Proceedings of the 5th WSEAS Int. Conf. on SIMULATION, MODELING AND OPTIMIZATION, Corfu, Greece, August 17-19, 2005 (pp627-632)

Business Process Unpredictable Exception Handling Using Workflow Based Method

DOVILĖ VOJEVODINA, GENADIJUS KULVIETIS
Information Technologies Department
Vilnius Gediminas Technical University
Saulėtekio 11, Vilnius, LT-10223
LITHUANIA

Abstract: - As business competition is getting stronger companies are seeking good relationships with their clients through the Internet technologies. The type of this relationship (B2C or B2B) usually depends on the business strategy of this company. Using e-business technologies companies expect quick and qualitative contact with clients. Still, the quality of this solution depends on many aspects, but the main is application performance. The medium for these applications is Internet and it is not always reliable through all the engagement process. As these processes are often modelled using workflow, it often faces the one major problem – lack of real-world deviation handling capabilities. So exception handling is becoming one of the most important aspects of these processes. Of these exceptions, the most dangerous are run-time exceptions and deviations. They are difficult to catch, their impact to the process can be detrimental to achieving the process' goals, and, of course they are annoying to users. The purpose of this paper is to discuss an approach for a universal exception handling mechanism, which would help achieve business process transparency and exception handling flexibility.

Key-Words: - Business process, Workflow, Exceptions, Exception handling, B2B, Exception handling method.

1 Introduction

Business processes are market-centred descriptions of an organisation's activities, implemented as information processes and/or material processes [15]. Basically, a business process is designed to achieve the business goal and satisfy the client.

The difference between e-business and traditional business is that e-business takes advantage of the communications potential of the Internet. In regard to application scope, e-business applications may be classified as both intra-business and inter-business applications. Intra-business includes all enterprise applications, which are designed for enterprise use only and they are connected to the internal business activities. Inter-business applications are used for communication between the enterprise and other organisations or customers. Business-to-business (B2B) applications are designed for business partners; business-to-consumer (B2C) applications are designed for customers [4].

This paper covers B2B applications (the B2C applications are discussed in [16]), which transform inter-organisational relationships with the goal of building stronger partnerships. They can be used to exchange information between businesses, for collaboration, information and knowledge sharing between businesses for mutual benefit [9, 10].

The most common B2B process is supply chain integration, which is the web-enabling of legacy systems to provide visibility and/or access to select partners, sup-pliers or customers.

All B2B applications are time critical, therefore easily configurable reminder mechanisms must also be implemented in these applications. Basically, e-business refers to the use of Internet technologies to improve and transform key enterprise processes [13, 14, and 4]. Business processes described using workflows are transparent, understandable to all process participants and easily con-trolled. E-business processes are usually constructed using workflow technologies and methods; as a result, they can merge customers, information and tasks into one unified environment.

Workflow itself is the automation of a business process in whole or in part, during which documents are passed from one workflow participant to another for action ac-cording to a set of procedural rules [8]. E-business workflow processes are of the production workflow type, because they involve repetitive and predictable business processes. However, it must be recognised that many business processes tend to be unpredictable and have a tendency to change or adapt to a current business situation. Taking account of this fact, production workflow is usually combined with ad-hoc characteristics or they become an ad-hoc workflow, as it must be adapted on demand.

As business processes are usually complex, e-business workflow can be divided into independent, but communicating between each other, workflows (or sub-workflows), which separately deal with discrete but related tasks. The results of these workflows later form a core workflow process which must be processed further according to the defined business rules.

However, defined workflow processes are rather difficult to support and are usually not tolerant of any kind of exceptions. These exceptions can be very diverse, with some of them being predictable and addressable in advance, some unpredictable. Unpredictable exceptions can change processes radically, decreasing the processes' chances of delivering the desired business outcome [5]. Even these exceptions can be solved; the process can be modified and the goal can be reached.

The meta-model for exception handling consists of these steps: exception detection, diagnosis and handling. Exception handling results are always dependent on the detection phase. However, the proper and timely detection of an exception, and gathering of necessary supporting data, is quite difficult to achieve. There are many methods to describe an exception, its source and ways of solving it. Seeing as exceptions have direct impact on the achieving of desired business outcomes, in business processes, an exception must be defined in the context of the business process goal. Unfortunately, current workflow modelling tools lack this important feature.

This paper explores technology, which could be used to handle unexpected exceptions intelligently and reduce their impact on the final business process outcome.

2 Proposed Exception Classification

Business activities, which cannot be executed in a predefined manner and cannot reach their desired outcomes, are referred to as exceptions. Workflow requires specification of both normal process flow and possible variations due to exceptional situations that can be anticipated and monitored [2].

Non-predicted situations can be divided into:

- Failures, which can be divided into [17]:
  - Basic failures – corresponding to failures at the system level (e.g. DBMS, operating system, or network failure); and
  - Application failures – corresponding to failures of the applications invoked by the Workflow Management System (WFMS) in order to execute a given task.
- Exceptions can be grouped into:
- Expected, which are [17, 12]:
  - Workflow exceptions – related to starting or finishing tasks;
  - Data exceptions – related to workflow data change;
  - Temporal exceptions – time related exceptions;
  - External exceptions – related to external events.
- Unexpected are tend to be classified as [3]:
  - Useful exceptions – a "key to effective and flexible" processes, usually easily handled;
  - Unanticipated exceptions – the result of an unexpected, infrequent and non-repetitive event.

These types' failures are closely related, since basic failures can cause application failures. The failure itself is not considered an exception, because it is usually a system failure and can be ascribed to an infrastructure level exception. The failure itself usually is a result of some core exception and must be solved through the core issue, rather than at the business process level.

Useful exceptions are always solved during the build-time. The dedicated process graph is supplemented with special branches, which would be used during the initiated process instance should the need arise. These are predefined exceptions. This paper focuses on unanticipated exceptions, whose handling cannot be predefined during build-time.

In essence, expected exceptions cannot be called true exceptions, simply because they do not impact the final goal of the business process – a method of handling them is already defined, which allows achieving the final goal.

Exceptions can occur in a process instance synchronously or asynchronously with the process flow. The "useful exceptions" tend to occur synchronously to the flow and do not overly impact the work model and the final goal is reached, though it can be slightly influenced or changed [3]. Unexpected and unknown exceptions usually occur asynchronously to the flow cause critical damage to the process' ability to achieve its goal. Such exceptions are dangerous, since they are usually identified only after the damage has already been done. Furthermore, the process step where damage is visible is usually not the step where the damage was done. The key to the handling of these unexpected exceptions is identification of the point where that damage was done.

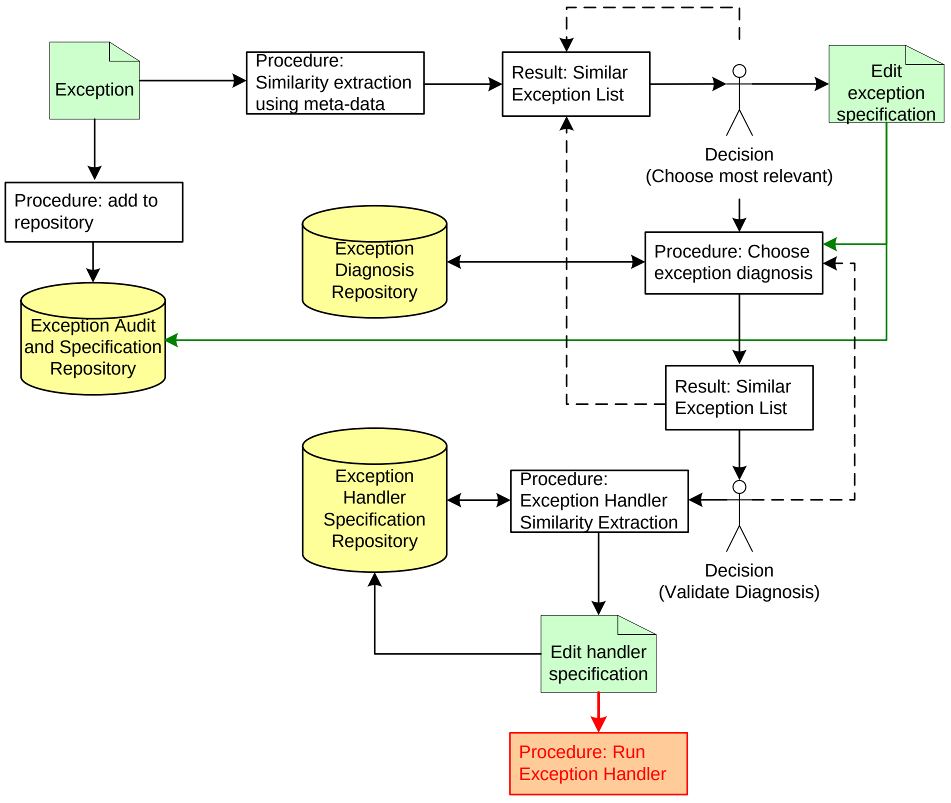
3 Conception for a Workflow Based Exception Handling

It is a truism that a system is as easy to support as it is flexible. Flexibility means:

- Easy design and change;
- Easy enactment of changes in running workflow instances;
- Fluent and transparent support of exception handling and failure recovery; and
- Dynamic workflow schema evolution [1].

With regard to business process exception handling, the business process context and its expected final result must be taken into consideration. The basic algorithm for the handling of any exception is detection, diagnosis and handling. The system must be able to specify the exception using JECA rules, as previously discussed [5]. That would be the exception detection step. Later steps depend on this step's specification, so systems must have the ability to specify the detection in detail and on time. Also, the system must be able to inform relevant users about the situation and give the direct access to the specification. The user must be supplied with the instruments, which would let the user select the right handling tool and use it in the JECA specified place.

The diagnosis step must be finished with the generation of the exception handling specification, which should be directly passed to the exception handling engine for processing. The exception handling process is completed with the handling phase. Figure 1 represents the workflow schema for the automated exception detection, diagnosis and handling sequence fulfilment.



Exception
Procedure: Similarity extraction using meta-data
Result: Similar Exception List
Decision (Choose most relevant)
Edit exception specification
Procedure: add to repository
Exception Audit and Specification Repository
Exception Diagnosis Repository
Procedure: Choose exception diagnosis
Result: Similar Exception List
Exception Handler Specification Repository
Procedure: Exception Handler Similarity Extraction
Decision (Validate Diagnosis)
Edit handler specification
Procedure: Run Exception Handler

Fig. 1. Exception handling procedure framework using workflow method..

Detection, diagnosis and handling must be related to business process scope, the expected process result and inter-acting resources. Business processes are usually modelled using workflow methods, therefore, it is logical to handle exceptional situations is a similar manner. This research paper models exception management using workflow. Most of the tasks in the exception handling workflow are automatic and only some of them require human interaction. Usually human interaction is needed at the end of the one of the three phases (detection, diagnosis, handling), to confirm the automatically made decisions. Confirmation is usually needed due to the unpredictability of real-world situations. The business process executor is almost eliminated from the exceptional situations handling, excluding rare situations, when the exceptional situation must be detailed immediately from the user's point of view. The automatically formed exception specification is sent to the exception handling engine for processing and/or to the system administrator for the approval.

A positive aspect of workflow-based exception handling is that the same exception handling process may be modified, adjusted or changed using the same exception handling methods.

4 Technologies Used to Fulfil the Proposed Method

A business process exception management system is as flexible and easily supported as it is decomposable. This requirement also ensures that the system will be independent from any platform or any other legacy systems. As a result, it will be easily adapt-able to business process management systems and other applications. Users would access the business management system using their distinct user permissions and roles profiles; different users can have different access to the different application modules. These profiles must be managed in common profile module where they must be created and stored for every user. Every profile defines application access and personalisation: access to different modules, main visualisation specifics, access rights and allowed actions.

According to these profiles, users would be given individual environments for in-putting their requests. Once the request is filed, it is passed to the e-business process logic, which defines processing sequence for that request. Deviations usually occur between these two steps and during the business process logic's processing. That means it is necessary to have an autonomic back-end exception handling application, which has access to both business applications and the user profile database.

The exception handling application must have a dedicated schematic of all business logics, represented as possible exception tree [3] and should have three running clients: exception detection, exception diagnostics and exception handling clients. The exception tree can be composed using either taxonomies or ontology; according to business process context and use cases. In terms of running clients, the exception detection client must be active all the time and compare activated process instance business logic execution to the exception description schematic exception cases. Sentinel components, which would look for the appropriate patterns in the behaviour of the basic components, can be used for this purpose [6][7].

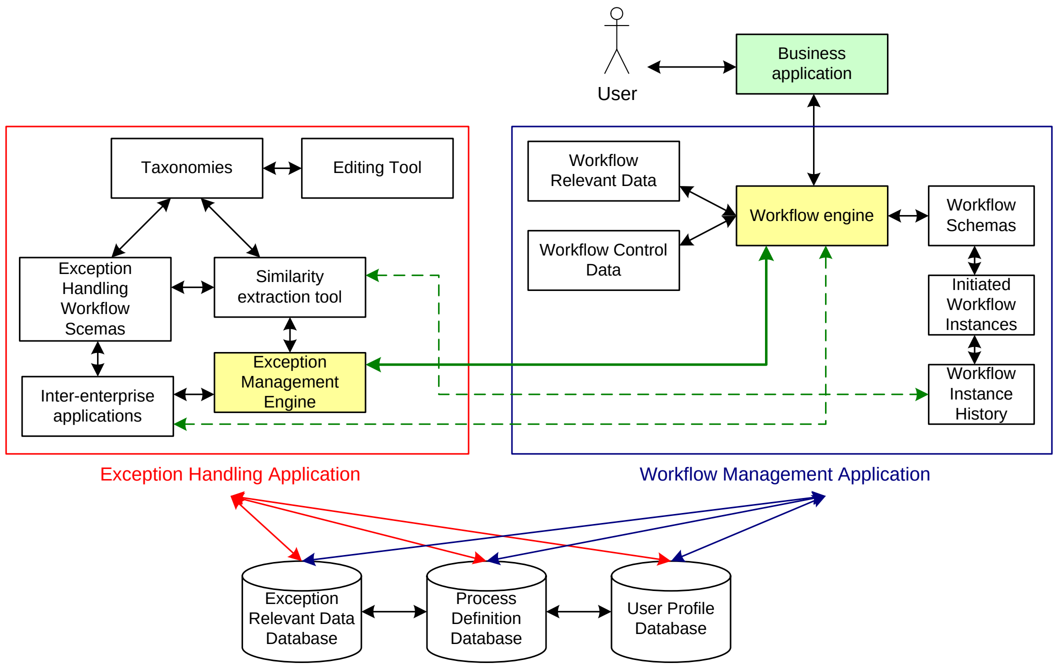
The business process exception management model, which is a result of the research described in this paper, consists of three parts and can be fully decomposed; together these parts compose the common business process management environment. The architecture described in the figure 2 consists of:

- Workflow management module - The core of this module is a workflow engine, which is usually used for the running workflow processing. The workflow engine acts according to the predefined workflow schemas (which are stored separately in the workflow schema creation and management application.) The workflow engine uses a workflow schema to define the particular instances of this schema, which differ in terms of input and output. For the proper detection of exceptions, workflow schema instance processing history is collected and is later used to deal and interact with the similarity extraction tool component of the exception handling module. The additional data needed for workflow schema instance processing is accessed from the workflow relevant (context or actors specifications,) and, workflow control (triggers, business rules, administrative information) data virtual repositories. Basically, the workflow management module has strong data analysis and auditing tools, which help to gather workflow instance data that is later used for data similarity extraction.
- Exception handling module – This module interacts with the workflow management module through the similarity extraction tool module. Its main function is to decide whether a running workflow schema instance has exceptional situations. For that purpose, the similarity extraction tool compares workflow schema instance history data with the appropriate taxonomy data. If a distinction is encountered, the similarity extraction tool refers to the exception handling workflow schemas selection component. This component's main function is to identify whether there is any proper exception handling schema for this particular exception. If none are found, the exception handling workflow schema component can provide the system administrator with the most similar schemas (according to the running workflow schema instance history data.) With this information, the

administrator can adapt one of the existing schemas or create a new one using the editing tool. The selected or newly produced exception handling workflow schema is passed to the workflow engine for processing.

- Repositories - Must be compatible with common database management systems and have all tools for the stored content management (creation, retrieval, processing, and delivery). According to this research model, there should be three types of repositories:
  - o Exception specification and relevant to exceptions data repository;
  - o Business process definition repository;
  - o User profile repository (as discussed above.).

The main requirements for these repositories are scalability, flexibility and extensibility.



User
Business application
Taxonomies
Editing Tool
Exception Handling Workflow Scemas
Similarity extraction tool
Exception Management Engine
Inter-enterprise applications
Workflow Relevant Data
Workflow Control Data
Workflow engine
Workflow Schemas
Initiated Workflow Instances
Workflow Instance History
Exception Handling Application
Workflow Management Application
Exception Relevant Data Database
Process Definition Database
User Profile Database

Fig. 2. Business process management system communication with exception handling system architecture framework.

If the exception handling model proposed in this paper would be implemented in a particular industry business process context, there would be good reason to have exception ontology rather than taxonomy (referring to the [11], which uses a taxonomy tree, but in this context, ontology would be more effective), which would define appropriate input, output and processing of the common data. The sentinels could com-pare actual data with an ontology tree. If data does not match, a sentinel could pass this data to the exception registration agent.

Any business process flow processing can involve many people with various roles, who have different tasks and responsibilities. The exception detection sentinels must take account of that - if the ontology itself provides or links to information on roles and task lists.

In fact, the business logic description and definition module, exception request handling or output generation modules can have different exceptions ontology. In such a way it is possible to achieve more detailed exception situation specifications. Different modules can have personal sentinels and exceptions registration agents.

5 Conclusion

Business processes will keep evolving and changing, human requirements probably will never be satisfied and technologies will keep emerging only faster. In case to cash with the progress every organization and business process engineers must look forward while designing applications or constructing processes. It would be better if business process applications should exchange data using XML, would be separated into modules, so that the configuration of system would be easier. The problem of avoiding exception and their handling is not such a trivial task. Still we hope, that application decomposition with their own configuration can lead to system flexibility. If we let user take part in configuring the system, it may lead into a better process execution performance and result. The exception handling can be reached only collecting knowledge, analyzing it and storing for the future cases. If we make a system to collect information, identify similar cases and decide how to solve with minimal human interaction, the e-business processes will be easier to change.

References:

[1] Hu J., Grefen P.: Conceptual framework and architecture for service mediating workflow management. *Information and Software Technology*, Number 45, 2003: pp 929-939.

[2] Casati F., Fugini M. G., Mirbel I.: An Environment for designing exceptions in workflow. *Information Systems*, Volume 24, Number 3, 1999: pp 255-273.

[3] Sadig W. S., Orlowska M. E.: On Capturing Exceptions in Workflow Process Models. The University of Queensland, Australia, Department of Computer Science and Electrical En-gineering.

[4] Ennser L., Leo P., Meszaros T., Valade E.: *IBM Redbooks, The XML Files: Using XML for Business-to-Business and Business-to-Consumer Applications*. ITSO, IBM Corp.

[5] Zongwei Luo, A. Sheth, K. Kochut and J. Miller.: Exception Handling in Workflow Sys-tems. *Applied Intelligence*, Volume 13, Number 2. September/October, 2000: pp125-147.

[6] Sadiq S.W., Marjanovic O., Orlowska M.E.: Managing Change and Time in Dynamic Workflow Processes. International Journal of Cooperative Information Systems, World Scientific Publishing Company.

[7] Dellarocas C.: Toward Exception Handling Infrastructures in Component-Based Software. *Proceedings of the International Workshop on Component-based Software Engineering*, 20th International Conference of Software Engineering (ICSE), Kyoto, Japan, April 25-26, 1998.

[8] *Workflow Management Coalition: Terminology & Glossary*. WFMC-TC-1011. Issue - 3.0. Winchester Hampshire, United Kingdom. 1999.

[9] IBM Corp.: DB Magazine: Strategies & Solutions for Database Professional. Volume 8, Number 2, Quarter 2, 2003, pages: 16-19.

[10] G. Alonso, R. Gunthor, D. Agrawal, A. El Addadi, C. Mohan: Exotica/FMDC: Handling Disconnected Clients in a Workflow Management System. *Proceedings of the 3rd Confer-ence on Cooperative Information Systems*, Vienna, May, 1995.

[11] Klein M., Dellarocas C.: Towards a systematic repository of knowledge about managing multi-agent system exceptions. Working paper AES-WP-200-01, Centre for Co-ordination Science, Massachusetts Institute of Technology, Cambridge MA USA, February, 2000.

[12] Worah D., Sheth A., Kochut K., Miller J., An Error Handling Framework for the ORBWork Workflow Enactment Service of METEOR, Large Scale Distributed Information Systems Lab, The University Of Georgia, Athens.

[13] Naick, I., Berkhoff, M., Verdugo Bosnich, D.: *IBM Redbooks*, Business-to-Business Integration Using MQSeries and MQSI Patterns for E-business Series. ITSO, IBM Corp.

[14] Дэвид, К., Электронная коммерция, *Microsoft Press*, 1999.

[15] Georgakopoulos, D., Hormick, M.: An Overview of Workflow Management: From Process Modelling to Workflow automation Infrastructure. Distributed and Parallel Databases, Vol. 3. Kluwer Academic Publishers, Boston (1995) 119-153.

[16] Vojevodina, D.: Business process exception handling using workflow methods. *Lithuanian Mathematical Journal*, Special Issue 43. Mathematics and Informatics Institute, Lithuania (2003) 210-315.

[17] Casati F.: Models, Semantics, and Formal Methods for the Design of Workflows and their Exceptions. *PhD Thesis*, Department of Electronics and Informatics, Politecnico de Mi-lano (1998).