Conception of Information Systems
Lecture 10: Workflow Management & B2B

17 May 2005
http://lsirwww.epfl.ch/courses/cis/2005ss/

Outline

- Overview
- Process Modeling
- Workflow Management Systems
- B2B Systems
- Summary
- References

1. Overview

- Process focus in modeling and integration of information systems
  - Data focus: e.g. relational data model and federated databases
  - Functional focus: function interfaces and transaction monitors
  - Behavioral focus: object models and object transaction monitors
  - Process focus: process models and workflow management systems
- Why focus on processes ?
  - At coarse granularity a **business process** describes best what a company is doing
  - **Business process re-engineering** is a practice to understand and optimize what a company is doing
  - At a coarse granularity large systems can be best described as processes (**programming in the large**)
  - The interaction of many different information systems and human users can be best captured by processes (**linking data islands**)

Up to now we have addressed the problem of integrating information systems by focusing on the following aspects of information processing:

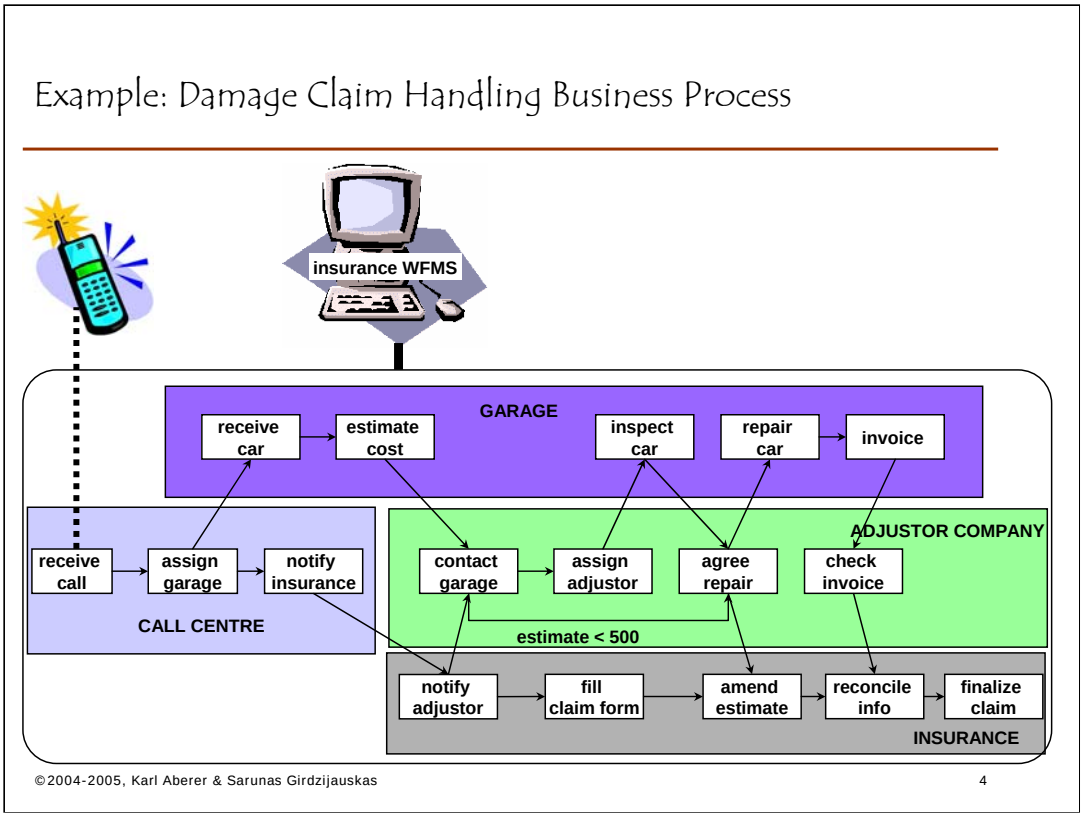
- Data-oriented, thus the "entities" we are integrating are data objects (with some standard behavior as provided by a DBMS through the DML)
- Function-oriented, thus the "entities" we are integrating are functions. Data was playing a secondary role, as it appears only in function interfaces as parameters. But the assumption was, that the function could change the persistent state of some underlying database and therefore the transaction concept was central.
- Behavior-oriented, thus the "entities" we are integrating are objects with a complex behavior (methods) and an encapsulated state. This allows to connect functions with their associated state.

With all the three approaches however one aspect is not addressed: the dependencies among the states of the objects/data and the therefore possible behavior, i.e. the underlying process. This aspect of an information system is represented in all of the three approaches implicitly, for example, as fixed life-cycles of components or objects.

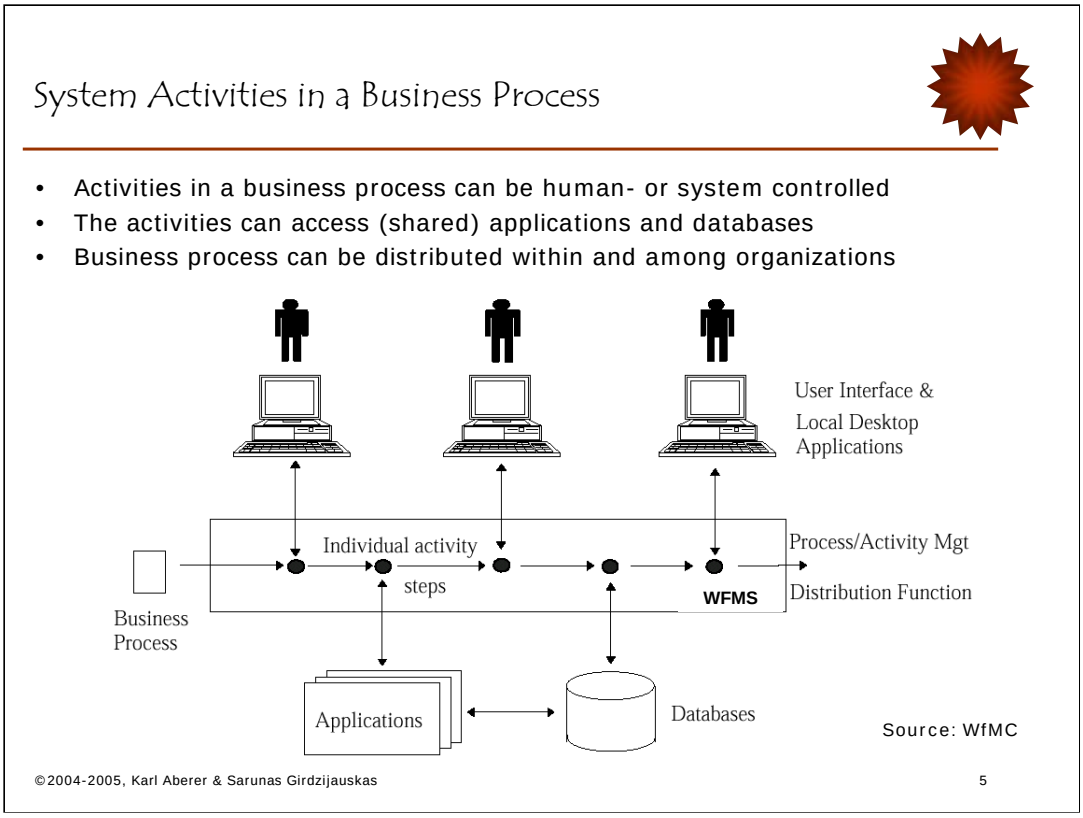
On the other hand, when looking at organizations (businesses), at a coarse granularity a process describes best what a business is doing. It captures, which actions can be performed under which conditions and in which situation (=state). This is the reason why a whole discipline in economic science, namely business process re-engineering, is taking a process-oriented view on businesses, in order to analyze and optimize their operation.

From a more technical perspective, processes are also a useful abstraction in order to describe the interaction of the components of large systems (not only information systems, but in particular), in a formal manner. For example, process models such as Petri-Nets, have been successfully applied in many domains in order to model complex technical systems (production plants, traffic, networks etc.). Therefore it appears to be natural to use such an approach also in the domain of integrating information systems at a large scale.

Since process models represent systems at a very abstract level, not necessarily referring to concepts related to programming or data representation (such as data types), they are also more suitable to integrate into the models more explicitly the human users that interact with the information systems and the activities they can perform.



This example illustrates of how a process model is typically used in order to capture the operation of a business. The example is taken from the insurance domain. The process describes the steps that are taken for processing an insurance claim. The process graph shows the dependencies among the activities, i.e. which activities must/can be performed upon the completion of predecessor activities and under which conditions activities can be performed (e.g. skipping the assignment of an adjustor, who should estimate the repair cost, if the estimate for the repair is below 500). The process state describes which activities have been performed. Notice that most of the activities are not necessarily performed by using an information system, at least in a traditional company. Thus the process description is orthogonal to the problem of implementing the process within an information system. However, many steps will involve information systems: for example, for assigning a garage a database is accessed, for estimating cost a computer program might be used, or for reconciling all information an information system might perform consistency checks. One also can see that different types of participants are involved (call centre, garage, adjustor, insurance).



From a systems viewpoint the business process consists of interdependent activities which are performed by both humans and information systems. The process is distributed both physically over the network and logically over different units of an organization and different organizations. Therefore, managing such a business process requires to address all the problem dimensions that we have identified for distributed information systems:

- Distribution of the activities over the network
- Heterogeneity, since the different application systems that are used can be heterogeneous
- Autonomy, in particular if different organizations are involved.

Traditionally, such a business process is managed by humans, adhering to certain organizational rules and using different forms of communication (e.g. phone, mail, email etc.). What workflow management now adds, is the computer-supported management of the business process. This implies, that there exists an information system, the so-called *workflow management system*, that knows the business process (i.e. the organizational rules) and that controls the execution of the different activities, either by interacting with human users through user interfaces that are running as desktop applications, or by interacting with backend systems such as applications or databases.

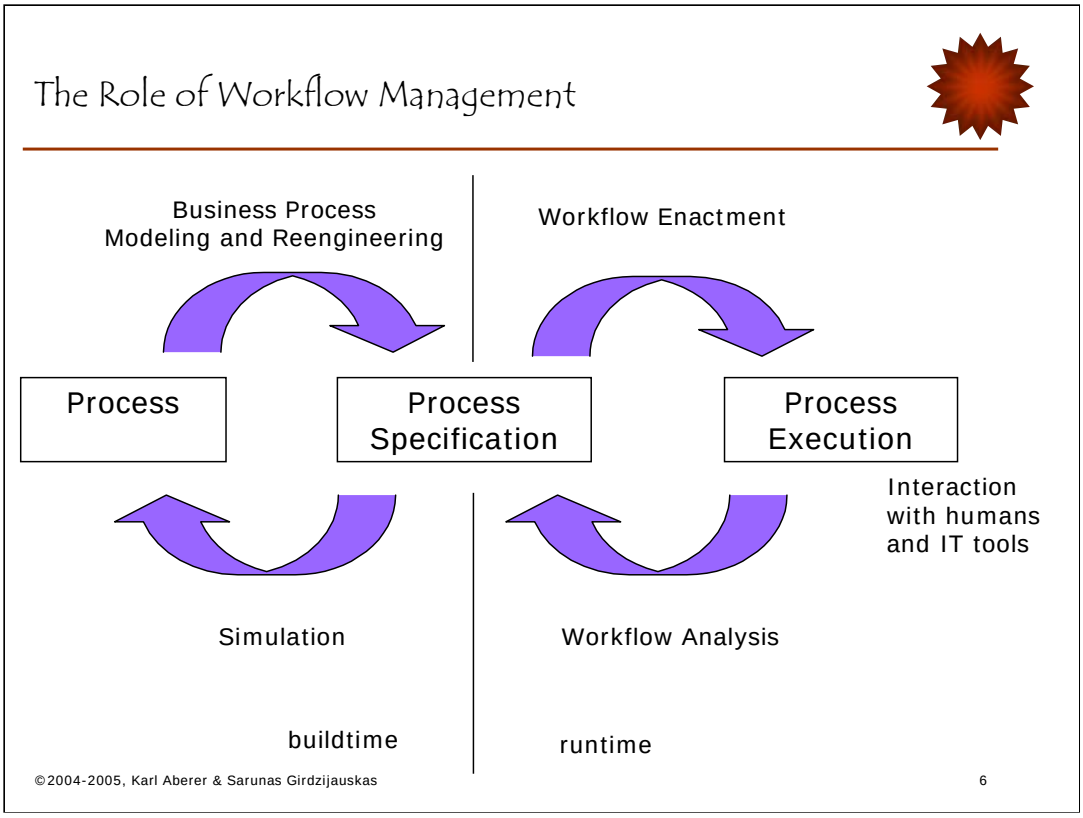
The benefits of Workflow Management Technology are manifold:

- It helps to organize, schedule, control and monitor processes in an automated fashion

- Being forced to provide formal models of the business processes helps to understand/improve them and allows to analyse, simulate, and reengineering them

- The implementation of the processes is made more efficient: It reduces paper work, supports on-line data entry where data originates, and supports data exchange and transactions across independent enterprises (EDI)

More generally workflow technology can be seen as "programming-in-the-large".



The automation of business processes using workflow management systems is part of a larger picture. By analyzing and formalizing business processes the following activities, each of them being a discipline of it's own, can be performed:

- *Business process modeling and reengineering*: This is the activity of analyzing the operation of businesses, of building process models and specifications, and use them in order to detect deficiencies and find optimizations.
- *Business process simulation*: Given a process model, there exist many tools in order to analyze them by means of simulation in order to find possible problems and optimization potentials in the process models.

The activities of business process modeling and reengineering and business process simulation belong mainly to the realm of economic science and are performed off-line, i.e. independent of the execution of the business processes in the real world. They are also not connected to the requirement that the execution of the business processes is automated by using a workflow management system. The following activities belong mainly to the realm of computer science. They are performed online, i.e. they are connected to the automated execution of the business processes. They can benefit substantially from models that originate from business process modeling and reengineering.

3. *Workflow Enactment*: given a formal specification of business processes they can be used to automate the execution of the process by using a workflow management system. This is the focus of this part of the lecture.

4. *Workflow analysis*: Having an automated execution of business processes, one can easily obtain data on their execution by logging them. The logged data on the process execution can then be used in order to analyze and optimize the processes. Note that this is different from simulation, where all the data origins from a simulation model, and not from real-world process executions.

Workflow Concepts 1

- An **Activity** is
  - a collection of events, a set of logically related operations or a bounded computation
  - that can be made to occur by a single logical actor (e.g., by a person with a given role) or a processing entity (information system, resource manager)
    - review a damage claim assessment
    - fill out a complaint form
    - perform a database transaction
- A **Process** is
  - a collection of activities related to a specific (business) case
    - each damage claim represents a single case
    - all damage claims (the same case) constitute the damage claim handling process

With respect to the notion of "process" we have to carefully distinguish, whether we are talking about a specific instance of a process, e.g. the handling of a specific insurance claim, or whether we are talking about the general business case, the handling of insurance claims in general. The distinction is exactly comparable to the distinction between a database schema and a database instance (or simply a database) for database systems. In general, when the notion of process is used, the "process schema" is meant, whereas a specific execution of a process is called a "process instance".

Activities are parts of a process and are operations that belong logically together. They are assumed to be executed atomically from the process viewpoint. The scope of what constitutes an activity is decided by the process designer. A necessary requirement for an activity is, that all its parts are executed by the same actor, but this is not a sufficient criterion. Also for activities the distinction between an activity (as part of a process) and an activity instance needs to be made. Typically, if activities are more complex, they can be decomposed into sub-processes, i.e. be composed from smaller constituent activities.

Workflow Concepts 2

- A **Workflow Process** is
  - the automation of a business process, in whole or part, during which documents, information, or tasks are passed from one participant to another for acting according to a set of procedural rules [WfMC]
- **Workflow Management** (WFM) is
  - the automated coordination, control, and communication of work both of people and computers in the context of organizational processes through the execution of software in a network of computers whose order of execution is controlled by a computerized representation of the business processes
- A **Workflow Management System** (WFMS) is
  - a technological system in which workflow processes are defined, performed, managed, and monitored through the execution of software whose order of events is driven by a process definition

These definitions have been given by the Workflow Management Coalition (WfMC), an industry consortium of workflow system vendors, that standardizes workflow system concepts, architectures and interfaces (similarly as X/Open or OMG). The definitions distinguish the

-general problem of automating business processes (workflow management)

-The automated business processes (workflow processes)

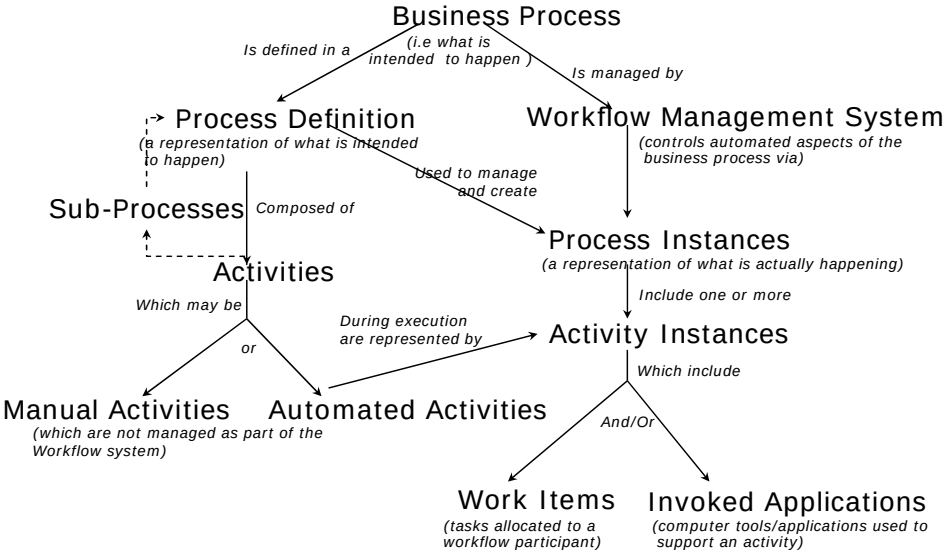
-And the systems for automating workflow processes (workflow management systems)

Workflow Concepts 3

- A **Work Item** is
  - a message that represents work to be performed.
  - is used to manipulate (e.g. relay, store, schedule) work for the purpose of doing it later or elsewhere
    - a fax asking about the damage claim assessment that should have been there 3 days ago.
- A **Data Item** is
  - An object that represents information, which is needed to perform some activity
    - a photograph of the damage, taken on location
- A **Work List** is
  - list of work items retrieved from a workflow management system.
  - a worklist is used by an actor as a to-do list

These notions are related to the way of how workflow processes are implemented. The notion of work item corresponds to the "unit of work" that can be exchanged among components in a workflow system. The units of work are related to the performance of activity instances. Typically, performing a work item requires some supporting data, which are then called data items. Work items are sent to the actors, that are supposed to perform them. Since actors may have at a given point in time multiple work items to perform, they maintain a worklist, that contains all the work items, that need to be performed.

Relationships among Concepts



Source: WfMC

This is an ontology for workflow management, that explains and relates the concepts we have introduced before and adds some refinements. In particular we see that

-activities can be defined through sub-processes

-The distinction between manual and automated activities is made

-The "invoked applications", that represent required application resources to perform a work item (not to confuse with applications that perform themselves a work item)

2. Workflow Modeling

- There exist almost as many workflow models as workflow systems
- Usually a workflow model distinguishes
  - Process definition
  - Resources
  - Resource management rules
- Process definition
  - Description of the process
  - Activities, usually executed atomically with internal state model
  - *Control and data flow among them*
  - Subprocesses
- Resources
  - Classes of resources that can be used to execute an activity

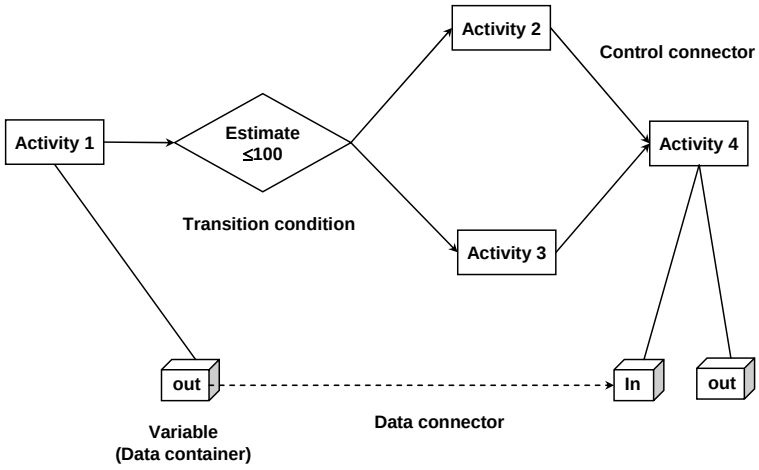
In what follows, we will discuss the standard process model, that is used in this or in a similar form by most major workflow systems. It corresponds to the model, that underlies the WfMC standards. Without elaborating too many details of workflow process models, we concentrate on some key concepts. Later we will see one concrete instance of such a workflow model, that is used by Weblogic.

The key component of each workflow model is the process model that describes the activities and their interdependencies. Both, for the process as a whole and for the activities, there exists a standard internal state transition model, In the simplest case such an internal state transition model would contain states "not enabled" – "enabled" – "running" – and "completed". Whenever an activity is completed the subsequent activities would change from "not enabled" to "enabled", and then eventually to "executing" and "completed". When the last activity is "completed" the whole process would change its state from "executing" into "completed". Thus a workflow process as whole can be viewed (typically and essentially) as a finite automaton.

The interdependencies between activities can be both expressed as control flows (i.e. enabling of activities based on the states that the activities and the process have) or as data flows (i.e. enabling of activities based on the availability of data).

Resources are required in order to implement activities. Normally the association between the activities and the resources, that can perform them, should be flexible. For example, the same agent in an insurance company could perform different types of activities (e.g. fill claim form, check invoice), and for the same activity (e.g. check invoice) multiple agents would be able to perform them. These dependencies are thus captured typically as rules.

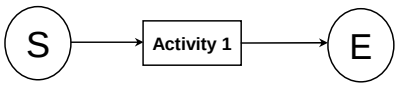
Process Definition: Control and Data Flow in Processes

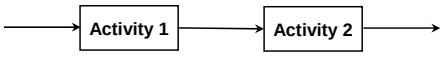


This example illustrates the difference between data and control flow in a process. The control flow is represented by the control flow connectors (straight arrows) in between activities. In the example, once activity 1 has finished (i.e. moves to state completed) a parameter "Estimate" is used to decide, which subsequent activity needs to be performed (activity 2 or 3). Once one of these two activities is completed, activity 4 can start. So the execution of the process is controlled by the "control flow". We see however that activity 4 needs in order to be able to work data from activity 1. This data is passed through variables (in workflow terminology normally called "data container"), that are connected through "data connectors" (broken arrow). In this particular example the control flow already makes sure, that the data is available once activity 4 starts. But assuming that activity 1 was executed independently with respect to the control flow, then activity 4 would have to wait for the availability of the data from activity 1 and the process execution would be controlled by the "data flow". Different workflow models use these two control mechanisms in different flavors (some are purely control-flow-based, others are purely data-flow-based and others support both ways). The more popular models and the WfMC model build however on control-flow.

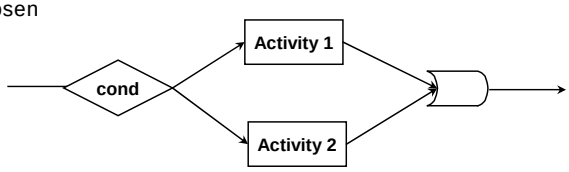
Control Flow (1)

- Each workflow has well-defined start and end node
- Sequential Execution





- Choice
  - One of the alternative activities executed
  - Decision made based on variable values
  - Workflow continues when the chosen activity is completed



Control-flow based process models consist always of similar basic elements (which originate essentially from the Petri-Net process model for parallel processes).

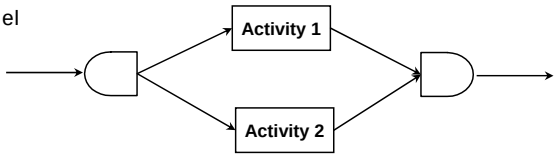
There exist uniquely determined start and end nodes of the process, such that the start and the termination of a process are unambiguously determined (i.e. there exists for example not the possibility to implicitly specify that a process has successfully terminated. This is important from a business perspective, since a business normally wants to be able to clearly identify when a certain job is finished.

The basic dependencies among activities is *sequential execution*, i.e. one activity can only start after another activity has terminated.

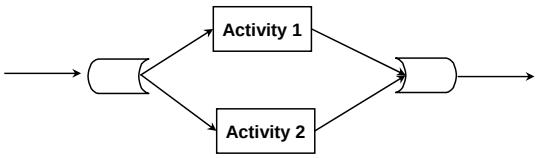
The choice constructor allows to implement decisions on how the execution of the process should continue based on workflow variables, that are accessible to the activities. Actually the parameters can be either passed as parameters from one activity to the next, or they can be obtained from globally accessible data items. Different workflow models use different approaches here. Once the decision is made the selected activity is performed and after its termination the workflow process can continue.

Control Flow (2)

- Parallel Execution (AND)
  - Both activities executed in parallel
  - Workflow continues when both activities are completed
  - AND-split and AND-join



- Parallel execution (OR)
  - Workflow continues when one activity completed
  - OR-split and OR-join



- Constraints
  - No Loops
  - Hierarchical composition of parallel constructors

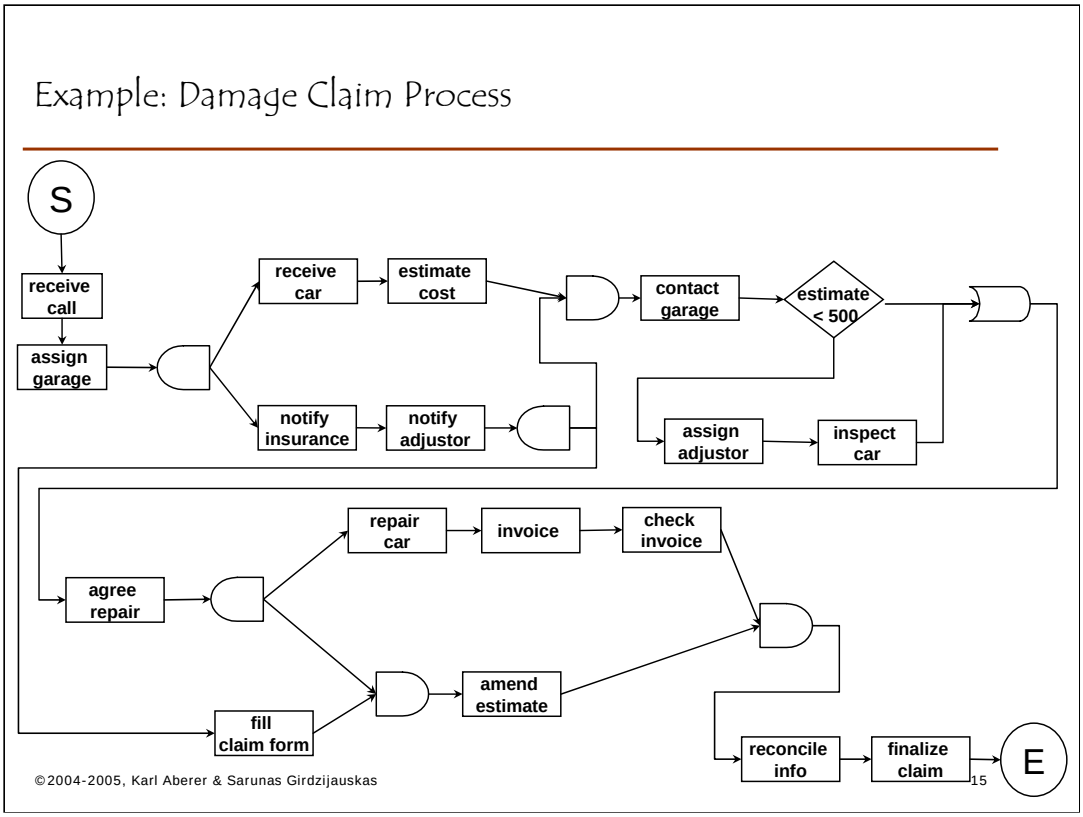
Multiple activities can also be executed in parallel. Then a precise specification needs to be given of what is happening after the multiple activities terminate. Their parallel execution has to eventually join into a single control flow. This guarantees that the workflow will end up in one final state. Thus multiple activities in parallel, starting from a *split node*, can only be introduced by a constructor that has a common outgoing control flow arrow. This is ensured by using a *join node* that joins the control flow.

With respect to parallel execution of activities there exist two possibilities: either the outgoing control flow connector becomes active (i.e. the workflow can continue) after all the parallel activities are terminated (then we have an AND-split and an AND-join), or it can already continue, when at least one activity has terminated (then we have an OR-split and an OR-join).

There exist typically also constraints on how these constructors can be combined:

•An frequently occurring constraint, is that loops are not allowed and thus activities are performed at most once in a process.

•The parallel execution constructors often can only be composed in an hierarchical manner, i.e. the split and join nodes must be hierarchically nested, similarly as start and end tags in an XML document must be nested and cannot be interleaved.



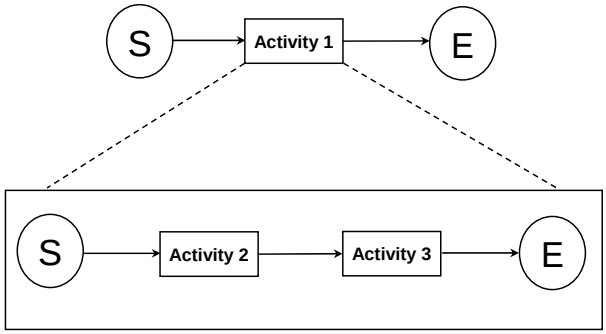
This is the insurance example process from before modeled in a more precise fashion using the control flow constructors we have introduced.

Notice of how the parallel execution elements are interleaved (thus the process model is not hierarchically nested with respect to parallel constructors): For example, after "notify adjustor" is completed the activity "fill claim form" can start, whereas the "activity "contact garage", requires the completion of both "estimate cost" and notify adjustor". However the activities "contact garage" and " fill claim form" should be executed in parallel, therefore the AND-split following "notify adjustor" is interleaved with the AND-join before "contact garage". The outgoing control flow arrow of the first is used as an incoming control flow arrow for the second.

Some models require that the parallel elements are nested in a hierarchical manner, which obviously limits the possibilities to model certain situations as the one shown in this example.

Sub-workflow

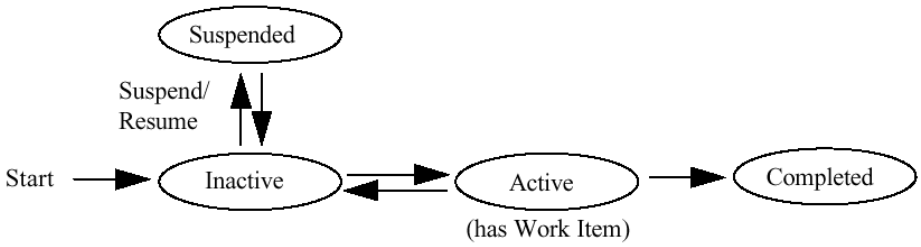
- A workflow activity itself can be a whole sub-workflow
  - The sub-workflow has one ingoing and one outgoing control connector



Each activity can be itself a complete sub-workflow. The requirement, that each process definition has a uniquely determined start and end, is also of importance for this construction. In this way the sub-workflow can be unambiguously mapped into a single activity.

Internal States of Workflow Activities

- A workflow process is instantiated in runtime
  - Multiple instances of the same workflow (cases)
  - State is given by the currently active activities and their internal states
  - Multiple instances of the same activity types can be instantiated
- Internal states of activities



Source: WfMC

The state of a workflow instance is given by the states of all the activities (this slide) and the global state of the workflow instance (next slide).

The internal states of the workflow activities according to the WfMC standard are:

-*inactive* - the activity within the process instance has been created but has not yet been activated (e.g. because activity entry conditions have not been met) and has no work item for processing

-*active* - a workitem has been created and assigned to the activity instance for processing

-*suspended* - the activity instance is quiescent and will not be allocated a workitem until returned to the running (inactive) state. This is useful to temporarily inhibit the activity to do any processing.

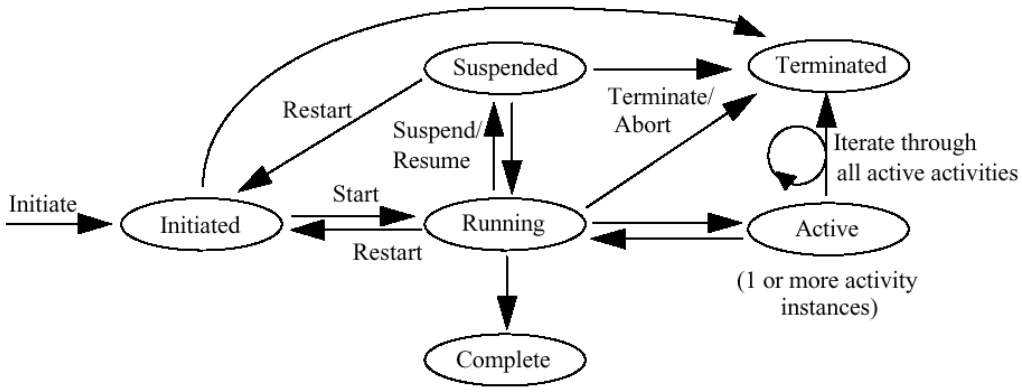
-*completed* – the execution of the activity instance has completed and any post-activity transition conditions will be evaluated for proceeding to the next activity

A workflow management system manages many instances of the same workflow process at the same time. For each of those instances it has to maintain the internal states of all activities and the global state of the process.

Remark: One can see that in particular an activity that has completed can no more be activated. For workflows with loops this would mean that for a further performance of the activity a new activity instance would be created.

Internal States of Workflow Instances

- A workflow instance has also a (global) execution state associated with



Source: WfMC

The basic states of a workflow process considered as a whole are according to the WfMC specification:

•*initiated* - a process instance has been created by the workflow management system, including any associated process state data and workflow relevant data, but the process has not (yet) fulfilled the conditions to cause it to start execution

•*running* - the process instance has started execution and any of its activities may be started (once any appropriate activity start conditions have been met)

•*active* - one or more of its activities have been started, i.e. a work item has been created and assigned to an appropriate activity instance.

•*suspended* - the process instance is quiescent and no activities are started until the process has returned to the running state (via a resume command)

•*completed* - the process instance has fulfilled the conditions for completion; the process instance will be destroyed

•*terminated* – the execution of the process instance has been stopped before its normal completion

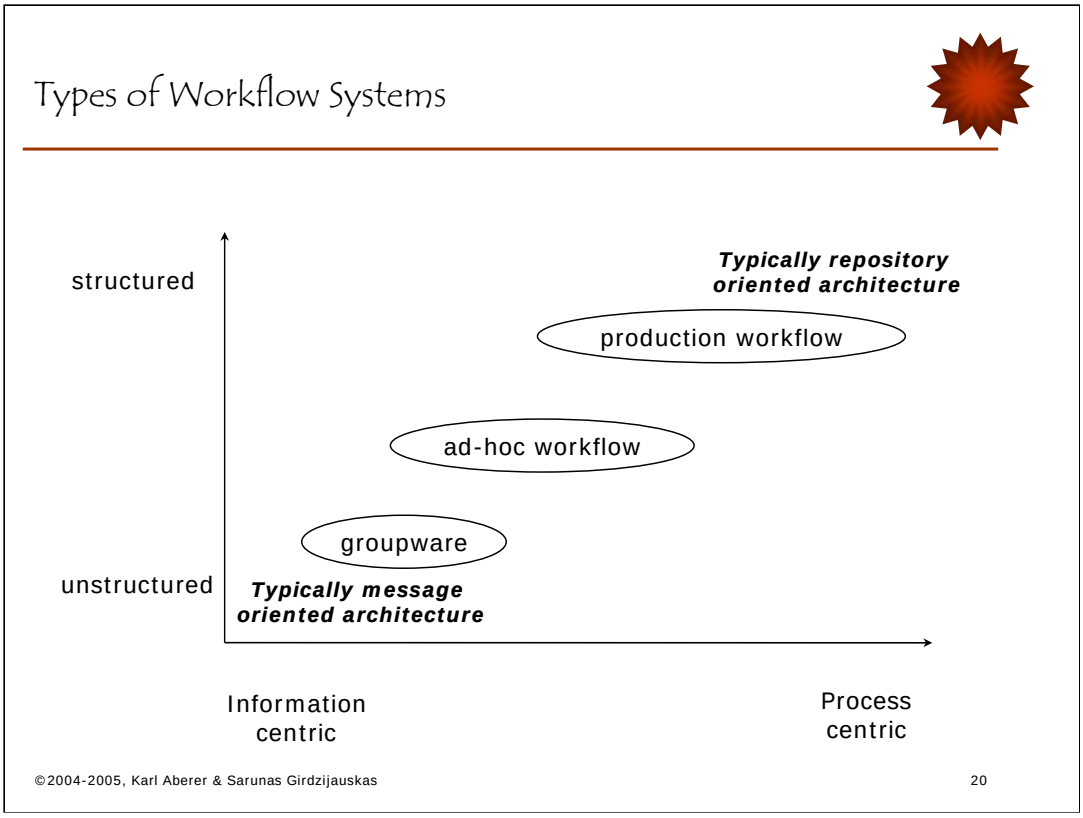
Resources

- Resource (participant, actor, user, agent)
  - A resource can execute certain activities for certain cases
  - Human or non- human (printer, modem)
- Resource class is a set of resources with similar characteristics
- A resource class is typically based on
  - Role (skill, competence, qualification)
    - Classification based on what a resource can do
  - Group (department, team, office, organizational unit)
    - Classification based on the organization
- The workflow specification contains rules that specify by whom an activity can be performed

For being able to formulate abstract rules on the association of resources with activities, workflow models provide typically two kind of mechanisms to classify resources:

- Implicit: by describing the properties that are required from the resources (typically captured by attributes). This is called *role-based classification.*
- Explicit: by enumerating the resources that belong to a specific class. This is called *group-based classification.*

Given these classification classes can then be associated with the performance of specific activities.



To position different types of workflow systems it is useful to consider two dimensions:

-structured vs. unstructured processes: this distinction relates to the "amount of structuring" that is imposed on the execution of activities in a business. More precisely, "amount of structuring" relates to the degree, to which the execution of activities is constrained by dependencies among them.

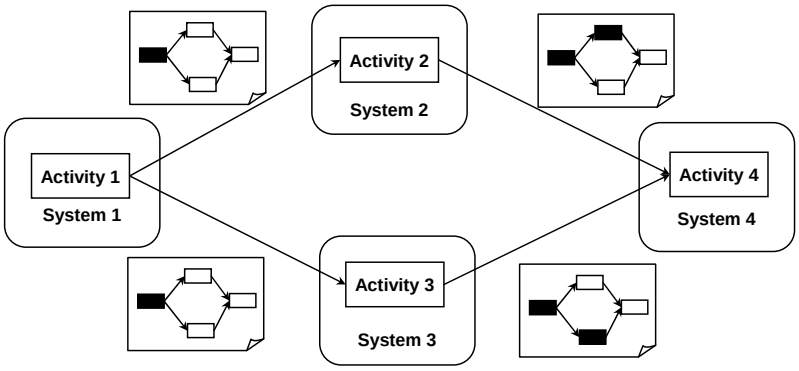
-Information vs. process centric workflow system: information-centric workflow systems concentrate mainly on the data flow, in order to control the execution of activities, whereas process-centric workflow systems concentrate mainly on the control flow in order to control the execution of activities.

In a typical groupware system, such as Lotus Notes or simply email, there exist generally very few constraints on the activity executions. Activities can be performed very freely and the groupware system provides the necessary communication infrastructure to connect different activities. Since these systems are mainly focused on data flow they normally adopt a message oriented architecture, where different activities are coupled through message exchanges and the data flow dependencies are establish through the flow of data with the messages. On the other end of the spectrum we have production workflow systems, which impose a rigid process structure, i.e. enabling conditions for activities are completely specified. They are used in the classical application areas for workflow technology, such as insurance, logistics or banking. These systems typically follow a repository-oriented architecture, where a central repository coordinates the workflow execution based on a process specification.

In between there exist different variants of so-called *ad-hoc workflow systems*, that use in principle a complete specification of the workflow process, as in production workflows, but allow in runtime to deviate from the process structure under certain circumstances, e.g. in case of exceptional events. This is motivated by the need of some application areas, such as hospital workflows, which in principle are well structured, but in exceptional cases, e.g. emergency, require the ability to deviate from the pre-defined processes.

Message-oriented Architecture

- Flow of structured messages between performers/users
  - messaging model
- Process definition part of messages
  - usually not stored in a repository
- No explicit process definition => difficulty in
  - knowing status (monitoring)
  - reuse



As we have mentioned, there exist two main types of workflow system architectures

1. Message oriented (the “lightweight” approach), where the workflow process definition is part of messages that are passed between activities.

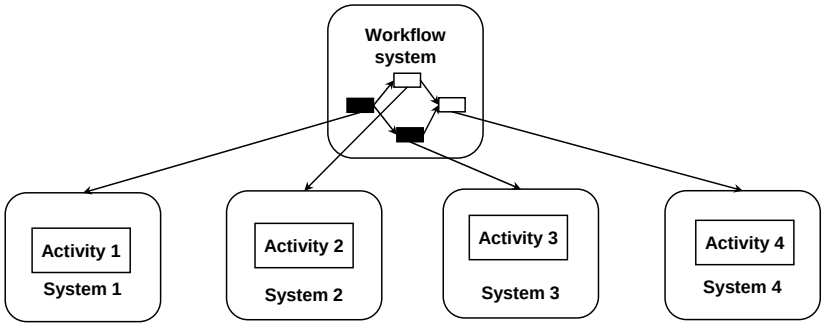
2. Repository oriented (the “heavyweight” approach), where the workflow process definition is stored in a repository/database and the workflow execution is controlled from there.

The trade-offs between these architectures lie in the infrastructure and technology needed, the robustness of the workflow system, which is usually higher in production workflow system, and ease of modification, which is typically better in message-based workflow systems. We have in the following a more detailed look at these two system architectures.

The important point with message-oriented architectures is that they do not require a central coordinator, that controls the workflow process. Rather the specification of the process is (more or less explicitly) stored in the messages themselves, and each activity implementation is able to interpret the message properly: this requires to identify which are the activities to be performed, and which are the subsequent activities to which messages need to be delivered in order to continue the workflow process. This approach benefits from a high flexibility and from avoiding a central coordinator, which could be bottleneck or a single point of failure for executing the workflow process. On the other hand, this approach might suffer from the problem of not having a global view of the process state and thus the inability of determining it's status (e.g. whether the workflow is still alive or completed).

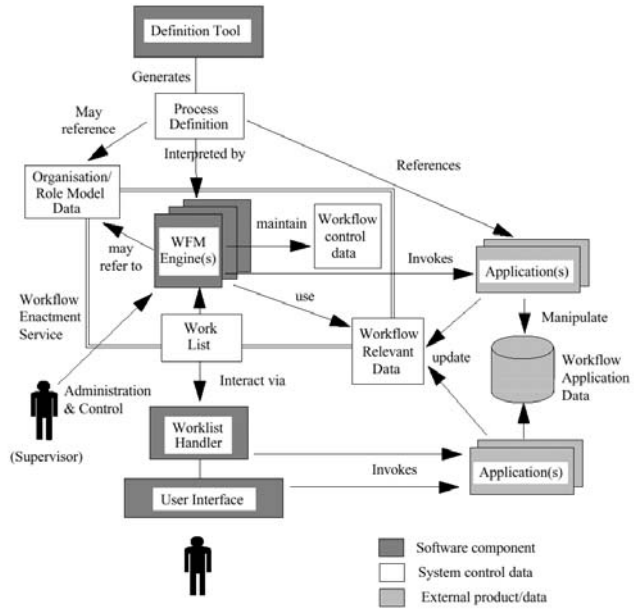
Repository-oriented Architecture

- Modeling and execution of process components
- Formal process model
- Both local view of individual execution component (worklist) and of complete process
- Repository of definitions to support reuse
- Use of databases or common/shared storage
  - usually supporting monitoring, tracking



In the repository-oriented architecture the process state is kept in the workflow system (which implements the repository) and the activities are coordinated by the WFMS. This provides a global view on the process execution and thus more possibilities to monitor and control it. This architecture also relies on explicitly represented process models (which is not always the case in message-oriented architectures), which fosters reuse and evolution of process specifications.

Implementation Architectures (repository-oriented)



Source: WfMC

The WfMC bases its standard workflow management system architecture on the repository-oriented approach. It defines a standard WFMS architecture for the implementation of a workflow management system, with the goal that different vendors can supply compatible components, that can work together through standardized interfaces. The different components in this architecture are the following:

Process Definition Tool: The process definition tool is used to create the process description in a computer processable form based on the formal workflow process model. The definition tool may be supplied as part of a specific workflow product or may be part of a business process analysis product, which has other components to handle analysis or modeling of business operations.

Process Definition: The process definition contains all necessary information about the process to enable it to be executed by the workflow enactment software (see the section on process models). The workflow enactment service then has the responsibility of linking roles with the specific participants within the workflow runtime environment.

Workflow Enactment Service: The workflow enactment software interprets the process description and controls the instantiation of processes and sequencing of activities, adding work items to the user work lists and invoking application tools as necessary. This is done through one or more co-operating workflow management engines, which manage(s) the execution of individual instances of the various processes. The workflow enactment service maintains internal control data either centralized or distributed across a set of workflow engines; this workflow control data includes in particular the internal state information associated with the various process and activity instances under execution. The process definition, in conjunction with any (run-time) workflow relevant data is used to control the navigation through the various activity steps within the process. The workflow engines also include some form of application tool invocation capability to activate applications necessary to execute particular activities. The generality of such mechanisms may vary greatly, with some simple systems only offering support of a single fixed tool such as a form or document editor, whereas others may provide methods for the invocation of a wider range of tools, both local and remote to the Workflow engine.

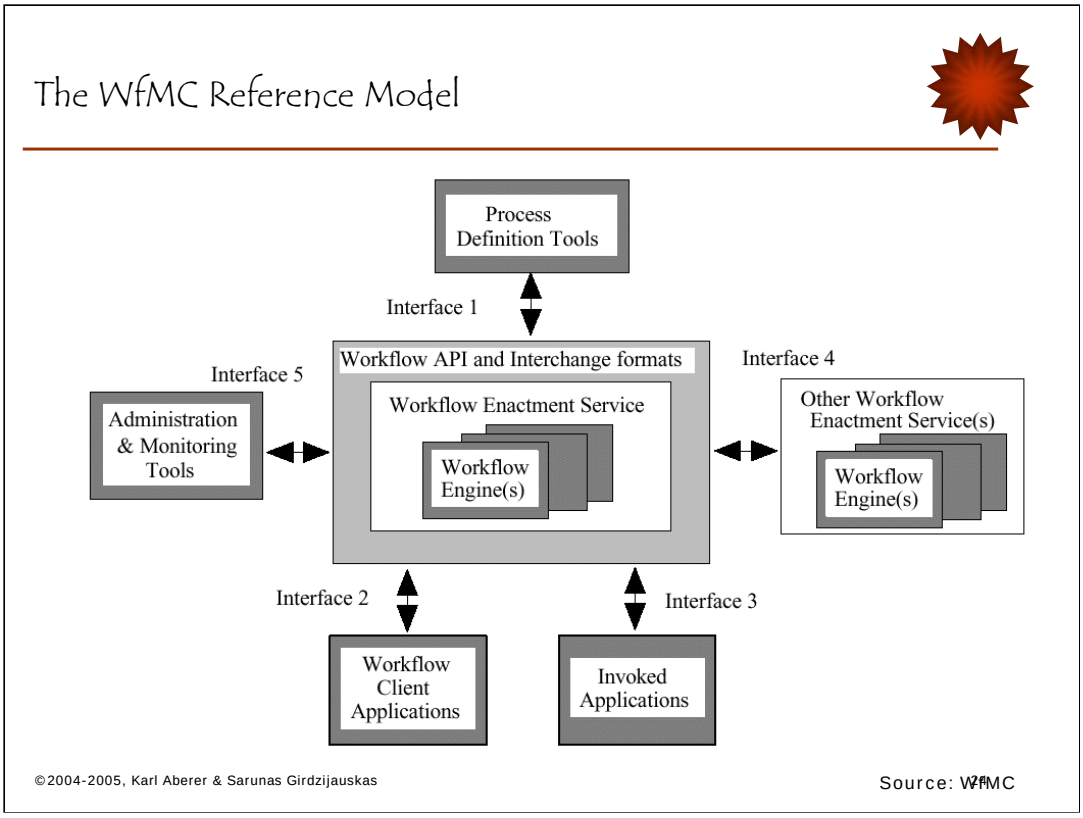
Workflow Relevant Data and Application Data: Where process navigation decisions, or other control operations within the workflow engine, are based on data generated or updated by workflow application programs, such data is accessible to the workflow engine and termed *workflow relevant data*; this is the only type of application data accessible to the workflow engine. Workflow application data is manipulated directly (and only) by the invoked applications, although the workflow engines may be responsible for transferring such data between applications if necessary, as different applications are invoked at different activity points within the workflow process.

Worklists: Where user interactions are necessary within the process execution, the workflow engine(s) place items on to worklists for attention by the worklist handler, which manages the interactions with the workflow participants.

Worklist Handler & User Interface: The worklist handler is a software component which manages the interaction between workflow participants and the workflow enactment service. It is responsible for progressing work requiring user attention and interacts with the workflow enactment software via the worklist. Within the reference model the term *workflow client application* is used in preference to "worklist handler", which includes process control functions as well as worklist manipulation.

In the diagram the User Interface is shown as a separate software component, responsible for the look and feel of the user dialogue and control of the local interface with the user. Invocation of local applications may be necessary to support the user in the particular tasks to be undertaken. There is a distinction between application invocation at the Worklist Handler/User Interface (which is not directly controlled from the workflow engine and may not be visible to it) and direct application invocation by the workflow enactment software.

Supervisory Operations: Within a workflow system there are a number of supervisory functions which are normally provided. These functions may enable supervisors to alter work allocation rules, to identify participants for specific organizational roles within a process, to track alerts for missed deadlines or other forms of event, to trace the history of a particular process instance, to enquire about work throughput or other statistics, etc.



Having defined the general WFMS architecture the WfMC derives from that different interfaces that are required to connect different components of the architecture. These are given by the WfMC Reference Model. The interfaces are in detail

Interface 1: Workflow Definition Interchange: This interface supports the exchange of workflow definitions between modeling and runtime tools based on a process meta-model specified by the WfMC. It consists of standard API calls to read an write process definitions.

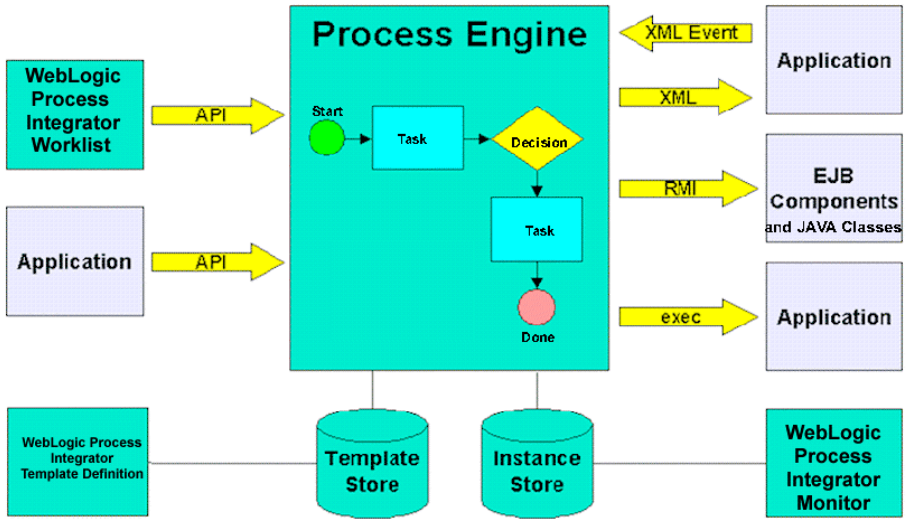
Interface 2: Workflow Client Application Interface: This interface supports the access of applications to the workflow engine and the worklists. It provides the client process and activity control functions.

Interface 3: Invoked Applications Interfaces: This interface needs to be implemented by workflow-enabled applications. It allows the workflow engine to start/suspend/resume/abort activities, receive notifications of the workflow engine, and perform data handling.

Interface 4: Workflow Interoperability Interfaces: This interface supports invocation and synchronization of processes and activities across workflow engines (see next slide)

Interface 5: Administration and Monitoring Interface: This interface allows the administration and monitoring tools to perform user/role management, audit management (event log), resource control and process supervision and status querying.

Example: Weblogic Process Integrator (J2EE)



Source: BEA

The Weblogic Process Integrator is an example of a fairly classical repository-oriented WFMS. It is based on a standard control-flow oriented process model. It implements all the main components of the WfMC architecture. It provides different interfaces to applications:

- An RMI interface in order to use EJB components or Java classes, that are executed on the Weblogic server for the implementation of workflow activities.
- An exec interface in order to invoke external applications.
- An XML message based interface in order to communicate asynchronously between the workflow engine and other applications. This interface allows to make the execution of activities dependent on external events, and among others can therefore be used in order to implement the synchronized model of workflow interoperability and to make the workflow processes interoperable with applications that use message based interfaces, such as EDI.

Weblogic Process Integrator Process Model

Table 1-1 Workflow Template Definition Nodes

Symbol	Shape Name	Purpose
	Start	Indicates the start of the workflow.
	Task	Defines a task in the workflow.
	Decision	Represents a condition in the workflow that evaluates to True or False. (If the condition is verified, it is True. If it is not, it is False.)
	And	Allows joining of one or more task, decision, or event with an AND condition.
	Or	Allows joining of one or more task, decision, or event with an OR condition.
	Done	Indicates the end of the workflow.
	Event	Represents an event that can be triggered either internally or externally by an XML message. Subactions can be performed and/or workflow variables can be set as the result of the trigger of the event.
	Connector	Used to connect workflow nodes. The arrow directs you to the subsequent task in the flow. **Note:** In the case of a Decision, when the arrow is drawn from a Decision shape, the Create Connection dialog box prompts you to specify whether the connection is True or False.

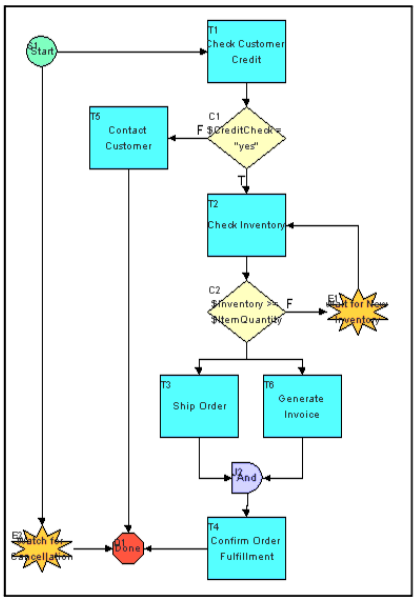
Source: BEA

The workflow model of Weblogic Process Integrator is a standard control-flow-oriented process model. Note that activities are called tasks. The representation of parallel execution and choice of activities differs slightly from the representation we have introduced earlier. The split operator is always implicitly represented by allowing tasks to have multiple outgoing connectors. Decision operators can be arbitrarily introduced. Different to many other workflow models, Weblogic also allows loops in the process model.

A special feature is the event constructor, which enables synchronization of the process with XML message based interfaces, either running internally on the Weblogic server or from external sources. The event can be considered as a special activity, that once it is activated, is running till it receives an XML message (from a further specified source). Once it has received the message this activity completes and enables the subsequent activity (see also the example on the next slide)

Events in Weblogic Process Integrator

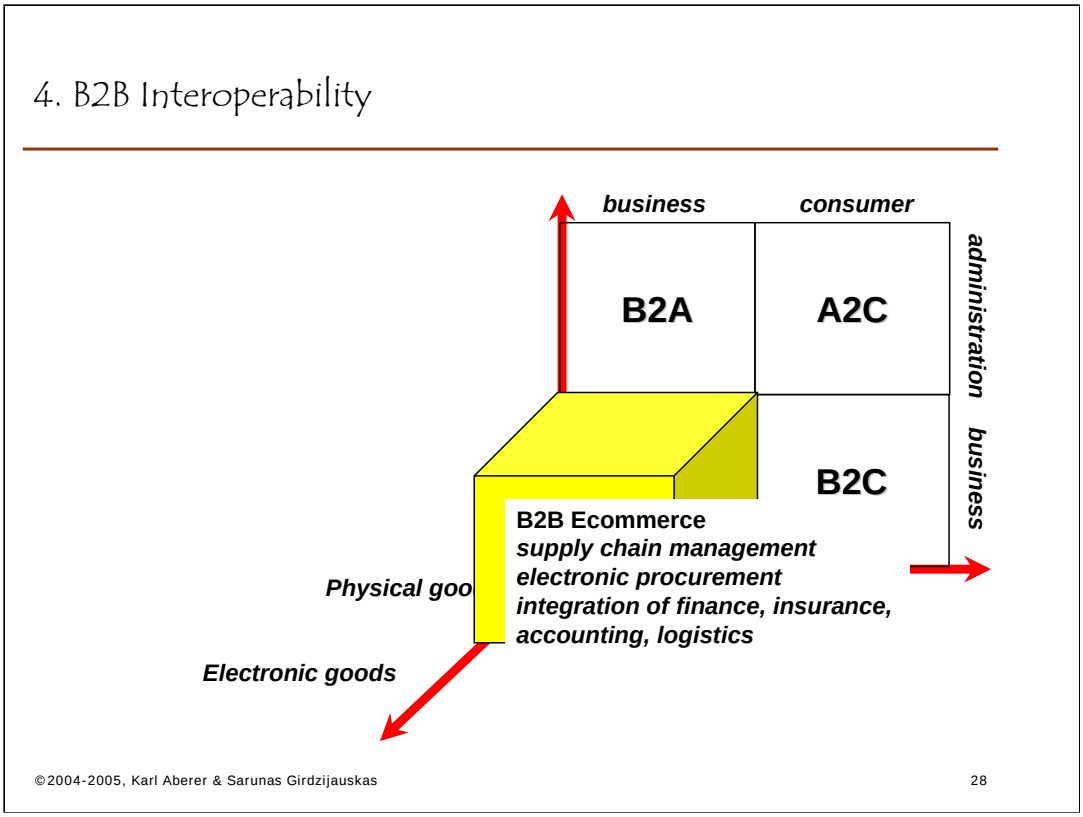
- The event node allows a workflow to enable control connectors based on the occurrence of (external) events
  - Events can be incoming XML messages
  - Interoperability with other systems (synchronised model)



This example of an inventory management demonstrates the use of the specific features of the Weblogic process model.

The event nodes are used in order to react to external events. There are two types of external events in this example:

- Cancellation of the order by the customer which leads to an unconditional termination of the process.
- In case of unavailability of the ordered items in the inventory waiting for a message informing about the arrival of new items, after which the inventory is checked again, till the order can be satisfied. This is also an example of how loops are introduced into a process.



Electronic commerce is usually classified according to who are the participants interacting with each other, i.e. businesses, consumers, and administrations (or governments). If we focus on the commercial interactions regarding the trading of physical goods (including services), we come to the probably most important subclass of electronic commerce, namely B2B ecommerce or electronic business. B2B ecommerce automates interactions among businesses, that traditionally have taken place using other means of communication (in particular paper-based communication). Important subclasses of electronic business are:

Supply chain management: enterprises produce partial products, that other enterprises require as part of their products (e.g. extremely simplified for car production: mining of iron – production of steel – production of parts from the steel – assembly of car parts, which are all done by different enterprises). Supply chain management requires often very tight integration of the business processes among enterprises in order to ensure that the supply chain works properly (e.g. the stocks are readily available for production).

Service integration: producing a product and selling it to a customer requires a number of additional services, such as finance, insurance, accounting, or logistics, which are usually not provided by the producing enterprise, but need to be tightly integrated with the production and sales processes.

Electronic procurement: enterprises require for their operation also goods, that do not constitute a part of the product, such as pencils, computers etc., and which are easily exchangeable. For this, they are trying to find vendors that provide them at the best conditions, similarly as a private consumer. This form of B2B ecommerce bears thus some similarity with B2C commerce, e.g. use of catalogues, auctions etc. It differs however in the scale of trading value.

Interoperability Issues in B2B Commerce

- Communication: Exchange of business data over the network
  - Transport of messages: synchronous and asynchronous communication
  - Message contents: B2B protocol standards
  - Message recipients: trading partners, security

➢ Semantic B2B integration
  - Organized and automated exchange of formalized business data between trading partners access across networks preserving business data semantics

- Coordination: No central process management (B2B can be also P2P)
  - Communication is performed bilaterally
    - No participant has a global view of the process
  - Coordination of process with communication
    - When, what and with whom to communicate
  - Integration of back end applications into process

➢ (Semantic) B2B integration server
  - Software component that manages the state and execution of semantic B2B integration by connecting to back end applications as well as to trading partners over networks
  - Has to address the complete problem space of semantic B2B integration

When studying the problem of interoperability in B2B commerce we can take two viewpoints:

- The communication viewpoint, looking at the problem of how business partners can exchange their data through messages. This requires essentially an agreement among the business partners on the way they communicate including all aspects (transport, contents, participants), but specializing from general communication protocols to specific business protocols (often to specific protocols for an industry). Using such semantically specialized communication protocols is the basis for a semantic integration of businesses. The agreements necessary for enabling semantic integration are normally captured in (industry-specific) standards.
- The processing viewpoint, looking at the problem how to coordinate the various activities in B2B commerce (communication, local workflow processes, existing back-end systems). This requires special software, that is capable to communicate through semantic B2B commerce protocols with trading partners and to integrate back-end applications. Since these servers typically support specific business protocols, they are called semantic B2B integration servers.

History of B2B Systems

- Roots
  - Standards
    - SWIFT
    - EDIFACT
    - X12
    - Etc.
  - Networks
    - Value added networks, e.g. FIN for SWIFT
- Old, but working fine
  - For at least 25 years
  - Adaptation through XML
    - From SWIFT to swiftml

B2B commerce has a long history as explained earlier. Some of the standards from earlier times are

-SWIFT: well-known, used for interbank clearing, supported by an own network (FIN)

-EDIFACT: a European standard covering almost every aspect of B2B interactions

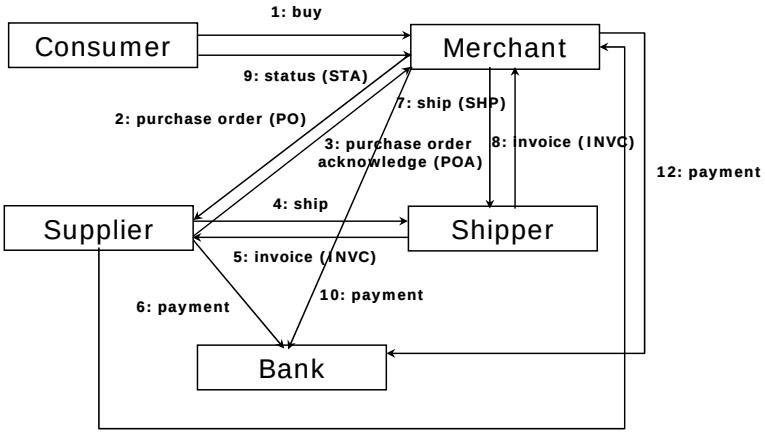
-X12: the US pendant to EDIFACT

Today also many of these standard are migrated to XML message formats. In fact, their importance lies not so much in the encoding of business messages, which is often rather intricate and was for a long time driven by efficiency considerations, but in the extremely rich semantic knowledge on the specific application domains.

Based on these EDI standards there existed (and exist) solutions that support the integration of application from different enterprises through B2B protocols, which were application specific (e.g. SAP has an own business message protocol). This leads however to the usual n^2 integration problem. Another solution were EDI message converters, which could convert from various in-house formats to standard EDI formats. These don't have the n^2 problem, but given the complexity of EDI standards and the very low demand for the converters, these products generally are extremely expensive and only very large enterprises, e.g. major banks, could afford their installation.

A Typical B2B Scenario

- Scenario involves different **roles**
  - Consumer, merchant, supplies, shipper, bank
- Communication through standard **message protocols**
  - Merchant-supplier: PO - POA
  - Merchant-shipper: SHP - INVC

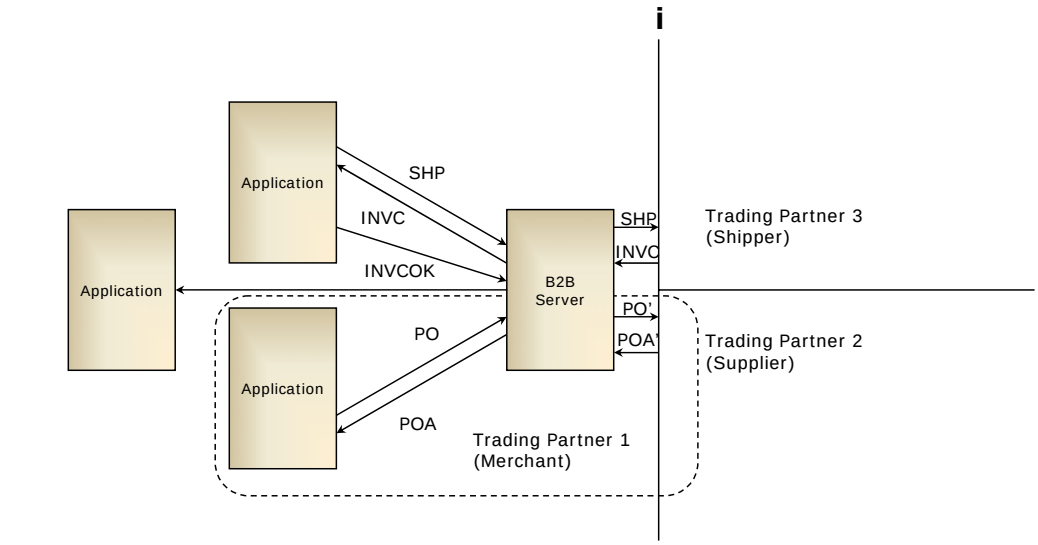


Handling a purchase involves the following activities, which are performed through message exchanges

- A buy message from the consumer to the merchant (this could be also be performed through a user interface, like a Web browser).
- A purchase order (PO) message of the merchant to the supplier
- The POA acknowledging the good delivery (or refusing it). We assume for the following that the delivery will be made.
- The supplier sending a shipping order to the shipper in order to ship the good to the merchant.
- The shipper sending its invoice to the supplier.
- The supplier paying the invoice of the shipper at the bank.
- The merchant sending a shipping order to the shipper for shipping the good to the consumer.
- The shipper sending its invoice to the merchant.
- In the meanwhile the customer may request on the status of processing his buy request.
- The merchant paying the shipper.
- The supplier sending an invoice for the good to the merchant (probably including the shipping cost)
- The merchant paying the good at the bank

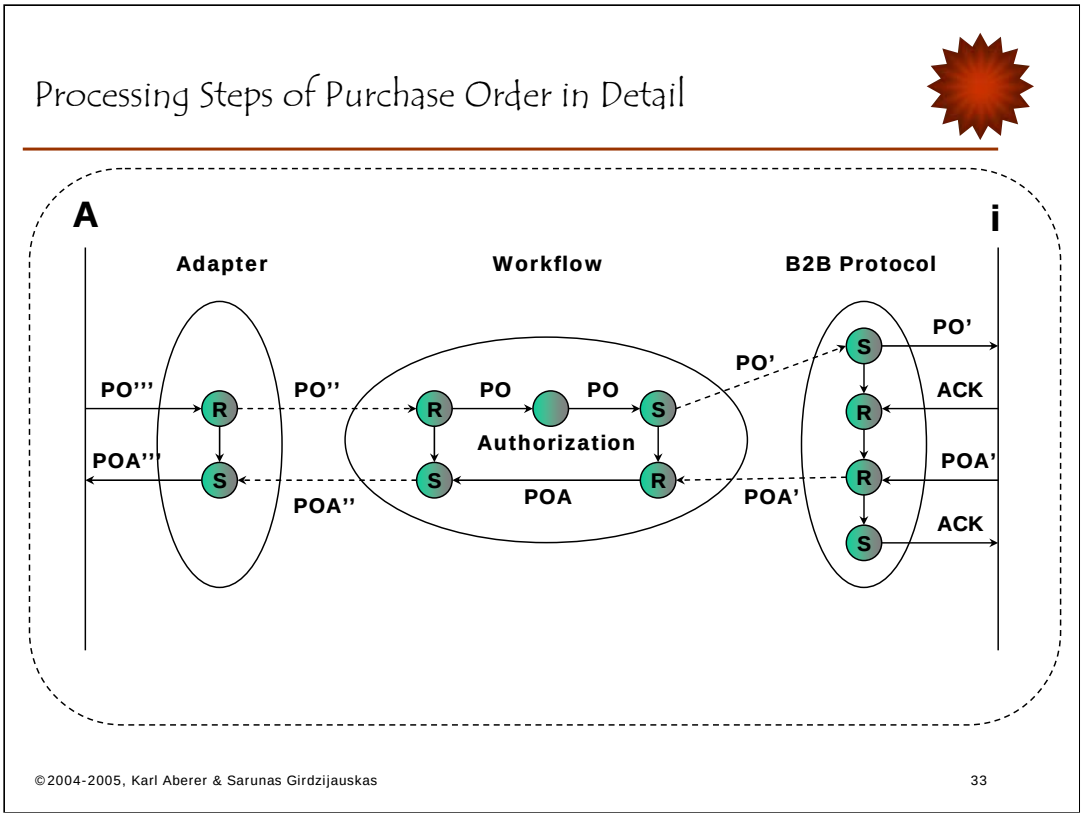
The remaining interaction between the customer and the merchant (paying the good) is not handled electronically and thus does not show up in this scenario.

System View of Merchant



If we zoom into the system, that the merchant is running in order to handle these interactions, we see that it has two types of components involved: the B2B server, that handles the business protocols with the other participants and the (inhouse) business application, like inventory management or accounting. Thus the B2B server has to handle protocols on two sides: the communication with trading partners and the communication with business applications, that are typically existing (legacy) systems and that handle the business logic. The necessary protocols usually will be different !

In addition the interactions between the in- and outgoing messages and the different applications need to be coordinated and are thus part of a business process.



If we again zoom into the detailed steps that are performed just in the interaction of the merchant with the supplier when exchanging the PO and POA messages, we see the following:

From the viewpoint of the B2B integration server the back-end application (an order processing application) behaves as follows:

- The B2B system receives from the back-end system the purchase order (PO''')
- The B2B system has in the next step to send to the backend system a purchase order acknowledgement (PO''')

This workflow is captured in the Adapter workflow. The Adapter is a wrapper for business applications, that translates message syntax and handles communication.

Similarly the B2B integration server perceives the B2B protocol as a process consisting of 4 steps:

- Sending the PO' to the trading partner
- Receiving an acknowledge of message receipt (in order to ensure that the message was correctly transmitted)
- Receiving a POA' message, acknowledging that the purchase can be conducted
- Sending a message receipt to the trading partner

Note how acknowledgements at two levels of semantics occur in the B2B protocol: one for ensuring the correct communication and one for implementing the business logic of purchase.

Example B2B Protocol Standard: RosettaNet

- RosettaNet is an independent non-profit consortium of technology companies
  - Defines open and common electronic business processes for information technology over electronic distribution channels implementation framework
- Interface processes (PIPs)
  - Define business protocol and business document format
  - Business Dictionary designates the properties for defining business transactions
  - Technical Dictionaries provide properties for defining products and services
  - Core processes
    - Administration, Partner, Product and Service Review, Product Introduction, Order Management, Inventory Management, Marketing Information Management, Service and Support, Manufacturing
- Implementation framework (RNIF 2.0)
  - Defines sequencing, packaging, transport binding, security

RosettaNet is standard for B2B commerce in the information technology area, that covers all aspects of B2B data exchange from the semantic specification of messages down to the implementation architecture for message exchange. In that sense it provides an alternative solution for message-based interactions to the Web service model we have seen before, together with the semantic definitions for a specific application domain.

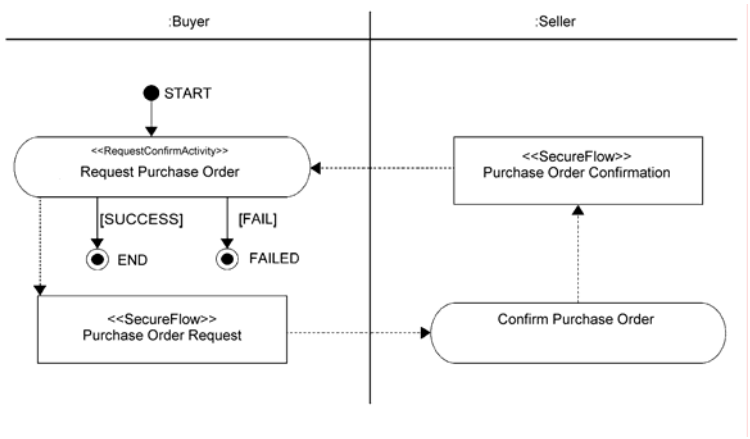
The semantic part of RosettaNet are the PIPs (Partner Interface Processes). PIPs include the specification of partner business roles (Buyer, Seller etc.), business activities involved between the roles and type, content, and sequence of business documents exchanged by the role-interactions while performing these activities. They also specify the time, security, authentication, and performance constraints of these interactions. Structure and content of the business documents exchanged is specified through XML Document Type Definitions (DTDs) and associated Message Guidelines. The specification of dictionaries can be used in order to constrain data values used to describe products and their properties.

The implementation framework is comparable to what is provided by SOAP, however it is more sophisticated, in particular with respect to security support.

We illustrate in the following the PIPs for the Purchase Order process.

Example: Request Purchase Order (1)

- PIP business protocol



This is the specification of the purchase order process as UML activity diagram. This view of the PIP is called Business Operational View (BOV).

The diagram contains "activities" (the rounded boxes) and "object flows" (the rectangles), which are use to control the activity state transitions by means of a data flow. Concretely, in state RequestPurchaseOrder the action of sending a message PurchaseOrderRequest will pass control to the ConfirmPurchaseOrder activity, which passes it back through the flow of the PurchaseOrderConfirmation message to the activity RequestPurchaseOrder, which then decides upon the incoming message whether to move to the success of failure state. The <<Secure Flow>> stereotype in the boxes containing the business actions implies that the business action MUST be transported from sender to recipient in a secure way.

Example: Request Purchase Order (2)

```
<!ELEMENT Pip3A4PurchaseOrderRequest (
          fromRole ,
          GlobalDocumentFunctionCode ,
          PurchaseOrder ,
          thisDocumentGenerationDateTime ,
          thisDocumentIdentifier ,
          toRole ) >

<!ELEMENT fromRole   ( PartnerRoleDescription ) >

<!ELEMENT PartnerRoleDescription (
          ContactInformation? ,
          GlobalPartnerRoleClassificationCode ,
          PartnerDescription ) >

<!ELEMENT ContactInformation (
          contactName? ,
          EmailAddress? ,
          facsimileNumber? ,
          telephoneNumber? ,
          PhysicalAddress? ) >
...
```


This is a (short) excerpt from the DTD for a (business) action message, illustrating of how the "semantics" of a purchaseorder is captured in detail in a DTD.

Summary

- Workflow management systems support
  - Explicit process specification and enactment
  - Thus useful for business process (re-)engineering and optimization and programming in the large
  - Process model define the control and data flow between the activities
- Workflow management systems
  - Are a process-oriented middleware
  - Can be implemented in a message-oriented or repository-oriented fashion
  - Are standardized by the Workflow Management Coalitions models
- B2B servers are the latest generation of business information systems enabling interoperability of businesses
  - Address the issue of distribution: Message-based communication between businesses
  - Address the issue of heterogeneity both at the data and process level: message transformation and application integration
  - Address the issues of autonomy: Trading partner management and contracts, security

References

- Books
  - F. Leymann, D. Roller: "Production Workflows", Prentice Hall, 2000.
  - P. Timmers, "Electronic commerce - strategies and models for business-to-business trading", Wiley 2000.

- Websites
  - Workflow Management Coalition: www.wfmc.org
    (http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf)
  - Weblogic Process Integrator:
    http://e-docs.bea.com/wlintegration/v2_0/index.html