

1987

A CONCEPTUAL MODEL FOR INTEGRATED AUTONOMOUS PROCESSING: AN INTERNATIONAL BANK'S EXPERIENCE WITH LARGE DATABASES

William F. Frank

Integrated Information Systems Associates

Stuart E. Madnick

Massachusetts Institute of Technology

Y. Richard Wang

University of Arizona

Follow this and additional works at: <http://aisel.aisnet.org/icis1987>

Recommended Citation

Frank, William F.; Madnick, Stuart E.; and Wang, Y. Richard, "A CONCEPTUAL MODEL FOR INTEGRATED AUTONOMOUS PROCESSING: AN INTERNATIONAL BANK'S EXPERIENCE WITH LARGE DATABASES" (1987). *ICIS 1987 Proceedings*. 45.
<http://aisel.aisnet.org/icis1987/45>

This material is brought to you by the International Conference on Information Systems (ICIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in ICIS 1987 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

A CONCEPTUAL MODEL FOR INTEGRATED AUTONOMOUS PROCESSING: AN INTERNATIONAL BANK'S EXPERIENCE WITH LARGE DATABASES*

William F. Frank
Integrated Information Systems Associates
Warren, Vermont

Stuart E. Madnick
Center for Information Systems Research
Sloan School of Management
Massachusetts Institute of Technology

Y. Richard Wang
Management Information Systems Department
College of Business and Public Administration
University of Arizona

ABSTRACT

This paper presents a case study of a major international bank where competitive pressure has encouraged the development and deployment of information technology. Three key conflicting forces were found to have significant impact on the technical strategies: autonomy, integration, and evolution.

A conceptual system architecture was developed to respond to these needs by integrating only those elements critical to integration and allowing complete autonomy of the vast amount of application specification, development, and operation. The architecture was used in the development of two complex application groups. One application group was a high-volume transaction ensemble involving 20 gigabytes of data, the other application group was a sophisticated management reporting system involving over 1.5 gigabytes of data.

The architecture is designed to be an evolutionary blueprint since the migration from autonomy to integration is slow, and may not even be desirable. The approach of separating the external interfaces, message control, data control, and database components from the application processing components has provided a high degree of integration while preserving significant autonomy -- and the capacity to evolve further in both directions.

*Work reported herein has been supported, in part, by the Department of Transportation's Transportation System Center, the United States Air Force, the United States Space and Naval Warfare System Command, the Center for Management of Information at the University of Arizona, and Citibank.

1. INTRODUCTION

The successful exploitation of information technology requires a delicate and coordinated interplay of strategic application identification, organizational constraints and adjustments, and selection of appropriate technologies, as depicted in Figure 1. To understand these issues better, a Strategic

Applications, Technology, and Organization Research Initiative (Project SATORI) has been undertaken within the Sloan School of Management at the Massachusetts Institute of Technology.

This paper describes a case study within a major international bank where an attempt was made to utilize *technology* to mediate the conflict of forces

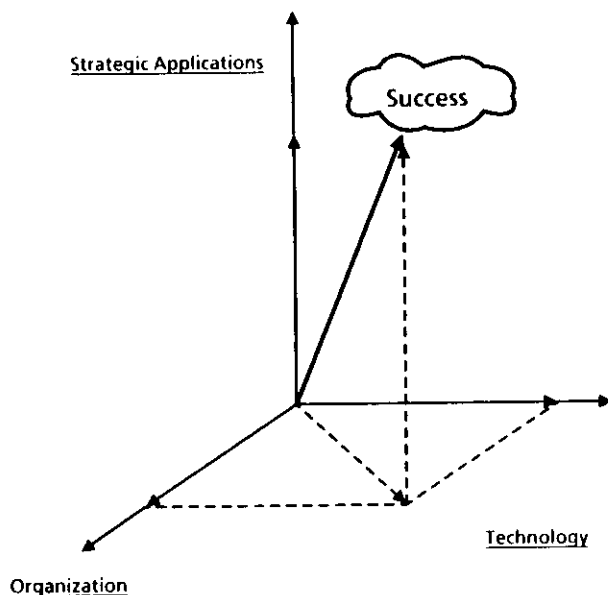


Figure 1. A Strategic Applications, Technology, and Organizational Research Initiative (SATORI)

toward integration, demanded by the needs of new *strategic* products and services and to respond to competition, and forces toward autonomy, inherent in the corporate culture of the *organization*.

A conceptual system architecture was developed to serve as a blueprint for integrated autonomous processing. Although still evolving, the conceptual model has been shown to be very effective at guiding system development and mediating the forces of integration and autonomy. The bank's environment, the motivation and description of this conceptual model, and the experiences of the past three years are presented in this paper.

2. BANKING ENVIRONMENT

The banking environment has experienced dramatic changes over the past two decades and it continues to change at an accelerated pace. Competitive pressure is increasingly being imposed on the banking industry on a multitude of fronts. There is little to constrain wholesale oriented institutions from crossing interstate barriers. Competition is also crossing international frontiers. In many countries, international wholesale banking is becoming much more aggressive as banks vie for the business of multinational corporations.

Information technology is being used extensively in banking (Lipis 1985). For example, in 1986 the value of computerized payments processed by the New York Federal Reserve and New York Clearing House often exceeded \$1 trillion in a single day. Technology has been critical to keep pace with the increased volume of financial activities; payments processed through New York financial institutions have increased 50-fold in the past 20 years, to the point where every four days an amount equal to the total annual US GNP is turned over.

During the 1990s, technological innovation will have an even more dominant effect on the financial services environment than interest rate volatility. The development of sophisticated information processing facilities are enabling institutions to offer a much broader range of products and services on a global basis with greater efficiency. Technologies such as distributed systems and database machines (Hsiao and Madnick 1977; Madnick 1977) are being employed by the industry to gain strategic advantage (Keen 1986; McFarlan 1984).

2.1 Autonomy, Integration, and Evolution

This paper reviews and analyzes the development and deployment of a conceptual model for integrated autonomous processing in a major international bank. The cultural forces in the bank favored an autonomous non-integrated approach, a tradition in which the responsibility for getting a job done is itself distributed to the lowest possible level, to maximize the independence of projects. To cope with the constantly changing environment and the distributed autonomous culture, it was recognized that three key goals need to be satisfied in designing information systems: autonomy, integration, and evolution.

On the one hand, each bank product (e.g., funds transfer, letter of credit, loans, cash management) is developed *autonomously*. Each product manager, in general, has complete freedom over his hardware acquisition and software development choices (e.g., hire his own development staff, retain outside contract programmers, or buy/modify suitable application packages). In the culture of this bank, this autonomy is critical since each manager is held solely responsible for his products. Excuses such as "the data processing people didn't do it right" are not acceptable. When information must be exchanged, it was often accomplished by "tape hand-offs," usually at night, as depicted in Figure 2.

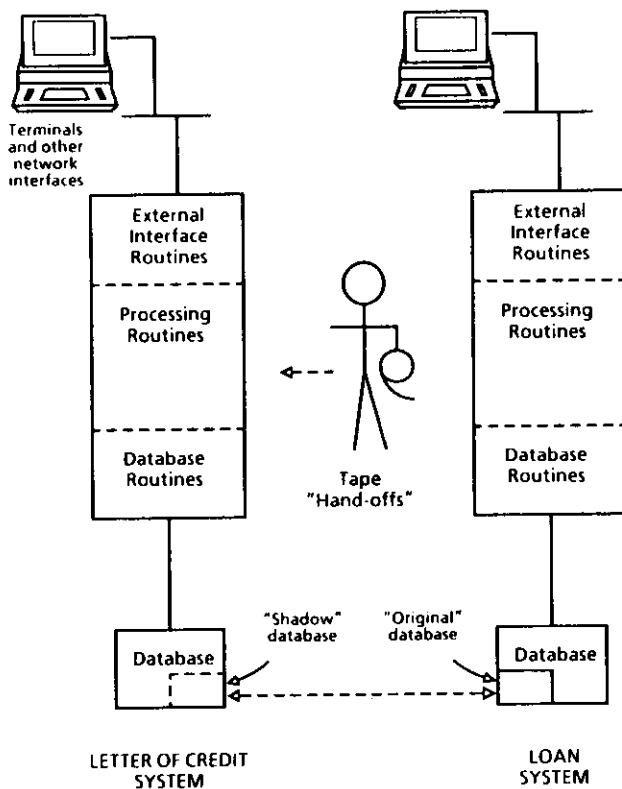


Figure 2. Independent Autonomous Systems

On the other hand, the need for *integration* has been increasing rapidly both at the user level and database level. Since each system had its own directly connected terminals, users that required access to multiple systems had to have multiple terminals in their office or walk to an area of the building that had a terminal tied to the system needed. The "tape hand-offs" mechanism was used to create "shadow" databases of each other's real databases. Since the shadow database diverges from the real database during the day, inconsistencies could result.

The problem of integration has been intensified by the need for *evolution* in at least three areas: current products, new products, and new technology. As the current products become more sophisticated, there is need to acquire more information from other systems. Increasing "tape hand-offs" leads to processing complexities and does not address the

need for up-to-date information. Many of the new products (e.g., cash management) are, in fact, a repackaging of existing products. To produce completely new systems would be expensive, time consuming, and impractical. Finally, to maintain an efficient and cost-effective environment, it is important to be able to take advantage of new hardware components without disrupting or discarding existing systems.

Traditional centralized database management system strategies provide integration, but have limited capabilities for evolution and autonomy. The purpose of this paper is twofold: 1) present a conceptual model, based upon the work of Lam and Madnick (1981) and Madnick and Wang (1987), that describes the architecture for an evolutionary, integrated, and autonomous environment; and 2) report the experience to date in implementing this model by the institutional banking group of a major international bank.

During the past three years, five products have been implemented using this conceptual model. This paper focuses on two of these systems: transaction investigations and management information system (MIS) reporting.

2.2 Transaction Investigations

Complex international financial transactions performed in high volume can be error prone, resulting in a significant number of discrepancies between customers' records and bank records. Customers inquire about the source of these discrepancies and the bank is responsible for reviewing its records to establish what it did and why, supplying these records to the customer, and making good if the bank is in error. The bank's activity with respect to such an inquiry is called an investigation.

Investigations are the job of sizable staffs in a large bank. They are usually performed with the aid of data stored on the transaction processing systems, microfiche, and printed reports. In addition, the investigation activity itself generates a history, which should be tracked to produce indicators of productivity, customer satisfaction, and the efficacy of the transaction processing systems. The Historical Data Base (HDB), needed by the bank to support these investigations and other applications covering the previous 90 days, must be able to hold at least 40 million records.

2.3 MIS Applications

Each transaction processing system generates various summary reports. Periodically, integrated MIS reporting for upper management was accomplished by manually entering data from a variety of these reports and other sources into an Apple computer spreadsheet.

Management desired new and more comprehensive MIS systems which integrated previously independently generated and inconsistently presented data. In total, a database of at least 12 million records, with over 250,000 additions per day, is needed to hold the relevant data. Furthermore, these new systems should facilitate the modelling of the effects of alternative decisions (i.e., what-if analyses fed by current performance data).

3. CONCEPTUAL MODEL

3.1 Model Architecture

A flexible system architecture was conceived to serve as a "blueprint" or "conceptual model" to guide the development of new systems. This conceptual model consists of seven major functional components as depicted in Figure 3. These components are separated into five layers, with the integrating application-independent layers (external interface, message control, data control, and shared data resource) surrounding the application processing components (Madnick and Wang 1987). For this bank, these application processing components are separated into three classes of applications: transaction processing, information processing, and administrative processing.

3.2 Autonomy Aspects of Model

This architecture attempts to mediate the conflicts between the goal of autonomy and the goals of integration and evolution. All of the layers, except for message control and data control, lend themselves to unlimited autonomy. Each product manager could acquire and manage his own resources including 1) terminal/network gateway hardware, 2) application processing computers and software, and 3) database computers and software.

In the past, as shown in Figure 2, these three decisions were bundled together. In practice, the primary concern for autonomy involved the application processing, with a lesser concern for the

database, and minimal concern for the external interface. It is in the application processing that the functionality of the product is manifest. It is important that enhancements and corrections, as well as the initial development, be able to proceed with minimal needs for coordination or delays due to the managers of other areas of the bank and other computer systems.

Given the architecture in Figure 3, each manager has complete control over his application processing system. Furthermore, as many separate database systems as needed, or desired, can be created. It is expected that initially there will be more databases than theoretically needed, due to the influence of past practice. The integrated autonomous architecture provides access to these databases by other applications as well as an evolutionary path for eventually integrating these databases, as the needs for integration intensify.

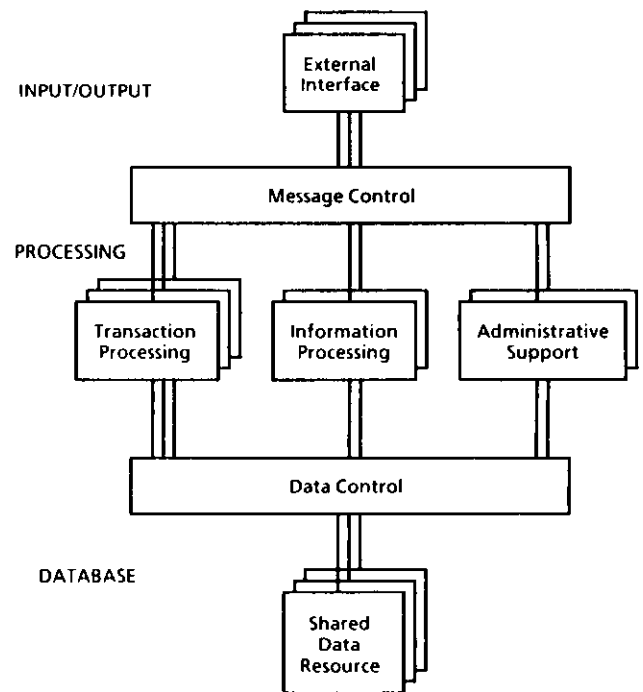


Figure 3. Conceptual Model

3.3 Integration and Evolution Aspects of Model

There are several underlying concepts and components of the model which address the issues of integration and evolution.

Message Control and Data Control perform unifying functions for the model. They are the points at which all processing will be coordinated. For example, in principle, any terminal can access any application. Furthermore, application subsystems can utilize the Shared Data Resource to manage and maintain data which is common to more than one component.

Message Control and Data Control are both conceptually *single* entities. The other components are *types* of processing functions. There may be many instances of each type (e.g., multiple transaction processing systems and multiple shared data resources).

3.4 Model Components

It is important to realize that the model components of Figure 3 are *logically* separate. They could be mapped to actual hardware in various ways. For instance, in the transaction-investigation system, each component resides on a separate processor; whereas, in the MIS system, many components reside on a single processor.

3.4.1 External interface

The external entities interfacing with these banking systems fall into five categories: 1) payment networks; 2) communication networks; 3) customer terminals; 4) professional workstations; and 5) other intra- and/or inter-bank systems.

3.4.2 Message control

Message Control coordinates the passage of messages between the processing components. This involves routing, translation, sequencing, and monitoring:

- o *Routing* accepts a request for the delivery of messages to a particular logical function and determines the appropriate physical address to which the message should be delivered. Routing can thus accommodate changes in the availability and location of functions. Routing can also help

coordinate requests for services that involve several functions.

- o *Translation* maps a limited number of protocols from one standard to another.
- o *Sequencing* determines the order in which messages are to be delivered to recipients on the basis of established priorities.
- o *Monitoring* determines the state of messages within the system at any given time. Monitoring thus includes the responsibility for the integrity of a message from the time it is presented by one component until it is accepted by another.

3.4.3 Transaction processing

Transaction Processing refers to the applications which execute the customer's financial instructions. These systems typically retrieve and update a significant amount of data (e.g., client balances). The sub-functions of Transaction Processing include validation, risk management, and accounting.

- o *Validation* functions are those which perform review of instructions to ensure that all information needed for processing is present and that the information is consistent with previously defined rules for data elements. Incomplete or invalid requests may be "repaired" by augmenting or clarifying the requested information, either with information available internally or from external sources such as the requestor.
- o *Risk Management functions* are those which verify that the transactions being processed do not violate limits, conditions, or policies established by the customer, the bank, or the various regulatory agencies. Ideally, these functions should be synchronized with the processing of transactions. In some cases where this is not feasible, "after-the-fact" Risk Management functions can be used to initiate corrective action.
- o *Accounting* records the cumulative impact of the transactions completed. Accounting functions are characterized as being continuous over time, in contrast to the discrete events which take place in most of the transaction processing functions. For example, in the banking environment, accounting takes place on two distinct levels:

customer accounting and organizational accounting.

3.4.4 Information processing

Information Processing refers to all the subsystems that perform analysis, calculations, or restructuring of the data (e.g., consolidated financial statement). The sub-functions of Information Processing include user interface, static reporting, ad hoc reporting, and access to outside data resources.

3.4.5 Administrative support

Administrative Support provides facilities for the performance of office functions by administrative or managerial personnel. This activity is required to maintain organizational, procedural or personal information. Example facilities include electronic mail, word processing, correspondence files, and inventory controls.

3.4.6 Data control

Data Control coordinates access, presentation, and the passage of data between processing functions and the Shared Data Resources. It routes queries and updates to the appropriate component of the Shared Data Resources, performs security and priority functions, maintains concurrence control over the shared data, and returns responses to the appropriate processing function.

3.4.7 Shared data resources

Shared Data Resource is the component responsible for holding the information common to one or more other components of the Model. Although this activity is logically centralized in the Shared Data Resource, it may contain multiple elements (storing different segments of the shared data, or different organizations of the shared data). The Shared Data Resource performs two functions:

- o *Information Management* determines what information must actually be stored and retrieved to satisfy the request, performs the transformations necessary to produce the required information, and determines how the information is to be stored or retrieved.
- o *Storage Management* determines physical locations of data and access storage devices.

4. IMPLEMENTATION EXPERIENCE

4.1 Summary of Results

The conceptual model of Figure 3 has been the general architectural guideline for system development over the last three years. Two particular projects completed in that time, including two large databases (20 gigabytes and 1.5 gigabytes), will be described. VAXCLUSTER technology and the ORACLE relational database management system were used extensively to implement the conceptual model. A portion of the development of one of the applications used the STRATEGIM language. Most of the other applications were either programmed in the C language or used existing packages.

The goals which led to the development of these two systems were threefold: 1) to create effective *applications* for particular user groups; 2) to provide a *central repository* and further dispatch point for all banking transaction processing historical data; and 3) to provide an *application systems architecture* in which MIS data of various kinds was organized according to a partial order of increasing levels of abstraction or aggregation, so that higher levels of data would be created by batched flows of data from lower levels.

The design and implementation of the systems to meet these goals were greatly affected by the cultural factors and business considerations mentioned earlier. The first and last of the goals (good applications and structured aggregation) have been largely achieved by the implementation. The second goal (shared historical data resource) was approached very cautiously. New technology and the experience gained makes full attainment of this goal more feasible today than it was in 1984.

Despite much skepticism about the performance of a relational database system on databases as large and active as these, particularly on minicomputer technology, the applications making use of these databases ultimately performed very well.

4.2 System Design

4.2.1 Application of the conceptual model

General plans for new systems called for the segregation of system development efforts, and of hardware and software components, into specific

classes of systems which would correspond to components of the conceptual model.

It was important that this be done on an evolutionary basis since the inventory of existing systems constitutes some 20 million lines of code. There was (and is) insufficient business motivation to replace all these working systems even to achieve such general goals as data integration. Instead, as new applications are built and old ones replaced, it is intended that they be brought into greater conformity with the model. This means that only those pieces of integrative components required to support a developing application may possibly be built.

In addition, it was not practical that the integrative software and hardware required for Message Control and Data Control be custom built for the bank. Instead, it was expected that these components would be purchased commercially. At the time the model was proposed and accepted as the basis for new development in the bank, complete Message Control and Data Control software was not commercially available. Over time, it has increasingly become available. Thus, one of the major values of the model has been positioning the bank for the arrival of such new technology.

The following two sections describe the ways in which the ideas of the conceptual model were applied to the development of the historical database system for transaction investigation and the related MIS system, first from the point of view of their functional organization and then from the point of view of implementation issues.

4.2.2 Role of a historical database

The historical database was envisioned as a data resource shared by multiple applications: Transaction processing systems would no longer have the responsibility of storing historical data nor need to produce a variety of different summaries of the same data for the different information processing systems. Instead, all completed transactions and proofed account balances of each type would simply be written to a common database. Information processing systems would each extract the data they needed from the historical database, and use that data independently from the uses put to it by other systems.

The goals of such a historical database were to simplify the work of transaction processing systems, and more importantly, to simplify inter-system data flows. The current flows had reached a level of complexity which were, in total, no longer known to any one person and had no discoverable principle of organization. They required several people several months to catalogue. Of most urgency, dependencies on tape hand-offs and other information flows between systems meant that each year more systems were unable to complete their off-line work in the daily time periods available, especially because customers all over the world demanded that some systems be available on-line virtually all the time. The data-flows to be provided by this new scheme are shown in Figure 4.

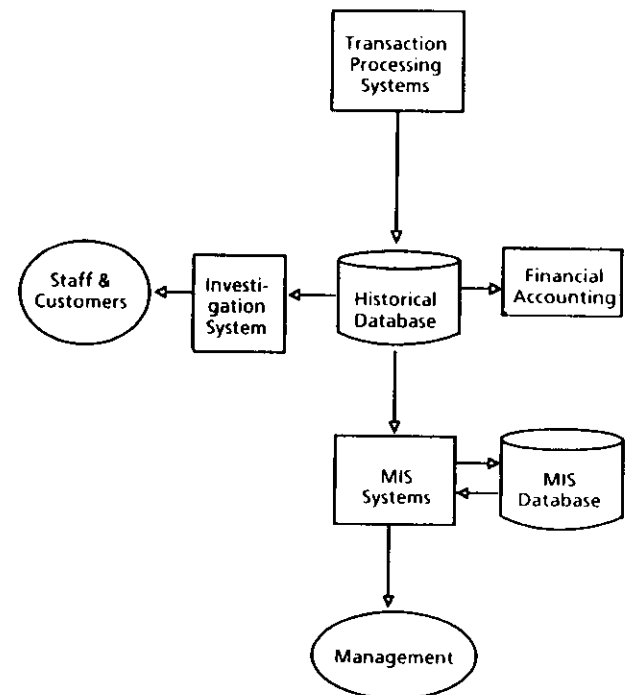


Figure 4. Role of Historical Database

4.2.3 Structured aggregation in MIS systems

In addition to the benefits mentioned above, the integrated MIS system was intended to improve the overall organization of inter-system communications through a systematic way by determining which MIS applications should communicate with which other applications.

This was to be accomplished by describing the input requirements and the output of various desired MIS applications in a uniform way and then identifying the lowest cost (least transformation required) connections between such applications so that the output of some applications becomes the input to others.

The output of each MIS application is treated as a database in its own right and the data-aggregation function is regarded as the primary role of this sort of MIS application (e.g., standard cost accounting). The result is a partial order of aggregated databases in which all databases depend ultimately on the raw historical data, as depicted in Figure 5. The current collection of MIS databases consists of seven levels and the dependency diagram barely fits on a large wall.

The general design of the MIS system is to regard various sets of tables as levels of aggregation. Although the processing is quite complex, the fact that all of the data is maintained as a single shared data resource has dramatically simplified the operations of the MIS system components.

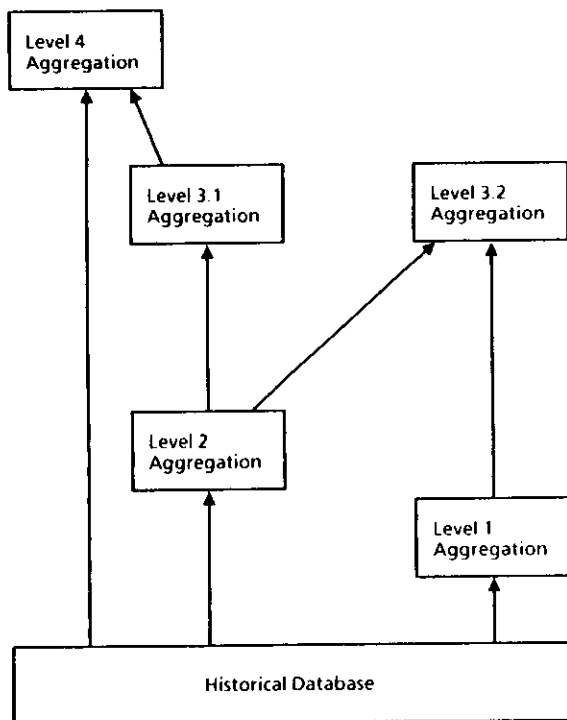


Figure 5. MIS Data Partial Order of Aggregation

4.3 Model Interpretation

By "interpretation" we mean a functional mapping of model components to hardware and software components. The interpretation of the model has varied considerably between the transaction processing and information processing environments and has varied over time. For instance, front-end processors are now being distributed geographically, which has caused some changes and extensions.

The initial interpretation involved a "null" message control layer, since the early information processing applications did not communicate with each other, and all screen management was done in the same processor as the applications, although with separate software. Only later, when screen management software capable of communicating via datagrams with the applications processors became commercially available on personal computers, did a message control layer become significant in information processing.

The five layers of the conceptual model are interpreted as shown in Figure 6: (1) The *external interface system* consists of terminal controllers, wide-area gateway boxes, a terminal to host Ethernet, network interfaces for the application host machines, and the screen management software on these hosts. (2) The *application layer* consists of the application software. (3) The *message-control layer* shares a host-to-host Ethernet with the data-control layer. (4) The *data-control layer* consists of software on the application hosts enabling communication with the database processors over the host-to-host Ethernet, the communications software on the front-ends of the database processors, and a portion of the database management system which includes query analysis and concurrence control. (5) Finally, the *shared data resources layer* consists of the back-ends of the database management systems running on the database processors and the storage systems to which they are connected. These components are described in more detail below.

4.4 System Software Requirements

The bank's development policies emphasize individual projects be as small as possible, both in number of personnel and time frame. Ideally, a project should take less than a year and require no more than five developers (Appleton 1986). The systems software chosen to support these information pro-

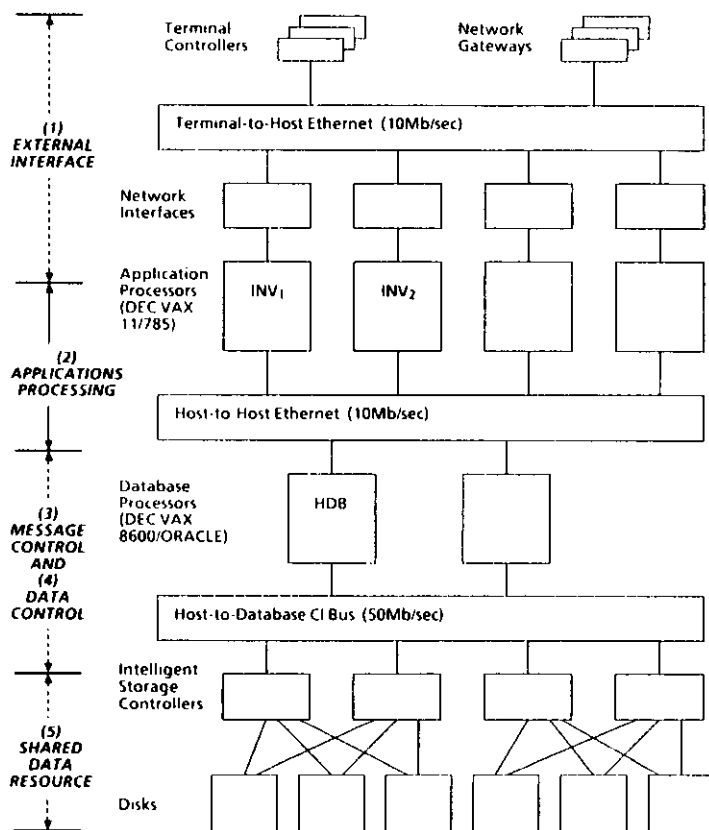


Figure 6. Model Interpretation

Processing applications differed from that chosen to support transaction processing applications (such as the funds transfer system). The major reasons for this difference are the technical and performance differences between the transaction processing and information processing systems.

The information processing systems communicate with the outside world largely interactively. Their databases are the only components which must be fully recoverable. The tolerable time to recovery may be as long as several hours, whereas for transaction processing systems this is typically a matter of minutes or even seconds. In addition, information processing systems are highly fluid: requirements change from month to month.

The primary software systems required on the information processing side were flexible screen managers, database management systems, and host-

to-host communications software. These systems lend themselves to prototyping and non-procedural languages.

4.4.1 Operating environment

The hardware and software chosen to realize these components was largely based on the DEC VAX. The group's processors currently include about thirty VAXes and two IBM mainframes. While the IBMs exchange messages with the VAX systems, the two environments are not currently integrated into the same conceptual model: the IBM systems, developed earlier, perform high volume batch work running purchased packages which provide end-to-end support for these applications.

The reasons for this preponderance of VAXes were the experience base of available developers and support personnel as well as the cultural forces, noted earlier, favoring separate computers for separate jobs. Thus, VAXes have invariably been chosen in the cases where preliminary analysis indicated that VAX systems were capable of handling projected workloads and where specialized software was not immediately available off the shelf for the IBM MVS systems.

In the case of the transaction investigation system, the availability on the VAX of a software package to support investigation tracking was an important factor in the selection of the VAX, since here the size of the database might have indicated that a mainframe would be a better *prima facie* choice.

As a result, VAXCLUSTER technology has been widely used in a variety of ways. To keep Figure 6 simple, certain complexities have been omitted. For instance, some of the applications processors are also connected to the Computer Interconnect (CI) Bus to allow them to take the place of a defective database processor. Furthermore, for increased reliability, each Ethernet shown has an additional backup Ethernet.

4.4.2 External interface

All communications with the ultimate system users are terminal-based. Certain data are downloaded to PCs for spreadsheet manipulation, graphics, and report printing. Terminal-to-host and PC-to-host communications (part of the External Interface

system) is accomplished physically by an Ethernet. In fact, two Ethernets are used: one for production systems and another for development systems which serves as backup for the production network.

Bridge controllers (local terminal controllers, dial-up systems, and X.25 gateway controllers, which provides a connection to remote terminals via an in-house worldwide network) are used to serve both the terminals and the computers. They communicate with each other via XNS. Sun Workstations are used on this network as network control and configuration management devices.

Screens are managed from the VAX application machines, using Viking Forms Management software, which in turn communicates with the application software.

4.4.3 Applications

The transaction investigation application software largely consists of calls to the screen management and the database management systems. In addition, it performs any processing required to make the necessary transformations between these two systems. The package purchased to support these investigations, in actuality, does considerably more than this.

4.4.4 Data control

The Data Control level consists largely of database management software and general purpose hardware. Some of the features of Data Control could not be implemented, owing to the lack of commercial distributed database management systems at the time of implementation.

In addition to development productivity, the major issues concerned performance, because of the large size of the databases. Other important criteria were: 1) preferences for relational systems as having the longest future and offering the greatest productivity gains, 2) relatively widely used systems, and 3) adherence to standards and ad hoc standards when possible, such as CODASYL or SQL.

At the start of the project, there was considerable, mostly unsolicited, expression of opinion and concern that relational systems "performed poorly" and were therefore unsuitable for large databases.

Of course, it is meaningless to talk about good or bad performance except with respect to a specific use and to specific parameters of performance. In this case, both the historical databases and the MIS database would be written almost solely in batch mode. Although the volume of such writes was quite large (300,000 512-byte rows per night), the writes were all appends, no updates; reads and writes were never expected simultaneously against the same tables; all reads were predictable since very little access to the databases would be ad hoc; and the database could be designed in such a way that most read requests would require no joins between tables.

Performance projections were established by study of the results of benchmarks performed by other institutions, by review of published performance analyses, including a study by the National Bureau of Standards, and by analysis of the likely effects of design features of the systems.

It was quickly established that the commercially available systems based on pointer chains (which happened to coincide with the available CODASYL systems under consideration) were incapable of performing the nightly number of batch appends even within a period of 24 hours, while the systems based on indexes (the relational systems under consideration) would permit these batch appends within a number of hours.

The two systems considered most seriously were INGRES and ORACLE. Benchmarks as well as general opinion suggested that ORACLE's performance in performing unlocked reads against single tables in large databases was superior to that of INGRES. ORACLE exhibited linear (and almost flat) increase in response times to such simple queries as the size of the database increased. INGRES was superior to ORACLE in the performance of complex joins and offered superior fourth generation development tools.

For the type of production applications envisioned, ORACLE appeared to be a better choice, although neither system would be, in its then current version for VAX hardware, capable of handling the number of simultaneous users ultimately projected for the investigation system. Special design approaches were used to overcome these problems. Furthermore, one could be confident that both hardware and software performance would be continuously improving.

4.4.5 Shared data resource

The data resource used in all the VAXCLUSTER systems is supported by DEC cluster storage, which consists of the very high speed CI Bus (70 megabits per second), connecting a number of processors and a number of intelligent controllers (currently a total of 16 such devices) in such a way that any processor can communicate with any controller. Controllers are in turn connected in pairs to dual ported disks, allowing full redundancy of hardware components. For instance, within the cluster, when a processor fails, its work can be taken over by another processor which has immediate access to all of the same disks.

4.5 Development Histories

4.5.1 Changes in the plans

In developing these systems, it became clear that the strength of certain features of the bank culture was even greater than anticipated: independence and competitiveness between managers and a predilection for achieving fast, tangible results. Steps to integration actually needed to be smaller than the original plans called for. In building and supporting stand-alone applications, each development group has the total responsibility for delivering an application to users, and each group has a strong aversion to relying on cooperation from other groups not equally responsible for some deliverable.

Only a portion of the transaction processing data (that involved with funds transfers) was required to support the needs of the investigation application. The MIS system, however, required summary data from all the transaction processing systems. These systems already had the capability of providing the lowest level of aggregation required by the MIS applications, since they were supplying this data to current applications. A coordinated effort would therefore provide theoretical future benefits, while increasing current development time, cost, and risk.

As a result, the transaction investigation and MIS projects, which were started at approximately the same time, wound up with each largely going its separate way. That is, instead of extracting and aggregating data from the historical database produced by the transaction investigation system, the MIS system receives its own partially aggregated data feeds from transaction processing

and financial accounting systems, as shown in Figure 7.

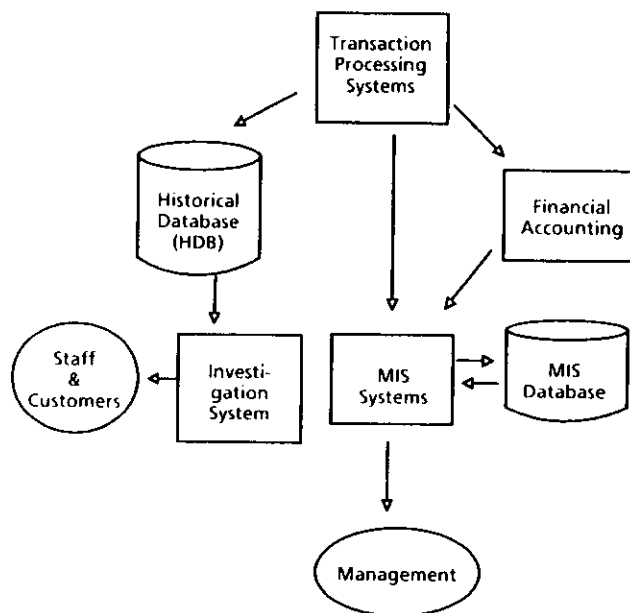


Figure 7. Current Data Flows

4.5.2 Transaction investigation system (HDB)

Projections called for at least 100 simultaneous users using the investigation system and having read-only access to its historical database (HDB), with maximum rates of one database request each five seconds.

The applications and the database are supported on a VAXCLUSTER, as shown earlier in Figure 6. A VAX 8600 is used to support the database, while the investigation application runs on multiple VAX 11/785 "front end" machines, which communicate with the database machine via DECNET and the VMS mailbox facility. The investigation application constructs a query in the form of a "query type" and a set of parameters. The historical database system translates this into an SQL query and returns the response table to the application.

The configuration has grown to two front-end 11/785s and an 8600 "database processor," while ORACLE has gone through two major new releases. Several changes in design to improve performance

have also been made over the years. The modular evolutionary capability of the conceptual model architecture greatly facilitated all of these changes.

The initially installed production system (on a VAX 11/780 front-end with a VAX 11/785 back-end) supported thirty users with database query response times of 90 to 120 seconds. The current software and hardware supports 160 simultaneous users with database response times of five to seven seconds. Before automation, each search for historical data required at least fifteen minutes of an investigator's time.

The full cycle of design, development, testing, installation, training, and live operation of the transaction investigation system required about six months, with one programmer responsible for the software development.

A significant number of small problems, and a few large ones, were discovered in the course of development. For example, it was discovered during the course of development that the ORACLE "money" data type would not hold large enough amounts for the needs of such a very large bank. It was also discovered that the tape recovery of very large tables would take much too long and that mirroring was not economically justifiable. It was also found that the re-indexing of very large tables was taking too long (the original analysis counted only the time required to index the newly appended data each day, not the entire 20 gigabyte database). To solve this problem, tables were partitioned by dates and application code was written to permit searches across ranges of dates.

Many of these problems could have been anticipated by careful up-front studies. However, such studies would have required as much time as building the system did. Without the flexible and powerful environment used, the same sorts of unanticipated problems would have been likely to cause development disasters, as they often had done in the past in the experimental culture of the bank.

4.5.3 MIS systems

The initial phase of MIS system development and implementation also required about six months, with three contract developers and one manager. About one third of this time was devoted to learning to read correctly the tapes from the various non-integrated sources of MIS data and developing programs

to map these tapes to relational tables without repeating groups, multiple record types, variable length records, etc.

A major innovation of the system was the virtual elimination of the printing of MIS reports: particular pages of reports of interest to individual managers are viewed on terminals, and printed locally if so desired.

The MIS system had intense advocates in the MIS department, who specified exactly what they wanted to see in the system. It is more common in the bank for users to ignore new systems until they are delivered to them. The system has been constantly enhanced and enriched, with the MIS group developing a great deal of skill in the use of SQL and ORACLE development tools for small applications. In fact, this system is not called the "MIS" system by most of the users, it is called the "ORACLE" system.

The summary of data from this system began to be used in business planning sessions, which lead to the desire to use the data for forecasting and analytic modelling purposes. The result was the creation of a strategic MIS system, using the STRATAGEM modelling language, which receives data from some of the highest level applications of the ORACLE MIS system. The use of this system for analytic and strategic purposes, rather than purely reporting purposes, has grown slowly but steadily.

5. CONCLUDING REMARKS

The conceptual model described in this paper has served as an evolutionary blueprint. The technology of separating the external interfaces, message control, data control, and the database components from the application processing components, provided the capability for high degrees of integration needed for new strategic applications while preserving significant autonomy inherent in the bank's culture. The lessons learned from these experiences may be valuable for the many other organizations now facing similar challenges.

REFERENCES

Appleton, D. S. "Very Large Projects." *Datamation*, January 1986.

Hsiao, D. K., and Madnick, S. E. "Database Machine Architecture in the Context of Information Technology Evolution." *Proceedings of the Third International Conference on Very Large Data Bases*, pp. 63-84, October 6-8, 1977.

Keen, P. G. W. *Competing in Time: Using Telecommunications for Competitive Advantage*. Ballinger, Cambridge, Massachusetts, 1986.

Lam, C. Y., and Madnick, S. E. "Composite Information Systems -- A New Concept in Information Systems." *CISR Working Paper # 35*, Sloan School of Management, Massachusetts Institute of Technology, 1981.

Lipis, A. *Electronic Banking*. John Wiley and Sons, New York, 1985.

McFarlan, F. W. "Information Technology Changes the Way You Compete." *Harvard Business Review*, May-June 1984, pp. 98-103.

Madnick, S. E. "Trends in Computers and Computing: The Information Utility." *Science*, Vol. 185, March 1977, pp. 1191-1199.

Madnick, S. E., and Wang, Y. R. "Evolution Towards Strategic Applications of Databases through Composite Information Systems." *Working Paper #1862-87*, Sloan School of Management, Massachusetts Institute of Technology, February 1987.