

DWARF: An Approach for Requirements Definition and Management of Data Warehouse Systems

Fábio Rilston S. Paim and Jaelson F. B. Castro
Universidade Federal de Pernambuco – Centro de Informática
Cx. Postal 7851, CEP 50732-970, Recife-PE, BRAZIL
Phone: (+55 81) 3271-8430, Fax: (+55 81) 3271-8438
{frsp, jbc}@cin.ufpe.br

Abstract

*In the novel domain of Data Warehouse Systems, software engineers are required to define a solution that integrates with a number of heterogeneous sources to extract, transform and aggregate data, as well as offers flexibility to run ad-hoc queries that retrieve analytic information. Moreover, these activities should be performed based on a concise dimensional schema. This intricate process with its particular multidimensionality claims for a requirements engineering approach to aid the precise definition of data warehouse applications. In this paper we adapt the traditional requirements engineering process and propose **DWARF**, a **Data Warehouse Requirements deFinition** technique. A case study demonstrates how the method has been successfully applied in the company wise development of a large-scale data warehouse system that stores hundreds of gigabytes of strategic data for the Brazilian Federal Revenue Service.*

1. Introduction

In recent years Data Warehouse has emerged as a powerful technology for integrating sparsely distributed operational data into a comprehensive analytical fashion to enable decision-making. The design of such systems is rather different from the design of the conventional operational systems that supply data to the warehouse. The former not only involves information requirements of decision makers, but also the structure and allocated requirements of the latter. Software engineers are required to deal with the complex process of extracting, transforming, and aggregating data while managing the deployment of a solution that timely integrates with a number of heterogeneous sources. Analytical results are required to be accurate and reliable. Flexibility at the front-end is paramount. To make things even worse, a comprehensive dimensional schema is mandatory.

Both operational and strategic visions have to be wrapped up in a multidimensional package to meet corporative analytical requirements that pervade pure decision-support functionality as well as quality constraints such as integrity, performance and other domain-specific requirements like *summarizability* [1]. This scenario clearly advocates the use of Requirements Engineering techniques to build a precise data warehouse definition.

To pursue this goal, we adapt the traditional requirements engineering process [26] to propose a methodological approach¹ for requirements definition and management of data warehouse systems. Our interactive, phase-oriented framework, coupled with a set of artifact templates, procedures and techniques, guides developers throughout the data warehouse requirements engineering cycle, which has been customized and extended to embrace decision-support specificities.

The following sections describe our approach. First, we discuss general concepts and system requirements pertaining data warehouse applications in Section 2. Section 3 details our method. Section 4 describes the contributions of our approach to the specification of a large data warehouse project conducted by SERPRO, a Brazilian Government software company, to its main client, the Brazilian Federal Revenue Service. Section 5 presents our conclusions and future works.

2. Data Warehousing: A Novel Philosophy

In this section we discuss the challenge of creating a requirements definition for data warehouse systems. Section 2.1 briefly introduces the data warehousing process and main concepts behind this new application philosophy. Section 2.2 presents a list of requirements

¹ The approach discussed in this paper extends our early work described in [3].

that should be addressed to correctly define data warehouse systems.

2.1. The Process of Data Warehousing

The term *Data Warehouse* was first coined to describe “a collection of consistent, subject-oriented, integrated, time-variant, non-volatile data in support of management’s decisions” [4]. More than just a collection of data, *data warehousing* is a defined process pertaining three phases: (a) **extracting** data from distributed operational sources (mostly legacy systems); (b) **transforming and aggregating** data consistently into the warehouse; and (c) **accessing** integrated data in an efficient and flexible fashion. The main contribution of a data warehouse is its capability of turning data into strategic information, accessible to decision-makers in the highest levels of an organization. Such capability is supported by OLAP (*OnLine Analytical Processing*) technology, which provides final users with configurable views of data from different angles and on different aggregation levels.

To achieve fast and flexible OLAP queries, data are arranged in a multidimensional form, where information is classified according to **facts** and **dimensions**. *Facts* are numeric or factual data that represents a specific business activity that we wish to analyze. *Dimensions* are single perspectives on the data that determines the *granularity* (data detail level) to be adopted for fact representation. Fact units and their values are referred to as *measures*. In addition, each dimension is described by a set of descriptive *attributes*, which qualify the data content. The data cube metaphor [5] is frequently used to clarify this multidimensional structure, each cell representing an intersection point between n dimensions, which holds a measure on which analysis is to be performed (Figure 1). Data cubes are usually implemented on relational

databases in a well-known multidimensional schema named “star” [13], in which a fact-table is connected by foreign keys to dimensional-tables that surround it.

For instance, the Brazilian Federal Revenue Service, an organization devoted to tax administration, considers **IRS (Internal Revenue Service) Collection** as a *fact* under the subject “Internal Revenue Taxes”. The related metric **quantity of IRS statements**, obtained from the *Tax Collecting System*, can be analyzed through the perspective of IRS statements registered for a given taxpayer class in each city within the fiscal year. The **taxpayer class**, the **federal unit** and the **time** (date) of a statement identify *dimensions*. These dimensions can be described in terms of *attributes* such as **category**, **economic group** and **type** (civil servant or enterprise), for the taxpayer class dimension; **city**, **state** and **region** for the federal unit dimension; and finally **day**, **month**, **quarter** and **year**, for the time dimension. Furthermore, attributes in a dimension can be arranged as a *hierarchical chain*, allowing measures to be classified and aggregated along hierarchical paths. Recalling our latter example, *category*, *city* and *day* represent the lowest *granularity* level of each dimensional hierarchy.

The development of an enterprise data warehouse is a cumbersome task. Developers commonly make use of the so-called “divide to conquer” strategy to identify and deploy meaningful subsets of data, in form of small data warehouses, containing information related to a certain business subject. These well-defined blocks of strategic information are referred to as *Data Marts*, and represent a starting point in an incremental cycle that aims to deliver the enterprise-wide data warehouse by integrating its independent blocks, one at a time.

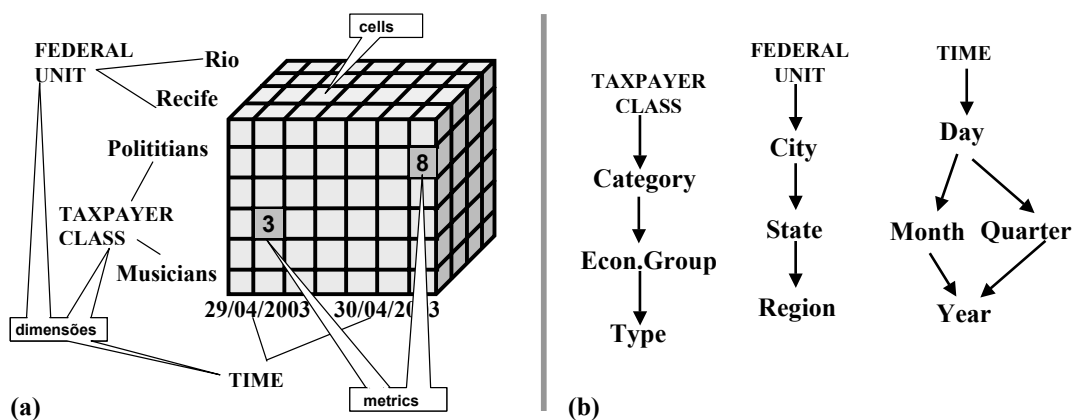


Figure 1. (a) Data cube and multidimensional elements; (b) Dimensional hierarchies (adapted from [6]).

2.2. Data Warehouse Requirements

Data Warehousing responds for a significant portion of information technology budgets in most organizations nowadays [7]. Successful projects have confirmed the high level of user satisfaction and return on investment. Despite the acclaimed potential, many projects fail to deliver the expected decision-support facilities [8]. There seems to be a consensus in the community that behind these failures lies a poor requirements definition phase [9,10,11,12]. Indeed, most data warehouse projects skip this phase to concentrate on design issues like query performance and database modeling.

The lack of a higher-level requirements vision results in design decisions that do not take into account certain aspects essential to project success such as user needs, multidimensional restrictions and quality constraints. Additionally, to better intertwine design and problem specification, the following set of concepts needs to be clarified while performing requirements engineering of data warehouses:

1. **Represent facts and their properties.** Facts are central to data warehouse. Analyzing user requirements implies identification of facts by perceiving the metrics behind user demands. Another requirement of concern is to determine the extent to which a metric is *additive*, i.e., it can be operated (*summarized, counted, drawn an average*) along dimensional hierarchies (e.g. *it makes sense to count the annual number of taxpayers, but to sum such amounts does not seem to yield a useful value*);

2. **Distinguish and connect dimensions to facts.** Dimensions offer the key to understanding fact measures by allowing the user to view data through different viewpoints. Sometimes the analysis of a single user statement gives rise to a number of candidate dimensions. For instance, the sentence “*Analyze the total revenues of individual tax types per year*” clearly specifies dimensions *time* and *tax*, as well as the *tax revenues* fact;

3. **Summarizability assurance.** An important issue in data warehouse systems is to guarantee the correctness of aggregation results for whichever combination of facts and dimensions, also known as *Summarizability* [1]. Drawbacks can be avoided by clearly expressing the constraint requirements regarding aggregation of data, as well as conforming Data Mart facts and dimensions to the enterprise data warehouse model. For example, the dimension “*time*” and the fact unit “*tax collection growth rate*” cannot be combined in the “*day*” attribute;

4. **Integration with data sources.** In a data warehouse environment, data is collected from several different

sources, inside and outside the enterprise environment. This activity involves not only importing data from all sort of bases, but also uncovering informal business requirements. Then, besides specifying procedures and requirements for data collection, one should determine how informal data will be acquired from spreadsheets, office databases or even entered by the own user.

5. **Fast track of user requirements changes.** A critical concern in data warehouse applications is evolution. Even sometimes so-called “*insignificant changes*” can impact the overall validity of derived data, weakening the decision-support potential of the application (e.g. *an increment of one digit in a metric size, though keeping the same content, can affect the loading process of this and other derived values*).

6. **High-quality documentation.** Developing a data warehouse always involves developers using pre-existing operational documentation to define data extraction and integration procedures. Apart from rare exceptions, such legacy documentation lacks the necessary quality, turning the extraction of technical requirements into a daunting activity. Thus, a high-level documentation designed to provide a general interface for requirements exploitation is extremely required.

Due to the aforementioned characteristics, it becomes necessary to define a new requirements engineering technique, tailored to the data warehouse domain.

3. The DWARF Technique

The **Data Warehouse Requirements deFinition (DWARF)** technique is structured in a set of phases (Figure 2). Each phase follows the abstraction levels of the application in depth, as the project requirements are gathered to form a requirements baseline. Surrounding this cycle stands a backbone phase named *Requirements Management Control*, aimed to perform permanent quality assessment of requirements evolution.

Coupled with the above process, document templates proposes a pre-defined structure for data warehouse requirements documentation, whereas work as recording instruments of facts and cutting-edge points for the system development. In the sequel we detail each phase of the DWARF technique. Section 3.1 states the need for a thorough requirements management planning prior to defining data warehouses and the issues that should be attended in this sense. Section 3.2 defines means to support a precise specification of this class of systems. Sections 3.3 and 3.4 present procedures to, respectively, validate and manage data warehouse requirements.

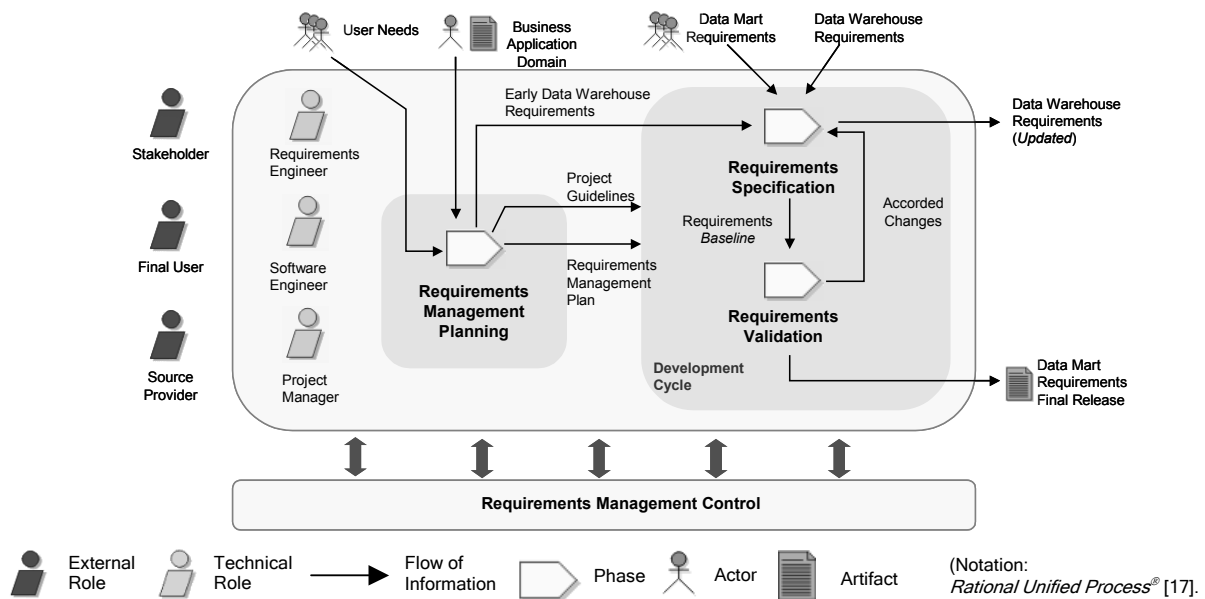


Figure 2. Phase-Oriented Framework of the DWARF Technique.

3.1. Requirements Management Planning

In the early stages of a requirements definition, rules for an effective requirements engineering process in data warehouse systems must be defined. Such general guidelines will guide the proper application of the DWARF technique as well as avoid that the absence of a common standard cause inconsistencies to acquisition, documentation and management activities. Guidelines can be defined in terms of business rules, procedures and processes commonly agreed to clarify, among others, the following aspects:

1. **Multidimensional Requirements Focus.** What is the granularity scope in each Data Mart (e.g. *queries to the industrialized products vision of our tax data warehouse should drill down to "manufactured goods"*)? What constraints (e.g. *legal terms, OLAP constraints*) restrict multidimensional analysis of data? In which ways would users like to have data summarized along dimensions (e.g. *multiple goods classification paths*)? The answers to these and many other important questions will rule the dimensional modeling and therefore must be depicted in the general project objectives.

2. **Source Integration Premises.** One must define clear rules for data exchanging between systems, centered in a standard integration layout. Periodicity, data loading priorities and responsibilities, among other issues, will constitute the core of integration guidelines (e.g. *only tax*

payment data whose total amount matches the extracted amount informed by the Tax Payment System must be loaded into the warehouse).

3. **Project Objectives.** The general flexibility of the dimensional schema and OLAP tools should not mask hidden limits of the data warehouse project. Clear objectives and project restrictions must be agreed on among all participants before the work gets started. For instance, some aggregated information (e.g. *customs tax flow*) related to clients' short-term decisions might not integrate the first warehouse version in case a source provider (e.g. *Customs System*) is unable to supply the data.

These aspects are furthermore documented on preliminary versions of the *Data Warehouse Vision*, *Multidimensional Requirements Specification* and *Requirements Management Plan* (see Section 3.2.3).

3.2. Requirements Specification

In data warehouse development, the requirements process underpins a cyclic approach of acquisition, representation and evaluation of requirements to gradually yield a system specification. Thus, an iterative process seems to be more appropriate to support such working flow. The initially elicited (raw) Data Mart requirements traverse a sequence of iterations, along which requirements are analyzed, negotiated within process participants, registered and conformed to a broader data warehouse specification.

As more increasingly refined information is fed back from a previous iteration, the following iteration tends to be faster and more easy-solving, its product being closer to an agreed requirements baseline. Modeling data warehouse systems requires an extra concern with reuse of earlier agreed requirements. The only way to reach a consistent integration among Data Marts is to specify common requirements (specially dimensional and factual ones) in a way that they mean the same across all Data Marts.

The following sections describe the (sub)processes that compound the *Requirements Specification* phase.

3.2.1. Requirements Elicitation. This phase aims to implement a process of requirements discovery by (a) communicating with stakeholders to identify decision-support needs; (b) describing system interaction with counterparts; (c) simulating OLAP behavior and (d) investigating architectural trade-off. Some techniques have been used (even adapted) to pursue this goal:

- **Interviews.** This technique is specially indicated to soothe users' natural inability to describe in concrete terms their strategic needs. We adopt two interviewing approaches [13,14] that propose straight questions tailored to data warehouse issues like granularity and multidimensionality;
- **Workshops.** In the context of data warehouses, requirements workshops (i) encourage consensus over application's multidimensional scope; (ii) promote fast agreement between involved parties about the course of actions for data mart delivering; (iii) solve political issues; (iv) reach cohesion within the development team; and (v) capture major functionalities and operational constraints.
- **Prototyping.** Used within the process to simulate OLAP behavior, prototypes show stakeholders how system facilities will aid in decision-making and offer a valuable opportunity to consolidate their ideas about how requirements will be allocated to the multidimensional solution.
- **Scenarios.** Use Cases can be used to specify scenarios in both textual and graphical (e.g. UML [15]) ways to clarify functional requirements. Figure 3 illustrates a standard use case model in UML for the main data warehouse functionalities, namely extracting, transforming and accessing data. "Querying" and "Loading" processes can be designed as single use cases using parameterized entries. Variant points in data transformation and extraction give rise to generalized use case trees, where child use cases describe integration steps particular to a source provider, whereas a central

use case holds commonalities of working on all sources. Elicitation efforts will focus on instantiate this standard model to specify business rules, procedures and querying features that make one project unlike the other. Additionally, use cases enable the reuse of common behavior shared among different Data Mart scenarios.

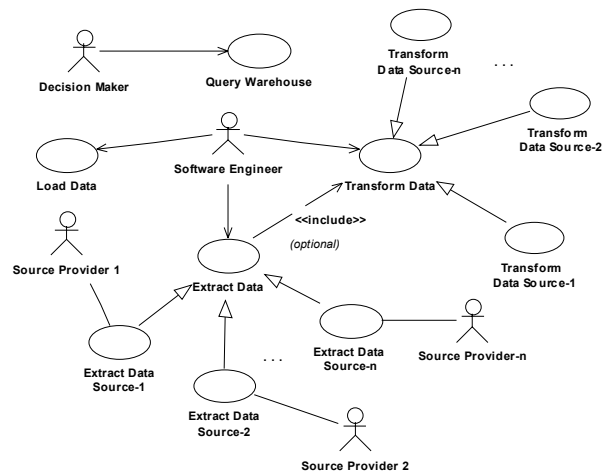


Figure 3. Use case model template in UML for data warehouse systems.

- **Non-Functional Requirements.** Few methods have proved to be as powerful as the NFR Framework [16] to deal with non-functional requirements. The framework allows for the definition and reuse of design catalogues which organize knowledge and methods about particular non-functional requirements (NFR) as well as their tradeoffs and interdependences. One possible usage of such catalogues is to investigate architectural alternatives that best fit a data warehouse solution. To achieve this goal, we extended in an early work [2] the NFR Framework to define a set of NFR types and operationalization catalogues regarding data warehouse applications, which we referred to as *Data Warehouse-Extended NFR Framework* or DW-ENF. The new framework applies to selecting indexing strategies, disk space optimized methods, loading schema and other critical parts of a data warehouse design. The DW-ENF non-functional requirements list also works as a checklist to guide the specification of system quality requirements.

3.2.2. Requirements Analysis & Negotiation. The requirements discovered during a prior elicitation phase, along with the existent requirements model are input to a detailed analysis that seeks to ensure that specification follows quality standards, multidimensional constraints,

OLAP tool restrictions and integration premises. The project team must assure that system requirements adhere to the global multidimensional schema. In particular, the analysis highlights points of adjustment that should be addressed to avoid aggregation creep in later development stages (e.g. *if metrics fail to add up through some hierarchical points, consider splitting the dimensional structure in two*).

To support this phase, we have defined a checklist for data warehouse requirements. Table 1 shows an excerpt of the proposed checklist.

Table 1. Requirements Checklist for Data Warehouse (excerpt).

Item	Description
Automatic aggregation	Do all dimensional levels lead to a complete summarizability approach, in terms of the multidimensional model elaborated?
Facts and dimensions representation	Are all stakeholders' analytical needs represented in terms of a multidimensional schema?
Facts and dimensions connection	Is the entire set of dimensional levels properly associated in all levels to the basic set of facts being analyzed?
Integration completeness	Are all integration requirements and procedures defined as to correctly incorporate external information from source providers?
Documentation quality	Do all defined documents serve as tools to accomplish all user needs under established quality standards?
Requirements conformity	Can we truly "drill" across fact tables by navigating through conformed dimensions without incurring in data loss or inconsistency?

3.2.3. Requirements Documentation. This phase is at the core of our approach. The purpose here is to provide a complete documentation of elicited requirements. We have designed a set of templates to register all data warehouse functional, non-functional and domain-specific requirements. The templates are *metadocuments* that describe their own purpose and appropriate usage. Our documentation model is based upon similar model proposed in the *Rational Unified Process* [18] and its structure reflects the requirements specification approach for *product families* [19], which suits precisely for the description of a data warehouse and its data marts. In the sequel, we present a brief description of each artifact:

- **Requirements Management Plan.** Documents the management aspects essential to project regulation, as discussed in Section 3.1.

- **Project Glossary.** Organizes terminology related to problem domains envisioned in the overall project to enhance understanding among all involved parties.
- **Data Warehouse Vision.** Describes enterprise data warehouse requirements, including descriptions of motivation and problem issues; general objectives; project scope; stakeholders' profile and other issues concerning the whole data warehouse.
- **Data Mart Vision.** Collects high-level data mart user needs, as well as features and actors to support such needs.
- **Use Cases Specification.** Detail the procedures required to implement all functionalities designed for each data mart, describing the possible sequences that might occur until the final result is achieved.
- **Multidimensional Requirements Specification.** Describes the multidimensional portrait, including restrictions; factual and dimensional information; and considerations about granularity, additivity and conformance of multidimensional requirements.
- **Non-Functional Requirements Specification.** Complements the use case specifications, describing non-functional requirements not covered by the use case model, as well as design constraints and other restrictive factors (ex. *legal issues, tool limits*).
- **Business Rules Specification.** Register all business rules that regulate data warehouse and data mart functionalities.
- **Revision Report.** A simple report to hold the actions agreed after a requirements validation session (see Section 3.3).

3.2.4. Requirements Conformance. This phase is unique to data warehouse specification and tackles an essential issue to this domain that is *conformance*. According to [13], if one hopes to build a data warehouse that is robust and resilient in face of continuously evolving requirements, one must adhere to a data warehouse definition on which common facts and dimensions are conformed among all data marts. A dimension is said to be "conformed" when it means the same to every attached fact table. Similarly, a fact conforms to the global model if the same terminology is used across data marts to represent its content.

In our approach, we extend this concept to an utmost level of abstraction on which all common system requirements are conformed, i.e., represented identically within the data mart requirements model. This principle not only includes multidimensional aspects, but also every functionality, characteristic or constraint to the system development that holds the same reasoning all over the project (e.g. *the taxpayer dimension should be*

unique to all data marts; similarly, all common features related to data extraction should be defined as identical as possible to all data marts). After a documentation phase, all artifacts are analyzed to carve off those requirements that represent enterprise data warehouse specificities. Practitioners look for requirements overlapping and similarities and adjust the specification to conform these requirements, even changing its (general/unique) nature if need be.

Among the benefits that conformed requirements bring to the data warehouse specification, we emphasize the following: (a) avoid redundancy and ambiguity between requirements that oversees the entire data warehouse; (b) allow common dimensional aspects such as dimension tables to be applied to multiple facts in the same database space; (c) in conjunction with a use case approach, promote reusability of agreed knowledge in the project, thus enhancing quality; (d) improve user interface and data content consistency; (e) enable OLAP operations involving multiple fact tables; (f) make data warehouse evolution a much easier task; (g) facilitates adherence to design and organizational standards.

3.3. Requirements Validation

After several iterations, some misunderstandings and/or misconceptions regarding the data warehouse analytical features might still remain. The validation phase corrects these potential pitfalls. *Review sessions*, together with *prototyping* prove to be an effective strategy to detect and remove defects in the target application, before they become part of the delivered data mart package. During the review meeting, the data mart final requirements release is presented to all involved parties and described in terms of its functional, non-functional and multidimensional aspects. In addition, an OLAP interface prototype helps users to recognize architectural aspects derived from defined requirements and validate the solution.

When problems are located, the validation team immediately attaches a list of actions in response to each problem, and agrees with the actions to be enrolled. The development process returns to the specification phase where actions are applied to conform the requirements baseline to the right specification. We have added external reviewers to this process as a means of bringing an unbiased perspective about the data warehouse solution. This practice have improved defect detection efforts and suggested new OLAP analysis over data.

3.4. Requirements Management Control

Requirements cannot be managed effectively without traceability [20]. In data warehouse environments,

traceability and change management must be carried out in both requirements and architectural spheres. The former is concerned with managing changes to agreed requirements and its impact to other requirements inside the same or in external documents. The later extends this investigation to the database architecture, in order to clear up what impact the underlying change will have in the multidimensional schema. Supporting tools exist for both investigation activities [21,22,23,24]. Such tools becomes mandatory for data warehouse systems as they generate large amounts of requirements and database attributes.

Traceability Matrices [20] can be used to show requirements dependencies. Different matrices must be defined to enable cross-reference analysis between requirements (ex. *functionalities versus facts*; *facts versus dimensional attributes*).

On the other hand, changes in requirements affect straightforwardly the database model in both Data Mart and Data Warehouse visions. Most relational databases offer tools [21,25] to search for affected attributes in the database, and discover how many (and at what extent) database elements are influenced by the change. These features reduce the effort of evaluating and performing the necessary maintenance to database table fields, thus preserving the safety of the solution.

4. Case Study: Data Warehouse for the Brazilian Federal Revenue Service

The Brazilian Federal Revenue Service (FRS) is a central organ subordinated to the Finance Ministry and responsible for the collection, administration and auditing of a plethora of federal taxes. To process the huge amount of data (billions of registers of all sorts) that originates from its fiscal activities, the FRS holds a partnership with SERPRO, a Finance Ministry subsidiary software company, for the development of automated solutions in support of tax analysis.

SERPRO is a large company with development units broadly spread throughout 10 capital cities of Brazil. The company employs 2.470 software engineers and has a history of successful and awarded solutions [27] built along 38 years as a partner of the Brazilian FRS. These solutions offer full-automated support for a multitude of aspects comprising fiscal actions at individual business segments of the Federal Revenue Service. The growing numbers of data and surpassing complexity of Brazil's tax system, however, have fostered the need for an integrated vision of fiscal actions in the highest administration levels of the Brazilian FRS.

The S.A.F.E. (in english, *Federal Revenues Strategic Information System*) data warehouse was then conceived to collect and store nearly three hundreds of gigabytes of

strategic information about federal revenues fiscal activities. Information is stored in a subject-driven perspective to perform complex OLAP queries. *Subjects* depict tax administration core business segments, and are modeled as eight integrated Data Mart solutions. Attached to each data mart fact table, a collection of dimension tables compounds star-schemas that are interconnected through their common dimensions in a complex data warehouse structure, well known in the literature as constellation [28]. Through a Web interface, end users operate the data warehouse from whichever point across the Federal Revenues Service intranet, 24 hours a day, 7 days of the week.

SAFE has been developed by SERPRO under the premises of the DWARF technique since the beginning of 2001. An engineering team was composed of two requirements engineers and a group of eight developers (including the project leader) and five database architects from two important development sites: Recife and Rio de Janeiro. The chosen strategy rested on capturing critical requirements, i.e., those tightly connected to the system bottlenecks: (a) integration process definition, (b) multidimensionality mapping, and (c) user needs change control. Then, our approach was to intensively use the processes and artifacts specified by DWARF to fight back bottleneck points, with special care to requirements specification and management. Two data mart visions have been deployed so far and two other ones are scheduled to be delivered by June of 2003, totalizing 158 gigabytes of information. On average, two iterations have been followed for each Data Mart development. The following sections summarize the contributions brought by DWARF to the project.

4.1. Requirements Management Planning

This phase wrapped clients into the novel experience of discussing the data warehouse inner aspects, objectives and construction guidelines together with the technical team. Such an experience turned out to be a critical step towards avoiding misconceptions about the decision-support potential of the application such as expensive queries to be avoided and data loading frequency. In particular, rules for data extraction and load were devised to differ the actions and roles involved with either directly harvesting and feeding data or outsourcing these activities. The rules were annotated onto the *Requirements Management Plan*, which also provided a common standard for system requirements attributes, from which classification and (mainly) prioritization data mart requirements became possible. At the end, not only clients' confidence in the project rose high, but also the so-called gap between clients and developers was considerably reduced. The phase also revealed the need

for a glossary of terms, considering the wide range of business domains tackled by the data warehouse.

4.2. Requirements Specification

Workshops and interviews enabled a high-level information exchange among stakeholders, which facilitated the definition of the project scope and data mart goals. Then, detailed use case definition stages focused on real user needs. By instantiating and evolving the proposed use case model template (see Figure 3) at each iteration, engineers were able to design functionalities to implement data mart needs for extracting, transforming and accessing data. This documentation also created a standard language between developers and their parties in other teams involved with extraction activities. Complementarily, prototyped versions added a broader understanding about data warehouse capabilities and helped both stakeholders and developers to refine the requirements model.

On the architectural side, the DW-ENF framework enabled engineers to investigate an optimal design solution for the data warehouse. Critical requirements like response time, updating frequency and cross-fact navigation were combined with the catalogue of methods to define indexing and data loading schemas, as well as an optimized relational schema (*constellation*).

All gathered requirements were first cataloged onto 51 instances of document templates, and later loaded into a RequisitePro® [23] requirements repository, from which they could be easier managed by the project leader. Comparing to earlier experiences in developing data warehouse solutions, this new approach provided means for correctly representing multidimensional elements, conformed requirements, summarizability constraints and other important issues not treated by previous approaches.

An artifact was highlighted in the process: the *Multidimensional Requirements Specification*, providing sections to neatly register the multidimensional scope, factual and dimensional requirements, and related issues like granularity levels, metric additivity and dimensional cardinality, essential to balance query performance and model a consistent dimensional schema. A distinct product of applying our method was the generation of a conformed structure of dimensions (*organizational level, periodicity, tax-payer, tax status, among others*) and facts (*financial turnover, fiscal action and tax collection, plus related metrics*), which worked as a backbone for subsequent Data Mart specifications. Similarly, quality assurance procedures were generalized and applied to the entire data warehouse development.

4.3. Requirements Validation

We note here that clients added a new perspective to the validation phase, promoting the scheduled review sessions to open *Workshops* that included 20 participants on average (including external observers), focused on analyzing the documentation and elaborated prototypes. Observers contributed with unbiased questions that stood out misconceived (or even forgotten) conceptual and architectural aspects, which reinforced the importance of such elements in this phase.

For example, based on their experience in the auditing field, observers could notice during one of the review events that the “organizational level” dimension, though structurally correct, did not reflect all valid associations in the revenue service hierarchy (*for instance, there are inspecting units with the status of a superintendence, therefore must be treated as such*). As a response to this evidence, an action report describing the required adjust in the dimensional domain was produced and tracked to completion by the project leader.

4.4. Requirements Management Control

To date, a total of 4.875 requirements of all sorts has been incorporated to the project requirements repository. As previously argued in this work, such amount would be unmanageable without automated support. RequisitePro[®] facilities has come to rescue in this task, allowing the project leader to evaluate the impacts of requirements changes to the project, in order to better negotiate the change with the client. Particularly, traceability matrices proved very helpful to demonstrate the impact of a certain change in a fact or dimension to the use cases modeling the data warehousing process.

On the architectural side, impacts were analyzed using Oracle Designer/2000[®] [29]. For example, during the definition of the IRS data mart, the client decided to insert another economic group level in the related dimension, which affected not only the dimensional schema, but also the taxpayer data loading process. In conjunction with the requirements change analysis executed, search procedures were applied to the Oracle database *application systems* [30] to identify the consequent impacts to the multidimensional schema as well as to the use cases that detail loading activities. The results were thus joined up to enrich discussions with the client.

5. Conclusions and Future Works

The adoption of the DWARF technique operated a deep change on the client-developer paradigm. Clients are now engaged in the construction process (unlike past

experiences), which guaranteed the development of an agreed solution. Moreover, the use of requirements management facilities increased clients’ perception of requirements changing consequences, and the impact of continuously changing needs to the data warehouse schedule.

On the other hand, developers are now aware of requirements reusability during the development of a data warehouse, looking for preserving conformance to the enterprise broader model. From a technical view, developers realized the potential of the approach to solve some problem issues in data warehouse development such as summarizability and cardinality analysis of queries; definition of facts and dimensions from a “user need” point of view; documentation of the data warehousing process in a high-level form and management of constant changing multidimensional requirements.

Our experience has indicated the achievement of project schedule; small incidence of errors; definition of faster and more agile auditing programs; elimination of informal investigation programs with considerable savings on related costs; and a sound client’s satisfaction with the quality obtained with the delivered product.

In spite of the benefits, the experience in the project SAFE also revealed some points of improvement in the technique that will be object of further researches:

- The large amount of requirements caused traceability matrices to become hard to manage or visualize. Higher-leveled requirements attributes were then introduced to more coarsely relate requirements in each matrix. In addition, filtering conditions were applied to matrices in order to restrict the list of retrieved requirements to those strictly affected by changes. These measures, however, did not keep developers from executing partial analysis to completely estimate the impact of requirements changes. Future researches will investigate the contributions of more elaborated traceability analysis such as the ones referred to in [20].
- The method does not consider maintenance projects. An improvement to the present approach will seek to customize its components (notably the set of templates) as to contemplate maintenance activities.
- The approach does not provide adequate solution to the problem of specifying multiple hierarchies for dimensional structures. Such problem requires some graphic support. Future works will evaluate the contributions of using graphic tools specially design to pursue this goal, like the one proposed in [6].
- Metadata have been considered an important aspect towards developing more robust and agile data warehouse systems [13,31]. The Project SAFE

experience revealed the need to enhance the specification phase to deal with metadata requirements, what will be object of future versions.

References

- [1] H-J Lenz and A. Shoshani, "Summarizability in OLAP and Statistical Databases", 9th International Conference on Statistical and Scientific Databases, 1997, pp. 132-143.
- [2] F. R. S. Paim and J. B. Castro, "Enhancing Data Warehouse Design with the NFR Framework", 5th Workshop on Requirements Engineering (WER2002), Valencia, Spain, November 11-12, 2002, pp. 40-57.
- [3] F. R. S. Paim, A. E. Carvalho and J. B. Castro, "Towards a Methodology for Requirements Analysis of Data Warehouse Systems", XVI Simpósio Brasileiro de Engenharia de Software (SBES'2002), Gramado, Rio Grande do Sul, Brasil, October 16-18, 2002, pp. 146-161, best paper out of 104 submissions.
- [4] W. H. Inmon, *Building the Data Warehouse*, John Wiley & Sons, 2nd edition, 1996.
- [5] M. Boehnlein and A. Ulbrich-vom Ende, "Deriving Initial Data Warehouse Structures from the Conceptual Data Models of the Underlying Operational Information Systems", 2nd International Workshop on Data Warehousing and OLAP (DOLAP), Kansas City, MO, USA, 1999.
- [6] J. Trujillo, M. Palomar, J. Gomez and I-Y Song, "Designing Data Warehouses with OO Conceptual Models", *IEEE Computer*, special issue on Data Warehouses, 34(12): 66-75, December 2001.
- [7] A. Sen and V. Jacob, "Industrial Strength Data Warehousing", *Communications of the ACM*, September 1998.
- [8] S. Kelly, *Data Warehousing in Action*, New York, John Wiley & Sons, 1997.
- [9] L. Cabibbo and R. Torlone, "A Logical Approach to Multidimensional Databases", *Lecture Notes in Computer Science* a. 1377, 6th International Conference on Extending Database Technology (EDBT98), Valencia, Spain, 1998, pp. 183-197.
- [10] W. Lehner, J. Albrecht and H. Wedekind, "Normal Forms for Multidimensional Database". 8th International Conference on Statistical and Scientific Database Management (SSDBM), IEEE Computer Society, 1998.
- [11] N. Tryfona, F. Busborg and J. Christiansen, "starER: A Conceptual Model for Data Warehouse Design". 2nd Workshop on Data Warehousing and OLAP (DOLAP), Kansas City, Missouri, USA, 1999.
- [12] P. Vassiliadis, "Gulliver in the Land of Data Warehousing: Practical Experiences and Observations of a Researcher", 2nd International Workshop on Design and Management of Data Warehouses (DMDW), Stockholm, Sweden, 2000, pp. 12.1-12.16.
- [13] R. Kimball, L. Reeves, M. Ross and W. Thorntwaite, *The Data Warehouse Lifecycle Toolkit*, New York, John Wiley & Sons, 1998.
- [14] W. Giovinazzo, *Object-Oriented Data Warehouse Design*, Prentice Hall, 1st edition, February 2000.
- [15] G. Booch, J. Rumbaugh and I. Jacobson, *The Unified Modeling Language User Guide*, Addison-Wesley, 1999.
- [16] L. Chung, B. Nixon, E. Yu and J. Mylopoulos, *Non-Functional Requirements in Software Engineering*, Kluwer Publishing, 2000.
- [17] Rational Unified Process: Artifacts Notation. <http://www.rational.com/products/rup/index.jsp>
- [18] P. Kruchten, *The Rational Unified Process*, Addison-Wesley, 1999.
- [19] D. Leffingwell and D. Widrig, *Managing Software Requirements: A Unified Approach*, The Object Technology Series, Addison-Wesley, 2000.
- [20] M. Toranzo, "A Proposal for Improving Requirements Traceability" (in portuguese), PhD Thesis, Federal University of Pernambuco, December 2002.
- [21] Oracle9i Developer Suite[®]: see <http://www.oracle.com/ip/develop/ids/index.html?designer.html>
- [22] Popkin's System Architect[®]: see <http://www.popkin.com/>
- [23] Rational RequisitePro[®]: see <http://www.rational.com/products/reqpro/index.jsp>
- [24] Telelogic DOORS/ERS[®]: see <http://www.telelogic.com/products/doorsers/index.cfm>
- [25] Microsoft SQL Server: see <http://www.microsoft.com/sql/default.asp>
- [26] G. Kotonya and I. Sommerville, *Requirements Engineering: Processes and Techniques*, Addison-Wiley, 1997.
- [27] Brazilian Federal Revenue Service Technologic Awards: see <http://www.receita.fazenda.gov.br/principal/Ingles/Premios/default.htm>
- [28] R. Kimball, *The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses*, John Wiley & Sons, 1996.
- [29] Oracle Designer/2000[®]: see http://www.oracle.com/rdb/rdb_journal/jn19711/index.html?oakey.html
- [30] "Documenting the Enterprise Using Oracle9i Designer", Oracle White Paper, July. Available from http://otn.oracle.com/products/designer/pdf/otn_9ides_doc_ent_wp.pdf.
- [31] M. Staudt, A. Vaduva and T. Vetterli, "The Role of Metadata for Data Warehousing", Technical Report 99.06, Department of Information Technology, University of Zurich, September 1999.