CONSIDERATIONS ON THE PARTICULAR FEATURES FOR PROCESS AND WORKFLOW MODELING

Victoria Iordan [(a)], Alexandru Cicortas [(b)]

[(a)] Department of Computer Science, West University of Timisoara, Romania
[(b)] Department of Computer Science, West University of Timisoara, Romania

[(a)]iordan@info.uvt.ro, [(b)] cico@info.uvt.ro

ABSTRACT
Modeling processes and workflow for complex systems constitutes a challenge for designers. The resources that are used in systems are in limited amounts and these must be shared between the processes and workflow instances activities.

Our considerations refer to the need of a data base in that the workflow constituents: instances and activities must be represented in a data base and also their time evolution. Concerning the resources that are used in the processes and in the workflow instances execution must be adequately represented. Two important classes of information systems, Workflow Management Systems (WfMSs) and Enterprise Resource Planning (ERP) systems, have been used to support e-business process redesign, integration, and management.

We intend to propose a model for resource representation and allocation and as conclusion propose considerations concerning the design of processes, workflows and workflow management systems.

This paper is partially supported by the project PN II 91-047/2007.

Keywords: modeling, workflow, processes, resources, resource management, activity coordination.

1. INTRODUCTION

Real world systems became more and more complex and require a lot of tools for their design and execution. In these systems having many processes is required that these processes: must cooperate; share the same system resources; coordinate and communicate one to other. In many systems there are different organizations that have their own processes that intervene in the fulfilling of some goals. The workflow that is designed to manage the process must be able to satisfy the above requirements.

Between the major problems we consider to be those of the resources and of the failures for both the processes and the workflows. How the ERP also manages the organization resources many considerations will be done also about it, based on (Kearney 2005).

Concerning the concepts, we intend to use those of the WfMC but we do not give the details that can be found in (WfMC 1994; WfMC 1996; WfMC 1999; Kearney 2005). Framework constitutes an indispensable feature in process and workflow design. Frameworks manage complexity by reducing vague or complex topics to a set of simpler questions (Sharp 2001). They organize work, maintain focus during interviews or facilitated sessions, ensure coverage of all aspects of a topic, and provide a standard format to document the findings. One of the most important frameworks is the one we use to consider all dimensions of a process: Workflow design, Information technology, Motivation and measurement, Human resources, Policies and rules, Facilities design.

Ontology or a reference framework (van der Aalst 2002) is a system of defined terms that describe a particular field of knowledge. Some specific aspects like:

- distribution of work (human interaction with processes);
- the relationship between the principal and the contractor;
- organizational structures and the management of processes;
- the role played by information systems in the establishment and management of business processes

must be seriously analyzed.

The paper is organized as follows. The second section treats the process modeling; in the third section, the specificities of workflow design are presented: BP re-engineering, workflow failure recovery, compensation and rollback and fault tolerance. The third section treats the resources. The fifth section analyzes the WfMS and ERP; the sixth section details our considerations concerning the resources, analyzes the failed workflow and their re-engineering. The last section sketch out the future works.

2. MODELING PROCESSES

2.1. General considerations on frameworks

A framework is a set of classes that embodies an abstract design for solutions to a family of related problems. According to (Fayad 1999, Eriksson 2002), the four primary benefits of frameworks can be described as follows:

- Modularity. Frameworks enhance modularity by encapsulating unstable implementation details behind stable interfaces. This modularity helps to improve software quality by localizing the impact of design and implementation changes. The localization reduces the effort required to understand and maintain existing software.
- Reusability. The stable interfaces provided by frameworks enhance reusability by defining generic components that can be reapplied to create new applications. Framework reusability leverages the domain knowledge and prior effort of experienced developers in order to avoid recreating and revalidating common solutions to recurring application requirements and software design challenges. Reuse of framework design components can yield substantial improvements in programmer productivity, as well as enhance the quality, performance, reliability, and interoperability of software.
- Extensibility. A framework enhances extensibility by providing explicit hot spots that allow applications to extend its stable interfaces. These extension points systematically decouple the stable interfaces and behavior of the application domain from the variations required by instantiations of the application in a particular context. Framework extensibility is essential to ensure timely customization of new application services and features.
- Inversion of control. The runtime architecture of a framework is characterized by an inversion of control, often referred as the "Hollywood Principle" (Don't call us, we'll call you). This architecture enables canonical application processing steps to be customized by event handler objects that are invoked via the framework's reactive dispatching mechanism.

The previous benefits of using frameworks allow having many advantages that are significant in design and modeling of systems. The modularity can be viewed in many ways. One is that where the system is service oriented and another is that where the system is component oriented. Permanent systems evolution needs for an adequate extensibility. When events occur, there actions of the system consist in invoking appropriate methods, which perform application-specific processing on the events. Inversion of control allows the framework to determine which set of application-specific methods to invoke in response to external events.

2.2. Details concerning process modeling

The process consists of a number of tasks that need to be carried out and a set of conditions that determine the order of the tasks.

The task is a logical unit of work that is carried out as a single whole by one resource (van der Aalst 2002). There exists another, in our opinion, view due to the fact that one task can be accomplished by more than one resource and in this paper that is the vision about it. The resource is the generic name for a person, machine or group of persons or machines that can perform specific tasks. This does not always mean to say that the resource necessarily carries out the task independently, but that it is responsible for it.

In most organizations there exists a hierarchy under which assignments that people receive can be passed on to people further down the hierarchy. A person who is assigned a task can be considered as a contractor, being also a resource. In many processes two or more tasks must be performed in a strict order. The processes include selection or repetition and as mechanisms are used in process structures: sequence, selection, parallelization, and iteration. The human participation in the process is very complex and in this paper it is detailed only when it implies our model goals.

In many works and products are used the cases for processes and workflow instances (see the van der Aalst works and papers). In most of processes the activity can be seen as a work (contract) that is required by someone (the principal) and executed by another one (the contractor). The communication protocol is illustrated in Figure 1.

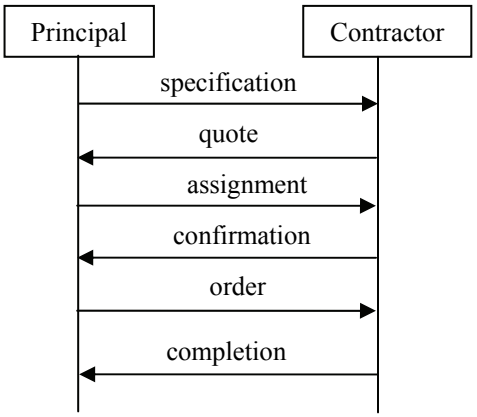


Figure 1: Communications protocol

2.3. Managing Processes

A management system can manage several systems, and in doing so, it ensures the ability of the managed systems to communicate with one another and with the outside world, that is, the managed system at a higher level. Between the management system and the managed system there occurs an exchange of information. This enables the management system to communicate objectives, preconditions and decisions to the managed system and the managed system conversely reports back to the management system.

Based upon these reports, the management system may revise the objectives, preconditions, and decisions. Process management has long been divided into four levels. The distinction between these is based upon the frequency and scope of the decisions to be made. By scope, we mean two things: the period of time over which the decision has an influence and its potential financial impact. The four levels are as follows:

1. Real-time management.
2. Operational management.
3. Tactical management.
4. Strategic management.

Decision making is an important feature of (process) management. The following phases that are always passed through when solving decision problems: Definition (establishing exactly what the problem is and, in particular, within what scope a solution to it must be found), Creation (formulating one or more solutions that fall within the scope defined or satisfy an optimization criterion), Evaluation (assessing different solutions, for instance by multi-criteria analysis) and Selection (selecting one solution that works in order to implement it).

2.4. Information Systems for Business Processes

Workflow management it is one the main trend in our days. It takes the business processes out of the applications. A workflow system manages the workflows and organizes the routing of case data amongst the human resources and through application programs. Just as databases are developed and used with the assistance of a database management system, so workflow management systems (WFMS) can be used to define and use workflow systems.

3. MODELING WORKFLOWS

Workflow Concepts used are: case, activity, and task, work item (van der Aalst 2002). Their interdependencies are shown in Figure 2.

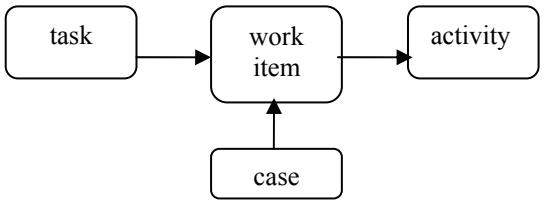


Figure 2: The relationship between the terms task, case, work item and activity

Based on the process definition we can specify:

- what tasks need to be performed for a particular category of case;
- the order in which they must be carried out.

The process definition does not indicate who should do it. The way in which the work items are allocated to resources (people and/or machines) is very important to the efficiency and effectiveness of the workflow.

A workflow system enables an organization to use and manage structured business processes. One important property of workflow systems is that, it becomes easier to change business processes. Exchanging or combining tasks or rearranging resource classes are easy modifications. Improvements influence performance criteria such as completion times, utilization of capacity, level of service and flexibility.

3.1. Business Process Re-engineering

Exist many tools for analyzing workflows that we not intend to detail here, but some specific problems like: soundness, reachability, performance analysis is of major importance (also these can be found in the works and papers of W.M.P. van der Aalst).

We shall consider the relationship between business process re-engineering (BPR) and workflow management. We can define BPR as the fundamental reconsideration of business processes. Its objective is to bring about entirely new business processes which enable drastic improvements to costs, quality and service. In order to achieve this objective, radical changes are often necessary.

Process-oriented thinking is crucial in the use of workflow management systems. Supporting old processes with a workflow system will only deliver a limited amount of improvement. Dramatic improvements are only possible if the old processes are separated from and replaced by new ones.

3.1.1. Workflow Failure Recovery

Workflow Management Systems more and more become the basic technology for organizations to perform their daily business processes and their workflows (Eder 1996). A consistent and reliable execution of such processes is crucial for all organizations. We claim that this can only be achieved by integrating transactional features especially workflow transactions into WFMSs.

Workflow recovery concepts are necessary for the reliable and consistent execution of business processes in the presence of failures and exceptions. In the Workflow Management Systems the workflow engine manages the workflow instance execution.

Many works and papers deal with workflow recovery that has connection with database transactions. Workflow technology targets supporting reliable and scaleable execution, for workflow management systems (WfMS) to support large-scale multi-system applications, involving both humans and legacy systems, in distributed and often heterogeneous environments (Luo 2000).

In case of failures, workflow processes usually need to resume their executions from one of their saved states, called a checkpoint, achieved by saving the states

from time to time persistently. The activity of restoring a checkpoint and resuming the execution from the checkpoint is called rollback. Those techniques have long been used in database systems. A checkpoint is an action consistent checkpoint if it represents a state between complete update operations. A consistent state in the database domain is a state when no update transactions were active. This checkpoint representing a consistent state is a transaction consistent checkpoint. A checkpoint does not need to satisfy any consistency constraints. But recovery after failure must always guarantee that the resultant state is transaction consistent even though any checkpoint used may not be. A checkpoint can be either local or global. A local checkpoint is a checkpoint taken locally, with or without cooperation with any other local check pointing activities at different sites. A local checkpoint can be a fuzzy or consistent checkpoint. During global reconstruction, a set of local checkpoints, usually taken at different site, will be used to find global consistent state. To facilitate the global reconstruction, a global checkpoint, derived from a set of local checkpoints taken at different site, provides a rollback boundary, thus reducing the recovery time.

Failure recovery is essential for transactional workflow management systems. When a failure occurs, compensation is usually used to rollback corresponding processes and ensures semantic atomicity. Previously, many mechanisms have been proposed while performance problem is rarely considered. This is a limitation since compensating activities are rather expensive.

3.1.2. Compensation and rollback

This can be treated as for the data base transactions. The treated way that a transactional scenario responds to failure differs from the way that non transactional scenarios respond to failure. When a non transactional scenario fails, the collaboration simply logs an error and terminates. When a transactional scenario fails, the scenario rolls back to leave data in a consistent state across all of the involved databases. When an error occurs, sub-transactions might have already caused applications to commit work. Therefore, rollback is done through the use of compensation steps, actions that counteract the effects of other actions. A compensation step executes only during rollback.

3.1.3. Fault Tolerance

Fault tolerance is the ability of an application to continue valid operation after the application, or part of it, fails in some way (Dialani 2002). The object data may return (rollback) to previous values if the current values are lost, and processes may return to a state in which a message is re-sent, if the previous attempt apparently failed.

In order to return to a previous consistent state, an application must record a replica of its previous state. The state of a process can be copied using a check pointing mechanism, or only the incremental changes to the process state using a logging mechanism and in every one of those two cases, it is possible to rollback to a previous valid state.

In distributed applications one process may fail without the other processes being aware of the failure. This can lead to the state of the application as a whole (the global state) being inconsistent.

An application is in a globally consistent state if whenever the receipt operation of a message has been recorded in the state of some process, then the send operation of that message must have been recorded also. Is needed a fault tolerance mechanism for distributed applications to keep an application to a consistent global state, or return to the last known consistent state (maximal state) in case of failure.

In figure 3, we present an overview of the fault tolerant system, as applied to the Web Services architecture.

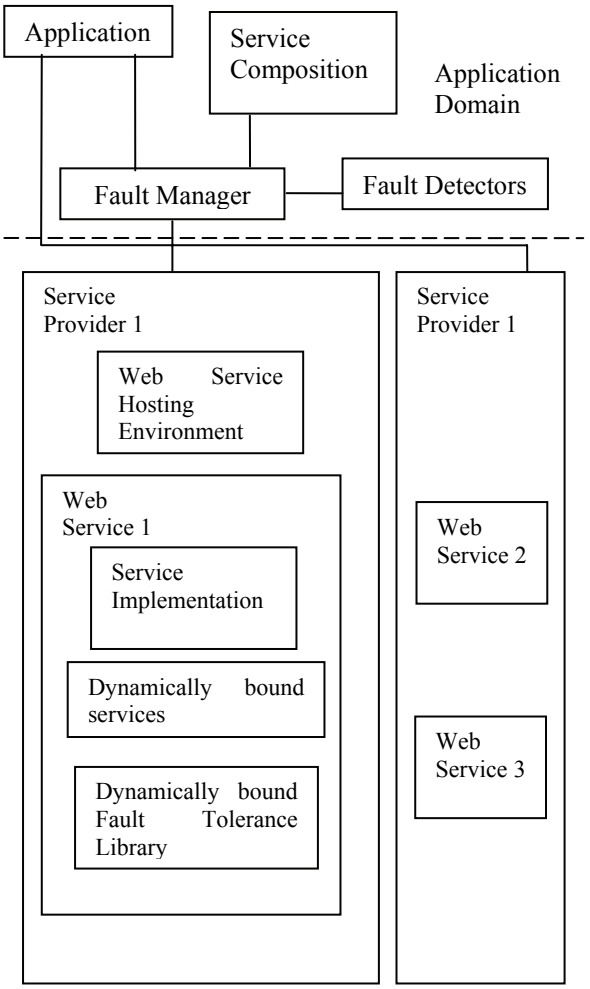


Figure 3: Architecture for Fault Tolerant Web Services

The frameworks for assuring the fault tolerance are various. For example, CORBA employs a two-pronged strategy to recover from a fault, namely the local recovery mechanism and the global recovery mechanism. The context of a local recovery is restricted to recovery of an individual service instance, while

global recovery applies to the entire application. The local recovery mechanism tries to revive the service instance with minimal or no intervention by the global recovery mechanism. A local recovery mechanism escalates the failure notification to the global recovery mechanism in case of its failure to recover the fault locally. The architecture imitates an hour glass model to restrict the dependency between the two layers to a minimal set of interfaces, for co-ordination between the two layers.

4. RESOURCES

The resource management in processes and in workflow was treated in many ways but only some of them will be referred in the following (Lerner 2000).

Many models for resource representation and management have particular specificities. Between those the (Lerner 2000) illustrates an interesting one and we believe a useful model. For resource model designing are needed specifications that they can be used as prescriptions for supporting the resource allocation and scheduling through the application of tools. It is needed to develop and evaluate resource specification formalisms that are sufficiently rigorous that they can be used to reason about complex activities from the processes and how to support these processes most efficiently. The lack of resources causes contention, occasions the need for some tasks to wait for others to complete, and generally slows down accomplishment of larger goals. Often potential delays can be avoided or reduced by using resource analysis to identify ways in which tasks can be made to execute in parallel that avoid resource contention.

In addition to the scheduling concerns, reasoning about resources requires an understanding of the similarities and differences among resources as well as the precise needs of the activities being coordinated. With this understanding it is possible to identify in which situations only a particular resource will suffice and in which situations any of a class of resources may be acceptable for the task. This identification of critical resource needs can also facilitate the identification of likely bottlenecks in the execution of a process or workflow.

The models require capabilities for describing and using major features that will allow emphasizing the resources and their possible interrelations (like similarity) and interactions in order that the execution of the activities can be done without major changes. From the different models we consider that the proposed model from (Lerner 2000) gives a resource model able to emphasize the main requirement for a complex model one. In the model are detailed many important features like:

- similarity between the resources that can be also source of major improvements in real systems;
- resource attributes (like measure, availability, etc.);
- relations and their types.

The resource management component provides information about the types and availability of resources and to track their usage. It also maintains a history of requests for resources that failed, and thus, evaluate the need for more resources of a particular type, is important in speeding up of an overall activity. The resource manager allows acquiring, reserve and releasing the resource under a centralized way.

4.1. Resource Management some Considerations

Resource management problems have almost the same underlying basic structure in common as timetabling problems.

Resource Allocations Modeling the consumption of Resources by Tasks is crucial to resource management and Rules are used in order to do it. The priorities and conflicts are also used widely in the resource management.

Tasks represent the actions to be performed during the execution of a Plan to achieve the set objectives. Tasks usually require certain Resources to be accomplished. The duration of some Tasks even depends on the amount of Resources assigned to them. Since there are Tasks that for example need to be carried out consecutively or concurrently, or with certain time-lags, interdependencies between them regarding their scheduling have to be modeled too.

Rules represent the constraints in models. They can be divided into two categories:

- Active Rules change models to ensure that they are met. Typical examples are the interdependencies between Tasks. An Active Rule could schedule one Task to start when another Task ends.
- Passive Rules are used to verify models and indicate violations, so it is possible to react immediately, like “no Employee should work more than 40 hours per week”.

The used Rules are usually very specific to the domain and even to the undertaking to be modeled.

Priorities are used to express the amount of importance assigned to certain elements of models. A Task with a high Priority assigned to it, for example, is more important to be executed than another Task with a lower one. The numbers of used Priority Levels as well as their meanings are usually very specific to the respective domain or even to the undertaking.

The Passive Rules are used to indicate problems. Those indications are represented by Conflicts and need to be managed to be of use.

Additional Functions are obvious for a good resource management:

- Reporting. Throughout the realization of a Plan, the involved parties usually need to be informed about the aspects which are of

interest to them. Teachers, for example should receive a timetable different from the one for a certain class. So the application should support the creation of Reports intended for different audiences like to-do lists for the members of a team, financial reports for the accounting department, et cetera.

- Tracking. The term Tracking refers to the process of constantly collecting data about the actual course of the implementation. This is necessary to make post-implementation analyses possible, which can be a significant part of a project. For example, the profit has to be figured out or somebody needs to be found guilty for the failure of a project.

5. WFMS AND ERP

A WfMS is implemented based on a process specification and execution paradigm. Under a WfMS, a workflow model is first created to specify organizational business processes, and then workflow instances are created to carry out the actual steps described in the workflow model. During the workflow execution, the workflow instances can access legacy systems, databases, applications, and can interact with users.

ERP systems are implemented around the idea of prefabricated applications. To achieve better fit between the prefabricated applications and the needs of the organization, ERP systems must be configured by setting various application parameters. The more parameters an ERP application has, the more flexibility in configuring the business process. However, the workflow model in conventional ERP systems is not explicitly specified because it is embedded in the applications and the parameter tables.

For understanding the differences between WfMS and ERP systems is necessary make a distinction between flow logic and function logic (Cardoso 2004). In ERP systems, flow logic and function logic are both embedded in applications and parameter tables. In contrast, a WfMS separates the two explicitly. Flow logic is captured in a workflow model, usually graphically represented, and function logic is captured in the applications, data, and people the model invokes. Thus, a WfMS enable developers to separate the flows among a system's components (applications, data, and people) from the workflow model. Workflow systems are process-centric, focusing on the management of flow logic. On the other hand, ERP systems are data-centric, focusing on managing function logic via a common homogeneous data infrastructure across the organization to support multiple applications.

One of the problems is that ERP and WfMS technologies are often managed by different groups within an organization.

In (Cardoso 2004) are compared the two technologies along three main dimensions: domain scope (that defines the suitability of a system for a specific type of application or organization), technological scope (characterizes the systems based on their technological capabilities) and system implementation. WfMS and ERP systems have been developed with distinct sets of technological capabilities. We can highlight these different capabilities by examining the different types of applications each system supports. Business process technology focuses its attention and effort on supporting three different types of applications:

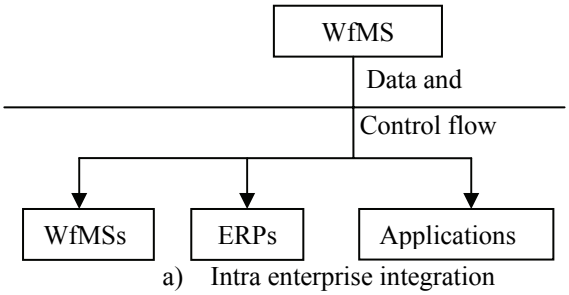
1. workflows involving humans;
2. workflows involving systems and applications;
3. transactional workflows.

ERP systems constitute applications that focus on the integration of data. The objective of ERP applications is to provide an integrated solution to all business areas. The underpinning of shared data structures across many applications eliminates the need to pass data step-by-step among applications by accessing data from a common structure. The focus of ERP systems is mainly on structured data transactions. ERP modules operate directly with common interoperable databases to ensure consistent information for all purposes. This makes the manipulation of data easy. The ERP concept makes the strong assumption that data infrastructures are homogeneous across the organization, that is, the data is stored in interoperable databases and in some cases, the databases used are all from the same vendor.

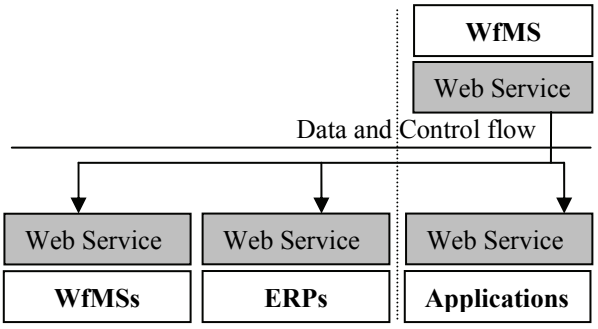
WfMSs can address all three types of workflows (outlined above); however, they are most suitable for modeling workflows involving humans and software systems, especially if the systems are autonomous and heterogeneous. On the other side, ERP systems are more appropriate to model transactional workflows, which are data oriented.

In the following some considerations on WfMS and ERP are done. Both WfMS and ERP systems will play a major role in organizations' application integration. We presented above how WfMS and ERP systems can be used independently or together to address intra- and inter-organizational application integration.

The current directions in application integration point to the architectures shown in figure 4.



a) Intra enterprise integration



b) Inter-enterprise integration

Figure 4: Intra and inter-enterprise integration using WfMS and ERP systems.

In both intra- and inter-organizational integration, flow logic is being separated out and captured in WfMS with function logic found in ERP systems, other applications or other WfMS. In intra-organizational area, we see this represented in the moves by ERP vendors to integrate WfMS into their ERP architectures and emergence of EAI/BMP workflow based tools.

In some cases, system integration expertise may be needed to manually code and integrate with the WfMS special features such as to link the workflow engine to legacy applications, set transactional properties, define recovery procedures, etc. Additionally, the integration of access control and user rights on both the workflow as well as the applications requires additional effort in a heterogeneous environment (zur Muehlen 2000).

6. OUR CONSIDERATIONS

6.1. Resources

Due to the major importance of the resource and their utilization we intend to propose some particular framework for their representation and a support for their management. For the resource:

- characterization we define its: type (RType), identifier (RId), name (RName), its decomposability (RDe) as logical value, the minimal amount necessary for being usable (effective) (RMin), the current available Amount (RCAv), creation instant time (RCret), similar functionality with (RSF);
- requirement of usage every activity or task that uses it, must specify its: identifier (RId), the necessary amount in the case if it is decomposable (implicitly 1 if it is not) (RNec) and the estimated time interval for its usage (REs);
- used entities concerning their states and localizations in time for every entity that can be used must specify its identifier (RId), state (RSt) that must be allocated or active, the user activity or task that uses it (RUser), and starting time interval in this state (RStaT). The place of the resource is an optionally attribute that can (or not) be used in many specific systems where it is appropriated;
- unsolved requirement list that contains the activity (task) identification, the required amount and the time instant when it was required;
- history that for every entity contains every state in that the entity was;
- history of waiting's that contains the item from unsolved requirement list and the time that it waited.

Resource manager will execute some specific actions:

- whenever when a new resource appears an item in the characterization will be placed;
- when appears a supplementary amount the RCAv is increased and the RCreT is modified;
- every time when an activity or a task asks for a resource and if it exists in the required amount, it will be created an item with the state of the entity and will decrease with the taken amount the current available amount (RCAv) and put the item in the allocated state. If the amount required does not exists the activity (task) enters in the blocked state (waiting for the resource availability) and an item will be placed in the unsolved requirement list;
- whenever when the activity or task starts its execution, the of its appropriate item (RSt) is modified into active;
- every time when a resource is released by its user (activity or task), it must enter in the history an item and increase with the used amount the current available amount (RCAv);
- every time when the RCAv is modified (by increasing it) then the unsolved requirement list is analyzed and if it is possible then the requirement is satisfied and the item is placed in an history list for waiting;
- when an action or a task that required a resource and it was in the waiting state it can cancel its requirement. In this case the Resource Manager will remove it from unsolved requirement list and will put it in to history.

The generic resource manager functions are given in the figure 5. As it can be seen from the above descriptions, the resource description is a complex one. Also it must contain some similarities between the resource functionalities (these are not detailed in the paper). The similar functionalities must be also qualified in performance. Here is not in detail specified the proposed multiagent model for the Generic Resource Manager, that will be given in another future paper. It must be able to have some intelligent agents that are able to learn from the history requirements for the resources.

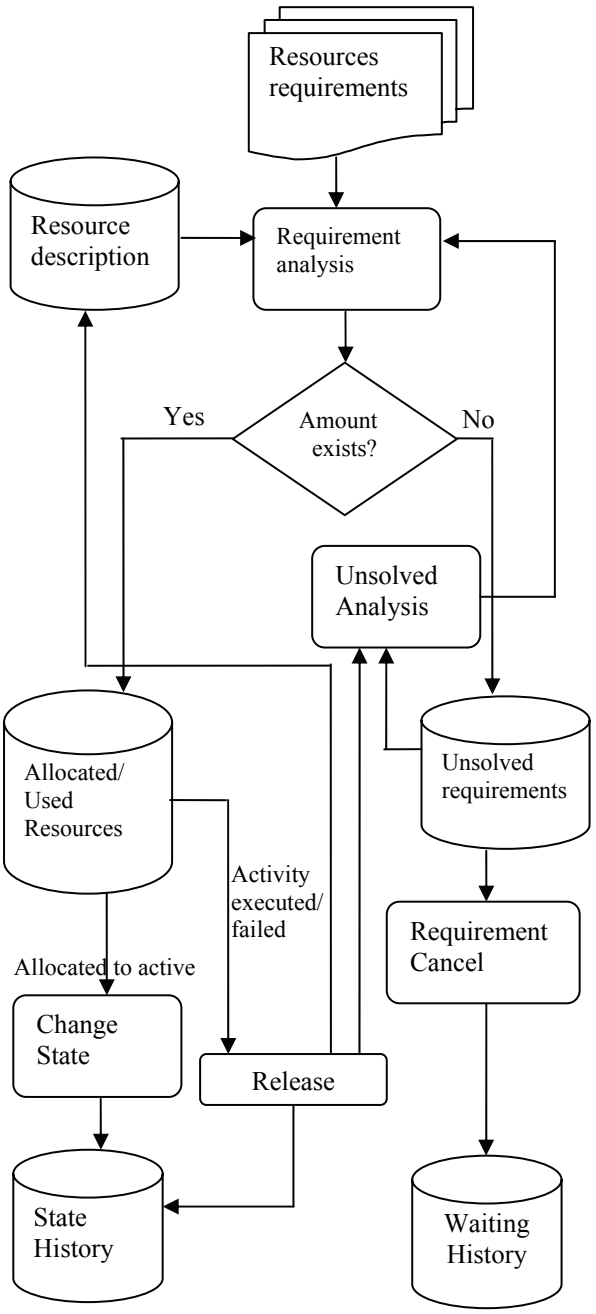


Figure 5: Generic Resource Manager Functions.

6.2. Reengineering failed instance workflows

An instance workflow is executed until the activity that failed and that generated the instance failure. The workflow engines that execute (run) the workflow instances interact with the system, the user application and in many cases with the user (human person). When an activity fails its workflow instance must be analyzed and must detect the reason of the failure.

Many products (like ACTIVE BPEL for BPEL4WS) use compensation and other works propose other solutions like worklets (Adams 2006) in order to solve the problem.

From proposed solutions in many works and papers and due to the fact that the human persons interact with and manage the systems and workflows, we identified some requirements for achieving a better workflow management:

- (unique) identification of workflows, workflow instances and activities;
- the activities results (after their execution) can be useful or volatile (useless) in the case of the reply the failed workflow instance;
- during the workflow instance execution from its start to the failed activity the state of the real system resources and their states (and also the data bases transactions) were affected. In the case of a reply of the workflow instance the state of the system (and the appropriate data bases) and the resources were affected and all these must be restored to the state just before the failed activity.

In (Cicortas 2006), we proposed a multi-agent system (MAS) that is able to analyze the workflow instance that was blocked and based on its state to redesign the reminded part of the workflow instance that will be executed. This analysis must decide for every activity that its result can be or not can be used in the conditions that allow it, if we dispose for appropriate information. In a workflow, we have some activities whose result is in one of the following categories:

- the activity result can be retained and in future can be used, we can say it is “remanent”;
- the activity result is volatile in future, it does not be used, if we want the activity result in the future, we must reply it;
- the activity result has affected the real system state and the system state must be restored in the case if we want to replay this activity we must remove this “effect”.

If the workflow fails, then it fails due some activities that it failed. The previous Workflow activities were executed and based on their result as previous analysis was done, we can find the activities whose results are reusable and are remanent.

In the case of the reply of the Workflow we must make an analysis that: restores the activities that were affected in some way the system state, find the activities that have reusable and remanent results, reconstruct a new Workflow with the remained activities and reusable results. (The problem is that on every parallel branch we must have a reusable result). The actions and their sequencing are the following:

- For every activity the designer must decide if it has remanent result and the way that allow to identify and storing it; also for the other activities it will specify their kind the result is volatile or the effect of the activity or the overall system, in the sense that it must be

discarded (example: updating data bases with the cancellation of the activity effects).

- The Workflow has a description of all its activities the above features.
- During the execution of a workflow instance it will record in a data base, the activities (their results if there is in the case of the remanence) as soon as these are executed.
- When a workflow fails then an analysis is done and the effects of activities on the system state are done; the last activities with remanent results will be retained. Here a mechanism for finding on every branch these activities is required.
- After the previous step when the workflow instance must be replied a new variant of the workflow is constructed: the input, a transition, and the locations of activities with remanent results.
- The new workflow instance can be start.

The figure sketches out our proposal.

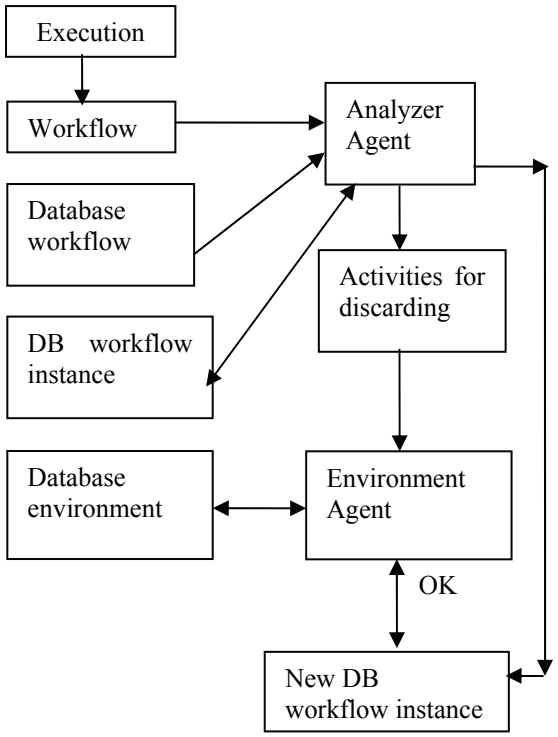


Figure 6: The flow of actions.

The workflow must be described with the above requirements and the system agents act as follows. In the case of a workflow instance failure the Analyzer Agent, make the analysis from that:

- all the activities on the every parallel branch are specified and their results are available;
- for all the activities that affect the overall system state prepares their cancellation actions;
- redesigns the new workflow instance.

The Environmental Agent executes the cancellation activities. The Effective Agent negotiates the conditions for the execution of the new instance of the workflow and launches it in execution. Decision for modeling is based on process complexity, failure of activities that can fail only the activity and "suspend" the process or fail all the process. The problem is to identify the reply of the process and its implications. Done to the process complexity the solution can be:

- restart the process from the beginning or
- reply the process from one of the previous "valid" states before its failure under some conditions.

Needs for replies in a failure case are:

- every activity must have specified its type concerning its result (reusable, affect the environment, insignificant);
- during the execution every activity recalls somewhere (in a data base) its result and this result can be used depending of activity type in the future (if the activity is of reusable type).

7. CONCLUSIONS AND FUTURE WORKS

The considerations form the paper will serve for designing the appropriate models:

- design a resource manager that is able to interact with the system, ERP, and workflow engine" in order that the resource requirements during of the execution of a workflow instance have a real support;
- Re-engineer a failed workflow instance such as the system and the workflow instance are able to (re)start their activities from a coherent and also consistent state.

For that, the communication between the Resource Manager and the real system (or the ERP) and workflow engines will be done using appropriate web services or in the simplest cases messages. The workflows and workflow instances will be described in appropriate and extended data bases so as was stated in (Cicortas 2006).

The failure of a workflow instance must be managed by a multi-agent system that:

- will discard the effects of the failure in the data base (so as in transactional data bases);
- will rebuild the rest of the workflow instance (that must be executed);
- will give to the workflow engine the new (rebuild) workflow instance.

REFERENCES

Adams, M., ter Hofstede, A.H.M., Edmond, D., van der Aalst, 2006. *W.M.P., Implementing Dynamic Flexibility in Workflows using Worklets*, BPM Center Report BPM-06-06, BPMcenter.org, 2006.

Cardoso, J., Bostrom, R.P., Sheth, A., 2004. *Workflow Management Systems and ERP Systems: Differences, Commonalities, and Applications, Information Technology and Management* 5, Kluwer Academic Publishers, pp. 319–338, 2004.

Cicortas, Al., Iordan, V., Fortis, Al., Fortis, F., 2006. Reengineering the Failed Workflows, *Annals of the Tiberiu Popoviciu Seminar, Supplement international Workshop in Collaborative Systems*, Vol. 4, ISSN 1584-4536, pp.57-66.

Dialani, V, Miles, L., De Roure, D., Luck, M., 2002. Transparent Fault Tolerance for Web Services based Architectures, *Eighth International 12 Europar Conference (EURO-PAR'02), Lecture Notes in Computer Science,* Padeborn, Germany, August 2002, Springer-Verlag, 2002.

Eder, J., Liebhart, W., 1996. Workflow recovery, *Proceedings of the First IFCIS International Conference on Cooperative Information Systems* (CoopIS'96) 1996.

Eriksson, B., Baharak, G. Fard , Grubesic, B.,2002. *Frameworks, an Introduction*, 2002.

Fayad, Mohamed E., Schmidt, Douglas C., Johnson, Ralph E., 1999. *Building application frameworks: object-oriented foundations of framework design*., John Wiley & Sons, Inc., 1999.

Iordan, V., Cicortas, Al., 2007. *Multi-Agent Systems Used for Managing Competences*, The Proceedings of the 8th International Conference on Informatics in Economy, ASE Publishing House, 2007, Informatics in Knowledge Society, ISBN 978-973-594-921-1, Bucharest, pp. 633-638, 2007

Kearney, R.J., 2005. *Enhancing and Extending ERP Performance with an Automated Workflow System*, Workflow Handbook, Lighthouse Point, Florida, ed. Layna Fisher, pp.92-102, 2005.

Lerner, B. S., Ninan, A. G., Osterweil, L. J., Podorozhny, R. M., 2000. *Modeling and Managing Resource Utilization for Process*, Workflow and Activity Coordination, University of Massachusetts, TR: UM-CS-2000-058, 2000.

Luo, Z, 2000. Checkpointing for workflow recovery. *ACM Southeast Regional Conference,* pp. 79-80, 2000.

Sharp, A., McDermott, P., 2001. *Workflow Modeling - Tools for Process improvement and Application development*, Artech House.

van der Aalst, W., and van Hee, K., 2002. *Workflow Management, Models, Methods and Systems,* The MIT Press Cambridge, Massachusetts London, England.

WfMC 1994. *The Workflow Reference Model, Version* 1.1, Standard TC00-1003, Workflow Management Coalition, 1994.

WfMC 1996. *Workflow Management Coalition*, Interface 1: Process Definition Interchange. Draft 5.0, Issued on February 03, 1996, Workflow Management Coalition, 1996.

WfMC 1999. Terminology & Glossary, Standard TC-1011, Issue 3.0, *Workflow Management Coalition*, 1999.

WfMC 2001. *Workflow Handbook 2001*. Workflow Management Coalition, Future Strategies Inc., Lighthouse Point, Florida, USA, 2001.

zur Muehlen, M. , Allen, R., 2000. Embedded vs. autonomous workflow – putting paradigms into perspective, in: Excellence in Practice, Vol. IV, *Innovation and Excellence in Workflow and Knowledge Management*, ed. L. Fischer (Future Strategies, Lighthouse Point, FL, 2000) pp. 49–58.

zur Muehlen, M., 2004. Workflow-based Process Controlling: Foundation, Design and Application of Workflow-driven Process Information, *Advances in Information Systems and Management Science*, Logos Verlag Berlin, 2004.

AUTHORS BIOGRAPHY

Victoria Iordan is Associate Professor Ph.D. at Computer Science Department of Mathematics and Computer Science Faculty, West University of Timisoara. Fields of scientific interest: modeling, simulation, complex systems design, workflow modeling, numerical methods.

Alexandru Cicortas. Actually is Professor Ph.D. at Computer Science Department of Mathematics and Computer Science Faculty, West University of Timisoara. Main of his research and papers concerns: the Discrete Event Systems (DEvS) modeling in distributed architectures; Petri Nets and component modeling are another domain of interest; complex systems design; workflow modeling (he is member of the European VISP project).