Universität-Gesamthochschule Paderborn

Diplomarbeit

# Development of a design-environment for the enterprise wide data integration by combining groupware based business processes with legacy systems

Prof. Dr. L. Nastansky

Wintersemester 1998/99

Betreuer:
Dipl. Inform.Carsten Huth

vorgelegt von:

Sebastian Risse

Wirtschaftsinformatik
Matr. Nr. 3176525

Zur Lüre 45
37671 Höxter

# Table of contents

## List of Abbreviations

| | | |
|---|---|---|
| API | = | Application Programmers Interface |
| ASAP | = | as soon as possible |
| BPR | = | Business Process Re-engineering |
| CPU | = | Central Processing Unit |
| DBMS | = | Database Management Systems |
| DECS | = | Domino Enterprise Connection Services |
| DLL | = | Dynamic Link Library |
| EAI | = | Enterprise Application Integration |
| e-Commerce | = | electronic Commerce |
| EIS | = | Executive Information Systems |
| e-mail | = | electronic mail |
| ERP | = | Enterprise Resource Planning |
| GB | = | Giga Byte |
| GUI | = | Graphical User Interface |
| HD | = | Hard-Disk |
| IBM | = | International Business Machines |
| ID | = | Identification |
| IS | = | Information Systems |
| IT | = | Information Technology |
| LAN | = | Local Area Network |
| LEI | = | Lotus Enterprise Integrator |
| MAN | = | Mobile Area Network |
| MB | = | Mega Byte |
| MQ | = | Message Queuing |
| MQEI | = | Message Queuing Enterprise Integrator |
| NC | = | Network Computer |
| ODBC | = | Open Database Connectivity |
| OLAP | = | On-Line Analytical Processing |
| OLTP | = | On Line Transactional Processing |
| PC | = | Personal Computer |
| RAM | = | Random Access Memory |

| | | |
|---|---|---|
| RDBMS | = | Relational Database Management Systems |
| SQL | = | Simple Query Language |
| UDB | = | Universal Database |
| UI | = | User Interface |
| WAN | = | Wide Area Network |

# 1 Introduction

## 1.1 Scenario

Many companies are finding the need to focus on new information technology to be competitive and to position themselves for the future. In part, this is due to the commercialization of the Internet, the growing globalization of business but also political issues such as the European Union and entailed, the introduction of a new currency, the Euro. Based on these demands for more efficient and effective operations and communications, corporate management is now intent on exploiting their existing enterprise systems data recourses to gain competitive advantage in a together growing world.

Companies are mainly using mainframe systems to store their crucial information, even though client/server systems with user friendly graphical interfaces are most commonly seen in modern offices. Large, medium and even small companies prefer keeping their important data saved on systems they have been able to rely on for more than twenty years. Client/server based systems are chiefly used for new applications that meet the demands related to new technologies such as the Internet, widespread e-mail systems and groupware environments. Slowly, companies are recognizing the advantages of this new technology and are starting to migrate their legacy systems to client/server technology.

Different approaches for transferring mainframe-based legacy systems to client/server based windowing technology. These approaches all have the same disadvantages: high investments and a time consuming transition period combined with the need for new skilled experts and additional staff training are a high prize to pay for the later benefits. The re-engineering of the existing and long proven legacy systems is gathering momentum[1].

To replace the mainframe computers with client/server based systems could easily lead to a fragmentation of the enterprise data. Organizations already complain that too much of their information is distributed inefficiently, which often results in information being mislaid, received late, or not able to be used at all. This is obviously a result of the extended use of PC's in all departments. Important information, which should be

[1] Compare to Miller 1998, p.3

available “at your fingertips” is lost on some personal computer because there is no network connection to allow other computers to share that information.

Almost independent of the above mentioned crucial data and strongly linked legacy systems, a new kind of information technology has developed in the past few years: groupware. Groupware is now a well known technical term for applications that allow users to communicate, collaborate and coordinate information between many different groups of people in many different ways. Groupware components include e-mail system, shared document databases, replication, off-line working, Internet functionality and workflow structures, to name just a few. The base for workflows is enriched by additional applications from different vendors, which provides a graphical design tool to build complex business processes.

At the time this work was produced, IT companies such as IBM push the combination of groupware and mainframe systems. The groupware platform Lotus Notes is able to run on systems such as AS/400 or S3/90, which includes the provision of the enterprise data from the legacy systems to groupware.

The advantages and the disadvantages of the legacy systems and the new possibilities of the modern technology lead to a new approach: the combination of these technologies to form an enterprise-wide process system which is able to link, distribute and collect data and applications throughout the organization.

### 1.2 Objective of this Work

The data of today's organizations is present in a very distributed and heterogeneous information system. Depending on the progress the companies have made migrating their technology to client/server based systems, the data is stored on mainframes, client/server systems and PC’s. To integrate this data into a human based workflow system without fragmenting the data requires the use of an easy to use process design tool.

The objective of this work is to use an existing process modeler and to enhance the functionality by integrating the enterprise data from any external database management system (i.e. mainframe based legacy systems, hosts of different technology or any relational databases from a client/server background) into the groupware based processes. The modified modeler allows the process designer to make use of enterprise

data to build business processes by using the advantages of groupware platforms and legacy systems. The modeler should support the new demands of organizations to exploit the benefits of the Internet, Intranet, Extranet, off-line working and e-Commerce combined with the advantages of the underlying “old fashioned” mainframe systems.

The work also provides an approach to the combination of groupware based workflows and host based transactional systems for enterprise application integration (EAI)

### 1.3 Outline of this Work

Chapter two gives an outline of the existing information systems. This is divided into three parts:

Part one is a short summary of today's groupware systems. The work concentrates on the most important components including shared document databases, e-mail, replication, off-line working, Internet functionality, compound documents and security. Another topic is the groupware based workflow, including the components of the basic process modeler.

Part two is a look into the existing hardware and software architecture with special focus on host based legacy systems. The advantages and disadvantages are laid out, as well as the current status of it in companies.

Part three demonstrates the opportunities which lie in the combination of groupware and legacy systems. The different available connections between these two technologies are shown together with its benefits and disadvantages.

Chapter three shows the demands for the new system and gives an overview of the components that have to be included. It outlines a step-by-step procedure from host based transactional systems to a transactional/groupware integration and beyond.

Chapter four shows a prototype that has been implemented during this work. The underlying architecture and the different components of the system are demonstrated. A look into the future is given, showing the components which could not be implemented during this work.

The results are explained in chapter five. A design environment that uses the enterprise wide data within groupware based business processes in combination with legacy systems. It shows the capacities of the approach, the structure of the solution and it

outlines the advantages and disadvantages. It points out the lacks and errors as well as system requirements, performance tests and installation guidelines. There is also a demonstration of example workflows. It refers to current issues like e-Commerce and order processes.

Chapter six concludes this work with a look at what will come next. Additional features and needs are explained as well as a look at an upcoming issue called Virtual ERP.

# 2 Terms and Concepts

In this chapter the underlying components will be examined. In the first two parts the existing technologies, groupware platforms and legacy systems are investigated. This allows to form a more complex combination of both in part three.

## 2.1 Groupware Platforms

### 2.1.1 Basic Definitions

Finding a uniform definition for groupware is impossible. There are as many definitions for groupware as there are people trying to define it. Here are three popular definitions for groupware:

“An intentional group process plus software to support them” Peter and Trudy Johnson-Lenz, 1978

“A co-evolving human-tool system.” Doug Englebart 1988

“Computer-mediated collaboration that increases the productivity of functionality of person-to-person processes.” David Coleman, 1992

Groupware applications provide groups of users with communications, collaboration, and coordination capabilities to improve business work.

### 2.1.2 Components within Groupware Systems

The basic structure of groupware systems is explained in great detail in different books[2], therefore only the components which are important for this work are outlined.

The underlying document database structure of groupware platforms allows end users to access the data from different locations. The databases are held mainly on servers,

[2] Please refer to Coleman 1997, Simon 1996, Lloyd 1996, Bock 1995, Greenberg 1991

which can be accessed by clients via LAN, WAN or MAN. This is called the share model, the user shares the same set of data. The send model uses e-mail systems to exchange information between the people involved. The documents in the databases are compound documents, which can store semi- and unstructured data as well as a logic or intelligence[3]. These documents can be replicated to any workstation, which allows off-line or mobile working. With different programming techniques the databases can be enhanced with macros and a certain logic.

### 2.1.3 Workflows based on Groupware Systems

Workflows are business processes containing a number of related tasks that have to be completed in a special sequence by chiefly persons or computer macros. Again, there is a number of books related to this topic[4].

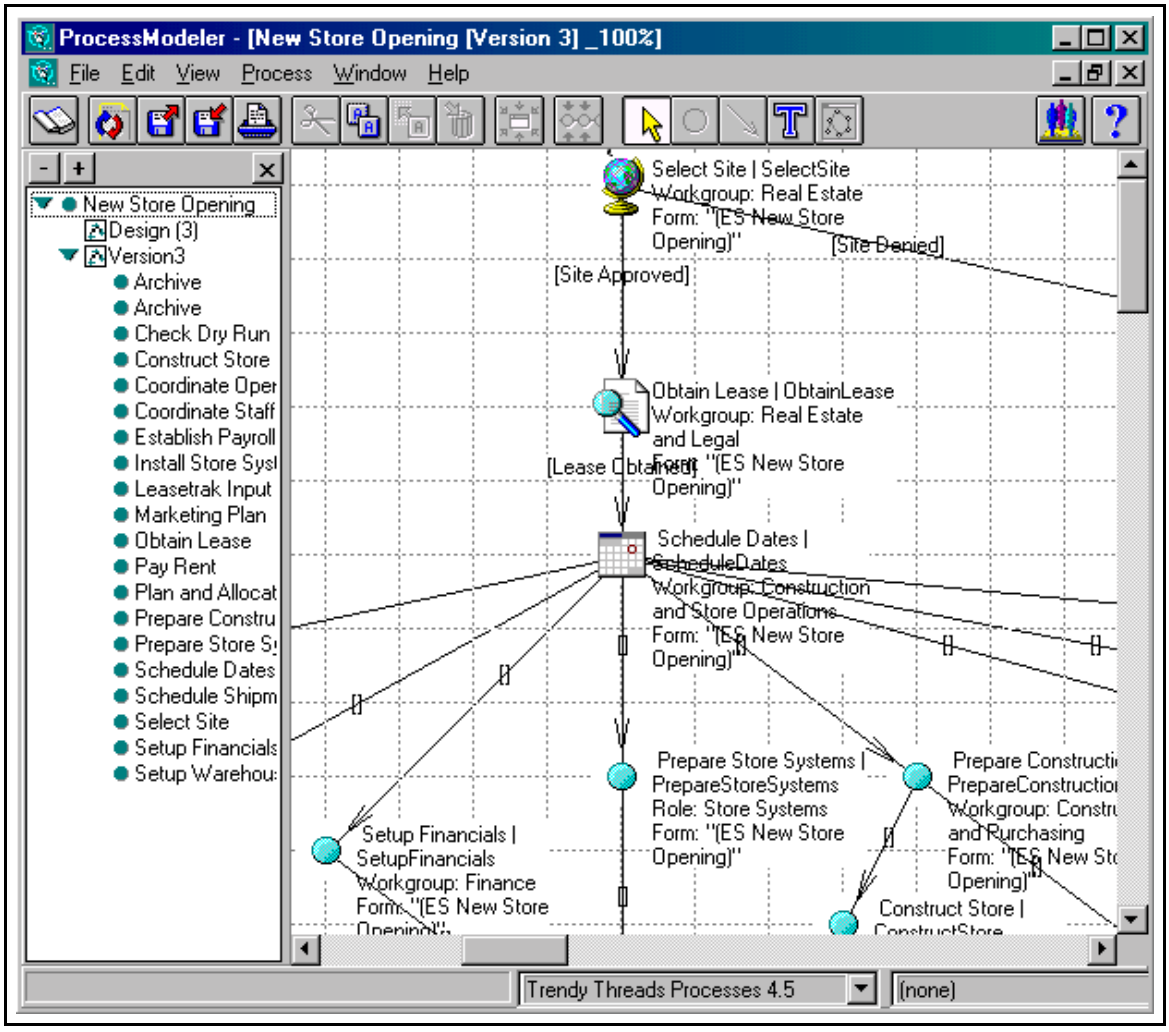
Within a groupware application, business processes can be hard coded directly into the groupware platform. This requires good programming skills and, dependent on the process length, a relatively long time. The hard coded workflow is difficult to modify and that has to be done by the programmer who did the initial implementation of the process. If different versions of a process are needed within the workflow, hard coding is almost impossible to realize.

To eliminate the difficult and time consuming programming of business processes, applications from different vendors are introduced to support the design and update phase. Graphical interfaces provide the process designer with the right set of tools for building predefined processes which can then be used within the groupware system. One application, which is the underlying base of the prototype, is the ProcessModeler, a part of the Espresso product family, from PAVONE Informationssysteme GmbH. This simple to use tool gives the process designer all the functionality to build state-of-the-art workflows which can then be used within the groupware platform Lotus Notes.

---

[3] Please to Dierker, 1998,

[4] Please refer to Chaffey 1998, Khoshafian 1995, Riempp 1998, Jackson 1997, Jablonski 1996




Picture 1: PAVONE Espresso ProcessModeler

### 2.2 Legacy Systems

"A legacy information system is any information system that significantly resists modification and evolution to meet new and constantly changing business requirements"[5]

Legacy Systems are programs that exist in companies for many years already. They are often written in programming languages such as Cobol or Fortran. These applications are often used on mainframes or minicomputers but can also be seen on client/server architectures.

In the 1990's, 70 percent of the worldwide data was stored on IBM mainframe systems[6]. This shows that it would be fatal to underestimate the status of these mainframe systems. This also leads to the conclusion that mainframe data, the legacy data, needs to

[5] Brodie 1995, page 3
[6] Lotus White Paper, May 1998, page 1

be included in the model of this work. The newer technologies such as client/server and Internet capabilities need to be considered as well, since this is getting more and more mission critical for organizations.

The next two paragraphs outline the existing hardware and software architectures.

### 2.2.1 Existing Hardware Architectures

Computer technology is present in all organizations worldwide. The four main hardware architectures, mainframe/minicomputer, client/server, LAN/WAN based systems and standalone PC's are used as the underlying base for software systems.

In the 1960's, mainframe computers were the first commercial use of computer technology. These are huge machines that fill rooms with equipment and are very expensive. Only skilled programmer can handle them and business users have to rely on these people. At the beginning of 1970, the minicomputer evolved. They are more affordable than the mainframe systems, fill departmental computing needs and can be handled by the departments. Despite that, minicomputers are not designed to integrate with other systems in the organization and, therefore, business users are still dependent on the IS professionals to deliver their information.

In 1980, the PC came to market which allowed end user computing and spreadsheet technology. The PC as a standalone system is still used in some offices.

The network computing in the 1990's connected the standalone PC's to a more complex infrastructure which allows file and resource sharing over LAN and WAN. From the network computers another paradigm, the client/server architecture, evolved. This allows the user to store the data at one or more central points, the servers, and access the data with intelligent clients.

The LAN/WAN architecture includes access over the Internet. Along with that, a new computer system is introduced, the thin client or NC (Network Computer). This machine fills the gap between the "dum" terminal and the intelligent clients. It allows Internet users to download data and applications over the net onto the NC.

The next table shows the advantages and disadvantages of these hardware platforms.

| Architecture | Advantages | Disadvantages |
|---|---|---|
| Mainframe/ Minicomputer | • Reliability<br>• Multiprocessor with high CPU performance<br>• Trained staff is available in organizations already | • Scalability only possible within the system architecture<br>• Highly expensive<br>• Needs much space<br>• Highly trained staff needed<br>• Older systems tend to demand more and more maintenance |
| Client/Server | • Scalability<br>• As a whole, the system is reliable<br>• Inexpensive hardware at the beginning<br>• Architecture can be upgraded easily<br>• Data can be highly distributed | • Reliability only ensured through data redundancy<br>• More staff is needed within a distributed environment |
| LAN/WAN | • System structure easy to build up<br>• Scalability within the network<br>• Inexpensive<br>• No special trained staff necessary | • Fragmentation of data<br>• No data consistency<br>• Data redundancy<br>• Difficult search for data |
| Standalone | • Easy to use even for non-technical people<br>• User errors are mainly harmless and do not effect the other systems | • Fragmentation of data<br>• Data redundancy<br>• No reliability<br>• No data-sharing |

Table 1: Advantages and disadvantages of hardware platforms

### 2.2.2 Existing Software Architectures

Today, all companies use some kind of software for all kinds of business work. The focus in this chapter is on legacy software. The next diagram gives an overview of how legacy software is seen in this work.

| **Database** | | |
|---|---|---|
| **Database Management System** | | |
| **Transaction Systems** | **Middleware** | **Enterprise Resource Planning** |
| **Rollback, Backup, Recovery, Logging** | **Message Queuing, API, NotesPump** | **SAP, JD Edwards, Baan, Peoplesoft, ...** |

Table 2: Legacy systems within the enterprise software

The shown database layer represents the existing databases in enterprises. They can be relational, sequential or object oriented, whereas this work concentrates on relational databases. The main advantages of these databases are the reliability, the huge data storage capacities and platform independence. The structured architecture also supports data consistency by means of certain rules[7] as well as fast data access. Relational databases are mainly used for mission critical data storage. On the other hand, relational databases have disadvantages such as the capability of storing only structured data and the difficulties in building dynamic databases. All in all, relational databases represent the standard data storage system and is one of the crucial points in setting up enterprise software systems.

The database management system layer indicates the system that takes care of the underlying database. Just as the database, relational database management systems are very important for enterprise software architectures. Components of DBMS's are SQL capacities, DDL compiler, DML compiler, etc.[8]. The system implies a strong reliability, consistency and fast data access.

Based on these layers, different systems are placed on top. Referring to legacy systems, this work means transaction systems such as IBM's CICS or IMS. Transaction systems

---

[7] Please refer to Ramakrishnan 1998, Connolly 1997, and Buckley 1986 for more information about relational databases

[8] Please refer to Kemper 1996 page 26 for more information.

handle the data in the databases which means it changes the data in the databases. There are certain properties of transaction systems which are known as the ACID properties[9]:

- Atomicity

  When changes are made in the data, they are completed either 100 percent or not at all.
- Consistency

  A transition is always correct. Before and after the transition is done, the whole system is in a stable status.
- Isolation

  Each transition runs by itself, even though more than one can run at the same time.
- Durability

  After the transition is completed, the changes are error-free.

To fulfill these properties, transaction systems contain different components:

- Rollback to reverse the changes that are already done
- Backup to keep an image of the original data status
- Recovery to reposition every system component to a known stable state
- Syncpoint prevents changes to the data before the transaction is complete

Transaction systems offer important advantages which can be used in enterprise critical operations. Of these, reliability and consistency are the most important, but high volume data throughput and high speeds are also reasons why transaction systems are one of the most important components in today's enterprises. In fact, large scale computer systems can not work without them. By way of contrast, transaction systems are process- and not business driven. They are primarily build for fast data changes with high volumes. They can be accessed only through a programming language and are certainly only manageable by skilled staff.

Middleware serves mainly two purposes: First, it is used as a gateway to other DBMS’s, second, it provides a standardized access to multiple applications for field mapping. How legacy systems and groupware environments are integrated and are able to exchange data is explained in the next paragraph.

---

[9] IBM Redbook, October 1997, p.24

Enterprise Resource Planning systems such as SAP R/3, JD Edwards, Baan or Peoplesoft are set upon a database system. Please refer to [10] for more information.

All software systems rely on one or another of the given hardware platforms. For example, SAP R/2 is a mainframe system, whereas SAP R/3 uses a client/server architecture. Relational databases are available for mainframes, client/servers and even standalone PC's. Within this software architecture, legacy systems can be attached to mainframes as well as client/server systems.

The main disadvantage is that legacy code inhibits changes because it was build for processing efficiency and not reuse[11].

### 2.3 Combination of Both

With the beginning of this work, many software producers started to introduce ways of combining groupware with legacy systems. This happened in part because of company mergers (i.e. IBM acquired Lotus in 1996), but primarily because of business demands. Legacy mainframe systems no longer constitute the only computer technology within an enterprise. Other systems, explained in the previous paragraph are broadly seen throughout the organization. To integrate these systems is one of the major goals of the computer industry in the late 90's.

Combining groupware and legacy systems often leads to a simple but effective structure: groupware serves as the frontend whereas legacy systems are used as the backend. This structure is supported by legacy systems, which are seen as the extensive data storage system with the popular properties such as data capacities, high data volume throughput, fast transfer and reliability. Groupware platforms on the other hand are flexible, dynamic, adaptable, easily customizable with a modern and friendly user interface (especially for non-technicians). Business driven components such as replication, document management, Internet connection and e-mail's are additional benefits of groupware platforms.

---

[10] Clewett 1998

[11] Please refer to Beth Gold-Bernstein 1998, page 5

### 2.3.1 Advantages

Combining legacy systems and groupware platforms provides the advantages of both systems and eliminate the disadvantages encountered when using these systems separately. This means that mainly the large data storage capacities and fast access of the legacy systems support the groupware components like e-mail, replication, mobile computing, data security and Internet capabilities, but also features like graphical user interfaces and semi-structured data storage are available though the combination.

In addition, instead of re-engineering the whole system architecture, the combination maintains the underlying data model, which therefore reduces error proneness and allows a continued usage without the need for additional staff training. Using the well established industry standards - the best-of-breed - provides a lot of backbone and training support.

### 2.3.2 Ways to Connect Groupware Systems to Relational Databases

There is a list of ways to connect groupware platforms and relational databases. They all differ in transfer speed, user friendliness, usage sectors and prizing considerations. The next parts show the different connection types and lay out the benefits and disadvantages.

**Simple SQL statements**

This is a very simple way for ad-hoc queries into relational databases. The SQL query has to be entered in a special file on the user client and the results are displayed in another text file. The client needs ODBC[12] drivers for the relational database as well as special operating system dependent DLL's for the connection.

A sample of the text file including the SQL statement is shown below:

*Customers.qry*
*CONNECTION_NAME= AS400*
*QUERY= SELECT name, firstname from enterprise.customers*

[12] ODBC is a standard interface for access to relational databases. Dependent on the operating system and the database that is intended to be accessed, a different ODBC driver needs to be installed on the client. Please refer to Geiger 1995

In the example above, the connection database containing information about the data source, user ID, and password (with the option to be encrypted) is located on the local workstation rather than on a Domino server. AS400 is the name of the ODBC client connection to the relational database. The SQL statement selects particular columns from the table CUSTOMERS in the library ENTERPRISE. Once the QRY file has been created, the user starts the query manually and retrieves the results in the text file.

Simple SQL statements are easy to use and a fast way to access data from relational databases for ad-hoc queries. It is not intended for programming or difficult search queries. Since it is only usable with client computers through ODBC access, it is not useful for applications on servers.

**Programmed Key Queries**

Groupware applications support the use of simple programming languages, which are specifically designed for document databases. It is mainly used for macros that automate tasks in the databases such as validation formulas or field updates. These languages contain functions and commands that allow the database designer to build the formulas.

There are different functions that allow the access to relational databases:

- Column return:

  This function returns a list of values of one or more columns of the relational table. There are no SQL statements necessary and it is mainly used for keyword field values.

- Key return:

  This function returns a list of values of one or more columns of a relational table based on a key. This key is compared with a specific column in the relational table and returns values from all matching records. Instead of SQL statements, the key is used to select the data.

- SQL statements:

  This function allows the use of SQL statements. The statement itself can be build on certain criteria's in combination with if-functions.

Sample code for a key return function in Lotus Notes:

*@DbLookup("ODBC":"Cache";"AS400";"Sebastian_Risse"; password; "Enterprise.Customers"; "Products"; "Lastname";Product)*

This sample returns the values in the LASTNAME column out of the CUSTOMERS table located in the ENTERPRISE library where the value in the column PRODUCTS contains the same value as the Notes field PRODUCT. The username SEBASTIAN_RISSE and the password PASSWORD are used to access the data.

The main advantage is the easy programming technique, which allows the user to access relational data on Notes field level. With the column and key return functions the user does not need SQL programming skills. These functions are read-only. To manipulate the data in the relational database, SQL statements are needed. Another disadvantage is the necessity of ODBC drivers when using these functions in the groupware application.

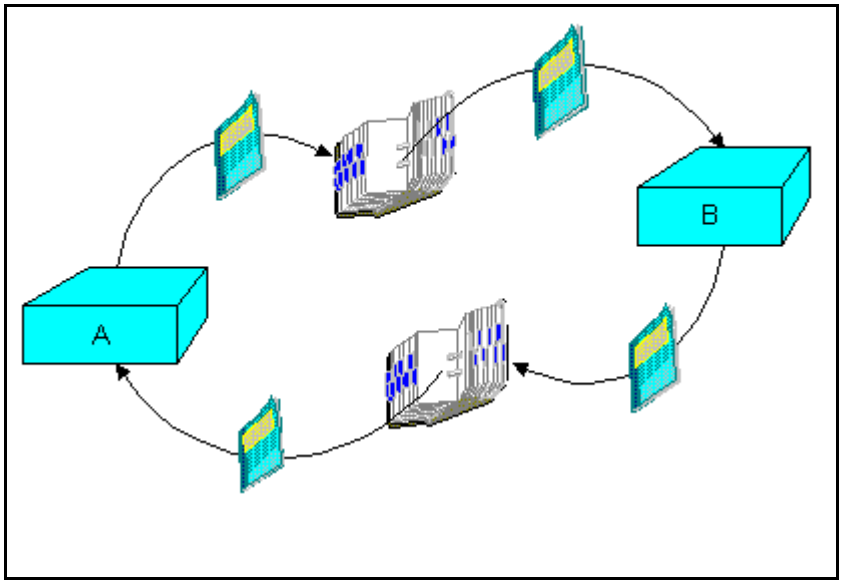
**Standard Programming language supported by SQL statements**

Programming languages such as C, C++, Java or Visual Basic support SQL statements and allow a very complex connection to relational databases from the groupware environment. The programming language consists of special Classes through which the programmer can access the groupware objects as well as the relational database connection objects such as the connection, the query or the result set. Once the classes are set and the SQL statements are executed, the result sets allow access to the relational records and the full integration to the programming language.

The main advantage is the possibility to build very complex queries as well as data manipulation through the full range of database objects. On the other hand, ODBC drivers as well as a initial connection setup has to be installed before use.

**Message Queuing**

Message Queuing enables applications to send data through queues to other applications or data sources. Message queuing is a method of program-to-program communication.




Picture 2: Message Queuing

In message queuing, data (messages) are being transferred from application A via a queue to application B. The transfer back runs via a different queue.

The main advantage of this system is the reliability, since message queuing is one of the safest way to transfer data. Other advantages are the speed, platform independency and the possibility to link into other applications, specially transactional systems.
On the other hand message queuing needs much programming and, since it is an extra application, uses additional resources and skills.

**Direct Data Transfer Applications**

Third vendor applications are often used for large data transfers. They allow the integration of enterprise wide data into any specific system. End users and programmers work only in predefined forms to set up the data transfer between the standard application and the relational databases. These transfer applications are updated regularly to make sure that they integrate with any existing database system.

The advantages of third vendor products are first of all the speed. This is often of the fastest ways to move data between a groupware application and relational databases.

They are designed for high volume data transfer at a frequent schedule. There are very few programming skills necessary and the products are mostly up to date.

On the other hand, an additional application is needed which results in additional costs for hard- and software and extra skills are needed. The usual asynchronous data transfer can be a disadvantage as well.

**Real Time Connection Services**

Real time connection application transfer data between the groupware application and the relational databases on the fly. This means that only the information that is needed at a time is shown to the user and than transferred back to the external database including the changes made by the user. For the end user, this process is transparent, so that he/she does not care where the data comes from or goes to.

Advantages of these applications are the flexibility, no programming needed, the speed of data access and the event-driven architecture, which allows a very complex model of data exchange.

One of the main disadvantages is that these tools needs much of CPU and memory resources as well as additional cost for the third party product.

The next table gives an overview of the ways to access data from relational databases through Lotus Notes:

| **Name** | **Method** | **Advantages** | **Disadvantages** | **Where to use** |
|---|---|---|---|---|
| Simple SQL statements | Display simple SQL results in a text file | • Simple to use<br>• Free of charge | • SQL needed<br>• Not for complex programming<br>• ODBC needed<br>• Extra file needed | • For end users<br>• simple, ad-hoc queries |
| Programmed Key Queries | Key or SQL selection through ODBC | • Easy to progamm<br>• Can be used anywhere in the application | • ODBC required<br>• Programming skills required<br>• slow | • For keyword lists<br>• Lookup by key |
| Standard Programming Language with SQL support | Programming language based SQL statements | • Allows complex queries<br>• Fast<br>• ODBC required only if code runs on a client | • ODBC required if code runs on a server<br>• Programming language and SQL required | • In complex macros<br>• If more than one row has to be returned |
| Message Queuing | Standard message queuing for transactional access | • Fast and reliable<br>• Data transfer and application connection | • Additional application needed<br>• Programming skills needed<br>• Not for mass data movement | • For transaction systems<br>• secure transfers |
| Direct Data Transfer Applications | Additional application for high volume data transfer | • Very fast with high throughput<br>• Only basic SQL skills needed<br>• Different transfer activities<br>• No ODBC required | • Additional application at extra costs needed<br>• Only scheduled or event driven transfer<br>• Vendor dependent | • For high volume data transfer<br>• Replication<br>• Scheduled transfer |
| Real Time Connection Services | Additional application for realtime data access | • Very fast<br>• Transparent to the end user<br>• No ODBC required<br>• No programming required<br>• For the web<br>• Very flexible | • Additional application at extra costs needed<br>• Uses much CPU and memory resources | • Activated through document events<br>• Gets the data automatically into groupware document |

Table 3: Connection between groupware and relational databases

# 3 Concepts and Methods

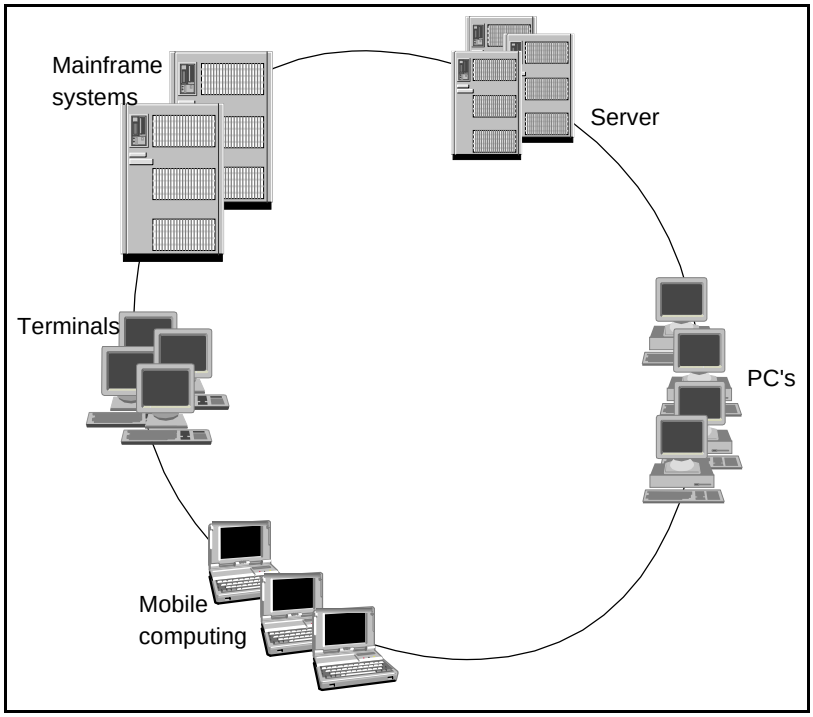
## 3.1 From Host Based Transactional Systems to Process Driven Groupware Applications

This chapter demonstrates a step-by-step change in the software architecture within an enterprise. Starting with host based transactional systems, the disadvantages of the system are shown and eliminated by the development of more complex systems including groupware and workflows, which leads to process driven groupware applications.

### 3.1.1 Distributed Computer Architectures

The shift in the computer technology[13] lead to a distributed, heterogeneous system, including all kinds of hard- and software. Through approaches such as Business Process Re-engineering (BPR) the whole architecture became even more complex and is for single users impossible to overlook. The systems in this network of computers include large storage systems such as relational databases on mainframes and hosts, terminals for the end user to access the data, legacy systems including transactional systems, client/server applications, networked PC's for file sharing or Internet access tools (browsers), standalone PC's for single applications such as word-processing and spreadsheets.

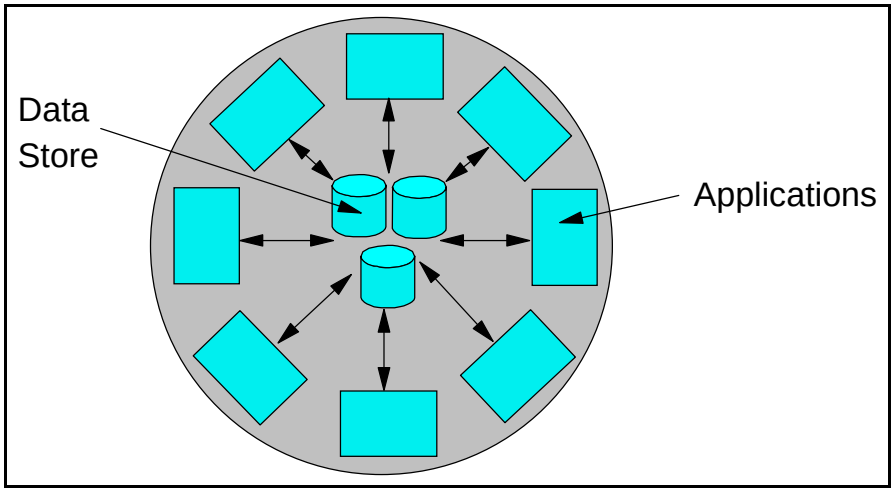
[13] As mentioned in chapter 2.2.1: From the 60's to the 90's




Picture 3: Hardware and software architecture in enterprises

This diagram does not show the relations between the components nor the connections to external models such as Internet, Extranet or WAN.

The shown components act mainly alone, every part represents an enclosed world for itself. Data transfer between the components is rare or does not exist, applications are bound tightly to the data model, which causes data redundancy and no integrity. The existing enterprise data is fragmented throughout the whole network and can not be shared, which leads to data ignorance. Departments in an organization do not know that data they need exists in other departments or they do not know where it is and how to access it. This scenario can be modeled from departments down to workgroups. This case is seen relatively often in mainframe based transactional systems. Applications and data are tightly bound together, the data can only be accessed through the given applications on top of it. These applications are not build for flexibility or user friendliness.

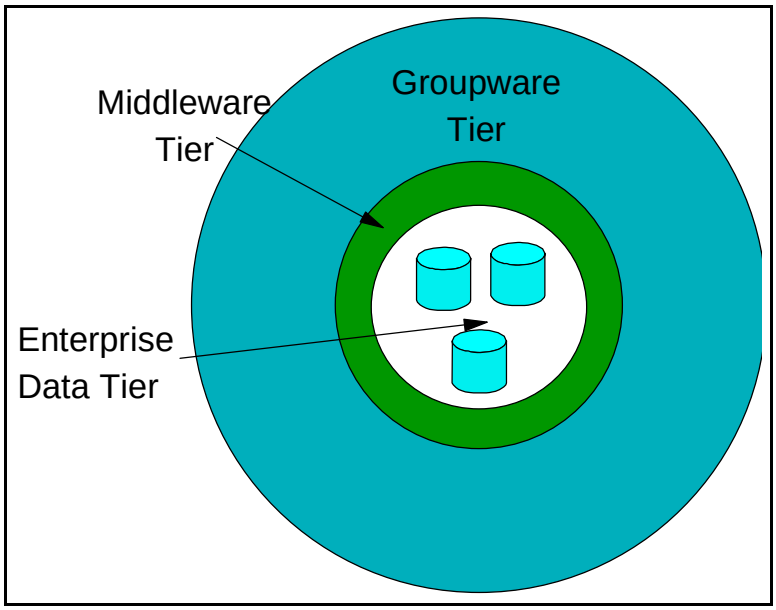



Picture 4: Host based transactional systems

Host based transactional systems can be seen as an enclosed world.

### 3.1.2 Multi Tiered Host Based Groupware System

To separate the data from the applications and to tear apart the enclosed world, groupware platforms can be used. Groupware is more user friendly than legacy systems and offer a customizable front-end that links into relational databases via middleware. This allows to make use of the crucial data in the enterprise out of the relational database system to groupware.




Picture 5: Multi tiered host based groupware system

During the creation of this work, many companies started to build this kind of architecture. Groupware is common in most corporates and the pure use of groupware does allow a lot of flexibility but makes no use of the existing data and data structures. Groupware is mainly used for e-mail, discussion databases or workflow, but without any link into the relational data. This starts to change and middleware is an important component in the enterprise software system. Please refer to chapter 2.3.2 for ways to connect relational databases to groupware.

This architecture promises a lot of advantages. The legacy data in the relational databases can now be used in the modern groupware system with most of it's advantages. The "old" data is not static any more but flexible in the groupware environment. It can be e-mailed, replicated to mobile offices, displayed and modified through the Internet or through a user friendly interface build on a client. Another profit the companies gain is out of the groupware point of view: the groupware system in an organization is extended to the use of relational databases and the mass data storage system. So far, groupware systems have a restriction of data capacities, but are also known to be lavish with CPU and memory resources. In this system, groupware data can be stored for archiving in relational databases and be transferred back into groupware systems whenever they are needed.

Despite that, this architecture implies a lot of disadvantages. The common middleware used here is asynchronous data transfer. Data out of the legacy systems are transferred into groupware on a scheduled or one-time base and then becomes information used for sharing and distribution[14]. When the information mass exceeds a certain level, the data is compacted and pushed into the underlying storage system. The problem that arises here is the question which data shall be stored in the groupware system and which in the relational data. Unfortunately the right data is often in the wrong system, which leads to the known problem of fragmented information. This work tries to eliminate this problem in the next chapters.

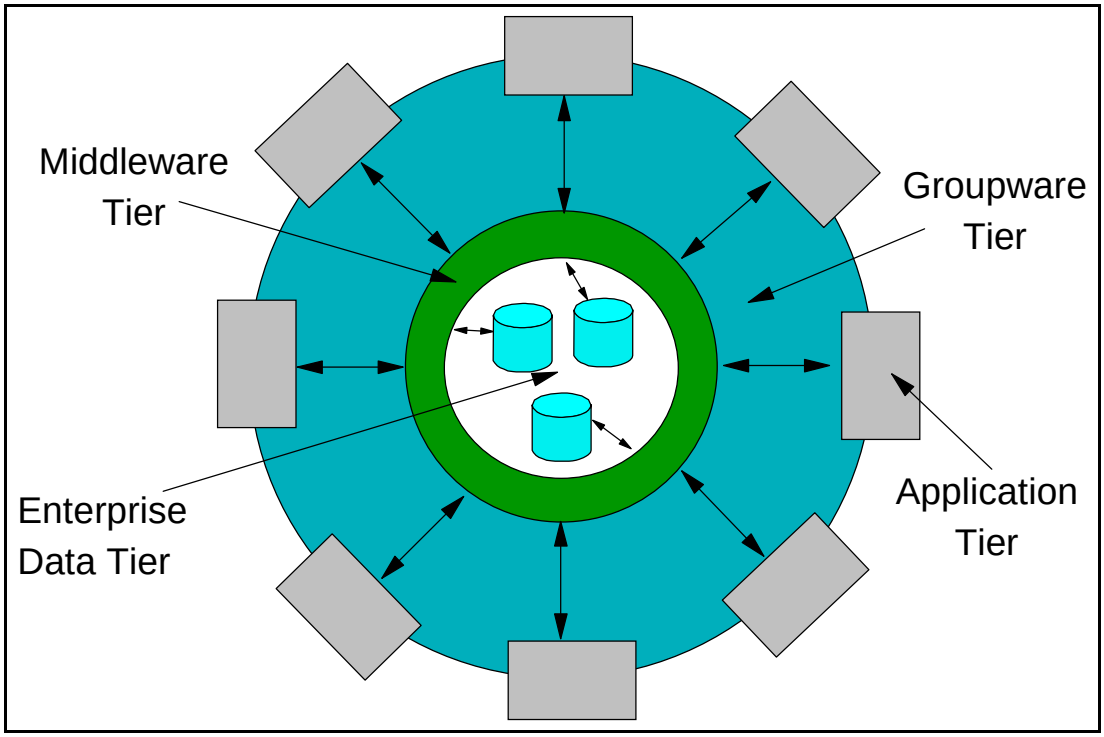
Another problem lies in the fact that not all systems are integrated here. Transactional systems are a main component in today's enterprise data systems but are not connected with groupware applications. Legacy data specialists complain about the carefree transfer of unstructured groupware data into relational systems which can lead to

[14] Through the groupware layer.

inconsistency, which is hard, troublesome and consequently expensive to solve. What is needed here is a full integration of legacy data, legacy systems, additional applications and groupware.

### 3.1.3 Multi Tiered Application Architecture

To connect all applications and data of the enterprise to a complete system, groupware provides the cohesion. Groupware acts as the underlying base for an enterprise wide data integration which includes legacy systems, distributed and heterogeneous components. This way, not only the relational databases need to be connected to the groupware tier, but also the connection to the application tier needs to be realized. The link between data and application looses and groupware steps in between.




Picture 6: Multi tiered applications architecture

The fact that the applications are loosely coupled to the data model via middleware implies that the data can be used anywhere. Applications access the same set of data which allows to keep the data central, consistent and no or less redundant. The look from the other way around, applications can be more flexible due to standard data interfaces and since the applications do not have to carry the data with it, they can be

more lightweight and easily administrated. The advantages of groupware still apply, but are also extended to the external application tier. These applications can be accessed through the groupware tier, so that for example transactions on legacy systems can be started. This allows a much greater flexibility than just data transfer.

The use of groupware as an underlying structure that connects the whole enterprise structure gives another advantage: the ability of workgroup computing to store semi- and unstructured information in document databases is a profit for other applications as well. Since the legacy applications can not utilize the unstructured data directly, the user accesses this data through the groupware application layer and links directly into the legacy systems. This allows a very flexible access of the non groupware applications by using unstructured information. These information can lead to the legacy applications by providing the end user with e.g. status reports or other data.

In contrast to that, this model consists of some problems that need to be considered: Building the above system would mean very much hard coding of the business rules. Field mapping, access levels, timing of data transfer and storage options, to name just a few, have to be set during the installation and changes are very complicated. The level of maintenance is very high and does not allow a great flexibility.

Security within the software systems is a major issue. There are two aspects of security: first, an access security has to be given. That means that only people or groups of people can access the data, that are permitted to do so. Secondly, data security is needed. Data security makes sure that no data is lost, so save copies and redundancy is the succession. The first part has to be integrated into the system, and has to be done at every access level. This can be through the legacy system in the old fashioned way: the use of terminal computers or through external applications or through the groupware layer. To build up a access secure system is a difficult process and causes the most administration work. The other kind of security is mainly provided through every single system, but needs to be set up correctly. This needs administration work as well.

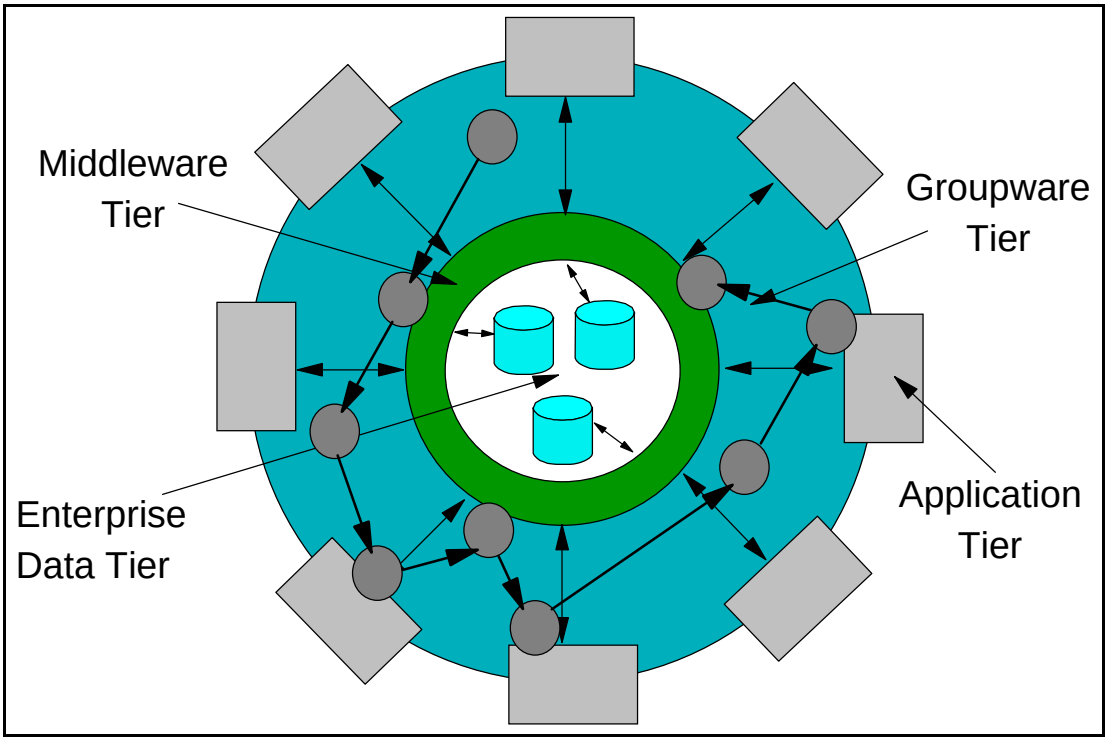
Other problems include data integrity and availability. Though this is given in the legacy system world, the combination with groupware arises more complex data structures and requires more administration and system design skills to keep up these important properties.

All in all this system provides a lot of advantages and features, but also much administration and setup work. Flexibility for the business rules is not given, due to the hard coding character of the system. When the groupware applications get more and more complex, this issue will be even more important. One solution is the use of groupware based workflow structures.

### 3.1.4 Transactional/Groupware Integration

The lack of automatically integrating any existing data source into a combined system is the major disadvantage of the Multi Tiered Applications Architecture. The large amount of administration is caused by the hard coded style of the architecture. The information that is needed for everyday tasks in an organization has to be collected in an efficient way. Business processes can be modeled to collect this information and also be able to distribute and manage the data within the enterprise structures. Business processes control the business rules for the exact data exchange between the applications, data layers and the groupware.

Many activities in a business environment can be seen as process driven. These business processes contain tasks that have to be performed by several resources. These resources can be a person, a group of persons, a computer program or a machine. The process delegates the tasks and the connected information to the people involved and picks up the completed status to rout the information to the next step. Building the processes on top of a groupware environment allows a very flexible process control with customized views and forms as well as benefits such as e-mail, replication, off line working and security. The tasks in this process contain information which has to be collected from any data source in the enterprise structure. During the flow of the process, the information is collected, updated and distributed among the different tasks. At the predefined end of a process, the status is in a stable state, which means the information is correctly distributed, conform and the data integrity is guarantied. The process needs to be able to connect into any data within the enterprise. That means that not only data transfer is from any data source to the user interface needed, but also the ability to link into OLTP (On Line Transactional Protocol) applications is needed as well. Efficient middleware transfers the data between the central enterprise data layer, the external application tier and the groupware. The existing applications also link directly into the data tier.




Picture 7: Transactional/groupware integration

The main advantage is the ability to keep the existing data and data structure, which comprises an extensive cost reduction due to the fact that no complete re-engineering of the whole system is needed. The already mentioned advantages of the standard combination of groupware and enterprise data applies to this model as well, but also extends it by many means. The processes use the existing data structures and control the data input and output. The processes support and extend the security and safety of the data layer and can act as the central data control system. Not only is the crucial enterprise data secure, but the groupware data is also secure through the controlling unit of the processes engine. The designer of the process is able to define business rules within the process itself, which means that hard coding of the data exchange is not necessary, a very flexible structure with a process driven data mapping can change these business rules at almost any time.

The data transfer consists of settings that need to be specified in the process engine. First of all, it needs to be able to link into any different data *structures*. The enterprise data is stored in a heterogeneous system based on many different vendor-dependent soft- and hardware solutions. This list of system changes frequently and is more likely to change more often in the future due to a faster growing and changing IT market. The

middleware needs to be able to link the latest computer structure to the oldest as well as anything in between. The process driven approach requires the ability to change the data source and with it the vendor specific data structure for every single task in the process.

The second vital component is the handling of different data *volumes*. To cover a largely distributed system, the process has to be able to transfer the exact amount of data to the right place at the right time. A process that is able to collect a selection of data from different systems and different locations is able to support key information distribution which allows wide area knowledge management. The needed functionality can be achieved by a data volume selection at task level within the process. Another factor here is the fact that a data volume selection allows a more effective use of mobile computing. Mobile computing is not able to handle large volume data due to hardware restrictions like smaller hard disks on laptop computers, relatively slow data throughput with modems and an overloaded Internet. Controlling data volumes in a distributed system uses the existing hardware structures more efficiently and more effectively.

The next part is different data *types*. Within the organizations, different departments produce, use and manage different kind of data. The HR department has person related data such as salary and personal information, production has production related data such as machine settings, the information can be displayed for marketing, sales, research and development, etc. Much of this is confidential data and may only be accessed by the designated people/groups. This horizontal structure of the organization has to be found in the process model. A process needs to be able to access all kinds of data to allow a maximum flexibility throughout the process. This kind of process is able to collect and distribute information between all units and departments in the organization. Just as the other requirements, the different data types must be able to be accessed at every task in the process.

The last point are the *people* involved in the process. At every task in the process the editor of the task has to be definable. Existing organizational hierarchies and access levels must be reflected in the process to ensure data security. This often results in each editor of a task being restricted to viewing only subsets of the collected data. At every step in the process, the person or group of persons should be defined to display and to modify the data. This allows a sharing of selected data to the right groups. The process itself controls the data security.

The Transactional/Groupware Integration allows a great range of flexibility. The backbone of the system, the process controls the collection and distribution of the enterprise information. At every task in the process, the data has to be controlled in different data structures, volumes, types, and with different people involved.

| | process requirements to handle | caused by |
|---|---|---|
| 1 | different data structures | heterogeneous system |
| 2 | different data volumes | distributed system |
| 3 | different data types | horizontal structures |
| 4 | different people | vertical structures |

Table 4: Process requirements

Two different kind of data transfer from the groupware process are necessary: first, the transfer between groupware and the data storage system, second the one between groupware and the applications.

The shown shift from a Host Based Transactional System to a Transactional/Groupware integration displays the advantages of each new step compared to the predecessor. Starting at an enclosed world without flexibility and data sharing, the evolution ends with a process driven and controlled distributed and heterogeneous system that is capable of integrating and controlling the enterprise data and application environment.

The next chapter gives a short overview of other approaches for this integration and shows the drawbacks and benefits in comparison with the Transactional/Groupware integration.

## 3.2 Re-engineering Legacy Applications to take Advantages of Client/Server Technology

Approaches to update legacy systems in an enterprise to a dynamic, flexible and complete IT structure are mainly seen as a complete redesign of the hardware and software infrastructure[15]. The first step is to replace the legacy hardware with newer technology which in the 90's has been client/server architectures. All data and applications have to be converted to utilize the benefits of client/server systems. The

[15] Please compare to Warren 1999, Miller 1998 and Brodie 1995 for more information about re-engineering legacy systems

same procedure has to be done for the software part. New technology like GUI and browser technology as well as new programming languages such as C++ or Java can help to build a more stable, user friendly, and accepted end user environment. It also is capable of integrating the company wide-data and preparing it for the user.

Completing the hardware and software redesign is a major effort that requires a large amount of time, money and system downtime. Companies hire consulting firms to do the redesign and to provide the skills needed for the planning, executing and maintaining a new system. Depending on the company size this takes many years or can be an endless life cycle which includes assess management, hardware and software installation and testing, data conversion, updating cycles, hiring and training of staff, etc.

Compared to that, the above approach of integrating these existing legacy systems with groupware based business processes has many benefits. First of all, the underlying data structure, which exists in the company already, does not have to be altered or replaced. That means that existing hardware and software as well as already well trained staff can be kept and be used as before. No new hardware, software or skills are needed so far. Based on the existing IS, the groupware is used as the gathering of information and layer between the data storage systems and the end user. This means that only the groupware components and the process parts have to be added to the computer structure. Groupware can be found in many organizational IT departments, so only the process engine has to be implemented. A standard process engine consists of a runtime which is in this case included in the groupware application layer and a process modeler, which allows graphical designing of the process including task definitions, routing specifications and other tools for enhancing the design and the runtime of the process. Surely, for these components special trained staff is needed.

### 3.3 The Process Engine

The implementation of the processes has to be done by special process designers. They need to be aware of the organizational structure as well as the existing processes that shall be implemented. Depending on the tool to develop these processes, the implementation itself is more or less easy. All information about the connection to the

legacy data and the legacy applications have to be entered at the process designer side. The designer has to be able to specify this connection at every task in the process.

The next chapter shows the implementation of an enterprise process modeler based on Lotus Notes and PAVONE Espresso.

# 4 The Prototype

Based on the model shown in chapter 3, this chapter explains an approach to implement a prototype build onto leading software components.

The first part gives an outline of the used standard applications. Built on that, the different approaches and techniques are explained in chapter 4.2. These approaches lead to a solution which is then implemented into the existing standard software products. The last part of chapter 4 concludes with a look at future features that are not implemented yet.

## 4.1 Existing Components

### 4.1.1 The Groupware Platform

Lotus Notes and Lotus Domino are used as the groupware component in this prototype. Lotus Notes allows easy document database design for the basic programmer and provides a very flexible application development environment for the skilled programmer. The runtime, hence the end user interface, will be mainly in Lotus Notes databases, which means that all the benefits, including compound documents, document categorization, replication, mobile computing, shared and send model, etc. are passed on to the end user. Additionally, the programmer is provided with different programming techniques. Basic functions such as form and view design, graphical navigators and agents (Lotus Notes macros) are enhanced by @Functions and @Commands as well as LotusScript and Java.

The prototype is developed for Lotus Notes version 4.6.2 and also tested with Notes 5.

### 4.1.2 The Process Modeler

Since Lotus Notes in these versions does not contain a graphical design tool for workflows, an external tool that links into the Lotus Notes environment is needed. The prototype is built on top of PAVONE Espresso[16], a complete business process design suite containing a graphical process modeler. The processes that are developed in the Espresso ProcessModeler are saved in Lotus Notes databases and automatically link into the runtime, which is part of the Espresso suite. For basic process designs, no programming skills are necessary.

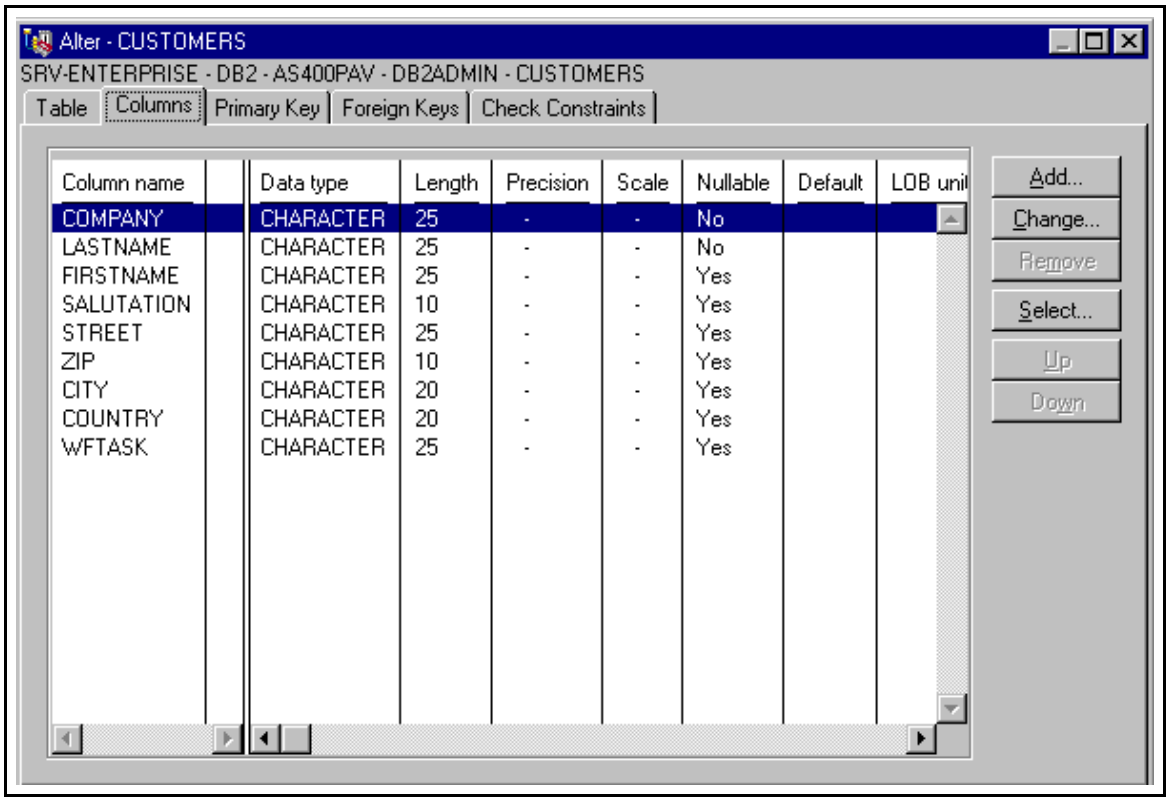
### 4.1.3 The Relational Database

The relational database is built in IBM DB2. The DB2 structure is seen in IT departments of many companies. Since 70% of the computer data is stored on IBM mainframes of the '90 and the database used is DB2, a very common relational database is used for this prototype. DB2 is available for many different operating systems such as OS/400, Windows NT, OS390, and many more.

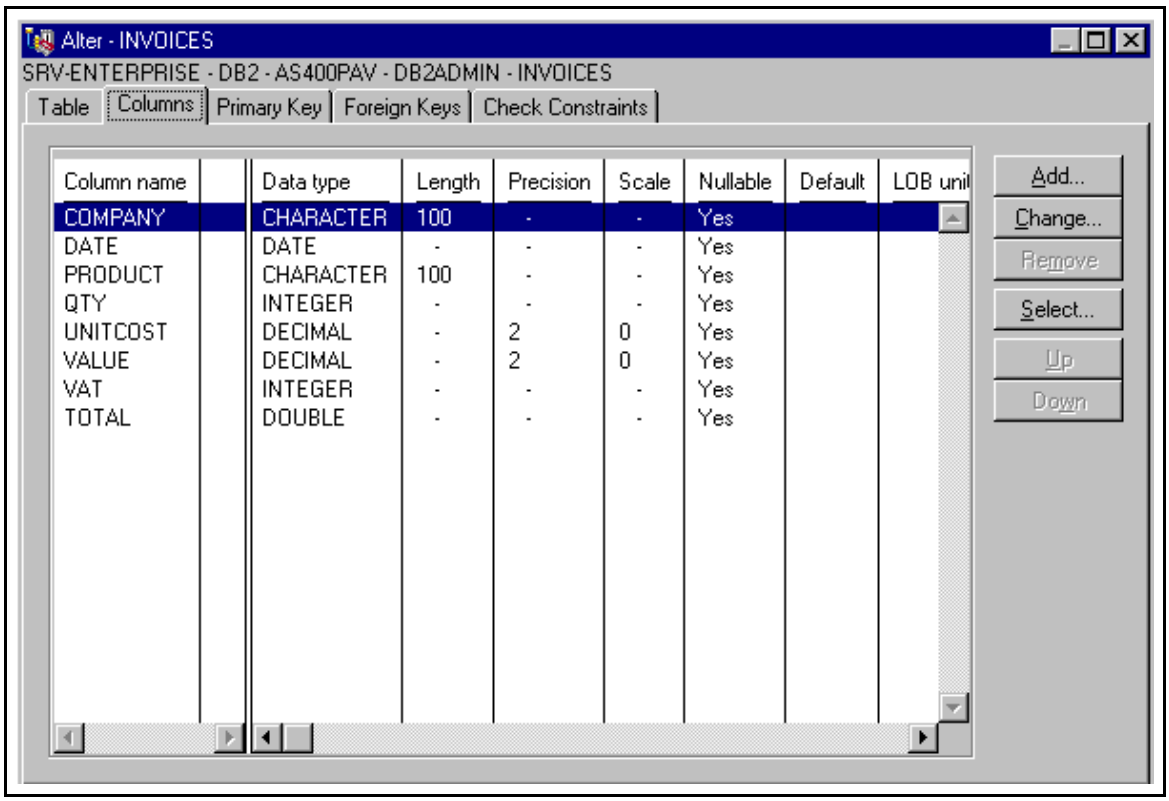
---

[16] For more information about PAVONE Espresso please refer to the Espresso on-line help

To build a test environment, two tables are created: Customers and Invoices:




Picture 8: UDB table definition: Customers




Picture 9: UDB table definition: Invoice

## 4.2 The Modified ProcessModeler

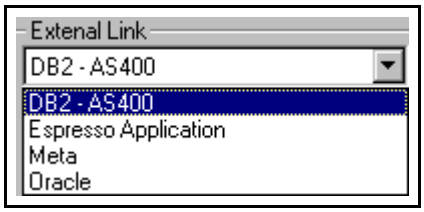
A look at the existing ProcessModeler of Espresso shows that most of the settings for each task in the process are Lotus Notes related settings. The task name, the editor, instructions and activities as well as the Notes form name can be specified for each task. The new requirement is that the link to the external relational database can be set for each task. In chapter 3.1.4 on page 25 the list for the required settings is given.

### 4.2.1 Different Data Structures

To build in the capability of integrating any data source from a heterogeneous system means that at every task the source can be specified. The information that has to be given is the type, e.g. the structure of the data source, a link name and, dependent on the link, a user name and a password.

The structure of the data source is vendor dependent, which means that the list of possible structures can change often and is probably never complete. There are, in fact, some vendor specific data structures that are used more frequently: ODBC is a very common structure, which allows an almost vendor independent approach. The ODBC source needs to be specified in the operating system and has a given name, a user name and password. Other data sources are DB2, which is used in this example, Oracle databases and SQL Server from Microsoft. They are all accessible through a database name, user name and a password. This means for the task definition in the process modeler that the process designer has to be able to enter these values to specify the data source. In a live environment, the number of external databases is usually limited, which leads to the idea that it is more user friendly for the process designer to chose the external link from a list which just shows a link name. The link name refers to the values external database name, user name and password. These values will be saved in a Lotus Notes database.

Here is an example of how the ProcessModeler user interface could look like:



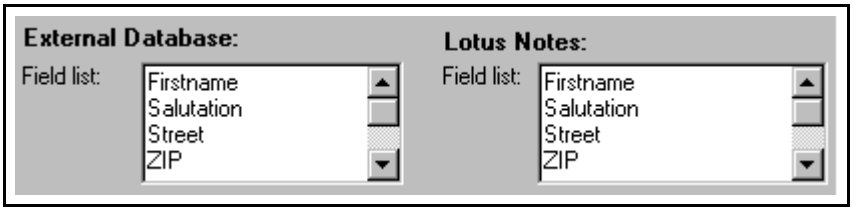

Picture 10: External link

After selecting one of the External Link names, the ProcessModeler has to pick up the values database name, user name and password from a Lotus Notes database. This will be implemented in the Espresso Runtime.

### 4.2.2 Different Data Volumes

The amount of data transferred and stored at every task is very important in a distributed system. The volume has to be specified down to field level, to be as flexible as possible. This implies that a field mapping can be given at every task of the process. Field mapping means that every field in a Lotus Notes document, that has to be linked into the relational database, can be mapped to a column of a specific table within the relational database. The effect is that the process designer can decide at every step in the process which data is transferred. Every field data in the field mapping will be exchanged with the related column.

This is an approach how it looks like in the task definition of the process modeler:




Picture 11: Field mapping

The first entry of the external database list is mapped with the first entry of the Lotus Notes list and so on.

### 4.2.3 Different Data Types

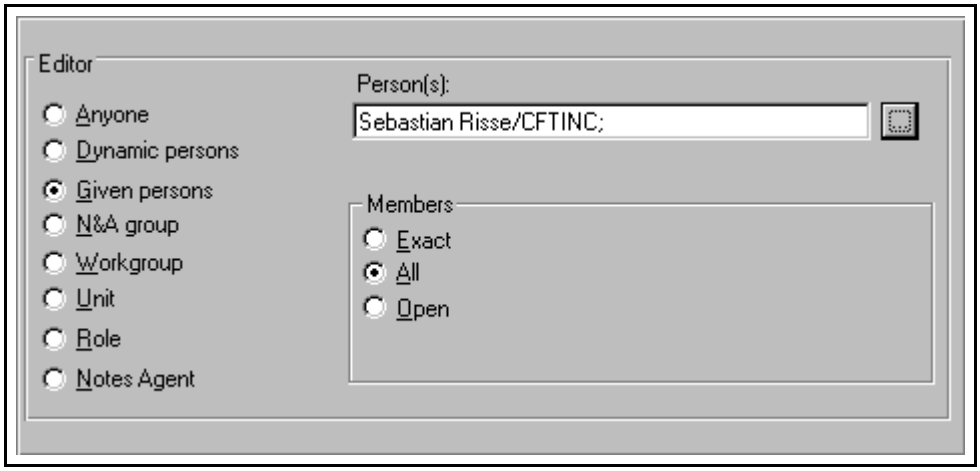
Different data types refer to the different departments within an organization. Every department stores different data that is linked for referencing. The different data types are stored in different relational tables in the enterprise databases. To specify the different tables, the process designer should be able to choose from a list of existing and accessible table names at every task of the process. For the prototype, the two tables “Customers” and “Invoices” are given, so the designer should be able to select one of these. Since this list is dependent on the relational database, the list must change if the process designer changes external link name.

The UI for this is very simple:


Picture 12: Relational table specification

### 4.2.4 Different People

The ability to specify different editors at every task is a standard feature of PAVONE Espresso:



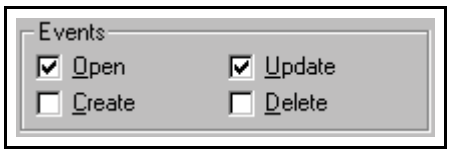

Picture 13: Task editor specification

To ensure that every person in the process can only interact with the relational data in the way they are set, the security features for the connection to the external database need to be specified at each step. The different access levels are: read, create, update and delete access. At every task the designer can declare if the data shown comes from

the relational database, if the editor is able to transfer the changes to the relational database, if the editor is able to create new records and if the editor is able to delete an external record. One approach is to control these access levels through the Lotus Notes document events:

1. If the task editor is opening the task, that is the Lotus Notes document, the data will be pulled in from the relational database.
2. If a document is saved, the changes will be made in the external record as well.
3. If a new document is created, a new record will be created as well.
4. If the task document is deleted, the referring record will also be deleted.

This allows a very simple to use UI:




Picture 14: Events

Dependent on the check boxes ticked, the external record will be read, updated, deleted or a new record will be created whenever the Lotus Notes document is opened, saved, deleted or a new Notes document is created.

### 4.2.5 Additional Functionality

Additional to the above, there are some more very helpful settings for each process task.

**Reduced redundancy:**

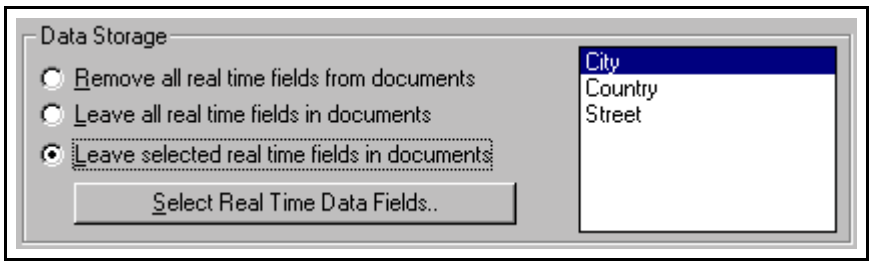
After the transfer from the Lotus Notes document, the data usually remains in the document as well. This leads to data redundancy which often ends in many conflicts. If the data is automatically removed from the document, not all Lotus Notes features can be used, due to the fact that replication and heredity will not work if the data is not kept in the Notes document. This conflict can be solved by giving the process designer the ability to choose from three options for every task:

1. Remove all transferred fields from the Lotus Notes document and keep them only in the relational database:

This will eliminate the redundancy. All the data will be kept in the relational database

2. Leave all fields in the Lotus Notes document and in the relational database:
This will allow the use of replication and data can be inherited to other documents
3. Leave selected fields in the Lotus Notes document and keep all fields in the relational database:
The designer can select from a list of fields. This list will show all fields from the field mapping (please see above). The process developer can now select all the fields that are necessary for replication and heredity. The other fields (i.e. large fields with file attachments) will be removed from the Notes documents and will save storage space and prevent redundancy.

This is a screen shot from the process task definition:




Picture 15: Data storage options

**Split workflow:**

If the process contains a document split, that is the automatic creation of additional Lotus Notes documents, the developer should be able to decide at every task if an external record will be created as well. At any stage of the process there is not necessarily an external record for every existing Lotus Notes document. A process might contain only Lotus Notes data until a certain point. Then, at a specific task, a record in the relational database must be created.

This can be achieved by a simple check box in the task definition:


Picture 16: External record creation

If the box is ticked, the runtime will automatically create an external record at the given task of the process, but only if it does not exist so far.

**Reference:**

In PAVONE Espresso, a Lotus Notes document flows through the process. In the enterprise edition, this document gets data from a relational record at every task. To reference the Lotus Notes document with the right record, a specific key is needed. A person address document for example is referenced through the persons last name and a company name. The key can be seen as the primary key of a relational table. It clearly references every entry to the Lotus Notes document and vice versa. Since the relational record can change during the process (for example: at one task the document refers to the address record in the customers table (primary key: last name and company name) and at the next task it refers to a record in the invoice table (primary key: company name, since all invoices are related to the whole company). This primary key can be one or more fields referring to one or more columns in the table. This reference can be part of the field mapping, but it needs to be a separate entry.

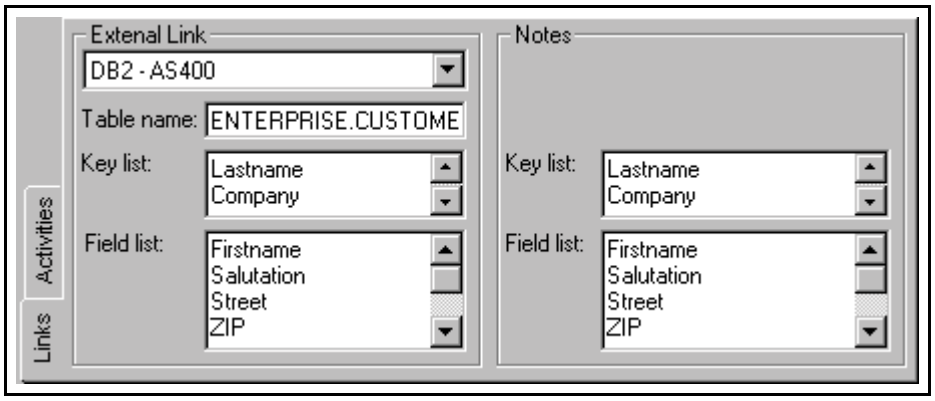
This is the user interface for the key field mapping:


Picture 17: Key list

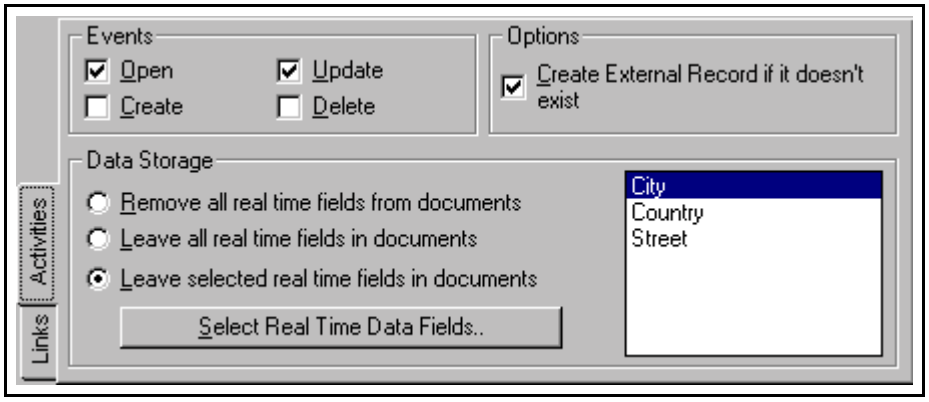
### 4.2.6 The new ProcessModeler Task Definition Tab

Adding all above components together gives a new tab within the ProcessModeler task definition which a Link and an Activities part.

This is the first draft:




Picture 18: The link settings




Picture 19: The activities settings

### 4.3 Connection between Lotus Notes and DB2

All new settings in the ProcessModeler need to be transferred to the runtime engine to build up the data exchange between Lotus Notes and the relational database. The possible connections between groupware and relational databases shown in chapter 2.3.1 need to be evaluated to utilize the best solution. The next chapters test the different ways and show sample code as well as the pros and cons.

#### 4.3.1 Direct SQL Statements

The DB2 Client Import Library is a very basic tool for using simple SQL statements entered by the programmer to exchange data between Lotus Notes and DB2. There are several reasons why this tool is insufficient:

The SQL statements need to be entered by the process designer. There is no way of building an automatic SQL generator, since no programming language is accessible. An extra file is needed to store the SQL statements, which is not acceptable in Espresso, it uses only Lotus Notes databases for storage. Another reason for not using the DB2 Client Import Library is the fact that it is very slow and it can only connect to an ODBC data source.

There is no need to evaluate this tool in any greater detail.

### 4.3.2 Programmed Key Queries

Lotus Notes makes use of @Functions as an internal programming language. Notwithstanding the things that can be done with these functions, vital features like loops and functions are missing.

There are three @Functions that can access data from non-Notes databases:

- @dblookup retrieves the value from a specific column from rows that meet a certain criteria (a key). This can be used for listing records that meet a certain criteria, for example the list of customers that purchased a specific product.
- @dbcolumn returns a whole column out of a table without any key. This can be for keyword lists, for example a list of all customers.
- @dbcommand allows the use of SQL statements. The return value is only one record out of the searched database. This function is comparable with the DB2 Client Import Library except that here no extra file is needed.

Sample code:

*@DbLookup("ODBC":"Cache";"AS400";"Sebastian_Risse"; password;*
*"Enterprise.Customers"; "Products"; "Lastname";Product)*

This sample returns the values in the LASTNAME column out of the CUSTOMERS table located in the ENTERPRISE library in the AS400 database, where the value in the column PRODUCTS contains the same value as the Notes field PRODUCT. The username SEBASTIAN_RISSE and the password PASSWORD are used to access the data.

The above formula can be build into the computed field formula for each field that gets data from the relational database. The primary key has to be kept in the Notes document

and is in this case the field PRODUCT. Depending on the settings in the ProcessModeler, whenever a documents opens, the data is retrieved from the referring external database.

The main disadvantage of these functions is the fact that they can only retrieve data, but can not send data to the relational database. There is also very much coding necessary within the Lotus Notes form. For every field the formula is needed and has to be calculated whenever the document is opened. This has an immense performance impact during the runtime. An ODBC connection in the operating system is also necessary.

This tool is not recommended due to the performance issues and the one way data transfer.

### 4.3.3 Programming Language Supported by SQL Statements

LotusScript is a programming language in Lotus Notes based on Visual Basic. It is more complex than @functions and allow integration with all data and meta data from the Lotus Notes databases. Special ODBC classes allow the transfer between Notes and relational databases based on SQL statements. The SQL commands can be more complex and are dependent on LotusScript entries.

Sample code:

```
Dim con As New ODBCConnection
Dim qry As New ODBCQuery
Dim result As New ODBCResultSet
Set qry.Connection = con
Set result.Query = qry
Con.ConnectTo("AS400")
Print "Connected to " & con.DataSourceName
defaultQuery = "SELECT DISTINCT * FROM ENTERPRISE.CUSTOMERS'"
qry.SQL = defaultQuery
result.Execute
```

This sample builds up a connection to a DB2 database called AS400 and displays the name of the connected database. The SQL statement

SELECT DISTINCT * FROM ENTERPRISE.CUSTOMERS

runs on the database and returns the SQL results into the object RESULT from where the results can be accessed.

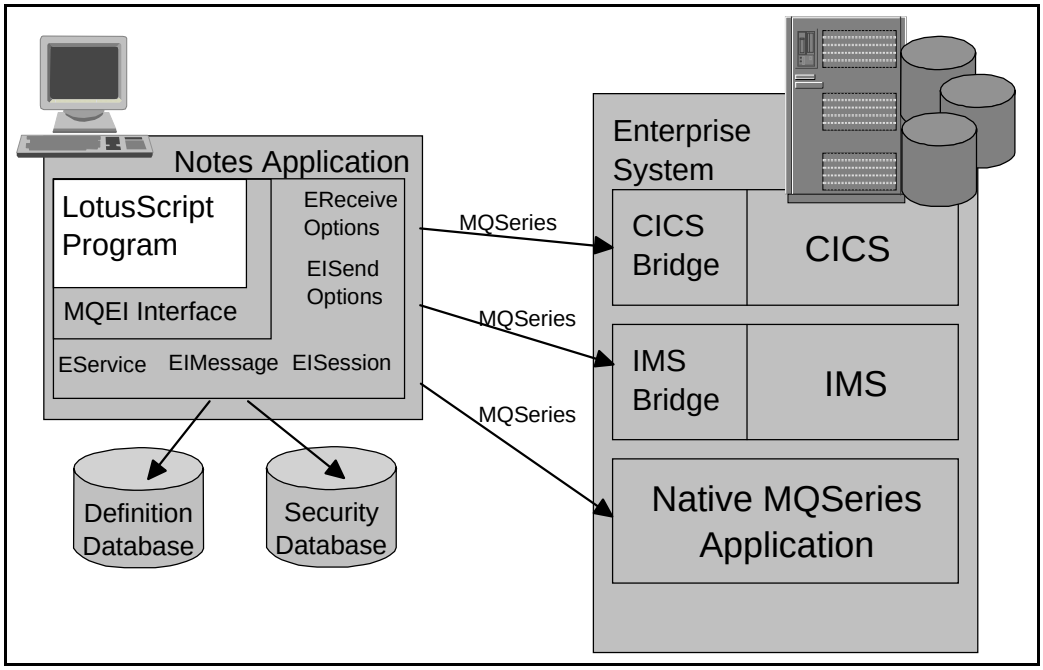
This script can be built into the open event of the Lotus Notes form and fill all fields specified in the field mapping in the ProcessModeler with the values from the relational database. A modified script can be entered into the exit event of the form and save back the data into the relational database. The same can be done with the create and delete events of a Lotus Notes database.

This is an acceptable solution with some disadvantages: All forms need to be modified to work properly. This can be done with the help of subforms and Script Libraries of Lotus Notes. The settings of the ProcessModeler still need to be transferred and be put into the code. All in all, this solution is faster than the ones before, but especially with a large number of fields unacceptable. Another problem is the need for ODBC drivers in the operating system on each workstation to ensure a proper connection.

The LotusScript code is definitively a necessary feature for building up keyword lists in single Lotus Notes documents.

### 4.3.4 Message Queuing

The MQEI is a standard tool from Lotus which provides a message queuing connection between Lotus Notes and transactional systems. MQEI is fully integrated in the Lotus Notes environment and the settings can be done through Notes databases and LotusScript classes. These classes provides the programmer with a simple interface to communicate with enterprise applications. MQEI uses MQSeries from IBM as the underlying architecture. MQSeries enables applications to use message queuing to participate in message-driven processing.




Picture 20: MQEI system structure

MQEI extends the LotusScript classes with the EReceive Options, EISend Options, EService, EIMessage and EISession classes to make use of the MQSeries. MQSeries transfers the data to the enterprise system which includes the transactional systems such as CICS or IMS from IBM. For that connection, special bridges are needed. The MQSeries can also link into the native MQSeries applications on the host. The MQEI application uses special Lotus Notes databases for data definitions and security options. This allows a central definition point which can be accessed from the LotusScript code.

The main advantage of this system is the reliability, since message queuing is one of the safest way to transfer data. Other advantages are the speed, platform independency and the possibility to link into other applications, specially transactional systems.

Notwithstanding these advantages, the MQEI needs much programming and, since it is an extra application, uses additional resources and skills.

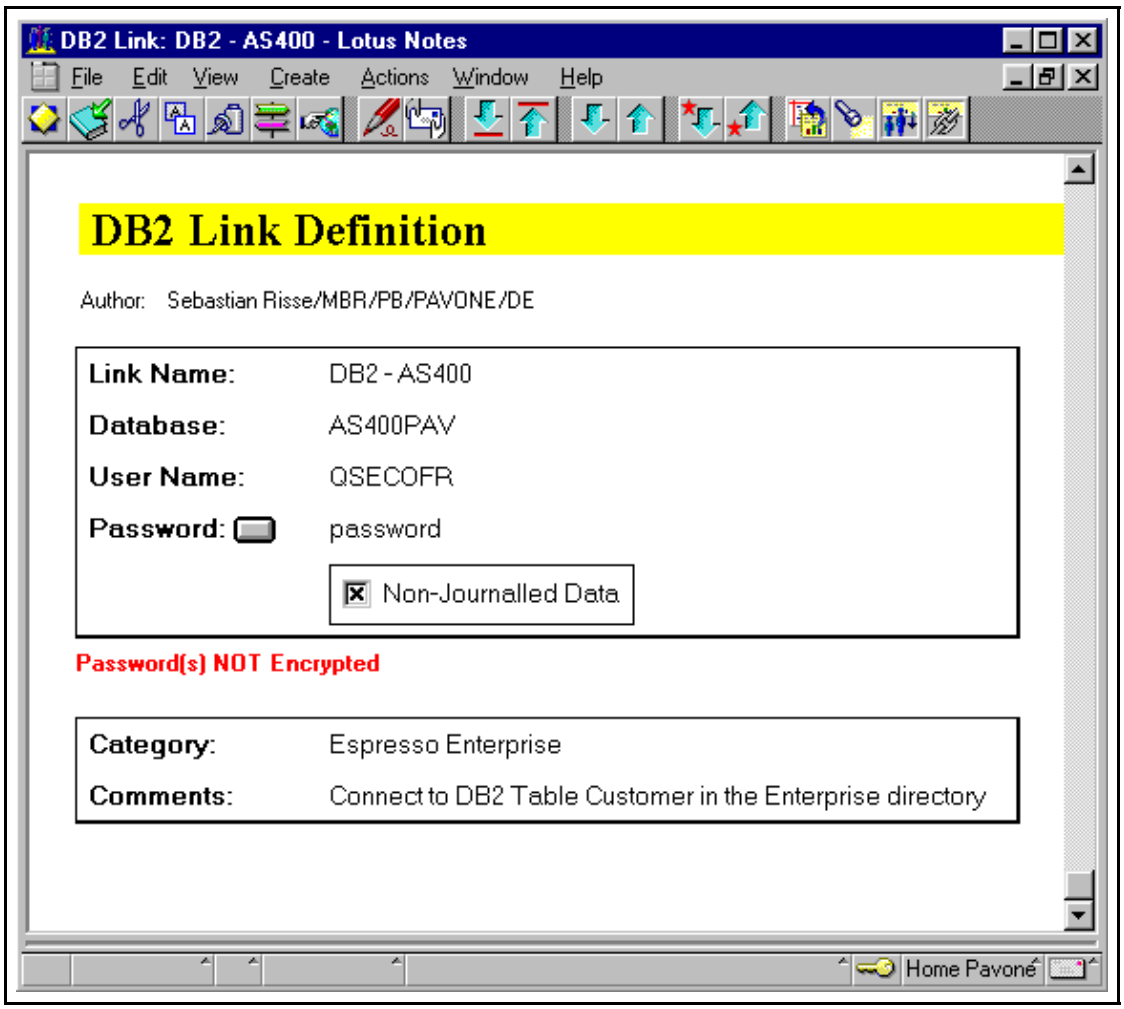
Evaluation of the MQSeries approach is very difficult, because special message queuing skills are needed to build test examples. The external database as well as the transactional systems need to be set up for MQSeries to do the data transfer. It was not possible to set up a testing environment during this work due to the lack of message queuing skills.

### 4.3.5 Direct Transfer Applications

The tested application for direct transfer is Lotus NotesPump 2.5 or Lotus Enterprise Integrator. This tool is capable of handling large amount of data at a very high data throughput.

Lotus Enterprise Integrator consists of a separate server that runs alongside with the Lotus Domino server. This server completes the scheduled transfer activities that are created in the LEI administration database.

To set up any data transfer, links need to be created. These links specify the database that shall be accessed during a transfer. The link type, a user name and a password for the non-Notes database, as well as a link name has to be entered. The link type can be DB2, EDA, File, Notes, ODBC, Oracle, Sybase or Text.




Picture 21: Link definition NotesPump

Each activity needs none, one or two links.

An activity can do the following[17]:

- Archive Activity

  Moves data from one database to the other deleting the original data as they move
- Command Activity

  Executes an action against one database, such as SQL statements.
- Direct Transfer Activity

  Moves data from one database to the other without deleting the original data.
- DPROPR Activity

  Enables propagation of IBM Data Propagator (DPROPR) Consistent Change Data (CCD) table information to any NotesPump data source.
- Java Activity

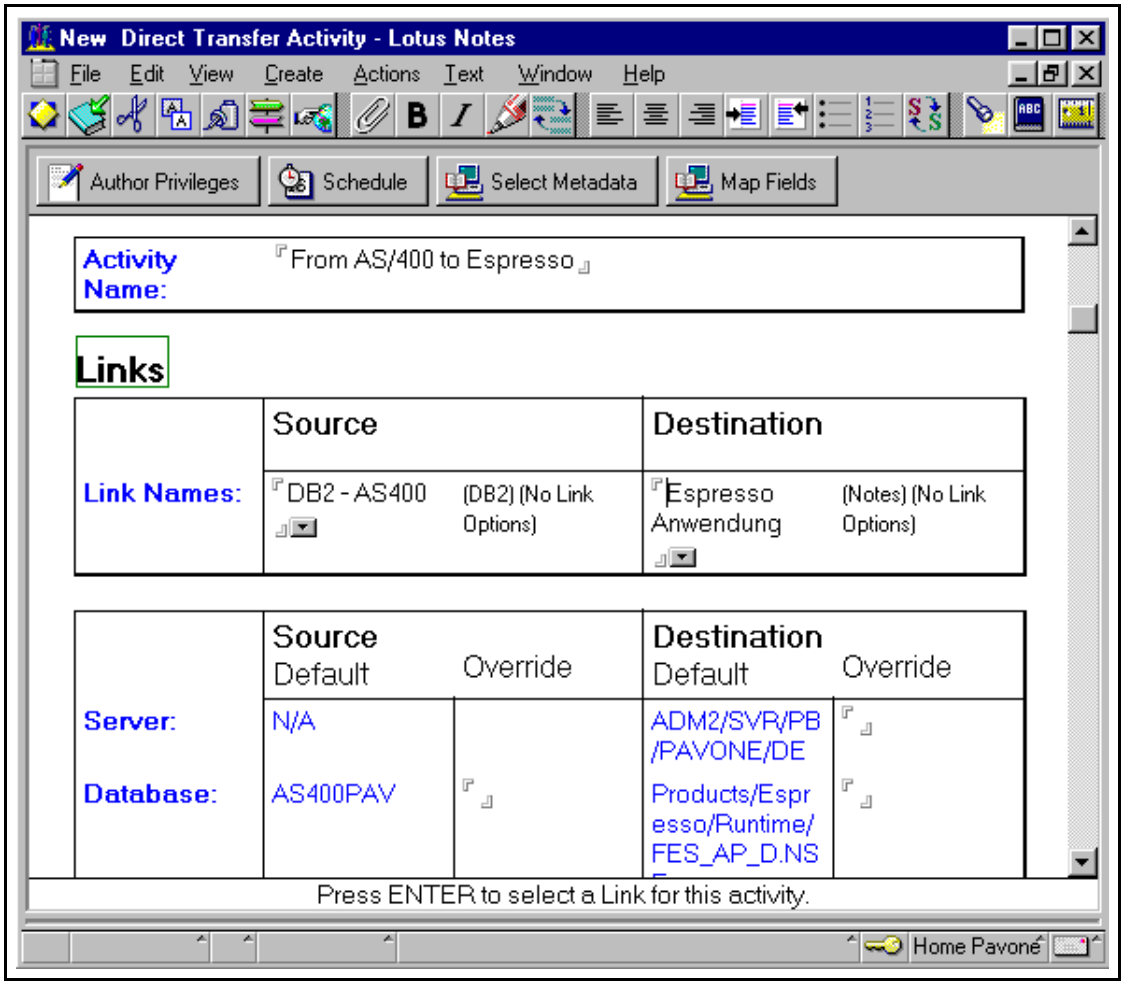
  Executes a Java application.
- Polling Activity

  Causes an activity to execute when a specific condition is met.
- Replication Activity

  Allows the synchronization of two databases
- Scripted Activity

  Executes a LotusScript. Therefore, special Script classes are available.

[17] Please refer to the Lotus NotesPump 2.5 manual for more information.




Picture 22: NotesPump Direct Transfer activity

Once the activity is created, it can run on a scheduled base (between seconds and years including exceptions such as disabled at weekends, etc.) or it runs as soon as possible. If it runs ASAP, the NotesPump server will start the activity as soon as there are no other activities in the activity queue. The activity can be set up to run either only once or on a regular basis.

For example during a direct transfer activity, NotesPump moves data from a DB2 database to a Lotus Notes database, i.e. for each record in the DB2 database a new document in the Lotus Notes database is created. This requires a certain field mapping, which can be specified in the activity document. Settings such as error handling, SQL selections and maximum number of records to transfer can be set there as well.

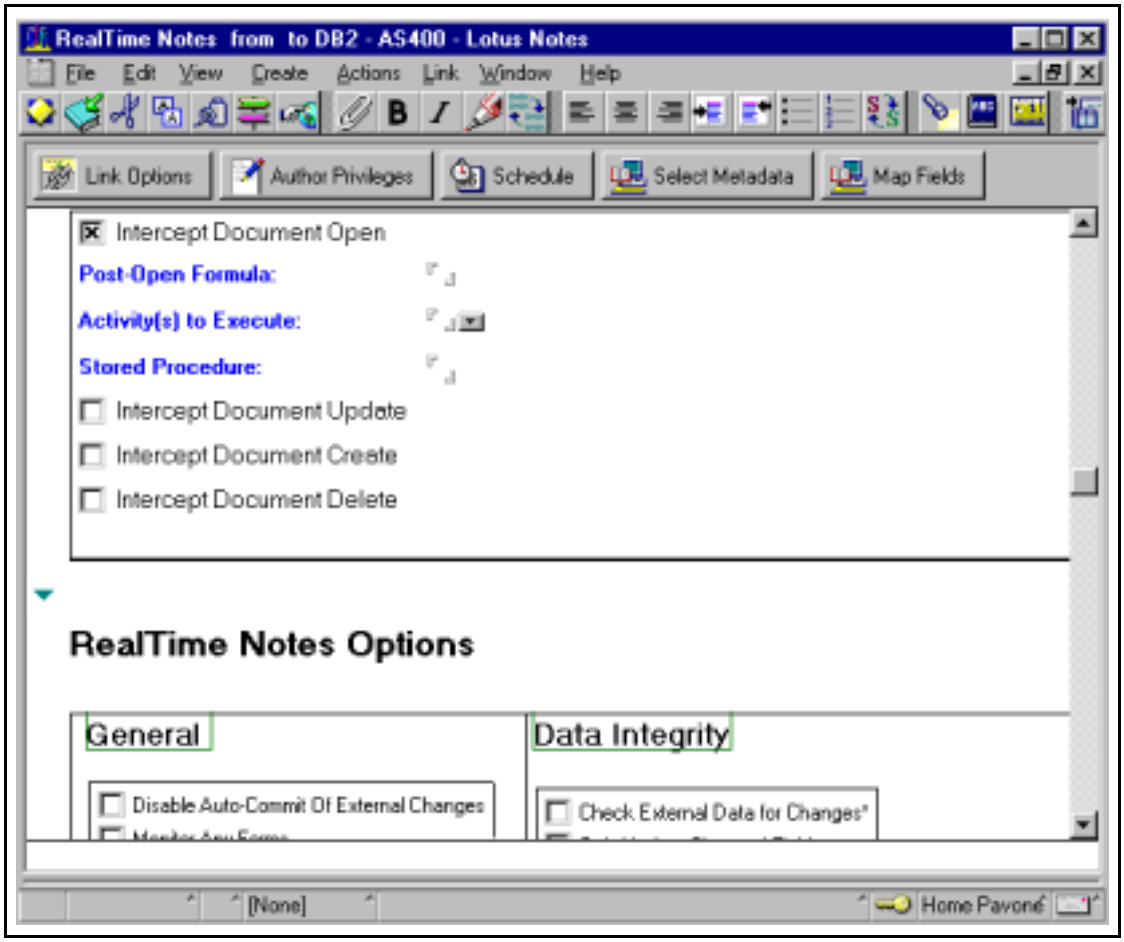
The advantages of LEI / NotesPump are first of all the speed. This is one of the fastest ways to move data between Lotus Notes and relational databases. NotesPump is designed for high volume data transfer at a frequent schedule. On the other hand, it is not profitable because of the high purchase costs of NotesPump.

For the prototype LEI is not recommended, since the data exchange is not real time. That means that the data from the relational database has to be in Lotus Notes already before the document is opened, otherwise the end user would have to wait after opening a document until the NotesPump server can handle the exchange and display the information in the document. To predict which data will be accessed next to transfer the right information before the document is opened is very difficult and could easily lead to performance problems on the NotesPump server. Exchanging the documents on a scheduled base can lead to inconsistencies and an overload of the Lotus Notes database.

### 4.3.6 RealTime Access Applications

RealTime Notes is another NotesPump activity. This activity is also known as a separate application called Domino Enterprise Connection Services (DECS) which is available at no charge with Lotus Notes version 4.6.3 or Notes 5.

DECS checks Notes documents in a special Notes database. When any event such as open, create, delete or update occurs in a document, data from a relational database interact with this document. For example, when a Notes user opens one document, DECS pulls the data from a record in a relational database automatically into the document. When the user saves the document, DECS transfers the data back to the corresponding relational record. This is transparent to the end user, since it is real time. In fact, depending on the number of fields in the Notes document, this works faster than standard Notes. This is caused by the fact that the speed at which Lotus Notes is able to open a document is related to the number of fields in the document. DECS can remove the fields from the document during the transfer to the relational record. When the user opens the document again, RealTime Notes moves the data back into the document, which is done via SQL statements. This is faster than standard Notes. Within DECS, activities set the values for the data transfer. Just like in NotesPump, links are used to determine the different databases and are chosen for the activities, where all settings can be made.




Picture 23: NotesPump RealTime Notes activity

DECS is an ideal solution for the prototype: First of all, it covers all needed settings specified in the Espresso ProcessModeler, secondly it is very fast, reliable and there is no charge in Lotus Notes versions 4.6.3 and Notes 5. DECS works through a web browser, too, which makes it even more flexible.

The only disadvantage is the large amount of resources needed on the server with increasing number of activities; but this is acceptable considering the effect and the fact that it is only related to the server, the client does not need any additional resources.

### 4.3.7 Conclusion

DECS will be used as the middleware tool for the prototype. Its performance and the fact that no programming is needed makes it the best solution so far. The LotusScript

code as well as the MQ Series will also play a major role for implementing special processes.

### 4.4 Espresso Enterprise Implementation

The first draft of the layout and the middleware component is chosen and has to be put together in one system. Therefore, a test environment is created with the following components:

- Windows NT 4.0 plus Service Pack 3 as the operating system
- Lotus Domino server 4.6.2 as the groupware platform
- Lotus NotesPump 2.5 including RealTime Notes (DECS) as the middleware
- IBM Universal Database (UDB) version 5.0 for Windows NT as the relational database
- PAVONE Espresso 4.5 international English as the process development tool

The RealTime Notes activities contain all the settings that are needed for the modified ProcessModeler. To have these activities act on the right documents of the process, one activity is needed for every task in the process. Every activity supervises all documents that are at a specific task, hence, they have a certain workflow status. If an event (open, update, create or delete) acts on the supervised document, the RealTime Notes activity controls the data transfer between the given document and the referring relational record.

The activities will be created by the ProcessModeler tool, so that whenever the process designer saves a process, the activities are automatically created and start to monitor the Espresso Application[18] database. If the process is deleted or not used any more, the activities can be deleted.

The next chapter determines the fields of the RealTime Notes activities that need to be set in order to monitor the right documents.

---

[18] Please refer to the PAVONE Espresso manual for the underlying database architecture.

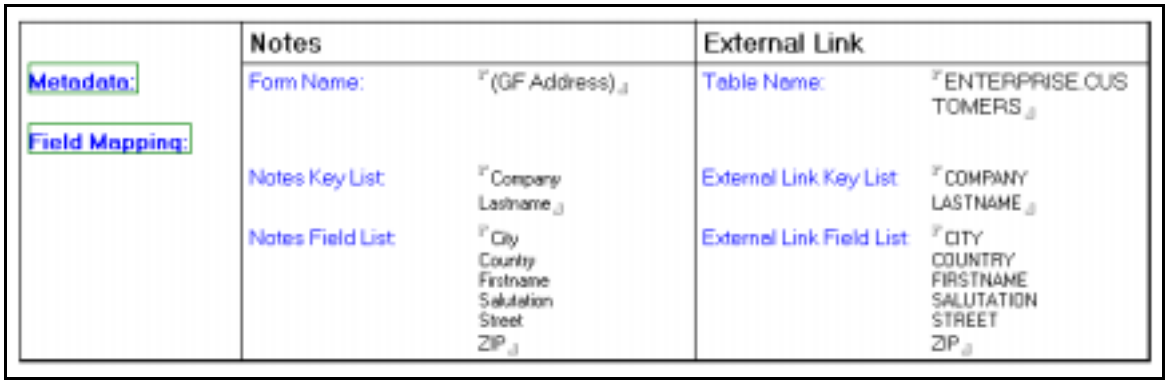
### 4.4.1 DECS/NotesPump RealTime Notes Activities

The list of settings in the ProcessModeler for each task of the process can be found in the activity documents. This is:

1. The external database link is specified in the link name fields:

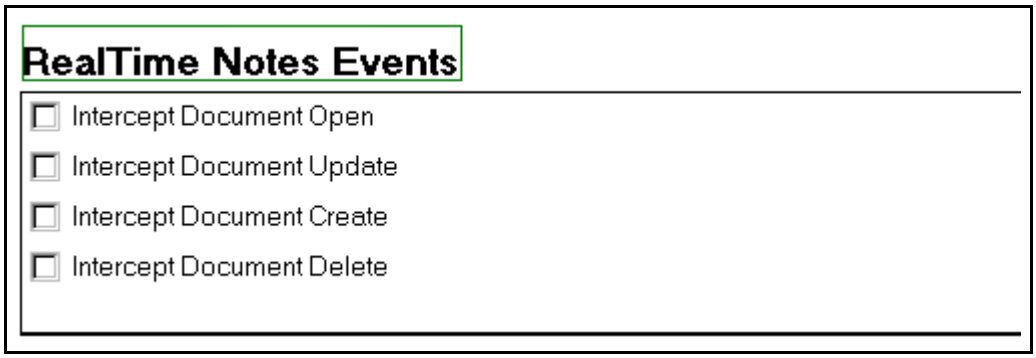

Picture 24: DECS External Link

2. The table name, field list and key list are located in the metadata/field mapping section:



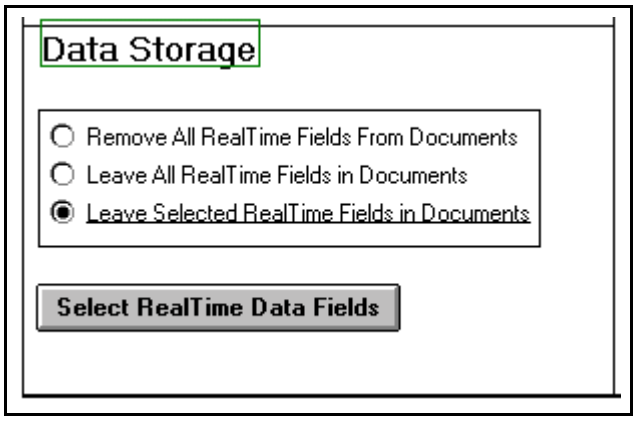

Picture 25: DECS field mapping

4. The document events are in the RealTime Notes events section:




Picture 26: DECS RealTime Notes Events

5. The data storage are in the Realtime Notes Options part:




Picture 27: DECS Data Storage

6. The option “Create external record if it does not exist” can be found in the RealTime Notes Option section as well:


Picture 28: DECS External record creation

In addition to that, the RealTime Notes activities need more fields to work properly:

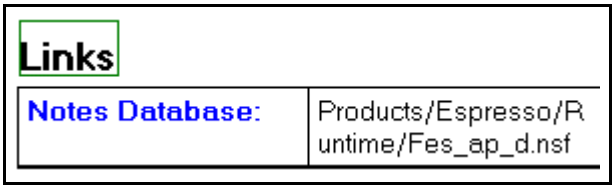
1. Name

A name for the activity. This has to be unique throughout the database and it will be referred to in the log database as well as the NotesPump server console. Since there will be an activity for each task that connects to the relational database, the activity name will be the internal Espresso task name[19]:


Picture 29: DECS Activity Name

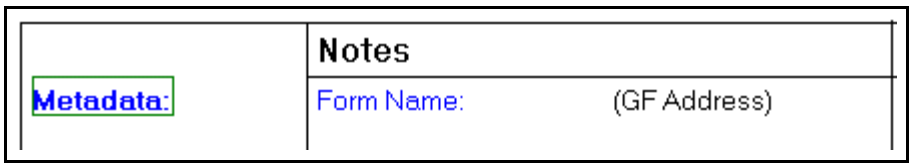
[19] Please refer to the PAVONE Espresso manual for internal task names.

2. The Lotus Notes database, which will be controlled. Since every process in Espresso is connected to an Espresso Application database[20], it is this database name that has to be specified here. The Application database is set in the ProcessModeler and will be derived from there:




Picture 30: DECS Link Notes Database

3. The Lotus Notes form name. The RealTime Notes activity can supervise all documents created with any form in one database, but a more specific way is to control only documents created with a certain form. In the ProcessModeler, this has to be specified at every task, which allows the document that is routed to be viewed through a different form at every task. The form specified in the modeler is the form that is watched by the related activity:

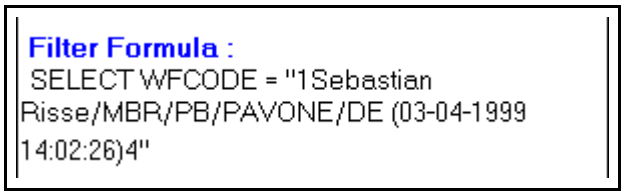



Picture 31: DECS Notes form name

4. Filter formula. Lotus NotesPump allows to specify a selection formula at every activity. This means that only documents that meet that criteria will be controlled. The prototype uses this functionality to control only the documents that are at one specific task.

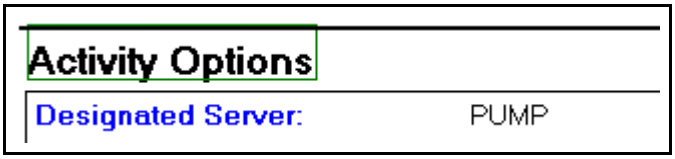
[20] Please refer to PAVONE Espresso user guide

Espresso uses the internal field wfCode for every document in the process to specify the exact momentary task of the document. The filter formula is then build as:
SELECT wfCode = “{internal task name}”:



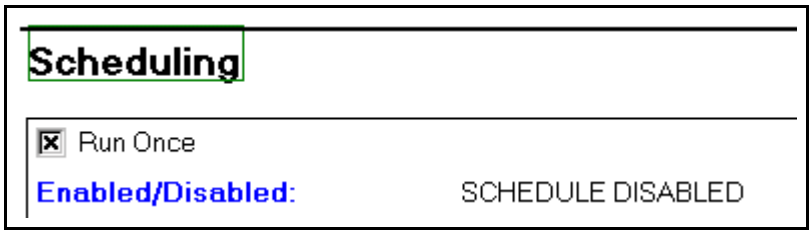

Picture 32: DECS Filter Formula

5. The name of the Lotus NotesPump server. The NotesPump Administrator database can control more than one NotesPump server. This leads to the fact that the name of the Pump server has to be set for every activity. For the prototype the name of the server is always set to “PUMP” as default:




Picture 33: DECS Activity Options

6. Scheduling: Run Once. NotesPump activities can be set to run on a scheduled base (which is important for direct transfer, replication, etc.) or they run only once, which means they start and, for RealTime activities, do not stop until user interaction or error occurs. For the prototype no schedule is set, the activities run once only:




Picture 34: DECS Scheduling

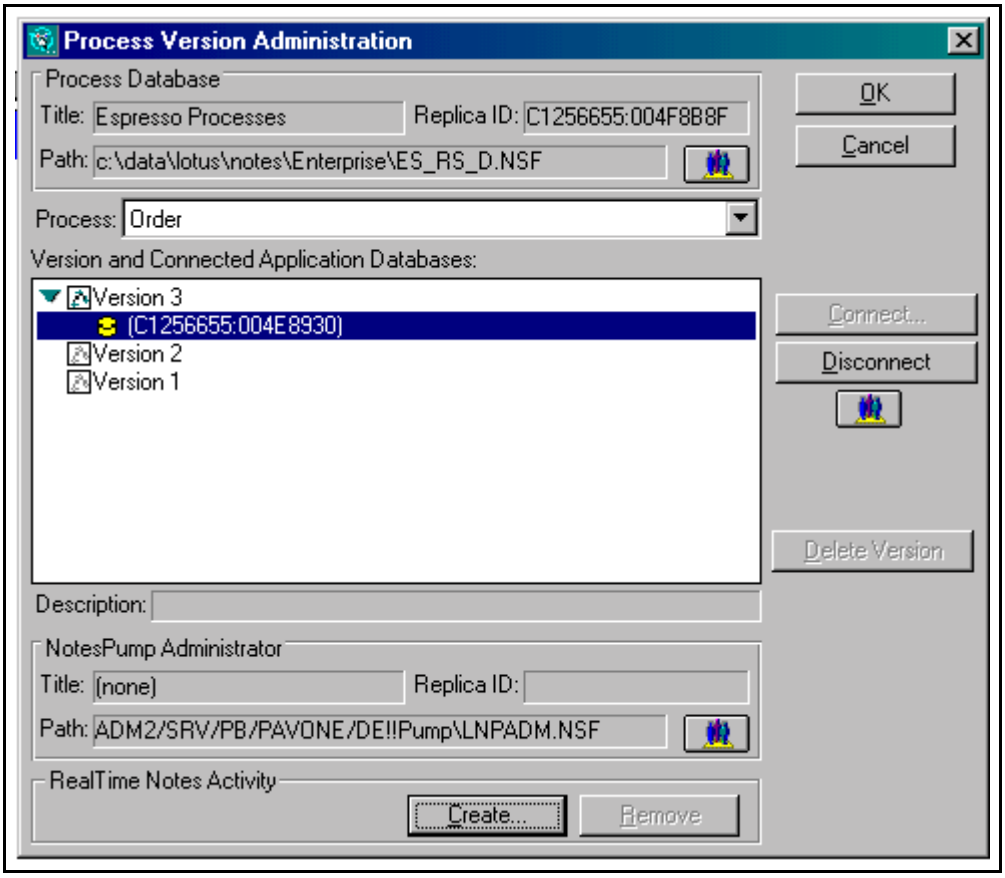
The RealTime Notes activities contain of more internal fields which have to be set if the activities are created automatically. The formula “Computewithform”[21] creates these fields as if the user saves the activity document manually. The ProcessModeler creates

[21] Please refer to the Lotus Notes on-line help for more information.

the new activity, enters all above values and then Lotus Notes runs the formula “computewithform”. Afterwards the activity document is saved.

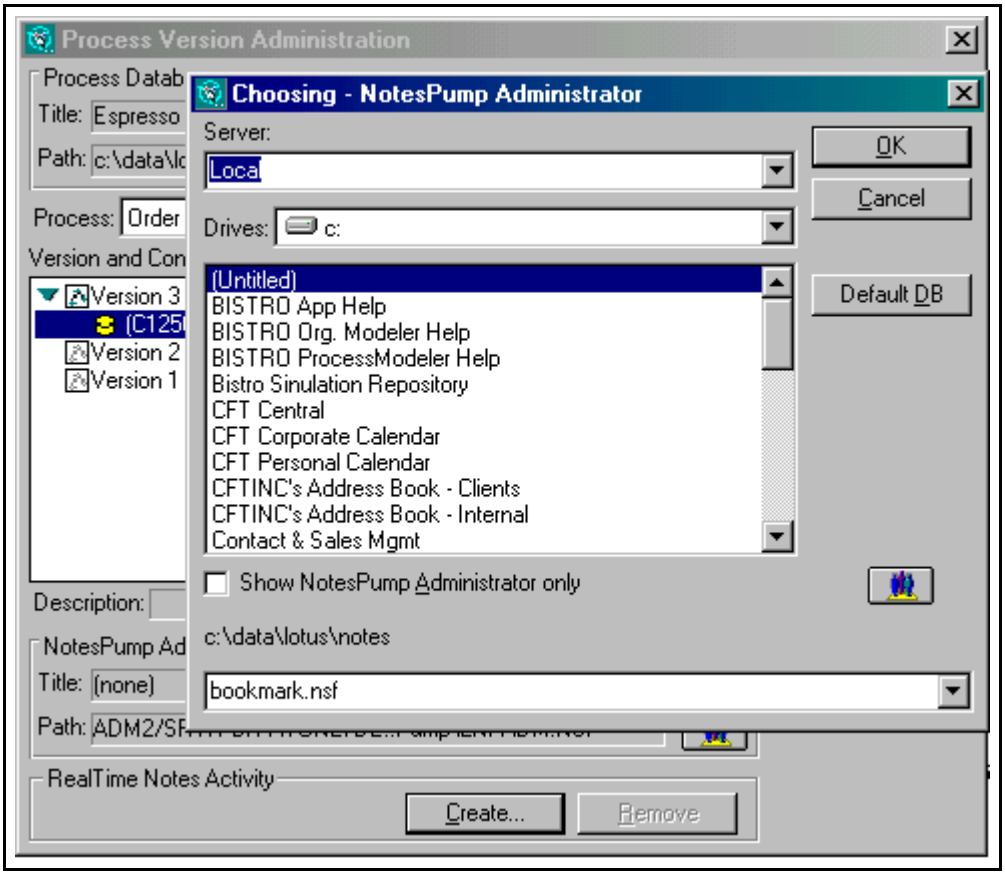
### 4.4.2 The Enterprise ProcessModeler

Within the ProcessModeler, the task definition dialog box contains an additional tab for the enterprise specifications[22]. After the task definitions are completed, the process can be saved. Every process in PAVONE Espresso is related to an Application database, which must be selected before putting a saved process in a live environment[23]. At that time the NotesPump RealTime activities can be created. The user has to select the Application database as well as the NotesPump Administrator database, in which the activities will be created.




Picture 35: Version management in PAVONE Espresso

[22] As shown on page 38.
[23] Please refer to PAVONE Espresso manual




Picture 36: Choosing the NotesPump Administrator database

**User friendliness**

Creating a process containing links to a relational database involves a redundant field mapping because there are many tasks of one or more processes connected to the same data of the same table in the same database. In addition, many process designers are not familiar with the relational and Lotus Notes database structure.

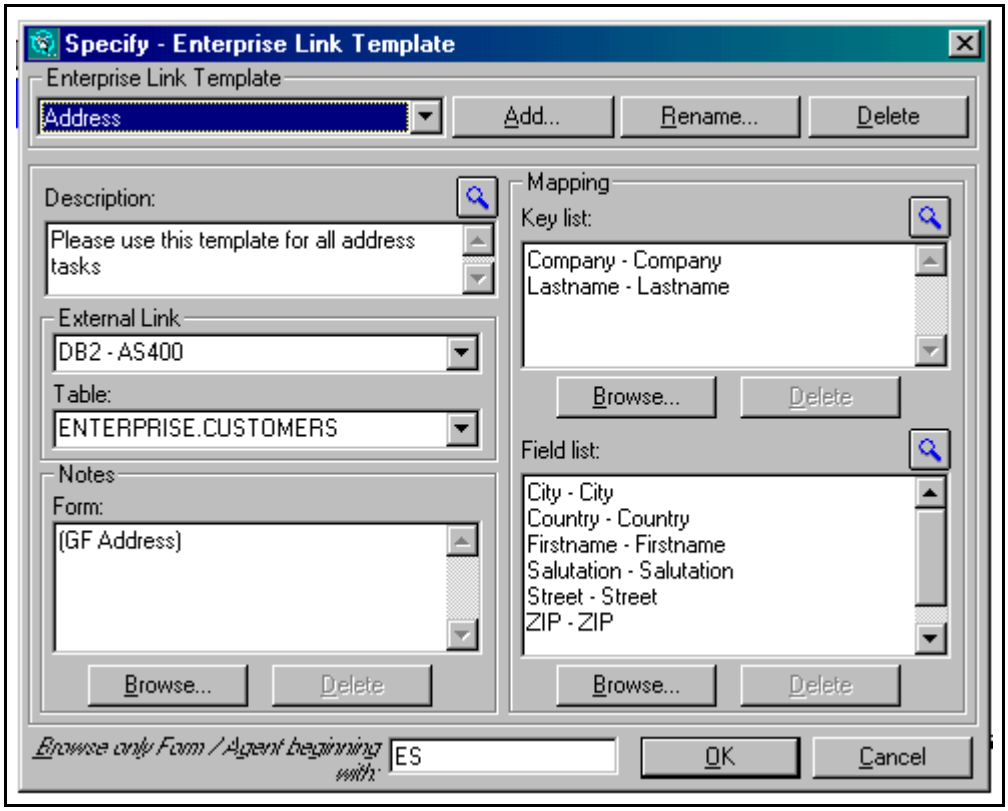
To make the Enterprise ProcessModeler more user friendly for easy external link specifications, Enterprise link templates are included in the Modeler. An Enterprise link template allows a set of different external database specifications and field mapping, created by a database administrator, which can than be used by the process designer for every task. It basically separates the link definitions from the activity settings within the Enterprise tab. The Enterprise link templates are stored in the Processes database[24], which allows use of the templates throughout several processes in one process database.

[24] Please refer to PAVONE Espresso manual

Every link template contains a template name to clearly identify it. It also has a description field for text entry. This will later be displayed to the process designer to give the template a meaning. Which link template is used for every task is mainly dependent on the Lotus Notes form used at the task. Based on the selected form in the task definition, the process designer should only see the enterprise templates that are useful for that form. An address form will mainly link into an address table, a product data sheet will mainly link into a product definition link. Therefore, every enterprise link template is related to a list of Notes forms, for which the template can be used. Every template can be used for more than one form, because there might be many forms alike with the same fields. This way, a new template does not have to be created for each alike Lotus Notes form.

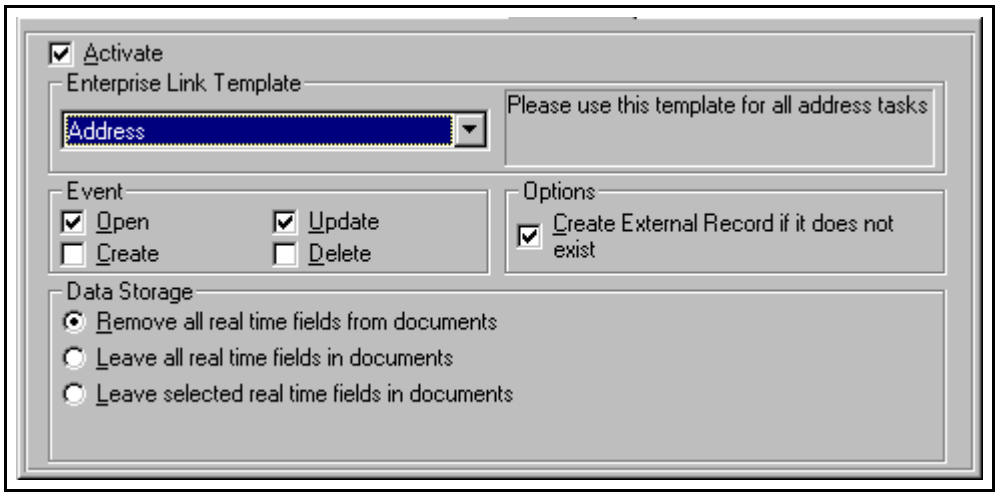
The other parts are derived from the link Enterprise tab of the task definition. The relational database and the table can be selected from a drop down list. The key and field list mapping displays the Lotus Notes field name and the relational column name in one row.

In addition, new link templates can be created and existing templates can be renamed or deleted.




Picture 37: Enterprise Link Template

The process designer can use the enterprise link templates in each task definition. Dependent on the selected form used in the runtime, the designer can choose from a list of templates. The template name and its description help to identify it. The following screenshot shows the modified Enterprise tab:




Picture 38: The new Enterprise tab in Espresso

### 4.4.3 Espresso Runtime

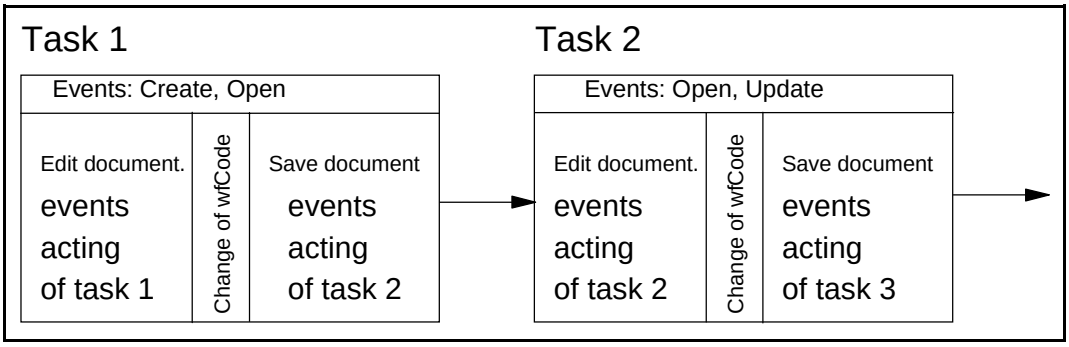
After the Enterprise ProcessModeler creates the RealTime Notes activities, they start automatically and monitor the Espresso Application database. Each activity looks for documents that are at a certain status, that is to say documents created with a specific form containing the wfCode field, which matches the internal Espresso task name, are watched for the events open, save, create and delete. If one of the events happen to the right document and the event is selected in the activity, the NotesPump server handles the data exchange with the specified relational database. Since this all runs in real time, it is transparent to the end user.

After a document is completed by the end user, the document changes it’s status to the next assigned task, which means that the wfCode field is set to the next internal Espresso task name. Afterwards, this document is saved and than closed. This means that the document is actually saved in the status of the next task, which lead to the problem that the activity for the next task is acting on the document of the current task. This is very problematic for activities that control the update event. A document is

updated by the activity settings for the next task and not by the activity of the current task, as it should. The create event causes the same problems.

There are no problems with the open and delete events, the problem exists only for the moment the document is saved after changing the status. Afterwards the document is at the right task, controlled by the right activity.

The next graphic shows an example where the problem affects the correct order of events.




Picture 39: Task event handling

The activity of task 2 supervises the open and update events. After the process document is completed at task 1, it changes it's status to task 2 and is saved. This means that the update event of task 2 is acting at a point where it is logically still at task 1.

**The solution**

One approach to solve the problem is to use two activities for each task. One activity is the main one, which acts the way it should be, the events are derived from the ProcessModeler settings. This main activity is called the "Go" activity within the prototype. The other activity acts during the saving of the document at the former task after the workflow status of the document has changed (that is the wfCode field is set to the current task). This activity is exactly the same as the main activity except for the update and create events, which are not activated. The open and delete events are selected in this additional activity only if they are selected in the main activity. This additional activity is called the "Stop" activity in the prototype. Since these events do not act during the saving of the process document, they will not act during the saving of the document at the preceding task. When the document is opened or deleted at the

current task, the events of the present task act. After the document is opened, the “Go” activity steps in and allows the update event to act.

This procedure repeats when the document is routed to the next task: It changes its status to the next task, saves it at the “Stop” activity of the next task interferes until the document is opened and the “Go” activity steps in.

Based on this model, the runtime in the Espresso Application database has to be changed. The key part is the routing of the document. When the user completes the task, the runtime saves the document to run the update event of the “Go” activity. Then the wfCode field is set to the next task and it sets a new field wfRoutingStatus to “Stop”. In LotusScript the code is:

```
' DOCUMENT ROUTING
    Call doc. ReplaceItemValue(“wfCode”,”{next task}”)
' ENTERPRISE
    Call doc.ReplaceItemValue("wfRoutingStatus", "Stop")
```

The runtime then saves the document once more to keep the changes of the two routing fields. The activity that acts at this time is the “Stop” activity of the next task, which means it does not update the relational record.

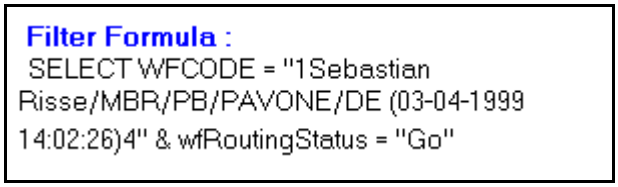
The wfRoutingStatus has to be set to “Go” as soon as the document is opened again at the next task. This is done in the Postopen Lotus Notes document event of the $Workflow subform[25] in the Espresso Application database. It contains the following LotusScript code:

```
' ENTERPRISE
    Call doc.ReplaceItemValue("wfRoutingStatus", "Go")
```

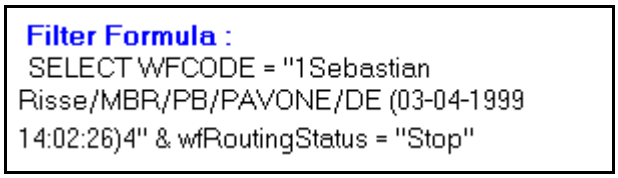
The two related activities created by the Enterprise ProcessModeler need to have a different filter formula to ensure the supervision of the correct process documents. The “Go” activity will control the documents where the wfRoutingStatus matches the value “Go”, whereas the “Stop” activity controls the documents with a wfRoutingStatus “Stop”:

---

[25] Please refer to PAVONE Espresso manual for more details.

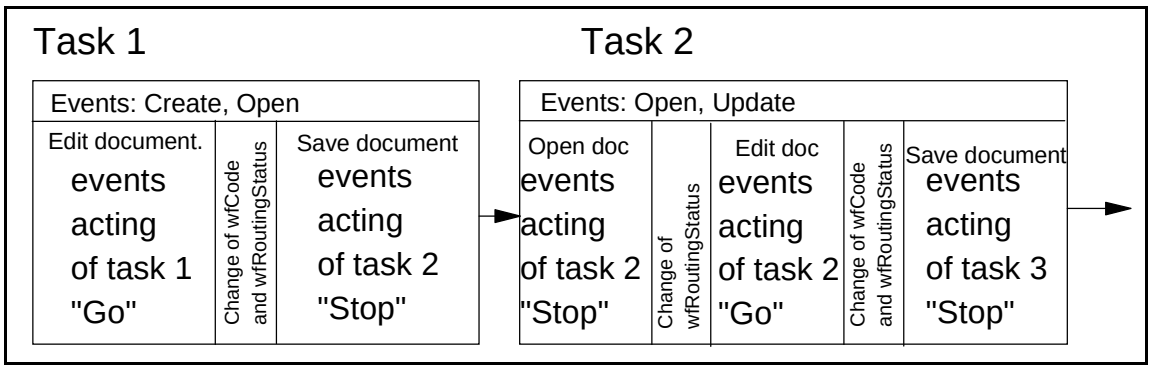



Picture 40: Filter Formula for the “Go” activity




Picture 41: Filter Formula for the “Stop” activity

The following model shows the double activity procedure for an example process:




Picture 42: Updated task event handling

# 5 The Solution: Espresso Enterprise

Enterprise Application Integration is the new catch phrase for the early 21st century. A graphical design tool for groupware based business processes with the capacity of integrating enterprise-wide data and legacy systems is a simple way to leverage existing IT infrastructures and legacy data.

The designed ProcessModeler allows to model business processes which link into any stored data within the organization at any given time. The tasks of the process contain information which is collected from mainframe data, client/server applications, the underlying groupware system or is entered by the end user. The process designer can

utilize mechanisms in the Modeler that can control data security, data integrity, data consistency and can reduce data redundancy.

The next chapter lays out the advantages and the disadvantages of Espresso Enterprise. A step by step procedure of how Espresso Enterprise is used to build a business process that connects to legacy systems is the part afterward, followed by some example processes that utilize the Enterprise functionality. Chapter five concludes with the list of errors and lacks as well as system requirements, performance tests and installation guidelines for Espresso Enterprise.

## 5.1 Advantages and Disadvantages

Espresso Enterprise as to be compared with basically two systems: The differences to the standard Espresso solution and the model of completely redesigning the enterprise IT infrastructure.

### 5.1.1 Standard PAVONE Espresso

Compared with the standard version of Espresso, the enterprise edition can handle greater data volumes within the processes. Lotus Notes in the version 4.x is only able to store up to four GB of data in one single database. But even this number is certainly lower if data access time matters. By outsourcing the data to relational databases, the effective amount of information used in the processes is significantly higher than with pure Lotus Notes based databases. This is especially true for archiving data at the end or within the processes.

Using the data storage option in the task definition of the ProcessModeler, field values that are transferred to the external database can be removed from the Lotus Notes document during the document closing. Reopening the document will pull in the values from the external database during the post open event. This process is faster than opening a standard document in Lotus Notes, particularly when the number of exchanged fields is very high. Another advantage is that Espresso Enterprise allows to spread information across distributed and heterogeneous systems within the enterprise. This includes the function to integrate mission-critical data into groupware processes, whereas Lotus Notes is usually used for non-crucial information sharing. Espresso Enterprise also keeps data consistency, integrity and reduced redundancy, which is not

necessarily provided by tools which just pull in data from external sources and keep it in Lotus Notes until it is pushed back to the relational databases at the end of the process.

With Espresso Enterprise, the data collected can be stored in other Lotus Notes databases as well. This allows to build a process across multiple Lotus Notes databases so all existing groupware applications can be connected to the process. Often the information is spread across several different Lotus Notes databases, but need a central point from where the data is seen and changed. The Enterprise ProcessModeler allows to specify Lotus Notes databases as the external links instead of relational databases and even is able to provide a connection to relational databases and groupware databases at the same time.

On the other hand, Espresso Enterprise needs more resources than the standard edition. As additional software, Lotus NotesPump has to be installed next to the Domino server and uses a significantly higher volume of system RAM. The additional software requires also additional skill sets and administration work. It has been shown that processes using the enterprise functionality are more complex and more difficult to build than basic ones, caused by the fact that the data used in the process has to be collected through additional process steps.

### 5.1.2 Redesign of the Complete Enterprise IT Infrastructure

The biggest advantage of Espresso Enterprise to the approach of a total redesign of the company's IT infrastructure is the fact that it is easy, fast and cheap to install and to implement. Compared to a few years of installation and upgrading enterprise systems, which is considered as gathering momentum, installation and administration as well as process development in Espresso Enterprise is a rather minor issue. With Espresso Enterprise, only few new technical staff are needed and training for the new ProcessModeler and the end user interface is in no contrast to the redesign approach.

Even though the enterprise edition of Espresso needs more hardware and software than the standard edition, significantly less resources are needed than a completely new system. It leverages not only the existing hardware and software as well as the existing meta data, but it also does not change the data itself during setup. This legacy system is enhanced by new technology such as groupware and Internet connectivity.

A disadvantage of the Espresso Enterprise solution can be the fact that a simple installation of Espresso Enterprise is not a complete system and means a complex and over a longer period of time evolving and upgrading design environment where the processes itself have to be implemented dependent on the companies specific needs. Espresso Enterprise can lead to an endless live cycle of process improvement. Many companies though see this as an additional advantage when the system can grow with the companies needs.

Espresso Enterprise is mainly an add-on to existing applications within the enterprise. Not all needs can be implemented as business processes, which means that the greatest benefit of Espresso Enterprise has to be the capability of integrating not only into legacy systems but also into upcoming applications and technology.

### 5.2 Enterprise Process Modelling Life Cycle

The standard implementation of an enterprise business process consists of eight steps:

1. Identification of the business processes and the tasks

   Assessment management including employee interviewing, questionnaires and work surveys help to identify the processes and the existing steps.
2. Process map design

   The Espresso Enterprise ProcessModeler is used to visualize the collected information of the first step and to review the integrity of the process. The basic information such as task editor, activity lists, instructions and the Lotus Notes form used to display the task to the end user has to be entered at each task.
3. Identification of the Enterprise template links

   After the tasks of the process are defined, the connections necessary for the process can be specified. This will be done by a database administrator who is able to complete the field mapping.
4. Identification of the connection points

   For every task that is connected to the external database, the Enterprise template link must be chosen. This identifies where the information are pulled and pushed to.
5. Implementation and specification of the links

   All settings for the enterprise connection can be entered now. This includes the

event settings (open, update, create and delete), the data storage options and the option for the creation of the external record if it does not exist.

6. Simulation of the process

   The designed process can be simulated with the simulation tool of Espresso or in a live test environment. The simulation will show bottle necks, maximum workload, time frames and cost information. The process can now be enhanced and optimized through update cycles.

7. Live environment

   The process can now be published to the live environment. Even though a standard procedure is to test the process within a small group of people first, it should also be tested in a complete live work environment to retrieve accurate information about the runtime.

8. Continuous process improvement

   Espresso Enterprise allows to build different process versions which can be exchanged during runtime. This allows the process designer and the process supervisor[26] to improve not only the process itself, but also change the way of work in general. The initial process is often not the best solution, but problems can only know be identified through the graphical design interface and simulation techniques.

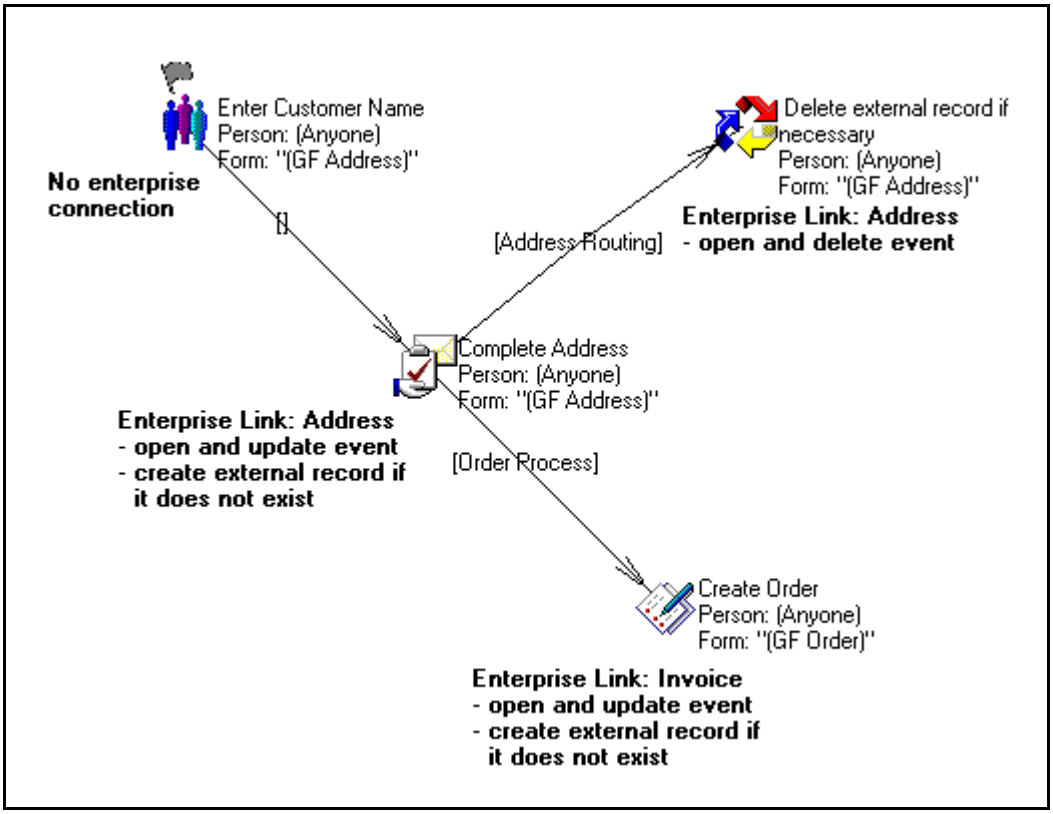
These steps should be seen as a guideline only. It will be different based on the existing resources, given information and work environments.

## 5.3 Examples and Practical Environments

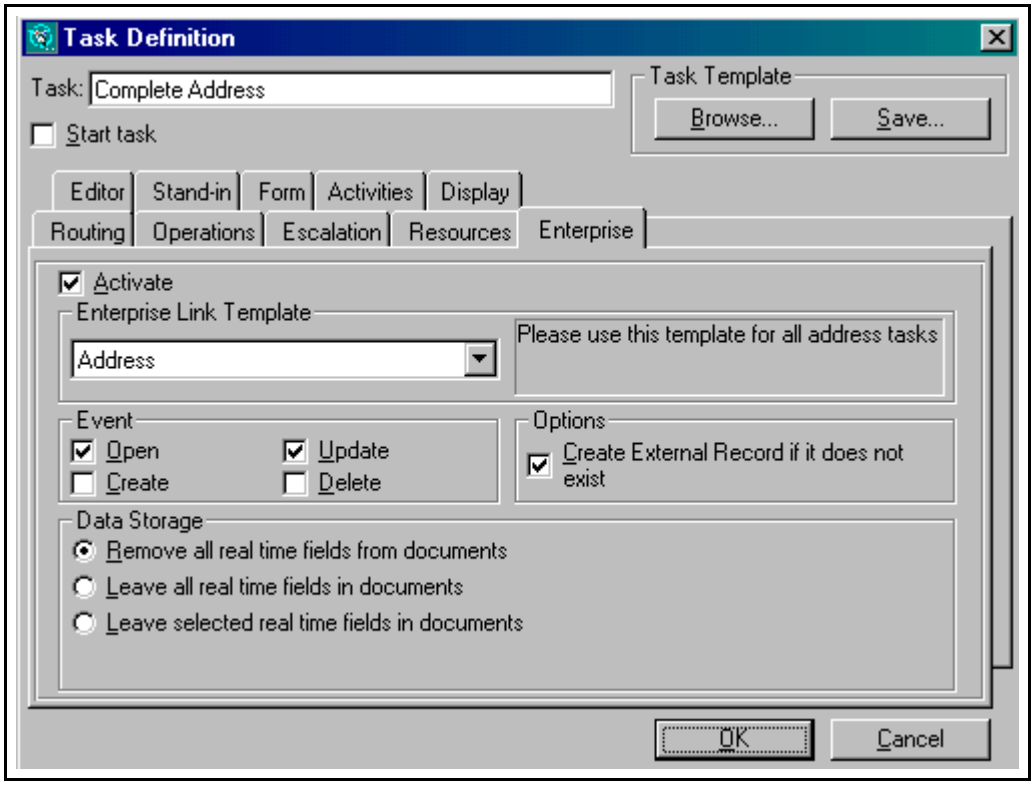
### 5.3.1 Simple Order Process

This is a basic process which shows how Espresso Enterprise is used to collect and to publish data from and to relational databases.

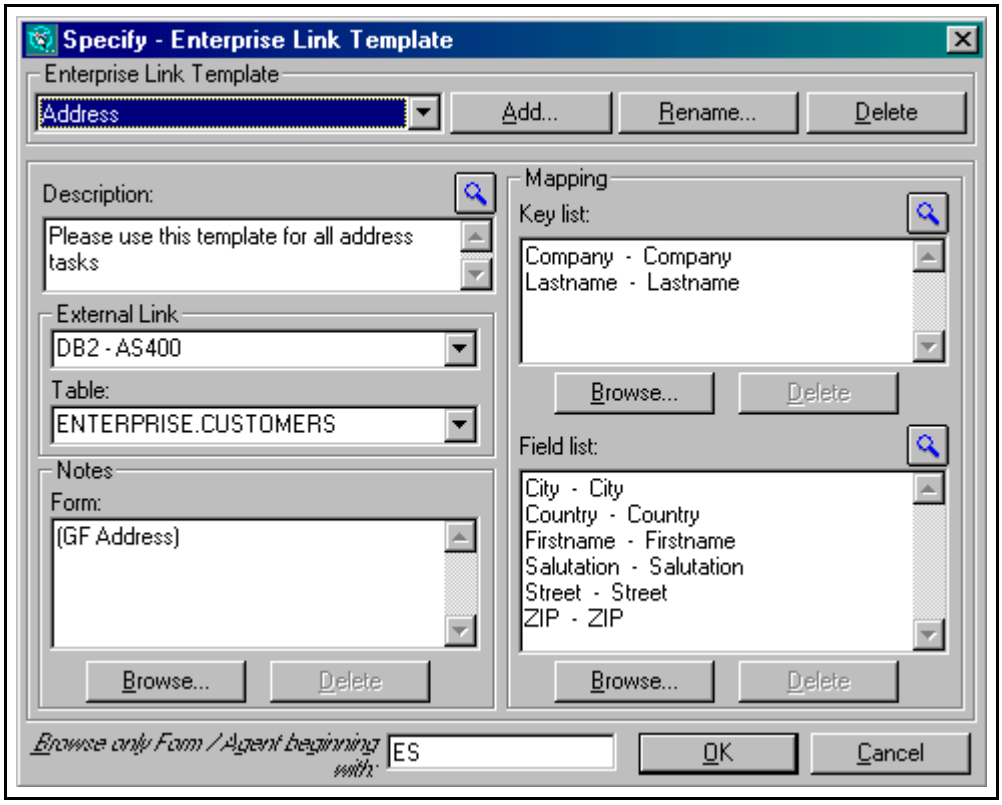
[26] The process supervisor controls the working process in the runtime. Please refer to the PAVONE Espresso manual for more details.




Picture 43: Simple order process




Picture 44: Enterprise task definition for order process




Picture 45: Enterprise Link Template Address for order process

In the runtime, the user starts a new job based on the Order process. A new address document is created in Lotus Notes and the user enters the last name and the company name. The first task is completed and is routed to the next task. So far, there is no connection to the enterprise data. When the user opens the document at the next task “Complete Address”, Espresso Enterprise automatically looks up the ENTERPRISE.CUSTOMERS table in the DB2-AS400 database. It will search for the key (fields Lastname and Company) and will create a new external record if it does not exist yet (“Create External Record if it does not exists”). If it finds a matching record, it will display all fields (City, Country, Firstname, Salutation, Street and ZIP) from the record in the Lotus Notes document. If changes are made to these fields, it will update the external record during the saving of the document (“update event”). It will also erase all mapped fields from the Lotus Notes document (“Remove all real time fields from documents”). The key fields will be kept in the document as the key link to the external record.

After the task is completed, the document is split and an order document (form “(GF Order)”) as well as the address document exist in Lotus Notes. If the document at task

“Delete external record if necessary” is deleted, the external record is deleted as well (“delete event”). When the order document is opened at task “Create Order”, Espresso Enterprise will create a new external record in the ENTERPRISE.INVOICES table and update all changes made at that task when the document is saved.

After the process is completed, the enterprise connection is not active any more, which means that deleting the Lotus Notes documents does not effect the external records.

### 5.3.2 E-Commerce

The fact that the enterprise functionality of Espresso is usable through a web browser, allows e-commerce[27] process design. Any Lotus Notes document is converted by the Lotus Domino server into html code and lets web users access the Lotus Notes databases directly through the browser. In combination with PAVONE Espresso, this means that process tasks can be completed not only through the Lotus Notes client, but also through the web browser[28]. In combination with the new enterprise functionality of Espresso, this means that legacy data can be accessed and be changed through the browser, hence the Internet.

Building an e-commerce process means that some tasks are performed by the customer over the Internet through the browser, and the other tasks are completed by the employee through the Intranet accessed through the Notes client or the browser as well.

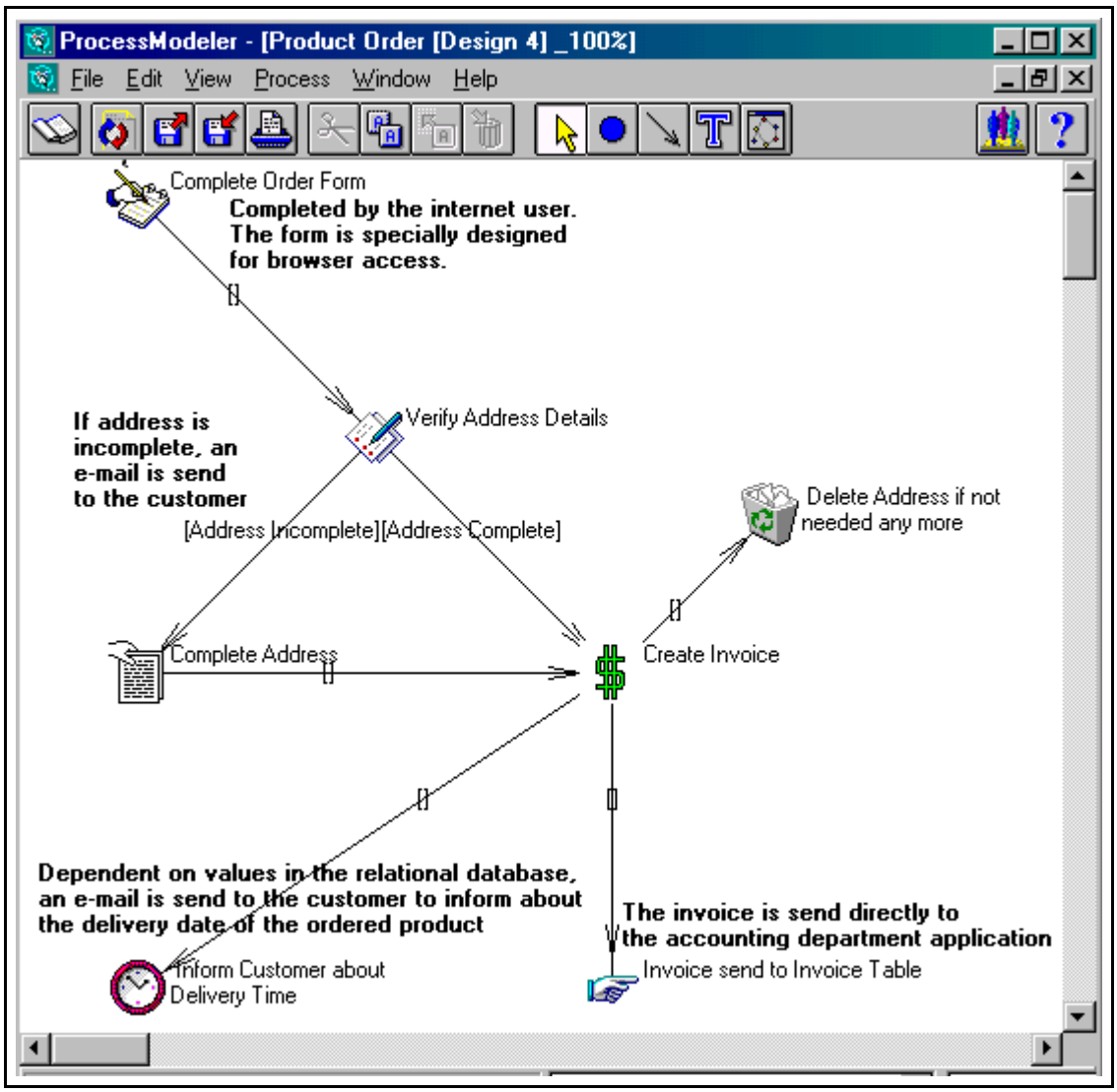
The next screenshot shows an example of an e-commerce process through the web. The first task is completed by the web customer, who fills out an order form. This is submitted and routed to the employee who takes the order, makes sure that the customer details are complete and the products are available. The process than creates an on-line invoice document that is send to the customer and an order form that is send to the production department and accounting for further processing.

---

[27] For more information on e-commerce please refer to Brans 1998, Adam 1999 and Kalakota 1996.

[28] Please refer to the PAVONE Espresso manual for more details on the Espresso web functionality.

The screenshot shows the basic steps:




Picture 46: Basic e-commerce order process

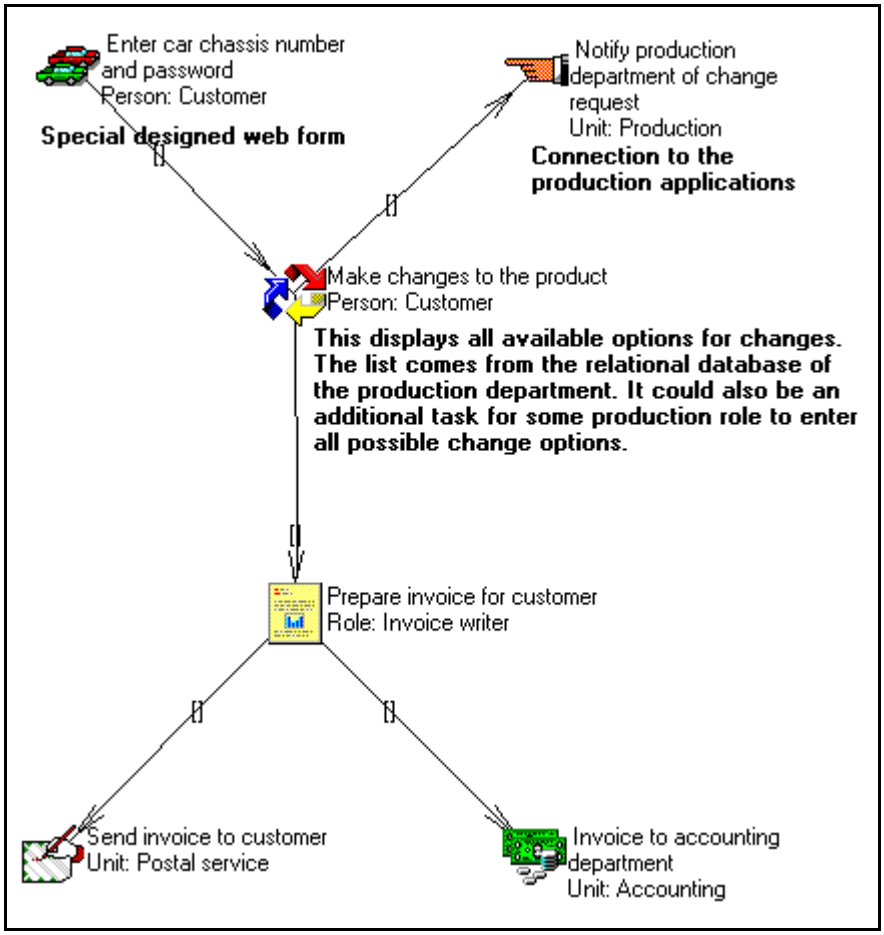
### 5.3.3 Production Interaction

This is a request of a German car manufacturer, but it can be used in all production areas what produce over a long period of time.

The customer, who has ordered a car, which is still in production shall be able to see at what stage the assembly has come. This customer inquiry can come in through the web, by phone or fax. Important here is a fast reply to the inquiry, because the system will be enhanced with the functionality to let the customer make changes to the product, while it is still at an early production stage. The system shall return a list of changes that are

still possible at the request time, for example the color, the engine size or some extras of the car. The car has the chassis number to identify it clearly and the inquiry process uses this number as the key field. This should be combined with a password or address details to ensure that only the right customer can to the inquiries as well as the change requests.

The next screenshot shows an example for this inquiry process, which has to be customized dependent on the manufacturers needs.




Picture 47: Production interaction process

### 5.4 Lacks in Espresso Enterprise

There are some missing parts in the Enterprise edition of PAVONE Espresso. Examples have shown that with additional features, a greater range of processes can be designed with this tool. The next chapters give some components that can be implemented into the momentarily product.

### 5.4.1 Keyword Lists

A very useful feature within Lotus Notes are keyword lists. If for example customer details such as the address are needed within the document, the user has to know the key value to retrieve these information. If the last name or the company name is misspelled or not known, Espresso Enterprise will not return the right address details in the order process example. In fact, if the misspelled values do not exist in the relational database, Lotus NotesPump will create a new external record, even if the demanded address exists.

A workaround for that and other cases is the use of keyword lists. In the “Enter Customer Name” task of the order process example the user can than choose the company name and the last name from a keyword list, which eliminates the errors of misspelling.

Lotus NotesPump is not able to produce keyword lists from multiple records and populate them in a single Lotus Notes document. A workaround is to utilize the Lotus Notes @Functions @dbcolumn and @dblookup.

Example:

*@DbColumn("ODBC":"Cache";"AS400";"Sebastian_Risse"; “password”; "Enterprise.Customers"; "Lastname")*

This example will return all values out of the “LASTNAME” column of the “ENTERPRISE.CUSTOMERS” table in the “AS400” database, which is accessed by the user “SEBASTIAN_RISSE” and the password “PASSWORD”.

This approach needs an ODBC connection to the data source, which leads to operating system dependent and means that programming skills are needed.

At the moment there is no other solution found. This needs to be investigated in greater detail.

### 5.4.2 MQSeries Integration

With the finish of this work, Espresso Enterprise was not able to connect to transactional systems via message queuing techniques such as MQSeries. There is a

great need for this component, since this allows the process designer to build enterprise application integration processes without or this few programming.

The basic approach is to allow the process developer to specify the settings for the message queuing at every task of the process. The following list shows the settings necessary to build up this connection:

1. Send or Receive

   This determines if the data shall be pushed to the legacy system or gets a message from it. Another option is the dual way, which means that it sends data and waits for a reply, which is then published in the Lotus Notes document
2. External data source

   The external systems name should be provided in a pick list. The external system can be a transactional system such as IMS or CICS from IBM
3. The data that will be transferred

   If it is a send value, the Lotus Notes field is specified, in which the data is stored. If it receives data, the Lotus Notes field is specified, in which the data will be saved.

Building MQSeries into Espresso Enterprise is a very important part, but is complicated and requires a very in-depth knowledge of message queuing and a transactional system background. Please refer to the next chapter on how this feature will be implemented in the prototype.

### 5.4.3 Multiple Records in one Process Document

Espresso Enterprise is able to send and retrieve information to/from only one external record. This limitation is given by the fact that only one Lotus NotesPump Real Time activity can monitor a specific set of documents through the filter formula and the Notes form used. If this tool is able to run more than one activity on the same Notes documents, then it would be possible to collect data from multiple records in one document, which would lead to a much greater flexibility within the process.

### 5.4.4 Automatic Document Handling with no User Interaction

The realtime Notes functionality of NotesPump works only during user interaction, that is that only documents that are opened and closed manually are able to exchange data with the external database. A LotusScript, which closes and opens a document even through the frond end classes[29] of LotusScript, does not activate the NotesPump realtime activity and the data is not exchanged with the relational database.
This causes two problems:
1. Tasks that are completed by a Notes Agent[30] can not retrieve or send data to the external record. A workaround is to store the required data in Lotus Notes before the task and to push it back after it. This leads to a more complex process design.
2. The user has to save and close and than manually reopen the Lotus Notes document in order to get new information from the external record. Completing a task without closing the document and reopening it manually (this could be achieved with a Lotus Notes agent) will not retrieve external information. A workaround is to manually close and reopen the document.

This problem is caused by LotusScript, due to the error that a script does not physically close and open the document, and by NotesPump, which is not able to interact with the script.

### 5.4.5 Actions based on External Record Changes

A standard requirement of business processes is to initiate a process based on values in the external database. A new record or a change in an existing one is often the reason to start a new job[31] in a process. This interaction is not implemented in Espresso Enterprise, a user or a macro needs to look in the relational database on a scheduled base and then decide if a job has to start.

Another workaround is to initiate the job out of the relational database through the API. This implies much programming within the relational database as well as in the Espresso runtime and does not require the Enterprise edition. In fact, Espresso Enterprise does not support this feature yet.

---

[29] Please refer to the LotusScript on-line help on Lotus Notes for more information
[30] A task in Espresso can be completed by a person, a group of people or a Notes Agent. Please refer to the PAVONE Espresso manual for more information.
[31] A job in Espresso is seen as one instance of a specific process.

### 5.4.6 Change of the Key Values

If one key field value within the Lotus Notes document changes during the process, a conflict occurs due to the fact that the key field values are the link to the external record. Lotus NotesPump offers three options how to handle the conflict:

1. Do Not Track Update to Key Fields

   This allows the update of the key fields in the Notes document only, but not in the external record.
2. Block Update to Key Fields

   If the key fields in the Notes document change, it does not update the external record. In the Lotus NotesPump Log database an error is shown.
3. Handle Update to Key Fields as Delete/Insert

   If the key is changed in the Lotus Notes document, NotesPump automatically deletes the old external record (with the old key) and creates a new record with the new key.

Option 2 and 3 create temporary hidden Notes fields in the document while opened and deletes them after closing.

These options can be included in the ProcessModeler in the next version, to allow the process designer to set the update handling.

Another way to block the update of the key fields is to make the key fields read only after entered the first time. This can be done through the Lotus Notes form design.

### 5.4.7 Additional Options within the NotesPump RealTime Activities

The realtime Notes activities in NotesPump contain additional options to control the data exchange between Lotus Notes and the relational databases. Most of them are not essential for standard connections, but can be helpful for special occasions. Instead of manually changing these settings manually[32], they can be included in the task definition in the ProcessModeler. The following shows a list of options ordered by importance which can be included in the ProcessModeler for the next version:

1. Monitor any forms

   This option allows the activity to monitor any Lotus Notes form, in contrast to the

[32] Changing the realtime Notes documents requires to stop the activity on the NotesPump server, to do the changes in the document and save it and then manually start the activity again. Please refer to the NotesPump manual for more information.

current way of watching only documents created by a special form. This is important for multi-document routing[33], so all routed documents can exchange data, regardless of their form name.

2. Designated Server

   Since a NotesPump administrator database can handle more than one Pump server, this would allow the process designer to spread the workload of the activities to all servers. This is interesting if many activities are running at the same time.

3. Time dependent activities

   To specify the dates and times when an activity is actually running, allows the process designer to build more flexibility into the activities. As an example the transfer to external databases is only allowed at certain times of the day because it is overloaded at the other times.

4. E-mail notification on error

   Here an administrator can be specified who will be notified if an activity returns an error. This allows faster acting and error correction than checking the log database on a regular base.

### 5.5 System Requirements and Performance Tests

The hard- and software requirements for Espresso Enterprise are significantly higher compared to the standard edition.

Espresso Enterprise was tested on an Intel Pentium II 300 MHz machine with 128MB RAM and has shown that this is the absolute minimum hardware equipment for only a test environment without live data and no more than 3 test persons. The memory requirements are very dependent on the number of activities that run at the same time. Generally one activity needs about 10MB of RAM. The HD size is dependent on the number and sizes of the Lotus Notes databases as well as the relational databases if they are stored on the same computer. For the software shown below about 3 GB are the minimum requirements.

The computer was used as a standalone systems for test runs, which means that all software necessary for Espresso Enterprise to work properly was installed on the same machine.

---

[33] Multi-document routing means that not only one Lotus Notes document is routed in a job, but many.

The operating system used was Microsoft Windows NT Server 4.0 with SP3. This is the only product out of the Windows family that supports Lotus NotesPump, which was installed in version 2.5a. Lotus Domino Server 4.6.1a was used as the groupware platform and an additional Notes client as the end user interface with all standard Lotus template databases represented the platform for Espresso Enterprise.

As a relational database Microsoft Access 97, IBM UDB2 enterprise edition and Oracle personal edition 8 where tested on the same machine as the external database structure.

Espresso Enterprise was fully installed with the OrganizationModeler, the Enterprise ProcessModeler and the Enterprise Runtime databases. This used about 40 MB of the hard disk.

After creating a process with ten enterprise tasks and creation of 20 NotesPump realtime documents the software needed about 300 MB of RAM, which implies that about 172 MB had to be swapped to the hard disc and significantly slowed down the system and thus the response time of the Domino server.

Another test was made on an IBM AS/400 with a 170 processor and 128 MB RAM. The Domino server, the NotesPump server and the DB2 database where installed on the same system. The same example process with the 20 activities did not slow down the system.

### 5.6 Installation Guidelines

The installation of the Enterprise edition differs only slightly from the guidelines of the standard version. To set up an Enterprise computer as shown in the chapter above, the order of software installation is crucial:

1. Step: Windows NT 4.0 and SP3
2. Step: Domino server and Notes client
3. Step: Creation of the Notes ID for the NotesPump administrator
4. Step: Lotus NotesPump 2.5a installation with connection to the Domino server
5. Step: Realtime Notes activation of NotesPump

   Therefore, an entry has to be set in the Notes.ini file:

   EXTMGR_ADDINS = lnpext.dll
6. Step: Installation of any relational database

7. Step: Installation of the Espresso Enterprise Runtime.
   This is the same installation as the standard Espresso installation[34].
8. Step: Installation of the Standard Espresso OrganizationModeler
9. Step: Installation of the Espresso Enterprise ProcessModeler
   This is the same installation as the standard Espresso installation[35].
10. Step: Creation of Link documents within the NotesPump administration database to connect to the relational databases.
11. Step: Creation of the processes

These steps should only be seen as a rough outline. The usual approach for installing Espresso Enterprise is using it in an existing system, which means that some of the above steps are not needed.

# 6 Future Functions and Developments

The next steps to enhance the functionality of Espresso Enterprise are to build in the components explained in chapter 5.4 on page 68 to page 73. This will extend the usage of the tool and allows the process designer to create more complex workflows. With the keyword lists, MQSeries integration, multiple records in one process document, automatic document handling with no user interaction, actions based on external record changes, changes of the keywords and additional NotesPump features in the ProcessModeler Espresso Enterprise will be marketed as a Lotus Notes based enterprise process development tool.

In addition to that, Espresso Enterprise provides the base for other approaches to extend, enhance and upgrade to a tool that allows complete control of enterprise data through business processes. The next paragraphs give some examples for that.

## 6.1 EIS Connection

A very important request by companies is the integration of business process design, live task routing and the reporting of the live process environment data. The runtime data gathered during the process usage is a valuable information. Based on these

[34] Please refer to the PAVONE Espresso Runtime installation manual for more details.
[35] Please refer to the PAVONE Espresso ProcessModeler installation manual for more details.

information, process supervisors can see bottlenecks, handle escalation management, change work plans and inform executives and process managers. Executives and process managers can than make business decisions with these information, for example delay the marketing of a new product, inform supply chain partners, etc.

The more accurate, easy to use and visual the information system and the presented data is, the more value it has. Senior management and executives are mainly interested in high level reports, whereas process managers and supervisors need to drill down to user level to find the cause for problems. The ShowBusiness reporting tool EIS with it's OLAP technology component Cuber link into Lotus Notes databases and visualize the Notes data graphically. Cubes of information are created and can be sliced, diced and drilled down in the graphical EIS application.

In combination with Espresso Enterprise, EIS can show process data of any kind. Examples include overdue tasks and processes, escalation management, comparison of live process data versus simulation data and many more.

The information needed is dependent on the process designed and on the company's needs. The integration of Espresso Enterprise and an EIS tool is the next step to build highly beneficial processes.

### 6.2 Virtual ERP

The prototype of Espresso Enterprise has been shown to many companies during the writing of this work. These companies can be divided into existing standard Espresso users and companies that are looking for a tool to integrate enterprise wide data in business processes. The existing Espresso users are focused on the idea of extending the current Lotus Notes based data storage solution to save data to external database for performance improvements and archiving, whereas the other group often includes companies that do not use Lotus Notes at all. For this group, building up a completely new infrastructure with Lotus Notes and Domino is a major effort that is not always justified by the resulting benefits.

Espresso Enterprise can therefore be seen as the first step to a paradigm called Virtual ERP. The second step is the MQSeries approach which allows the process designer to access any data source and any application on any user level. This allows more

flexibility for the enterprise integration and it opens the way for the next step, the Virtual ERP.

Virtual ERP does not use a groupware platform as the end user interface or the actual document routing. The end user interface can be any existing application, for example transactional systems, e-commerce applications, reporting and development tools as well as a groupware application. This means that the end user will have the same user interface as before, but also allows to integrate new applications to participate in the data sharing. The actual routing of the task will be done with the help of MQSeries. The ProcessModeler interface can be used, but it will not create the routing structure in a Lotus Notes database, but in message queuing techniques. Instead of using MQSeries, an approach with IBM's San Francisco is possible.

The definition for Virtual ERP is not 100 percent set yet. Much development and reconstruction is needed in the future to define the involved components and requirements.

# 7 Bibliography

*Adam, Nabil R (1999):*
Electronic commerce: technical, business, and legal issues. Prentice Hall, Upper Saddle River, NJ, 1999

*Bock, Geoffrey (1995):*
Designing groupware: a guidebook for designers, implementors, and users. McGraw-Hill, New York, 1995

*Brans, Jean-Pierre, (1998):*
Decision support systems, groupware, multimedia and electronic commerce. In: Feature issue decision support systems, groupware, multimedia and electronic commerce, Elsevier, Amsterdam 1998

*Brodie, Michael L (1995):*
Migrating legacy systems: gateways, interfaces & the incremental approach. Morgan Kaufmann, San Francisco, 1995

*Buckley, Jo A (1986):*
Database management systems. Meckler, Westport, 1986

*Dierker, Markus (1998):*
Lotus Notes 4.6 und Domino: Integration von Groupware und Internet. Addison-Wesley, Bonn 1998

*Chaffey, Dave (1998):*
Groupware, workflow and intranets: reengineering the enterprise with collaborative software. Digital Press, Boston, 1998

*Clewett, Annette (1998):*
Network resource planning for SAP R/3, BAAN IV, and Peoplesoft: a guide to planning enterprise applications. McGraw-Hill, New York, 1998

*Coleman, David (1997):*
Groupware: collaborative stratagies for corporate LANs and Intranets. Prentice-Hall, Upper Saddle River, 1997

*Connolly, Thomas (1997):*
Database systems: a practical approach to design, implementation and management. Addison-Wesley, Wokingham, 1997

*Geiger, Kyle (1995):*
Inside ODBC: Der Entwicklerleitfaden zum Industriestandard fuer Datenbank-Schnittstellen. Microsoft Press, Unterschleissheim, 1995

*Gold-Bernstein, Beth (1998):*
Designing Enterprise Client/Server Systems, Prentice Hall, Upper Saddle River, NJ, 1998

*Greenberg, Saul (1991):*
Computer supported cooperative work and groupware. Harcourt Brace Jovanovich, London, 1991

*IBM Redbook (October 1997):*
Lotus Solutions for the Enterprise, Volume 4: Lotus Notes and the MQSeries Enterprise Integrator, IBM, 1997

*Jablonski, Stefan (1996):*
Workflow management: modeling concepts, architectures and implementation. Internat. Thomson Computer Press, London, 1996

*Jackson, Michael (1997):*
Business process implementation: building workflow systems. ACM Press, Harlow, 1997

*Kalakota, Ravi (1996):*
Frontiers of electronic commerce. Addison-Wesley, Reading, Mass. 1996

*Kemper, Alfons (1996):*
Datenbanksysteme: Eine Einfuehrung. Oldenbourg, Muenchen, 1996

*Khoshafian (1995):*
Introduction to groupware, workflow, and workgroup computing. Wiley, New York, 1995

*Lloyd, Peter (1996):*
Transforming organizations through groupware: Lotus Notes in action. Springer, London, 1996

*Lotus White Paper (May 1998):*
Domino Enterprise Integration: Lotus Notes and Enterprise Integration, White Paper, Lotus, 1998

*Miller, Howard Wilbert (1998):*
Reengineering legacy software systems. Digital Pr., Boston, 1998

*Ramakrishnan, Raghu (1998):*
Database management systems, WCB McGraw-Hill, Boston, 1998

*Riempp, Gerold (1998):*
Wide area workflow management: creating partnerships for the 21st century, Springer, London, 1998

*Simon, Alan R (1996):*
Workgroup computing: workflow, groupware and messaging. McGraw-Hill, New York 1996

*Warren, Ian (1999):*
The renaissance of legacy systems: method support for software system evolution. Springer, London, 1999

## Grateful Acknowledgment

I would like to thank the employees of PAVONE Informationssysteme GmbH for their help. Especially Howard Almond for his support, helpful thoughts and overall help during the whole project. Hong Zhang and Francis Sarver helped me during the implementation of the prototype. Stefan Meyer took over the project for future developments. Frank W Iveson of NotesWare Limited and Joe Sheehan helped me during the initial thoughts. Amy Jean and Graham Rae from CFT Consulting gave me much input for future developments to get Espresso Enterprise to the next level. A special thank to Royston Small, Susanne Driskell and Nicole Cummings from CFT Consulting for prove reading the thesis.