9

Migration of legacy applications to a CORBA platform: a case study*

D. Konstantas
University of Geneva
Centre Universitaire d'Informatique
24 rue General Dufour
CH-1211 Geneva 4, Switzerland
e-mail : dimitri@cui.unige.ch
tel: +41 (22) 705.7664, fax:+41 (22) 705.7780

Abstract

An important problem that many large organizations face today concerns the support and evolution of their large (legacy) information systems. In this paper we present an approach for migrating large and interdependent legacy information systems to CORBA based platforms. Our approach allows continuous availability of the information systems, minimal manpower for the implementation and preservation of the autonomy of the independent information systems. The information systems of ABB AG serve as case study for the presentation of our approach.

Keywords

Legacy Systems, Interoperability, Interface adaption

1 INTRODUCTION

Typically the information systems of big organizations are large (10s of millions lines of code), old (more than 10 years old) and have evolved in an unstructured manner with extensions added in different languages and with minimal, if any, documentation available. In addition these systems are mission critical and should remain functional at all times. These information systems define what we today call **legacy information systems**. The problems that these systems pose to their host organizations are numerous and important:

- First, the systems cannot evolve to provide new functionalities required by the organization.
- Second, it is extremely difficult, if not impossible, to port them to new hardware platforms. This results in high costs for the organization due to the need to maintain very old and obsolete hardware, and because of the reduced productivity due to the low speed of the old hardware.

*This project was supported by the Swiss Federal Government by the SPP-ICS 1992-1995 project "CHASSIS" (project number 5003-34355/2), and the FNRS project "Object Integration" (Project Number 20-40592.94).

A. Schill et al. (eds.), *Distributed Platforms*

- Third, the maintenance of the information system (tracing failures and correcting them, training of new system administrators etc.) becomes increasingly costly and time consuming due to the lack of documentation of the overall structure of the system and the added-up extensions. It is not atypical for a piece of mission critical software to be working correctly without anyone knowing why and how it is working.

Although in the past the strategy of large organizations was to maintain the legacy information systems, nowadays the strategy is to replace them with new modern and extensible systems. However it is not clear to the organizations' management how this can be done with minimal cost and without stopping or compromising services to the clients. A typical method for the replacement of relatively integrated information systems (that is, composed of a single main information system) is to install the new information system and run it in parallel with the old one for a sufficiently long time period so that the new one can be extensively tested. However this is not possible for information systems that are based on independent component-information systems running on heterogeneous hardware platforms. The reasons are that first the interfaces between the component-information systems cannot be modified and second the interactions between them are in many cases not known (that is, undocumented).

An example of this type of information systems is found at Asea Brown Boveri AG (ABB) in Baden, Switzerland. The ABB information systems include CAD systems, application-specific calculation systems (e.g., mechanical stress, heat transfer, energy efficiency etc.), composition, configuration and parameterization of customer-specific systems from standard parts and according to customer requirements (e.g. of a process control system), document preparation systems, etc. The functionality of these systems is highly interdependent and in addition, the various organizational aspects, such as collaboration of many teams (often located in different regions), optimization and integration of the engineering process (from the offer preparation to the final acceptance test), and project management and control, play an increasingly important role in their operation.

This paper outlines a solution proposed in the framework of the CHASSIS project (Configurable, Heterogeneous, And Safe, Secure Information Systems) (Niertstrasz and al., 1993), for the migration of the ABB information systems on a Common Object Request Broker Architecture (CORBA) based platform. This is achieved via an **interoperability support layer** that will:

- allow the interoperation between applications
- provide an enhanced security model and mechanisms
- support the integration of the different data repositories under a federated database
- permit an easier re-configuration of the independent information systems and of the complete ABB design and development environment (which can finally lead to the migration of the different ABB legacy information systems).

CHASSIS is a Swiss federal research project of the Swiss Priority Programme (SPP-ICS) that aims to provide a platform for the security- and reliability-oriented systematic design and construction of heterogeneous information systems from individual existing and newly developed application software components and database systems. Partners in CHASSIS are the University of Zürich, the University of Geneva, and the Asea Brown Boveri Research Centre at Baden. The case study for CHASSIS is from the domain of engineering information systems for electrical engineering (the speciality of ABB).

2 INTEROPERABILITY AND SECURITY PROBLEMS

At ABB the design and development process of a product involves several heterogeneous information systems (Rohrer, 1993) including CAD–CAM systems and very large FORTRAN programs. These systems were incrementally built, modified and merged over a period of more than twenty years. This slow evolution occurred without any real global design or master plan. Modifications and extensions of the software followed the changing needs and requirements of each department in the company. To keep up with this evolution, ad hoc data translation programs had to be developed. The net results of this growth are twofold: first, duplication of functionality and data among applications and second, serious security problems behind every data transfer, be it to ABB customers, associates, or even from one department to the next. Although ABB acknowledges these problems, solutions and strategies to resolve them are in short supply.

2.1 Requirements for the Interoperability Support Layer

Our target is to design an interoperability support layer incorporating a database federation and an enhanced security model, and implement a prototype demonstrating our ideas. However, in order to be relevant to ABB, some important industrial requirements must be taken into account.

- First and foremost comes the availability of the system. At ABB, information systems are in continuous use by the design and development departments. Clearly, all software systems must remain operational at all times. Unavailability of any system should not be longer than a few hours, in the worst case.
- Secondly, the interoperability support software should not require excessive manpower to be implemented. (The word of the day is recession!).
- Thirdly, the different information systems must retain their autonomy. Different departments are responsible for their maintenance, and development often takes place at different sites.
- Finally, the interoperability support should, if possible, be compatible with emerging standards, like the Common Object Request Broker Architecture (CORBA) (Object Management Group and X Open, 1991) of the Object Management Group (OMG).

Based on the above requirements the interoperability support design must anticipate an incremental development and part-by-part incorporation of the existing applications and their components, in the interoperability platform (Brodie and Stonebreaker 1993). The basic elements of each application will need to be identified and incorporated into the interoperability platform. To incorporate these elements three alternatives should be considered: rewrite the code entirely, reuse certain portions of the code and rewrite others or leave the existing software intact encapsulating it in some kind of a wrapper.

During the development process, software layers (**gateways** (Brodie, 1992)) will be used as bridges between old and new software components. The goal is that every software component can be independently tested and, in case a problem is found, the original software can be reinstalled to keep the system operational. Often the dependencies between components are not obvious and can go unnoticed. The ability to compare the old component with the new and to reinstall the old component is therefore a very important requirement of the overall design.

In order to allow maximum openness and flexibility, conform with emerging standards, and bring to ABB a robust, industrial–strength implementation which can be extended to meet future

requirements, the CORBA can be used to provide the base of the interoperability support. However, the CORBA was not designed to support interoperability between existing applications: it assumes new, CORBA–conforming, applications. In the CORBA a client can access a service using either the generated Interface Definition Language (IDL) stubs' interface or the Dynamic Invocation Interface (DII). These interfaces use CORBA specific constructs (that is, syntax, operation call arguments, exception handling, data and object types etc.) which are not known to applications that were not designed and implemented for CORBA integration. Thus, in order to allow **existing** applications to access the CORBA interfaces, it is necessary to introduce to the CORBA an **interface adaption layer** that will allow the systematic and consistent adaption of non-CORBA interfaces to CORBA interfaces. This interface adaption layer can be based on the ideas of object–oriented interoperability (Konstantas, 1993b) (Konstantas, 1995) found in the Cell framework (Konstantas, 1993a).

The interoperability support can be implemented from scratch, as we did in our prototype implementation of the Cell framework (Konstantas, 1993a), or by extending an existing system like an Object Request Broker (ORB) implementation. In the following sections we will simply refer to the **Interoperability Support Platform** (ISP) without mentioning any specific implementation technology.

2.2 Security and federated databases

Two key aspects of the CHASSIS project are the integration of existing data repositories used by the information system into a new federated database and the addition of more powerful security mechanisms than these of the existing information system.

An outline of how federated databases and the Cell framework can be merged to provide a solution to the heterogeneous ABB data repositories is given in (Joschenr and Konstantas, 1993) The basic architecture is depicted in Figure 1. The original information system has to be separated into basic elements, applications and data repositories, which are both encapsulated as cells. The membranes of each cell take care of catching incoming and outgoing calls and dispatching them

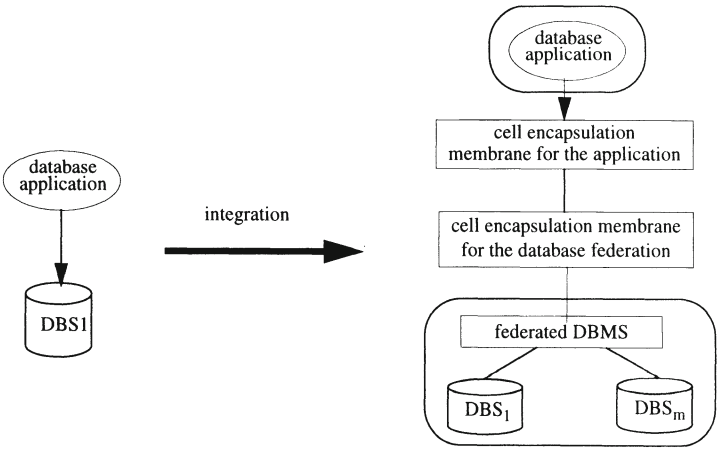

Figure 1 Integration of an existing database application with a database federation.

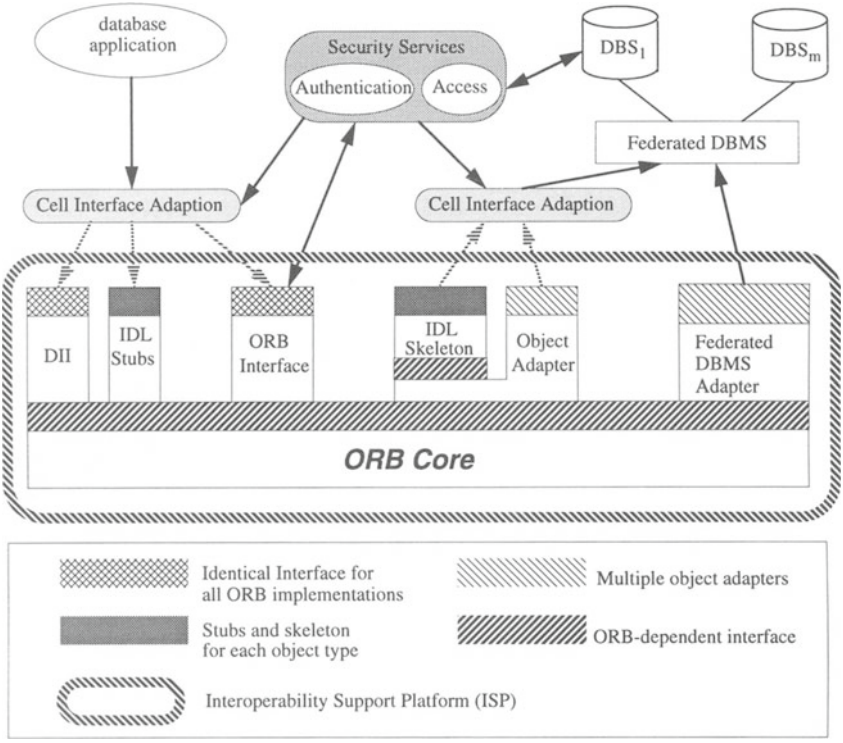

Figure 2 CHASSIS architecture using the CORBA.

accordingly. The security model (Argos capabilities) (Jonsher and Dittrich, 1993)(Jonscher and Dittrich, 1994) is incorporated in the membrane and enforced by the federated database.

The different data repositories of the ABB environment can be integrated into a federated database and linked together via a federated database object adaptor of the ORB similar to the one proposed in the ASSET project (ASSET ESPRIT Project 7703, 1994). This way the federated database can be accessed by newly developed applications through the ORB. Furthermore, the security support can be incorporated into the CORBA as described in the OMG's White Paper on Security (Object Management Group, 1993).

An outline of the CHASSIS architecture using the CORBA to implement the concepts of the Cell framework is given in Figure 2. The main extension to the CORBA is the addition of cell interface adaption layers between the application and the ORB, and between the federated database and the ORB. These interface adaption layers translate the data-repository access requests of the applications to CORBA requests either using Interface Definition Language (IDL) stubs or the Dynamic Invocation Interface (DII). A second cell interface adaption layer will be used to link the ORB with the federated database, until a federated database adapter is implemented. The security services will cooperate with the federated database -they might even be part of it- and answer to requests from the cell interface adapters as well as from the ORB.

3 CASE STUDY: THE TURBO-GENERATOR PRODUCTION PROCESS

From the various information systems of ABB we have chosen one application domain for a case study and namely the turbo-generator production process. The main applications used in this part are *MIA*, *CATIA*, *AINFO*, *PROFORe* and a mechanical layout program; they are described in greater detail in (Rohrer, 1993). These five applications are used by the sales, order engineering, development and production departments of ABB. Figure 3 shows usage patterns of these applications. Three main issues in the operation of these applications have been identified:

1. flow of information between the applications,
2. consistency and confidentiality of the data and
3. interoperation of these applications.

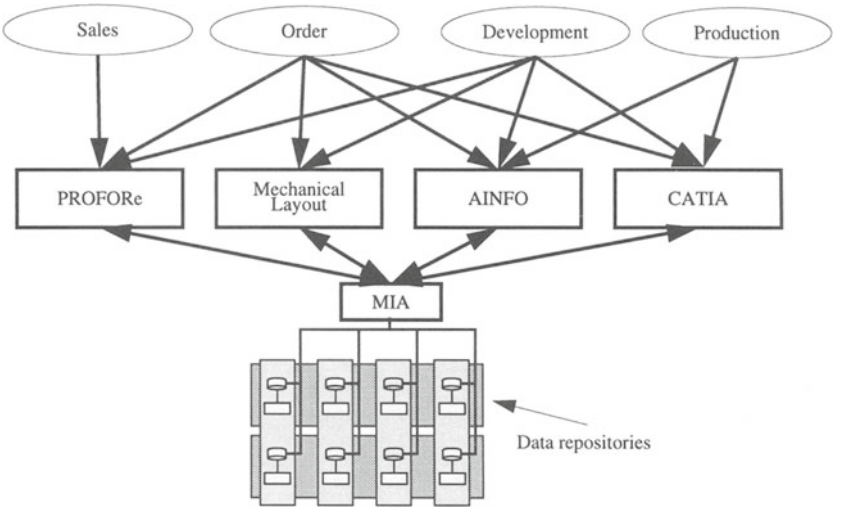

Figure 3 Present architecture of the turbo generator production process.

Information exchange is achieved mainly via data files. One application generates some data which are later passed to another application. However, due to the heterogeneity of the applications and their fragmentation, conversion programs have to translate the data from output file formats to input file formats. These conversions, as well as information retrieval from different sites or putting them into archives, represent the primary responsibility of the MIA (Make Information Available) application.

Two outstanding problems in this process are data consistency and data confidentiality and protection. Because of the duplication of information in numerous formats over different systems and sites, preservation of consistency is very difficult. In addition, security of the data is controlled in an ad hoc way requiring sometimes even manual modifications of the files that need to be passed to the clients. Furthermore, unauthorized modifications of data have to be prevented.

The interoperation of the applications in their present configuration is practically non-existent. Each application runs independently of every other, producing a bunch of data for the next application in the chain, which in many cases need to be translated to a different format. Thus, the

overall operation of the turbo-generator production process can be characterized as a batch process. The consequences of the non-interoperation of the applications are numerous. Among the most serious ones are delays in the production process, repetition of complex calculations, and extensive duplication of code and data.

To be noted that the above issues are **not** specific to the applications described but exist to some extent in all of the ABB information systems.

3.1 Overview of the proposed interoperability support

The main element to support interoperability of the five applications is the introduction of a common database federation and the CHASSIS security mechanisms (Jonsher and Dittrich, 1993). The first step in the ISP design anticipates only minor changes of existing applications. This step consists of introducing the federated database and the associated cell interoperability layer in the system (Figure 4).

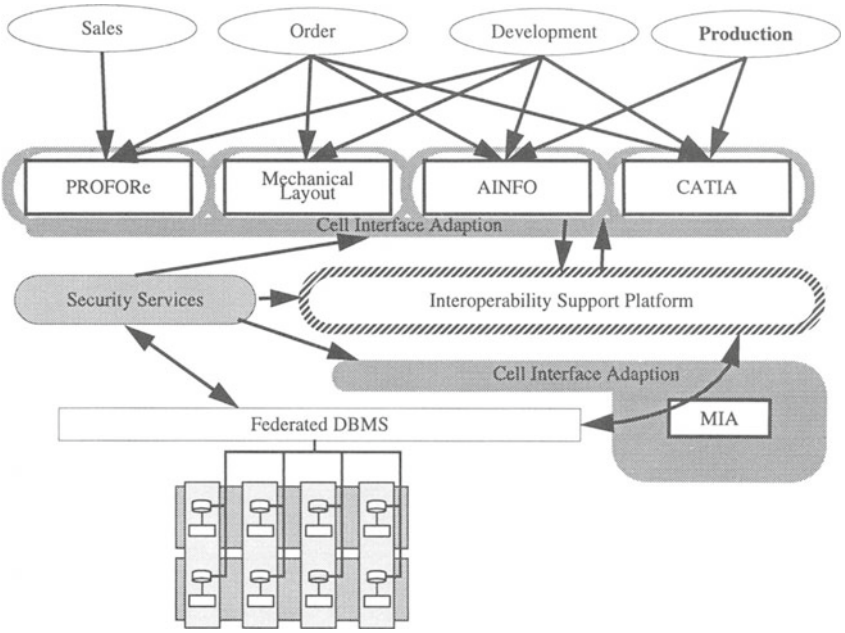

Figure 4 Introduction of the federated DBMS and cell interoperability in the production process of turbo-generators

The Cell Interface Adaption Layer (CIAL) will encapsulate the different applications and will translate their requests for data access into equivalent requests to the federated DBMS. In addition, a thin layer of the CIAL, between the user and the application, will collect and handle information needed for the security support. Next, the request along with the security information will be forwarded via the ISP to the federated DBMS. The federated DBMS will then obtain the

needed data from the MIA data repositories. Whether requests will be forwarded from the FDBMS to MIA or directly to the controlled data repositories (after consultation of MIA) remains to be determined.

Once the above architecture is in place, the different applications can be incrementally modified within a consistent overall architecture. In addition, new applications will be designed and implemented on top of the ISP (Figure 5).

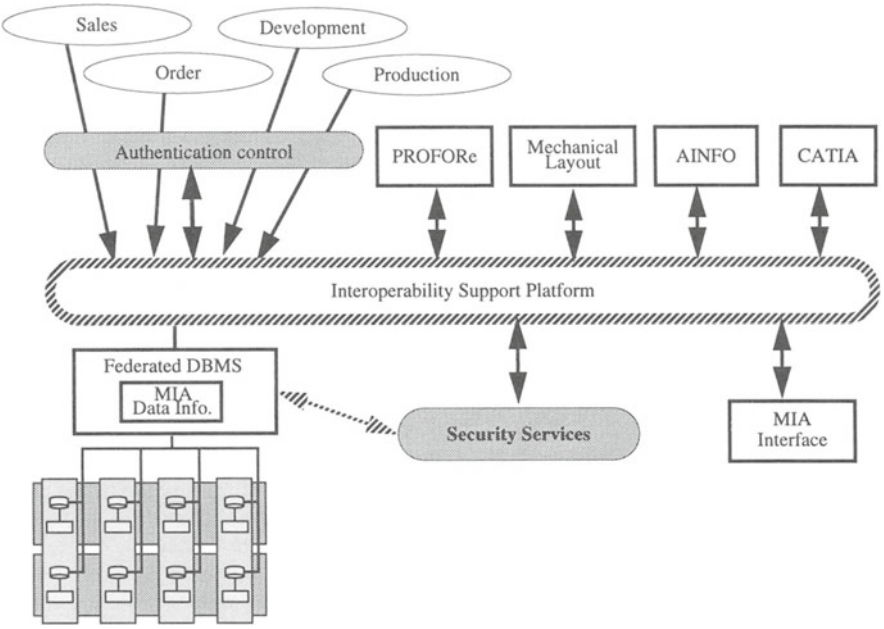

Figure 5 Target architecture of the Production Process of turbo generators

In the final architecture the different applications will have been modified to use directly the ISP's interfaces in accessing not only the database(s) but also services offered by other applications. The CIAL for the applications will thus be suppressed. Similarly, the federated DBMS will be directly accessed by the ISP (either via an object adapter of the CORBA or via a cell interface adapter) and the data repository information services of MIA will have been integrated in the FDBMS. Finally, the security services will provide an implementation of the required security model and will be directly accessed from the ISP and (probably) from the FDBMS.

3.2 PROFORe

To be more precise, let us take a look at the PROFORe application. PROFORe is used to compute the electromagnetic layout of a turbine and is composed of two main modules: GENOB and HT547. Both modules are written in FORTRAN, represent in total more than 100.000 lines of code (Rohrer, 1993) and no complete documentation of their structure and functionality exists.

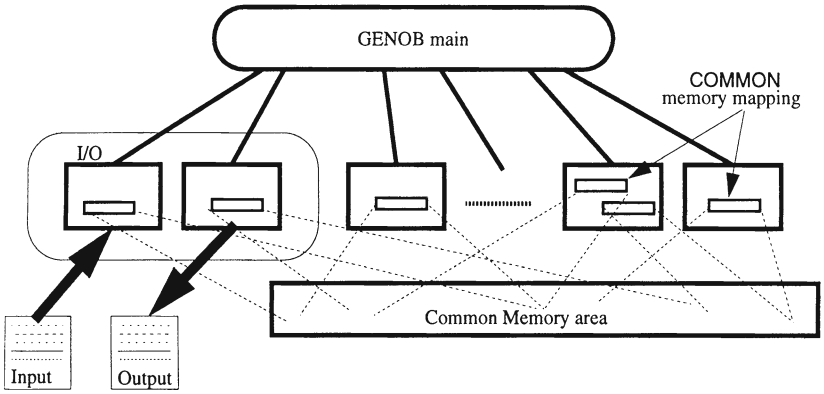

Figure 6 PROFORe: GENOB architecture.

GENOB is a computational application composed of different submodules that communicate via FORTRAN COMMON variables (Figure 6*). The execution of GENOB is controlled by a main program that calls the sub-modules in some order. Each sub-module's program performs its calculations using a set of the COMMON variables and placing its results into another or the same set of COMMON variables. It has to be noted that no documentation exists describing which COMMON variables are used by each sub-module and, to make things worse, each submodule interprets the COMMON memory area in its own way. For example, one module might map a COMMON memory area into two integers and one array of 50 integers, while another one might map it into a 52 element array. One of the sub-modules is the I/O sub-module. Its task is to read data from a file and place them in the COMMON memory area and at the end of the GENOB processing to write the data from another or the same COMMON memory area into a file.

HT547 is another suite of programs which, in contrast to GENOB, exchange information via FORTRAN streams (files). The HT547 programs are clustered in different modules according to their functionality (for example HT5476, GMO33 etc.)(Figure 7). Each program is reading data

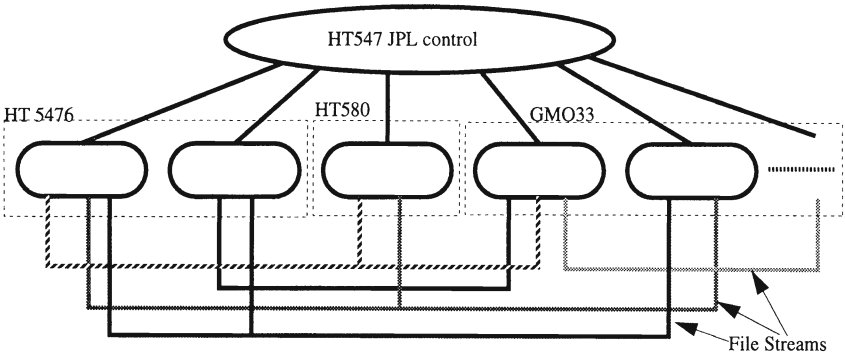

Figure 7 PROFORe: HT547 architecture.

*Both Figure 6 and Figure 7 give only an *outline* of the architecture of the PROFORe modules.

from a set of specified streams and writes its output into another set of streams. The streams used by each program are documented, in contrast to GENOB's COMMON memory areas. The invocation of the programs is controlled by a JCL (*Job Control Language*) program which sequentially invokes the programs attaching them to the needed streams. Two of the streams are used for initial data input and final data output, one is used for error reporting and the rest for passing intermediate results between the programs.

An example of code duplication can be found in PROFORe. Within GENOB and HT547 there exist computational modules for fluid dynamics: one for liquids and one for gases. Clearly this is a case of code duplication since the same module can serve for the calculation of both cases with simple changes of parameter values.

Porting GENOB to an interoperability platform

In order to port GENOB onto an interoperability platform we have first to decompose it in sub-modules. However, since the different sub-modules use the common memory blocks in an undocumented way for their inter-communication, we must isolate these memory accesses. The way to do that is to introduce a software layer that will access **all** the common memory area and store it into the database and vice-versa (Figure 8). The reason to treat the complete common memory area is because we simply do not know what parts are used by each submodule.

During execution (note that GENOB executes sequentially) the invocation of each module will be preceded by the invocation of a DBMS-to-common memory data transfer module and followed by a common memory-to-DBMS data transfer module. This way all data will be stored in the database. Once the common memory to/from DBMS software has been installed, modification of the modules can begin. Each module will be rewritten or modified so that it no

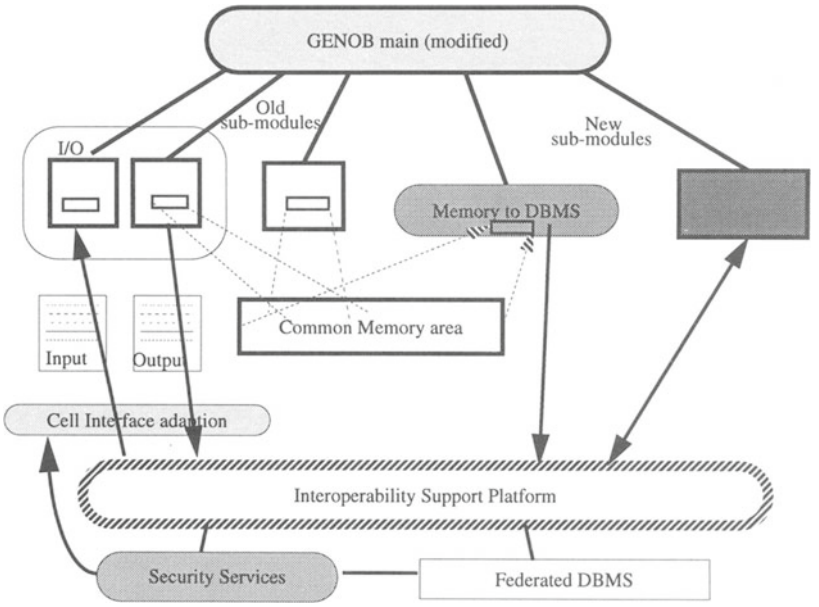

Figure 8 GENOB: Interoperability support based architecture (intermediate state).

longer uses the common memory area but directly accesses the FDBMS and possibly receiving intermediate results as operation call parameters. In this re-writing process the main computational parts of the code will be reused with or without modifications. In order to minimize the invocation of the memory to/from DBMS software, the re-implementation of the sub-modules can follow their invocation order thus creating a fire-wall between the new and the old implementations. It should be noted that the functionality of the re-developed modules will be accessible via the Interoperability Support platform from other (external) applications.

Porting HT547 to an interoperability platform

Porting HT547 to the interoperability platform has the same requirements as GENOB. The main difference however is that HT547 uses I/O streams for inter-module communication instead of common memory areas. This simplifies the problem since each sub-module has well defined I/O requirements. What we need to provide is a module that stores the data of the different file streams in the database and creates, on request, I/O streams from the data of the database (Figure 9).

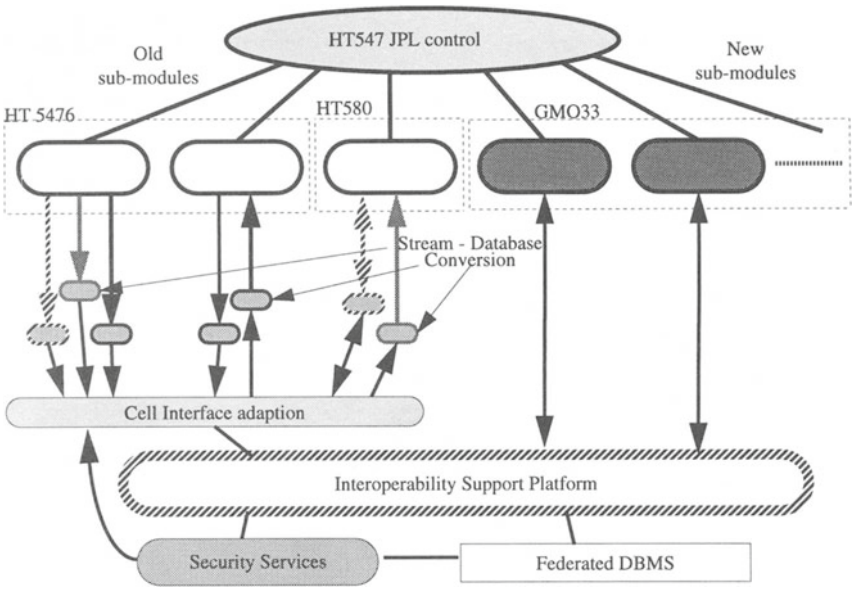

Figure 9 HT547: Interoperability support based architecture.

4 CONCLUSIONS AND WORK PLAN

An important problem that many large organizations face today concerns the support and evolution of their large (legacy) information systems. A typical example demonstrating the problems and related issues are the information systems of Asea Brown Boveri AG, Baden, Switzerland. The information systems of ABB were developed in an unstructured way over a period of more than twenty years and today it has become extremely difficult or even impossible

to extend them for satisfying new organization requirements. In the frame of the CHASSIS project (Niertstrasz and al., 1993) we developed an approach for the incremental migration of the ABB legacy information systems on a CORBA based platform that allows continuous system availability of the information systems, minimal manpower for the implementation, and preservation of the autonomy of the involved information systems. This is achieved with the introduction of the an Interface Adaption Layer that allows existing applications to communicate without any changes with CORBA.

Our approach can be applied to any organization that wishes to migrate its legacy information systems to new technologies introducing at the same time advanced security mechanisms and unification of the different databases. The use of the OMG CORBA for the support of the interoperability service makes our approach even more interesting to industrial organizations since it allows a better integration with the latest available technology and ensures the ability of evolution and integration to applications and services that will become available in the future.

The main questions to be answered before undertaking a task like porting the ABB legacy information systems onto a new platform are: how far do we **want** to go, and how far are we **ready** to go. That is, do we just want to preserve the existing and provide support for new applications or do we want to "upgrade" existing applications and port them onto a new environment. The reply to this question cannot be given by the technical personnel of the organization, since any decision will affect a large part of the organization's information systems. However, no matter what the answer will be, our design can be used in both cases. We can very well stop after implementing the gateways, in which case new applications can be developed on top of the interoperability support platform, or we can continue by replacing existing applications step-by-step. In fact this flexibility is one of the major advantages of CHASSIS approach.

It is clear that a full implementation of the interoperability support, security mechanisms, database federation and the migration of the information systems is out of the scope of CHASSIS project. However, a prototype demonstrating our ideas is under implementation. For this prototype we use a typical application and implement the required software. Once we have completed the prototype, we will be able to use the interoperability services and study the problems of a possible reimplementation of the legacy application. This will give us further insight in the problem of application interoperability and will pave the way for a full-scale implementation. Currently, we are implementing the interface adaption layer using the IONA Object Request Broker on top of a SUN Solaris OS. When the implementation is completed we plan to make a detailed performance analysis of the interface adaption layer. To be noted however that first results indicate that the added performance overhead is less than 5% of the overhead introduced by the ORB. (We measure with ORB and fine tuned custom wrapper, and with ORB and adaption layer). The security mechanisms(Jonscher and Dittrich, 1995) and the database federation organization (Härting and Dittrich, 1992)(Jonscher and Dittrich, 1994) is implemented at the University of Zürich.

5 REFERENCES

ASSET ESPRIT Project 7703 (1994), ASSET Object Oriented Distributed Platform Architecture, Deliverable of ASSET Project (Advance System and Software engineering Enabling Technologies), No. ASSET/WP-C/C1.2/Architecture/Rev 1.0, January 12 1994.

Brodie M. L.(1992), The Promise of Distributed Computing and the Challenge of Legacy Information Systems, in *Proceedings of IFIP TC2/WG2.6 Conference on Semantics of Interoperable Database Systems*, p. Elsevier North Holland, Lorne, Australia, November 1992.

Brodie M. L. and Stonebraker M. (1993), *DARWIN: On the Incremental Migration of Legacy Information Systems,* DOM Technical Report, TR-0222-10-92-165, GTE Laboratories Inc., March 1993.

Härtig M. and Dittrich K. R. (1992), An Object Oriented Integration Framework for Building Heterogeneous Database Systems, in *Proceedings of the IFIP DS-5 Conference on Semantics of Interoperable Database Systems*, p. Elsevier North Holland, Lorne, Australia, November 1992.

Jonscher D. and Dittrich K.R. (1993), *A Formal Security Model based on an Object Oriented Data Model,* Technical Report No 93.41, Institut für Informatik der Universität Zürich, November 1993

Jonscher D. and Konstantas D. (1993), Project Report*: CHASSIS co-operation between IFI and CUI*, November 30, 1993

Jonscher D. and Dittrich K. R. (1994), An Approach For Building Secure Database Federations, in *Proceedings of the 20th VLDB Conference*, Santiago, Chile, Aug. 1994.

Jonscher D. and Dittrich K. R. (1995), Argos - A Configurable Access Control Subsystem Which Can Propagate Authorisations, in *Proceedings of the 9th Annual IFIP WG 11.3 Working Conference on Database Security*, Rensselaerville, NY, Aug. 1995.

Konstantas D. (1993), *Cell: A Framework for a Strongly Distributed Object Based System*, Ph.D. Thesis No. 2598, University of Geneva, May 1993.

Konstantas D. (1993), Object Oriented Interoperability, in *Proceedings of the Seventh European Conference on Object Oriented Programming ECOOP 93*, Kaiserlautern, Germany, July 26-29

Konstantas D. (1995), Interoperation of Object Oriented Applications, chapter 3 in *Object Oriented Software Composition*, (Ed. O. Nierstrasz & D. Tshichritzis), Prenitice Hall 1995

Nierstrasz O. M., Konstantas D., Dittrich K. R. and Jonscher D. (1993), Une Plate-forme pour la Construction de Systémes d'Information Ouverts, in *Proceedings of AFCET 93*, Versailles, France., 8-10 Juin 1993.

Object Management Group and X Open (1991), *The Common Object Request Broker: Architecture and Specification*, Document Number 91.12.1 Revision 1.1.

Object Management Group (1993), *OMG White Paper on Security,* OMG Security Working Group, Draft 0.0 November 24th 1993.

Rohrer A. (1993), *CHASSIS: Requirements for Engineering Information Systems at ABB*, ABB Technical report CHCRC 93-47, 14-12-1993

6 BIOGRAPHY

Dimitri Konstantas holds a degree in Electrical Engineering from the National technical University of Athens, an M.Sc. in Computer Science from the University of Toronto, and a Ph.D. in Computer Science from the University of Geneva. From 1985 to 1987 he worked at FORTH (Heraklion - Crete, Greece) as research assistant, system administrator and project manager for different ESPRIT Projects. From 1987 to 1993 he worked as researcher at the University of Geneva participating in several European and Swiss projects and collaborating with different European companies and organizations. Since 1993 he is Assistant Professor at the University of Geneva. His present interests include baseband and broadband networks, distributed systems and multimedia communication systems.