Prof. Dr.-Ing. Stefan Deßloch
AG Heterogene Informationssysteme
Geb. 36, Raum 329
Tel. 0631/205 3275
dessloch@informatik.uni-kl.de

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

Chapter 12 – Business Process Modeling and Workflow Management

Introduction & Motivation
Business (Re-)Engineering
Workflow Management Systems
WF and Transactions

Prof. Dr.-Ing. Stefan Deßloch
AG Heterogene Informationssysteme
Geb. 36, Raum 329
Tel. 0631/205 3275
dessloch@informatik.uni-kl.de

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

Introduction & Motivation

Terminology

- Material process
  - assembles physical components, delivers physical products
  - may include moving, storing, transforming, measuring, assembling objects
- **Information process**
  - relates to automated and partially automated tasks that create, process, manage, provide information
    - involves programs, humans (interacting with computers)
  - infrastructure provided by database, transaction processing, and distributed systems technology
- **Business process**
  - market-centered description of an organization's activities
  - implemented as an information process and/or material process

Business Process Examples

- Manufacturing
  - Assembly lines of cars, PCs, cloths,...
- Insurance
  - Handling of claims, policies,...
- Finance
  - Stock brokering, settlement, clearing,...
- Banking
  - Loans, savings, current accounts,...
- Database administration
  - Backup & recovery, reorganization, tuning,...
- Software development
  - Waterfall model, spiral model,...
- Telecommunications, administration, government, data warehousing...

There is nothing like a "typical business process"!!!

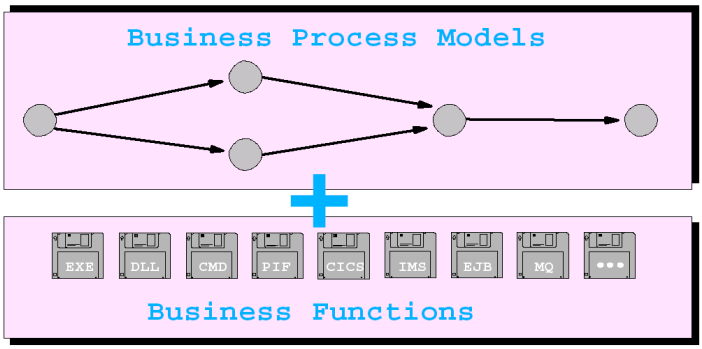
Why Care About Workflow Technology?

- Companies use computers to support their business,
  - most frequently
- The way to do business is prescribed via a business process,
  - very often
- Applications support business processes and have to ensure compliance with business processes
  => Application = Business Process + Business Functions
- Changes in how to perform business must be reflected as soon as possible in applications
- A workflow is a business process in execution (an instance of a process model) in a computing environment
  - Not all parts of a process are run in a computing environment - some processes are not run on a computer at all!
  - Often, "workflow" and "process" is identified

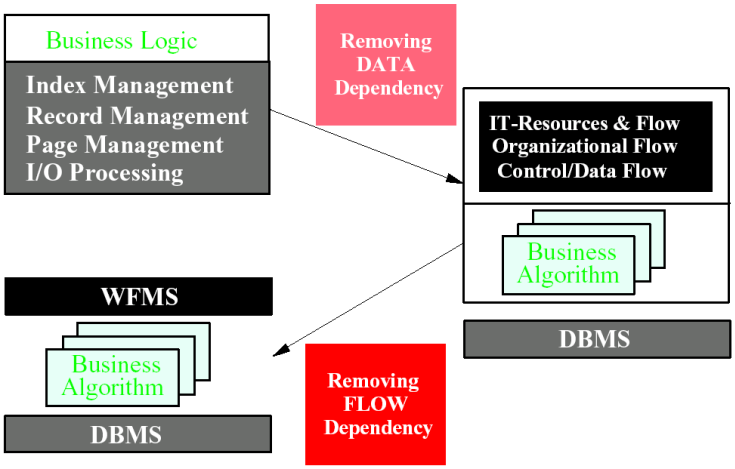
Workflow-Based Applications: Evolution

- Typically, large applications use special "control programs" to ensure the appropriate/correct sequencing of business functions
- Control programs often represent business processes
  - Requires code changes [which part to change?...], recompilation, redistribution of code,... to reflect new business processes
    - What if users of standard applications want to reflect their own processes?
      Very difficult, cumbersome, expensive (service specialists, consultancy),... thus an obstruction to buy standard software
- Consequence: Implementation of control programs via workflows
  - Application consists of collection of business processes and collection of business functions (= "usual" programs)
  - Business processes are enacted by workflow system that invoke business functions "appropriately", i.e. according to process model
- No coding,... to adapt application to changed business process

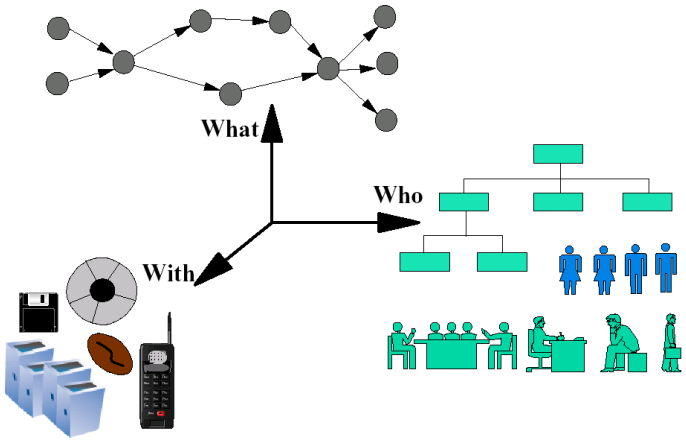
Workflow-Based Applications: Structure



Flow Dependency Removal



The Three Dimensions Of Workflow



People Workflow Evolution: 1st Generation

- Electronic document and folder routing (late 80s)
  - Document = image, folder,...
  - Routing through enterprise's organizational structure
  - User associated electronic basket is key
    - Container for documents a certain user has to work on to contribute to a case
  - Potential flow of documents prescribed in advance
    - Routing conditions in terms of document content or document properties
    - Actual routing based on actual content or properties of subject document
- In "paper factories" (administration, insurance, banking,...) work mainly equates to processing documents, thus the term **workflow** has been used for routing documents between people

People Workflow Evolution: 2nd Generation

- Functions performed by users in 1st generation WFMS are mainly retrieval, browsing, editing, archiving,...
- But cases represented by documents were recognized to be only part of larger business processes
  - Not only performance of document management functions required but also usage of other functions provided by application systems supporting the operation of an enterprise
- WFMS extensions needed to invoke any kind of executable
- In-/Out-Basket grew towards worklists
  - Launch-pad for executables
  - Workitem management
    - Prioritization, duration management, life-cycle,...

People Workflow Evolution: 2nd Generation (cont.)

- Launching executables requires parameter passing
- Thus, data flow features complemented available control flows
- In turn, control flows can now be expressed in terms of these new parameters ("business rules")
- Data flow is used for integrating applications with long temporal delays between their initiations
  - Parameters managed by data flow must be persistent
  - Data flow must be allowed to be different from control flow
    - Data produced by application A might be used by application B to be started after a couple of intermediate applications run

People Workflow Evolution: 2nd Generation (cont.)

- Being able to support large spectrum of business processes in computing environments made WFMS of strong interest for Business Process Reengineering (BPR) projects - early 90s
- Goal of BPR is to speedup business processes and reduce their costs. Resulting requirements:
  - Parallelism in workflows (-> speedup)
  - Deadline processing (-> speedup)
  - Monitor actual workflow status (-> speedup)
  - Auditing of significant events, i.e. processing history (-> cost reduction)
  - Maintain execution history for analysis (-> cost reduction)
  - Process activities without human intervention (-> speedup + cost reduction)
    - So-called automatic activities
    - Consequence: (parts of) business processes can be automated ("macro-scripts")

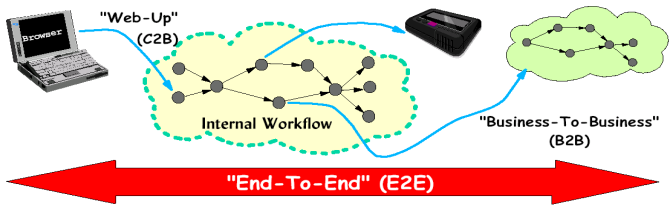
People Workflow Evolution: 3rd Generation

- Workflow-based applications become state-of-the-art (mid 90s)
  - Strict separation of business process logic and business functions
    - Business processes implemented via workflow system
    - Business functions implemented "traditionally" (TP-monitor, ORB,...)
- Enterprises become dependent on WFMS
  - Similar to TP-Monitors and DBMS before
  - The term **production workflow** has been coined to indicate that WFMS is driving operational aspects of an enterprise
- Consequences:
  - WFMS had to provide quality of services known before from "production systems" like DBMS and TPM
    - High/continuous availability
    - Scalability
    - Robustness

People Workflow Evolution: Latest Moves

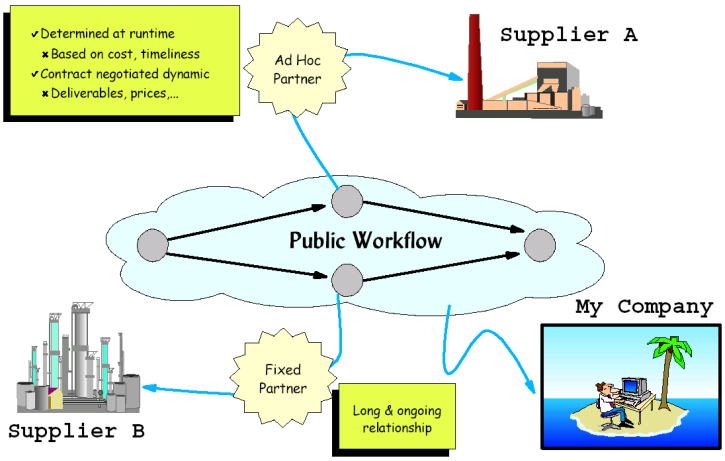
- Application integration becomes important
  - Integrate diversity of application functions
    - legacy applications, newly written applications (e.g. component based),...
    - new invocation paradigms (e.g. message queuing, publish/subscribe)
    - workflows as granules to be integrated
- Organizational integration becomes more and more important
  - Workflows expand across business units of enterprise ("intra-enterprise")
  - Workflows across enterprises become necessary ("inter-enterprise")
    - Creation and enactment of workflows in virtual enterprises
    - Stimulated by mergers and acquisitions, outsourcing, supply chains,...
  - Interoperability of WFMS (building blocks) and web access required
- Workflows understood as business oriented "logical units of work"
  - Advanced transaction management functions required
  - Forward recovery of workflows as well as workflow-based applications
  - Backward recovery (spheres of atomicity and compensation)

Workflows And External Communications

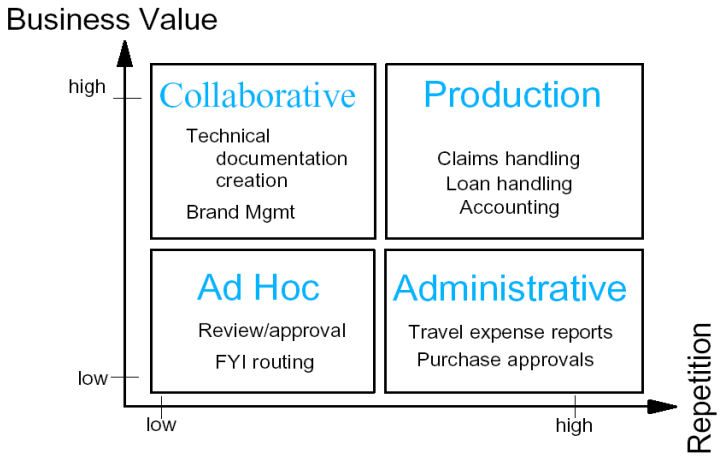


- Customers invoke company's applications to perform certain steps of the business process
  - E.g. place on order, inquire status,...
  - Company's applications must get a browser-based front-end for that purpose ("web-up")
- Workflow activities may directly communicate with the outside
  - Send e-mail, faxes, messages,...
- Workflow activities may trigger actions in another company
  - Simple invocation of program or start of another workflow ("sub-process" from invokers point-of-view)
  - Such "business-to-business" scenarios are the base for realizing sophisticated "supply chains"

Finding Trading Partners



Workflow Classification



Characteristics Of Production Workflow

- Coordination
  - Process models as enterprise resource
  - Model driven execution of applications
  - Application integration
- Operation
  - Transaction support
  - Reliability
  - Availability
  - High capacity
  - High performance
  - Scalability
- Enterprise
  - Multi platform
  - System management
  - Standard compliance
  - Security
  - Process tracing

Transactional Workflow Evolution

- Success of TP Monitors and concept of (classical) transactions have been overwhelming
- Hidden assumption behind classical transactions:
  - Short duration (fractions of a second to a few seconds)
- Technical underpinnings based on this assumption
  - 2-phase-locking, log based recovery,...
- Early 80s started to extend transaction technology towards longer durations
  - Technical underpinnings have to be adapted
- Most famous "transaction models"
  - Nested transactions (closed & open)
  - Sagas
  - Multilevel transactions

Transactional Workflow Evolution: Structures

- Structures of transactions have been extended from sequences and trees to directed acyclic graphs
  - Dependencies between transactions are described (e.g. "flexible transactions")
- Backward recovery based on ACID semantics as well as compensation has been folded in
  - E.g. "ConTracts"
- Late 80s, early 90s:
  The term "transactional workflow" has been coined for prescribing control flow dependencies between transactions and their joint backward recovery

Transactional Features of Production WF (cont.)

- Production workflows invoke a lot of non-transactional programs too (i.e. programs that cannot be simply undone)
- Thus, supporting compensation based recovery in production workflow systems is only natural
- Especially, a "unit of work" must allow to include
  - transactional as well as non-transactional programs
  - long running programs
  - programs that demand human interactions
- Ability to involve people in recovery:
  - In exceptional situations people can be notified as part of recovery processing
  - Human beings might "repair" the exceptional situation allowing to continue processing

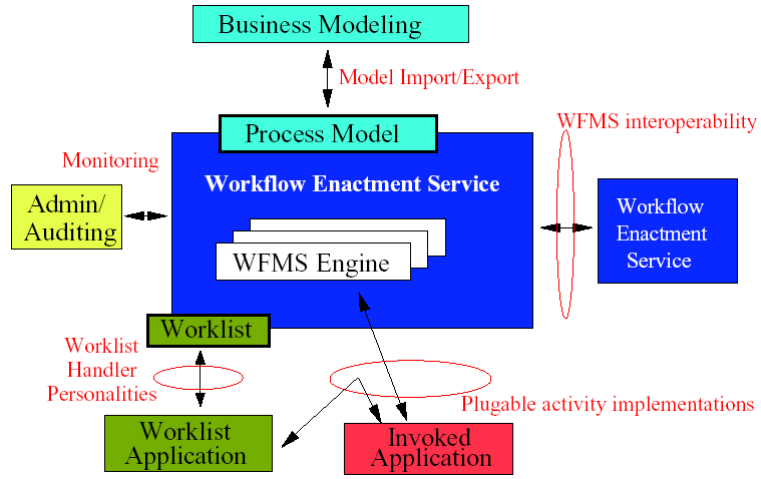
Transactional Features of Production WF (cont.)

- Today's workflow systems maintain complex states for whole workflow as well as for each single step in underlying database
  - Result: Each workflow itself is forward recoverable
- Few workflow systems can include user provided transactions in their own internal transaction processing
  - Result: Complete workflow-based application is forward recoverable
    - More precise: All parts involving transactional steps are forward recoverable
- Today's workflow systems manage long running units of work
  - Spectrum reaches from seconds to hours, days,..., even years!
  - Consequence: Unit of work must be interruptable at "any" point in time
    - Not only between execution steps but execution steps themselves (the latter involves exit conditions and persistent context for activities)

WF-Based Apps: Industry Acceptance

- Large companies adopted this paradigm in the early 90s
  - Built their own workflow systems at that time
    - No real production workflow system was available
  - Benefits: Time to market for new/modified products
- Standard application vendors adopted this paradigm mid 90s
  - Most vendors built their own workflow system because no system dominated the market
  - Benefits: Customization and internationalization
- Standardization started mid 90s
  - Workflow Management Coalition (WfMC) since 95
    - The standard consortium for workflow standards since 99
  - OMG's Workflow Management Facility = Objectification of WfMC
- Vendors roll out production workflow systems 2nd half of 90s
  - IBM MQSeries Workflow, Oracle Workflow, HP ChangEngine, SAP Business Workflow...

The WfMC Reference Model



Prof. Dr.-Ing. Stefan Deßloch
AG Heterogene Informationssysteme
Geb. 36, Raum 329
Tel. 0631/205 3275
dessloch@informatik.uni-kl.de

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

Workflow Management – Business (Re-)Engineering

The Notion of Business (Re)Engineering

Business Reengineering =

The fundamental rethinking and radical redesign of business processes to achieve dramatic improvements in critical, contemporary measures of performance, such as cost, quality, service, and speed.

M.Hammer and J.Champy, Reengineering the corporation, HarperCollins Pub.Inc., 1993.

So, What Is BPR All About?

- Business Process (Re-)Engineering (BPR) one of the most important topics on many company's agenda
- Recall, that very often: Process Model = Product
- Goal is to make company more flexible, react faster to change
  - outsourcing of processes, supply-chains, virtual enterprises,...
- Criteria for success include
  - minimize process execution time/cost, maximize executed number
    - Eliminate unnecessary tasks, perform as many tasks as possible in parallel, parallel tasks use different resources (personnel, equipment, program,...)
- New processes are defined, existing are changed or abandoned
- Scope is not only intra-enterprise but also inter-enterprise
  - Business-to-Business, Consumer-to-Business, Business-to-Administration,...
- Reengineered processes supported by distributed and heterogeneous computing environment

What Has To Be Done

- Existing business processes must be
  - Analyzed
  - Specified
  - Modeled
  - Optimized
    - this includes simulation
- Important to include resources used to perform processes
  - Organizations
  - Roles
  - People
  - IT resources
- Huge number of BPR methods have been proposed!
  - ...and many tools accompany these methods!
  - Examples
    - ARIS Easy Design (IDS Prof. Scheer)
    - Workflow BPR (Holosofx)

Deliverables of Business Modeling

- Process goals
  - Strategic targets like
    - growth of company over period of time
      - Number of customers, products sold, employees,...
    - profit level
    - customer satisfaction
  - Agreement on these goals is vital for success of any BPR project!
- Business processes ("Ablauf-Organisation")
  - High-level view only
    - major activities, organizational units involved, goods/materials/... required, computer (sub)systems used, data processed,...
  - Activities will be refined later on
    - typically, at this level activities are often processes itself
      - will be refined into subprocesses later on (top down / bottom up)
  - Data often is just name of database to be used
    - customer database, product definition database,...

Deliverables of Business Modeling (cont.)

- Number of process instances
  - Reflects one of the strategic targets
  - Used for simulation later on
    - determines number of people needed, cost of the business process,....
- Organizational structure ("Aufbau-Organisation")
  - Very important aspect of business modeling
  - Includes specification of
    - broad areas of responsibilities, span of control, reporting structures
  - Typically, organizations are hierarchically structured, crisp responsibility
    - result: crossing organizational boundaries become "barriers"
    - negotiations about responsibilities, funding, revenue sharing,...
    - delays in performing activities of business processes
  - Hierarchical structures are obstructions in business process efficiency
  - Imperative to change organization

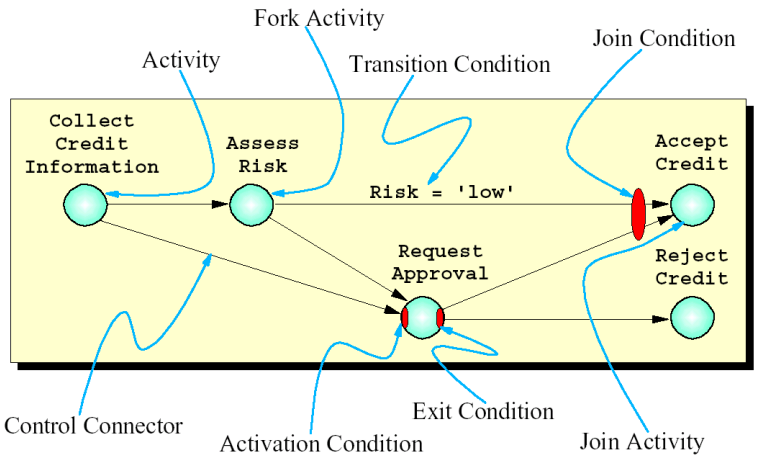
Deliverables of Business Modeling (cont.)

- Business objects
  - Activities of business processes work with/on business objects
    - not necessarily an "object" in the OO-sense
    - activities work with it, get as input, produce as output
      - customer address, credit history, actual stock price, risk assessment,...
    - may determine actual control flow between activities
      - amount of a loan, severity of an accident, risk assessment,...
    - could be used to determine access rights people need to perform a task
      - nobody must see salary of managers in own reporting chain,...
  - Needed when implementations of the activities are built
    - database structure required to support the activity when executed
    - core entities for conceptual database design
- Number of business objects
  - Reflects one of the strategic targets
  - Used to derive required storage space,...
    - when combined with access frequencies used for physical database design

Deliverables of Business Modeling (cont.)

- Critical success factors (CSFs)
  - Prerequisites to successfully execute a business process
  - Crucial for achieving all the goals set during the other modeling actions
  - CSFs include
    - skills of people
      - hands-on experiences with tools
      - knowledge in application areas
    - properties of IT infrastructure
      - power of workstations used by personel
      - power of servers used to run automatic activities

Business Process: Control Flow



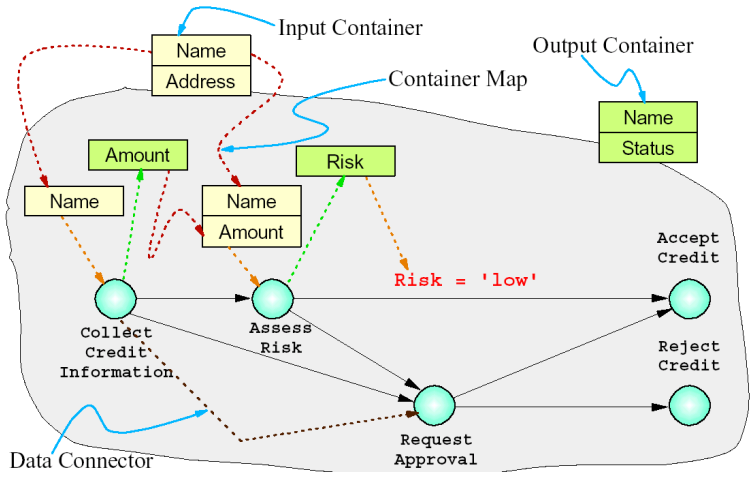
Control Flow – Details

- Activities
  - describe task to be performed
  - different types of activity
    - program activity, person activity, process activity (subprocess), block (do-until loop)
- Control Connectors
  - describe potential sequence in which activities are carried out
  - connect **source** and **target** activities
  - **start/end** activities have no incoming/outgoing connectors
  - parallelism supported through **fork/join** activities
    - multiple outgoing/incoming connectors
    - join conditions act as synchronization points for parallel activities
- Conditions
  - **join** condition defines whether an activity can be started
  - **activation** condition specifies when an activity can be started
    - evaluated after the join condition
  - **exit** condition confirms that the activity has been successfully completed
  - **transition** condition describes a condition for following a control connector

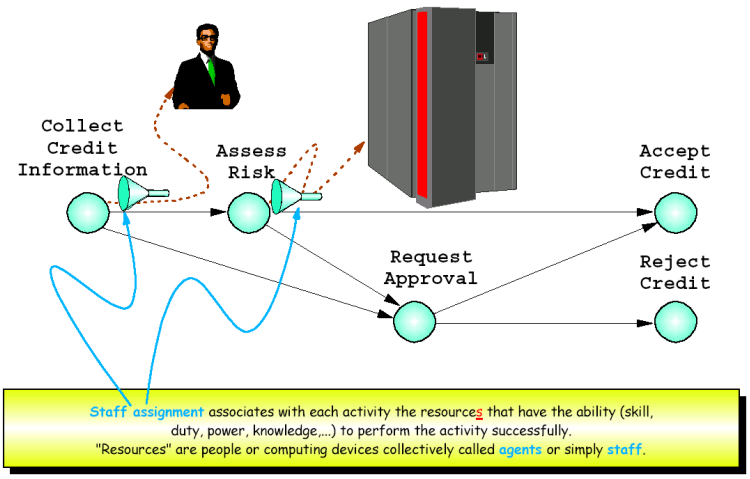
Control Flow – Navigation (→ WFMS)

- Navigation – processing the process graph
  - begin at the start activities, after completion follow the outgoing connectors
  - control connectors are associated with truth values
    - initially unknown
    - evaluate to true/false after source activity completes, based on transition condition
  - control follows a connector to the target activity only if it is labeled "true"
  - navigation stops at a join activity until all incoming connectors are labeled either "true" or "false"
    - join condition determines whether the join activity is executed
      - can refer to truth values of incoming connectors
      - simple conditions: all true, at least one true
- Dead Path Elimination
  - if all incoming control connectors of an activity have been evaluated, but the activity cannot be carried out because the start/join condition evaluates to "false", then the outgoing connectors of that activity evaluate to "false"
  - repeated until navigation halts or reaches an end activity
  - a process terminates if all end activities have been reached (carried out or skipped)

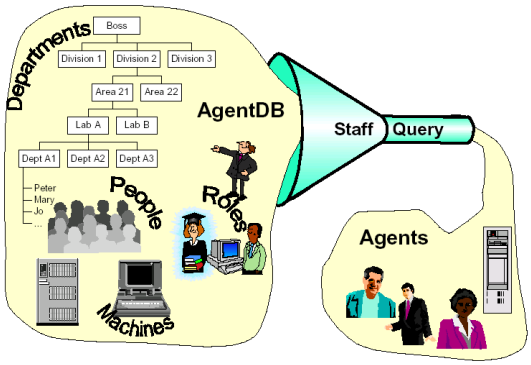
Business Process: Data Flow



Business Process: Staff Assignment

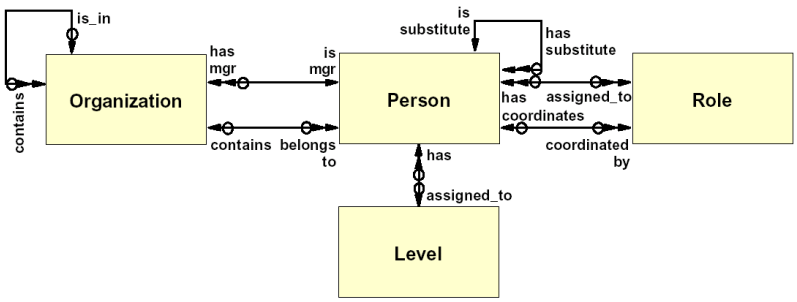


Staff Resolution: Flexibility in Staff Assignments



Staff assignment is done by attaching a declarative query to each activity, i.e. no fixed agents are associated with an activity leaving flexibility to exchange agents without having to modify all affected process models and instances!

Org Database: The "Base" For Staff Assignment



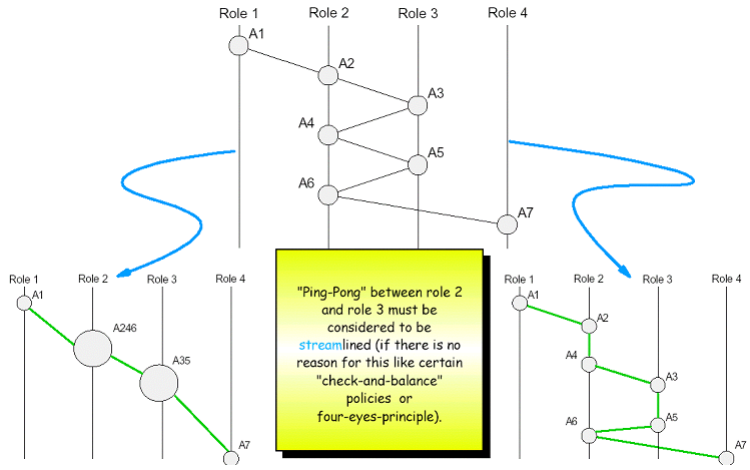
- WFMS has integrated org-db or can access external one (e.g. custom or standard HR-systems [SAP R/3, Peoplesoft,...])
- Org-DB can have simple structure (like the one shown) or very complex/rich
- Populating the org-db schema is called organization modeling (often, unload HR-DB and load into WFMS Org-DB)
- Modeling the org-db schema is called organizational meta modeling

Simple Steps Towards Process Optimization

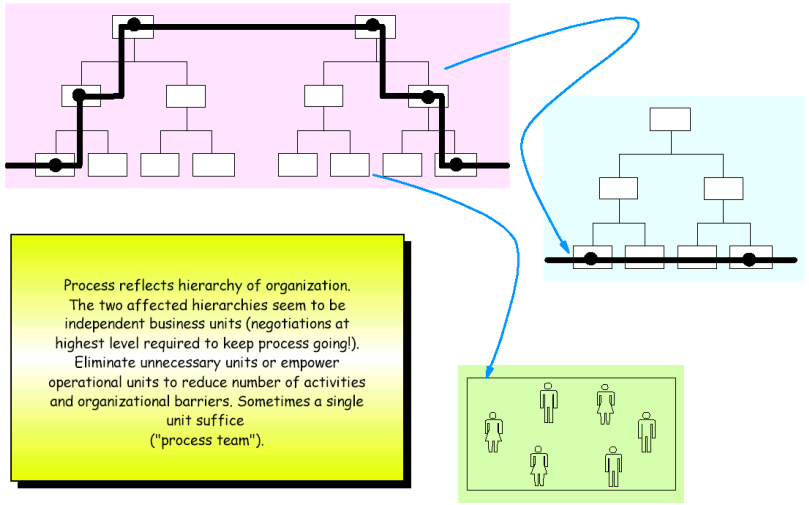
- Characteristics of optimized process:
  - Minimal number of crossing organization boundaries
  - High level of parallelism

    Often, simple **static analysis** of flows through organization result in big improvements!
- This allows processes to be performed fast
  - ... but does not guarantee it
  - Duration/deadline management of WFMS helps further
    - Specify maximum time
      - an activity must be worked on (with/without interrupts or idle time)
      - an activity must be started once scheduled by the WFMS
      - an escalation may take (notification of manager,... if time threshold is exceeded)

Process Optimization: Streaming Activities



Process Optimization: Restructuring Organizations



Process Analysis

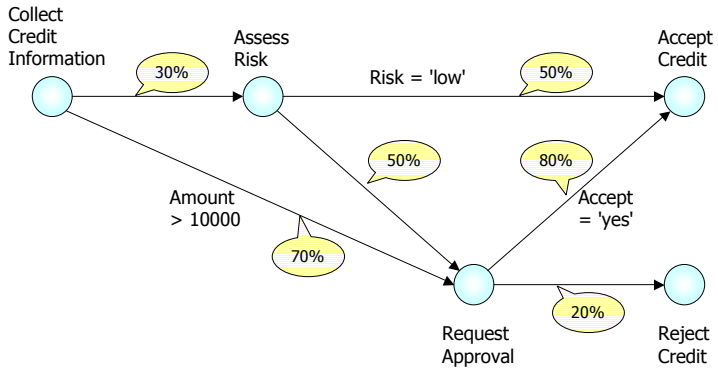
- Dynamic analysis...
  - takes into account quantitative aspects
    - number of processes per time unit, probabilities that certain paths are taken,...
  - produces quantitative aspects
    - resources consumed to perform certain activities, to carry out business process,...
- Simulation generates information about...
  - human resources needed to execute business process
    - impact on hiring strategy
  - skills needed to handle business process
    - impact on skill planning
  - time and cost for performing business process
    - indicator for outsourcing
- Used to compare and select from alternative models of a given business process the "optimal" one
    - optimal in terms of metrics like cost, duration,...

Purpose of Simulation

Verify capability of organization to support expected workload

- Performed based on metrical information ("instrumentation")
- Instrumentation requires to specify
  - Number of processes started per time interval, i.e. distribution patterns of starts - for example:
    - constant: same number for each time interval
    - exponential: smaller numbers more frequent than larger numbers
    - uniform: numbers random within lower and upper bound
    - customer defined: 57 between 9AM and 11AM, 341 between 11AM and noon,...
  - Probability of transition conditions (likelihood of different branches taken)
  - Probability of activation-, join- and exit conditions (likelihood of repetitions)
  - Average duration of activities (work time, idle time,...), i.e. their distribution patterns
  - Processing power of resources, availability (based on calendar, shifts,...)

Sample Instrumentation Of Control Flow

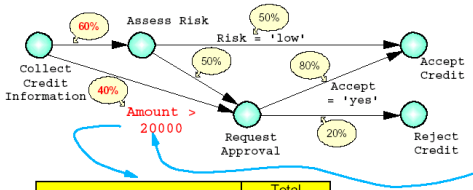


Analytical Simulation

- Calculates
  - ...how often each activity has to be performed
    - based on instrumentation of control flow and probability theory
    - no automatic association of activities with individual resources
    - simply association with corresponding "staff assignment" statement
  - ...different paths taken through process model and their probability
  - ...corresponding durations for performing process and their probability
- Advantages
    - limited instrumentation needed
    - no huge compute power required
    - if result shows that workload cannot be handled, deadline cannot be met,... no further sophisticated discrete simulation needed
- Disadvantage:
  - does not consider
    - resources and their availability
    - resource competition by concurrent processes

Sample Cost Optimization

Activity	Total Cost [$]
Collect Credit Information	6000
Assess Risk	900
Request Approval	15300
Accept Credit	2490
Reject Credit	510
Total	25200



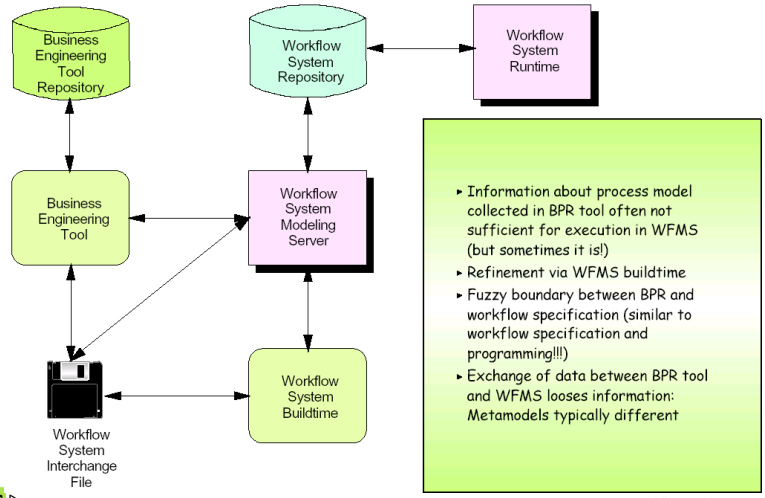
Activity	Total Cost [$]
Collect Credit Information	6000
Assess Risk	1800
Request Approval	12600
Accept Credit	2580
Reject Credit	420
Total	23400

- More than half of the costs due to approval by financial officer
- Number of necessary approvals reduced by adapting transition condition (i.e. loan officers get more responsibility!) and adapting affected probabilities
- Result: Costs reduced from $25.20/proc to $23.40/proc !!!

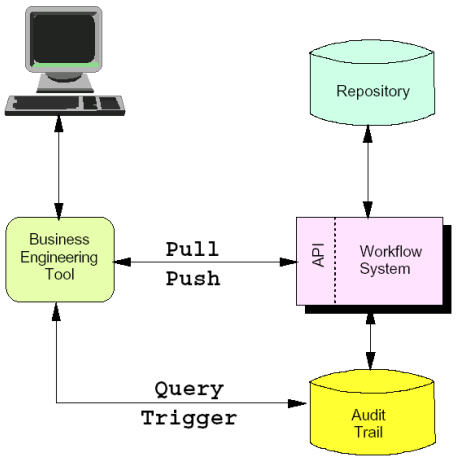
Discrete (Event) Simulation

- Calculates...
  - for each individual resource
    - activities to be performed
    - required time for executing each activity
    - considers availability and processing power of each resource (time schedule, vacation, shifts, experience level,...)
- Considers...
  - impacts of concurrent processes competing for same resources (people,...)
  - probability distributions for start and execution times
- Ideally,
  - navigation engine of target WFMS is used (to avoid mismatch in interpretation semantics)
  - staff resolution is performed based on organizational database

BPR-WFMS Exchange Of Information



Monitoring And Auditing

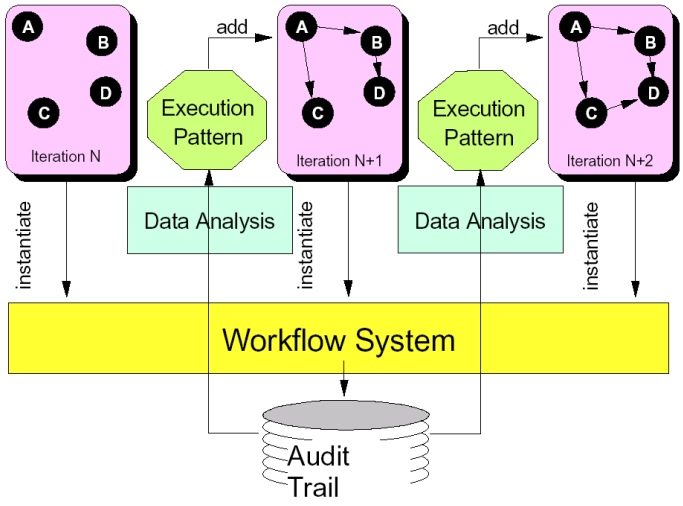


- Business modeling based on assumptions about cardinalities, duration, etc.
- Based on these assumption process characteristics are derived (costs,...) which trigger optimizations
- Thus, incorrect assumptions result in non-optimal process models
- WFMS allows to access actual state (monitoring) as well as history (auditing) of each workflow
- Analyzing audit trail ("vanilla" SQL, OLAP, mining) derives "real data" for optimizing process models (re-engineering)
- Monitoring (manually or automatically) individual processes or instances of the same model allows to detect out-of-line situations and to react accordingly (re-assignment of work,...): Leitstand

Obstructions To Process Modeling

- Business process modeling is often costly because it might be time consuming and error-prone
  - in general, no single person knows/understands a particular business process
    - many people must be interviewed to get as much details as possible
      - usually, process participants only have local knowledge of the process (often they know what they are doing, the tools they use, sometimes who gets involved next or who got involved before)
    - details must be combined/inferred to get full picture
      - sequence of activities must be derived/determined (control flow and data flow)
      - inconsistencies must be detected, analyzed and resolved (similar to "view integration")
- Reducing this cost is highly desirable
- Simple idea: Allow to put partial models into production
  - first iteration uses local knowledge only
    - Time reduction: No need to specify large and complex model
    - Error-reduction: "View integration" reduced/avoided
  - use execution history (WFMS **audit trail**) to find template of the "real" business process

Process Discovery: Main Idea

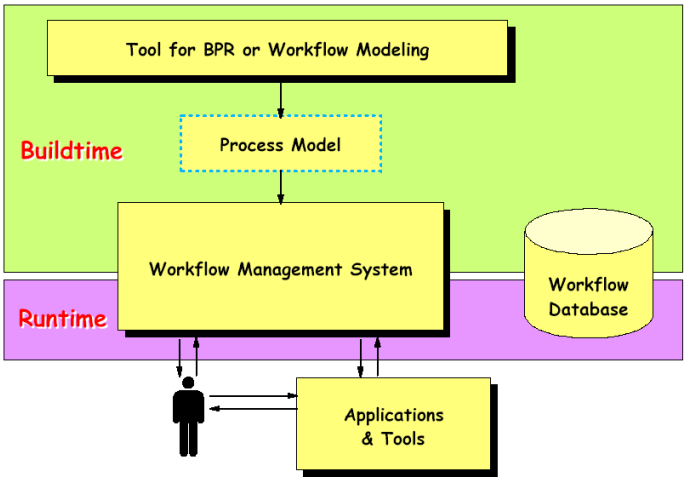


Prof. Dr.-Ing. Stefan Deßloch
AG Heterogene Informationssysteme
Geb. 36, Raum 329
Tel. 0631/205 3275
dessloch@informatik.uni-kl.de

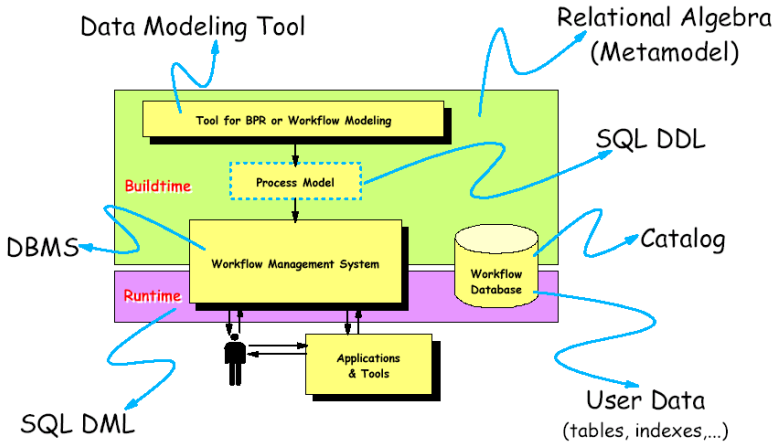
Technische Universität Kaiserslautern

Workflow Management Systems

Major Building Blocks Of A WFMS



...And Their Correspondence In DBMS



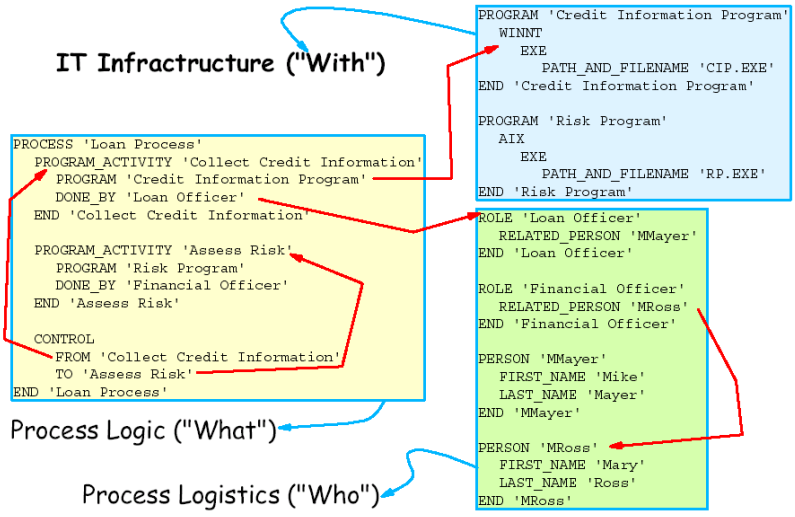
Types Of Users In Workflow Environments

- **End Users**: Perform work assigned to their worklist(s).
  - But also: Transfer work from others to them + being substitute for others
- **Process Modeler** or **business analyst**: Defines process models, organizational structure, IT structure
- **Process Administrators**: Manage running workflows
  - A person becomes ProcAdmin by
    - Explicit specification in process model or category (= grouping of process models)
    - Dynamic assignment via values associated with particular process instance
  - ProcAdmin is informed when something goes wrong with the workflow he is responsible for
    - E.g. staff resolution returns empty set, or activity implementation fails execution,...
- **Operation Administrators**: Keep the WFMS properly running
    - E.g. add resources when more users must be supported,...
- **System Administrators**: Responsible for the overall environment
- **Customer Support**: Mediate between customers & business
    - E.g. inquire state of workflows, or start, terminate,... workflows
- **External Users**: Mainly customers interacting with WFMS
  - Can replace mediation by customer support

Buildtime

- Component providing all functions and capabilities to define, test and manage all workflow related information
  - Especially, all three workflow dimensions are covered
  - Often, administrative and systems management information are included, e.g.
    - Session threshold, i.e. maximum period of time a user can work with the WFMS
    - Actions to be taken when average response time exceeds threshold
  - All information stored in WFMS own database ("buildtime database")
- Two different kinds of interfaces
  - Graphical end user interface
  - Workflow Definition Language
    - ASCII text with special syntax/semantics
      - Most often vendor specific
        - e.g., IBM's FDL (Flow Definition Language)
      - Standard developed by Workflow Management Coalition (WfMC)
        - WPDL (Workflow Process Definition Language)
        - XPDL (XML Process Definition Language)
  - Both GUI and FDL cover all concepts of the WFMS Meta Model

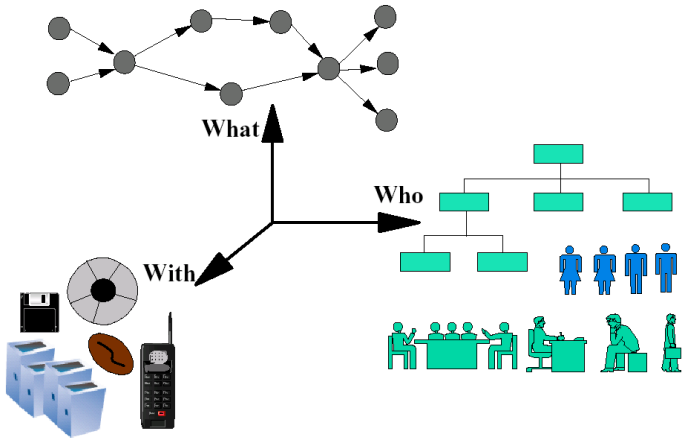
Workflow Definition Language: Example (FDL)



Putting Process Models Into Production

- When modeling a process is finished it can be put into production
- Putting a process model into production means
  - ...to "freeze" the model, i.e. nobody can change it any more
    - Only "what" dimension (the activities and control-/dataflow between them) is really frozen
    - Organization model ("who dimension") can of course be modified
      - E.g. people can change departments
      - Might impact staff queries (e.g. dropping a department a query refers to): If no agent is found process administrator is notified
      - Often, organizational structure is completely maintained via separate application (e.g. Human Resource) and replicated periodically into the WFMS database in batch mode
    - Activity implementations ("with dimension") can be "early bound" or "late bound"
      - Early bound process model is frozen too, late bound process model is resolved at runtime
  - ...often to TRANSLATE the corresponding data into a different format
    - Modeling tool and WFMS runtime might use database structure optimized to their needs
  - ...often to create a new version of an already existing process model ("valid from")
    - Existing instances of earlier versions are run according to the model which was valid when the instance has been created (auditability is a key requirement!)
    - New instances are created according to the new version
- Once put into production, instances can be made from a model

Workflow Meta-Model



"Who" Dimension: Organization Metamodel

- WFMS can support fixed or dynamic organization meta-model
  - **Fixed** org meta-model does not allow to change the entity and relationship types supported to model organizations
    - Org. metamodel is built-in by the WFMS vendor
    - Can be implemented efficiently
    - Simple org. metamodel (Person, Department, Role,..., Managed_By, Substitute_Of) often sufficient
  - **Dynamic** org. meta-model allows to change the entities and relationships of the built-in meta-model, or even to create a complete new meta-model
    - Very flexible, but hard to achieve
      - efficiency
      - schema versioning
    - Requires WFMS to dynamically
      - translate modified org. meta-model to an underlying DBMS schema
      - translate staff queries over org. meta-model to queries over org. database

Where Organizational Data Is Managed

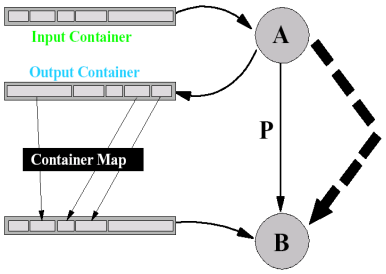
- WFMS manages org. data in its database
  - Pro: Database schema is optimized for access by WFMS
  - Data might be replica of "real" org. database (often the case!)
    - Pro: WFMS does not influence performance of source system and vice versa
    - Pro: WFMS might be a distributed system; replica allow local access, no access to central org database required (efficiency, availability is the issue)
    - Con: Data might run out of sync
- WFMS shares org. data with other systems, and each of the systems can modify the data
  - Ideal when holding org. data in a directory (LDAP, X.500,...) or HR system
  - Pro: No redundant data
  - Con: Performance
  - Con: Org metamodel of directory very likely different from that of WFMS
    Thus, dynamic mapping of org metamodels required at runtime (at build time only, if org data is replica!)
- WFMS has read only access to the org data in another system
  - Same pros and cons as before

Performing Staff Queries

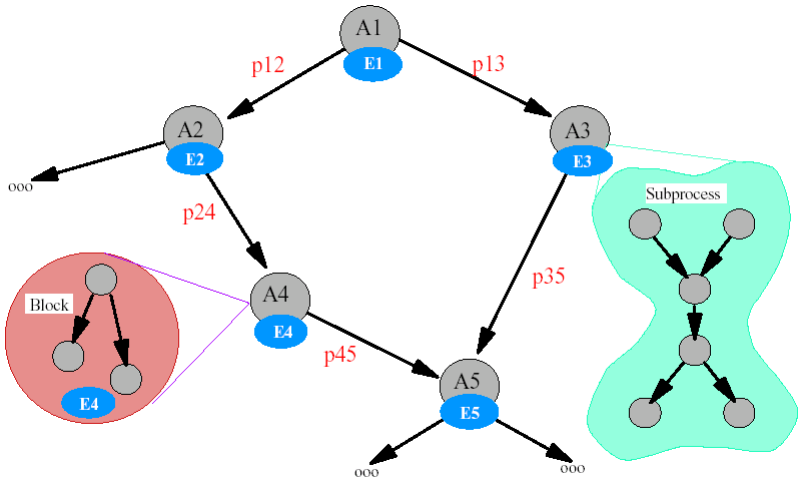
- When org data is managed by WFMS it can execute staff queries directly on its own internal database
- When org data is not managed by WFMS it must run each staff query against the external org data system...
  - Using a staff resolution exit
    - When WFMS must retrieve agents it simply invokes a user provided program
    - This program can perform any kind of computation but must return a set of agents
    - Parameter of the exit can be a query supported by the external org data system
  - By mapping the WFMS org metamodel onto the external metamodel
    - Problem: Can the metamodels be mapped at all without loosing too much semantics?
    - If metamodels can be mapped a tool is needed to transform each staff query formulated in terms of the WFMS metamodel into the external query language
  - By directly using the external system's database
    - Can be done if
      - WFMS and external system use the same type of DBMS (e.g. relational)
      - WFMS metamodel is a "subset" of the external metamodel (e.g. as views on external tables)

"What" Dimension: Data Flow

- Input/Output Container
  - defines data passed to/returned by process or activity
    - based on simple/structured types
      - definitions can be shared
    - can also specify default values
  - provides the execution context
- Data Connectors
  - specify which data needs to be copied where
  - details provided by container map
    - field/data type mapping
    - data transformations
- WFMS at runtime
  - materializes input container instance before activity is started
    - may utilize so-called dead data maps
  - de-materializes output container instance (makes it persistent)



"What" Dimension: Control Flow Specification



Different Types of Activities

- Information Activity
  - inform user to take some actions
  - no implementation
- Program Activity
  - implemented by a program
    - different types of binding
- Process Activity
  - activity implemented as a sub-process
    - different types of connection
- Bundle Activity
  - the same activity is implemented on a set of objects
    - parallel execution as an option
- Block Activity
  - provides DO-UNTIL behavior as special construct
    - process model often restricted to DAG
  - exit condition determines looping behavior

Subprocesses

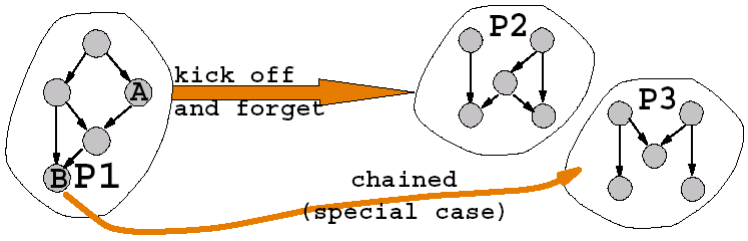
- An activity implementation may be another process, called subprocess from the point of view of the process that owns the implementing activity (so called parent process)
- A subprocess is called
  - **local** if it is performed by the same WFMS that runs the parent process
  - **remote** otherwise
- The WFMS running the remote subprocess can be...
  - ...from the same vendor
    - Private vendor-specific FAPs (Formats And Protocols) can be used for communication (e.g. parameter passing, state exchange, monitoring data,...)
  - ...from a different vendor
    - FAPs must be standardized (e.g. via WfMC) or negotiated between vendors
      - Much more cumbersome than in "homogeneous" environment

Autonomy Of Subprocesses

- A subprocess is a process in its own rights
  - It is derived from a complete and correct process model that has been defined independently
  - Especially, the model of the subprocess can be instantiated alone (i.e. without being invoked by some parent process) resulting in a "standalone" workflow
  - Even as subprocess the workflow runs to a certain degree "independent" from the parent process
- The degree of independence is governed by autonomy rules
- Autonomy rule defines the rights of a parent on a subprocess
  - Completely **autonomous**: Once kicked-off the parent cannot influence the execution of the subprocess
    - E.g. termination of the parent does not terminate the subprocess
  - Totally **controlled**: The life-cycle of the subprocess is determined by the parent process, e.g.
    - Suspension of the implemented activity forces the subprocess to suspend
    - The process administrator of the subprocess must the administrator of the parent
  - Whole spectrum between these extremes can be defined

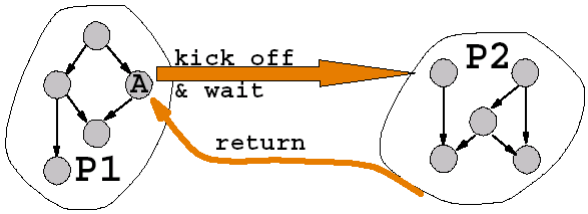
Spawned Subprocesses

- Activity A of workflow P1 causes another workflow P2 to start
- P1 and P2 operate completely independent from each other, e.g.
  - A can complete without having P2 complete
  - P2 might terminate abnormally without affecting P1
- P3 is started when P1 completes (i.e. the end-activity B is implemented by P3)
  - P3 is called "chained": Special case of a spawned subprocess
- Chained workflows are often used



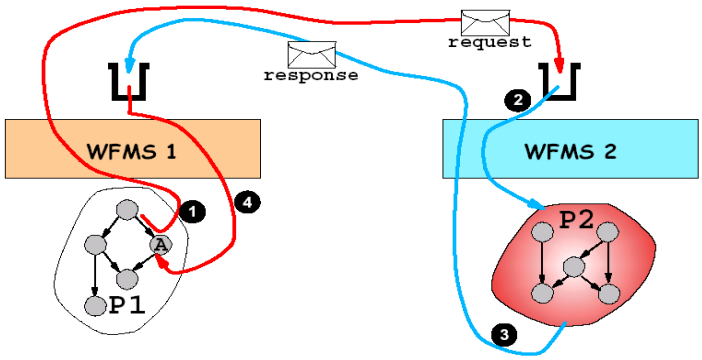
Nested Subprocesses

- Activity A of workflow P1 causes another workflow P2 to start
- P1 and P2 are dependent on each other, e.g.
  - A completes if and only if P2 completes and returns
  - Termination of P1 or A causes P2 to terminate
- A whole hierarchy of nested subprocesses can be defined, i.e. P2 might have nested subprocesses etc.

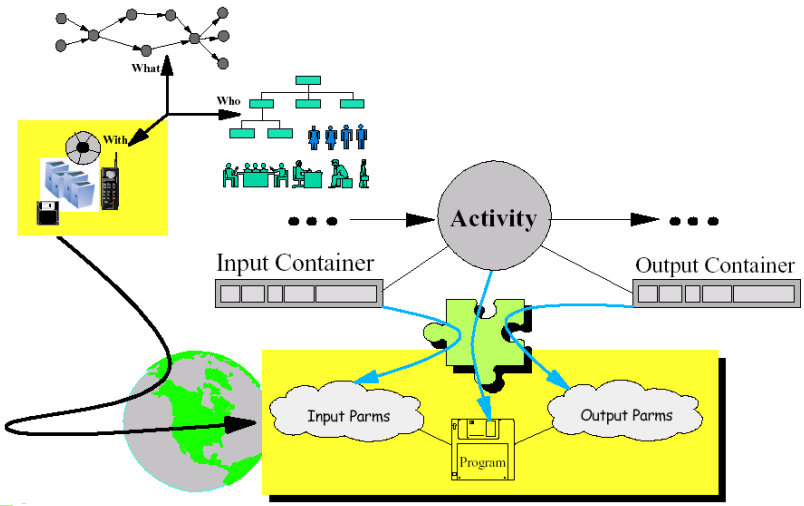


Remote Nested Subprocess

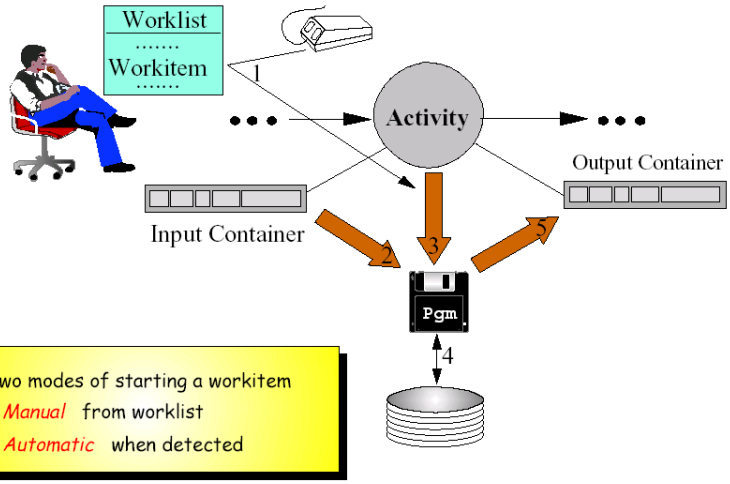
- Request/response messages could be exchanged via message queuing, e-mail etc..
- Exchange mechanism determines properties like guaranteed delivery (MQ), ability for "ad hoc" bindings between WFMSs etc..



"With" Dimension: Program Registration



Invoking Activity Implementations

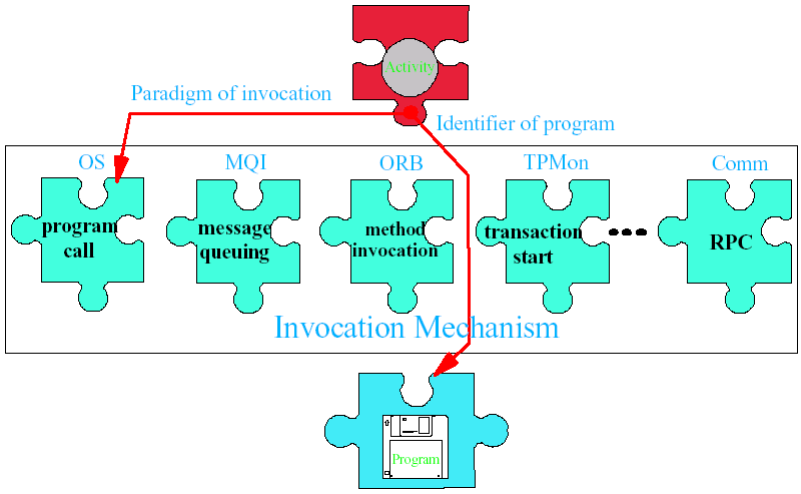


Two modes of starting a workitem
- *Manual* from worklist
- *Automatic* when detected

Decoupling Activities And Their Implementation

- Business modeler want to focus on process models
  - Thus, allowing to specify programs separately and link them to activities separates between activities as conceptual constructs and programs as implementation constructs
- Programs depend on the environment they are running in
  - In general, their signatures depend on the environment
    - Mapping from container to signature must be specified: "Data Mapping Language"
  - Each environment has its own environment parameters and formats
- Programs should be able to be exchanged without requiring to modify process models ("late binding")
  - WFMS resolves actual program to call when activity implementation must be invoked
  - Of course, "early binding" is supported too

Program Invocation: Metadata

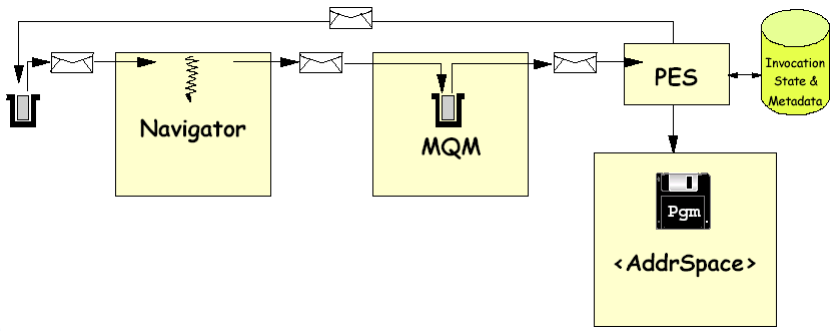


Sample Metadata

- Program call
  - Mechanism to invoke EXE, DLL, CMD files on workstations
  - Requires the name of the program to call
- Message queuing
  - Asynchronous protocol
  - Requires the name of the queue to send the invocation message to
  - Requires the name of the response queue where the reply is expected
  - WFMS continuous navigation iff reply is received
- Method invocation
  - Mechanism to invoke remote objects
  - Requires the identity of the object and the method name to invoke
  - Requires the signature of the method to map container onto
- TP Monitors
  - Requires the transaction identifier
  - Requires to map between containers and "wire format" of transaction

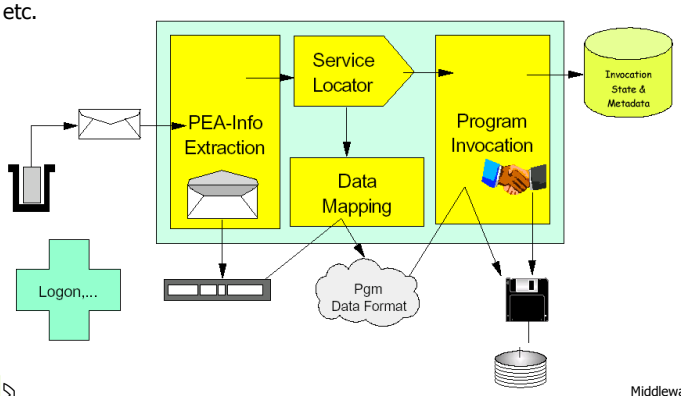
Internal Component Flow

- WFMS Navigator determines program to be executed
- "Execution Messages" sent to launching component called Program Execution Server (PES)
- "Completion Message" sent back to Navigator when invoked program returns

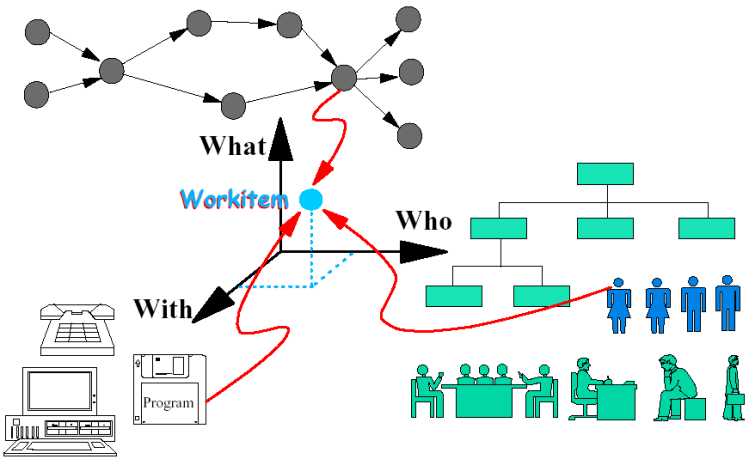


Program Execution Server

- WFMS won't be able to support all mechanisms to launch executables
- Thus, users should be able to build their own PESes, i.e. WFMS
  - provides interfaces between PES building blocks
  - defines required messages exchanged between navigator and PES (User-provided PES (UPES))
- Specific metadata are needed by PES, e.g. security information, mapping prescriptions etc.



The Three Dimensions Of Workflow



Defining Worklists

- Worklist: collection of workitems that have the same common characteristics
- Characteristics are defined via queries on workitem properties
  - Especially, a workitem can be on multiple worklists
    - Worklists of different agents
    - Different worklists of the same agent
- Not only people or program executors may have worklists but also each instance of any element of the org metamodel
  - Worklists associated with an org instance that collects multiple people is called a group worklist
  - All users belonging to the group associated with the group worklist can pick a workitem from that list
- Modes of worklists
  - Pull
    - explicitly request refresh
    - suitable for high throughput environments, where certain worklists might be in constant flux!
  - Push
    - are always up to date
  - Grab
    - deliver a matching workitem on request ("get next workitem")
    - convenient for group worklists

Deadlines

- Most processes must be performed in a certain time
  - E.g. for legal reasons or to meet company specific quality goals
- To support this, the WFMS allows to specify...
  - ...time limits at both, the process model level and the activity level
  - ...actions that should happen when a time limit is exceeded
    - Typical action is to notify somebody who has to take corrective actions
      - This facility is called "notification"
  - The processing of deadlines is called "escalation"
    - Deadlines can also be specified for actions associated with escalations
    - Escalations are escalated via notifying the process administrator
- The time measured for detecting out-of-line situations can be
  - ...the absolute time passed since the beginning of the situation to be monitored ("soccer semantics")
    - Time since activity has been scheduled, arrived on worklist, started to be worked on,...
  - ...the time passed on working on the activity or process to be monitored ("base ball semantics")

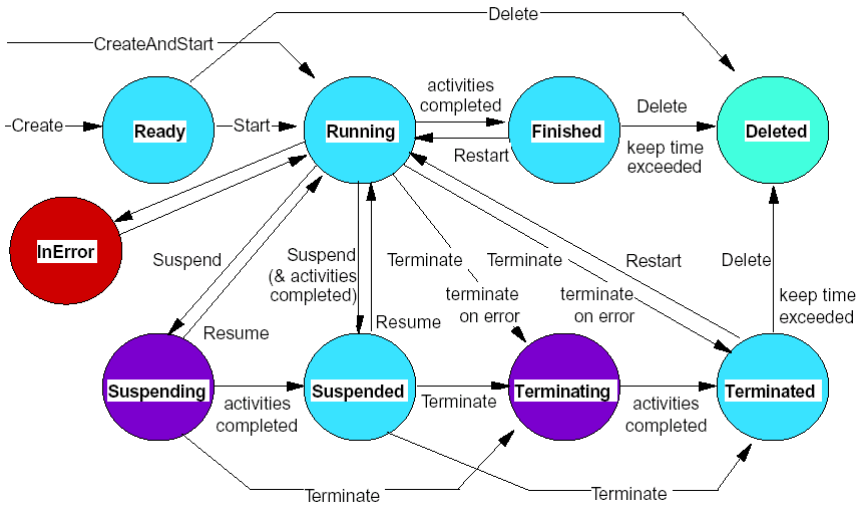
Managing Errors

- A large number of errors can occurs while a workflow is running
  - Activity implementation cannot be located, or it returns wrong data in its output container (e.g. wrong type), or a resolved user is not authorized to execute it etc.
- WFMS supports default actions to cope with such situations
  - Put the activity into the state InError
  - Inform the process administrator to correct the situation
- Sometimes, default actions must be overridden and more specific actions must be taken
  - Both, at the process level or at the activity level

Runtime

- WFMS proactively drives the processes
  - process navigation
  - interaction with end users, applications
- Based on life-cycle models for processes, activities, and workitems

The Life Of A Process



User Session

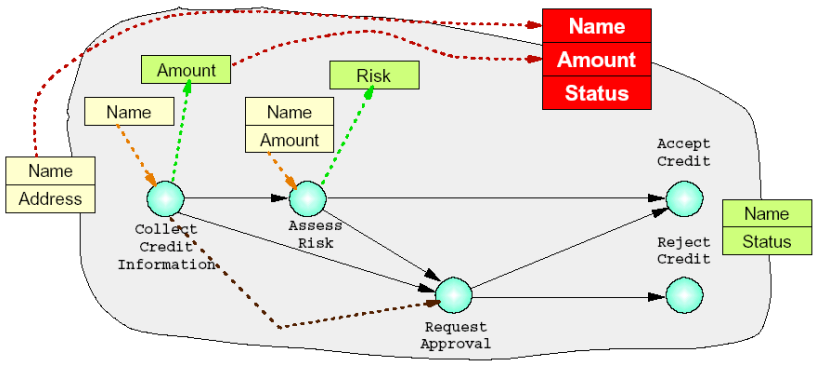
- To work with a WFMS a user has to establish a session
- Session is initiated by starting appropriate client component of the WFMS and by providing user_id and password
- Within a session WFMS assumes that all requests come from the user identified before via user_id and password
- Session is ended...
  - ...when user explicitly terminates the session
  - ...automatically when user was inactive for a predefined period of time
    - used to avoid unnecessary resource consumption
    - reduces risk of unauthorized access if user forgets to terminate the session

Process Queries

- Purpose is to locate a particular process or set of processes
- Two different kinds of selection criteria
  - Operational, e.g. start date, state,... of a process
    - Often used by process administrators
  - Business, e.g. name of customer, order value,...
    - Used by business people, e.g. call center personnel
- Queries return process identifier, especially
  - Process identifier can be used to start process monitor or to retrieve other data about the process
    - Enables combination with prediction capabilities, e.g. time to finish a given process
  - Detailed execution history inquired by accessing the audit trail

Key Container

- Process model may have a key container assigned. Key container can be filled by usual data flow mechanisms. It can be used to locate process instances via queries: No knowledge of process identifier needed!



Audit Trail: Structure

- All important events in the life of a process can be recorded as an entry in the audit trail
- Sample events:
  - Creation, start & termination of a process
  - start, termination, restart & completion of a WI
  - transfer of a WI
  - ...
- Sample fields of an audit trail entry
  - Date & Time the event took place
  - The name/identifier of the event itself
  - Requester of the action (e.g. a certain user or the WFMS itself)
  - Name/identifier of the associated activity, process model
- Key container stored in audit trail to allow for better analysis of execution histories.

Audit Trail: Purpose

- Sample usages of the audit trail
  - Laws require to maintain the life cycle of certain business processes
    - The life cycle will be audited on demand
    - Audit records must often be kept for many years
      - E.g. in airline industry for 30 years
  - Process reengineers want to derive statistical data about processes
    - Average durations of processes or activities
    - Paths taken through process models
  - Audit trail might become extremely huge!
    - WFMS must allow to specify which data is written to the audit trail
      - Influence on amount of data:
        - Fields to record for each event
        - Events to record (e.g. only start and completion, not terminations and restarts)
    - Archiving/Restore functions must be provided
- In distributed environments merge facilities must be provide to consolidate specific audit records from different locations
  - E.g. all records for a specific process model, involving a particular user,...

Monitoring Process Collections

- Notification is appropriate if out-of-line situations occur infrequently
  - Otherwise, people get swamped by notifications!
- Aggregated monitoring functions try to avoid individual out-of-line situations
  - The execution of (definable) groups of processes is monitored
    - Snapshots are taken to trace and graphically represent
      - the workload generated and processed by individuals as well as groups of users
  - Thresholds can be defined in terms of workloads and actions that have to take place when thresholds are exceeded

Leitstand

- Leitstand reports for groups of instances of a particular process model the...
  - State of each activity within each instance of the group
  - Number of current instances within the group
  - Min/Max/Average number of
    - processing time of each activity
    - number of corresponding workitems,...
- Based on defined thresholds...
  - Results are depicted in a color code
  - Actions take place
    - like notification to process administrator [instead of individual notifications!]
- Leitstand reports worklists of users and groups of users
  - Administrator can reassign work from places where work piles up to places where not enough work is available

Process Repair

- Administrator gets notifications about erroneous situations
- WFMS must provide functions to fix such situations, e.g.:
  - Input and output containers must be updatable from the outside
    - E.g.: An activity implementation ABENDed because of incorrect input due to data mapping from incorrect values in an output container. The administrator can manually correct the input data of the ABENDed activity.
    - E.g.: An activity implementation returned incorrect output and the WFMS cannot continue processing (like evaluating the exit condition, performing data mapping,...). The administrator can manually correct the output data.
  - The state of an activity must be updatable from the outside
    - The administrator can force restart an activity (e.g. after repairing its input)
    - The administrator can force finish an activity (e.g. after repairing its output and navigation continues with the manually provided data)
  - The implementation an activity must be exchangeable for all running instances of a process model
    - The implementation might not be locatable, i.e. this is an error applicable to all running instances
      - corrective actions on a per instance base is not sufficient

Prof. Dr.-Ing. Stefan Deßloch
AG Heterogene Informationssysteme
Geb. 36, Raum 329
Tel. 0631/205 3275
dessloch@informatik.uni-kl.de

Technische Universität Kaiserslautern

Workflows and Transactions

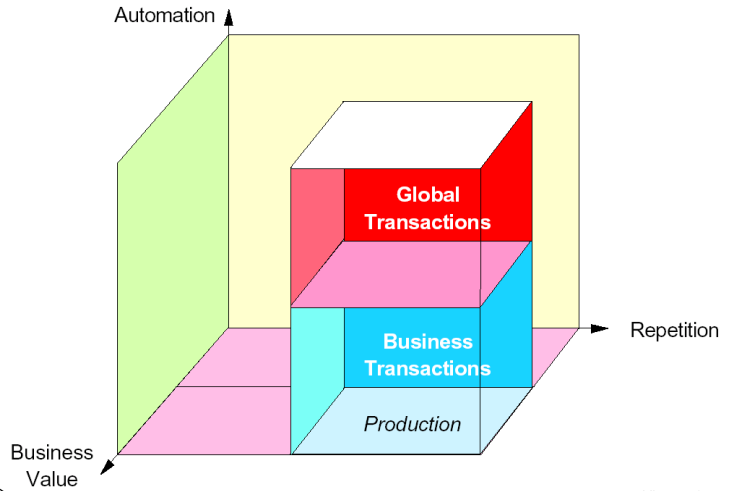
ACID Transactions

- ACID properties
  - Atomicity, consistency, isolation, durability
- Distributed transactions
  - (distributed) two-phase commit
  - DTP X/Open
    - Transaction coordinator, resource managers
    - Transaction "trees"
- Flat transaction model
- Foundation for DBMS, TP monitors
  - Hidden assumption: transactions are short

Long Transactions

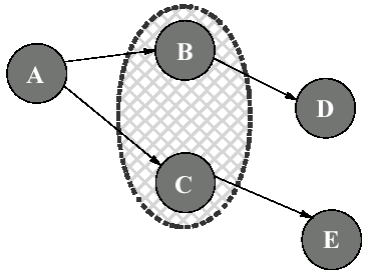
- "Long" is a couple of seconds to years
  - Batches
  - Multi-step transactions
  - Design activities
  - ...
- Basic characteristics are:
  - Must survive (planned as well as unplanned) interrupts
    - Including power-off
  - Backing out of whole transaction due to local failure is not tolerable
- Often, corresponds to a business process

Transactional Workflows



Atomic Spheres (global TAs)

- Set of TAs/activities where either all TAs in a sphere commit, or none
- Properties:
  - Each activity in an atomic sphere is **transactional**
    - Manipulates resources in RM according to DTP X/OPEN
    - Does not establish TA boundaries by itself
  - All connectors entering the sphere have the same activity as an origin
    - Ensure fast execution of sphere (distributed TA)
  - If an activity in an atomic sphere is reachable via control flow from another activity in the same sphere, then all activities along the control flow path are elements of the atomic sphere as well
  - If an activity is rolled back, then all previously completed activities in the sphere are rolled back as well



Atomic Sphere (cont.)

- Specifying atomic spheres may also involve
  - number of retries in case of failure
  - action if the sphere finally fails
    - notify administrator, "skip" the activities of the sphere, …
- WFMS implementation
  - Start global TA when control flow enters atomic sphere
  - Wait for running activties in sphere to complete when control flow leaves the sphere, and commit global TA
    - If commit fails, carry out further steps (repeat, exception WF, …) based on sphere parameters
- Global Transactions: Practice
  - Transaction with multiple participants
  - Atomic commitment is the issue
    - E.g. 2-phase-commit protocol
  - Not realistic across organization boundaries
    - Not only „efficiency" issues but additional legal-, ownership-, privacy-,... issues
    - Especially not in Internet scenarios

Advanced Transaction Models

- Nested transactions
  - Top-level transaction has ACID
  - Closed
    - Subtransaction has A, I, (C)
  - Open
    - Subtransaction has A, D
    - Rollback of top-level TA requires compensation of committed sub-TAs
      - not automated
- Sagas
  - Sequence of (Sub-)Transaction/compensating action pairs
  - DBMS guarantees LIFO execution of compensation actions during abort/rollback of Saga
  - ACID for each sub-TA

Transactional Workflow Evolution: Nested Transactions

- Structure transaction into a tree of subtransactions
- Allow intra-transaction parallelism to speedup processing: siblings may run concurrently
- Overall nested transaction has ACID properties
- Durability of subtransactions is given up (ACI remain)
- Overall nested transaction isolated from other nested transactions ("closed")
- Result
  - Possible speedup of a single closed nested transaction
  - Moderate throughput increase of environment

Closed Nested Transactions

Definition:

A nested transaction is a collection of transactions with the following properties:

1. The collection has a tree structure.
2. Each transaction can commit or abort.
3. The root transaction has the ACID properties.
4. The commit of a transaction will only become effective if its predecessor transaction commits.
   Thus, all transactions can finally commit only if the root commits
5. If a transaction aborts, all transactions of its subtree are aborted too.
   If the root aborts all other transactions abort, too (i.e. subtransactions not durable at time of their commit)
6. Modifications on resources of a transaction become visible to its immediate predecessor transaction ("parent") if and only if the transaction commits.
   Each subtransaction is atomic from its parent point of view
7. Modifications on resources of a transaction are only visible to itself and to its immediate successor transactions ("children").
   Each transaction is isolated from its parent transaction and from its parent's siblings

Open Nested Transactions

- Open nested transactions give up isolation and to a certain degree atomicity
- Subtransactions commit their changes to the outside as soon as they commit
- Consequence:
  Recovery via restoring before-images does not work any more
- Already performed subtransactions of an aborting root must be undone by running application specific logic ("**compensation action**")

Compensation

- Not every action has a reverse (real action)
- In reality, the effects of an arbitrary action cannot be simply undone, i.e. the initial state cannot be recreated
- An action used to reverse the effects of another action is called compensation action
- Semantic Recovery: Recovery schema based on compensation
- Compensation very likely one of today's most frequently exploited techniques in transaction processing

Compensation – Examples

- Compensation attempts to repair actions that cannot be simply undone
  - E.g. an already committed update on a database, sending an email, dispensing money by an automatic teller machine, etc.
- Compensation action is often dependent on context
  - E.g. writing an offer and sending it via mail to a customer
    - If letter is still in outbasket, simply remove it from outbasket
    - If letter is already received by the customer, write and send a countermanding letter
- Compensation often cannot recreate the same state that existed before the proper action had been performed
  - E.g. canceling a flight might cost a cancellation fee
    - Even more complicated, the cancellation fee might depend on the point in time, i.e. it is higher the later the cancellation is requested

Transactional Workflow Evolution: Sagas

- Open nested transactions assumed that compensation actions are performed/scheduled manually (as part of enclosing transactions)
- Sagas require to specify compensation actions in advance and run them automatically on abort

Definition:

A Saga is a sequence [(T1,C1),..., (Tn,Cn)] having the following properties:

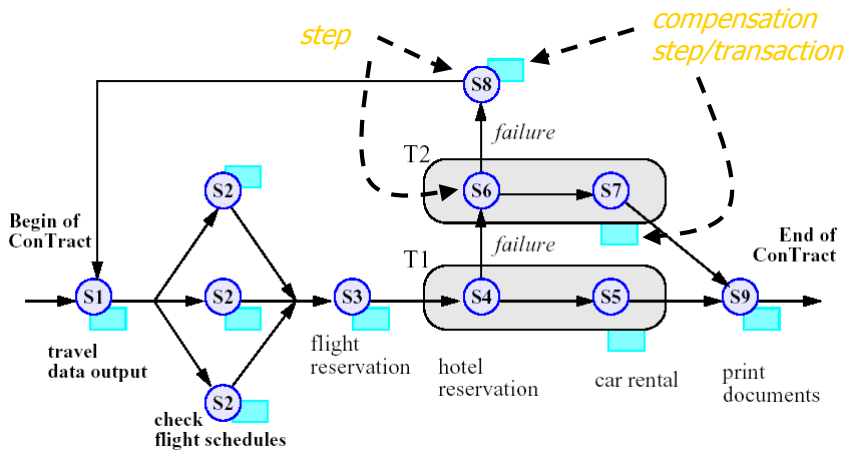
1. T1,...,Tn and C1,...,Cn are two sets of transactions, such that Ci is the compensation function for Ti,
2. [(T1,C1),..., (Tn,Cn)] is executed as one of the following sequences:
   i. [T1,...,Tn], if all Ti committed, or
   ii. [T1,...,Ti, Ci-1,..., C1] if Ti aborts and T1,...,Ti-1 committed before.

ConTracts

- Extends Sagas with
  - Rich control structures
    - Sequence, fork, parallel steps, loops, …
  - Separate description of sub-TAs (**steps**) and control flow (**script**)
  - Management of a persistent **context** for global variables, intermediate results, terminal output messages, …
  - Step synchronisation using invariants
  - Flexible conflict/error resolution
- Target applications are long-running activities
  - Tolerate (planned and unplanned) outages
  - Forward recovery of long-running activity
  - Subset of steps can have ACID semantics (global transaction)
  - (Groups of) steps can be undone after commit using compensation functions
- Limitations
  - Steps have to be transactional
  - Does not cover all dimensions of workflow (data flow, staff assignment, …)

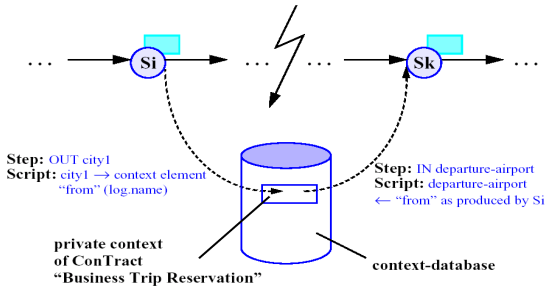
Wächter, H., Reuter, A.: The Contract Model, in Elmagarmid, A.K. (Hersg.): Transaction Models for Advanced Applications, Morgan Kaufmann, San Mateo, CA, 1992, S. 219-264.

ConTracts – Example



Forward Recovery and Context Management

- **Forward Recovery**: after a crash, recover youngest step-consistent state and "roll-forward"
- Requires persistent **context management**
  - Context element attributes
    - Logical name, conTract identifier, step identifier, creation timestamp, version number (multiple activations of same step), counter (parallel activations)



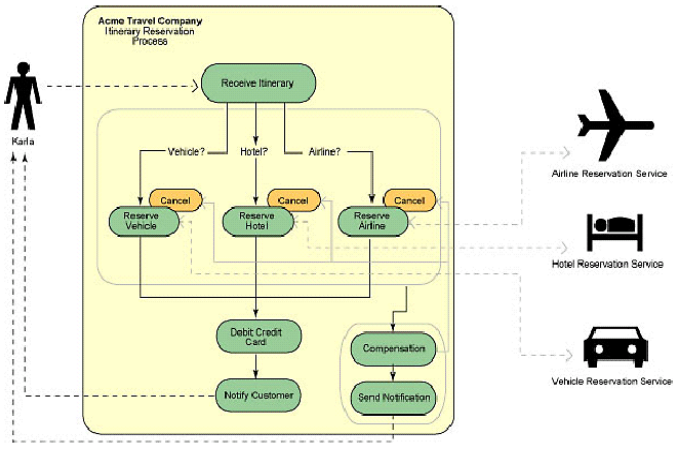
ConTracts – Compensation

- Compensation is directed by user
  - started upon explicit request of application program or administrator
- Rules
  - Every step/transaction must have a compensating transaction
  - At commit of a step, all data needed for compensation must have been computed/persisted
  - Local data needed for compensation steps must be safe from deletion until End-Of-Contract
  - Compensation of a ConTract forces rollback of all running steps and prevents starting new steps
  - Compensations can be aborted
    - Requires repeating the compensation
    - No (automatic) treatment of repeated compensation failures

Compensation Spheres

- Set of activities that must complete successfully as a whole
  - Otherwise it must be undone semantically
- Activities can be arbitrary
  - Don't have to be realized as transactions
- Each activity in the sphere and the compensation sphere itself is associated with a compensating action
  - Could be the NULL action
- A compensating action may be an activity or (complex) business process
- If an activity fails
  - Compensating actions of all completed activities in the sphere are executed in 'reverse' order
  - Compensating action associated with the compensation sphere is executed
- Advantages compared to explicit modeling of exception/failure handling steps as part of the process model
  - Reduces complexity of the process
  - Separation of regular business logic from exception/failure handling
  - Increased flexibility
    - compensation of spheres that have completed successfully

Compensation Spheres – Example



Properties of Compensation Spheres

- **Compensation granularity**: defines the manner in which a sphere is aborted
  - discrete – each activity is compensated separately, in reverse order
  - global – abort by invoking a global comp. activity for the whole sphere
- **Mode** of the sphere: defines details about the overall compensation
  - retry – try to execute the sphere again after running the compensation action
  - undo – run the compensation and then inform a predefined agent
  - rerun – stop processing in the sphere, rerun without any compensating actions
- **Proliferation** of the sphere: defines whether dependent spheres (through output-input containers) should be aborted as well
  - sphere vs. cascade
- Activities can have properties that define the details of compensation behavior
  - **deep vs. shallow compensation** for subprocess activities.
- Special **compensation containers** can be introduced to establish the context used for compensating actions

Phoenix Behavior

- Forward Recovery ("recover out of the ashes")
  - workflow itself must be recoverable
    - persistent, recoverable workflow context (using DBMS)
    - reliable messaging for communicating workflow events
  - implementations of activities must be included in the recovery processing of the workflow
    - stratified transactions

Conclusions

- Business (Re-)Engineering, Business Process Modeling
  - goal: efficient execution of core business processes
  - explicit specification of process models
    - focus on control flow, rudimentary data flow, organizational aspects
  - process optimization and analysis
    - static optimization
    - simulation (analytical, discrete event/based)
- Workflow Management Systems
  - middleware for management, control and execution of business processes
  - build time
    - extend process models created using BPR tools
      - data flow, control flow details, organization aspects, activity implementation
  - run time
    - work item lists, process life cycle and process management, audit trail
- Workflows and transactions
  - ACID is too strict for long transactions
    - only appropriate for individual activities or restricted subset of activities (atomic spheres)
    - advanced transaction concepts, compensation required
  - ConTracts
    - example for transactional workflows
    - activities have to be ACID transactions!
  - compensation spheres: sets of semantically linked transactional (sub-)activities