

# CoSM: A Maintenance Tool for Data Warehouse Structures

Johann Eder and Christian Koncilia

University of Klagenfurt  
Department of Informatics Systems  
{eder, koncilia}@ifi.uni-klu.ac.at

## Abstract

CoSM (Comet Structure Manager) is a tool for the maintenance of data warehouse structures based on the temporal data warehouse model COMET. Current data warehouse systems do not care for changes in dimension data. CoSM allows keeping track of modifications made in the dimension-structure of multidimensional cubes stored in an OLAP (On-Line Analytical Processing) system. CoSM can download dimension data from an OLAP System, and upload dimension data into an OLAP system, compare dimension data of an OLAP system with stored versions and provides functions for insertion, deletion and change of dimensions, levels, dimension members, and their relationships.

## 1. Introduction

Data Warehouses are materialized collections of data from probably different heterogeneous data sources with the aim to improve the decision-making process [EN00]. Hence, data warehouses are building blocks for many information systems, in particular systems supporting decision making, controlling, revision, customer relationship management (CRM), etc. [HLV00]. OLAP (On-Line Analytical Processing) tools are being used to analyze the data stored in data warehouses along different dimensions relevant to the application domain [EKM02]. Examples of dimensions are time, organizational structure, space (cities, regions, etc.), and product data.

In contrast to snapshot-based OLTP (On-Line Transaction Processing) systems, data warehouses provide long term data that can be analyzed along the time axis. OLAP systems and data warehouses are built to deal with changing values of fact data, e.g., changing turnover or costs. Surprisingly, they are not able to cope with modifications in the dimension structure in a sophisticated way, although, such modifications happen frequently, too. It is, however, vital for the accuracy and correctness of results of OLAP queries that modifications in the structure of dimensions are correctly taken into account, in particular, when comparing data over several periods, computing trends or computing benchmark values from data of previous periods.

The maintenance of structural modifications in data warehouses is a crucial point for keeping track of structural modifications and considering these modifications in analytical queries. In this paper, we will present our data warehouse maintenance system CoSM (COMET Structure Manager) built on our temporal data warehouse model COMET [EKM02].

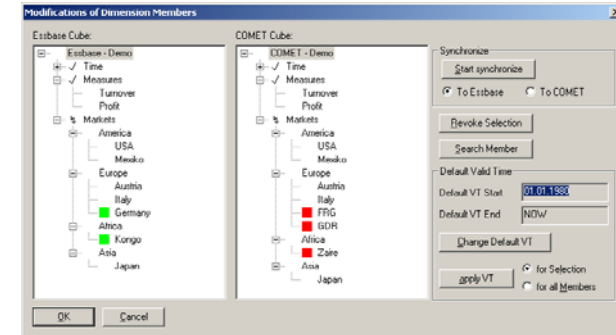


Figure 1: CoSM screenshot: comparing two cubes

## 2. System Overview & Functionality

The COMET metamodel [EKM02] is the general framework for the CoSM system. COMET introduces the extensions necessary representing and correctly treating the change history of dimension data in multidimensional data warehouses. We use the term *dimension data* for both schema (dimensions and categories) and instance (dimension members) data.

Time stamps are used to represent the valid time of dimension data. Valid time defines the time, in which a fact is true in the real world [EJS98]. We use time stamps for both schema and instance data. Hence, our system allows to correctly deal with modifications of both the schema and the instances of a data warehouse.

By providing time stamps for dimension data we are able to identify different structure versions. A structure version is a temporal view defined on a data warehouse (its schema definitions and its instances) that is valid for a given time interval such that during this interval no changes to the dimension structure took place.

We have adopted these two ideas for the CoSM system. The CoSM tool provides three main functionalities: „Upload“, „maintain“ and „download“ of dimension data.

**Upload:** Existing multidimensional cubes stored in the OLAP-Server software may be uploaded into the COMET database. Each element is timestamped, i.e., we record the valid time of each dimension, category, instance and so on. If the uploaded cube already exists in the COMET database, then both cubes get compared. A new structure version of the cube is generated in the COMET database, if CoSM detects any differences during this comparison.

Figure 1 shows a screenshot of the GUI supposed to compare two cubes. Both cubes are depicted as trees. The tree on the left shows the cube to be uploaded and the tree in the middle of the dialog shows the most recent version within the COMET database of this

cube. The information to the right of this window may be used to set the valid time of all or of all selected instances.

**Maintain:** Cubes stored in the COMET database can be maintained, i.e., the user may change the structure of a cube. This can be done by selecting one or more dimensions stored in the COMET database. After the user has selected the dimensions that he or she wants to maintain, the CoSM system generates a cube in the OLAP-Server software with these dimensions and uses its GUI to enable the user to modify elements of the dimensions. After all modifications have been made, the user may synchronize them with the COMET database and create a new structure version of these dimensions.

**Download:** Finally, the user may download a cube from the COMET database into the OLAP-Server software. To do this, the user selects a cube and a structure version of this cube. The result of this operation is an accurate copy of the selected cube, i.e., the selected structure version of this cube. This comprises not only the structure of the cube, but also all database settings of the OLAP-Server that were valid for this cube, such as: user defined attributes, aliases, dimension attributes, database attributes, and so forth.

Several reports may be generated, e.g. a comparison report of two dimensions selected by the user, where the dimension may be even part of different cubes.

Moreover, cubes stored in the COMET database may not only be exported to the OLAP system, but also to text files. The user selects a cube and a structure version of this cube: the dimension data of this cube are then exported to a plain text file. This file may be used to import a cube into different OLAP products, for instance Oracle Express, Cognos PowerPlay, Hyperion Essbase, etc.

### 3. CoSM Architecture & Implementation

Figure 2 shows the architecture of the CoSM system. Hyperion Essbase was selected as OLAP-Server software for the alpha implementation of CoSM. We have used the API (Application Programming Interface) provided by Hyperion Essbase to access the cubes stored in Essbase, and ODBC to communicate with a relational database.

The Comet Structure Manager itself has been implemented with Microsoft Visual C++. All functions are accomplishable through a graphical user interface (GUI). It serves as an interface between a non-temporal OLAP database (Hyperion Essbase) and a temporal relational database (the COMET metamodel implemented in Oracle).

Database security was a crucial point during design and implementation. The CoSM system enables the administrator to define several users. For each user the administrator may grant different authorization levels for different dimensions. Furthermore, all operations performed using the CoSM system may be monitored.

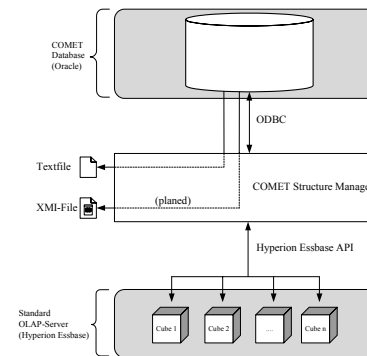


Figure 2: CoSM Architecture

### 4. Conclusions

The COMET Structure Manager (CoSM) is useful to keep track of modifications made in the dimension structure of multidimensional cubes. We showed how users can upload cubes (stored in Hyperion Essbase) into the COMET database and how they can timestamp each element of the cube. These cubes can be maintained using the CoSM system. Furthermore, CoSM is able to export an accurate copy of a cube (i.e., the selected structure version) to Hyperion Essbase.

The CoSM system is currently in use in a Carinthian company with more than 7.000 employees. Hyperion Essbase (Version 6.1) is used as OLAP-Server to store several multidimensional cubes. The CoSM system supports the maintenance of these cubes, i.e., to update their structure, to keep track of the valid time of different structure versions of the cubes, to store information about who modified which parts of a cube and so forth.

### References

[EJS98] O. Etzion, S. Jajodia, and S. Sripada, editors. Temporal Databases: Research and Practise. Springer-Verlag (LNCS 1399), 1998.  
[EKM02] J. Eder, C. Koncilia, and T. Morzy: The COMET Metamodel for Temporal Data Warehouses. In Proc. of the 14th Int. Conf. on Advanced Information Systems Engineering (CAISE'02), 2002. Springer Verlag (LNCS 2348).  
[EN00] R. Elmasri, S.-Navathe. Fundamentals of Database Systems. Addison-Wesley, Reading, MA, 3 Edition, 2000.  
[GJM03] C. Ghezzi, M. Jazayeri and D. Mandrioli. Fundamentals of Software Engineering. Pearson Education, 2 edition, 2003.  
[HLV00] B. Hüsemann, J. Lechtenböcker, and G. Vossen. Conceptual Data Warehouse Design. In Proc. of the Int. Workshop on Design and Management of Data Warehouses (DMDW 2000), 2000.