Data Warehousing Has More Colours Than Just Black & White

Thomas Zurek
Email: Thomas.Zurek@sap-ag.de

Markus Sinnwell
Email: Markus.Sinnwell@sap-ag.de

SAP Business Information Warehouse
P.O. Box 1461
61985 Walldorf
Germany

Abstract

Data warehouses are frequently described to be orthogonal to operational database systems. In this paper, we show that such a point of view is often misleading as a data warehouse (a) often has to share structures and issues with the OLTP systems that are linked to the data warehouse and (b) serves as an infrastructure element to support other tools for strategic management. We describe how data warehousing requirements evolve in that context.

1 Introduction

Since the advent of data warehousing in the early 1990s, many (marketing and research) papers, books, brochures etc. have been written on the topic. A lot of them argue that data warehouses (DW) are somewhat othogonal to the traditional operational database (ODB) systems. To that end, many contrasting pairs of expressions have been cited in order to underpin that notion:

- transactional vs. analytical processing
- operational vs. informational processing
- application-oriented vs. subject-oriented
- predictable retrieval vs. ad-hoc retrieval
- OLTP vs. OLAP
- detailed vs. summarised data
- normalised vs. denormalised data

See [Chaudhuri and Dayal, 1997], [Anahory and Murray, 1997], [Berson and Smith, 1997], [Bontempo and Zagelow, 1998] or the many industrial white papers that you find at locations like [DataWarehouse.com, 1999] or [DW-Institute, 1999].

In this paper, we will not argue against these 'black & white' patterns; they certainly focus on important issues. They also clarify to the non-expert what the ideal distribution of tasks between ODBs and DWs should look like. In this paper however, we will abandon the prototypical situation and turn to real-world-scenarios of data warehousing. There, many of the patterns that are mentioned above look more colourful rather than just 'black & white'.

The background of this paper is as follows. In 1998, SAP released the *Business Information Warehouse (BW)* [SAP, 1997], a complete standard solution for data warehousing. In March 1999, it had been delivered at over 400 customer sites worldwide. During that year, we were able to watch many customer projects, their approaches, their requirements and especially the way in which they perceived data warehousing as the technology that can possibly and hopefully resolve some of their business problems. In this paper, we will summarise some of this experience, particularly with respect to those patterns above.

The remainder of this paper is organised as follows. In section 2, we describe the typical environment that is created through enterprise resource planning (ERP) systems and that is the base of our discussion. In section 3, we discuss four data warehousing issues that do not match the general, orthogonal patterns that should distinguish DW and ODB systems. Finally, the paper is concluded in section 4.

Proceedings of the 25th VLDB Conference, Edinburgh, Scotland, 1999.

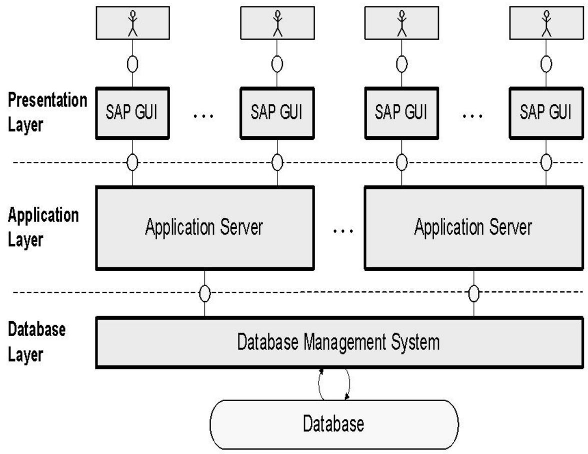

Figure 1: The three-tier client-server architecture as employed by R/3.

2 ERP-Environments

2.1 ERP-Systems

In order to understand a user's point of view on data warehousing, one has to look at the state of ODB data management. In this section we want to focus on the particular sector of enterprise resource planning (ERP) systems. The latter constitute a considerable part of the wider OLTP market.

A typical ERP-system, such as SAP's R/3 product, integrates a wide number of business applications that cooperate over a single database. Typical business applications are financial accounting (FI), controlling (CO), sales and distribution (SD), materials management (MM), production planning (PP), plant maintenance (PM) etc. Many of these applications share data. For instance, objects such as *customer orders* or *articles* play major roles in SD, CO, PP and many more. The major benefit of an integrated ERP-system is that all these applications can cooperate over the same database. In that way, one matches the natural fact that an object (like *order*) passes several departments (application areas). Ideally, every application could run on its own, private server. In summary, this motivates a three-tier client-server architecture (see figure 1). Someone who uses such an integrated system might wonder why something like a data warehouse is required: the central database theoretically provides opportunities for cross-application data analysis and reporting. And in fact, most of these systems provide (at least statical) querying facilities, even across several systems. Thus, this widely cited argument in favour of a data warehouse might fade away in front of some users.

2.2 ERP & Data Warehousing

Figure 2 shows an example of data warehousing in an ERP-environment. There are tools for planning, such as SAP's APO (advanced planning & optimisation) or SEM (strategic enterprise management) products, and for CRM (customer relationship management). In practice there are many more such tools. They are all based on anlytical information about the state of the company and its businesses. Such information is provided by a data warehouse system which itself is linked to many sources of operational data, such as R/3, B2B (business-to-business procurement) or legacy systems.

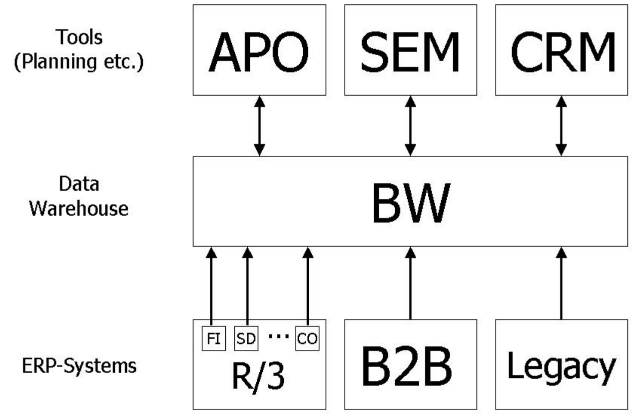

Figure 2: Data warehousing in an ERP-environment.

In such a context, a data warehouse (a) provides not only powerful staging and OLAP engines for consolidated data extraction and integrated reporting, respectively, and (b) serves as an infrastructure element to support further tools for strategic management. Below, we will discuss some of the services that a DW might provide for those tools.

3 Non-Standard Data Warehousing Requirements

In order to show the wide (colourful) spectrum of data warehousing we now dive into some issues that go beyond *slicing*, *dicing* and the traditional black & white patterns that we mentioned in section 1. Those issues have come up with many of our customers. We will try to show the motivation of such requirements.

Realignment

The term *realignment* is used to refer to structural changes in the data warehouse. They might – but do not necessarily imply – changes to the data schema. Realignment plays an important role in data warehousing as such systems are tuned towards fast reporting which is achieved by fixing and precomputing certain portions of the data. Any changes can impose major reorganisations of the underlying data. We would like to look at one issue that is likely to affect a major overhaul of certain structure in a data warehouse, namely *hierarchies*.

Hierarchical relationships are frequently used in OLAP. They are used to summarise information on certain (hierarchical) levels of interest. Simply imagine the time hierarchy of year – month – day that is used to look at sales figures on a yearly, monthly or daily level, respectively. Other typical examples for reporting hierarchies are:

1. In a sales scenario, there are *sales regions* divided into *sales districts* which are again divided into *sales offices.*
2. *Products* are usually bundled to form *product groups.* However, product groups itself can be considered as products[1] that form part of a superior product group.
3. *Cost centres* usually match the organisational structure of a company with departments being divided into several (sub-) departments that consist of several groups. Here, there is no fixed depth of the hierarchy as some departments might have a deep organisational structure while others might not.

These examples show a number of problems with respect to realignment; such problems do not exist in static and simple examples as the one of the time hierarchy:

- All these hierarchies might change regularly: the sales organisation (1.) or the cost centre (3.) hierarchies are likely to change frequently due to reorganisations within the company. This can trigger major reorganisations within the data warehouse if the hierarchical changes are to be applied to historical data as well. The latter situation is not rare as data warehousing data is frequently used for planning purposes (e.g. to improve organisational structures).
- Many hierarchies do not have a fixed number of levels. Usually, each level is represented as a column within the table that describes a dimension of a multidimensional data cube. This solution does not work when the number of levels is not known in advance. Workarounds comprise the use of recursive levels as in the product hierarchy (2.). There, a product group can be considered as a product. However, this moves the problem to query processing where the recursive definition has to be resolved.
- Originally, many of the hierarchies are defined and maintained in the OLTP systems. Cost and profit centre hierarchies, for example, are exposed to frequent[2] changes. Many customers expect to synchronise the OLTP and the OLAP hierarchies.

[1] Just imagine software products like word processing, spreadsheet, drawing, database etc. programs being sold individually as well as part of a software package.

[2] Monthly changes are not rare. In extreme cases, we have seen daily and weekly changes.

Level-of-Detail (Line Items)

In contrast to OLTP-systems, it is assumed that data is *summarised* when loaded into a data warehouse. However, there are various scenarios that do require data on a detailed level. A commonly used term to refer to data on a detailed level is *line item.* A typical example for a line item is an order or a receipt. In areas of intense competition, many companies only achieve a competitive advantage by carefully analysing their customers' buying habits (basket analysis). To that end, they require detailed data such as the point of sales information.

In terms of a multidimensional data schema, this translates into a degenerated dimension with many entries. In turn, this creates query processing problems as many star query processing techniques cannot be applied in such a case.

Updates

Another assumption that is frequently cited to be a distinguishing characteristic between DW and ODB management is that data is read-only, i.e. it is never updated. Unfortunately, this is not true as we describe in the following two scenarios.

Many companies regularly evaluate the quality of their services. A prominent example is that of customer order being delivered on time. In that context, orders are classified to be *open*, *delivered*, *billed* or *closed.* Now, a controller might want to see how many customers do not receive any items within two weeks. To answer this query, it is necessary to already hold those orders in the DW whose state is *open.* However, this implies that – later, when the state changes to *delivered*, *billed* and *closed* – updates in the DW are required.

A second example arises in the context of planning tools being used in conjunction with a DW. Many of those tools are based on the kind of data that is found in a DW, i.e. summarised, aggregated on certain (e.g. hierarchical) levels, cleansed and scrubbed. Those tools start to plan on that kind of data in order to project sales, revenue etc. figures to the future. Afterwards, one usually wants to compare planned versus actual data. To that end, the plan data is pushed back into the data warehouse. This is not a particular problem. However, planning is an iterative process: initially, crude estimates are created and then those estimates are subsequently refined. Translated into database management terms this means that a certain portion of the fact table data (i.e. the part representing the plan data) is subsequently updated. This, in turn, imposes a considerable burden on (query) performance enhancing techniques in the data warehouse such as the maintenance of materialised views.

Frequency of Data-Uploads

An OLTP system processes many writing transactions each second. In contrast to that, the writing operations in a DW are concentrated on certain windows with no or few reading operations, such as weekends or nights. This is again a result of a data warehouse creating redundant, precomputed data portions (e.g. denormalised schemas or materialised views) that allow fast reporting possibilities.

Frequently, it is not easy for a user to accept that inserting new data into a DW requires major efforts. While some users simply regard a data warehouse as a reporting tool (that should be synchronised with the OLTP systems) there is a whole bunch of scenarios that require an upload frequency that is well below a daily interval. A prominent example is again planning on top of historic data (provided by a DW) and subsequently refining that plan data.

One can think of various mechanisms to support such processes without destroying the existing data warehousing mechanisms. However, from a conceptual point of view, this concedes that, in such scenarios, DW data has to be much more up to date and that the actuality of data is not necessarily as a distinguishing factor between OLTP and DW as it is often assumed.

Predictable vs. Ad-Hoc Retrieval

Many companies have sets of standard reports that are run daily, weekly or monthly to determine key (company) performance indicators. Very often, such indicators have been well defined (and refined) by those companies over many years. Those standard reports are the result of this process.

Ad-hoc reporting is certainly an issue, especially with respect to data mining. However, it is only one part of OLAP beside the sets of standardised queries.

4 Conclusions

In this paper, we have discussed and motivated certain issues of data warehousing that contrast the black & white patterns that are frequently cited to show that DW and ODB systems are somehow orthogonal. We argue that in real-world scenario the orthogonality often disappears as there is a need to bridge the gap between the prototypical DW and ODB systems.

References

Anahory, S. and Murray, D. (1997). *Data Warehousing in the Real World*. Addison-Wesley, 1st edition.

Berson, A. and Smith, S. (1997). *Data Warehousing, Data Mining & OLAP*. McGraw-Hill.

Bontempo, C. and Zagelow, G. (1998). The IBM Data Warehouse Architecture. *Communications of the ACM*, 41(9):38–48.

Chaudhuri, S. and Dayal, U. (1997). An Overview of Data Warehousing and OLAP Technology. *SIGMOD Record*, 26(1).

DataWarehouse.com (1999). DataWarehouse.com – White Papers. http://www.datawarehouse.com.

DW-Institute (1999). Data Warehousing Institute – White Papers. http://www.dw-institute.com.

SAP (1997). Business Information Warehouse – Technology. White Paper. available via http:// www.sap-ag.de / products / bw / index.htm.

SAP (1998a). Advanced Planner and Organizer. White Paper. available via http:// www.sap-ag.de / products / apo / index.htm.

SAP (1998b). R/3 System. White Paper. available via http:// www.sap-ag.de / products / r3 / index.htm.

SAP (1999). SAP Customer Relationship Management Initiative. White Paper. available via http:// www.sap-ag.de / crm / index.htm.