Evolution not Revolution: The Data Warehousing Strategy at Credit Suisse Financial Services

Markus Tresch and Dirk Jonscher

CREDIT SUISSE FINANCIAL SERVICES
Technology & Services
8070 Zurich, Switzerland
{markus.tresch, dirk.jonscher}@csfs.com

Abstract. Data Warehousing is not new to Credit Suisse Financial Services. Over the past twenty years, a large number of warehouse-flavored applications was built, ranging from simple data pools to classical management information systems, up to novel customer relationship management applications using state-of-the-art data mining technologies.
However, these warehouse projects were neither coordinated nor are they based on the same infrastructure. Moreover, dramatic changes of the business design had a huge impact on information analysis requirements. Both together resulted in a nearly unmanageable complexity.
Therefore, Credit Suisse Financial Services started a 3-year enterprise-wide data warehouse re-engineering initiative at the beginning of 1999. This paper presents the motivation, experiences, and open issues of this strategic IT project.

1 Introduction

Driven by urgent business needs, various data warehouse-like systems were built in Credit Suisse Financial Services (CSFS) over the past twenty years. The early systems are mere data pools serving batch reporting purposes for financial and management accounting. The second generation of data warehouses supports preliminary ad-hoc analysis capabilities and serves credit, payment, and securities businesses. Customer relationship management (CRM) started ten years ago with a data pool dedicated to customer profitability analysis. The system was recently replaced with a state-of-the-art warehouse now supporting data mining and on-line analytical processing (OLAP).

Recent trends are nearly real-time data warehouses (based on operational data stores) and hybrid systems, where analysis results are fed back into online transaction processing systems (e.g. reclassification of banking customers in accordance with analysis results). Moreover, ERP packages, like SAP or Peoplesoft, are advertising their own interpretation of data warehousing.

Moreover, a large number of applications implemented warehousing and analysis functionality, like for example, economic research, credit early warning, comprehensive risk analysis, click stream analysis in e-business, or market positioning.

Summing up, data warehouses in CSFS have very complex application and data dependencies. Data sharing is not coordinated, data sources generate multiple (overlapping) feeder files for different warehouses (resulting in unnecessary load at the feeder systems), and there are cascading warehouses, etc. (cf. Fig. 1). Even worse, there is no global view which data are stored where and are used for which purpose.

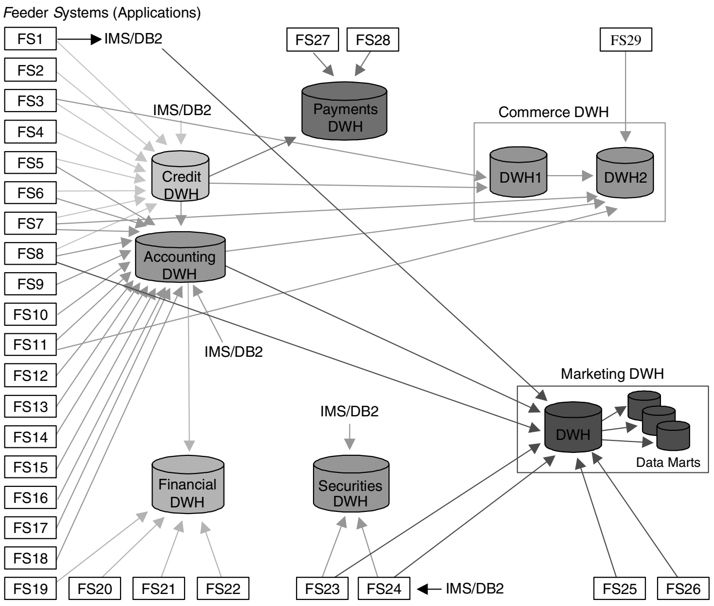

Fig. 1. Historically evolved data warehouse environment

From the technical point of view, many different platforms, tools, and architectures are in place. Platforms include Mainframe / DB2, IBM RS 6000 SP2 / DB2, Sun E10000 / Oracle and Mainframe / SAS. Tools like SAS, ETI, PowerCenter, Trillium, Darwin, Clementine, Microstrategy and Brio are heavily used. However, most of the old systems use their own home-made analysis tools. The systems are in different states of the product life cycle (from antic to state-of-the-art). Consequently, each warehouse was maintained by a dedicated team of IT people resulting in a very poor overall productivity.

From a business perspective, there is an end-to-end ownership and sponsorship of each data warehouse system.

2 Strategy of Managed Evolution

An in-depth analysis of the existing data warehousing systems revealed the following:

- Data warehouse development has become expensive and slow because of the high complexity of the historically evolved environments and the very different warehouse platforms being used. Even worse, a stable production environment could not be provided any more.
- A high percentage of resources were required to *maintain* old systems instead of developing new solutions for business users.
- The integration of off-the-shelf software was very difficult because interfaces were either not compliant with standards or were not clearly defined and documented.
- The implementation of new business requirements, like alternative distribution channels, was very hard.
- An enterprise-wide data quality process could hardly be established, since each warehouse implemented its own data semantics.

The following major alternatives to improve the situation were investigated:

- *Green field approach*

  Starting from scratch, a completely new huge (multi-terabyte) enterprise data warehouse is designed, which replaces all existing systems. This approach was not considered a real option due to very high costs, long duration, unpredictable risks, and a general lack of skills and resources.
- *Migration to an off-the-shelf package*

  A data warehouse solution, for example from an ERP vendor, is purchased that fulfils the requirements to a large degree. This was also no feasible option because of the huge data volume and the specific needs of CSFS.
- *Take over a proven data warehouse solution from another bank*

  A warehouse system is bought from a competitor. However, such a solution is only possible in case of a merger with or acquisition of another bank.
- *Opportunistic approach*

  Existing warehouses are improved without any fundamental change. In principle, this means to follow the same road down as we did in the past. As mentioned above, this is no long-term solution, especially since time-to-market of new projects gradually increases.
- *Managed evolution*

  Existing warehouses are improved, including *fundamental* changes with respect to the warehouse platform and architecture. Managed evolution requires a continuous balance between a purely IT-driven approach, opting for very fast solutions which are maximum coherent to a target data warehouse architecture, and a purely business driven approach, based on opportunistic implementations of business functionality for maximizing quick wins (cf. Fig. 2). For CSFS, managed evolution was the only realistic option.

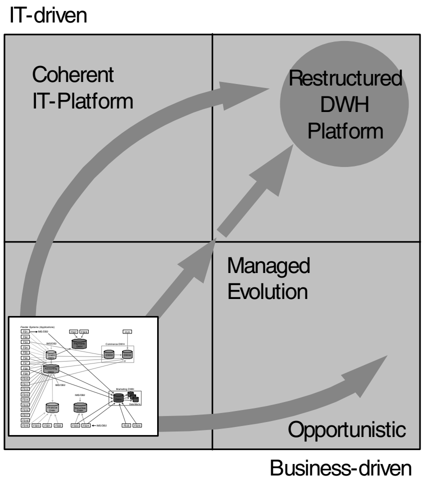

Fig. 2. Managed evolution data warehouse strategy

3 Data Warehouse Re-architecture Program

In 1999, the data warehouse re-architecture program was started to migrate all existing warehouses to an enterprise-wide target architecture in a step-wise process (managed evolution). The main task was to design and maintain an overall DWH architecture (target architecture), including system, application, data and metadata architecture. Furthermore, a migration path for all existing DWH to the target architecture was designed to ensure that all warehouse projects eventually conform to the target architecture. A standard tool suite for data warehouse design, implementation and maintenance was set up. A dedicated technical data warehousing platform was defined, built, and maintained. The required business organizations and processes/responsibilities to manage DWH were also implemented.

The migration is being executed in two major steps:

• migration phase:	all warehouses are moved to the dedicated data warehousing *platform*
• consolidation phase:	all warehouses conform to the standard warehousing *architecture*

The first phase was successfully completed at the end of 2000, except for one of the legacy systems.

The following picture shows a high level view of the data warehouse target architecture. Data warehouses are strictly separated from operational systems (in Fig. 3, the latter is referred to as application domains).

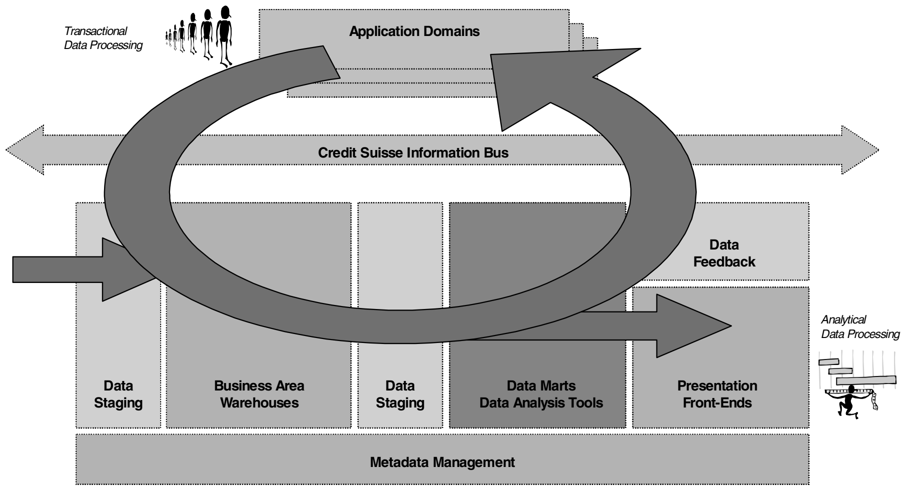

Fig. 3. Data warehouse target architecture

The Credit Suisse information bus (which is based on CORBA technology) is used to periodically transfer data from operational systems to the warehouse platform, and to feed analysis results back into operational systems. Note that data are never "improved" within the warehouse processing chain (errors have to be fixed in the source systems) and that no on-line access from DWH to OLTP systems for data extraction is allowed.

The warehouse domain consists of the following building blocks:

- The *data staging area* stores the feeder files which are delivered by feeder systems. An ETL-tool (PowerCenter from Informatica) is used to *e*xtract and *t*ransform these data and *l*oad them into business area warehouses.

  The data staging area ensures that each data item is extracted only once from the source systems and is shared between all warehouse applications. External data (like demographic information purchased from data providers) are also loaded into the data staging area.

- *Business area warehouses* are the large strategic warehouses which serve several analysis and decision support systems. They are centrally managed by IT experts.

  In contrast to an enterprise data warehouse, we are implementing multiple systems each storing data of one particular business area (e.g. financial data, credit data, marketing data, ...). These warehouses are fully normalized and store data at entity level (i.e. non-aggregated). They can be understood as the shared stock of data for all warehousing applications.

  Only a few power users are allowed to access business area warehouses directly.

- In a second staging step (where the same ETL-tool is used) data are extracted from business area warehouses and loaded into *data marts* which are optimized in accordance with well-defined business cases. This means, data marts only contain de-normalized subsets of all data based on multi-dimensional data models. Data marts have a single business owner and serve a rather homogeneous set of end users.

Several *data analysis* tools (application servers) can be used to access data marts (reporting, OLAP and data mining tools).

- *Presentation front-ends* – the warehouse clients – are strictly separated from analysis tools since we aim at a thin client strategy. End users only need a standard PC and a web browser like Netscape (which connects to application servers) for analyzing data.
- The *data feedback area* is like a staging area but prepares analysis results for being returned into operational systems.

  We support pull mechanisms, where operational systems fetch data from the feedback area, as well as push mechanisms, where the feedback area passes data to operational systems using standard application interfaces.
- *Metadata management* has an IT and a business view. The IT view allows IT professionals to centrally manage the warehouse (operational metadata). It also allows for an efficient implementation of new reports or even data marts. The business view allows end users to learn which data are where available in the warehouse system (data model and data location) and to get additional explanations on the data in the warehouse (data semantics and data lineage).

4 Experiences and Open Issues

After two years of the re-architecture program, the platform – including scalable hardware and all software tools – has been implemented to a large degree and is pretty stable. The platform consists of 2 Sun E10000 Starfires (2 x 64 SPARC II CPU), approx. 15 TB storage managed by 3 Hitachi disk subsystems (1 HDS 9900 and 2 HDS 7700) using Veritas files systems, and several Oracle 8 databases. The platform provides for separated development, test, and production environments. It is designed to support about 10'000 end users (mostly for reporting purposes).

A novel data staging process was implemented based on Informatica's PowerCenter. Migration of the existing warehouses (except of one) to the new platform is completed and the first business area warehouse is operational. The other business area warehouses will be implemented during the rest of this year.

The following experiences have been made (this list is not complete):

- The platform proved to be very powerful and reliable. The staging process is on average 4 times faster than on the old platforms, and the database performance is satisfactory[1].

[1] The remaining performance problems are mainly caused by poor physical designs of some data marts or poor application designs.

An efficient load process becomes increasingly important, since new business requirements demand for a daily load of data while the batch window decreases at the same time due to on-line banking (which does not care about "office hours")[2].

However, one should not spend too much time on evaluating platforms. Other high-end Unix servers and enterprise storage systems would do as well. Platform selection should mainly depend on the skills being available in a company, and whether the tools business users are most interested in run on this platform.

- The new architecture (based on business area warehouses) allows for a very quick implementation of new data marts (if the required data are already available in business area warehouses and the project team is familiar with the environment). Now a completely new data mart can be implemented within 3 months (from design to production).
- Interface management between the warehouse platform and the feeder systems is a very important issue. In the past, the warehouse projects were responsible for implementing the data extraction programs (usually in PL/I) in order to generate the feeder files. This approach (pull principle) leads to various problems:

  When a feeder system is changed, the extraction programs have to be adapted, too. However, if these changes have to be implemented by two different project teams, the release schedules of both projects must be synchronized. In a large-scale environment with hundreds of feeder systems and dozens of warehouses, such a synchronization is not easy.

  Moreover, if the warehouse platform and the platform of the feeder system are different, the warehouse project team has to maintain additional skills which often results in a waste of resources.

  Therefore, we switched over to a *push principle*. The warehousing and the feeder system projects sign *service level agreements* that define which data must be delivered in which format at which time to the warehouse platform. If a feeder system is changed, the same project team will also modify the extraction program.

  The push principle also simplifies a smooth transition of feeder systems to new banking paradigms. The distinction between on-line and batch processing (where data extraction is done as part of the end of day processing) will soon no longer exist (7 x 24 hours on-line processing). When the push principle is used, warehouse systems need not care whether the feeder files have been generated by a data extraction program during the end of day processing, or have been generated continuously using, e.g., message-oriented middleware.
- Reengineering existing systems “on-the-fly” is a tedious process which can only be successfully completed if this process is strongly supported by the top management.

[2] The next release of the reference architecture will include an operational data store (ODS) as part of the staging area to support nearly real-time warehousing. An ODS is also required when the batch window is completely closed (7 x 24 hours operation of banking applications). ODS data will be transferred to the warehouse platform using MQ Series middleware.

A business steering committee must be established that decides strategic issues of the whole warehousing platform and keeps all business projects on track. Since the re-architecture program is a long-term project that binds lots of resources, it is always tempting for smaller projects to opt for quick win solutions that (once again) do not comply with the defined standards.

A general problem is that it is easy to find business owners for data marts (each data mart serves a particular business need so that business people have a natural interest in these systems), but much more difficult to find business owners for business area warehouses (the common stock of data).

However, clearly defined responsibilities (including data security decisions and data quality measures) are even more important for business area warehouses than for data marts, since these databases are the common foundation of all data marts.

- The organization of data warehouse developers needs to be adapted in accordance with the new architecture. It is no longer meaningful that project teams care for the whole processing chain of a warehouse application (end-to-end). A separation between data staging/development of business area warehouses and development of data marts is required. The "classical" project teams then develop solutions for given business requests by defining data marts and developing reports, etc. A close cooperation of both groups is of course required since new business projects may need additional data that are not yet available in any business area warehouse.

In the following areas we did not make the expected progress[3]:

- Metadata management is still a big problem. On the one hand, metadata management tools are still immature, and on the other hand the warehousing tools (in particular staging and reporting tools) have poor interfaces to upload metadata from these tools into the global metadata repository.

  However, an automated management of metadata (bidirectional) is essential for end user acceptance. If IT users (developers of staging processes, warehouses and data marts) and power users (report developers) have to maintain metadata explicitly, the metadata repository and the warehousing system will diverge over time, rendering metadata management useless.

  Even if metadata can be captured automatically, user-friendly interfaces are required to present these data to the end users. Interfaces need to include search engines where casual users can look for warehousing data based on ontologies. Reports must be classified automatically in accordance with these (customizable) ontologies.

  A fundamental problem is data lineage at data item level where script-based transformation rules (staging tool) and reporting *programs* make it very questionable whether it is possible to achieve any progress at all.

  At least a case study would be helpful that describes examples of successful metadata management implementations, highlights success/failure factors (from

[3] In these areas additional (applied) research seems to be meaningful.

the technological to the business point of view) and gives a deeper understanding of realistic expectations.

- Systems management (in particular performance management) is also difficult in our warehousing environment. It is almost impossible to guarantee a defined minimum of system resources (in particular CPU) for critical applications if database instances are shared. So far performance management is only possible either at operating system level (Sun resource manager) or database level (Oracle resource manager), since both are not compatible with each other.

  This does certainly not require basic research, since the required technologies are known from the mainframe world for ages. In this respect, even high-end Unix servers are still in their infancy (not to mention NT).

  Another issue is performance monitoring in n-tier environments. It is easy to check whether the warehousing components are up and running (including network connections, web servers, application servers and database servers). However, it is hard to find out where the bottleneck actually is if end users experience a poor end-to-end performance.

- Today's warehousing tools each have their own security manager. Implementing security – in particular access control – over the whole warehousing chain such that data are consistently protected irrespective of their actual location and representation is almost impossible. In principle, a comprehensive end-to-end security model would be required that allows for an integrated privilege management over heterogeneous tools (including ad-hoc reporting). Otherwise, security management is not only an error-prone task, but also hardly understandable for auditors.

  So far, the problem has only partially been solved with integrated user administration tools like Control-SA from BMC, SAM from Systor/Schumann or Tivoli (User Administration and Security Management) from Tivoli/IBM.

  Another open issue is the provision of a *complete* audit trail over all warehousing tools. In environments with high security requirements like ours, auditors sometimes need to know which end user has read which data. Therefore, all tools must provide for the required log data and these log data must be integrated into a global log database.

  For integrating audit data, warehouse technologies can be applied (e.g. an audit warehouse), but today most tools do not make the required audit data available. Even the database system does not deliver these data, since permanently running Oracle in trace mode is not really an option.

  What we need is a publish/subscribe mechanism for all tools so that warehouse administrators can collect all data needed according to *their* requirements without introducing too much overhead on the warehouse platform.

- Performance can always be improved, in particular for on-line analytical processing (OLAP). Currently, OLAP based on very high data volumes (terabyte range) is only possible using relational OLAP (ROLAP). Since response times of ROLAP tools are rather high, we would prefer to use a multidimensional or hybrid OLAP system. Unfortunately, these tools are not yet able to handle such high data volumes, in particular if the underlying data change every day (creating a huge data cube can take several days).

- Another open issue is archiving in the context of evolving data models (schemas). Today it is easy to generate backups (enterprise storage systems even allow for serverless backups, and storage area networks can easily handle the data volume), but systems evolve over time. In most cases, it is not feasible to migrate all existing backups when new system releases are introduced.
- One of the major problems is data quality management. More than 500 feeder systems deliver data to our warehouse platform. The data quality differs from system to system considerably.

  Warehouse engineers can only measure the quality of data to some degree (using heuristics), but they cannot improve the quality. Data quality management has to be a business-driven process where the correction of errors is initiated by the business users. We have made the experience that it is very hard to convince business users in the early phases of a warehousing project to spend some time on data quality management. However, the more the warehouse platform is being used the higher gets the interest of business users to fix errors that they have found in their reports. A data quality management process must be in place which ensures that errors that have been found are really fixed.

  Unfortunately, business users often argue that these corrections should be made in the data warehouse (where it is often much easier than in the source systems to access these data, and the interfaces are much more user-friendly), since it "should be no problem" to feed these corrections back into the source system. We strongly argue *not* to apply this "solution". The resulting system will tend to be unmanageable, since cyclic dependencies between systems at the data instance level are never fully understood in large-scale environments.
- Finally, we envision that our warehouses will have to manage semistructured and multimedia data. There are plenty of examples of such data in e-commerce applications, e.g. contracts, insurance policies, geographical data for mortgages, and for risk management in the insurance business.

  Today's analysis tools (reporting, OLAP and data mining) are not ready yet to cope with such kinds of data.

Summing up, the basic technologies for implementing complex data warehousing and analysis systems are available. Though tool providers (database vendors and analysis tool vendors) may benefit from new and improved algorithms (in particular to improve the performance of their tools), only evolutionary improvements are to be expected.

According to our experiences, warehouse projects usually do not fail for technological reasons. They fail due to organizational problems, insufficient involvement of business users, missing top management attention or poor skills management.