From User Requirements to Conceptual Design in Data Warehouse Design – a Survey

Matteo Golfarelli
DEIS - University of Bologna
Via Sacchi, 3 47023 – Cesena ITALY
voice: +39 - 0547 - 338862
fax: +39 - 0547 - 338890
email: matteo.golfarelli@unibo.it

ABSTRACT

Conceptual design and requirement analysis are two of the key steps within the data warehouse design process. They are to a great extent responsible for the success of a data warehouse project since, during these two phases, the expressivity of the multidimensional schemata is completely defined. This paper proposes a survey of the literature related to these design steps and points out pros and cons of the different techniques in order to help the reader to identify crucial choices and possible solutions more consciously. Particular attention will be devoted to emphasizing the relationships between the two steps describing how they can be jointly used fruitfully.

INTRODUCTION

Data Warehouse (DW) systems are used by decision makers to analyze the status and the development of an organization. DWs are based on large amounts of data integrated from heterogeneous sources into multidimensional schemata which are optimized for data access in a way that comes natural to human analysts. Generally speaking, a multidimensional schema is made up of facts, measures and dimensions. Facts are a focus of interest for the decision-making process (e.g. sales, orders) and can be monitored through measures and dimensions. Measures are numerical KPIs (e.g., quantity of product sold, price of the products, etc.), and dimensions represent the context for analyzing these measures (e.g., time, customer, product, etc.). Owing to their specificities, the development of DWs is particularly complex and requires ad-hoc methodologies and an appropriate life-cycle.

Conceptual design and requirement analysis are two of the key steps within the DW design process. While they were partially neglected in the first era of data warehousing, they have received greater attention in the last ten years.

The research literature has proposed several original approaches for conceptual modeling in the DW area, some based on extensions of known conceptual formalisms (e.g. E/R, UML), some based on ad hoc ones. Remarkably, a comparison of the different models pointed out that, abstracting from their graphical form, the core expressivity is similar, thus proving that the academic community has reached an informal agreement on the required expressivity.

On the other hand, the proposed solutions are not always coupled with an appropriate technique for requirement analysis to form a methodological approach ensuring that the resulting database will be well-documented and will fully satisfy the user requirements. DW specificities make these two steps even more related than in traditional database systems; in fact 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, fulfil user requirements and on the other, can be fed from the operational data sources.

This paper proposes a survey of the literature related to these design steps in order to help the reader make crucial choices more consciously. In particular, after a brief description of the DW lifecycle, the specific problems arising during requirement analysis and conceptual design are presented. The approaches to requirement analysis are then surveyed and their strengths and weaknesses are discussed. Afterwards, the literature related to the DW conceptual models is also surveyed and the core expressivity of these models is discussed in order to enable the reader to understand which are the relevant pieces of information to be captured during user-requirements analysis.

BACKGROUND

The 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 real 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 (Demarest, 1997), namely s*ocio-technical* i.e. related to the impact a DW has on the decisional processes and political equilibriums, t*echnological* i.e. related to the usage of new and continuously evolving technologies, and *design-related* i.e. related to the peculiarities of this kind of systems. 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.

The choice of a correct life-cycle for the DW must take into account the specificities of this kind of system, that according to Giorgini et al. (2007), 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 costs 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 (Jensen et al., 2004). A *data mart* 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.

It should be noted that 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* (Kimbal et al., 1998).

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 (Kimbal et al., 1998). At this stage the physical architecture of the system must also be defined: 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. 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.
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 (Golfarelli & Rizzi, 2009) 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* 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.

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:

2.1 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.

2.2 Conceptual design: aims at deriving an implementation-independent and expressive conceptual schema for the data mart, according to the conceptual model.

2.3 Logical design: takes the conceptual schema and creates a corresponding logical schema on the chosen logical model. 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).

2.4 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.

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

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, fulfil user requirements and on the other, can be fed from the operational data sources.

REQUIREMENT ANALYSIS TECHNIQUES

In this section we classify the requirement analysis techniques proposed in the literature discussing their strengths and weaknesses in order to help the reader understand when and why they can be successfully adopted in a real project.

First of all it is necessary to distinguish between *functional* and *non-functional* requirements (Mylopoulos et al. 1992, Chung et al. 1999). Informally speaking, in software systems functional requirements answer what the system does and non-functional requirements answer how the system behaves with respect to some observable quality attributes like performance, reusability, reliability, etc. More specifically within DW systems functional requirements are mainly related to *what* information the DW is expected to provide, while non-functional ones answer *how* this information should be provided for a correct use. Unfortunately, as has already happened in many other software engineering fields, much more work has be done for the first type than for the second one.

As for functional requirements three different design principles can be identified: supply-driven, goal-driven and user-driven. Part of the literature refers to the last two approaches as *demand-driven* since requirements are mainly obtained by interviewing the company personnel. We will use this term when the peculiarities that distinguish the two original ones are not relevant.

The *Supply-driven* approach (also called *data-driven*) is a bottom-up technique that starts with an analysis of operational data sources in order to identify all the available data (Golfarelli et al. 1998, Jensen et al. 2004). Here user involvement is limited to select which chunks of the available data are relevant for the decision-making process. Supply-driven approaches are feasible when all of the following are true: (1) detailed knowledge of data sources is available a priori or easily obtainable; (2) the source schemata exhibit a good degree of normalization; and (3) the complexity of source schemata is not too high. While the supply-driven approach simplifies the design of the ETL because each data in the DW corresponds to one or more attributes of the sources, it gives user requirements a secondary role in determining the information contents for analysis and gives the designer little support in identifying facts, dimensions, and measures. Consequently, the multidimensional schemata obtained could not fit the user requirements. This happens not only when business users are asking for information that is not actually present in the data sources, but also when the desired KPIs are not directly available but could be obtained through some computations. In other words, with the supply-driven approach there is the risk of generating performance information targeting a non-specified user group; such a risk is particularly high when the target users are at the strategic levels where the use of complex and compound KPIs is more common. On the other hand, the supply-driven approach is simpler and cheaper (in terms of both time and money) than other approaches since its duration only depends on the designer skills and on the data sources complexity. A further strength of this approach is the quality of the resulting multidimensional model that will be very stable since it is based on the schema of the operational data sources that do not change as frequently as the personal requirements of the business users.

The *user-driven* approach is a bottom-up technique that starts from determining the information requirements of different business users (Winter & Strauch, 2003). Their points of view are then integrated and made consistent in order to obtain a unique set of multidimensional schemata. 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. Although this approach is highly appreciated by business users that feel involved in the design and can understand the rationale of the choices, it is usually time expansive since the business users at the tactical level rarely have a clear and shared understanding of the business goals, processes and organization. Consequently, this approach usually requires great effort by the project manager, that must have very good moderation and leadership skills, in order to integrate the different points of view. Furthermore, the risk of obsolescence of the resulting schemata is high if requirements are based on the personal points of view of the users and do not express the company culture and the working procedures.

The g*oal-driven approach* focuses on the business strategy that is extrapolated by interviewing the top-management. Different visions are then analyzed and merged in order to obtain a consistent picture and finally translated into quantifiable KPIs. This approach (Boehnlein & Ulbrich vom Ende, 2000) is typically top-down since by starting from the analysis of a few key business processes, their characterizing measurements are derived first and than transformed into a data model that includes a wider set of KPIs that characterize such processes at all the organizational levels. The applicability of this approach strictly depends on the willingness of the top management to participate to the design process and usually require the capability of the project staff in translating the collected high-level requirements into quantifiable KPIs. Goal-oriented approaches maximize the probability of a correct identification of the relevant indicators, thus reducing the risk of obsolescence of the multidimensional schema.

In many real cases the difference between adopting a goal-driven instead of a user-driven approach may become very vague, on the other hand, it should be clear that the goal-driven process is top-down and based on the progressive refinement of a few goals defined by the top-managements, while in the user-driven approach, requirements are obtained by merging several simpler requirements gathered from the business-users in a bottom-up fashion. The result of a goal-driven approach differs from a user-driven one whenever the users do not have a clear understanding of the business strategy and the organization's goals.

Table 1: Comparison of the basic user requirement techniques

	Supply-Driven	**User-Driven**	**Goal-Driven**
Basic approach	Bottom-up	Bottom-up	Top-Down
Users involvement	Low: DB Administrators	High: Business users	High: Top management
Constraints	Existence of a reconciled data level	Business users must have a good knowledge of the processes and organization of the company	Willingness of top management to participate in the design process
Strengths	The availability of data is ensured	Raise the acceptance of the system.	Maximize the probability of a correct identification of the relevant KPIs.
Risks	The multidimensional schemata do not fit business user requirements.	Quick obsolescence of the multidimensional schemata due to changes of the business users.	Difficulties in being supported by top management and in translating the business strategy into quantifiable KPIs.
Targeting organizational level	Operational and tactical	Depends on the level of the interviewed users, typically tactical	Strategic and tactical
Skills of project staff	DW designers	Moderators; DW designers	Moderators; Economist; DW designers
Risk of obsolescence	Low	High	Low
Number of source systems	Low	Moderate	High
Cost	Low	High	High

Table 1 reports a comparison of the three basic approaches and may be useful to choose the one that is most suited to a given project. The main technical element influencing such a choice concerns the availability and the quality of the schema of the operational data sources, while several non-technical factors are involved in the choice. In particular, the cost and time constraints suggest a reduction of the time devoted to interviews and discussions with the users; similarly when the business users have a limited knowledge of the business process and strategy a user-driven approach should be avoided.

In order to avoid the drawbacks of the single approaches some mixed strategies have been developed. In particular, Bonifati et al. (2001) mix the goal-oriented and the supply-driven techniques. They initially carry out a goal-driven step based on the GQM method (Vassiliadis et al. 1999; Basili et al. 1994) in order to identify the business needs the data mart is expected to meet. The outcome of this phase is a set of *ideal* star schemata obtained by a progressive top-down definition of the KPIs necessary to measure the goals. Then, the schema of the operational data sources are examined using a semi-automatic approach; the *candidate* facts, dimension and hierarchies are extracted and modelled using a graph-based representation (i.e. *star join graph*). Finally, the results of the two phases are integrated, thus determining the multidimensional model of the data mart as the set of candidate star join graphs that is most suited to matching the ideal star schemata.

Similarly, in GRAnD, the approach proposed by Giorgini et al. (2006), a goal-oriented step based on the Tropos methodology (Bresciani, 2004) is carried out in order to identify the business goals and the terms relevant to their monitoring. These terms are then mapped onto the operational source schema that is finally explored in a semi-automatic fashion in order to build the final multidimensional schemata.

A third approach coupling a goal-driven step with a supply-driven one has been proposed in (Mazon et al., 2007). The approach builds on a Model Driven Architecture (Mazon & Trujillo, 2008) where information is formalized using UML. In particular, project goals are modelled using a UML profile based on the *i** framework that has been properly adapted and extended to fit the DW specificities. Starting from the collected goals an initial conceptual model is obtained; then its correctness and feasibility is checked against data sources by using the Multidimensional Normal Forms (Lechtenboerger & Vossen, 2003).

The main difference between the three methods is that while in GRAnD the results of the goal-oriented step are used to drive the supply-driven one, in the other ones the goal-driven and the supply-driven steps are almost independent and they are used jointly a posteriori to verify the correctness of the conceptual model obtained.

Although, the formalisms used in the three approaches for the goal-oriented step are different, their expressivity is very close, showing that a core of common information to be captured has been identified. In particular, the second and the third ones are both based on *i** (Yu, 1995; Yu, 1997). In *i** (which stands for "distributed intentionality"), early requirements are assumed to involve social actors who depend on each other for goals to be achieved, tasks to be performed, and resources to be supplied. The *i** framework includes the strategic dependency model for describing the network of relationships among actors, as well as the strategic rationale model for describing and supporting the reasoning that each actor has about its relationships with other actors. These models have been formalized using intentional concepts such as goal, belief, ability, and commitment (Cohen & Levesque, 1990). The framework has been related to different application areas that, beside DW, include requirements engineering (Yu, 1993), business process reengineering (Yu & Mylopoulos, 1996), and software processes (Yu & Mylopoulos, 1994).

A further mixed method is the one proposed in (Guo, 2006) that puts together all the three basic approaches: a goal-oriented step determines the subject area of interest, defines the main KPIs and the target users. This information is exploited in the data- driven step in order to select the source systems and in the user-driven one to select the users to be interviewed. On the other hand, the results of these two steps refine and detail the results of the goal-oriented one and enable a more complete multidimensional model to be delivered.

Till now we considered techniques oriented to capturing *information* (functional) requirements for DWs. On the other hand, the final product of the DW design process is not just a data model but a whole DW system, where users require the information to have some characteristics when it is provided (security, performance tuning, user configurations, etc.). Here non-functional requirements come into play. Only a few works in the DW literature have specifically addressed this issue. In particular, Paim & Castro (2003) propose the Data Warehouse Requirements Definition (DWARF) approach that adapts a traditional requirements engineering process for requirements definition and management of DWs. DWARF requires particular attention to non-functional requirements that are captured through an ad-hoc extension of the NFR framework. The NFR framework (Chung et al., 1999) is a goal-oriented approach specifically devised for non-functional requirements that are considered as potentially conflicting or synergistic softgoals to be achieved. A softgoal represent a goal that has no clear-cut definition and/or criteria as to whether it is satisfied or not. A softgoal is said to be "satisfied" when there is sufficient positive evidence and little negative evidence against it. The same authors provide a detailed classification of non-functional requirements (Paim & Castro, 2002) that must be addressed in the development of DWs, and guidelines for their operationalization. At the top level requirements are clustered into four classes, namely: performance, security, multidimensionality and user-friendliness.
Soler et al. (2008) investigate the security aspects by integrating a non-functional requirement analysis step into an existing DW approach for information requirements (Mazon & Trujillo, 2008).

CONCEPTUAL MODELS

Conceptual modeling is widely recognized to be the necessary foundation for building a database that is well-documented and fully satisfies the user requirements. In particular, from the designer point of view the availability of a conceptual model provides a higher level of abstraction in describing the warehousing process and its architecture in all its aspects.

In the last few years multidimensional modeling has attracted the attention of several researchers that defined different solutions each focusing on the set of information they considered strictly relevant. Some of these solutions have no (Agrawal et al., 1997; Pedersen & Jensen, 1999) or limited (Cabibbo & Torlone, 1998) graphical support, and are aimed at establishing a formal foundation for representing cubes and hierarchies and an algebra for querying them. On the other hand, we believe that a distinguishing feature of conceptual models is that of providing a graphical support to be easily understood by both designers and users when discussing and validating requirements. So we will classify "strict" conceptual models for DWs according to the graphical formalism they rely on, that could be either E/R, object-oriented or ad hoc. Some claim that E/R extensions should be adopted since:

1. E/R has been tested for years;
2. designers are familiar with E/R;
3. E/R has proved to be flexible and powerful enough to adapt to a variety of application domains
4. several important research results were obtained for the E/R model (Franconi & Kamble, 2004; Sapia et al., 1999; Tryfona et al., 1999).

On the other hand, advocates of object-oriented models argue that:

1. they are more expressive and better represent static and dynamic properties of information systems;
2. they provide powerful mechanisms for expressing requirements and constraints;
3. object-orientation is currently the dominant trend in data modeling;
4. UML, in particular, is a standard and is naturally extensible (Luján-Mora, 2006; Abello, 2006).

Finally, we believe that ad hoc models compensate for designers' lack of familiarity since:

1. they achieve better notational economy;
2. they give proper emphasis to the peculiarities of the multidimensional model;
3. they are more intuitive and readable by non-expert users (Golfarelli, 2008; Hüsemann et al., 2000; Tsois et al. 2001).

Remarkably, a comparison of the different models made by Abello et al. (2006) pointed out that, abstracting from their graphical form, the core expressivity of most of the conceptual models proposed in the literature is similar, thus proving that the academic community has reached an informal agreement on the required expressivity (see Figure 1).

We emphasize that, within the DW field, conceptual models and formal user-requirement techniques are rarely discussed together to form a comprehensive methodology. Furthermore, even in these cases (Bonifati et al., 2001; Giorgini et al., 2007; Guo, 2006; Mazon et al., 2007), non-functional requirements are almost neglected or they have been presented as a second class type of requirement, frequently hidden inside notes. On the other hand, the experiences in the broader area of software engineering show that capturing non-functional requirements without mapping them into the conceptual model may determine an information loss. In (Cysneiros & Sampaio do Prado Leite, 2004) the authors show how to integrate non-functional requirements into the Class, Sequence, and Collaboration UML Diagrams. The elicitation of non-functional requirements at the conceptual level enables a traceability mechanism. This mechanism provides a way of representing in the models, which aspects are there because of a non-functional requirement. This has shown to be quite useful during the model reviewing process. In different situations, the reviewers were surprised by the inclusion of elements in the conceptual models that did not fit their perception of the application; they only became convinced of the necessity by following the traces to the NFR graphs. The traceability mechanism has also proved to be very helpful when evaluating if a non-functional requirement was satisfied or not since it was easier to check the models to see what possible impacts would arise from dropping one non-functional requirement or satisfying another.

To the best of our knowledge the only conceptual model that considers non-functional requirements in the area of DW is the one by Fernández-Medina et al. (2006), that, limitedly to security requirements, set out an Access Control and Audit model that allows the designer to specify access control and audit considerations when carrying out the conceptual modeling phase.

A different proposal comes from Peralta et al., (2003) that propose modelling the non-functional requirements through guidelines that are not directly related with the conceptual model but are instead exploited during logical design, where most of the choices related to performance (e.g. star vs snowflake schema, view materialization) and security are kept.

In the rest of this section we will present the Dimensional Fact Model (DFM) as a representative in order to give the reader a clear understanding of the required expressiveness and in order to determine the core of information that must be collected, during the requirement analysis phase, to allow effective multidimensional modelling in the next phase.

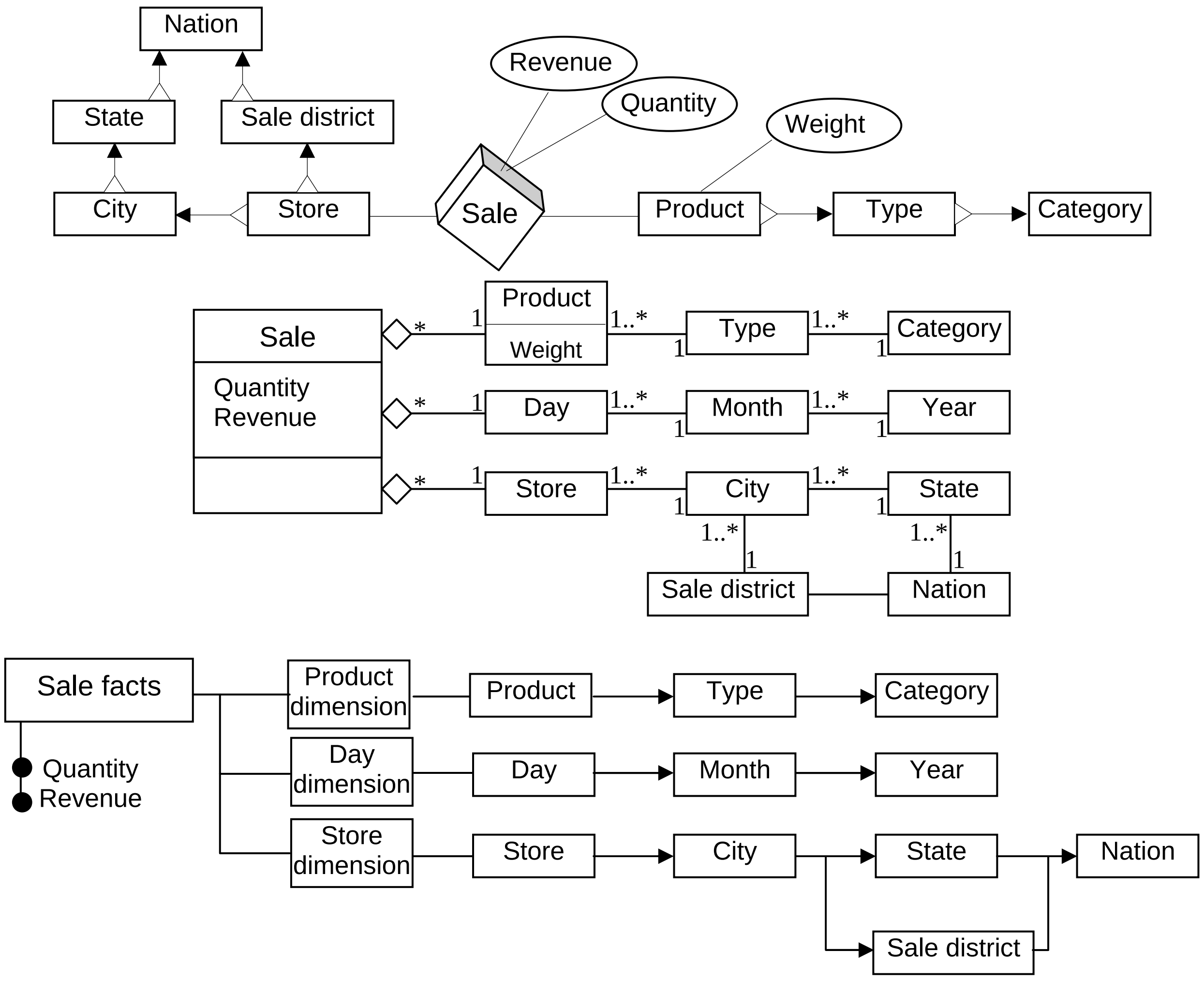

Figure 1: the SALE fact modeled, from the top to the bottom, using a StarER (Sapia et al., 1999), a UML class diagram (Luján-Mora et al., 2006), and a fact schema (Hüsemann et. al, 2000).

The Dimensional Fact Model

The DFM is a graphical conceptual model, specifically devised for multidimensional design, aimed at:

- effectively supporting conceptual design;
- providing an environment in which user queries can be intuitively expressed;

- supporting the dialogue between the designer and the end-users to refine the specification of requirements;
- creating a stable platform to ground logical design;
- providing an expressive and non-ambiguous design documentation.

DFM was first proposed in 1998 by Golfarelli and Rizzi and continuously enriched and refined during the following years in order to optimally suit the variety of modeling situations that may be encountered in real projects of small to large complexity.

The representation of reality built using the DFM consists of a set of fact schemata. The basic concepts modelled are facts, measures, dimensions, and hierarchies. In the following we intuitively define these concepts, referring the reader to Figure 2 that depicts a simple fact schema for modelling invoices at line granularity; a formal definition of the same concepts can be found in (Golfarelli, 2008).

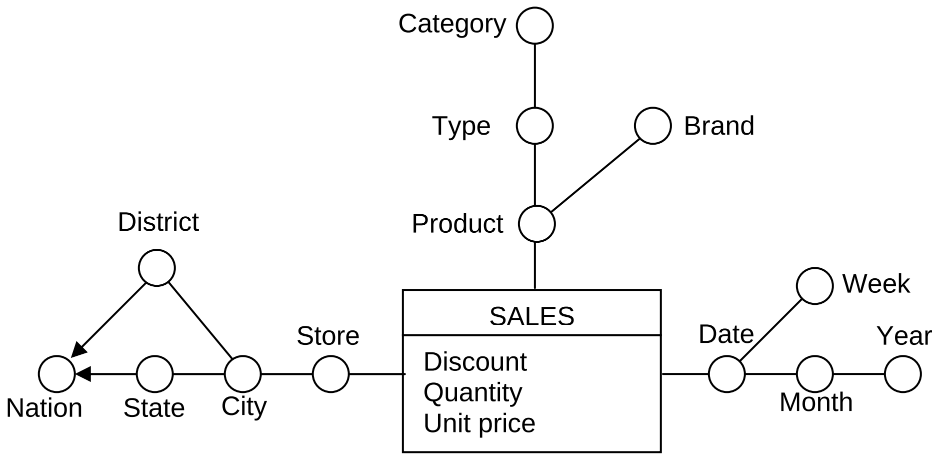

Figure 2: A basic fact schema for the SALES fact.

A *fact* is a focus of interest for the decision-making process; typically, it models a set of events occurring in the enterprise world. A fact is graphically represented by a box with two sections, one for the fact name and one for the measures. Examples of facts in the trade domain are sales, shipments, purchases; in the financial domain: stock exchange transactions, contracts for insurance policies. It is essential for a fact to have some dynamic aspects, i.e., to evolve somehow across time. The concepts represented in the data source by frequently-updated archives are good candidates for facts; those represented by almost-static archives are not. As a matter of fact, very few things are completely static; even the relationship between cities and regions might change, if some borders were revised. Thus, the choice of facts should be based either on the average periodicity of changes, or on the specific interests of analysis.

A *measure* is a numerical property of a fact, and describes one of its quantitative aspects of interests for analysis. Measures are included in the bottom section of the fact. For instance, each invoice line is measured by the number of units sold, the price per unit, the net amount, etc. The reason why measures should be numerical is that they are used for computations. A fact may also have no measures, if the only interesting thing to be recorded is the occurrence of events; in this case the fact schema is said to be empty and is typically queried to count the events that occurred.

A *dimension* is a fact property with a finite domain, and describes one of its analysis coordinates. The set of dimensions of a fact determine its finest representation granularity. Graphically, dimensions are represented as circles attached to the fact by straight lines. Typical dimensions for the invoice fact are product, customer, agent. Usually one of the dimensions of the fact represents the time (at any granularity) that is necessary to extract time series from the DW data.

The relationship between measures and dimensions is expressed, at the instance level, by the concept of event. A *primary event* is an occurrence of a fact, and is identified by a tuple of values, one for each dimension. Each primary event is described by one value for each measure. Primary events are the elemental information which can be represented (in the cube metaphor, they correspond to the cube cells). In the invoice example they model the invoicing of one product to one customer made by one agent on one day.

Aggregation is the basic OLAP operation, since it allows significant information to be summarized from large amounts of data. From a conceptual point of view, aggregation is carried out on primary events thanks to the definition of dimension attributes and hierarchies. A *dimension attribute* is a property, with a finite domain, of a dimension. Like dimensions, it is represented by a circle. For instance, a product is described by its type, category, and brand; a customer, by its city and its nation.

The relationships between dimension attributes are expressed by hierarchies. A *hierarchy* is a directed graph, rooted in a dimension, whose nodes are all the dimension attributes that describe that dimension, and whose arcs model many-to-one associations between pairs of dimension attributes. Arcs are graphically represented by straight lines. Hierarchies should reproduce the pattern of inter-attribute functional dependencies expressed by the data source. Hierarchies determine how primary events can be aggregated into secondary events and selected significantly for the decision-making process. Given a set of dimension attributes, each tuple of their values identifies a *secondary event* that aggregates all the corresponding primary events. Each secondary event is described by a value for each measure, that summarizes the values taken by the same measure in the corresponding primary events.

The dimension in which a hierarchy is rooted defines its finest aggregation granularity, while the other dimension attributes progressively define coarser ones. For instance, thanks to the existence of a many-to-one association between products and their categories, the invoicing events may be grouped according to the category of the products. When two nodes a_1, a_2 of a hierarchy share the same descendent a_3 (i.e. when two dimension attributes within a hierarchy are connected by two or more alternative paths of many-to-one associations) this is the case of a *convergence*, meaning that for each instance of the hierarchy we can have different values for a_1, a_2, but we will have only one value for a_3. For example, in the geographic hierarchy on dimension customer (Figure 2): customers live in cities, which are grouped into states belonging to nations. Suppose that customers are also grouped into sales districts, and that no inclusion relationships exist between districts and cities/states; on the other hand, sales districts never cross the nation boundaries. In this case, each customer belongs to exactly one nation whichever of the two paths is followed (customer$\rightarrow$city$\rightarrow$state$\rightarrow$nation or customer$\rightarrow$sale district$\rightarrow$nation).

It should be noted that the existence of apparently equal attributes does not always determine a convergence. If in the invoice fact we had a brand city attribute on the product hierarchy, representing the city where a brand is manufactured, there would be no convergence with attribute (customer) city, since a product manufactured in a city can obviously be sold to customers of other cities as well.

SOLUTIONS AND RECOMMENDATIONS

Requirement analysis and conceptual design are to a great extent responsible for the success of a DW project since, during these two phases, the expressivity of the multidimensional schemata is completely defined. The paper has shown how most of the risk factors can be handled by the adoption of proper methodologies and formalisms. In particular, though some basic approaches to user-requirement analysis are available in the literature the adoption of a “pure” one is not sufficient to protect from its own weaknesses. On the other hand, some authors have proposed, and tested in real projects, mixed techniques proving that the three basic approaches are not mutually exclusive but, as evidenced by (List, 2002), are instead complementary and when used in parallel may overcome many of the problems. In particular, we believe that coupling a data-driven step with a demand-driven one can lead to a design capturing all the specifications and ensuring a higher level of longevity as well as acceptance of the users. Choosing whether the demand-driven step should actually follow the goal-driven or the user-driven approach mainly depends on the project peculiarities that include several factors like budget and time constraints, company environment and culture, goal of the project, etc.

From the analysis of the methodological steps the extent to which the two design phases studied in this paper are related clearly emerges and their synergy can be exploited at best if the two steps share a common and well-structured formalism that facilitates and disambiguates the exchange of information. Whenever a demand-driven step is adopted we recommend the adoption of formalisms that are intuitive and readable by non-expert users in order to facilitate their interaction with the designers. For this reason we believe that ad-hoc formalism should be preferred whenever business-users are involved.

FUTURE TRENDS

While the topic of conceptual modelling in DWing has been widely explored, the subject of user requirement analysis as well as the relationships between these design steps has been only partially studied. In particular, while the way for capturing functional requirements has been paved, techniques for non-functional requirements have been just sketched and still not massively tested in real projects. Thus, we believe that the effort in this area should be twofold: on the one hand, a lot of research is still needed to obtain a comprehensive approach to user requirement analysis, on the other hand, the effectiveness of this step can be exploited at best only if it can be coupled with the conceptual design phase to form a unique framework.

From the practitioners' point of view, we emphasize that the adoption of a structured approach during user requirements analysis and conceptual design is still extremely limited in real projects (Sen & Sinha, 2005), while the need for solutions that reduce the design efforts and that lower, at the same time, the failure risk is strongly felt. This gap is probably due to a lack of commercial design software for DWs. In fact, 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, (Golfarelli & Rizzi, 2001; Jensen et al. 2004), 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 (Trujillo, 2002) 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 cubes.

CONCLUSIONS

In this paper we have surveyed the state of the art of the literature related to the first steps of data mart design, namely user requirement analysis and conceptual design. The approaches to requirement analysis have been surveyed and their strengths and weaknesses have been discussed in order to enable the designer to choose the more appropriates for a given project. Similarly, the main conceptual models for DWs have been reported and their basic concepts have been discussed using DFM as a representative.

From our analysis it emerges that although it is evident that the two design steps are strictly related, no comprehensive approach has been devised yet. On the other hand, the research community seems to share some common ideas: the core expressivity of the conceptual models for DW is shared by most of the models discussed regardless of the formalism adopted; furthermore, as for user requirement analysis, several authors chose a mixed method coupling a goal-driven step with a supply-driven one.

REFERENCES

Abello, A., Samos, J., & Saltor, F. (2006). YAM^2: a multidimensional conceptual model extending UML. *Information System*, 31(6), 541-567.

Agrawal, R., Gupta, A., & Sarawagi, S. (1997). Modeling multidimensional databases. In *Proceedings International Conference on Data Engineering*, Birmingham U.K., 232-243.

Basili, V. R., Caldiera, G., & Rombaci, H. D. (1994). The Goal Question Metric Approach. *Encyclopedia of Software Engineering* - 2 Volume Set, John Wiley & Sons, Inc. pp 528-532.

Boehnlein, M., & Ulbrich vom Ende, A. (2000). Business process oriented development of data warehouse structures. In *Proceeding Data Warehousing 2000*, Physica Verlag.

Bonifati, A., Cattaneo, F., Ceri, S., Fuggetta, A., & Paraboschi, S. (2001). Designing data marts for data warehouses, *ACM Transactions on Software Engineering Methodologies* 10(4), 452–483.

Bresciani, P., Giorgini, P., Giunchiglia, F., Mylopoulos, J., & Perini, A. (2004). Tropos: an agent-oriented software development methodology. *Journal of Autonomous Agents and Multi-Agent Systems* 8(3), 203–236.

Cabibbo, L., & Torlone, R. (1998). A logical approach to multidimensional databases. In *Proceedings International Conference on Extending Database Technology*.Valencia, Spain, 183-197.

Chung, L., Nixon, B.A., Yu, E., & Mylopoulos, J. (1999). *Non-Functional Requirements in Software*. Springer.

Cohen, P., & Levesque, H. (1990). Intention is Choice with Commitment. *Artificial Intelligence*, 32(3), 213-261.

Cysneiros, L.M., & Sampaio do Prado Leite, J. C. (2004). Nonfunctional Requirements: from Elicitation to Conceptual Models. *IEEE Transaction on Software Engineering,* 30(5), 328-350.

Demarest, M. (1997). The politics of data warehousing. Retrieved August 2008 from http://www.noumenal.com/marc/dwpoly.html.

Fernández-Medina, E., Trujillo, J., Villarroel, R., & Piattini, M. (2006). Access control and audit model for the multidimensional modeling of data warehouses. *Decision Support Systems,* 42(3), 1270-1289.

Franconi, E., & Kamble, A. (2004). A data warehouse conceptual data model. In *Proceedings International Conference on Statistical and Scientific Database Management*. Santorini Island, Greece, 435-436.

Golfarelli, M. (2008). The DFM: A Conceptual Model for Data Warehouse. *Encyclopedia of Data Warehousing and Mining* (Second Edition), John Wang (Ed.), IGI Global.

Giorgini, P., Rizzi, S., & Garzetti, M. (2007). GRAnD: A goal-oriented approach to requirement analysis in data warehouses. *Decision Support System*, vol. 45(1), 4-21.

Golfarelli, M., Maio, D., & Rizzi, S. (1998). The Dimensional Fact Model: A Conceptual Model for Data Warehouses. *International Journal of Cooperative Information Systems*, 7(2-3), 215-247.

Golfarelli, M., & Rizzi, S. (2001). WanD: A CASE Tool for Data Warehouse Design. In *Demo Proceedings 17th International Conference on Data Engineering (ICDE 2001),* Heidelberg, Germany, 7-9.

Golfarelli, M., & Rizzi, S. (2009). A Survey on Temporal Data Warehousing. *International Journal of Data Warehousing and Mining (IJDWM)* 5(1), 1-17.

Guo, Y., Tang, S., Tong, Y., & Yang, D. (2006). Triple-driven data modeling methodology in data warehousing: a case study. In *Proceedings ACM International Workshop on Data Warehousing and OLAP*, 59-66.

Hüsemann, B., Lechtenbörger, J., & Vossen, G. (2000). Conceptual data warehouse design. In *Proceedings International Workshop on Design and Management of Data Warehouses*. Stockholm, Sweden, 3-9.

Jensen, M., Holmgren, T., & Pedersen, T. (2004). Discovering Multidimensional Structure in Relational Data. In *Proceedings of International Conference on Data Warehousing and Knowledge Discovery*, Zaragoza, Spain, 138-148.

Kimbal, R., Reeves, L., Ross, M., & Thornthwaite, W. (1998). The Data Warehouse Lifecycle Toolkit. *John Wiley and Sons, Inc*., N.Y..

Lechtenboerger, J., & Vossen, G. (2003). Multidimensional normal forms for data warehouse design. *Information Systems*, 28(5), 415–434.

List, B., Bruckner, R., Machaczek, K. & Schiefer, J. (2002). A comparison of data warehouse development methodologies: case study of the process warehouse. In *Proceedings International Conference on Database and Expert Systems Applications*, 203–215.

Luján-Mora, S., Trujillo, J., & Song, I. Y. (2006). A UML profile for multidimensional modeling in data warehouses. *Data & Knowledge Engineering*, 59(3), 725-769.

Mazon, J. N., & Trujillo, J. (2008). An MDA approach for the development of data warehouses. *Decision Support Systems*, 45(1), 41-58.

Mazon, J. N., Trujillo, J., & Lechtenboerger, J. (2007). Reconciling requirement-driven data warehouses with data sources via multidimensional normal forms. Data & Knowledge Engineering, 63 725-751

Mylopoulos, J., Chung, L., & Nixon, B. (1992). Representing and Using Non-Functional Requirements: A Process-Oriented Approach. *IEEE Transactions on Software Engineering,* 18(6) 483-497.

Paim, F.R.S., & Castro, J. (2002). Enhancing Data Warehouse Design with the NFR Framework. In *Proceeding of the Workshop em Engenharia de Requisitos*. Valencia, Spain, 40–57.

Paim, F.R.S., & Castro, J. (2003). DWARF: An approach for requirements definition and management of data warehouse systems. In *Proceeding 11th IEEE International Conference on Requirements Engineering*. Monterey Bay, USA, 75–84

Pedersen, T. B., & Jensen, C. (1999). Multidimensional data modeling for complex data. In *Proceedings International Conference on Data Engineering*. Sydney, Australia, 336-345.

Peralta, V., Illarze, A., & Ruggia, R. (2003). On the Applicability of Rules to Automate Data Warehouse Logical Design. In *Proceedings Decision Systems Engineering Workshop*. Klagenfurt, Austria, pp, 317-328.

Sapia, C., Blaschka, M., Hofling, G., & Dinter, B. (1999). Extending the E/R model for the multidimensional paradigm. Lecture Notes in Computer Science, 1552, 105-116.

Sen, A., & Sinha A.P. (2005). A Comparison of Data Warehousing Methodologies. *Communications of the ACM,* 48(3), 79-84.

Soler, E., Stefanov, V., Mazon, J. N., Trujillo J., Fernandez-Medina, E.,& Piattini, M. (2008). Towards Comprehensive Requirement Analysis for Data Warehouses: Considering Security Requirements. In *Proceedings of the Third International Conference on Availability, Reliability and Security - ARES 2008,* Barcelona, Spain, pp, 4-7.

Trujillo, J., Luján-Mora, S., & Medina, E. (2002). The Gold model case tool: An environment for designing OLAP applications. In *Proceedings of International Conference on Enterprise Information Systems*, Ciutad Real, Spain, 699-707.

Tryfona, N., Busborg, F., & Borch Christiansen, J. G. (1999). starER: a conceptual model for data warehouse design. In *Proceedings ACM International Workshop on Data Warehousing and OLAP*. Kansas City, USA, 3-8.

Tsois, A., Karayannidis, N., & Sellis, T. (2001). MAC: Conceptual data modeling for OLAP. In *Proceedings International Workshop on Design and Management of Data Warehouses*. Interlaken, Switzerland, 5.1-5.11.

Vassiliadis, P., Bouzeghoub, M., & Quix, C. (1999). Towards Quality-Oriented Data Warehouse Usage and Evolution. In *Proceedings of the International Conference on Advanced Information Systems Engineering*. Heidelberg, Germany, 149-163.

Winter, R., & Strauch, B. (2003). A method for demand-driven information requirements analysis in data warehousing. In *Proceedings of Hawaii International Conference on System Sciences*, Hawaii, 1359-1365.

Yu, E. (1993). Modeling Organizations for Information Systems Requirements Engineering. In *Proceedings First IEEE International Symposium on Requirements Engineering*, San Jose, USA, 34-41.

Yu, E. (1995). Modelling Strategic Relationships for Process Reengineering, *Ph.D. thesis, Department of Computer Science,* University of Toronto, 1995.

Yu, E. (1997). Towards modeling and reasoning support for early-phase requirements engineering, In *Proceedings 3rd IEEE International Symposium on Requirements Engineering,* Annapolis, USA.

Yu, E., & Mylopoulos, J. (1994). Understanding 'Why' in Software Process Modeling, Analysis and Design. In *Proceedings Sixteenth International Conference on Software Engineering,* Sorrento, Italy

Yu, E., & Mylopoulos, J. (1996) Using Goals, Rules, and Methods to Support Reasoning in Business Process Reengineering. *International Journal of Intelligent Systems in Accounting, Finance and Management,* 5(1), 1-13.

KEY TERMS & DEFINITIONS

Conceptual model: a formalism, with a given expressivity, suited for describing part of the reality, based on some basic constructs and a set of logical and quantitative relationships between them.
ETL - Extraction Transformation and Loading: is the process that enables data to be loaded in the DW. It is usually carried out using specialized software and entails extracting data out of the sources, transforming it to fit the business needs and to match the quality requirements, and finally loading it into the end target.
KPI – Key Performance Indicator: are financial and non-financial metrics used to help an organization define and measure progress toward organizational goals
Life-cycle: the set of phases a software system usually goes through from its conception to its retirement.
Multidimensional model: is a data model optimized for data access in a way that comes natural to human analysts. A multidimensional model is centred on a fact that is a focus of interest for the decision-making process and can be monitored through measures and dimensions.
User Requirements: the needs of the of the stakeholders who directly interact with the system