Improving OLTP Data Quality Using Data Warehouse Mechanisms

Matthias Jarke, Christoph Quix
Informatik V, RWTH Aachen, D-52056 Aachen, Germany
{jarke,quix}@informatik.rwth-aachen.de

Guido Blees, Dirk Lehmann, Gunter Michalk, Stefan Stierl
Team4 Systemhaus GmbH, D-52134 Herzogenrath, Germany
firstname.lastname@team4.de

Abstract

Research and products for the integration of heterogeneous legacy source databases in data warehousing have addressed numerous data quality problems in or between the sources. Such a solution is marketed by Team4 for the decision support of mobile sales representatives, using advanced view maintenance and replication management techniques in an environment based on relational data warehouse technology and Lotus Notes-based client systems. However, considering total information supply chain management, the capture of poor operational data, to be cleaned later in the data warehouse, appears sub-optimal. Based on the observation that decision support clients are often closely linked to operational data entry, we have addressed the problem of mapping the data warehouse data quality techniques back to data quality measures for improving OLTP data. The solution requires a warehouse-to-OLTP workflow which employs a combination of view maintenance and view update techniques.

1 Introduction

Team4 is a software house focusing on sales support solutions for medium and large enterprises, centering around customized data warehouses for typically 50 to more than 1000 users. Customers include, among others, the chemical giants Bayer and Hoechst, Siemens and other major engineering companies. The company was started in 1994 by four people, partially from industry, partially from research, and has since grown to almost 100 employees. Based on the experience gained in specific customer solutions, the company has for the last few years also been developing a line of tools for data warehouse development and data quality.

The basic product strategy of Team4 involves a solution where you have arbitrary legacy sources, a relational data warehouse kernel and data marts/client caches replicated via Lotus Notes/Domino (cf. figure 1). Methods and tools have been developed to manage the usual stream of data, in particular focusing on aspects of data quality as required by sales personnel. There are also methods and tools for the rapid set-up of data warehouse solutions. These tools were developed based on research results of the IS group at RWTH Aachen [SJ96] and have been used by the company since late 1997. A short description is given in section 2.

The positive impact of improved data quality gained from these usage experiences led to the approach described in the present paper. The observation is the following: Based on the cleaned, integrated and often aggregated materialized views in the client data caches, the sales people make decisions which lead to operational activities such as orders or sales forecasts. However, these orders are again made to the legacy operational systems with all the traditional usability hassle and quality problems. So, the question was: can we somehow re-use the data cleaning mechanisms of the data warehouse to avoid the pollution of the sources, and to use the much more user-friendly data warehouse client front-end to support the update of operational sources? To our surprise, we did not find this problem discussed in the research literature.

The new tool combines ideas from view update technology with the data integration, data replication and data cleaning concepts from data warehousing. A prototype of the tool has been completed in late 1998 and it is currently undergoing beta testing in some applications.

2 Data integration and maintenance

The first tool to be developed was a data warehouse design tool at the relational level, aiming at several data warehouse quality goals [JJQV99]: reusability of solutions, the reduction of telephone costs on the client side, sufficient and flexible freshness of data, ability for evolution of source or data warehouse schemas, and clear process definitions for data integration and refreshment. Prior to the development of this tool, especially the goal of flexibility was hampered by the need to re-program scripts whenever schema or policy changes happened.

Source data are typically stored in distributed data sources. First, the data is extracted from the sources and then pre-aggregated in the warehouse. Further aggregations can be made inside a Notes document. The relational views in the warehouse are designed for use in the Lotus Notes docu-

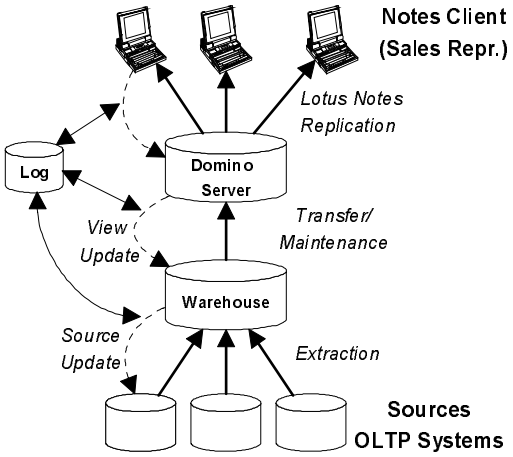

Figure 1: Architecture of the Team4 system

ments, the transfer to the Domino server is therefore straightforward. The documents are typically replicated once per day between the Notes clients and the server.

Direct access to the sources by the extract programs can not be guaranteed, sometimes only snapshots of the source relations are available. The maintenance process must therefore be able to detect changes between two snapshots. The views are decomposed into several self-maintainable views [HZ96]. To deal with this, the view maintenance algorithm of [SJ96] has been extended to cover aggregations.

One goal of the initial project was the easy maintenance of the system. Therefore, a design tool was created to record the schema definitions of the source systems and the view definitions of the warehouse. Most importantly, the design tool automatically creates the SQL statements to initialize and incrementally maintain the views and the transfer program between the data warehouse and the Lotus Notes server. The mobile sales representatives get the relevant and updated documents by using the built-in replication mechanism of Lotus Notes. The results show, that the view maintenance is more efficient than reloading the full replicated views, as long as less than 20% of the source data is updated, which is usually the case.

3 Updating data sources using data warehouse technology

We made the observation, that data warehouse users often make updates on OLTP data while looking at data warehouse data. Traditional systems require that users switch from their data warehouse front-end to another data entry program. The disadvantages of this method are that this manual entry process enforces data quality problems, the semantic information is lost and the relationship with other data in the client views is not recorded.

In our system, sales representatives update their documents in a Notes client. The updates are transferred to the Domino server with the built-in replication mechanism of Notes. This mechanism handles the typical replication conflicts if different users make changes on the same document. The updates in the documents are detected by a Notes script and written to a log. Another program reads the log, and translates updates into updates on the source relations and transfers them to the source databases.

View updates are only possible for a restricted set of views, and often the designer has to resolve conflicts in the translation of view updates to base relation updates [Kel85]. The design tool developed for the view maintenance process was thus extended to deal also with view updates. Because of the complexity and the distribution of the data warehouse system, a transaction control mechanism for the whole system controls the update process between Lotus Notes server, data warehouse and source databases. Any conflicts in the update process are written to the log, and reported to the administrator and user by a Notes document. If an update is rejected by the sources, the update must be canceled, and the already committed transactions in the warehouse and Lotus Notes must be undone. Restrictions in access rights are handled either in the translation programs for updates or already in the design of a Notes document. The behavior of the whole update and maintenance system can be configured at design-time in the design tool and at run-time with a general control document in Notes.

The approach can also be used for adding manual estimates to data warehouse views during the decision support task itself, such as the sales forecast for a company leading to internal order operations. Based on the orders in the past months and some knowledge on the production plan of the customer, a sales representative enters the expected orders into his document. The information is then transferred to the central warehouse, where the expected orders of all customers are aggregated and the plant utilization for the next months can be predicted.

4 Conclusions

Data warehouse research has contributed significantly towards improved data quality for decision makers [JJQV99], cleaning data as they flow from sources towards clients. However, a global optimization of information flows might be at least equally well reached by cleaning data already before they reach the sources. The Team4 solution sketched in this short paper is a first step towards this goal.

References

[HZ96] R. Hull, G. Zhou. A framework for supporting data integration using the materialize and virtual approaches. *Proc. ACM SIGMOD Intl. Conf. Management of Data*, Montreal, Canada, 1996.

[JJQV99] M. Jarke, M. Jeusfeld, C. Quix, P. Vassiliadis. Architecture and quality in data warehouses: An extended repository approach. *Inform. Sys. 24:4*, 1999.

[Kel85] A.M. Keller. *Updating Relational Databases Through Views.* PhD. dissertation, Stanford University, 1985.

[SJ96] M. Staudt, M. Jarke. Incremental maintenance of externally materialized views. *Proc. 22nd Conf. Very Large Data Bases*, Bombay, India, 1996.