GRAnD: A goal-oriented approach to requirement analysis in data warehouses

— Source link ⧉

Paolo Giorgini, Stefano Rizzi, Maddalena Garzetti

Institutions: University of Trento, University of Bologna

Published on: 01 Apr 2008 - Decision Support Systems

Topics: User requirements document, Data warehouse and Requirements analysis

Related papers:

- Designing data marts for data warehouses

- A model-driven goal-oriented requirement engineering approach for data warehouses

- A method for demand-driven information requirements analysis in data warehousing projects

- Building the data warehouse

- The dimensional fact model: a conceptual model for data warehouses

GRAnD: A Goal-Oriented Approach
to Requirement Analysis in Data Warehouses

Paolo Giorgini   Stefano Rizzi   Maddalena Garzetti

Abstract

Several surveys indicate that a significant percentage of data warehouses fail to meet business objectives or are outright failures. One of the reasons for this is that requirement analysis is typically overlooked in real projects. In this paper we propose GRAnD, a goal-oriented approach to requirement analysis for data warehouses based on the Tropos methodology. Two different perspectives are integrated for requirement analysis: organizational modeling, centered on stakeholders, and decisional modeling, focused on decision makers. Our approach can be employed within both a demand-driven and a mixed supply/demand-driven design framework.

1 Introduction

Data warehouse (DW) projects are inherently risky. In confirmation of this, several surveys indicate that a significant percentage of data warehouse projects fail to meet business objectives or are outright failures. The reasons for a failure are often related to how the organization reacts to the project or to the low quality of data, but in several cases they may be traced back to how the design process was conducted. In particular, a common design-related reason for failure is that requirement analysis is typically overlooked in DW projects, mainly since [1]:

- Warehousing projects are long-term ones, and it is very difficult to anticipate future requirements, so only a few requirements can be stated from the beginning.

- Information requirements for DW applications are difficult to specify since decision processes are flexibly structured, poorly shared across large organizations, jealously guarded by managers, and unstable in time to keep pace with evolving business processes.

- Requirements for decision making often refer to information that does not exist in the required form, and must be derived from data sources by integrating, transforming, and cleaning them.

Most methodologies for DW design claim there must be a phase dedicated to analyzing the business requirements (e.g., [2, 3, 4, 5]), still there is no consensus on what relevance and temporal priority should be assigned to it. Indeed, the approaches to DW design are usually classified in two categories [1]:

- *Supply-driven* (also called *data-driven*) approaches design the DW starting from a detailed analysis of the data sources [6, 7, 8]. User requirements impact on design by allowing the designer to select which chunks of data are relevant for the decision making process and by determining their structuring according to the multidimensional model.

- *Demand-driven* (or *requirement-driven*) approaches start from determining the information requirements of business users [9, 10]. The problem of mapping these requirements onto the available data sources is faced only *a posteriori*, by designing proper ETL routines.

While supply-driven approaches somehow simplify the design of ETL, since each 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, and give the designer little support in identifying facts, dimensions, and measures. Conversely, demand-driven approaches bring requirements to the foreground, thus ensuring that the DW will be tightly tailored to the users' needs, but require a larger effort when designing ETL.

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; (3) the complexity of source schemata is not too high. In practice, when the chosen architecture for the DW relies on a *reconciled level* (or operational data store) these requirements are largely satisfied: in fact, normalization and detailed knowledge are guaranteed by the source integration process. The same holds, thanks to a careful source recognition activity, in the frequent case when the source is a single relational database, well-designed and not very large. In a supply-driven approach, conceptual design is heavily rooted on source schemata and can be largely automated (e.g. see [7]). Our on-the-field experience shows that requirement analysis can then be carried out informally, based on simple requirement glossaries (such as in [11]) rather than on formal diagrams. On the other hand, such an informal approach is unsuitable for a demand-driven framework, that asks for a more structured and comprehensive technique.

In this paper we propose GRAnD (Goal-oriented Requirement Analysis for Data warehouses), a goal-oriented technique for requirement analysis in DWs based on the Tropos methodology [12]. GRAnD can be employed:

- within a demand-driven framework, that is the only alternative whenever a deep analysis of data sources is unfeasible, or data sources reside on legacy systems whose inspection and normalization is not recommendable. In this case, conceptual design will be directly based on requirements.

- within a mixed supply/demand-driven framework. In this case, requirement analysis and source inspection are carried out in parallel; conceptual design is still carried out in a semi-automated way, like in the supply-driven framework, but leaning on user requirements to reduce its complexity. The mixed framework is recommendable when source schemata are well-known but their size and complexity are substantial. In fact, the cost for a more careful and formal analysis of requirement is balanced by the quickening of conceptual design.

GRAnD adopts two different perspectives for requirement analysis: *organizational modeling*, centered on stakeholders, and *decisional modeling*, focused on decision makers. Decisional modeling is directly related to the information needs of decision makers; with reference to the terminology introduced in [1], it achieves *to be* analysis. On the other hand, organizational modeling is aimed at *as is* analysis; it has a primary role in enabling identification of facts and in supporting the supply-driven component of the approach. The diagrams produced, that relate enterprise goals to facts, dimensions, and measures, are then used during conceptual design: within a demand-driven design framework, the requirements are translated into a conceptual schema to be mapped on data sources *a posteriori*; within a mixed framework, while the data sources are still explored to shape hierarchies, user requirements play a fundamental role in restricting the area of interest for analysis and in determining facts, dimensions, and measures.

The paper is structured as follows. In Section 2 we summarize the most relevant literature related to requirement analysis in DW design. Section 3 illustrates the technique we propose for requirement analysis by discussing organizational and decisional modeling. Section 4 shows how our technique results in conceptual design within both demand-driven and mixed design frameworks. Section 5 introduces the prototype supporting our approach. Finally, Section 6 draws the conclusions.

2   Related Literature

In the field of DW design, it is necessary to distinguish between supply- and demand-driven approaches. The prototypical supply-driven approach dates back to 1992, when Inmon claimed that the development of DWs is data-driven, as opposed to the requirement-driven development of operational systems [13]. Other supply-driven approaches were proposed in [6], [7], and [8], where conceptual design of the DW is rooted in the schema of operational sources and is carried out starting, respectively, from the identification of measures, from the selection of facts, and from a classification of the operational entities. Also the comprehensive design method described in [4] leans on a conceptual model; a

mixed approach to conceptual design is recommended, but no further details are given.

In demand-driven approaches, collecting user requirements is given more relevance. The approach described in [14] shares some similarities with ours, since it is based on the i^* framework. However, requirements are directly used to build a conceptual model in a fully demand-driven perspective, without any feedback from the data sources. Besides, organizational modeling is not supported. In [1], a wish-list for DW design methodologies is proposed, and a multi-stage technique for requirement analysis is outlined. Here, two different phases are interlaced: *as is analysis*, aimed at analyzing and describing the actual information supply, and *to be analysis*, aimed at analyzing the information demand and matching it with the supply. In [9], a goal-oriented approach based on the *goal-decision-information* model is proposed. Though this approach shares some similarities with ours, it mainly focuses on requirement analysis and does not show how to move from requirements to design. A process-oriented approach is presented in [10], where three different perspectives at increasing levels of detail, each associated to a specific requirement template, are used. Though the authors recommend to iteratively and incrementally gather requirements with use cases, a few details are given and no examples are provided, so a comparison is very hard. A comprehensive framework for requirement management, based on the Unified Process, is proposed in [5]; though there is a strong emphasis on documenting the project through *document templates*, nothing is said about how to model requirements.

In [15], a goal-oriented method to support the identification and design of DWs is presented. This approach can be regarded, like ours, as mixed demand/supply driven. The main difference is that organizational modeling is not supported and that requirement analysis starts from the goals of decision makers. Goals are analyzed separately using abstractions sheets, and general considerations about how they relate to the organization activities are given in natural language. Conversely, in our approach an explicit goal model of the organization is given and the analysis of decision makers' goals is directly related to such a model. Moreover, in our goal analysis, goals are decomposed in subgoals and

specific relationships between goals are specified. Another important difference with our approach, is that we support early requirements analysis [16, 17] that allows for modeling and analyzing processes that involve multiple participants (both humans and software systems) and the intentions that these processes are supposed to fulfill. By so doing, one can relate the functional and non-functional requirements of the system-to-be to relevant stakeholders and their intentions.

An interesting case-based comparison of supply- and demand-driven approaches can be found in [18]. Remarkably, it is concluded that data-oriented and goal-oriented techniques are complementary, and may be used in parallel to achieve optimal design.

Finally, it is worth to mention that a few CASE tools for DW design have been implemented, either from software vendors or as research protoypes. In ADAPT [19] and in GOLD [20] the conceptual schema for the DW is directly drawn by the designer, thus a demand-driven approach is implied – though no active support for requirement analysis is given. Conversely, in WAND [21] the conceptual schema is semi-automatically derived from the source schemata, thus implementing *de facto* a supply-driven approach.

3   Requirement Analysis with GRAnD

Tropos [12, 22] is an agent-oriented software development methodology, based on the i^* conceptual framework [17], where the concepts of *agent*, *goal*, and related mentalistic notions are used to support all software development phases, from early requirement analysis to implementation. Tropos differs from other goal-oriented methodologies since it moves the notions of agent and goal to the early stages of software development. During early requirement analysis, the requirements engineer identifies the domain stakeholders and models them as social actors, who depend on one another for goals to be fulfilled, tasks to be performed, and resources to be furnished. Through these dependencies, one can answer *why* questions, besides *what* and *how*, regarding system functionality. Answers to *why* questions ultimately link system functionality to stakeholder needs, preferences, and objectives.

The Tropos methodology has been successfully applied in different domains. In the following we summarize the part of the Tropos notation that can be used in the DW context:

- *Actors.* An actor represents an enterprise stakeholder. More precisely, it can model a physical or software *agent* (e.g., Mr. Brown), a *role*, meant as an abstract characterization of the behavior one or more agents take in a specific context (e.g., sale analyst), or a *position*, i.e. a set of roles generally played by a single agent (e.g., marketing manager). Graphically, actors are represented by circles.

- *Dependencies.* A dependency represents an "agreement" between two actors, one depending on the other to respect the agreement. The agreement can be a goal to be fulfilled, a task to be performed, or a resource to be delivered. In our context, the main interest is on *goals*, that are represented as ovals.

- *Actor diagram.* It is a graph of actors related by dependencies, used to model how actors depend on each other.

- *Rationale diagram.* It is used to represent the logical foundations that rule the relationships between actors. It appears as a balloon within which goals of a specific actor are analyzed and dependencies with other actors are established. Goals are decomposed into subgoals, with either AND (all subgoals must be achieved) or OR (any of the subgoals must be achieved) semantics, possibly specifying the positive/negative contributions of subgoals to goals. The intuitive meaning of a positive (negative) contribution is that the satisfaction of a goal encourages (discourages) the satisfaction of another goal. Notations + and ++ (− and − −) specify the different strength of the contribution.

When analyzing user requirements for DWs, two perspectives should be taken into account. Firstly, it is important to model and analyze the organizational setting in which the DW will operate (*organizational modeling*); this includes designing the actor diagram as well as the rationale diagrams for each stakeholder. Secondly, in order to capture the

7

functional and non-functional requirements of the DW, we need to design rationale diagrams for the decision makers, who are the main actors in the decisional process (*decisional modeling*).
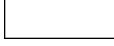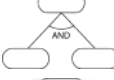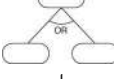
In the following subsections these two perspectives are described in detail, together with the analysis phases they encompass, with reference to real case study, the BI-BANK project, developed at the University of Trento in collaboration with *DeltaDator S.p.a.*. BI-BANK is a project for developing a Banking Business Intelligence System able to support the decisional process with a set of basic banking analyses. For simplicity, in this paper we only focus on analysis of banking transaction.

As concerns notation, using Tropos in the DW context requires some new concepts to be introduced:

- *Facts*. In organizational modeling, a fact models a set of events that happen when a goal is achieved. In decisional modeling, a fact is more properly meant as a possible focus of analysis related to an analysis goal. Graphically, facts are represented as rectangles connected to a goal.

- *Attributes*. They are fields whose value is provided when a fact is recorded to fulfill a goal. They are denoted as small diamonds connected to goals.

- *Dimensions*. A dimension is a fact property that describes a possible coordinate of analysis, i.e. a possible perspective for looking at the fact to fulfill an analysis goal. Dimension are represented as small circles connected to goals.

- *Measures*. A measure is a numerical property of a fact that describes a quantitative aspect that is relevant for decision making. Graphically, measures are represented as small squares connected to goals.

The graphical notation is summarized in Table 1.

Table 1: Notation for actor and rationale diagrams

Symbol	Meaning
	actor
	goal
	dependency
	fact
	AND decomposition
	OR decomposition
	attribute
	dimension
	measure

3.1 Organizational Modeling

Organizational modeling consists of three different phases: (i) *goal analysis*, in which actor and rationale diagrams are produced; (ii) *fact analysis*, in which rationale diagrams are extended with facts; and (iii) *attribute analysis*, in which rationale diagrams are further extended with attributes. Each phase is a different iterative process taking in input the diagrams produced by the previous one.

3.1.1 Goal Analysis

The first step for goal analysis is to represent the relevant stakeholders for the organization and their dependencies by means of an actor diagram, in which actors can represent agents, roles, or positions within the organization. This starts with a very high-level analysis aimed at expressing the responsibilities and relationships involving the different components (e.g., departments) of an organization. Then, the analysis proceeds in more detail by decomposing the high-level actors in sub-actors (e.g., the divisions of a department) and ending up with the identification of the agent(s) who is responsible for each

single activity. The analysis is conducted by interviewing the stakeholders and producing a documentation organized in a number of templates. Three different type of templates are provided:

- *main actor*, where the main actors of the analyzed environment are identified together with their strategic objectives and goals. A table of the form (`Actor,Objectives`) is used to collect the information.

- *sub-actor*, where each main actor is decomposed into a number of sub-actors (including roles, positions, and agents). For each main actor, a table of the form (`Sub-actor,Type,Goals`) is used to collect the information.

- *dependencies*, where all the dependencies among actors are identified and analyzed. A table of the form (`Depender, Dependee, Goal`) is used to collect the information.

The second step consists in analyzing goals of each actor in more detail to produce a rationale diagram for each actor. Goals are AND-decomposed and contribution links between goals are discovered. See for instance [23, 24] for details on how goal analysis can be carried out. Goal analysis ends when all the relevant goals of each actor have been analyzed and all the dependencies between actors are established. As for the first step, also here the information are collected and organized using a predefined template. For each actor, we mainly use a table of the form (`Goal, Sub-goal, InContrib, OutContrib`), where `InContrib` is the list of goals that contribute to the satisfaction of the goal and `OutContrib` is the list of goals that receive a contribution from the satisfaction of the goal.

Note that analyzing the dependencies is important since it allows new goals to be discovered for each actor, and in general it explains the reasons behind some of the goals thus leading to a comprehensive view and model of the organization. The analysis, of course, is limited to some areas of the organization, thus excluding some aspects that are not considered to be relevant for the design of the DW. It is responsibility of the analyst to decide what has to be included and what not; however, we emphasize that it is necessary to include in the model the external actors (e.g., clients of the bank) who do not belong
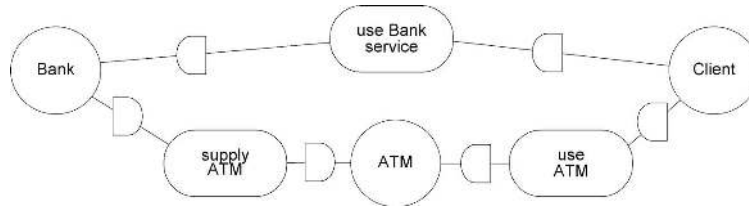
Figure 1: An actor diagram for the BI-BANK case study

to the organization but still play a crucial role in its activities.

Example 1 *Figure 1 shows a partial actor diagram for the BI-BANK case study. The* **Client** *depends on the* **Bank** *for achieving the goal* **use Bank service**, *and on the* **ATM** *for the goal* **use ATM***. Moreover, the* **Bank** *depends on the* **ATM** *actor for the goal* **supply ATM***. Figure 2 presents a part of the rationale diagram for the* **Bank** *actor focusing on the goal of managing transactions. The goal* **manage a/c transactions** *is decomposed into* **manage debit transactions** *and* **manage credit transactions***, and in turn* **manage debit transactions** *is decomposed into* **manage permanent payments** *and* **manage occasional payments***. New dependencies may be discovered at this point, for example the* **Client** *depends on the* **Bank** *to* **manage transaction***.* □

3.1.2 Fact Analysis

The objective of fact analysis is to identify all the relevant facts for the organization. The analyst navigates the rationale diagram of each actor and extends it by associating goals with facts that model the set of events to be recorded when goals are achieved. Here the information are collected using two different type of templates. The first one is a simple table that describes each single fact (`Fact, Description`), and the second a table in which each goal is associated with a number of facts (`Goal, Facts`). Fact analysis is carried out according to a top-down approach, in fact the analyst starts by identifying relevant facts from the top goals of each actor and then moves down towards the leaf goals. Usually, facts associated to leaf goals are not considered.

The structure of the rationale diagrams induces some relevant relationships between facts. In particular, when a goal is decomposed in subgoals, all the facts related to the
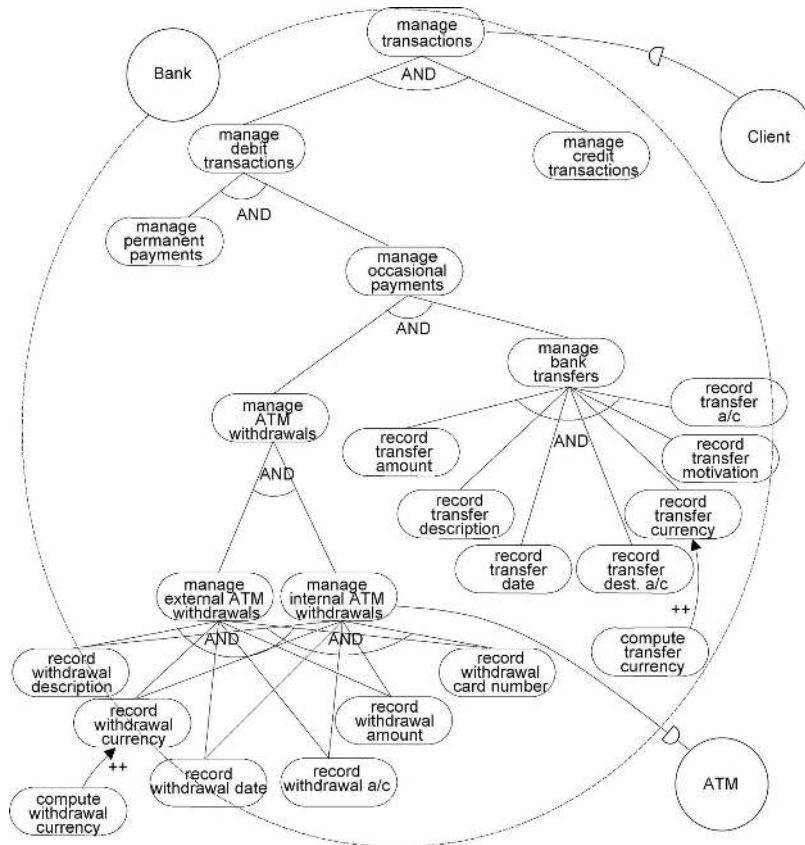
11

Figure 2: Rationale diagram for the Bank actor from the organizational perspective

subgoals become subfacts of the fact associated to the goal. As we will see in Section 4.1.3, these induced relationships are useful for defining the final conceptual schema for the DW..

3.1.3 Attribute Analysis

Attribute analysis is aimed at identifying all the attributes that are given a value when facts are recorded. Starting from the extended rationale diagrams produced in the previous phase, the analyst explores all the subgraphs to associate goals with the attributes they use. Note that, in this phase, the attributes are identified without specifying their possible role as dimensions or measures; from the organizational point of view, attributes are simply data associated to goals. To collect information for attribute analysis, the analyst uses a table of the form (Attribute, Goal, Fact).

Example 2 *Figure 3 shows an extended rationale diagram for the Bank actor, still focus-*
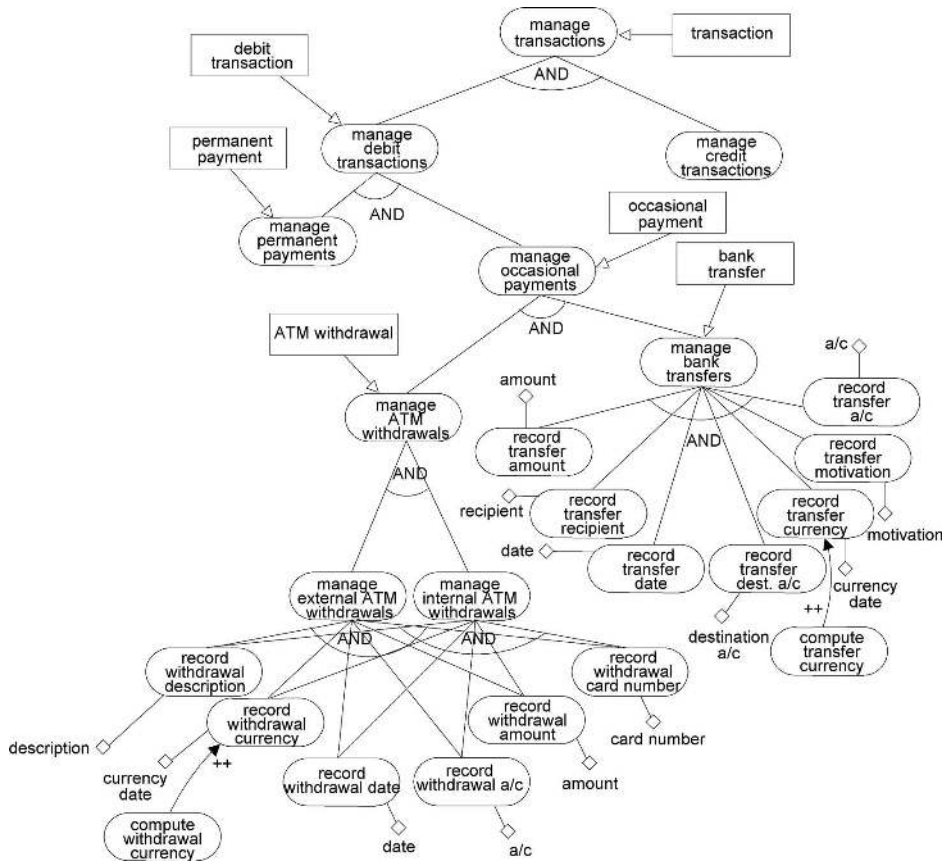
12

Figure 3: Extended rationale diagram for the Bank actor from the organizational perspective

ing on goal **manage transactions**. *First, the fact* **transaction** *is associated to the main goal* **manage transactions**, *the fact* **debit transaction** *to the goal* **manage debit transactions**, *and so on. Then, by analyzing the subgraph of the goal* **manage ATM withdrawals** *that fact* **ATM withdrawal** *is associated with, we introduce attributes* **currency, date, card number, amount,** *etc.*   □

3.2 Decisional Modeling

After organizational modeling, the methodology proposes a second type of analysis focused on the goals of decision makers, i.e., the actors that play the most relevant role in the decisional process. Here the analysis is substantially different from the organizational analysis since the main objective is not to model the organization as it is, but rather to model how the DW can support the decisional process of the organization. Basically, we

focus on the requirements of the DW from the perspective of the decision makers. The organizational model is extremely important in this phase since it actively supports the analyst during the identification of the facts to be associated with the decision makers' goals.

Firstly, all the decision makers are identified; then, for each of them, four steps are carried out: (i) *goal analysis*, that produces rationale diagrams; (ii) *fact analysis*, that extends them with facts; (iii) *dimension analysis*, that further extends them with dimensions; and (iv) *measure analysis*, that further extends them with measures.

3.2.1   Goal Analysis

As for organizational modeling, goal analysis starts by analyzing the actor diagram for the decision makers. Decision makers are identified and initial dependencies between them are established. The goals associated to each decision maker are then decomposed and analyzed in detail, to produce a set of rationale diagrams. Goals may be completely different from those analyzed during organizational modeling, indeed they are part of the decision process and might be not included in the operative process of the organization. The same templates used for organizational modeling are used here to collect and organize information about goal analysis.

Example 3 *Figure 4 shows a rationale diagram for decision maker* Financial Promoter, *focusing on the goal of analyzing transactions. The goal* analyze transactions *is OR-decomposed into* analyze debit transactions *and* analyze ATM withdrawals, *which in turn are further decomposed. So, for instance, the goal* analyze debit transactions *is OR-decomposed into* analyze total amount *and* analyze number of transactions. □

3.2.2   Fact Analysis

Like for organizational modeling, rationale diagrams are extended by identifying facts and associating them to the goals of decision makers. Facts are possible objects of analysis, and correspond to business events that dynamically happen within the enterprise. Facts are
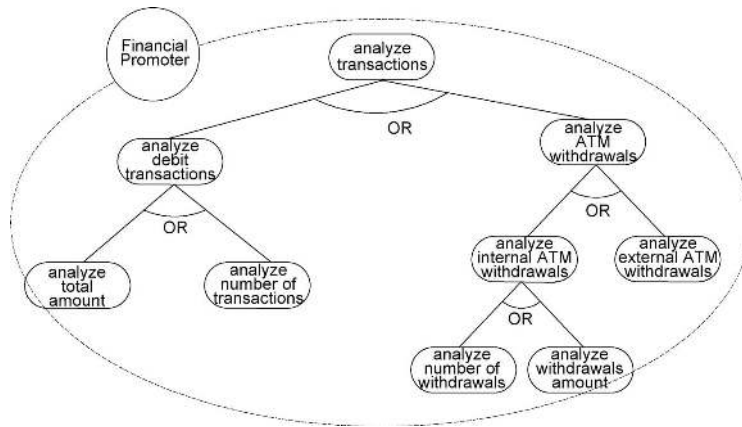
Figure 4: Rationale diagram for the Financial Promoter decision maker from the decisional perspective

normally imported from the extended rationale diagrams produced during organizational modeling. Indeed, very often the goals of decision makers are related to the information produced in the operational process, so the facts associated to the organization activities are fundamental for fulfilling the decision makers' goals. In some cases, the analyst can also introduce some new facts by directly analyzing the decision maker rationale diagrams.

3.2.3 Dimension Analysis

In this phase, each fact is related to the dimensions that decision makers consider necessary in order to satisfy their decisional goals. Dimensions are identified by analyzing the leaf goals of the rationale diagram of the decision makers and the relevant facts associated to the upper level goals. Here, analysis requires a strong interaction with the decision makers in order to capture the possible perspective of analysis. To support analysis we propose a template based on two different tables, the first capturing the relationship between goals, facts, and dimensions (`Goal, Fact, Dimensions`), the second describing dimensions (`Dimension, Description`).

3.2.4 Measure Analysis

Finally, the analyst associates a set of measures to each fact previously identified. Even here, analysis is carried out by analyzing the leaf goals and the facts associated to the
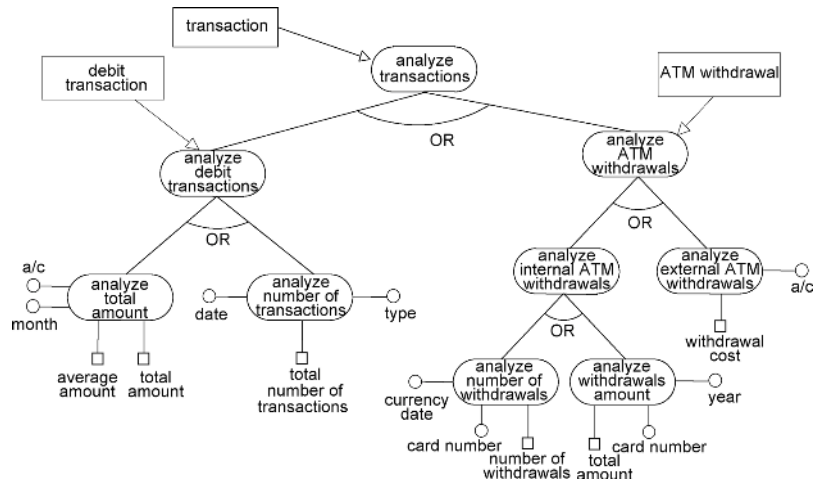
15

Figure 5: Extended rationale diagram for the Financial Promoter decision maker from the decisional perspective

upper-level goals, and requires a strong interaction with the decision makers.

Example 4 *In Figure 5 the analyst associates fact* transaction, *identified during organizational modeling (see Figure 3), to the goal* analyze transactions. *Then, dimensions are connected to the goals associated to the fact; for instance, dimensions* account number *and* month *are associated to goal* analyze total amount. *Finally, two measures are identified for goal* analyze total amount: total amount *and* average amount. □

4 From Requirement Analysis to Conceptual Design

The organizational model produced by requirement analysis represents the main data on which the enterprise operation is based, thus it comprises the most relevant attributes that are part of the source database. On the other hand, the decisional model describes the decision makers' needs, thus summarizing the role played, in glossary-based requirement analysis, by the glossaries of facts, dimensions, and measures and by the preliminary workload. In this section we explain how these diagram are used for conceptual design within, respectively, a mixed and a demand-driven framework.

4.1 Mixed Design Framework

The mixed framework joins the facilities of supply-driven approaches with the guarantees of demand-driven ones. In fact, the requirements derived during organizational and decisional modeling are matched with the schema of the source database to generate the conceptual schema for the DW. Three phases are involved: (i) *requirement mapping*, where facts, dimensions, and measures identified during decisional modeling are mapped onto entities in the source schema; (ii) *hierarchy construction*, where a basic conceptual schema is generated by navigating the source schema; and (iii) *refinement*, where the basic conceptual schema is edited to fully meet the user' expectations.

4.1.1 Requirement Mapping

During this phase the facts, dimensions, and measures included in the extended rationale diagrams produced by decisional modeling are mapped, where possible, onto the source schema. More precisely:

1. Assuming that sources are modeled by a relational schema, the facts of decisional modeling are mapped onto relations.

2. As to dimensions and measures, mapping is achieved by using the attributes represented during organizational modeling as a bridge. In fact, each such attribute is mapped to a physical attribute in the source schema on the one hand, and to a dimension or a measure in the decisional model on the other.

3. As to the attributes in the organizational model that were not mapped to dimensions or measures in the decisional model, they are mapped nonetheless on the source schema; as we will see in Subsection 4.1.2, this may be useful to provide the designer with an additional choice of dimensions and measures for the fact.

Example 5 *In our case study, as shown in Figure 6, fact* **ATM withdrawal** *is mapped to some* **WITHDRAWALS** *table in the source schema. Then, attribute* **card number** *associated to goal* **record withdrawal card number** *in the organizational model is mapped to dimension*
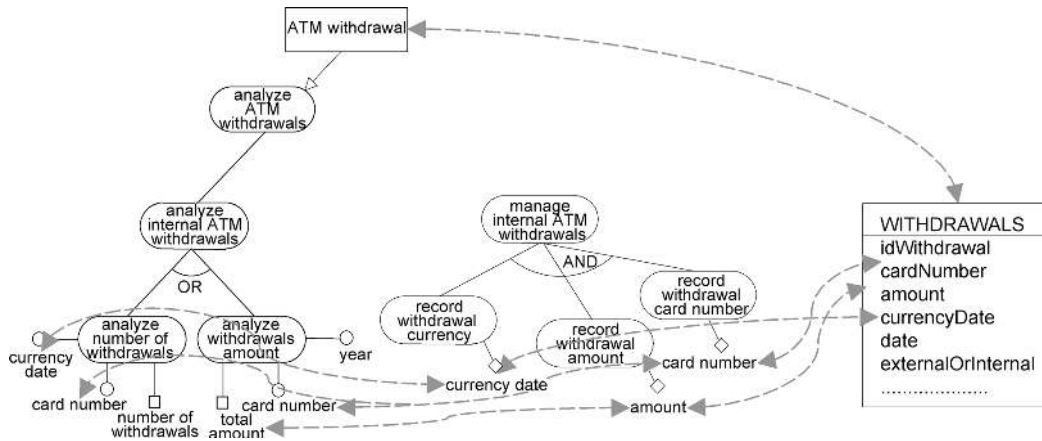
Figure 6: Mapping from the decisional model (left) to the source schema (right) passing through the organizational model (center)

card number associated to the analysis goals **analyze withdrawals amount** *and* **analyze number of withdrawals** *in the decisional model; the same attribute* **card number** *might for instance correspond, on the source schema, to an attribute* **cardNumber** *within the* **WITHDRAWALS** *table. Similarly, attribute* **amount** *of goal* **record withdrawal amount** *corresponds to measure* **total amount** *of the analysis goal* **analyze withdrawals amount** *and to an attribute* **amount** *on the* **WITHDRAWALS** *table. An example of an attribute in the organizational model that cannot be mapped to the decisional model but should nevertheless be mapped to the source schema is* **destination a/c**, *associated to goal* **record transfer dest. a/c**. □

Interestingly, if the names in the extended rationale diagrams are chosen by the analyst consistently with those in the source schema, this phase can be partially automated. In particular, if the source schema was actually obtained by normalizing and integrating different sources – which very often is the case, especially when complex cleaning and transformation procedures are necessary to improve data quality – its name space is largely under the designer's control. Otherwise a Thesaurus must be built, as suggested in [15], in order to establish the mappings between the organizational model and the source schemata.

4.1.2 Hierarchy Construction

This phase implements the supply-driven part of GRAnD. For each fact identified during decisional modeling and successfully mapped onto a relation F in the source schema, the

many-to-one associations expressed in the source schema by foreign keys are iteratively navigated, starting from F, to build the attribute hierarchies and create a basic conceptual schema, e.g. in the form of a *fact schema*. Fact schemata are the conceptual artifacts provided by the Dimensional Fact Model, proposed in [7] as a support for conceptual design of DWs. Note that any other conceptual model for multidimensional databases could be equivalently adopted.

This phase can be largely automated, as discussed in [7] where an algorithm is proposed that create attribute hierarchies from the source schema. The underlying idea is that, given a relation, a functional dependency – i.e., a many-to-one association – holds between its primary key and each of its attributes, including foreign keys. Since a foreign key maps on the primary key of another relation, that in turn functionally determines each of its attributes, a tree of attributes related by functional dependencies – in the terminology of multidimensional modeling, a hierarchy – can easily be built by recursively navigating the source schema. In the following, considering that functional dependencies are transitive, we will say that a *many-to-one path* exists from a relation R to any attribute a in the schema if the primary key of R (transitively) functionally determines a.

Remarkably, while in the supply-driven approach described in [7] navigation is "blind", meaning that all the attributes connected to the fact F by a many-to-one path are reached and included in hierarchies, here navigation is actively biased by the user requirements. In fact:

1. Every dimension d associated to a goal related to F and successfully mapped from the decisional model to the source (see Section 4.1.1) is included in the conceptual schema, and the full hierarchy rooted in d is generated by navigation. If the path from F to d is not many-to-one but many-to-many, d is modeled as a *multiple dimension* [25]; this means that multiple values of d can be related to a single instance of the fact F.

2. Every measure m associated to a goal related to F and successfully mapped from the decisional model to the source is included in the conceptual schema, provided that a many-to-one path exists from F to m, and no hierarchy is generated for it.

3. For each attribute a in the organizational model that could be mapped onto the source schema but not onto the decisional model, the designer has different choices:

3.1 Decide that a is not interesting for analysis and ignore it.

3.2 Decide that a may be interesting for analysis of some fact F, in which case it should be included in the conceptual schema for F. The designer also has to choose the primary role of a:

3.2.1 The role may be that of a dimension, in which case the full hierarchy rooted in a is generated by navigation and a is labeled as "supplied" in the conceptual schema. If the path from F to a is not many-to-one but many-to-many, a is modeled as a multiple dimension.

3.2.2 The role may be that of a hierarchy attribute. In this case, if a has not already been reached by navigation of the source schema starting from some dimension (see point 1.), the designer selects one of the attributes in the path from F to a as a new dimension d, and labels d as "supplied".

3.2.3 The role may be that of a measure, provided that a many-to-one path exists from F to a. Even in this case, a is labeled as "supplied" in the conceptual schema.

4. Among the dimensions and measures associated to a goal related to F on the decisional model, those for which no mapping on the source schema could be found are nevertheless included in the conceptual schema for F and labeled as "demanded".

5. All the attributes in the source schema that were not mapped and were not reached by navigation of many-to-one associations starting from a dimension are not included in the conceptual schema for F.

Remarkably, the basic conceptual schemata generated here may be considerably simpler and smaller than those generated in [7]. Besides, while in [7] the designer is asked for identifying facts, dimensions, and measures directly on the source schema, here such identification is driven by the diagrams developed during requirement analysis. We also
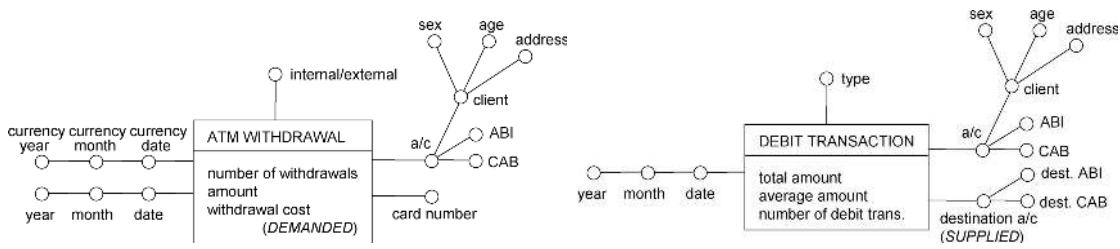
Figure 7: Preliminary fact schemata for facts ATM WITHDRAWALS and DEBIT TRANS-ACTION in a mixed framework

note that the names used for measures in decisional diagrams can give the designer precious suggestions regarding which aggregation operators to use: for instance, from Figure 5 we may presume that measure `amount` is to be aggregated through both the sum and the average operators.

Example 6 *The preliminary fact schemata obtained for facts ATM WITHDRAWAL and DEBIT TRANSACTION in the bank example are reported in Figure 7. Consistently with the Dimensional Fact Model, the fact is represented as a box containing the measures; dimensions are circles connected to the fact; hierarchies are represented as trees rooted in dimensions. For fact ATM WITHDRAWAL, dimensions currency date, card number, and a/c are derived from the decisional model, as well as measures number of withdrawals and amount. The hierarchy rooted in a/c has been built by navigating the source schema. Dimension internal/external has been added since, though the decisional model does not represent it explicitly, the two separate subgoals analyze internal ATM withdrawals and analyze external ATM withdrawals suggest that the user will ask for distinguishing between these two types of withdrawal. Finally, measure withdrawal cost is labeled as "demanded" since it appears as a measure associated to goal analyze external withdrawals but not as an available attribute on the organizational model. As to fact DEBIT TRANSACTION, we only remark that dimension destination a/c is labeled as "supplied" since it is present in the organizational model but has not been indicated by decision makers as a dimension in the decisional model.* □

21

4.1.3 Refinement

This phase is aimed at rearranging the conceptual schemata in order to better meet the user's needs. The basic operations that can be carried out to this purpose are:

1. *Operations on facts*

    1.1 *Fact merge.* Merging facts is beneficial when the user is often interested in analyzing them together since they can be seen as particular cases of a more general fact. Two or more facts $G_1 \ldots G_n$ can be merged into a single fact if they have a common supergoal F in the decisional model. The resulting fact takes the name of F and has the union of the measures, dimensions, and hierarchies of the G_i's. A dimension that exists only within a subset of the G_i's is marked as optional in F, meaning that it characterizes only a subset of the events modeled by fact F.

    1.2 *Fact split.* A fact F may be split into n facts $G_1 \ldots G_n$ when most of its dimensions and measures are related to specific subsets of events. Each resulting fact G_i includes only the dimensions and measures that better characterize it; each dimension is accompanied by the corresponding hierarchy.

2. *Operations on hierarchies*

    2.1 *Attribute prune.* An attribute a may be dropped from a hierarchy while preserving the subtree rooted in a. This is useful when neither a nor its descendant attributes are relevant for analysis.

    2.2 *Attribute graft.* An attribute a may be dropped from a hierarchy while preserving the subtree rooted in a; the children attributes of a become children of the parent attribute of a. This is useful when a is not interesting for analysis but its descendants are.

    2.3 *Association adding.* A many-to-one association between attributes a and b may be added to a hierarchy, which results in changing to a the parent of attribute b in the tree representing the hierarchy [7]. Adding an association may be necessary in

order to model some functional dependency that was not correctly captured in the source schema.

2.4 *Optionality.* Optional attributes, i.e., attributes that have a value only for a subset of instances of the hierarchy, should be detected and properly marked on the conceptual schema.

3. *Operations on measures*

3.1 *Measure merge.* Two or more measures that only differ for their aggregation operator may be unified, provided that the conceptual model adopted supports the representation of different aggregation operators for the same measure.

Note that, thanks to the labeling of dimensions carried out during hierarchy construction, the decision makers and the designer are enabled to distinguish, on conceptual schemata, what is *needed and available* (unlabeled dimensions/measures), what is *needed but unavailable* (dimensions/measures labeled as "demanded"), what is *available but not needed* (dimensions/measures labeled as "supplied"). While the second category may drive the designers in enriching the source database or in considering new data sources, the second may stimulate the decision makers to undertake new directions of analysis.

Example 7 *We assume that (1) users are not interested in the client granularity, so attribute* client *is grafted; (2) the client address is not interesting, so* address *is pruned; (3) measure* withdrawal cost *can be computed during ETL; (4) dimension* destination a/c *is considered by users to be relevant for analysis. The two facts* ATM WITHDRAWALS *and* DEBIT TRANSACTION *are merged into fact* TRANSACTION, *that is the fact associated, in the decisional model, to the common supergoal of subgoals* analyze debit transactions *and* analyze ATM withdrawals. *In merging the facts, measures* number of withdrawals *and* number of debit transactions *converge in measure* number of transactions. *Measures* total amount *and* average amount *are merged into measure* amount, *to be aggregated by both the sum and the average operators. Dimension* internal/external *is incorporated into dimension* type *(meaning that, among the values for* type, *there also will be "internal ATM*
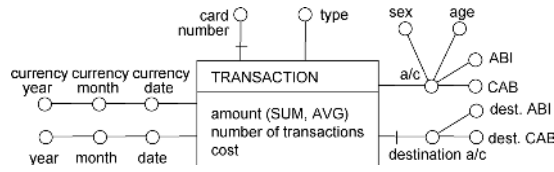
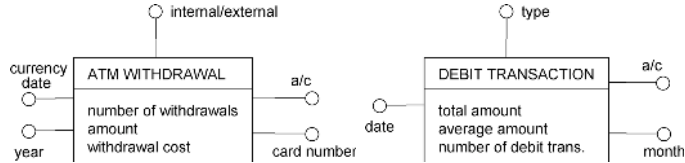Figure 8: Fact schema for fact TRANSACTION after refinement in a mixed framework

Figure 9: Preliminary fact schemata for facts ATM WITHDRAWAL and DEBIT TRANS-
ACTION in a demand-driven framework

*withdrawal" and "external ATM withdrawal"). Dimension **card number** is marked as op-
tional since a card number is defined only for some types of transactions; the same holds
for **destination a/c**. The final fact schema obtained is shown in Figure 8.* □

4.2 Demand-Driven Design Framework

Within a demand-driven framework, in the absence of *a priori* knowledge of the source
schema, the building of hierarchies cannot be automated; the main assurance of a satisfac-
tory result is the skill and experience of the designer, and her ability to fruitfully interact
with the domain experts to capture the existing dependencies between attributes.

The starting point is a set of preliminary conceptual schemata obtained by associating
each fact from the decisional model with the corresponding dimensions and measures.
In the bank case, from the rationale diagram in Figure 5 we immediately derive the
preliminary fact schemata in Figure 9.

The next step is related to the building of hierarchies: differently from the case of
the mixed framework, where hierarchies are determined by navigating the source schema,
here they must be defined manually by strictly interacting with business users. This
is undoubtedly the most difficult phase of the demand-driven approach; we suggest to
proceed for each fact F as follows:

1. Detect functional dependencies between dimensions of F and represent them in the

form of hierarchies.

2. If F is also represented on the organizational model, select all the attributes associated to the goals related to F on the organizational model. For each attribute a that is not already present in the conceptual schema and is considered to be interesting for analysis of F, understand its primary role:

  2.1 The role may be that of a dimension, in which case a is included in the conceptual schema and labeled as "supplied".

  2.2 The role may be that of a hierarchy attribute. In this case, the designer selects one of the existing dimensions, d, and properly connects a to the hierarchy rooted in d.

  2.3 The role may be that of a measure, in which case a is included in the conceptual schema as a measure and labeled as "supplied".

3. Repeat step 2. for all the other attributes in the organizational model.

4. Enrich hierarchies with other attributes possibly indicated by the user, labeled as "demanded".

The issues to be faced afterwords are the same described for the refinement phase in the mixed framework.

Example 8 *In fact* ATM WITHDRAWAL, *among all the attributes associated to goals* manage ATM external withdrawals *and* manage ATM internal withdrawals *on the organizational model,* date *is inserted as a supplied dimension. In fact* DEBIT TRANSACTION, *attributes* currency date *and* card number *are inserted as supplied dimensions. Then, during refinement, the functional dependency between* date *and* month *is detected, and the two facts are merged as seen in the mixed framework to produce the fact schema in Figure 10.*
□

This approach is, in our view, more difficult to pursue than the previous one. Nevertheless, it is the only alternative when a detailed analysis of data sources cannot be made
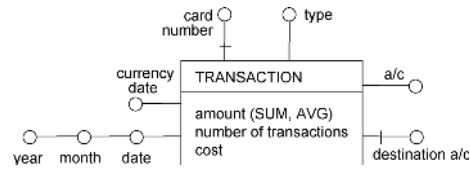
Figure 10: Fact schema for fact TRANSACTION after refinement in a demand-driven framework

(for instance when the DW is fed from an ERP system, whose logical schema is huge and hardly understandable), or when the sources come from legacy systems whose complexity discourages recognition and normalization.

5 DW-Tool

GRAnD is supported by a prototypical CASE tool named DW-Tool[1], which has been developed in Java and supports the analyst and the designer in the following activities:

- *Collection of requirements.* During the interviews with the stakeholders, the analyst organizes the information acquired by filling in the templates introduced in Section 3.

- *Dictionary management.* The content of the templates is used by DW-Tool to generate a general dictionary from which the analyst can select and see the description of each single entity in the models. Figure 11 shows an example where the description of a fact is provided.

- *Organizational modeling.* The content of the templates is also used to generate a basic organizational model, including actor and rationale diagrams, that the analyst can further rearrange to better meet the user's expectation (see Figure 12).

- *Decisional modeling.* After the basic rationale diagrams have been generated from the templates, the analyst can extend them with facts, dimensions, and measures, typically by importing entities from the diagrams produced during organizational modeling (Figure 13).

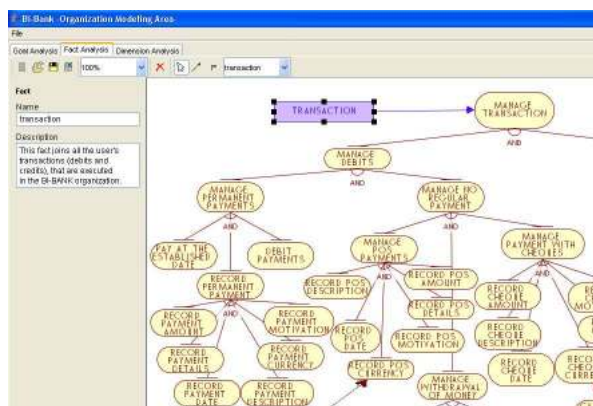[1]See the Tropos web site (http://www.troposproject.org) for more details.

Figure 11: Accessing the dictionary in DW-Tool

- *Conceptual design.* If the designer decides to follow a mixed approach, the source schema is acquired via JDBC from the operational DBMS. As to requirement mapping, DW-Tool proposes a basic choice of mappings, derived where possible by matching names in the diagrams. Hierarchy construction is fully automated, and the basic refinement operations are supported. On the other hand, if a demand-driven approach is undertaken, the attributes related to the fact in the organizational model are imported, so that the designer can decide which role to give them within the conceptual model of the DW.

6 Conclusion

In this paper we have proposed a goal-oriented methodology for requirement analysis in DWs, which can be used within both a demand-driven and a mixed supply/demand-driven design framework. We believe that the adoption of our methodology can help the designer to reduce the risk of project failure by ensuring that early requirements are properly taken into account and formalized – which ensures a "good" design – and, at the same time, that the resulting DW schemata are tightly rooted to the operational database – which makes the design of ETL simpler.

The methodology was applied to the BI-BANK case study, a project developed in collaboration with a company based in Trento. The experience with the company was
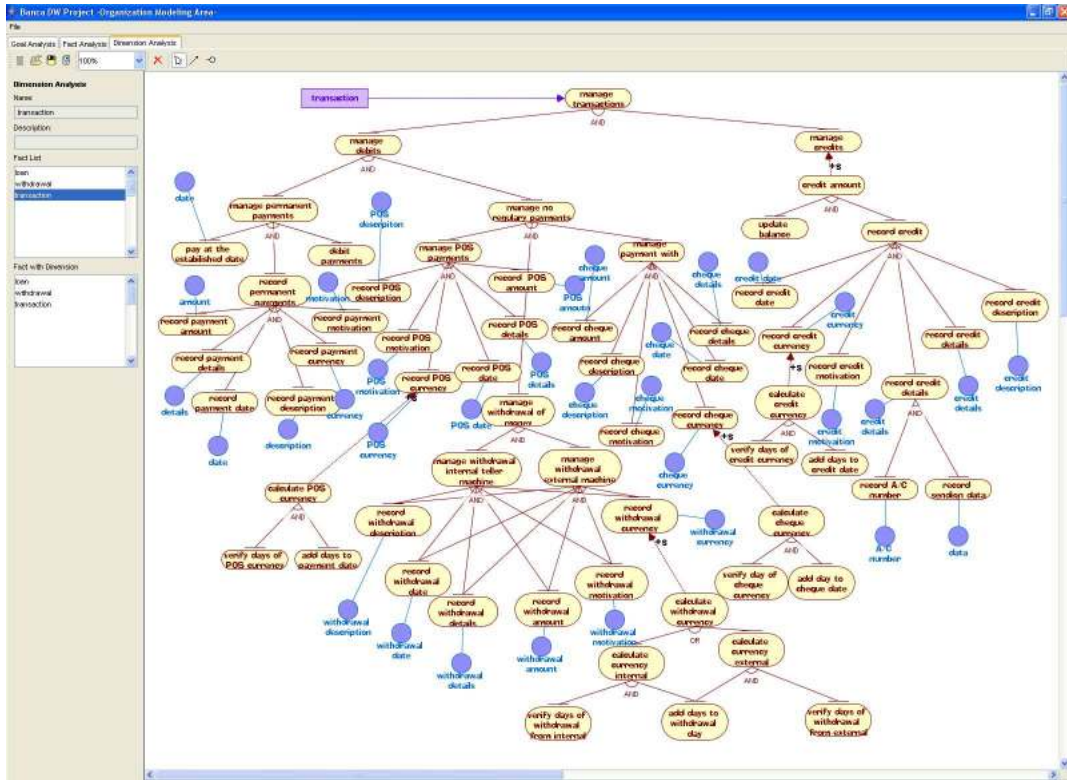
Figure 12: Organizational modeling in DW-Tool

extremely useful for refining and validating our approach. We received a positive feedback about the methodology and, in particular, about the importance of deriving the requirements directly from the analysis of the stakeholders and decision makers goals. The case study also supported us in investigating the scalability of our approach. With regard to this, we verified that associating an actor diagram with several rational diagrams, one for each actor, has a crucial role in dealing with complex application domains. In fact, detailed requirement analysis is carried out on rationale diagrams, whose complexity depends on how articulated the decisional tasks of a single actor are (which is not directly related to how large the application domain is). On the other hand, the actor diagram – on which the size of the application domain, in terms of number of stakeholders, directly impacts – is drawn at a high level of abstraction, thus its complexity never becomes overwhelming.
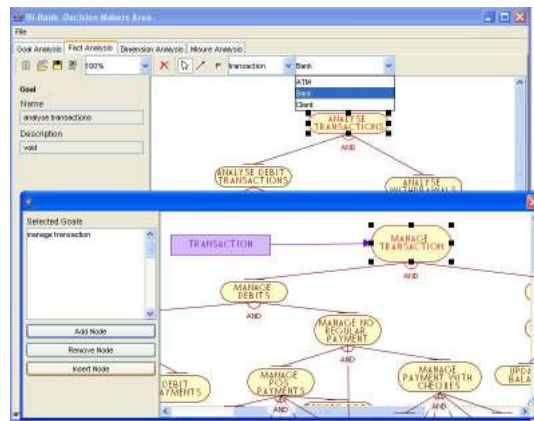
Figure 13: Importing facts for decisional modeling

References

[1] R. Winter, B. Strauch, A method for demand-driven information requirements analysis in data warehousing projects, in: Proceedings HICSS, Hawaii, 2003, pp. 1359–1365.

[2] M. Golfarelli, S. Rizzi, A methodological framework for data warehouse design, in: Proceedings DOLAP, 1998, pp. 3–9.

[3] R. Kimball, L. Reeves, M. Ross, W. Thornthwaite, The Data Warehouse Lifecycle Toolkit, John Wiley & Sons, 1998.

[4] S. Luján-Mora, J. Trujillo, A comprehensive method for data warehouse design, in: Proceedings DMDW, Berlin, Germany, 2003.

[5] F. R. S. Paim, J. B. Castro, DWARF: An approach for requirements definition and management of data warehouse systems, in: Proceedings International Conference on Requirements Engineering, Monterey Bay, CA, 2003.

[6] B. Hüsemann, J. Lechtenbörger, G. Vossen, Conceptual data warehouse design, in: Proceedings DMDW, Stockholm, Sweden, 2000, pp. 3–9.

[7] M. Golfarelli, D. Maio, S. Rizzi, The dimensional fact model: A conceptual model for data warehouses, International Journal of Cooperative Information Systems 7 (2-3) (1998) 215–247.

[8] D. Moody, M. Kortink, From enterprise models to dimensional models: A methodology for data warehouse and data mart design, in: Proceedings DMDW, Stockholm, Sweden, 2000.

[9] N. Prakash, A. Gosain, Requirements driven data warehouse development, in: CAiSE Short Paper Proceedings, 2003.

[10] R. Bruckner, B. List, J. Schiefer, Developing requirements for data warehouse systems with use cases, in: Proceedings Americas Conference on Information Systems, 2001, pp. 329–335.

[11] J. Lechtenbörger, Data warehouse schema design, Tech. Rep. 79, DISDBIS Akademische Verlagsgesellschaft Aka GmbH, (2001).

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

[13] W. H. Inmon, Building the Data Warehouse, QED Press/John Wiley, 1992.

[14] J.-N. Mazon, J. Trujillo, M. Serrano, M. Piattini, Designing data warehouses: From business requirement analysis to multidimensional modeling, in: Proceedings International Workshop on Requirements Engineering for Business Need and IT Alignment, Paris, France, 2005.

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

[16] A. Dardenne, A. van Lamsweerde, S. Fickas, Goal-directed requirements acquisition, Science of Computer Programming 20 (1–2) (1993) 3–50.

[17] E. Yu, Modelling strategic relationships for process reengineering, Ph.D. thesis, University of Toronto, Department of Computer Science (1995).

[18] B. List, R. Bruckner, K. Machaczek, J. Schiefer, A comparison of data warehouse development methodologies: Case study of the process warehouse, in: Proceedings DEXA, 2002, pp. 203–215.

[19] D. Bulos, Designing OLAP with ADAPT, Tech. rep., Atos Origin (1999).

[20] S. Luján-Mora, J. Trujillo, I. Y. Song, The Gold Model Case Tool: An environment for designing OLAP applications, in: Proceedings ICEIS, 2002, pp. 699–707.

[21] M. Golfarelli, S. Rizzi, E. Saltarelli, WAND: A CASE tool for workload-based design of a data mart, in: Proceedings SEBD, Portoferraio, Italy, 2002, pp. 422–426.

[22] J. Castro, M. Kolp, J. Mylopoulos, Towards requirements-driven information systems engineering: The Tropos project, Information Systems 27 (6) (2002) 365–389.

[23] P. Giorgini, E. Nicchiarelli, J. Mylopoulos, R.Sebastiani, Formal reasoning techniques for goal models, Journal of Data Semantics 1 (2003) 1–20.

[24] R. Sebastiani, P. Giorgini, J. Mylopoulos, Simple and minimum-cost satisfiability for goal models, in: Proc. CAiSE'04, 2004, pp. 20–33.

[25] S. Rizzi, Conceptual modeling solutions for the data warehouse, in: R. Wrembel, C. Koncilia (Eds.), Data Warehouses and OLAP: Concepts, Architectures and Solutions, Idea Group, 2006, To appear.