Data Warehouse Quality: A Review of the DWQ Project

Matthias Jarke
Lehrstuhl fur Informatik V
RWTH Aachen
Ahornstrasse 55
52056, Aachen, Germany
jarke@informatik.rwth-aachen.de

Yannis Vassiliou
Computer Science Division
Dept. of Electr. and Comp. Engineering
National Technical University of Athens
Zografou 15773, Athens, Greece
yv@cs.ntua.gr

Abstract. *DWQ is a cooperative project in the ESPRIT program of the European Communities. It aims at establishing foundations of data warehouse quality through linking semantic models of data warehouse architecture to explicit models of data quality. This paper provides an overview of the project goals and offers an architectural framework in which the individual research contributions are embedded.*

1. Introduction

A Data Warehouse (DW) is a collection of technologies aimed at enabling the knowledge worker (executive, manager, analyst) to make better and faster decisions. It is expected to present the right information in the right place at the right time with the right cost in order to support the right decision. The practice proved that the traditional online Transaction Processing (OLTP) systems are not appropriate for decision support and the high speed networks can not by themselves solve the information accessibility problem.

Data warehousing has become an important strategy to integrate heterogeneous information sources in organizations, and to enable online Analytic Processing. A report from the META group that was published during the Data Warehousing Conference (Orlando, Fl) in February 1996, presents very strong figures for this new area:

- Data warehousing will be a $13,000 million industry within 2 years ($8,000m hardware, $5,000m on services and systems integration), while 1995 represents $2,000m levels of expenditure. The numbers are extrapolated from buying plans listed by customers, not from optimistic vendor-supplied data.
- The average expenditure for a DW project is $3m. This is set to accelerate. 59% of the survey's respondents expect to support data warehouses in excess of 50Gb by the middle of the year 1996 and 85% are claiming they will be supporting over 50 users in the same timeframe.

Analogous numbers are given by Gartner Group and other evaluators and magazines.

The DW movement is a consequence of the observation by W. Inmon and E.F. Codd in the early 1990's that operational-level on-line transaction processing (OLTP) and decision support applications (OLAP) cannot coexist efficiently in the same database environment, for two main reasons, both having to do with trade-offs in data quality:

- **data characteristics**: OLTP databases maintain *current data* in great detail locally to their immediate operational usage, whereas OLAP deals with lightly aggregated and often globally reconciled *historical data*, covering much more than just the current ones; mixing both causes complex compromises between different degrees of detail, and varying needs for history information.

- **transaction characteristics**: OLTP emphasizes efficiency of *short update* transactions each covering a small part of the database, whereas OLAP requires *long queries* each surveying a large part of the database; mixing the two causes concurrency control problems.

A DW therefore caches selected data of interest to a customer group, so that access becomes faster, cheaper and more effective. As the long-term buffer between OLTP and OLAP (fig.1), DW's face two essential questions: how to reconcile the stream of incoming data from multiple heterogeneous legacy sources? and how to customize the derived data storage to specific OLAP applications? The trade-offs driving the design decisions concerning these two issues change continuously with business needs, therefore design support and change management are of greatest importance if we do not want to run DW projects into dead ends. This is a recognised problem in industry which is not solvable without improved formal foundations.

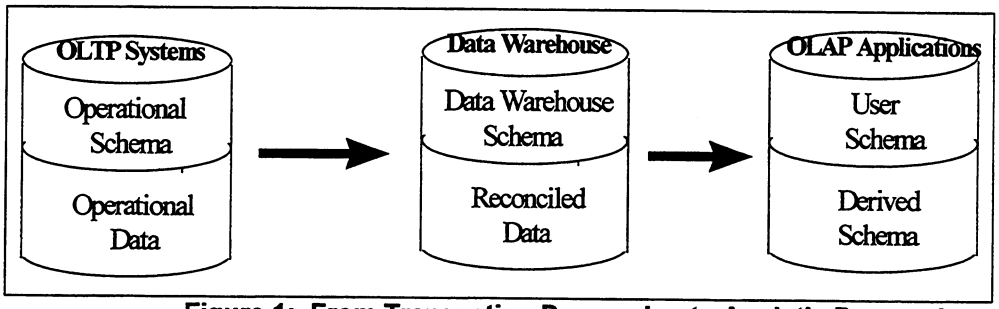

Figure 1: From Transaction Processing to Analytic Processing

Vendors agree that data warehouses cannot be off-the-shelf products but must be designed and optimized with great attention to the customer situation. Traditional database design techniques do not apply since they cannot deal with DW-specific issues such as data source selection, temporal and aggregated data, and controlled redundancy management. Since the wide variety of product and vendor strategies prevents a low-level solution to these design problems at acceptable costs, only an enrichment of metadata services linking heterogeneous implementations constitutes a promising solution. This requires research in the foundations of data warehouse quality.

The goal of the DWQ project is to develop semantic foundations that will allow the designers of data warehouses to link their choice of deeper models, richer data structures and rigorous implementation techniques to quality-of-service factors in a systematic manner, thus improving the design, the operation, and most importantly the evolution of data warehouse applications. These semantic foundations will be made accessible by embedding them in methodological advice and prototype tools. Their usefulness will be validated in the context of vendor methodologies/tool suites and a number of sample applications.

The rest of this paper is organized as follows. After a brief overview of the project goals in section 2, section 3 presents an architectural framework for data warehousing that makes an explicit distinction between conceptual, logical, and physical design issues and thus clarifies a number of issues related to data warehouse quality. In section 4, the specific research DWQ research questions arising from this framework are outlined, and some pointers to related work and preliminary DWQ results are provided. Section 5 outlines conclusions and future work.

2. The DWQ Project

In this section, we briefly review the structure of the DWQ project, the basic components of data warehouse solutions, and the linkage to formal quality models.

2.1 DWQ Project Structure

DWQ (acronym for Foundations of *D*ata *W*arehouse Quality) is a three-year cooperative research project (1996-1999) funded by the European Communities under the Reactive Long Term Research Branch of their ESPRIT IV R&D Program. Partners include the National Technical

University of Athens, Greece (Y. Vassiliou, T. Sellis), Aachen University of Technology, Germany (M. Jarke, F. Baader), the German AI Research Center DFKI (W. Nutt), INRIA Rocquencourt, France (E. Simon), the University of Rome, Greece (M. Lenzerini), and the IRST Research Center, Italy (E. Franconi).

The number and complexity of DW projects is indicative of the difficulty of designing good data warehouses; their expected duration highlights the need for documented quality goals and change management. It is an important goal of DWQ to validate the results against such requirements out of a sample of important real-world applications. The project closely cooperates with Software AG, a leading European vendor of DW products and solutions. It also cooperates with major ongoing DW efforts by user organizations such as the Italian Telecom, the French Ministry of Environment, the French Institute for Environment (IFEN), Air France, and the Office of Statistics of the City of Cologne in Germany.

2.2 DWQ Objectives

DWQ's research objectives address three critical domains where quality factors are of central importance to data warehousing:

- enrich the semantics of meta databases with formal models of information quality to enable adaptive and quantitative design optimization of data warehouses;
- enrich the semantics of information resource models to enable more incremental change propagation and conflict resolution;
- enrich the semantics of data warehouse schema models to enable designers and query optimizers to take explicit advantage of the temporal, spatial and aggregate nature of DW data.

During the project the whole spectrum of the DW modeling, design and development will be investigated:

- The DWQ framework and system architectures
- DW metamodeling and designing methods and languages,
- Optimization.

After developing an initial reference model, the results will be delivered in two stages to enable effective project control and coherence.

The first group of results will develop enriched formal meta models for describing the static architecture of a DW and demonstrate how these enriched foundations are used in DW operation. The corresponding methodologies and tools include architecture modeling facilities including features for addressing DW-specific issues such as resolution of multiple sources and management of partially aggregated multi-dimensional data, as well as semantics-based methods for query optimization and incremental update propagation.

The second group of results focuses on enhancing these enriched models with tools that support the evolution and optimization of DW applications under changing quality goals. The corresponding methodologies and methodologies and tools include: evolution operators which document the link between design decisions and quality factors, reasoning methods which analyze and optimize view definitions with multi-dimensional aggregated data, and allow efficient quality control in bulk data reconciliation from new sources; and quantitative techniques which optimize data source selection, integration strategies, and redundant view materialization with respect to given quality criteria, esp. performance criteria.

2.3 Data Warehouse Components

The DWQ project will provide a neutral architectural reference model covering the design, the setting-up, the operation, the maintenance, and the evolution of data warehouses. Figure 2 illustrates the basic components and their relationships as seen in current practice. The terms used in this figure can be briefly explained as follows:

- sources: any data store whose content is subject to be materialized in a data warehouse,
- wrappers to load the source data into the warehouse,
- destination databases: data warehouses and data marts,
- meta database: repository for information about the other components, e.g. the schema of the source data,

- agents for administration (data warehouse design, scheduler for initiating updates, etc.) and
- clients to display the data, for example statistics packages.

The detailed elaboration of these components and relationships can be filled in according to many different user requirements and modeling techniques. The DWQ results in this area concern a general architecture modeling framework (cf. section 3).

The simplest architecture consists only of source databases, a central data warehouse and several clients. As data warehouse applications are becoming more complex, warehouses are being built using multi-tier architectures to increase performance, i.e. there is not only one central data warehouse but also data marts which allow to place the data closer to the end-user.

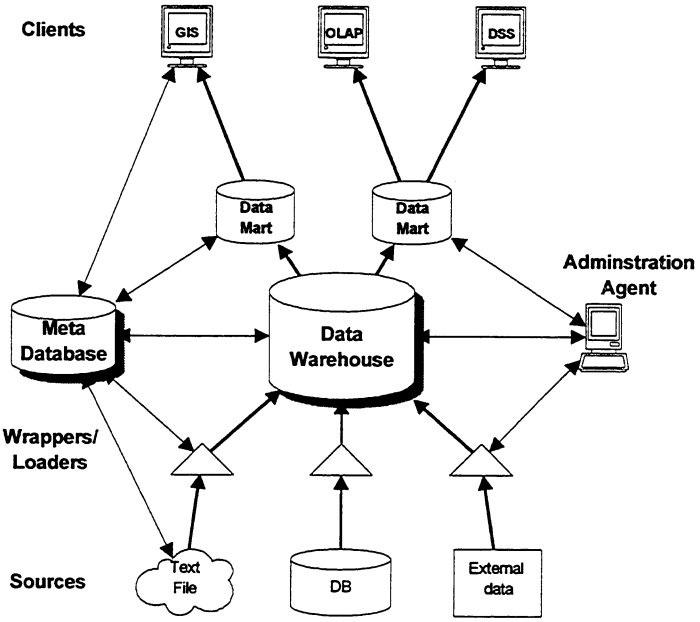

Figure 2: Sample structure of a data warehouse

2.4 The Linkage to Data Quality

DWQ provides assistance to DW designers by linking the main components of a DW reference architecture to a formal model of data quality. The model has been specialized for use with data warehousing from a quality goal hierarchy proposed in (Wang et al. 1995). Main differences to the initial model lie in the greater emphasis on historical as well as aggregated data.

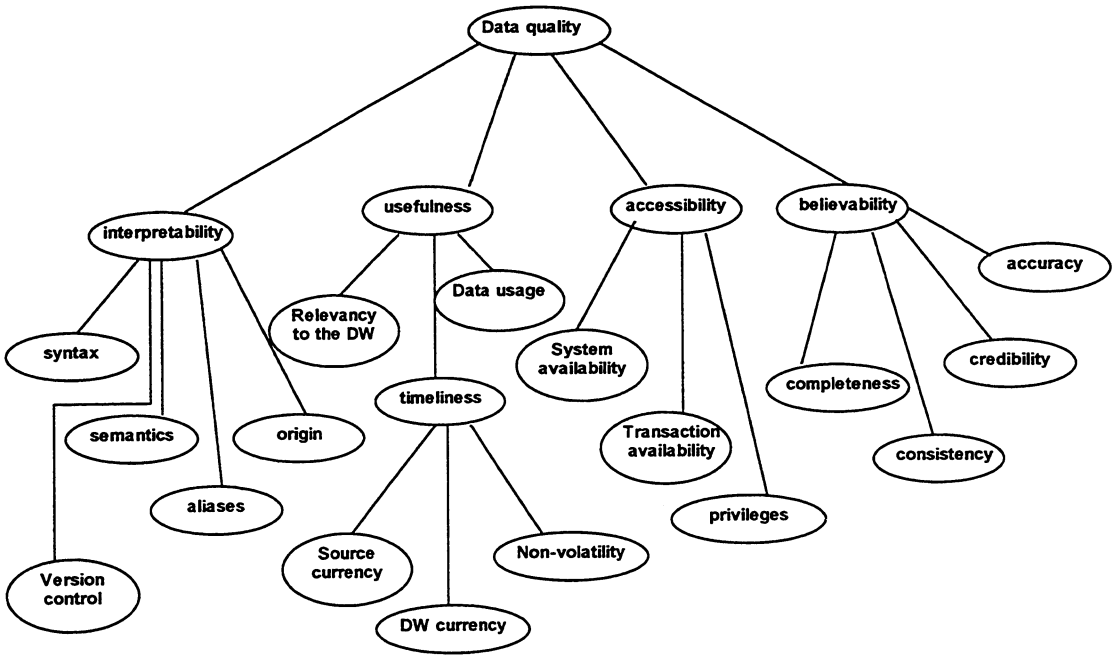

Figure 3 : Quality Factors in Data Warehousing

A data quality policy is the overall intention and direction of an organization with respect to issues concerning the quality of data products. Data quality management is the management function that determines and implements the data quality policy. A data quality system encompasses the organizational structure, responsibilities, procedures, processes and resources for implementing data quality management. Data quality control is a set of operational techniques and activities which are used to attain the quality required for a data product. Data quality assurance includes all the planed and systematic actions necessary to provide adequate confidence that a data product will satisfy a given set of quality requirements.

Consequently, the problem which arises is the modeling and the measurement of the quality of the data warehouse. As the data warehouse is a system composed of several subsystems and processes between them, there must be a mapping between the data warehouse components and the quality model we will introduce. Furthermore, we must seek the goal of developing measures for the quality indicators and a methodology for designing quality for specific data warehouses.

A closer examination of the quality factor hierarchy reveals several relationships between quality parameters and design/operational aspects of DW's. In figure 4 we can see some of these relationships. The goal of the DWQ project is to investigate these relationships in a systematic manner, and to propose formal solutions that will help in DW design and development.

3. DWQ Architectural Framework

When looking at the literature on data warehousing in more detail, we found that figures 1 and 2 must be interpreted in at least three different ways. We call these the conceptual (or business) perspective, the logical (or data modeling) perspective, and the physical (or distributed information flow) perspective. Any given data warehouse component or substructure can be analyzed from all three perspectives. Moreover, in the design, operation, and especially evolution of data warehouses it is crucial that these three perspectives are maintained consistent with each other. Finally, quality factors are often associated with specific perspectives, or with specific relationships between perspectives.

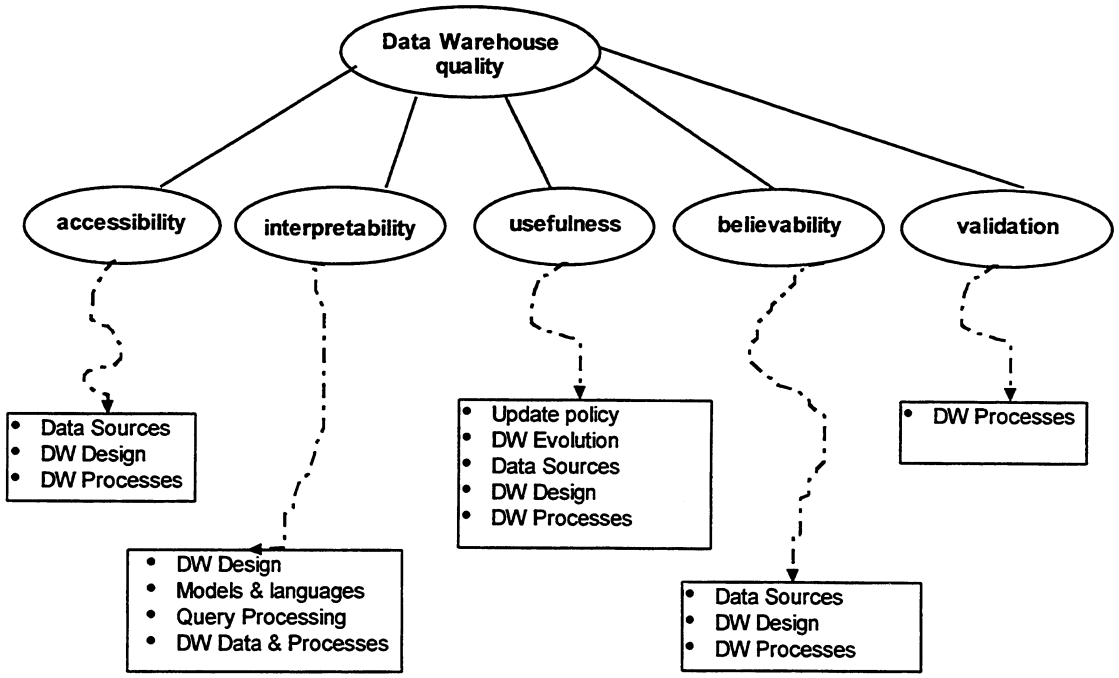

Figure 4: Linking quality factors to data warehouse tasks

In figure 5, we find correspondingly that there are 9 data store component types, linked by at least 12 kinds of relationships maintained by what we call DW agents in our meta model.

We discuss each perspective in turn. Note that the discussion is purely at the schema level. Each DW component is an instance of all three perspectives.

2.1 The Conceptual View

For strategic management, a data warehouse can serve the purpose to impose an overall business perspective – a conceptual enterprise model – on the information resources and analysis tasks of an organization.

If this enterprise model is understood as a class structure, its instance is the organization the DW maintains information about, and thus not part of the DW itself. Instead, the DW architecture is a network of direct and derived views (i.e., recorded observations) on this reality. The schema of information sources is defined as a collection of pre-materialized conceptual views on the enterprise model, not vice versa! This important observation has already been made in the Information Manifold project at AT&T [Levy et al. 1995].

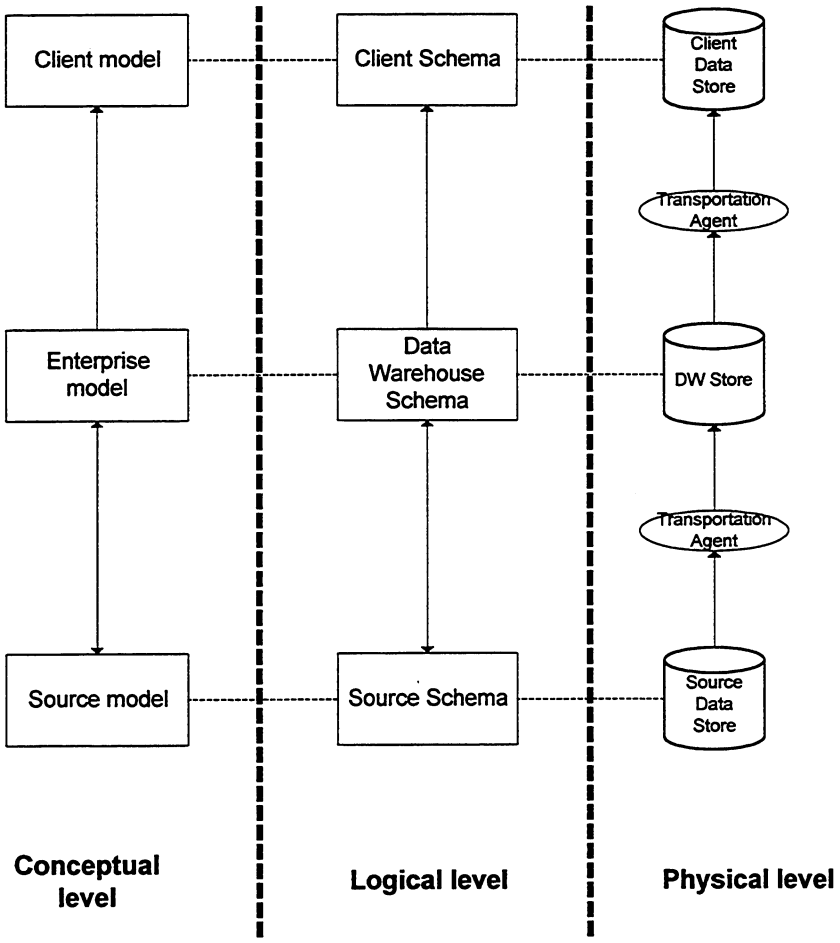

Figure 2: The DWQ Framework: Three Aspects of Data Warehouse Architecture

On the client side, the interests of user groups (such as analysts) can also be described as conceptual views on the enterprise models. However, these client views cannot be satisfied by direct observation of reality; instead, they must rely on existing materialized views, i.e. the information sources. As a consequence, we have the problem of mapping conceptual client views on conceptual source views – a theorem proving problem, where the proofs correspond to the queries to be asked by client views to information sources.

Furthermore, given the quality factors of performance and security, client views are often not allowed to access the sources directly. A set of intermediate conceptual views on the enterprise model is therefore often constructed – the conceptual schema of the DW. This set must have the formal property that it allows all expected client views to get answers to their queries, preferably the same they would get if they would access the source views directly.

In terms of DW quality, the enterprise model defines a theory of the organization. Actual observations (i.e. information sources) can be evaluated for quality factors such as accuracy, timeliness, completeness with respect to this theory. Moreover, the fact that data warehouse views intervene between client views and source views can have both a positive and a negative impact on information quality. Positive impact is achieved by using the enterprise model for data cleaning and evidence integration – probably the single most important contributions of the conceptual level in DW practice. Potential negative impact to be taken care of is given by delays in updating the materialized DW views (timeliness of information usually deteriorates in comparison with OLAP data), and by limitations of expressiveness in the conceptual model which may, for

reasons of decidability in reasoning and simplicity of usage, not include all linguistic facilities of all sources.

2.2 The Logical Perspective

The logical perspective conceives a DW from the viewpoint of the actual data models involved. Researchers and practitioners following this perspective are the ones that consider a DW simply a collection of materialized views on top of each other, based on existing information sources. Examples of projects focusing on the logical perspective include TSIMMIS project at Stanford University [Chawathe et al. 1994, Ullman 1997], Squirrel at the University of Colorado [Zhou et al. 1995] and the WHIPS project at Stanford University [Hammer et al. 1995, Wiener et al. 1996].

On the source side, the main problems addressed in this perspective include the wrapping of heterogeneously represented data sources, and the integration of data from multiple sources into particular views. These problems are interpreted both in terms of initial query processing (also called bulk loading in this context) and in terms of incremental maintenance. Particular attention has been paid to so-called self-maintainable warehouse views [Quass et al. 1996] where the DW itself contains sufficient information to maintain all its views, given just the changes in the base data (but no additional back requests to the OLAP system to see how these changes propagate into the views).

On the client side, the two questions of view-based query optimization and incremental update propagation remain central but the focus shifts because user data models are different from the source data models. In particular, the introduction of multi-dimensional data models with aggregation and summarization have a strong impact on algorithms and materialization strategies [Dar et al. 1996, Harinarayan et al. 1996, Levy and Mumick 1996, Srivastava et al. 1996].

Data warehouse products differ in where they introduce these complexities. While MOLAP products already keep the DW views themselves as multi-dimensional data structures, ROLAP products use relational databases to store the DW views and make the transformation to multi-dimensional data models only in the mapping to client views.

Quality factors impacted by the logical perspective concern mainly the coverage of data models at the source and client side, and thus the factor of accessibility of information as well as the interpretability on the client side. In addition, logic-level optimization of queries as well as updates have a strong impact on the amount of data transported and thus on the performance of the system. For example, a good incremental view maintenance algorithm can reduce the daily data transfer from Giga Bytes to small numbers of meta bytes in many large organizations, thus also reducing the time window in which OLTP systems are unavailable for transaction processing due to DW loading. (24X7 availability).

2.3 The Physical Perspective

The physical perspective interprets the data warehouse architecture as a network of data stores, data transformers, and communication channels, aiming at the quality factors of reliability and performance in the presence of very large amounts of slowly changing data.

On the source side, a major topic is to minimize interference with OLTP systems while maintaining a reasonable degree of actuality in the data. Typical research questions therefore include replication policies, recovery from broken bulk transfers, and decisions to what components to allocate certain functions. For example, when reconciling data differences, this could be done in the DW itself, or it could be outsourced to mediator agents with local caches. The SourcePoint tool marketed by Software AG is an example of a commercial loading environment that already allows a lot of customization at the physical level, but could benefit from a more declarative approach in linking to the logical and conceptual perspectives.

On the client side, a central question is how to configure a network of pre-materialized views as to minimize average response times for a given mix of updates and queries. As shown in figure 2, this can involve the usage of tailored data marts as well as a central data warehouse. It can also involve client-side caching of intermediate results to minimize data transfer.

2.4 Relationships between the Perspectives

Generally speaking, the physical perspective has been mostly explored by industry, whereas the conceptual perspective is mainly approached by research. Research and practice meet in the logical level, but with different emphasis, depending on where they come from. Given the quest for quality in data warehousing, these streams of interest are bound to merge.

The steps from conceptual to logical to physical design are common to all database design tasks, however, they have not yet been thoroughly investigated for the special case of data warehousing. Basically, when going from conceptual to logical, we face two problems: (a) the move from a rich conceptual model to an impoverished data model such as the relational one; (b) the fragmentation of a coherent conceptual schema into individual logical data structures. In DWQ, we are exploring the hypothesis that the link between logical and conceptual can be defined in a very similar manner as the link between conceptual source views and the conceptual DW schema, namely through special kinds of views on the conceptual schema.

Going from logical to physical requires the allocation of the identified data fragments and transformation tasks to a network of distributed, cooperating software agents. In this step, considerations of reliability and performance must be added, while semantic considerations must be indirectly expressed through mechanisms that enforce them.

When studying the specific tasks that need to be addressed in adding quality to data warehousing, techniques from all three perspectives are often needed. The next section presents the DWQ approach to each of these tasks.

4. Research Issues

The DWQ project research will be based in the framework shown at figure 2. As we have already mentioned, a DW is a collection of modules, technologies and techniques. The project expects to focus its research effort on the issues described in the rest of this section. Furthermore we will investigate in more details the relationships of these issues with the data quality factors.

4.1 Rich DW Architecture Modeling Languages.

Although many data warehouses have already been built, there is no common methodology which supports database system administrators in designing and evolving a data warehouse. The problem with architecture models for data warehouses is that practice has preceded research in this area and continues to do so. The sheer amount of data to be manipulated under tight constraints forces the industry to quickly introduce new implementation methods. For example, caching and logging is used to recover from failures in the data transfer between data warehouse components. Such methods are typically implementation-oriented. Consequently, the task of providing an abstract model of the architecture becomes more difficult.

The following questions come up:

- What kind of components are used in a data warehouse environment?
- What architectures are used for data warehouses?
- How do the components exchange (meta-)data?
- What is the role of metadata in a data warehouse?
- How can quality factors and architecture be combined?

Our strategy for dealing with this dynamic situation in DW Architectures is to use a representation language whose features can be adapted to an evolving understanding of the what an architecture for data warehouses is about. The framework uses meta models for encoding the key concepts and their semantics. Such meta models can be updated in the same way with database objects. Formally, the reference model corresponds to the schema structure of the meta database that controls the usually distributed and heterogeneous set of data warehouse components and therefore is the essential starting point for design and operational optimization. Expressiveness and services of the meta data schema are crucial for DW quality. The purpose of architecture models is to provide an expressive, semantically well-defined and computationally well-understood meta modeling language. This language is being developed by observing existing approaches in practice and research. The outcome will be a unified terminology that forms the

basis for the repository for data warehouse administration. The architectural reference model will be formally described in this language and its practical usefulness will be evaluated through a prototype implementation, extending currently existing meta modeling tools in the consortium. The structural DW framework will be linked to the quality hierarchy shown in figure 3 through traceability links, following recent work in the representation of non-functional requirements in requirements engineering.

The starting point for the linguistic framework will be our earlier research in description logics and logic-based meta modeling facilities which allows us to treat a DW schema as a set of interacting knowledge bases. Projects such as COMPULOG, AIMS, and CLASSIC have identified description as a promising candidate for rich schema languages for schema integration and inter-view relationship handling, as well as for query optimization and explanation. However, application experiments in database interoperation (e.g. SIMS at ISI, AIMS and MEDWIS in Europe) and data mining (e.g. IMACS at Bell Labs) have shown that the expressiveness of these formalisms is still unsatisfactory with respect to specific areas.

Complementing the traditional multi-database issues which are addressed in existing ESPRIT projects such as IRO-DB, DWQ will focus on modeling and reasoning support for the two central DW mapping problems shown in figure 1: Extraction and Integration, and Aggregation and Customization. The main advance in both cases is the ability to reason about planned integration and aggregation tasks at the schema level, thus facilitating the planning of the integration process and making the quality of data cleaning and the relationships of overlapping derived views visible.

4.2 Data Extraction and Reconciliation

Data extraction and reconciliation are still carried out largely on an intuitive basis in real applications; existing automated tools do not offer choices in the quality of service. The integration process is not sufficiently detailed and documented, so that integration decisions taken are usually very difficult to understand and evaluate. DWQ will provide coherent methodological and tool-based support for the integration activity. The idea of declaratively specifying and storing integration knowledge will be of special importance for supporting high quality incremental integration, and for making all relevant metadata available.

Reconciliation is first a task at the schema level, similar to the traditional task of view integration, but with a richer integration language and therefore with more opportunities for checking the consistency and completeness of data [Catarci and Lenzerini 1993, Huhns et al. 1993]. Second, tools based on the proposed language extensions will facilitate the arduous task of instance-level data migration from the sources to the warehouse, such that a larger portion of inconsistencies, incompatibilities, and missing information can be detected automatically. The integration methodology will provide specific guidelines for controlling the quality of both the integration process and the product of the integration activity.

4.3 Data Aggregation and Customization.

As already mentioned, we consider a data warehouse as a 'subject-oriented', integrated, time-varying, non-volatile collection of data that is used primarily in organizational decision making. The purpose of a data warehouse is to support on-line analytical processing (OLAP), the functional and performance requirements of which are quite different from those of the on-line transaction processing (OLPT) applications traditionally supported by the operational databases. To facilitate complex analyses and visualization, the data in a warehouse are organized according to the Multi-dimensional data model [Gray et al. 1997]. Multi-dimensional data modeling means the partial aggregation of warehouse data under many different criteria, within tuples but also, for example, across time (e.g. time series analysis) and geographical areas. A generic user interface toolkit for such multi-dimensional models is presented in [Gebhardt et al. 1997].

Combining and extending earlier results from the fields of statistical databases with description logics and constraint logic programming, DWQ will enrich schema languages so that they allow the representation of time, space, and numerical/financial domains as well as aggregates over these domains [de Giacomo and Lenzerini 1995]. "Aggregation' means here a grouping of data by some criteria, followed by application of a computational function (sum, average, spline, trend, ...)

for each group. Results on the computational complexity of these language extensions will be obtained, and practical algorithms for reasoning about metadata expressed in these languages will be developed and demonstrated. This will enable design-time analysis and rapid adaptability of data warehouses, thus promoting the quality goals of relevance, access to non-volatile historical data, and improved consistency and completeness.

The semantic enrichment developed in DWQ will not only enhance DW design but can also be used to optimize DW operation, by providing reasoning facilities for semantic query optimization (improving accessibility) and for more precise and better controlled incremental change propagation (improving timeliness). While basic mechanisms stem mostly from active databases, reasoning and optimization techniques on top of these basic services use AI-based reasoning techniques together with quantitative database design knowledge.

4.4 Query Optimization.

DW's provide challenges to existing query processing technology for a number of reasons: typical queries over DW's require costly aggregation over huge sets of data, while OLAP users pose many queries and expect short response times; users that explore the information content of a DW apply sophisticated strategies ("drill down," "zoom out") and demand query modes like hypothetical ("what if") and imprecise ("fuzzy") querying that are beyond the capabilities of SQL based systems.

Commercial approaches fail to make use of the semantic structure of the data in a warehouse, but concentrate on parallelism or make heavy use of traditional optimization techniques such as generating indices or choosing low cost access paths. As a support for OLAP, intermediate aggregate results are precomputed and stored as views.

The approach in DWQ is to exploit knowledge and reasoning about the structure of the DW to process queries efficiently and to support the user in organizing the space of queries that are of interest to him. There are two kinds of meta-knowledge in the DW that are relevant: integrity constraints expressed in rich schema languages and knowledge about redundancies in the way information is stored. Both kinds of knowledge will be used simultaneously for optimization. The starting point for representing schemas and queries are variants of so-called description logics [Buchheit et al. 1994, Baader and Hanschke 1993].

Optimization for nested queries with aggregates will transform a query in an equivalent one that is cheaper to compute. Techniques for constraint pushing will prune the set of data to be considered for aggregation. Integrity constraints will be used to establish the equivalence of queries that are not syntactically similar. Other techniques will reformulate queries in such a way that materialized views are used instead of recomputing previous results. To accomplish its task for queries with aggregation, the optimizer will be capable to reason about complex relationships between the groupings over which the aggregation takes place. Finally, these basic techniques will be embedded into complex strategies to support OLAP users that formulate many related queries and apply advanced query modes.

4.5 Update Propagation.

Updates to information sources first need to be controlled with respect to the integrity constraints specified during the design of the DW and derived views. A constraint may state conditions that the data in an information source must satisfy in order to be of a quality relevant to the data warehouse. A constraint may also express conditions over several information sources that help to resolve conflicts during the extraction and integration process. Thus, the update of an information source database may degrade the quality of information, which therefore should not be considered anymore, or in the same way as before, at the data warehouse. Violations of constraints must be handled appropriately, e.g., by sending messages or creating alternative timestamped versions of the updated data.

Updates that meet the quality requirements defined by integrity constraints must then be propagated towards the views defined at the data warehouse and user level. This propagation must be done efficiently in an incremental fashion [Zhuge et al. 1995, Staudt and Jarke 1996, Mumick et al. 1997]. This recomputation can take advantage of useful cached views that record intermediate results. The decision to create such views depends on an analysis of both the data

warehouse query workload and the update activity at the information sources. This will require the definition of new design optimization algorithms that take into account these activities and the characteristics of the constraints that are considered for optimization.

Integrity constraints across heterogeneous information sources are currently managed manually, resulting in unreliable data. At best, a common reference data model enforces the satisfaction of some structural and quality constraints. However, this approach is quite rigid and does not enable an easy integration of other information sources yet pertinent. By contrast, the DWQ approach suggests a flexible and declarative definition of constraints during the design phase. Constraints will be mapped into active rules, which are increasingly accepted as a suitable base mechanism for enforcing constraints. Building on earlier results concerning the generation and effective management of large rule bases, DWQ will make significant contributions towards the distributed maintenance and execution of active rules in an heterogeneous network. A first approach is reported in [Llirbat et al. 1997].

A final important aspect of data warehousing is its ability to evolve with the user and organization needs. DWQ will develop methodologies for specifying re-design and evolution parameters and processes to allow a DW to follow the dynamics of the underlying information sources as well the user needs.

4.6 Schema and Instance Evolution.

The static architecture, integration, and aggregation modeling languages will be augmented with evolution operators which support controlled change in the DW structure. This includes the addition, deletion, or re-evaluation of information sources, the efficient migration between different structures of derived aggregated data, and the tracing of such changes across the DW architecture. Taken together, these evolution tools will evolve the vision of a data warehouse as a passive buffer between operational and analytic processing towards that of a competitive information supermarket that carefully selects and prepares its products, packages and displays them nicely, and continuously adapts to customer demands in a cost-efficient manner.

4.7 Quantitative Design Optimization.

The above operators provide the means to control and trace the process of evolving a DW application. However, many steps involve options such as what information sources to use, what update policies to select, and what views to materialize. DWQ will develop a 'leitstand' for DW planning which will help the DW manager to quantitatively evaluate these options and find an optimal balance among the different policies involved, prior to executing evolution steps. In particular, policies will be developed for optimized view materialization based on workload parameters (sets of expected queries and updates), while the enriched metadata representation and reasoning facilities used in the DW design will be exploited in order to provide rigorous means of controlling and achieving evolution. In particular, the tools will exploit the potential of subsumption reasoning in order to determine whether certain information sources can substitute each other, whether certain materialized views can be reused to materialize others cheaply. The availability of such subsumption reasoners will significantly change the cost trade-offs, making evolution easier than currently feasible.

Continuous alignment of DW offerings with management needs is a central long-term survival issue for the whole DW idea, therefore our industry partners see a particularly high potential for innovation coming out of this part of the project. An initial quantitative model has been proposed in [Theodoratos and Sellis 1997].

5 Summary and Conclusions

To summarize, the DWQ project will develop techniques and tools to support the rigorous design and operation of data warehouses

- based on well-defined data quality factors,
- addressed by a rich semantic approach,
- realized by bringing together enabling technologies,
- demonstrated in a prototype suite of design and operation tools, and
- validated in co-operation with European DW vendors and users.

DWQ provides the framework and modeling formalisms for DW design, development and maintenance tools that can be tuned to particular application domains and levels of quality. The potential of such an approach has been demonstrated already by successful commercial usage of the ConceptBase metamodeling tool developed in the COMPULOG project.

DW development time for a given level of quality will be significantly reduced and adaptation to changing user demands will be facilitated. There is a high demand for design tools for distributed databases which is not satisfied by current products. DWQ has the potential to satisfy this demand for an increasingly important market segment. In addition, a host of quality-enhanced query and update services can be derived from DWQ results. Prototypes of specific models and tools developed in the project will be experimented with in various industry and public administration settings, in order to gain reference experiences for industrial uptake of project results.

While the development of industrial-strength tools is far beyond the resources of the project, the availability of solid prototypes for semantic software repositories (SIS from ESPRIT ITHACA [Constantopoulos et al. 1995]), meta modeling (ConceptBase from ESPRIT COMPULOG [Jarke et al. 1995]), description logic reasoners, and active databases (A-RDL prototype from ESPRIT STRETCH) provides powerful demonstrator platforms which allow our prototypes to focus directly on the kernel DW problems. Industrial uptake of results will be facilitated by presenting a coordinated demonstration of all project prototypes on a common set of application examples.

The results of DWQ will be demonstrated with a set of prototype tools that are linked together and that are embedded into an environment consisting of commercial DW software.

In the following we give a description of the components to be developed and their interplay, linking over to the approach and workplan section. The *repository* has a key role since it prescribes the structure of the DW and provides the information how to operate it. It is formalized in TELOS [Mylopoulos et al. 1990], a dedicated language for modeling information systems, including aspects like requirements, architectures and operation. Through the powerful metamodeling facilities offered by the ConceptBase implementation of TELOS, virtually all existing data models can be expressed in TELOS. The project will produce the meta-models for quality requirements, architecture, and schemas that are needed to operate the repository.

The design tools are the means by which DW designers and administrators modify the content of the repository. The *integrator* to be built will be used to retrieve the schemas describing the sources underlying the DW and to incrementally generate a DW schema describing the information which is made available. The *design optimizer* determines which views over the integrated schema will actually be stored in the DW to satisfy the information needs of the users. The *evolution support tool* allows for monitoring structural changes both in the information supply and demand and to take appropriate action. It is able to directly access the repository but also to use the view integrator and the design optimizer to optimally adapt the DW structure.

The *update propagator* observes changes at the sources on the data level, checks their consistency with integrity constraints, decides whether they are relevant for the DW, and determines an efficient way of transforming the source update into a DW update. Closely connected to the update propagator is the *data reconciliation module* that makes implicit information in the source data explicit and resolves semantic discrepancies. The *query optimizer* rewrites queries in such a way that the integrity constraints in semantically rich DW schemas and the redundancies in the stored information are exploited. These three components access the repository during the operation of the DW.

All the components described so far have to perform reasoning about integrity constraints and queries that essentially can be reduced to a set of generic tasks including the checking of consistency, containment and view usability. The DWQ project will also produce a reasoning module for a rich description logic by which the various types of descriptions in schemas, queries and views can be captured. The module makes its services available to each of the other components.

References

F. Baader and P. Hanschke. A schema for integrating concrete domains into concept languages. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 452–457, Sydney, 1991.

M. Buchheit, M. A. Jeusfeld, W. Nutt, and M. Staudt:. Subsumption of queries over object-oriented databases. *Information Systems*, 19(1), 1994.

T. Catarci and M. Lenzerini. Representing and using interschema knowledge in cooperative information systems. *Journal of Intelligent and Cooperative Information Systems*, 2(4):375-398, 1993.

S.Ceri, and J. Widom. Deriving production rules for incremental view maintenance. In *Proceedings of VLDB*, Barcelona, Spain 1991.

S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS project: Integration of heterogeneous information sources. In *Proc. of IPSI Conference*, Tokyo (Japan), 1994.

P. Constantopoulos, M. Jarke, J. Mylopoulos, Y. Vassiliou: The software information base -- a server for reuse. *VLDB Journal* 4(1):1-42, 1995.

Sh. Dar, H.V. Jagadish, A.Y. Levy, and D. Srivastava. Answering queries with aggregation using views. In Proc. 22nd International Conference on Very Large Data Bases, Bombay (India), September 1996. Morgan Kaufmann Publishers.

J. Gray, A. Bosworth, A. Layman, and H. Pirahesh. Data cube: A relational operator generalizing group-by, cross-tabs, and sub-totals. *Data Mining and Knowledge Discovery Journal,* 1(1), 1997.

G. De Giacomo and M. Lenzerini. What's in an aggregate: Foundations for description logics with tuples and sets. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, 1995.

M. Gebhardt, M. Jarke, S. Jacobs. A toolkit for negotiation support interfaces to multi-dimensional data. Proc. ACM-SIGMOD Conf., Tucson, Az, 348-356, 1997.

A. Gupta, H. V. Jagadish, and I. S. Mumick. Data integration using self-maintainable views. In *Proc. of the 5th Int. Conf. on Extending Database Technology (EDBT-96)*, LNCS 1057, pages 140-144. Springer-Verlag, 1996.

Michael N. Huhns, Nigel Jacobs, Tomasz Ksiezyk, Wei-Min Shen an Munindar P. Singh, and Philip E. Cannata. Integrating enterprise information models in Carnot. In *Proc. of the Int. Conf. on Cooperative Information Systems (CoopIS-93)*, pages 32-42, 1993

J. Hammer, H. Garcia-Molina, J. Widom, W. Labio, Y. Zhuge. The Stanford Data Warehousing Project. *Data Engineering, Special Issue on Materialised Views on Data Warehousing,* 18(2), pp. 41-48. June 1995.

V. Harinarayan, A. Rajaraman, and J. Ullman. Implementing data cubes efficiently. In Proc. 1996 ACM SIGMOD International Conference on Management of Data, pages 205-227, Montreal (Canada), June 1996.

M. Jarke, R. Gallersdörfer, M. Jeusfeld, M. Staudt, S. Eherer: ConceptBase - a deductive object base for meta data management. *Journal of Intelligent Information Systems*, 4, 2, 1995, pp. 167-192.

A. Y. Levy and I. S. Mumick. Reasoning with aggregation constraints. In *Proceedings of the International Conference on Extending Database Technology (EDBT-96)*, Avignon, France, 1996.

A.Y. Levy, A.O. Mendelzon, Y. Sagiv, and D. Srivastava. Answering queries using views. In Proc. 14th Symposium on Principles of Database Systems, 95-104, San Jose, Ca, 1995..

F. Llirbat, E. Simon., D. Tombroff. Using versions in update transactions. Proc. 23rd VLDB Conf., Athens, Greece, 1997.

J. Mylopoulos, A. Borgida, M. Jarke, M. Koubarakis, "Telos: a language for representing knowledge about information systems", *ACM Transactions on Information Systems,* 8, 4, 1990, 326-362.

I.S. Mumick, D. Quass, and B. S. Mumick. Maintenance of Data Cubes and Summary Tables in a Warehouse. In *Proc. ACM SIGMOD Conf.,* Tucson, Az, 1997.

D.Quass, A. Gupta, I. S. Mumick, and J. Widom. Making Views Self-Maintainable for Data Warehousing. *Proceedings of the Conference on Parallel and Distributed Information Systems.* Florida, December 1996.

Software AG: *SourcePoint White Paper.* Software AG, Uhlandstr 12, 64297 Darmstadt, Germany, 1996.

M. Staudt, and M. Jarke. Incremental Maintenace of Externally Materialized Views. In *Proc. VLDB.* Bombay, India, 1996.

D. Srivastava, S. Dar, H. V. Jagadish, and A. Y. Levy. Answering queries with aggregation using views. In *Proceedings of the 22. International Conference on Very Large Data Bases (VLDB-96)*, Bombay, India, 1996.

D. Therodoratos, T. Sellis. Data warehouse configuration. Proc. 23rd VLDB Conf., Athens, Greece, 1997.

Jeffrey D. Ullman. Information integration using logical views. In *Proc. of the 6th Int. Conf. on Database Theory (ICDT-97)*, Lecture Notes in Computer Science, pages 19-40. Springer-Verlag, 1997.

J. L. Wiener, H. Gupta, W. J. Labio, Y. Zhuge, H. Garcia-Molina, J. Widom. *A System Prototype for Warehouse View Maintenance.* Proceedings of the ACM Workshop on Materialised Views: Techniques and Applications , Montreal, Canada, June 7, 1996, 26-33.

R.Y. Wang, M.P. Reddy, H.B. Kon, 'Towards Quality Data: An attribute-based Approach', *Decision Support Systems* 13(1995)

Y. Wand, R.Y. Wang, 'Anchoring data quality dimensions ontological foundations', *Comm. ACM* 39, 11, 1996

G. Zhou, R. Hull, R. King. *Generating Data Integration Mediators that Use Materialization.*In Journal of Intelligent Information Systems, Volume 6(2), pp. 199-221, 1996.

Y.Zhuge, H.Garcia-Molina, J.Hammer, and J.Widom. View maintenance in a warehousing environment. *Proc. ACM SIGMOD Int. Conf. Management of Data*, pages 316-327, 1995.