CORBA / UML Success Story
Zuercher Kantonalbank
Banking/Finance

Contact: Wilfried Heinickel, KLEIN+STEKL, GmbH
heinickel@klst.com
+49 711 9672 152

Tanja Feldmann, KLEIN+STEKL, GmbH
feldmann@klst.com
+49 711 9672 156

Technology and Tools used*:

Platforms	Windows NT	AIX	OS/390
Software	ZKB SDV Tool (see below) IBM MQSeries WorkFlow IONA Orbix (CORBA)	IONA Orbix (CORBA) ZKB MPF (see below) IBM MQSeries Oracle	IBM MQSeries WorkFlow ZKB MPF (see below) IBM MQSeries IBM DB2 IBM IMS
Hardware/ servers	Intel	IBM RS/6000	IBM Mainframe
Languages	Sun Java XML	C++	C++
Tools	RationalRose (UML)		

Description: The German software company KLEIN+STEKL GmbH provided a solution to the Operations Department of Zuercher Kantonalbank (ZKB), one of the three leading banks in Switzerland. The system was developed by a team of four application engineers at ZKB in addition to a team of seven software engineers at KLEIN+STEKL who developed the underlying SDV-Tool. The Application/Project's name was ASD, which stands for **A**llgemeine **S**tamm**D**aten (General Master Data). The system serves about 60 NT-clients. The users are dealing with about 330 different data classes and about 220,000 objects saved in the database. The SDV Tool consists of some 109,000 Java statements in 1,335 classes.

Development of the new system began in July 1999, successfully passed all acceptance tests in March 2002 and was launched at the end of that month. The activities also included an extensive clearing up of banking data. Due to its generic design, the system provides a perfect basis for future applications---it is expected that further applications will make use of this system, making integration easier with a short time-to-market.

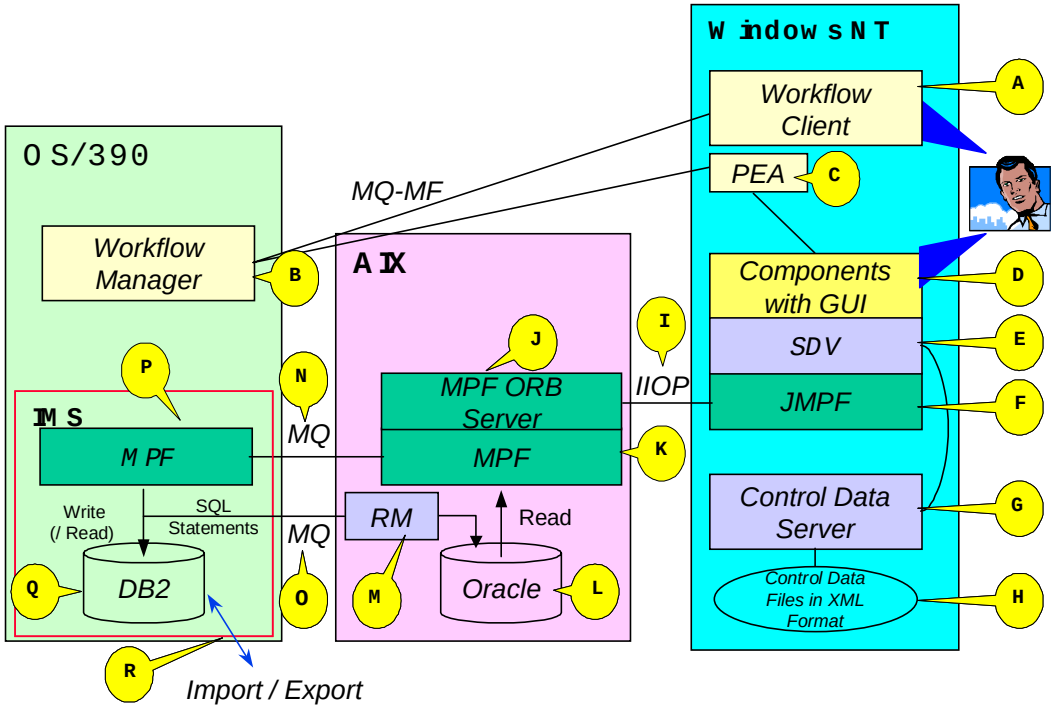
Problem:
An IMS **(I**nformation **M**anagement **S**ystem**)** database keeps parameters and reference data for virtually all mainframe-based ZKB banking applications. The corresponding tables contain information on the internal organization, on ledger structures, authorizations for user applications, postal codes, stock exchanges, currencies, output-control for documents, taxes etc.

These large amounts of data have to be maintained by the users, who have to ensure that the required changes are carried out error-free and on time. Countless consistency rules across the tables have to be followed with very little software support. This caused various problems, ranging from outdated information over redundancies to wrong associations between data, which further led to occasional malfunctioning of the bank's core applications. ZKB needed a new system in place to prevent any future problems.

Solution:

The solution is designed not only to solve the actual problem, but also to provide far more functionality. A tool has been built which interprets metadata and forms a generic GUI and a generic data management application without requiring additional programming: the so-called SDV Tool. The SDV Tool is an in-house product.
The NT client consists of some 109,000 Java statements in 1,335 classes, which form the SDV Tool. The SDV Tool automatically interprets metadata, which is kept in the same database as the reference data, to build a ready-to-use application for data management. The GUI is built at runtime by interpreting XML control data. These XML data also control the use of consistency rules, complex logical business objects and the use of java-encoded parts.

To be able to meet special user requirements, its generic behavior can and actually is partly overruled by control data (XML) and dynamically invoked java classes. The data are now kept in a bitemporal database, i.e. a database which supports both valid and transaction time (cf. The TSQL2 Temporal Query Language [1]). Users can change data for a future date with full view of the database's state at that particular date; moreover consistency checks are automatically carried out for the whole period of time concerned by the changes. For investigation purposes a user can view online the state of the database at a certain time in the past simply by entering the corresponding date.



Abbreviations:

MPF:	Metadata Platform (in-house product)
JMPF:	Java Wrapper for MPF
ORB:	Object Request Broker
RM:	Replication Manager
PEA:	Program Execution Agent
SDV:	Tool for Master Data Management (in-house product)

Explanations:

A The user starts a process from the workflow client

B The workflow manager uses an agent (C) to start a workflow component (D).

C The PEA (Program Execution Agent) starts a workflow component.

D Any workflow component requests control data from SDV as soon as the control data server is started.

E The SDV framework and the SDV‘s control of the sequence of actions interpret metadata and build at runtime the required GUI.

F JMPF (Java Wrapper MPF) requests an MPF server from the ORBIX demon (communication via CORBA).

G Control data server is started by means of autostart.

H The control data files (XML) are part of the local installation.

J When a workflow component terminates, the corresponding MPF ORB server is terminated via ORBIX.

K The MPF kernel in C++ emulates a bitemporal database on a relational database (Oracle).

L The central Oracle database keeps the replicated data (cache function).

M The replication managers (one for each decentral server) wait for requests from the central server. They receive DB2 SQL statements (inserts and updates) in XML, convert them into Oracle SQL statements and execute them.

N The request/result queue transmits requests and results between AIX and IMS.
O The queue transmits XML replication messages to the Replication Manager.
P The MPF server waits for requests from the decentral server and executes them. It acknowledges the request by either providing data according to a read-request, saving data by transmitting insert/update statements via MQ to the replication manager (asynchronous, no acknowledgement) or by providing an error message.
Q The master database contains all versions of business objects.
R Data can be exchanged with other (external) databases.

The whole system was designed with UML (RationalRose). As UML is easy to understand, use cases, data models and interactions can be discussed even with the end users, thus ensuring a practical solution. The connection between NT clients and AIX server is completely based on CORBA (IONA Orbix). Connections, server start/stop and transfer are completely implemented using Orbix and work perfectly.

“We chose UML for the project to make a design which is clear and precise. UML is the best way to visualize and manage integrated use cases, interaction diagrams and class models. Moreover using UML modeling tools, it is possible to generate code from the model or update the model whenever the code is further elaborated or supplemented: At the first stage in the development we have used round trip engineering to build a model-based prototype of the SDV Tool,” said Wilfried Heinickel, Systems Architect, KLEIN+STEKL.

He added, “We chose CORBA because we wanted to implement a high performance interface of interoperable systems (C++ and Java) on different platforms (AIX and Windows) using well-established standard technology. The configuration and activation of the connection is completely independent of the developed SDV Tool.”

Benefits:
The users benefit from being supported in complex operations, providing faster transactions with less errors. ZKB benefits from trouble-free operation of the banking applications as well as from being able to backtrack any changes in the database. The quality of parameters and reference data is dramatically increased by comprehensive consistency rules as well as by double-checking data before they become effective. This reduces the costs for investigating data-related malfunctions of applications, considerably (close to zero).

The data is now kept in a fully bi-temporal database. Users can change data for a future date with full view of the database's state at that particular date; moreover consistency checks are carried out for the whole period of time concerned by the changes.

For investigation purposes a user can view online the current state of the database at a certain time in the past simply by entering the corresponding date. The data will be provided to the banking applications just in time on the day they become valid. Moreover with this system a generic basis has been provided which makes it easy to integrate further applications with little effort and a short time-to-market.

Considering all bank processes affected by the project, it is expected that after having accomplished all extensions, the maintenance costs will be reduced by 2/3.

In conclusion, Andrea Haussener, ASD system owner at ZKB, had this to say about KLEIN+STEKL's solution, "By means of workflow-controlled standardized procedures and consistency checks, the system helps to ensure an excellent data quality in the whole process of data management. Moreover the bi-temporal database is always available to inform about what a certain data record and its relations looked like, or will look like at a certain point of time (past, present, future). The transparency of the system ensures a high-quality and future-oriented maintenance of master data."

Heinickel stated, "When using object technology, the terminology and objects used in models, use cases and so on are closer to the real world than in other technologies. This means that the subject is easier to understand and easier to discuss with the end users. Moreover experiences with other projects within the ZKB have shown that projects using object technology are a lot more flexible, thus reducing maintenance costs considerably."

Further explanations of the software products used:

- ZKB MPF:
  MPF is an in-house product. It represents the "Metadata Platform" of the ZKB, which implements a fully bitemporal database on top of Oracle and DB2 and is controlled by metadata.
- IBM MQSeries WorkFlow:
  IBM MQSeries WorkFlow supports the business process. The Workflow manager first assigns the order to enter or change data to an authorized user. After the order has been executed, it assigns the order to another user for double-checking the resulting changes before they become effective.

Literature

[1] Richard Thomas Snodgrass (ed.), 1995: "The TSQL2 Temporal Query Language" Kluwer Academic Publishers