A Design Methodology for Data Warehouses

Olaf Herden

Oldenburg Research and Development Institute
for Computer Science Tools and Systems (OFFIS),
Escherweg 2, 26121 Oldenburg, Germany
olaf.herden@offis.de

Abstract

The aim of this work is to construct a design methodology for engineering data warehouses. The methodology is founded on three–level–modeling whereby a main focus is on conceptual modeling. But subsequent phases like logical and physical database design are also treated with special consideration of data warehouses.

Keywords: data warehousing, conceptual modeling, database design.

1. Introduction

The increasing popularity of data warehouses (DWs) [3] as backbone of decision support systems reflects the rising requirement to make strategic use of data integrated from heterogeneous sources. Some examples for using the data stored in a DW are database marketing and controlling. These application scenarios are very important for the organizations and they all have in common that the underlying DW has to be reliable, maintainable and expandable. For ensuring these quality aspects a systematic engineering of DWs is necessary. To attain this goal a design methodology both falling back on the experiences from designing classical OLTP (Online transaction processing) databases and considering the special aspects of DWs is necessary.

The aim of this work is to build such a design methodology. Lessons learned while designing conventional OLTP databases should be considered, e. g. starting with conceptual modeling, followed by a transformation to the logical level and a physical design step. On the other hand, different requirements to both types of databases have also to be taken into account, e. g. considering the multidimensional model on the conceptual level or optimizing the physical design for minimal response time in a read–mostly environment instead of optimizing an OLTP database for high throughput of transactions.

To tackle the research problems described above the main actions of designing data warehouses should be sketched, algorithms and methods should be constructed. The design methodology should be tool–supported continuously. One aim is to extend existing products. Those components where this is not possible are implemented prototypically in order to be able to demonstrate their basic functionality. To prove the soundness of the design methodology and its implementation, we finally want to apply it to a real–world application.

2. Preliminary Results

Up to now, the design process is outlined (see figure 1). On the conceptual level we are distinguishing between language and graphical representation to offer interoperability between tools. Therefore, we have developed a multidimensional meta language called MML (Multidimensional Modeling Language) [1], [2]. MML is an object–oriented language and therefore provides a good basis for flexible,
implementation–independent modeling. Moreover, MML meets the needs of conceptual multidimensional models like e. g. distinguishing between dimensional– and fact–classes or providing the possibility to model sophisticated dimensional structures. The MML is specified semiformally by an UML–diagram, class descriptions and elucidating prose.

![Diagram of methodology for designing DWs]

With the MML as basis different front end tools can be used. Exemplarily, we have developed an extension of the UML (Unified Modeling Language), called mUML (multidimensional UML). By using the concept of stereotypes for extending the UML we have defined new stereotypes to model the different types of classes and to mark the connections for building hierarchies. The implementation of the mUML has been done as an extension of the commercial CASE tool Rational Rose. The MML is implemented as a class library in C++. For storing MML–diagrams persistently an ORACLE database is used at the moment, but we want to change to an extended version of the Microsoft Repository as soon as possible.

Assuming a relational database, for satisfying the mapping from the conceptual to the logical level we have defined and implemented a mapping T transforming MML–diagrams into a special kind of relational schema, called common star schema. Beside relations and attributes, a common star schema consists of multifarious meta data. These meta data carry the information of the multidimensional aspects of the MML–diagram which can not be mapped to tables and attributes directly. The work on the physical design step is just in the beginning. As basic idea a three–step approach should be realized: first, an algorithm transforming a common star schema into a basic working database schema should be realized. This schema should be independent of the DBMS and OLAP–tools to be used. Then follows an algorithm transforming the basic working database schema into a schema already considering some special needs of the DBMS and OLAP–tools (e. g. denormalization of hierarchies) but not considering any issues of tuning. This should be done in the last step of physical database design when the schema will be optimized for analytical use.

3. References

