Supporting Direct User Interventions in Exception Handling in Workflow Management Systems

Hernâni Raul Mourão
Escola Superior de Ciências Empresariais do Instituto Politécnico de Setúbal /
LASIGE - Laboratório de Sistemas de Informação de Grande Escala,
Campus do IPS – Estefanilha
2914 – 503 Setúbal
hmourao@esce.ips.pt

Pedro Antunes
Faculdade de Ciências da Universidade de Lisboa,
LASIGE - Laboratório de Sistemas de Informação de Grande Escala,
Campo Grande, Edifício C8, Piso 1
1749-016 Lisboa
paa@di.fc.ul.pt

Abstract. We developed a framework to handle exceptions in WfMS. Specially, unexpected exceptions, which are situations not predicted during the design phase, and require human involvement. A good characterization of the exception is needed to help the user in the identification of the solution(s) from an available tool kit: redesigning the flow, ad hoc executing the affected tasks, and manipulating engine status. The proposed characterization results from integrating operational, tactical and strategic perspectives over unexpected exceptions. An open source platform was selected to establish a test base on which the framework will be tested. The framework will be implemented in one company and data from another company will be used for simulation.

Resumo. Desenvolvemos uma plataforma para tratar excepções em sistemas de gestão de fluxos de trabalho. Em particular para as excepções não esperadas, que correspondem a situações não previstas durante a fase de modelação e necessitam do envolvimento do operador. Uma boa caracterização da excepção é importante para auxiliar o operador na escolha da solução(ões) num conjunto de ferramentas disponibilizadas: alterar a definição do fluxo de trabalho; executar de forma ad hoc as tarefas afectadas; ou manipular o estado do motor. A caracterização proposta resulta da integração das perspectivas operacional, táctica e estratégica sobre excepções não esperadas. Foi seleccionado um projecto "*open source*" para definir um ambiente onde a plataforma possa ser testada. O projecto será implementado numa empresa e dados recolhidos de uma outra empresa serão utilizados para simulação.

1 Introduction

Workflow exceptions are situations not predicted in the workflow model or when there exists a deviation from the model and the real world [Luo 2001]. Workflow exceptions have been accounted for almost half of the total working time in an office [Saastamoinen 1995].

Up to now workflow exceptions have mostly been addressed in a systems perspective, even though a consensus seems to arise on the necessity of involving the user in the recovery of a specific kind of exceptions. Therefore we developed a framework aimed to fulfill this identified gap. The framework is built around the idea that human intervention must be supported with quality information about the problematic situation, status of the workflow engine and possible recovery solutions.

Considering the work in progress, this paper currently offers two contributions to exception handling in WfMS: (1) an integrated perspective over exceptions and exception handling, considering three different organizational levels (strategic, tactic and operational) affected by exceptions and respective human roles in recovery (design changes, ad hoc execution, and

Mourão, H. and P. Antunes (2005) "Supporting Direct User Interventions in Exception Handling in Workflow Management Systems." Sistemas de Informação, 17, pp. 39-51. ISSN: 0872-7031.

engine status manipulation); and (2) a set of WfMS components necessary to help and support these roles: event handler, situation awareness, problem characterization and recovery toolkit. These components are currently being validated on an open source platform.

The paper is organized as follows. Section 2 reviews the literature and establishes the grounds for the proposed framework. Section 3 presents the integrated perspective, where different exception handling approaches are mapped to the organizational levels where they occur and are handled. In section 4 we describe the components that constitute our solution. In section 5 we describe the project current status. In section 6 we identify the main current contributions and discuss the expected results.

2 Literature Review

[Eder and Liebhart 1995] characterize failures and exceptions in a single dimension, encompassing two types of failures and two types of exceptions:

- Basic failures – Associated with failures on the systems supporting the WfMS (e.g., operating system, database management system and network failures);
- Application failures – Failures on the applications invoked to execute tasks (e.g., unexpected data input);
- Expected exceptions – Events that can be predicted during the modelling phase but do not correspond to the "normal" behaviour of the process. These exceptions can happen frequently and can cause a considerable amount of work to process (e.g., a flight cannot be booked because it is already booked up);
- Unexpected exceptions – When the semantics of the process is not accurately modelled by the system (e.g., changes in rules, a structural change in the organizational environment, or a change in the order processing of a very important client). This type of exception cannot be predicted at the modelling stage and may require human intervention or even force the system to stop.

[Eder and Liebhart 1996] suggests an escalating concept to transform the failures that cannot be resolved in the level where they occur into exceptions. This way, the classification can be reduced to exceptions.

[Chiu 2000] based on research work from [Eder and Liebhart 1995], combines the above view with another orthogonal characteristic described as "exception source." The exception source can either be internal or external to the workflow. External sources are consequences of failures in components that participate in the WfMS such as operating systems, database management systems, software applications, machines and equipment, or operations in external organizations. Internal sources are directly related to workflow management issues, e.g., being unable to find a component to execute a task, or a missing a deadline.

A similar classification is adopted by [Worah and Sheth 1997] but without any distinction between expected and unexpected exceptions.

Next, we will analyse the expected and unexpected exceptions in more detail, primarily basing our comments on results produced by the WfMS research community. Later, we will present a broader perspective, oriented towards the organizational semantics. Finally, to complete this review, we describe the exception handling strategies found in the literature.

2.1 Expected and Unexpected Exceptions

Expected exceptions are cases that can be predicted during the modelling stage but do not correspond to the "normal" process behaviour. Some mechanisms should be implemented to handle these situations because they can occur frequently [Eder and Liebhart 1995]. The example used by [Casati, et al. 1999] to explain expected exceptions is a car rental company. When a customer phones to report an accident, the company has to re-schedule all future rentals for that specific car and make the necessary arrangements to repair the damaged car. The "normal" behaviour of the process should have been the returning of the car to the company, as planned, while the accident corresponds to a deviation or an "occasional" behaviour: an expected exception.

[Casati 1998] identifies four classes of expected exceptions, according to the events that generate them:

- Workflow exceptions – On starting or finishing a task;
- Data exceptions – When workflow relevant data is changed;
- Temporal exceptions – On the occurrence of a given timestamp (e.g., car not returned on time, check every week on car repair situation);
- External exceptions – Activated by external events (e.g., the above example of a customer reporting an accident).

Also important is to identify whether the exception is synchronous or asynchronous to the evolution of the process. The only synchronous class of exceptions is the workflow type since it happens after a change in the process state. Asynchronous exceptions are difficult to model with flow diagrams because it is not possible to insert them in the process evolution.

Unexpected exceptions result from inconsistencies between process modelling in the workflow and the actual execution [Casati 1998]; and result from incomplete or design errors, improvements or changes in the business manoeuvre or quality and customer satisfaction issues not known during the modelling stage. This type of exceptions is frequent in highly complex or dynamic environments. Usually, unexpected exceptions force the process to change to a halt state and require human intervention [Heinl 1998].

In situations where this kind of exceptions occurs frequently, one should consider redesigning the workflow model, if it is out of date, or adopting different technologies based on collaborative work or adaptive workflow systems [Casati, et al. 1999].

2.2 Organizational Perspective

[Saastamoinen 1995] proposes a taxonomy based on the organizational semantics associated to exceptions. This work defines a set of base concepts necessary to construct a consistent conceptual framework and fundament the characterization of organizational exceptions. Next we will summarize some of the main concepts defined and later the taxonomy is described.

Roughly we could say that any situation for which the organization has no rule is an exception. This is the line of thought presented in this work. First, the concepts of rule and event are defined and than the rules to handle different types of events which are the base for the organizational procedures. Finally, a concrete definition of exceptions is presented.

Event is defined as a peace of work to be handled by an office that is caused by a detected phenomenon or a state of the system. An event type is a specification of the common features found in certain events. The concept of rule plays an important role in this work as the bases for exception definition. The author defines rule as a formal way of specifying a recommen-

dation, a directive or a strategy, expressed as "IF premise THEN action" or "IF condition THEN action" and then, event handling rule, as an orderly set of rules that precisely and accurately guides an actor in handling certain types of events. An office procedure has a broader scope and is defined as an orderly set of event handling rules aimed at reaching a specified goal in an office by directing an entire handling of an event.

Once the association between rules and event handling rules is established the concepts of normal event, main line, variation, main line event, and variation event constitute the last step towards exception definition. A normal event is an event with the necessary rules for identifying and for handling. A main line is an office procedure for the most predictable events of a certain type and a variation is work that is added to the main line. Note that a variation is an office procedure for less predictable but still known events of a certain type. A main line event is a normal event that can be handled by the main line while a variation event is an event not handled by the main line but handled by another office procedure.

The environment for exception definition is now created. An exceptional event (exception) is an event that neither the main line or the variation procedure can handle.

As discussed by the author, the definition of rule presented is narrower than the variety of rules that exists in an organization, e.g., good business practice, precepts, regulations, conventions, principles, guiding standards, rules of thumb, and even maxims. These kinds of rules are not precise enough to establish a consistent ground to serve as the basis for a framework even though they represent all the knowledge of the organization. Therefore, [Saastamoinen 1995] uses the rule-based knowledge presentation technique for the definition of rules. In a way, procedures and functions are a kind of rules as they guide the actions of a computer.

[Saastamoinen 1995] developed a taxonomy from empirical studies. A special attention was made on the social and financial impacts of exceptions. Six different criteria are used to classify exceptions:

- Exceptionality – Difference between the exceptional and "normal" event;
- Handling delay – Time elapsed between the exception identification and handling;
- Amount of work – Extra work required to handle the exception when compared to the normal event;
- Organizational influence – Number of people involved in the exception;
- Cause – A measure of the importance of the reason for the exception;
- Rule impact – Number of changes in the organization's rules due to the exception.

Three classes of exceptions were identified according to exceptionality: established exceptions, otherwise exceptions, and true exceptions. Established exceptions occur when rules exist in the organization to handle an event but the right ones cannot be found. Otherwise exceptions occur when the organization has rules to handle the normal event but do not apply completely to the case. Finally, true exceptions occur when the organization has no rules.

According to the organizational influence criteria, exceptions can also be classified at employee, group and organizational level. Employee exceptions are situations that affect only the work of one person, group exceptions, affect a group of people working within the same process, in the same kind of job or in the same process, and organization exceptions, affect the work of persons in more than one department or project in the organization.

A relationship between this approach and the WFMC definitions is established to provide integration with concepts widely used. According to the WFMC, an organizational model represents organizational entities and their relationships; it may also incorporate a variety of attributes associated with the entities, such as skills and roles. The organizational model can

be represented in the system and should be the basis for the characterization of the organizational influence area.

The WFMC definition of a business process is a set of one or more linked procedures, which collectively realize a business objective or policy goal, normally within the context of an organizational structure defining functional roles and relationships. A workflow is the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules. The definition of office procedures in [Saastamoinen 1995] and the associated definition of exception seem to be coherent with this approach.

On another work oriented towards organizational semantics, [Strong and Miller 1995] defines five alternative perspectives on the nature of exceptions: random event; error; political system; total quality management; and human-computer system. It is important to realize that this classification mixes causes and solutions. We will discuss solutions in more detail in the next session. Note also, that apart from the random event, [Saastamoinen 1995] integrates the other four criteria so we will only consider his work in the future. [Strong and Miller 1995] mentions random event only for completeness, considering that it cannot be predicted nor can efficient procedures be developed to handle them. Since

2.3 Exception Handling

As it is recognized by most literature, predicting any possible cause of failure or exception during design is very difficult or even impossible and makes the system very complex and hard to manage [Eder and Liebhart 1998; Dayal, et al. 1990; Casati 1998; Klein and Dellarocas 2000]. Therefore, being prepared to deal with failures and exceptions at execution time is a critical factor for WfMS success.

A significant effort has been invested so far to deal with failures and exceptions in a similar way of traditional DBMS, i.e., the transaction processing mechanism. However, task semantics in WfMS far exceeds the transaction models. E.g., if a call to a customer fails because s/he is not reachable, nothing else needs to be done at the moment; maybe trying later will be enough. Organizations are complex environments and the traditional approach based only over data integrity and consistency does not constitute a complete ground to base a framework able to resolve these problems [Worah and Sheth 1997].

Nevertheless, in a primitive approach, we could rely on the system that supports the WfMS. In fact, most of the commercially available DBMS on the market implement the necessary mechanisms to react in case of failure, returning the system to a coherent state and enabling forward execution [Casati 1998].

On the other hand, a typical task can span over a long period of time, and in heterogeneous, autonomous and distributed environments there are tasks implemented by equipment/tools that do not run in transactional environments. This is a more complex environment where typical DBMS solutions are not adequate.

The error handling semantics of traditional transaction processing systems is also too rigid for workflow [Worah and Sheth 1997; Luo 2001]. The handling of system errors based on transactional approaches offer, in general, extreme and expensive solutions in terms of lost work and should be avoided. System errors should then be treated as expected exceptions [Casati 1998].

Some suggestions to overcome these problems consider relaxing the ACID properties and incorporating compensation mechanisms for backward recovery and forward execution, incorporating the flexibility required for workflow systems [Luo 2001]. Several researchers are

working in these issues [Eder and Liebhart 1998; Dayal, et al. 1991; Georgakopoulos, et al. 1996; Krishnakumar and Sheth 1995]. A good survey of this area, which became known as Extended Transaction Models (ETM), is presented by [Worah and Sheth 1997].

Some other studies have proposed the adoption of dynamic and adaptive workflow systems to react to exceptions during workflow execution. The operator, on the presence of an exception, could change the system by either creating a new path for the exceptional process or change all the processes running on the system to the newly created path. Some studies have been carried out to keep the system's consistency and correctness during/after the change, e.g., [Ellis, et al. 1995; Aalst 1999; Reichert and Dadam 1998; Myers and Berry 1999].

[Dayal, et al. 1990; Dayal, et al. 1991] recognized the fixed control sequence and the rigid compensation policy of the ETM approach and developed the Event Condition Action (ECA) rules to decouple the detection and handling of exceptions from the system itself and to increase flexibility. [Casati, et al. 1999] describes a system to deal with expected exceptions based on ECA rules. The language Chrimera-Exc is used to specify exceptions and augment the WfMS characteristics to automatically detect and handle expected exceptions.

[Luo, et al. 2002] describes a system using a Case Base Reasoning (CBR) scheme to derive patterns for exception handling. A measure of similarity is computed to identify the nearest pattern in the knowledge base with the current exception and determine the appropriate action to handle the specific case. The system implements a user intervention mode to allow handling totally new cases or defining a new action for a specific case.

[Klein and Dellarocas 2000] developed a knowledge-based approach that incorporates "sentinels" in the system model to automatically identify exceptions. During execution time, when an exception is detected, it is matched to a handbook's of knowledge and an appropriate action is proposed to the operator. The focus of this approach is mainly on coordination processes rather than core processes.

Despite all these efforts to automatically handle exceptions, the majority of authors involved in research recognize the limits of the proposed solutions. They either recognize the importance of interrupting the process and integrating some manual mechanism and then send the process back to workflow control [Eder and Liebhart 1998; Klein and Dellarocas 2000], or explicitly state that in some situations the role of humans is crucial to collect process specific data that is not available to the workflow system.

It currently seems impossible to include task specific semantics within a generalized recovery framework because task behaviour is orthogonal to workflow process [Worah and Sheth 1997]. [Casati 1998] clearly states that unexpected exceptions should be handled by humans, since they have not been predicted during systems modelling. [Luo, et al. 2002] also recognizes the necessity of human involvement when an exception handler cannot be automatically obtained.

According to [Heinl 1998], the WfMS has only the information about the process that was defined during the modelling phase, and has no possible way of identifying the case when the real business process differs from the model. The author even states that only a human with an overall view can identify the discrepancies and find the way to handle the case.

To complete this review, it is important to mention two other approaches introducing a broader perspective over workflow exceptions. [Guimarães, et al. 1997] proposed an integrated architecture of formal coordinated processes with informal cooperative processes. When the system identifies that it is not able to proceed with process execution, it gathers all the information concerning the specific situation and generates a flow interrupt. The interrupt triggers a tool able to select the best suitable cooperative technique to handle the specific case. [Saastamoinen 1995] presents an approach focussed on organizational semantics. Petri

nets, outside the scope of the WfMS, define the reactions to the various types of exceptions, and should be interpreted as a global organizational reaction to exceptions.

3 Integrated perspective

From the above discussion we can assume that accounting for all possible exceptions requires an integrated approach, where different levels of the organizational system affected by exceptions must be involved. Moreover, the different techniques proposed in the literature should be considered if a complete framework is to be developed. These techniques have a specific scope of application with a correspondence in the level of the organizational system where they should be taken into account. We propose the framework shown in Figure 1.

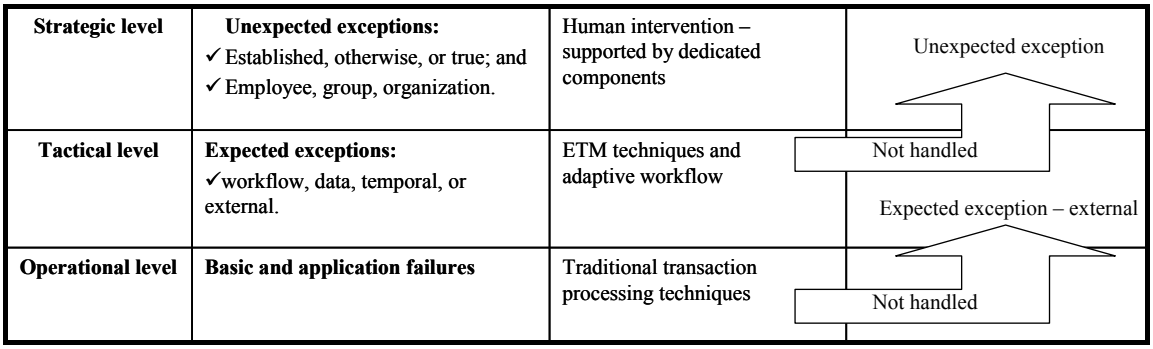

Fig. 1 – Different levels and corresponding exceptions

Depending on the cause and impact of the abnormal situation, a suitable recovery mechanism at the most adequate level should be invoked. These relationships are represented in the second and third columns of figure 1. On the other end, a propagation mechanism is foreseen to enable the exception/failure to be handled in a different level. As described below, the propagation mechanism can be automatically triggered by the system or manually, by an operator, which identifies the necessity to change the level where the exception must be handled. The arrows in figure 1 represent the propagation mechanism.

Another key issue in the integrated perspective is to associate all exceptions to a responsible person. This should prevent the exceptions to get "lost" in the system with no one being responsible for them. The propagation mechanism can also change the responsible person associated with the exception.

Next, we will describe the levels of the organizational system where the different techniques found in the literature to handle exceptions should be invoked. The propagation mechanism is triggered whenever the corresponding layer cannot handle the exception. Then we describe our approach to human involvement in dealing with situations not solved by the previous techniques. At the end of this section we compare some methodologies found in the literature to involve human with our integrated perspective.

The operational level provides an environment for handling basic and application failures only, where the traditional transaction processing techniques can be sufficient to bring the system back to a coherent state and continue execution without human intervention. Whenever these techniques are not able to solve the problem, the event is propagated to the tactical level and the failure is converted into an external exception.

At the tactical level, the WfMS may automatically handle exceptions in many different ways. For instance, using ETM techniques [Eder and Liebhart 1998]. The extensive work done on adaptive workflow, which falls in this level, should increase the system flexibility and augment its adaptation to real world situations in handling, mainly, the expected exceptions. At this level, human contribution is possible but limited to producing the information necessary to have the system applying the exception handling techniques (e.g. compensation, retry, ignore, etc). The works developed by [Casati, et al. 1999; Luo, et al. 2002; Klein and Dellarocas 2000] should also be considered at this level as they contribute to expected exceptions handling.

Eventually, if none of the techniques implemented at this level are able to handle the event, it should be propagated to the strategic level, where the attention of a human operator to an unexpected exception is raised.

When the system has no possible way of identifying the abnormal situation the exception is directly reported by an operator and classified as unexpected exception.

Even though several authors present some ideas on how to incorporate human involvement in exception handling, we believe that the problem has not yet been completely addressed. [Eder and Liebhart 1998] describes the idea of a system that incorporates human intervention in two possible modes: (1) ad hoc extension, where the user can suspend the execution of a task and choose alternative paths or change the execution model; and (2) ad hoc refinement, where the user interrupts the task execution to execute one or more activities and later on proceeds with the interrupted task. Even though the authors integrate this model in their framework, we believe that it should be completed in terms of the tools/methodologies available to help and support the user manipulating the environment in which s/he operates.

In [Luo, et al. 2002], a user-interface is provided to assist human intervention. The exceptions tagged as requiring human intervention come to the attention of an expert that can choose one candidate solution within the system proposals, or rewrite a new solution. In order to react to unexpected exceptions, the system is notified by an external signal and generates an internal exception event. However, it is not clear how this mechanism is implemented neither how the system can deal with any particular type of unexpected exception. Moreover, it is not foreseen any assistance mechanism to provide a general view of the situation nor any support tool to change the model.

Note that although human participation is considered at both the tactical and strategic levels, the strategic level envisages a more dramatic intervention in the WfMS. This will be discussed in more detail in the next section.

4 Our Approach to a Solution

We now propose a system that supports human intervention to handle unexpected exceptions. The interaction between the WfMS and the user is supported by the following components:

Event handler. This component is responsible for launching the recovery process whenever an exception is detected and human intervention is required. It interacts with the user in order to show any upstream and downstream exception propagations related with the exception (e.g., a basic failure previously propagated as an external exception, or a group exception subsequently propagated as an organizational exception).

Situation awareness. This component is responsible for gathering and displaying to the user generic data about the workflow and engine status associated with the exception. In par-

ticular, it lists all tasks defined by the workflow, their status and other details, like task goals. This component also allows the user to identify which tasks are affected by the exception.

Problem characterization. This component uses the criteria defined by [Saastamoinen 1995] to obtain from the user the information necessary to classify an exception as a true, established or otherwise exception. It also allows the user characterizing the organizational influence of the exception (employee, group or organizational level).

Recovery Toolkit. Based on the situation awareness and problem characterization, this component offers a collection of pre-defined actions, which can be combined by the user to manipulate the process design, process execution and engine status. The following collection of pre-defined actions is actually considered:

- Engine status – Start/terminate or suspend/continue several tasks; propagate the exception to other users; manipulate any other information that is necessary, such as process relevant data or workflow participants;
- Process execution – Support ad-hoc extensions and ad-hoc refinements to the process instances affected by the exception; apply design changes according to specification;
- Design – Change one or several process definitions, either affected by this exception or not; define how and when the modifications are applied (one/all instances, immediately or after completion).

Note that the user can execute, in any order, multiple actions in each one of the above three categories. The heuristic used for proposing the best solutions according to the characterization of exceptions is the following: otherwise exceptions are commonly handled by ad-hoc extensions or refinements applied to the affected process instances; established exceptions can be handled by instantiating already available process definitions; and true exceptions require design changes and possibly cascading the exception to other users. If, in the first two cases, the organizational influence of the exception is all the organization, the approach must be coordinated with other areas.

5 Project Status

We selected OpenSymphony ["The OpenSymphony project" 2001] to implement this project. OpenSymphony is an open-source platform based on J2EE technology.

The project consists of Enterprise Java Beans (EJB) for functional components and web-tier components for front end – all components are platform, database and application server independent.

Figure 2 represents the 5 component layers of the OpenSymphony architecture. The Foundation layer comprises a set of core libraries used by the rest of the suite. The Business components implement the business functions, in special the workflow. Our Integrated Perspective is also implemented in this component. The Interaction layer allows web-based applications to interact with system components, whereas the Presentation aid in the final presentation on the web. Finally, the application layer is the ready built solutions for real needs.

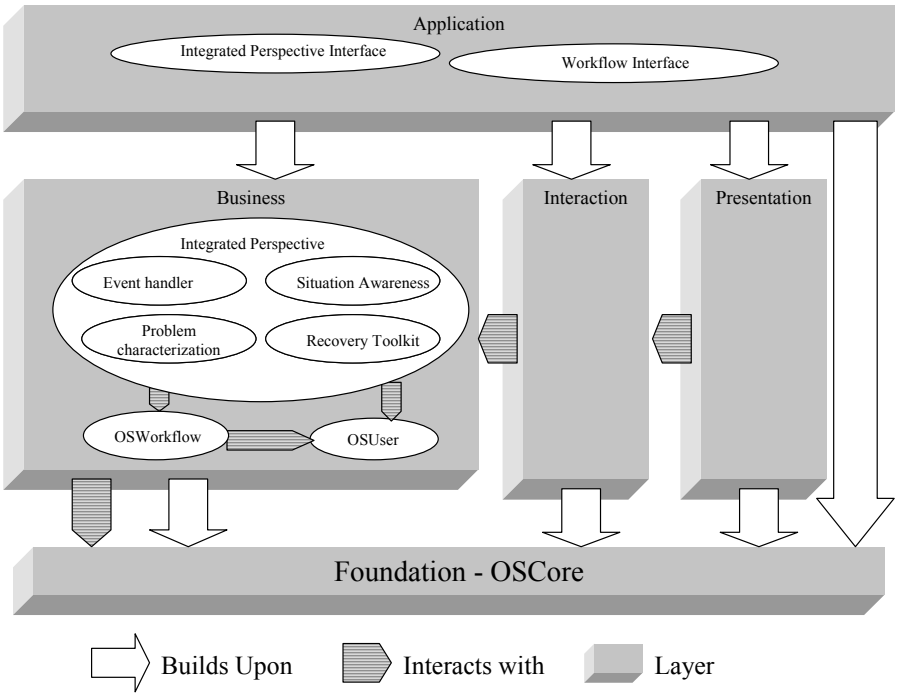

Fig. 2 – OpenSymphony components architecture

The workflow specification uses a XML file rather than a graphical user interface, which gives more flexibility to cope with changes in the process definitions. The workflow engine is based on the concept of a finite state machine where each state is represented by a combination of StepID and status. A transition cannot be fired without an action occurring first. The engine supports Java-based functions (class loader, classes retrieved via JNDI, local EJB, and remote EJB), BeanShell scripts, and Bean Scripting Framework (BSF) scripts. Another type of functions, the Trigger Functions, are triggered by outside sources and run under the system context.

These base components of the framework, together with its capability to coexist with other frameworks oriented our decision to choose this platform.

Figure 3 shows the user interface implementing our exception handling approach. The user interface highlights the four main components necessary to support exception handling.

The user can analyse the list of exception propagations in the Event Handler area, if any. The interface allows analysing upstream and downstream exceptions. Note that downstream exceptions only exist if the user decides to propagate the current exception.

In the Situation Awareness area, the user sees a list of all the tasks running on the system (left side). Selecting the affected ones and pressing the right arrow button, they are transferred to the right side. This way it is possible to identify all the currently running tasks that are affected by the exception. Whenever necessary, the specific task details can be viewed/edited.

The Problem Characterization area enables the selection of the exception characteristics according to our approach. At this stage only the exceptionality and organizational influence are considered. The other four characteristics (handling delay, amount of work, cause, and rule impact) can only be defined at the end of the exception handling, for historical records. In the organizational influence zone there is an area that changes according the selection, allowing to select an employee, group leader or organizational manager, respectively. This way someone is always associated with an exception. There is also the possibility of sending emails to everyone involved.

In the Recovery Toolkit area the user can decide the recovery action(s) to implement on this specific case. As noted before, any combination of the above pre-defined actions can be used for the particular case, although the system suggests some pre-defined actions. On the type, action, and parameters columns the user selects one option from a list.

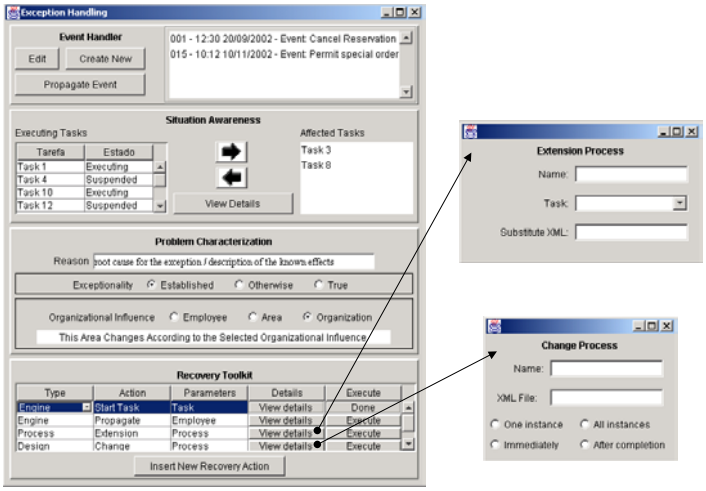

Fig. 3 – Interface for exception handling

A design company was selected to test the prototype in real world situations. The processes were studied and are being modeled. The framework is going to be implemented in the company and the exceptional situations together with the company reactions will be identified and registered. This test will enable the adjustment of the framework to operator response and test it's applicability.

Another company is being selected to collect a set of situations that will be used in a simulation study. The data from this study will complement the data from the implementation augmenting the adjustment of the framework concrete situations.

6 Expected Results

It is expected that the implementation and the simulation will raise issues concerning the ability of the framework in dealing with adjustment of the modeled processes to changing environments and to situations not predicted in the modeling stage. The robustness issues associated with the run time changes to the processes design will be a major concern and the support to the user during this changes will be a critical factor to maintain the coherence of the system.

The taxonomy used to classify exceptions and the level in the organizational system that should deal with the abnormal situation should also be a subject of deeper investigation as they have a critical role in the choice of the best handling mechanism.

With this work we also expect to improve the OpenSymphony platform, allowing the workflow components to cope with unexpected exceptions.

References

Aalst, W. v. d., "Generic workflow models: how to handle dynamic change and capture management information," Proc. of IFCIS, International Conference on Cooperative Information Systems, CoopIS '99, IEEE International, 1999, 115 -126.

Casati, F., Models, Semantics, and Formal Methods for the Design of Workflows and Their Exceptions, PhD Thesis, Politecnico di Milano, 1998.

Casati, F., S. Ceri, S. Paraboschi, and G. Pozzi, "Specification and Implementation of Exceptions in Workflow Management Systems," ACM Transactions on Database Systems, 24, 3 (1999), 405-451.

Chiu, D. K., Exception Handling in an Object-Oriented Workflow Management System, PhD Thesis, Hong Kong University of Science and Technology, 2000.

Dayal, U., M. Hsu, and R. Ladin, "Organizing Long-Running Activities with Triggers and Transactions," Int. Conf. on Management of Data (SIGMOD'90), Atlantic City, NJ, USA, 1990.

Dayal, U., M. Hsu, and R. Ladin, "A Transactional Model for Long-Running Activities," 17th Int. Conf. on Very Large Data Bases (VLDB'91), Barcelona, Spain, 1991.

Eder, J., and W. Liebhart, "The Workflow Activity Model WAMO," Int. Conf. on Cooperative Information Systems, Vienna, Austria, 1995.

Eder, J., and W. Liebhart, "Workflow Recovery," 1st IFCIS Intl. Conf. on Cooperative Information Systems (CoopIS'96). Brussels, Belgium: IEEE International, 1996, 124 - 134.

Eder, J., and W. Liebhart, "Contributions to Exception Handler in Workflow Management," Int. Conf. on Extended Database Technology (EDBT'98), Workshop on Workflow Management Systems, Valencia, Spain, 1998.

Ellis, C., K. Keddara, and G. Rozenberg, "Dynamic change within workflow systems," Proc. of conf. on Organizational computing systems. Milpitas, CA, USA: ACM Press, 1995, 10-21.

Georgakopoulos, D., M. Hornick, and F. Manola, "Customizing Transaction Models and Mechanisms in a Programmable Environment Supporting Reliable Workflow Automation," IEEE Transactions on Knowledge and Data Engineering, 8, 4 (1996), 630-649.

Guimarães, N., P. Antunes, and A. P. Pereira, "The Integration of Workflow Systems and Collaboration Tools," in A. K. Dogaç, Leonid Ozsu (Eds.), Advances in Workflow Management Systems and Interoperability, Instambul, 1997.

Heinl, P., "Exceptions During Workflow Execution," Int. Conf. on Extended Database Technology (EDBT'98), Workshop on Workflow Management Systems, Valencia, Spain, 1998.

Klein, M., and C. Dellarocas, "A Knowledge-Based Approach to Handling Exceptions in Workflow Systems," Computer Supported Cooperative Work, 9, 3 (2000), 399-412.

Krishnakumar, N., and A. P. Sheth, "Managing Heterogeneous Multi-system Tasks to Support Enterprise-wide Operations," Journal on Distributed and Parallel Database Systems, 3, 2 (1995).

Luo, Z., Knowledge Sharing, Coordinated Exception Handling, and Intelligent Problem Solving for Cross-Organizational Business Processes, PhD Thesis, Dep. of Computer Sciences, University of Georgia, 2001.

Luo, Z., A. P. Sheth, K. J. Kochut, and I. B. Arpinar, Exception Handling for Conflict Resolution in Cross-Organizational Workflows, LSDIS Lab, Computer Science, Un. of Georgia, 2002.

Myers, K. L., and P. M. Berry, "At the Boundary of Workflow and AI," proc. of the AAAI-99 Workshop on Agent-Based Systems in The Business Context held as part of AAAI-99, 1999.

The OpenSymphony project. (2001, 20-04-2001), Http://www.opensymphony.com.

Reichert, M., and P. Dadam, "ADEPTflex - Supporting Dynamic Changes of Workflows Without Loosing Control," Journal of Intelligent Information Systems, 10, 2 (1998), 93-129.

Saastamoinen, H., On the Handling of Exceptions in Information Systems, PhD Thesis, University of Jyväskylä, 1995.

Strong, D. M., and S. M. Miller, "Exceptions and Exception Handling in Computerized Information Systems," ACM Transactions on Information Systems, 13, 2 (1995).

Worah, D., and A. P. Sheth, "Transactions in Transactional Workflows," in S. K. Jajodia, Larry (Eds.), Advanced Transaction Models and Architectures, Kluwer Academic Publishers, 1997.