Data warehouse life-cycle and design

Matteo Golfarelli
DEIS – University of Bologna
Via Sacchi, 3 Cesena – Italy
matteo.golfarelli@unibo.it

SYNONYMS

Data Warehouse design methodology

DEFINITION

The term *data warehouse life-cycle* is used to indicate the phases (and their relationships) a data warehouse system goes through between when it is conceived and when it is no longer available for use. Apart from the type of software, life-cycles typically include the following phases: requirements analysis, design (including modeling), construction, testing, deployment, operation, maintenance and retirement. On the other hand, different life-cycles differ in the relevance and priority with which the phases are carried out, that can vary according to the implementation constraints (i.e. economic constraints, time constraints, etc.) and the software specificities and complexity; in particular, the specificities in the data warehouse life cycle derive from the presence of the operational database that feeds the system and by the extent of this kind of system that must be considered in order to keep under control the cost and the complexity of the project.

Although the design phase is only a step within the overall life-cycle, the identification of a proper life-cycle model and the adoption of a correct *design methodology* are strictly related since each one influences the other.

HISTORICAL BACKGROUND

The *data warehouse* (DW) is acknowledged as one of the most complex information system modules and its design and maintenance is characterized by several complexity factors that determined, in the early stages of this discipline, a high percentage of project failures. A clear classification of the critical factors of Data Warehousing projects was already available in 1997 when three different risk categories were identified [3]:

- *Socio-technical:* DW projects have deep impact on the decisional processes and political equilibriums, thus reducing the power of some stakeholders that will be willing to interfere with the project. For example, data ownership is power within an organization. Any attempt to share or take control over somebody else's data is equivalent to a loss of power of this particular stakeholder. Furthermore, no division or department can claim to possess 100% clean, error-free

data. The possibility of revealing the data quality problems within the information system of the department is definitely frustrating for the stakeholders affected.

- *Technological:* DW technologies are continuously evolving and their features are hard to test. As a consequence problems related to the limited scalability of the architecture, difficulty in sharing meta-data between different components and the inadequate expertise of the programmers may hamper the projects.
- *Design:* designing a DW requires a deep knowledge of the business domain. Some recurrent errors are related to limited involvement of the user communities in the design as well as the lack of a deep analysis of the quality of the source data. In both these cases the information extracted from the DW will have a limited value for the stakeholders since they will turn out to be unreliable and outside the user focus.

The awareness of the critical nature of the problems and the experience accumulated by practitioners determined the development of different design methodologies and the adoption of proper life-cycles that can increase the probability of completing the project and fulfil the user requirements.

SCIENTIFIC FUNDAMENTALS

The choice of a correct life-cycle for the DW must take into account the specificities of this kind of systems, that according to [4], are summarized as follows:

a) DWs rely on operational databases that represent the sources of the data.
b) User requirements are difficult to collect and usually change during the project.
c) DW projects are usually huge projects: the average time for their construction is 12 to 36 months and their average cost ranges from 0.5 to 10 million dollars.
d) Managers are demanding users that require reliable results in a time compatible with business needs.

While there is no consensus on how to address points (a) and (b), the DW community has agreed on an approach that cuts down cost and time to make a satisfactory solution available to the final users. Instead of approaching the DW development as a whole in a top-down fashion, it is more convenient to build it bottom-up working on single data marts [11]. A *data mart* (please refer to the “Data Mart” entry for more details) is part of a DW with a restricted scope of content and support for analytical processing, serving a single department, part of an organization and/or a particular data analysis problem domain. By adopting a bottom-up approach, the DW will turn out to be the union of all the data marts.

This iterative approach promises to fulfil requirement (c) since it cuts down development costs and time by limiting the design and implementation efforts to get the first results. On the other hand, requirement

(d) will be fulfilled if the designer is able to implement first those data marts that are more relevant to the stakeholders.

As stated by many authors, adopting a pure bottom-up approach presents many risks originating from the partial vision of the business domain that will be available at each design phase. This risk can be limited by first developing the data mart that plays a central role within the DW, so that the following ones can be easily integrated into the existing backbone, this kind of solution is also called *bus architecture*. The basis for designing coherent data marts and for achieving an integrated DW is the agreement of all the design teams on the classes of analysis that are relevant for the business. This is primarily obtained by the adoption of *conformed dimensions* of analysis [12]. A dimension is conformed when two copies of the dimensions are either exactly the same (including the values of the keys and all the attributes), or else one dimension is a proper subset of the other. Therefore, using the same Time dimension in all the data marts implies that the data mart teams agree on a corporate calendar. All the data mart teams must use this calendar and agree on fiscal periods, holidays, and workdays. When choosing the first data mart to be implemented the designer will probably cope with the fact that the most central data mart (from a technical point of view) is not the most relevant to the user; in that case the designer choice must be a trade-off between technical and political requirements.

Based on these considerations the main phases for the DW life-cycle can be summarized as follows:

1. *DW planning*: this phase is aimed at determining the scope and the goals of the DW, and determines the number and the order in which the data marts are to be implemented according to the business priorities and the technical constraints [12]. At this stage the physical architecture of the system must be defined too: the designer carries out the sizing of the system in order to identify appropriate hardware and software platforms and evaluates the need for a reconciled data level aimed at improving data quality (please refer to the “Data warehousing systems: foundations and architectures” entry for more details). Finally, during the project planning phase the staffing of the project is carried out.
2. *Data mart design and implementation*: this macro-phase will be repeated for each data mart to be implemented and will be discussed in more detail in the following. At each iteration a new data mart is designed and deployed. Multidimensional modeling of each data mart must be carried out considering the available conformed dimensions and the constraints deriving from previous implementations.
3. *DW maintenance and evolution*: DW maintenance mainly concerns performance optimization that must be periodically carried out due to user requirements that change according to the problems and the opportunities the managers run into. On the other hand, DW evolution concerns keeping

the DW schema up-to-date with respect to the business domain and the business requirement changes: a manager requiring a new dimension of analysis for an existing *fact schema* (please refer to the "Multidimensional modeling" and to the "data warehousing systems: foundations and architectures" entries for more details) or the inclusion of a new level of classification due to a change in a business process may cause the early obsolescence of the system (please refer to the "Data warehouse maintenance, evolution and versioning" entry for more details).

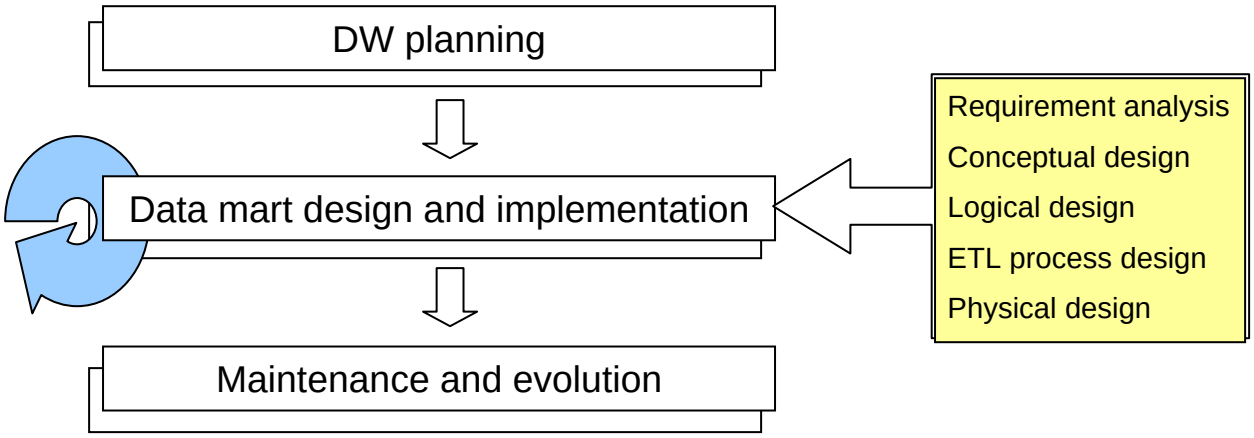

Figure 1: The main phases for the DW life-cycle.

DW design methodologies proposed in the literature mainly concern phase 2 and thus should be better referred to as data mart design methodologies. Though a lot has been written about how a DW should be designed, there is no consensus on a design method yet. Most methods agree on the opportunity of distinguishing between the following phases:

- *Requirement analysis:* identifies which information is relevant to the decisional process by either considering the user needs or the actual availability of data in the operational sources.
- *Conceptual design*: aims at deriving an implementation-independent and expressive conceptual schema for the DW, according to the conceptual model chosen (see Figure 2).
- *Logical design*: takes the conceptual schema and creates a corresponding logical schema on the chosen logical model (please refer to the "Cube Implementation" entry for more details). While nowadays most of the DW systems are based on the relational logical model (ROLAP), an increasing number of software vendors are proposing also pure or mixed multidimensional solutions (MOLAP/HOLAP). Figure 3 reports the relational implementation of the SALE fact based on the well-known star schema [12].
- *ETL process design*: designs the mappings and the data transformations necessary to load into the logical schema of the DW the data available at the operational data source level.

- *Physical design*: addresses all the issues specifically related to the suite of tools chosen for implementation – such as indexing and allocation.

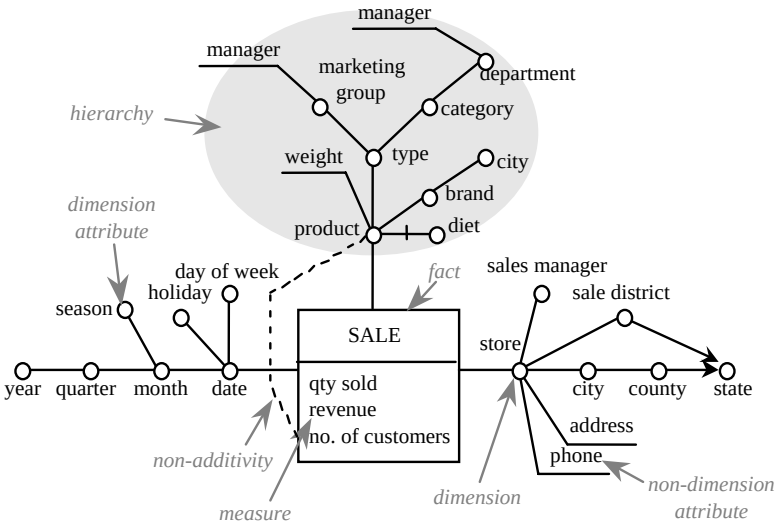

Figure 2: A conceptual representation for the SALES fact based on the DFM model [5].

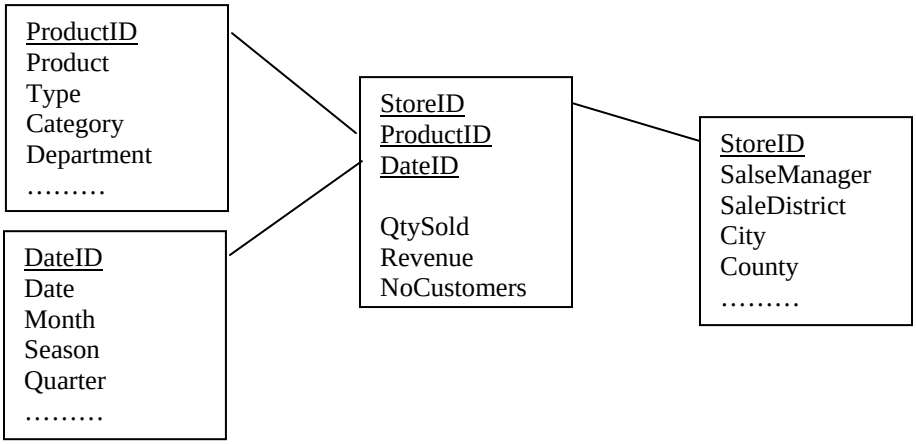

Figure 3: A relational implementation of the SALE fact using the well-known star schema.

Requirement analysis and conceptual design play a crucial role in handling DW peculiarities (a) and (b) described at the beginning of the present section: the lack of settled user requirements and the existence of operational data sources that fix the set of available information make it hard to develop appropriate multidimensional schemata that on the one hand fulfill user requirements and on the other can be fed from the operational data sources. Two different design principles can be identified: supply-driven and demand-driven [10].

- *Supply-driven* approaches [5, 11] (also called *data-driven*) start with an analysis of operational data sources in order to reengineer their schemata and identify all the available data. Here user involvement is limited to select which chunks of the available data are relevant for the decision-making process. While supply-driven approaches simplify the design of the ETL because each piece of data in the DW corresponds to one or more attributes of the sources, they give user requirements a secondary role in determining the information contents for analysis as well as giving the designer little support in identifying facts, dimensions, and measures. Supply-driven approaches are feasible when all of the following are true: (1) detailed knowledge of data sources is available a priori or easily achievable; (2) the source schemata exhibit a good degree of normalization; and (3) the complexity of source schemata is not too high.
- *Demand-driven* approaches [13,20] start from determining the information requirements of business users. The emphasis is on the requirement analysis process and on the approaches for facilitating user participations. The problem of mapping these requirements onto the available data sources is faced only a posteriori, and may fail thus determining the users' disappointment as well as a waste of the designer's time.

Based on the previous approaches some mixed modeling solutions have been proposed in the last few years in order to overcome the weakness of each pure solution.

Conceptual design is widely recognized to be the necessary foundation for building a DW that is well-documented and fully satisfies the user requirements. The goal of this phase is to provide the designer with a high level description of the data mart possibly at different levels of detail. In particular, at the DW level it is aimed at locating the data mart within the overall DW picture, basically characterizing the class of information captured, its users and its data sources. At the data mart level, a conceptual design should identify the set of facts to be built and their conformed dimensions. Finally, at the fact level a non ambiguous and implementation-independent representation of each fact should be provided. If a supply driven approach has been followed for requirement analysis, the conceptual model at the schema level can be semi-automatically derived from the source schemata by identifying the many-to-one relationship [5, 11]. Concerning the formalism to be adopted for representing information at this level, researchers and practitioners agreed that, although the E/R model has enough expressivity to represent most necessary concepts, in its basic form, it is not able to properly emphasize the key aspects of the multidimensional model. As a consequence many ad-hoc formalisms has been proposed in the last years (e.g. [5,9]) and a comparison of the different models done by [1] pointed out that, abstracting from their graphical form, the

core expressivity is similar, thus proving that the academic community reached an informal agreement on the required expressivity (please refer to the “Multidimensional modeling” entry for more details).

Logical design is the phase that most attracted the interest of researchers in the early stage of Data Warehousing since it strongly impacts the system performance. It is aimed at deriving out of the conceptual schemata the data structure that will actually implement the data mart by considering some sets of constraints (e.g., concerning disk space or query answering time) [15]. Logical design is more relevant when a relational DBMS is adopted (ROLAP) while in the presence of a native multidimensional DBMS (MOLAP) the logical model derivation is straightforward. On the other hand, in ROLAP system, the choices concern for example the type of schema to be adopted (i.e. star o snowflake), the specific solution for historicization of data (i.e. slowly changing dimensions) and schema (please refer to the “Data warehouse maintenance, evolution and versioning” entry for more details).

ETL process design is considered to be the most complex design phase and usually takes up to the 70% of the overall design time. Complexity arises from the need of integrating and transforming heterogeneous and inconsistent data coming from different data sources, this phase also includes the choice of the strategy for handling wrong and incomplete data (e.g. discard, complete). Obviously, the success of this phase impacts the overall quality of DW data. Differently from other design phases little efforts have been made in the literature to organize and standardize this phase [18,19], and actually none of the formalisms proposed have been widely adopted in real projects that usually rely on the graphical representation obtained from the ETL tool for documentation purposes.

Finally, during physical design, the logical structure is optimized based on the means made available by the adopted suite of tools. Specialized DBMSs usually include ad hoc index types (e.g. bitmap index and join index) and can store the meta-knowledge necessary to automatically rewrite a given query on the appropriate materialized view (please refer to the “Data warehouse optimization and tuning” and “Data warehouse indexing” entries for more details). In DW systems, a large part of the available disk space is devoted to optimization purposes and it is a designer task to find out its assignment to the different optimization data structures in order to maximize the overall performance [8].

Despite the basic role played by a well-structured methodological framework in ensuring that the DW designed fully meets the user expectations, only a few of the cited papers cover all the design phases [5, 19]. In addition to them, an influential book, particularly from the practitioners’ viewpoint, is the one by Kimball [12], which discusses the major issues arising in the design and implementation of data warehouses. The book presents a case-based approach to data mart design that is bottom-up oriented and adopts a mixed approach for collecting user requirements.

Finally it should be noted that, though most vendors of DW technology propose their own CASE solutions (that are very often just wizards capable of supporting the designer during the most tedious and repetitive phases of design), the only tools that currently promise to effectively automate some phases of design are research prototypes. In particular, [6, 11], embracing the supply-driven philosophy, propose two approaches for automatically deriving the conceptual multidimensional schema from the relational data sources. On the contrary the CASE tool proposed in [18] follows the demand-driven approach and allows the multidimensional conceptual schemata to be drawn from scratch and to be semi-automatically translated into the target commercial tool.

KEY APPLICATIONS

The adoption of an appropriate methodological approach during design phases is crucial to ensure the project success. People involved in the design must be skilled on this topic, in particular:

Designers

Designers should have a deep knowledge of the pros and cons of different methodologies in order to adopt the one that best fits the project characteristics.

Business users

Users should be aware of the design methodology adopted and their role within it in order to properly support the designer's work and to provide the correct information at the right time.

FUTURE DIRECTIONS

Research on this topic should be directed to generalizing the methodologies discussed so far in order to derive a consensus approach that, depending on the characteristics of the project, will be made up of different phases. Besides, more generally, mechanisms should appear to coordinate all DW design phases allowing the analysis, control, and traceability of data and metadata along the project life-cycle. An interesting approach in this direction consists in applying the Model Driven Architecture in order to automate the inter schema transformations from requirement analysis to implementation [14]. Finally, the emergence of new applications for DW such as spatial DW [2], web DW, real-time DW [16] and business performance management [7] will have their side-effects on the DW life-cycle and inevitably more general design methodologies will be devised in order to allow their correct handling.

CROSS REFERENCES

Cube Implementation

Data Mart

Data warehouse indexing

Data warehouse maintenance, evolution and versioning

Data warehouse optimization and tuning

Data warehousing systems: foundations and architectures

Data warehousing

Multidimensional modeling

Snowflake schema

Star schema

RECOMMENDED READING

[1] Abello, A., J. Samos, F., and Saltor F. YAM2: a multidimensional conceptual model extending UML. Information System, 31(6), 541-567, 2006.

[2] Bimonte, S., Towards S., and Miquel M..Towards a Spatial Multidimensional Model. In Proceedings of DOLAP, 2005.

[3] Demarest, M. The politics of data warehousing. Retrieved June 2007 from http://www.noumenal.com/marc/dwpoly.html.

[4] Giorgini, P., Rizzi, S., and Garzetti, M. GRAnD: A goal-oriented approach to requirement analysis in data warehouses. To appear on *Decision Support System*, 2008.

[5] Golfarelli, M., Maio, D., and Rizzi, S. The Dimensional Fact Model: A Conceptual Model for Data Warehouses. *IJCIS* 7(2-3): 215-247, 1998.

[6] Golfarelli, M., and Rizzi, S. WAND: A CASE tool for data warehouse design. In Proceedings of *ICDE*, 2001.

[7] Golfarelli, M., Rizzi, S., and Cella, I.. Beyond data warehousing: What's next in business intelligence? In Proceedings *DOLAP*, 2004.

[8] Golfarelli, M., Rizzi, S., and Saltarelli, E. Index selection for data warehousing. In Proceedings of *DMDW*, 2002.

[9] Hüsemann, B., Lechtenbörger J., Vossen G. Conceptual data warehouse design, *DMDW*, 2000.

[10] Jarke, M., Lenzerini, M., Vassiliou, Y., and Vassiliadis, P.. *Fundamentals of Data Warehouses*. Springer 2000.

[11] Jensen, M., Holmgren, T., and Pedersen T. Discovering Multidimensional Structure in Relational Data. In Proceedings of *DaWaK*, 2004

[12] Kimbal, R., Reeves, L., Ross, M., and W. Thornthwaite. *The Data Warehouse Lifecycle Toolkit*. John Wiley and Sons, Inc., NY, 1998.

[13] Laender, A., Freitas, G., and Campos, M. MD2 – Getting users involved in the development of data warehouse applications. In Proceedings of *CAiSE*, 2002.

[14] Mazon, J., Trujillo, J., Serrano, M., and Piattini, M. Applying MDA to the development of data warehouses. In Proceedings of *DOLAP*, 2005.

[15] Theodoratos, D., and Sellis, T. Designing data Data warehouses. *DKE*, 31(3):279–301, 1999.

[16] Tho, N., and Tjoa, A.. Grid-Based Zero-Latency Data Warehousing for continuous data streams processing. In Proceedings of *IIWAS2004*, 2004.

[17] Trujillo, J., and Luján-Mora, S. A UML Based Approach for Modeling ETL Processes in Data Warehouses. In Proceedings of *ER*, 2003.

[18] Trujillo, J., Luján-Mora, S., and Medina, E. The Gold model case tool: An environment for designing OLAP applications. In Proceedings of *ICEIS*, 2002.

[19] Vassiliadis, P., Simitsis, A., and Skiadopoulos, S. Conceptual Modeling for ETL Processes. In Proceedings of *DOLAP*, 2002.

[20] Winter, R., and Strauch, B.. A method for demand-driven information requirements analysis in data warehousing. In Proceedings of *HICSS*, 2003.