8.1 Application Structure

Applications that are built and executed according to the workflow paradigm are called workflow-based applications. They are no longer monolithic or executed on large central computers supporting dumb terminals, nor are they client/server applications where the still-monolithic application has been split into two pieces that are deployed to a large central computer and smaller intelligent workstations or net stations for better exploitation of resources.
These new applications execute on the network where the different pieces of the application are executed on different computers, each of them possibly running a different operating system, database system, or network protocol. Figure 1 illustrates this approach schematically.

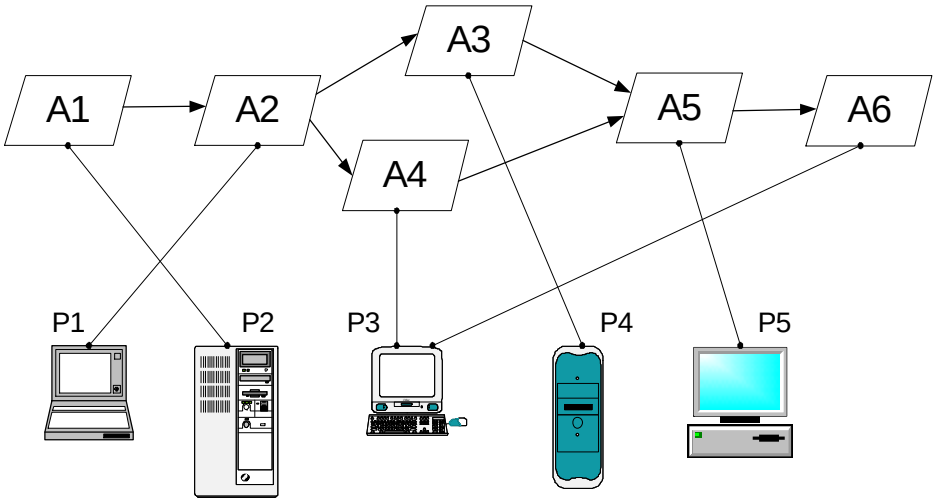

Figure 1 Schema of Workflow-based Application

The first activity A1 is processed without any user interaction on workstation P2 running AIX; the second activity A2 is carried out on laptop P1 running Windows 98. Since these applications are not just carried out on a single processor but on different processors connected via a network, they are sometimes also called networked applications.

The heart of these applications is the business process that is executed by the workflow management system. Based on the context in which the business process executes, different paths are taken through the process model. Thus, different processes can take completely different routes through the network, are being worked on by different people, and are processed by different programs.

The business process can even be executed by different workflow management systems if a business process invokes a business process outside the domain of the current workflow management system, for example, by sending a request via the Internet to some other company. Agreements between the different vendors and standards established by standards bodies are making this kind of execution happen.

8.2 Workflow and Objects

Enterprises are investing in object technology today to improve the productivity of their programmers and to enable even non-data-processing professionals to build applications by using visual builders. Vendors of standard software split their applications into business objects, allowing programmers to reuse the objects in a different context.

A cornerstone of object technology is the insight that the robustness of systems is normally achieved by encapsulating things that might become subject to change. If, for example, the order in which operations are to be performed can change or if operations can be added or removed, the guidelines of object technology consequently recommend the introduction of a dedicated control object. This control object encapsulates the scheduling of the various operations.

When the functions of the control object are examined, it is obvious that they are just a subset of the functions offered by workflow technology. The control object is represented by a process defined to the workflow management system. This representation enables vendors of standard software to glue together their business objects to form the original, monolithic business application.

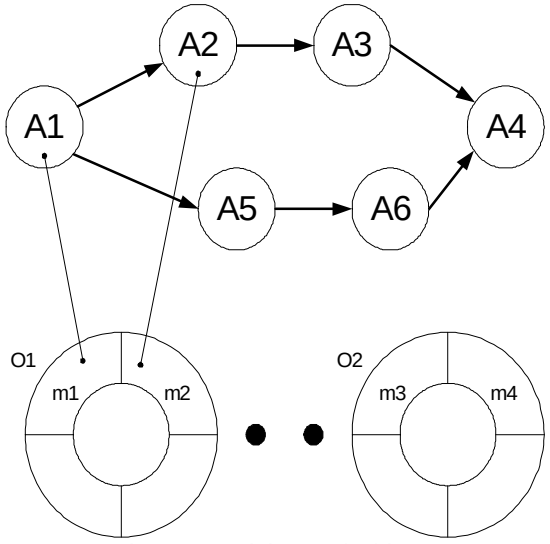

Figure 2 Workflow and Objects

Each activity of the process is implemented as a business object that is invoked by the workflow management system. As shown in Figure 2 an application thus consists of a business process and a set of business objects, where each activity is implemented as the invocation of a method of a business object. As illustrated, activity A1 invokes method ml of business object O1, and activity A2 invokes method m2 on the same business object. This scheme should be compared to the more traditional approach in which a workflow-based application consists of a business process and a set of programs.

8.3 Application Operating System

The purpose of an operating system is to provide an environment for the execution of programs. This environment includes the management of resources such as processors, memory, disk storage, and communication lines. The operating system effectively hides these resources by providing appropriate application programming interfaces (APIs) for the programs. A scripting language allows the user of such a system to specify the sequence in which the various executables are to be processed. The operating system will then make sure that the right resources are available to the program at the right time and at the right place.

When an application is made up of different components, the individual components must be called in the correct sequence (flow of control) and data must be passed correctly from one component to other components (flow of data). This knowledge is either built into each of the components or localized in some special components that manage for example the flow of control. If the application is distributed, the various components must exploit the underlying infrastructure to manage this flow of control and data. In addition, they must implement the various recovery mechanisms. The components must therefore not only be built to deal with the local operating system but must also consider the characteristics of the operating systems of other components they are communicating with. With workflow, the component must only deal with the local operating system characteristics. The workflow management system takes care of the differences in operating systems, network protocols, and communication mechanisms. It just calls the component by using the invocation mechanism of the local operating system. If an operating-system-independent invocation mechanism, such as message queuing, is used, even this dependency can be removed. If the component is written in a portable language and does not perform operating-system-specific input/output operations, the individual component is portable.

Workflow management systems thus provide to applications the same functions that operating systems provide to the individual programs. It relieves the applications from dealing directly with operating systems and communication mechanisms by providing appropriate APIs. Workflow management systems are therefore sometimes referred to as *application operation systems*.

8.4 Software Stack

Figure 3 shows the position of workflow technology within a simplified software stack. This software stack shows only those components that are important for the points we want to make. The base is the operating system, which provides an environment for the execution of programs.

On top of the operating system is a set of programs that are collectively called middleware. They provide the infrastructure necessary for application programs to manage data (database management system), communicate with each other via messages (message queuing system), and implement distributed transactions (transaction managers and TP monitors).

These are the components used by the workflow management system to provide the function needed for an application operating system. The database management system is used to store the persistent information such as processes or organizational information. The message queuing system provides the infrastructure that is needed for communication between the different components of the workflow management system, either on the same computer or on different ones. The transaction manager helps the workflow management system process messages and make changes to the database under transaction control, providing the robustness required for an application operating system.

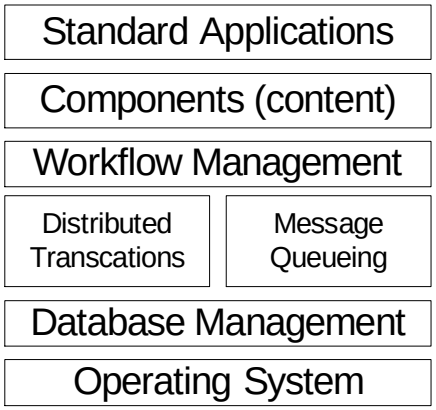

Figure 3 Software Stack

The workflow management system is then used by the next layer, which provides the object management infrastructure. This object support, together with workflow support, is then used by standard applications, such as payroll, general ledger, or groupware systems.

8.5 Document/Image Processing

Some business processes are centered around sets of documents that are routed from one person to the next. Document management systems have been developed to support this type of work. Because documents are quite often in the form of scanned images, these systems are also known as image-processing systems.

- These systems focus on the management of documents/images, and offer a wealth of functions to manage and manipulate documents/images:
- Functions to scan images and annotate them with text that makes it possible to efficiently locate the images later
- A document store to support not only the simple storage of documents but also to provide functions to group related documents into folders and to search for documents using sophisticated searches

- Archiving functions to remove documents and images from the document library and put them into some slower, but cheaper, storage
- A document editor to create, modify, and annotate documents
- A forms editor to define forms and the rules that should apply to the data entered into the forms by appropriate users
- The capability to define how documents, forms, and folders are routed from person to person within the organization, and the type of action to be performed by each individual

In document/imaging systems, the workflow functions are in general an integral part of the system. A business process is typically represented as a folder. This folder is the entry into all activities associated with the process; it allows the user to perform queries into the document store and to perform actions such as adding a document, filling out a form, or determining the next user to work on the folder. These systems are sometimes also called folder-management systems, since the folder is the focal point. Since a folder typically represents a case, such as the loan in a bank or an insurance claim, the term case processing is used sometimes for this type of processing.

In contrast to these systems, workflow management systems focus on the underlying business process and the execution of the business as a set of activities. These activities can be anything from mailing a letter via e-mail or making a telephone call to filling out a form, for example, that automatically calculates taxes, or updates a personnel record in a corporate database.

It should be noted that document management systems have their own internal workflow management system. In fact, it is much better to implement them on top of a stand-alone workflow management system since that implementation integrates better with other systems and programs in the enterprise.

8.6 Groupware and Workflow

Workflow addresses only one aspect of groupware, systems that help people to work together on the same piece of work. These systems provide the implementations for computer-supported cooperative work (CSCW). Figure 4 classifies groupware according to the three aspects of CSCW: collaboration, communication, and coordination. Because each of the listed implementations reflects the three aspects more or less strongly, a charge diagram illustrates the strength of each aspect in a particular implementation. The closer an implementation is to an aspect, the more prominent the aspect is in the implementation. Video conferencing, for example, only has the aspect of communication; it does not have any coordination or collaboration aspects.

Communication systems help people exchange information, quite often as a one-way communication. This exchange includes video conferencing, email, and bulletin boards on which people post their opinions.

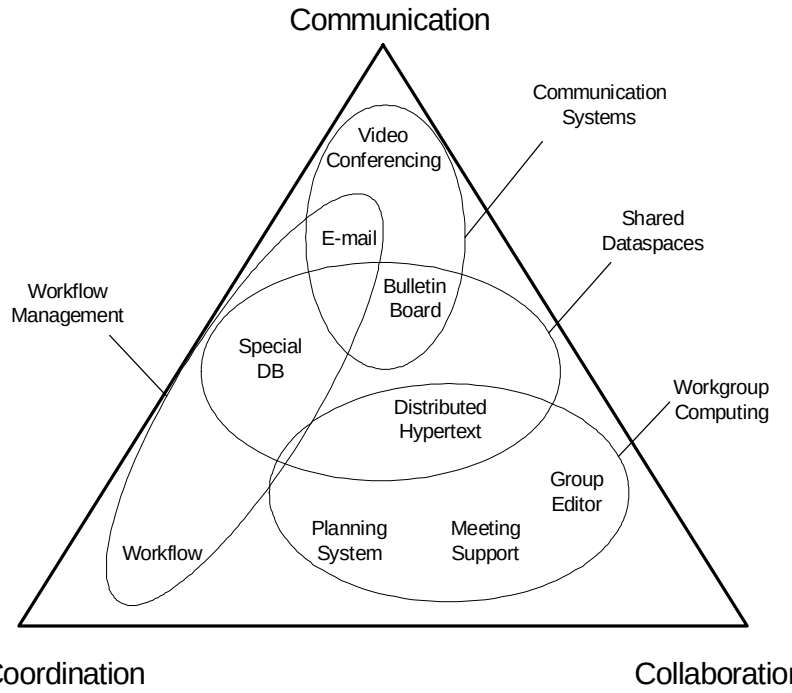

Figure 4 Classifying Groupware According to Communication, Coordination, and Collaboration

Shared data spaces provide people with the facilities to share data. Bulletin boards are used as a means of sharing data as in the case of Question & Answer forums on the Internet. Distributed hypertext links documents, making it possible for a group to connect their individual data to a larger web of information. Special databases are typically used to allow people work together by sharing information.

Workgroup computing helps small projects work together by focusing on the collaboration aspects. The most prominent example is a group editor that helps people jointly write one document, for example, the specification of a computer system. Meeting support helps people organize their collaboration; planning systems to coordinate this working together.

Workflow management systems focus on the coordination of people performing different tasks to create the final work product. Special databases are typically used as part of the workflow-based application; this means the activity implementations work with these databases. A lightweight version of a workflow management system can also be built on top of an e-mail system by using scripting capabilities within the e-mail system.

8.7 Different Views of Applications

Different people within an organization view the company's applications differently Figure 5 illustrates two major views.

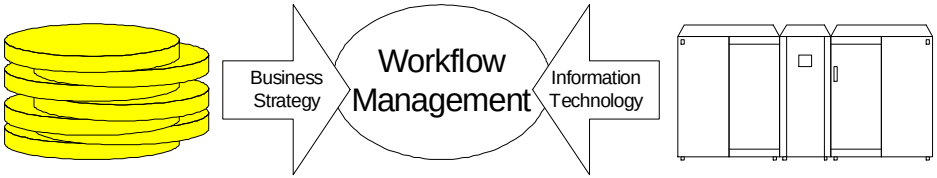

Figure 5 How different People View Applications

Business administration and management typically view applications as some kind of business process that is performed by different parts of the organization and that results in the delivery of the required item. From their perspective the mechanic of achieving this result is immaterial; their major concern is the efficiency of the business process to achieve the goals specified in the business strategy. This concern causes them to mainly focus on business goals, such as how long it takes to process a customer order, how satisfied the customers are with the company's service, o: how quickly a customer complaint can be processed. A second set of business goal: deals with the efficiency of business processes, such as how efficient the business process is in terms of costs and people.

In contrast to business administration and management, information technology (IT) personnel view applications as a combination of a vast amount of data stored in files and databases that are processed by a large amount of programs operating on the data and interacting with the users. These programs are so important for the enterprise that they are implemented as transactions. The IT people are mainly concerned with the availability and efficiency of the programs and the data those programs access. Consequently, they are concerned with the availability and efficiency of the underlying systems, such as database management systems, operating systems, transaction managers, or communication systems, on which those programs depend.

Workflow helps to bridge this gap because it connects these views into a set of workflow-based applications. These workflow-based applications consist of the process models that the business administration looks at and the activity implementations that are the transactions the information processing people see. Workflow thus helps to achieve a consistent and coherent view of applications.

8.8 Transactional Workflow

Applications that implement the core functions of the company's business are implemented as transactions. The ACID paradigm, the fundamental concept of transactions, when implemented makes these applications robust and safe. The ACID paradigm defines that a collection of operations within a transaction has the following properties:

- Atomicity-Either all of them are applied to the system or none of them are,
- Consistency-They lead to a new valid state of the system,
- Isolation-They do not affect (until explicitly made visible) operations out side the collection,
- Durability-They are not undone because of any later system failure.

The transfer of money from one bank account to another bank account is the prototype of a transaction. The transaction consists of two operations; the withdrawal of money from one account and the deposit of money into the other account. In this case, both operations must succeed or none of them may succeed (atomicity). As the amount of money remains the same, the new state is a valid state of the system (consistency). The new state of the two accounts is not available to any other operation, until the transaction has completed (isolation). The changes must not be lost under any circumstances (durability).

Workflows or parts thereof must also have transactional properties. Depending on the time required to execute this transaction, different techniques need to be applied.

In highly automated workflows, that is workflows with little or no user interaction, the individual activities are often carried out with programs that have transactional behavior. Because of the stream-like behavior (short-lived, no user interaction) of these workflows, collections of such activities can be grouped together to build a new transaction. Such a transaction is called a *global transaction*. Global transactions allow automated recovery in such workflows; whenever an error occurs, all effects of the different activity implementations are automatically undone. The definition of the transaction boundaries is part of modeling the workflow. We call such a set of activities that are part of the transaction an *atomic sphere*. A typical example of an atomic sphere is the funds transfer we discussed earlier. Enforcement of the global transaction, that is atomic sphere, can be achieved by the workflow management system using traditional techniques.

The typical execution time for an atomic sphere is in the range of tenths of seconds to seconds at the most. If the execution of a transaction takes longer, the standard techniques that are applied in atomic spheres can no longer be used efficiently. A number of advanced transaction models have been developed to cope with long execution times by weakening some of the ACID properties. Typically, those new transaction models require special programming by the application developer. This type of long-running transaction is called a *business transaction*.

The processing time of the transaction is measured in minutes or hours; however, none of the advanced transaction models can be directly applied here. The implementers of the different reservation systems had no knowledge of each other at the time they built the system, nor did they want to take this fact into consideration. Each of the systems performs its own transactions and exposes its results independently of each other. This behavior requires that the individual transactions need to be undone by compensation transactions that compensate the effects of the original transactions. The compensation of a flight reservation is the cancellation of the reservation. The definition of the business

transaction boundaries and the compensation activities is, as for global transactions, part of modeling the workflow. We call the set of activities that are part of the transaction a compensation sphere.

Figure 6 shows the space that global transactions and business transactions occupy in the classification scheme

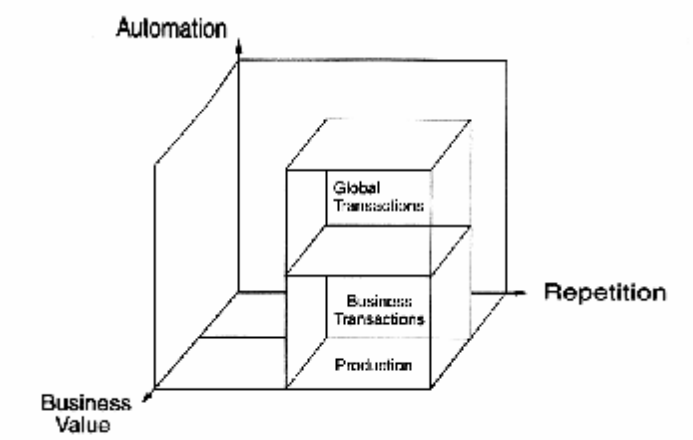

Figure 6 Classifying Business Transactions and Global Transactions

8.9 Project Management

Workflow technology not only applies to the traditional applications we have discussed so far, such as loan processing, it can also be used beneficially for other applications, such as for project management. Extra information on project management please refer attached complementary handout.

A project management system helps managers and team leaders plan and control the execution of projects. To support these tasks, it keeps a database in which all project-related data is stored.

At the beginning of the project, the user specifies all project-related information, such as the individual tasks, the sequence in which the tasks need to be performed, how much of the tasks can be performed in parallel, and the resources, such as people, to perform these tasks. With this information, the project management system generates the appropriate charts, in particular, the GANTT charts (project plan), that show which resource must be available at which time. In addition, the user can specify which resources are actually available at which time. This specification enables the project management system to generate an adjusted GANTT chart that reflects the actual allocation of resources. This chart is refined, by different resource allocations, until the project plan fits the desired time frame.

During the lifetime of a project, actual data, such as the actual time or resource it took to perform a particular task, is collected and entered by the project manager. This information makes it possible to control the project progress and take appropriate actions, such as allocating more resources, if required. The quality of the project plan depends on the actual data being accurate, complete, collected, and entered into the project management system as early as possible.

Figure 7 illustrates how workflow technology can significantly add to the capabilities of a project management system.

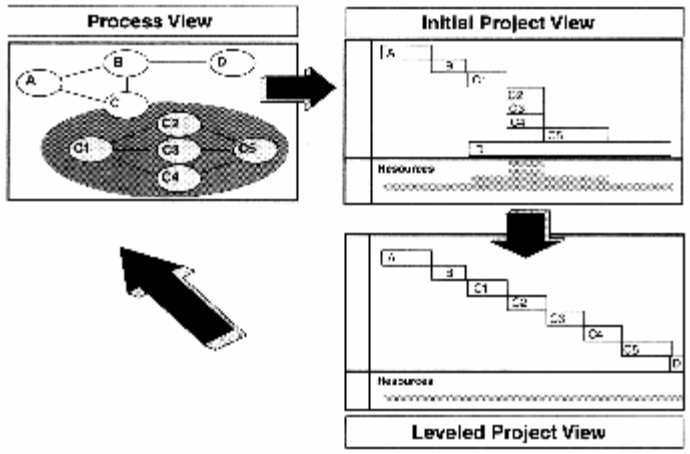

Figure 7 Project Management

One models a project in the workflow management system as a business process either by modeling it directly in the workflow management system or by reverse engineering it from an existing project plan in the project management system. We call this representation of a project the *process view*. When a new project is started, the process view is translated into the Gantt chart, which represents the *project view* of the project. This transformation is done under the assumption of unlimited resource and results in an initial project plan. The Gantt chart is then reworked using the available level of resources, until a satisfactory project plan has been found. We call the project to be leveled and the appropriate Gantt chart the *leveled project view*. When this stage has been reached, information in the Gantt chart that is process relevant, such as the amount and type of resources that are needed or the time scale of the project, is extracted and imported into the workflow management system. The workflow management system updates the process model with this information, for example, the assignment of roles to activities.

When the project is started, an appropriate process is created from the process model. This process now helps to control the execution of the process by proactively tracking the project progress. It collects the appropriate actual data from the project members and, if this data is not provided in a timely fashion. Sends out reminders or even notifications to the project manager.

Autonomy

A sub-process is a process in its own rights. It is derived from a process model that has been defined independently; that means it can be carried out as process in its own right. Even when executed as a sub-process, it executes independently of the parent process that created it.

Typically, autonomy rules define the rights the parent process has for the sub-process. It spans the whole spectrum from the sub-process being absolutely autonomous to the sub-process being totally controlled by the parent process. For example, if the sub-process is completely autonomous, actions such as terminating the parent process do not affect the sub-process.

Connected Discrete

In this model, the connected discrete, an activity within a process A connects to an activity in another process B. The execution of the appropriate activity in process A causes the creation and start of process B. Both processes operate independently of each other with no synchronization between the two processes. For obvious reasons, this model is also called a chained services model.

Figure 8 shows that end activity A5 causes the creation of process B. Whereas this may be a typical case, it is not required that the process is started from an end activity; any one of the activities of process A can start the sub-process.

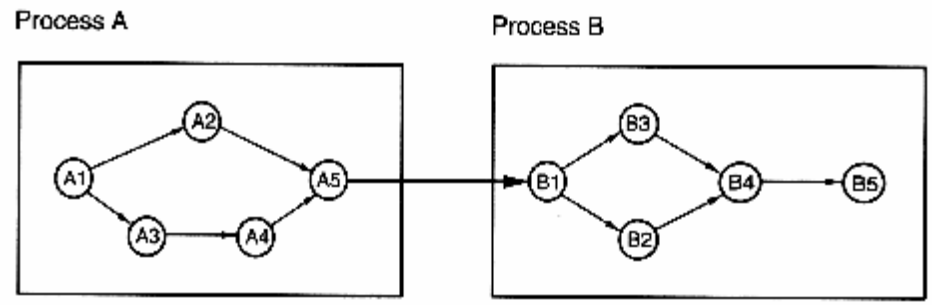

Figure 8 Connected Discrete Models

Hierarchical

In this model, the hierarchical, an activity within a process A represents the execution of another process B. The execution of the activity causes the creation and start of process B. Processing of the activating activity is suspended until process B terminates. After control has returned, navigation through the process graph continues.

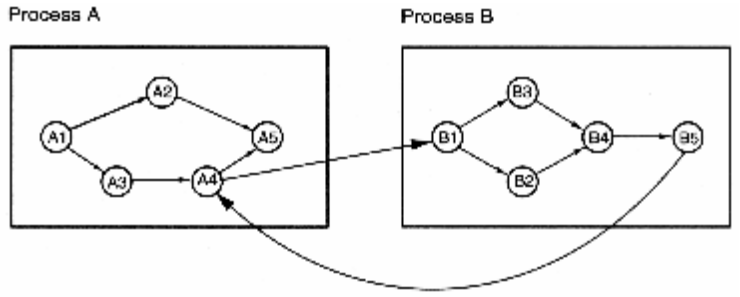

Figure 9 Hierarchical Model

Figure 9 shows such a model. Activity A4 represents the execution of process B. When A4 is executed, an instance of process B is created and started. When activity B5, the end activity of process B, completes, control is returned to activity A4. Now, navigation continues to activity A5.

As can be seen from the figure, there exists a hierarchical relationship between the calling process, the parent process, and the child process, the sub-process. People usually have this model in mind when talking about sub-processes. In programming, this structure would be called a nested structure and this model is also called a nested model.

8.10 System Requirements

In Chapter 1, we discussed process models as an enterprise resource. They therefore must be treated in the same way as data that the company owns. They must be protected against any accidental or intentional damage. Tight access control must be exercised to make sure that the structure of the business processes is not revealed to unauthorized users and that the structure of business processes cannot be altered with malicious intent. It is imperative that the business processes be executed exactly as defined.

Similar functionality has traditionally been delivered by transaction processing (TP) monitors. The structure of the business processes was an integral part of the application that was executed by the TP monitors. A production workflow management system therefore must provide the same level of operational and enterprise characteristics as that delivered by TP monitors. Actually, a production workflow management system must exceed these levels as workflow-based applications typically operate in a heterogeneous and distributed environment.

Operational Requirements

The following list highlights some of the more prominent operational requirements that a production workflow management system must fulfill.

- It must support *business transactions* and *global transactions*.
- It must be *reliable*. All internal operations must be performed as transactions. For example the invocation of an activity implementation, must be performed once and

only once. The workflow management system must ensure that in the case of a failure, no completed action is undone and all actions that have not yet been completed are undone and started anew.

- It must show high *availability*. Actually, the workflow management system should support continuous operation; that means that the management of the underlying data and the communication software should not require the system to be shut down even when changes, such as upgrading the different components of the workflow management system, are made to the workflow management system. High availability means that the workflow management system implements a system structure that allows the workflow management system to be up and running 24 hours a day 7 days a week.

- It must be designed for high *capacity*. Ideally, this would mean the support of an unlimited number of users working with the system and an unlimited number of processes maintained by the workflow management system.

- It must be *scalable*. This requires that the workflow management system be structured in such a way that additional resources either make the system perform faster or provide for higher throughput.

- It must provide the capability to *record process traces*. The trace information should include all relevant actions, such as the start of a process or the execution of an activity implementation. All trace information must be written to persistent storage. This trace information is usually called the audit trail. It allows for the monitoring of the system characteristics, such as the current number of actions being performed or the average duration of a process. It also helps check if the corresponding goals are met. And, it provides the data for statistical analysis, such as calculating the costs associated with carrying out a process.

Enterprise Requirements

Since workflow-based applications are typically executed in a heterogeneous and distributed environment, even within the same corporation, the workflow management system must also address issues that are important from an enterprise perspective.

- It must support *multiple platforms*. Platform refers to the operating system and network protocol the workflow management system operates on. In addition, it is required that the workflow management systems on the different platforms can communicate with each other. Platform also means that the workflow management system must be able to invoke applications that implement activities on different platforms using different invocation mechanisms.

  This flexibility is required to support the integration of a wide variety of existing programs, such as legacy applications running on a mainframe using IBM's Customer Information Control System (CICS) or IBM's Information Management System (IMS) as transaction monitor, or standard software such as SAP R/3.

- It must participate in *systems management*. This requirement has multiple facets. First, the workflow management system must be enabled to assist the systems management system in the distribution of workflow management system code to the various servers and clients. Second, it should assist the systems management system in the automatic distribution of the software each user needs to perform the assigned activities. Third, it must provide the systems management system with information about its status. This information allows the systems management system to monitor the workflow management system and take proper actions if the workflow management system becomes unavailable for whatever reason.

- It must provide *central administration*. Each one of the individual workflow management systems offers administration functions. These administrative functions fall into two categories: the administration of processes and the administration of the workflow management system. It is important that both types of functions are available centrally.

- Standards *compliance* is an important requirement for a number of reasons.

  1. In a large enterprise, it is likely that workflow management systems of different vendors are involved in the execution of workflow. This variety requires that the different workflow management systems implement some standard that defines the exchange format between different workflow management systems.

  2. Uncontrollable events, such as the disappearance of a workflow management system vendor, may require the installation of a new vendor's workflow management system. It is important that (ideally) the activity implementations and process models need not be adapted or even rewritten. This desired flexibility requires that the workflow management system honors the standard that regulates the invocation of activity implementations and complies with a common metamodel.

  3. Central administration requires that the workflow management system implements the standard that defines the information to be returned and the actions to be taken as the result of an administration request.

  4. Statistical analysis of process traces requires that the audit trail written by the different workflow management systems follow the appropriate standard. Otherwise, significant pieces of information are lost.

  5. Users typically interact with the workflow management system by using a graphical user interface (GUI). This GUI is very identical throughout the company. This standardization is particularly important in cases where different workflow management systems are installed within a company. User should not need to learn to use a new GUI every time they change jobs. This need for continuity requires that the workflow management system implements the standard that defines the functions that a client performs. Without this standard, the customer would need to invest in various differing GUIs.

- A sophisticated *security* system is a definitive must for a workflow management system. We subsume under this requirement all aspects of security and authentication. These aspects include the protection of the workflow management system's data against any access coming outside the workflow management system, the encryption of messages that flow between different workflow management systems, the authentication of users when they access the workflow management system, and an elaborate access control scheme that allows the granular assignment of access rights to workflow management system entities such as process models, processes, and audit trail information.

8.11 Relation to Other Technologies

So far, we have sketched how workflow is related to object technology, transaction management, and project management. There are many additional software technologies that are more or less closely related to workflow. Figure 10 lists some of the more prominent ones. We start our discussion with business engineering and then continue clockwise around the circle.

Figure 10 Relations to Technologies

Business Engineering is a collection of activities and techniques which helps define the optimal business processes for a company to make the company competitive. Those pieces of the collected information that can be managed by a computer are moved over to the workflow management system.

Development of applications is typically performed with the aid of an application development methodology particularly suited for the type of application that needs to be implemented. We call this approach process-based CASE (computer-assisted software engineering). This approach starts with the global definition of business processes, using one of the many business process modeling tools, and derives from the collected information the workflow specifications and the structures of the databases that are used by the activity implementation. Another approach is to start with object-oriented analysis

and design (OOA/D) and derive from this information the workflow specifications and database structures.

The Internet is the backbone of electronic commerce. It provides the mechanism that enables workflows to easily span companies all over the world. Users have access to workflow management system functions, such as starting a business process or querying the state of a particular business process, using their familiar browser, such as Netscape browser or Internet Explorer. For example, users can order a customized car and track its delivery.

Applications often operate on multiple databases, where each database is managed by a different database management system. The involved database management systems must provide a particular behavior if the application needs to be executed as a transaction. The database management systems must operate as resource managers in a two-phase commit protocol. Not all database management systems support this protocol, nor are all applications structured for the exploitation of these features: transaction integrity must be achieved differently.

Systems management systems help to manage the complete information technology infrastructure, including distributing code to the proper computers, observing the correct operation of applications, communication lines, and processors, informing operations people about out-of-line situations, and taking predefined corrective actions. Applications must be instrumented to provide the proper information to the systems management system and to accept requests from the systems management system. Workflow-based applications are no exception. However, no particular instrumentation is required from the activity implementations; the workflow management system provides the appropriate hooks into the systems management, for example, to allow the systems management system to perform workload balancing. We discuss these aspects in detail in Chapter 10. In addition, the workflow management system's knowledge about IT structures and organizational information can be easily used to assist the systems management system in its tasks. The workflow management system, for example, knows which user needs which piece of code. This information is required by the systems management system to perform code distribution.

Not all participants in a business process use office workstations; sales or marketing people need to have access to information when they are on the road. Thus, workflow management systems must support the mobile user by providing access to the workflow management system at any time.

8.12 Web Applications

The integration aspect we focused on in this chapter until now was integration of applications. Another aspect of integration becoming more and more important is stimulated by Internet technology in general and web technology in particular: the integration of processes and people.

At the beginning browser technology provided the capability to access static web pages only. These pages were Hypertext Markup Language (HTML) documents with fixed content that could be retrieved and rendered. In the next step, web pages and their

content could be dynamically constructed with CGI (Common Gateway Interface) scripts and, more recently, servlets. Basically, the browser requests from the web server the invocation of a particular program that returns a web page as result. Such a program can be simple, such as composing a web page out of fragments locally available on the web server, or more complicated, such as a program that invokes a transaction on the next tier to get the required data from an operational database Figure 11.

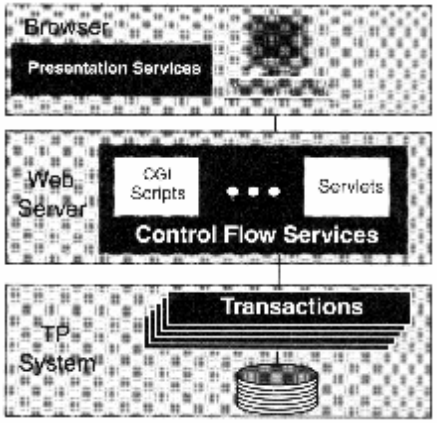

Figure 11 The Web as TP-Monitor Environment

By exploiting Java technology, a browser can even load applets, which are "little" programs that perform local functions such as the verification of user input. An applet can use the object bus of an ORB to access server objects. For this purpose, the client machine either has an ORB installed, or the required part of the ORB is loaded as an applet (then called "ORBlet") to the browser. Figure 12 shows some of the possible combinations. For example, a browser first loads a simple HTML page offering the user a menu. When the user selects a menu item, another HTML page is loaded together with Java code that verifies user input and finally invokes services from server objects via the object bus. The server objects in turn access data at the backend.

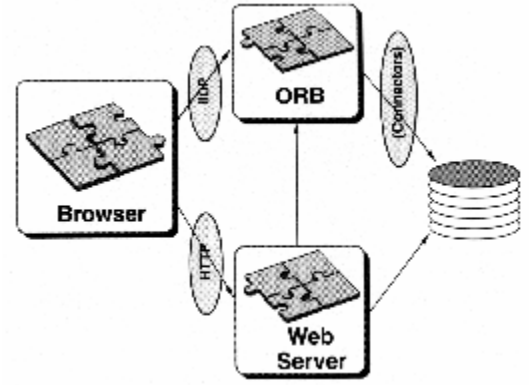

Figure 12 Three-Tier Structures on the Web

The browser thus performs some sort of presentation services. The web server or the ORB acts as a control flow service, determining which CGI script, servlet, server object, etc., to invoke. At the backend, transactions can be invoked to manipulate operational data. Based on this combination of web technology with Java, ORB, and transaction technology, the Web can be perceived as a gigantic TP monitor.

Browsers are ubiquitous. Theoretically, this ubiquity enables access to all functions and data available in an enterprise. What is needed to make the theory real is that an enterprise provides appropriate functions accessible from the Web. For this purpose, servlets or ORB server objects can be used from within appropriate web pages, as sketched above. With this approach, people can be closer to the services provided by an enterprise, customers can browse through online catalogs and place orders from the Web, sales persons can check availability of goods and their delivery dates even if at customer sites, employees can manipulate their worklists and start workitems even if off site, and so on. Also, people can be better connected with processes and processes between enterprises can be better coupled: for example, a customer can check the state of their orders; a company can start a delivery process at a supplier and monitor its state, and so on. Applications from the areas of customer relationship management, sales force automation, value chains, supply chains, etc., are obvious. These are only a few sample applications from the area of "electronic commerce." Figure 13 depicts a "virtual enterprise" scenario, a refinement from the one given in Chapter 1.

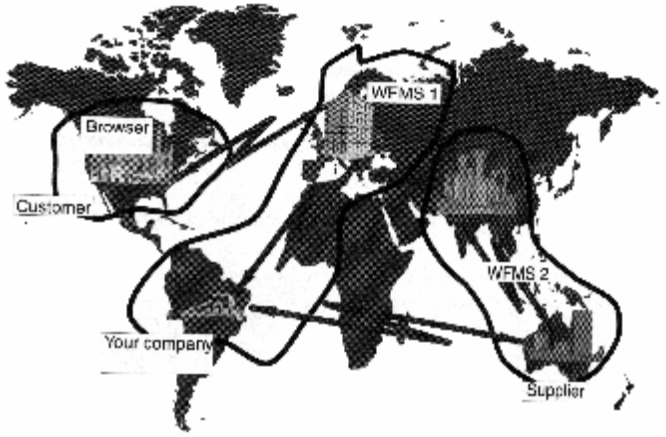

Figure 13 Virtual Enterprise

A customer browses an online catalog of a company. When he wants to place an order, he starts an associated order process. For this purpose, the browser may make available an applet that uses server objects at the company's ORB, implementing the OMG workflow management facility. The order process may be distributed, involving people from organizations in different locations of the company. One of the sub-processes of the order process may be outsourced to another company. To start and monitor this process, the workflow management system of the company uses an implementation of the WfMC interface 4 to communicate with the workflow management system of the company to whom the process has been outsourced. This process again can be distributed.

8.13 Workflow-based Applications

We discussed the relevance of flow independence for an application. By separating its business process aspects from its business algorithm, an application becomes very flexible. Instead of determining all applications affected by changes of a business process and modifying the corresponding application code (including all related efforts like compiling, testing, etc.), the business process itself is directly modified by changing the corresponding process model in the underlying workflow management system.

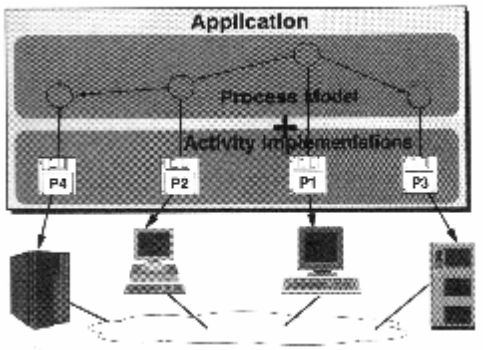

Figure 14 Structure of Workflow-based Applications

The resulting application structure is shown in Figure 14. The application consists of a collection of process models, such as credit applications, order placements, or trip reservations, and a collection of activity implementations, that is programs that encode business algorithms that support users in performing activities (e.g., entering customer data, computing interest rates, editing letters, etc.). Running the application means that the workflow management system navigates through the process models and invokes the activity implementations of the appropriate activities of the process models. Such an application is called a workflow-based application.

Furthermore, workflow-based applications have a lot more properties that are automatically inherited as the result of exploiting workflow technology and that are cumbersome to achieve an application is implemented completely with a programming language. A workflow-based application has the following characteristics:

Distributed. The workflow management system invokes an activity implementation wherever it is located. With activity implementations placed on different machines, the resulting application consists of pieces that run at different locations.

Heterogeneous. Different activity implementations can run not only on different machines but the hosting environments on these machines can be very different. For example, the operating system can differ from machine to machine; the activity implementations can run in native operating system address spaces or in TP monitors or on an ORB, they can be written in any programming language, they can comply with a variety of programming models; and so on. The workflow management system invokes the activity

implementations independently of any of these aspects. Thus, the resulting application consists of heterogeneous pieces.

Parallel. Activities on many different paths can be active within an instance of a process model. This is simply achieved by specifying multiple start activities of a process model or by specifying an activity with multiple outgoing control connectors whose transition conditions evaluate to true (or are simply specified to be constantly true). Thus, the resulting application runs multiple pieces in parallel.

Recoverable. Activities can be grouped into compensation spheres, and activities with transactional implementations can be grouped into atomic spheres. By using these unit-of-work concepts, the resulting application becomes backward recoverable. By simply using a workflow management system, the resulting application has a persistent context. This persistence automatically provides forward recoverability of the application to its latest context, and by exploiting safe activities full forward recoverability is achieved.

In particular, a workflow-based application is a distributed application. Finally, recall the other properties that the workflow management system provides for a workflow-based application, such as enforcing constraints like durations by means of notifications, or tracking the history of application executions with the aid of auditing and monitoring. A workflow-based application is much more than just a coded script invoking business algorithms!

Customization

Often, the business process aspects of an application must be changed if the application is used at different locations. For example, a company wants to use an application at different sites or countries it operates in, or an application is an off the-shelf application offered to a variety of companies. The origin of these changes is that different companies or sites of the same company follow different business processes even when deploying the same application (e.g., companies process orders differently), or that business processes are affected by country specific law (e.g., tax rates or auditing regulations vary from country to country).

Workflow-based applications allow companies to easily customize their business processes. To adapt a workflow-based application according to different business processes, the affected process models of the application are simply changed by use of the graphical build-time features of the underlying workflow management system. As a result, no coding efforts (and resulting tasks) are required for this aspect of customization. Consequently, the introduction of an application within different companies or at different sites of a particular company is (often dramatically) accelerated. Because of this inherent benefit, standard applications are building as workflow-based applications. It is interesting to note that depending on the coverage of a standard application, the overall system can be made up of many hundreds of process models and many thousands of activity implementations.

An activity implementation of a workflow-based application is flow independent. This means that (ideally) it makes no assumptions about its position in a series of invocations of activity implementations, about privileges related to its invocation, etc. Such an activity

implementation can be used in a variety of process models that means it is a granule of reuse. This kind of reusability is independent of object technology. When object technology is used for activity implementations, the activity implementations are methods of business objects; even business algorithms can be customized according to an individual company's need by overriding selective method implementations of particular business objects. Because this capability further increases the flexibility of a workflow-based application, some vendors of standard applications operate with business objects in this way.

Integration

The workflow-based application paradigm also supports application integration. It is the result of the capability of a workflow management system to invoke many different kinds of executables and to manage data flow between application pieces. Figure 9.21 shows a workflow-based application that integrates a variety of applications into a new application.

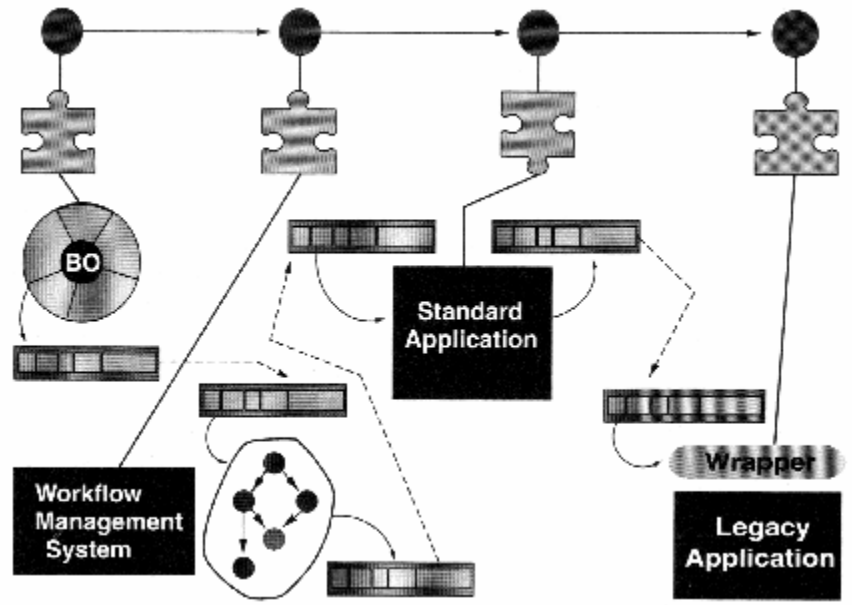

Figure 15 Integrating Applications Based on workflow Technology

The scenario assumed in Figure 15 is the following. A company wants to build a new application system with a minimum programming effort. They use existing applications as well as newly purchased standard applications. The functionality offered by the standard applications has to be modified and extended according to the company's needs. Extensions to the standard applications are provided by existing functionality of legacy applications, by business objects that are newly written or purchased, and by other workflow-based applications that the company already built. These functions are composed into the required application system by specification of appropriate process

models. The activity implementations of these process models are methods of the business objects, sub-processes matching other workflow-based applications, functions or processes provided by the standard applications, and functions externalized by legacy applications. When this application system is run, the workflow management system navigates through instances of the underlying process models and invokes the corresponding executables. It calls methods, starts sub-processes, invokes transactions or workflows within the standard applications, and calls legacy applications. The data needed by the different executables is passed, based on the specification of the data flow in the process models.

Wrappering

To exploit legacy applications in other applications (like workflow-based applications) the functions implemented by legacy applications must be invocable from the outside. The standard technique to furnish this requirement is to use so-called wrappers or gateways. A wrapper is a program that provides callable interfaces for selective functionality of the legacy application, as shown in Figure 16.

Such an interface can externalize a simple function, such as interface f8 in the figure, or a micro script implemented within the legacy application that already ties a collection of simple functions together, such as interface fl. The reason for not externalizing all simple functions is that in many situations only a certain invocation sequence of simple functions maintains consistency. This means, that the invocation of an arbitrary subset of simple functions of a legacy application is not meaningful.

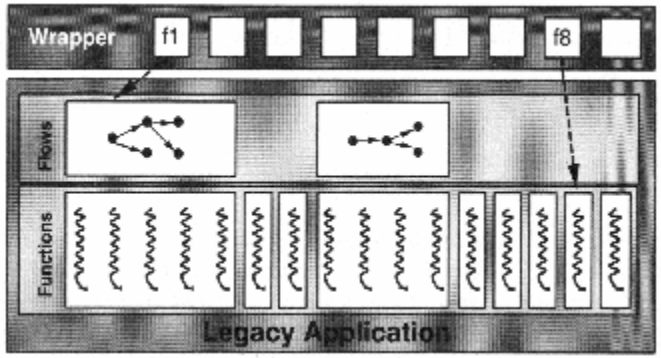

Figure 16 Preparing Existing Applications for Exploiting in Workflows

Note that a similar argument applies for standard applications. Although many simple functions of a standard application may be callable from the outside, separate interfaces are sometimes provided for integrating a standard application with other applications. Using these separate interfaces guarantees consistent manipulations of the underlying resources. In this sense, these separate interfaces are wrappers for a standard application.

Finally, wrapper techniques are often used for a lazy migration of legacy applications. By use of a wrapper, one externalized function after the other can be rewritten. Once a function callable via a wrapper has been rewritten, the new implementation can be called instead of the old interface provided by the wrapper. With workflow technology, this result is achieved by changing the specification of the program associated as an activity implementation. A program activity simply refers to the name of the program that implements the activity. The properties of this program (like the name of the executable to be called) is specified separately.

Changing the name of the executable to be invoked applies to all referring activities at once if the workflow management system binds programs late to activities; for programs bound early the affected process models must be translated.

Thus, the workflow-based application paradigm helps in reengineering legacy applications over time. Without time pressure functions of the legacy application can be rewritten and intermixed with functions that have not yet been rewritten yet. If all needed functions are newly implemented, the legacy application can be switched off.