing the data warehouse: key steps and crucial issues", *Jour. of Computer Science and Information Management*, vol. 2, n. 3, 1999.

[5] H. Gupta, "Selection of views to materialize in a data warehouse", *Proc. Int. Conf. on Database Theory*, Delphi, Greece, 1997, pp. 98-112.

[6] S. Kelly, "Data warehousing in action", John Wiley & Sons, New York, 1997.