VILNIUS GEDIMINAS TECHNICAL UNIVERSITY

Dovilė Vojevodina

MODELLING OF E-BUSINESS PROCESS EXCEPTION HANDLING USING WORKFLOW METHOD

Doctoral Dissertation

Technology Science, Informatics Engineering (07T)

Vilnius, 2006

The dissertation has been written during 2002 - 2006 at Vilnius Gediminas Technical University.

Scientific supervisor:

prof. habil. dr. Genadijus KULVIETIS (Vilnius Gediminas Technical University, Technology Sciences, Informatics Engineering – 07T).

CONTENTS

Picture Index 5
Table Index 8
Abbreviation Index 9
Introduction 13
1 The Analysis of Business Process Modelling Methods 19
1.1 Introduction 19
1.2 Analysis of Usage of Workflow Methods in Business Process Modelling 20
1.3 Analysis of Business Process Management Technologies 27
1.4 Business Process Exceptions Classification 37
1.5 Research Object and Tasks 41
2 Business Process Exception Models 43
2.1 Introduction 43
2.2 Basic Exception Handling Methodology 43
2.3 The Models of Business Process Exceptions in Business-to-Client environment 47
2.4 The Models of Business Process Exceptions in Business-to-Business environment 54
2.5 The Usage of Workflow Methods in Business Process Exception Handling 65
2.6 Conclusions 78
3 Business Process Business-to-Client Exception Handling 80
3.1 Introduction 80
3.2 Business Process Business-to-Client Exception Handling Algorithms 81
3.3 Business Process Business-to-Client Exception Handling Realisation 96
3.4 Conclusions 109
4 Business Process Business-to-Business Exception Handling 111
4.1 Introduction 111
4.2 Business Process Business-to-Business technologies 112
4.3 Business Process Business-to-Business Exception Handling Models Using Workflow Method 122
4.4 Conclusions 141
Conclusions 143
References 145

Annex No. 1 – B2B exception handling system overview 152

Picture Index

Picture 1. Business process structure. 21
Picture 2. Basic e-business process example. 25
Picture 3. The layers of enterprise architecture model. 26
Picture 4. The dimensions for basic exception specification. 40
Picture 5. The sequence of exception lifecycle. 41
Picture 6. Business to Consumer model conception [15]. 49
Picture 7. Business-to-Consumer Application Framework for e-business architecture [15]. 50
Picture 8. Purchase order workflow. 51
Picture 9. The example of exceptional situation in B2C application usage. 53
Picture 10. E-business and business process [26]. 55
Picture 11. Separation between Markets and Hubs. 55
Picture 12. B2B solution components. 56
Picture 13. Business-to-Business model conception. 56
Picture 14. Business-to-Business Application Framework for e-business architecture [15]. 57
Picture 15. Trading process with associated exceptions [12]. 60
Picture 16. Trading process properly instrumented with logging processes [12]. ..61
Picture 17. E-banking application main execution process. 61
Picture 18. B2B type exception, encountered performing credit card payment. ...63
Picture 19. B2B exception example encountered trying to connect to business application. 64
Picture 20. Widely used process notations. 65
Picture 21. Node state interaction basic model [66]. 66
Picture 22. WfMC Reference Model [21, 80]. 67
Picture 23. Exception knowledge sharing model [74]. 69
Picture 24. Conceptual model of coordinated exception handling in cross-organisational environment [73]. 72
Picture 25. Coordinated exception recovery [70]. 73
Picture 26. Exception handling and resolution [70]. 75
Picture 27. CBR based exception handling. 75

Picture 28. Case adaptation model [73]. 77
Picture 29. Intelligent problem solving framework architecture [73]. 77
Picture 30. E-business application classification scheme [30]. 80
Picture 31. Task execution process flowchart. 83
Picture 32. Task execution lifecycle. 83
Picture 33. Task module in overall application conception. 86
Picture 34. Advanced task execution lifecycle. 87
Picture 35. Advanced task process execution flowchart. 88
Picture 36. Executor re-assignment embedded process flowchart. 91
Picture 37. Organisational structure, used for actor roles identification [68]. 92
Picture 38. Process instance state transitions meta-model [68]. 93
Picture 39. Activity instance state transitions meta-model [68]. 94
Picture 40. Enhanced application architecture. 95
Picture 41. B2C exception handling scheme. 100
Picture 42. B2C application exception handling conceptual architecture. 101
Picture 43. The conceptual model of system coordinator. 103
Picture 44. The conceptual model of the handler mechanism. 106
Picture 45. The physical model of the handler. 106
Picture 46. Exception handling system architecture. 108
Picture 47. High level SOA solution model [7]. 115
Picture 48. Main pieces of BPM architecture and their relationships [20]. 117
Picture 49. BPEL4WS logical view [5]. 118
Picture 50. Artificial neuron structure [87]. 119
Picture 51. “Black-Box” system conception. 120
Picture 52. B2B application high-level conception [28]. 122
Picture 53. Issue taxonomy, containing exception, change and failure types. 125
Picture 54. E-Commerce application exception, received after failed money payment. 127
Picture 55. “Butterfly effect” initiated by unexpected exception. 128
Picture 56. Diagnosis taxonomy, containing failure, exception and change types. 129
Picture 57. Handlers’ taxonomy, including failure, exception and change types. 129
Picture 58. Supply Chain Management process using BPMN. 132

Picture 59. Flexible exception detection and handling e-business architecture, incorporating mid-layer between business and application layer. 133
Picture 60. Exception handling procedure meta-model (algorithm) using workflow method. 138
Picture 61. The exception handling workflow model (designed during this research using EMC Documentum Business Process Manager). 139
Picture 62. Similarity level extraction workflow (created during this research), modelled as sub-process for the main exception handling workflow process. 140
Picture 63. Main interface of B2B exception handling system. 152
Picture 64. Exception handling process monitoring dashboard. 153
Picture 65. Exception taxonomy module, which has all possible branches and can be supplement with new ones at any moment. 153
Picture 66. Exception taxonomy branch record. 154
Picture 67. Diagnosis taxonomy module, which has all possible branches and can be supplement with new ones at any moment. 155
Picture 68. Diagnosis taxonomy branch record. 155
Picture 69. Handler taxonomy module, which has all possible branches and can be supplement with new ones at any moment. 156
Picture 70. Exception handler taxonomy branch record. 156
Picture 71. Exception specification repository with main lifecycle steps and collected records. 157
Picture 72. Exception repository record, containing full description of attributes and descriptive information in attachment box. 157
Picture 73. Diagnosis specification repository with main lifecycle steps and collected records. 158
Picture 74. Diagnosis repository record, containing full description of attributes. 158
Picture 75. Exception handlers specification repository with main lifecycle steps and collected records 159
Picture 76. Exception handler repository record, containing full description of attributes. 159

Table Index

Table 1. Workflow process classification due to the degree of their structure. 23
Table 2. Workflow automation features table 29
Table 3. Specific functions and features of scientific workflow technology 30
Table 4. Scientific workflow tools features evaluation. 32
Table 5. Commercial workflow management tools evaluation/comparison [91, 62]. 36
Table 6 The possibilities of instance resuming after exception handling. 46
Table 7 Main B2C and B2B distinctions. 58
Table 8 Task execution lifecycle description 84
Table 9 Task execution lifecycle definition. 89
Table 10 Table of state list projected on tasks described above in task execution flowchart. 93
Table 11 The matrices of goals and critical issues. 98
Table 12 The problems, their solutions and limitations. 99
Table 13 The summary of research results in percentage. 109
Table 14 Six Challenges of SOA [46]. 116
Table 15 BPM standards [66]. 116
Table 16 Attributes and weights for exception, change and failure. 126
Table 17 Legend of notations used in Picture 60. 130
Table 18 The explanations of notations used to describe exception handling algorithm. 138

Abbreviation Index

A2A (Application-to-Application) – internet usage for insuring application communication,

ANN (Artificial Neural Network) – information processing paradigm, inspired by the way biological nervous systems process information,

API – (Application Program Interface) - a set of routines provided in libraries that extends a language's functionality,

ASC X12 (Accredited Standards Committee X12) - a committee responsible for writing generic EDI standards,

B2B (Business-to-Business) – e-business type involving only companies as participants,

B2Bi (Business-to-Business Integration) - usually is one-to-one relationship between buyer and seller and covers programmatic links between arms length businesses,

B2C (Business-to-Client) – e-business type involving companies and customers as participants,

B2E (Business-to-Enterprise) – intra-network usage for product delivery to employees,

B2M2B – (eMarketPlace) – B2B sub-type, a relationship one-to-many, many-to-many and many-to-one between buyer and seller,

BPDM (Business Process Definition Meta-model) – business execution language and/or notation language,

BPE (Business Process Exception) – non-predicted and unexpected workflow management system behaviour,

BPM (Business Process Management) – a methodology for managing business process elements,

BPMG.org (The Business Process Management Group) - a global business club exchanging ideas and best practice in business process and change management,

BPMI.org (Business Process Management Initiative) - provides leadership and industry standards for this vital and growing industry,

BPML4WS (Business Process Modelling Language for Web Services) – a standard developed by IBM, BEA and Microsoft for BPM delivery through Web services,

BPMN (Business Process Modelling Notation) - provides capability of understanding business procedures in a graphical notation and standard manner,

BPQL (Business Process Query Language) - a management interface to a business process management infrastructure,

BPRI - is a leading research based consultancy, offering a customised approach to client needs and providing tailored solutions to their business issues,

BPSS (Business Process Specification Schema) - a standard framework, by which business systems may be configured to support execution of business collaborations consisting of business transactions,

CA action (Coordinated Atomic Action) – business process activity behaviour do not interacting with the rest of activities,

CBR (Case-Based Reasoning) - the process of solving new problems based on the solutions of similar past problems,

CPR (Compensation Proceeding Rework) – exception handling method using rework as a base,

CRM (Client Relationships Management) – a collection of methodologies, strategies, software and web-based capabilities helping to achieve client management,

DAG (Directed Acyclic Graph) - a directed graph with no directed cycles,

DBMS (Database Management System) - software environment that structures and manipulates data, and ensures data security, recovery, and integrity,

DMZ (Demilitarized Zone) – network area that sits between organisations internal network and an external network, usually Internet,

EAI (Enterprise Application Integration) - a technology that connects enterprise-wise systems within a company,

E-business – any business process, which relies on automated information system,

ECA (Event-Condition-Action) – a form of rules that can be specified within the scope of different activity and sub-activity levels,

EDI (Electronic Data Interchange) – computer-to-computer exchange of structural information,

ERP (Enterprise Resource Planning) – management information system integrating and automating business practices associated with operations aspects of company engaged in manufacturing products or services,

FIFO (First In First Out) - a method of coordinating the sequential flow of data through a buffer,

G2B (Government-to-Business) – the integration of business and government initiatives and activities,

G2C (Government-to-Citizen) - denotes online communication between Government and consumer/individuals,

G2G (Government-to-Government) – the integration of government application for common goal achievement,

HR (Human Resource) – organisation management aspect concerning labour hiring, firing, training and other personnel issues,

Internet – the vast collection of inter-connected networks across the world that all use the TCP/IP protocols,

JECA (Justification-Event-Condition-Action) – a set of rules for exception specification,

OMG (Object Management Group) - an international, open membership, not-for-profit computer industry consortium,

PKI (Public Key Infrastructure) – publicly accessible system for interconnected computer networks, transmitting data using Internet protocol,

PNML (Petri Net Mark-up Language) - is a proposal of an XML-based interchange format for Petri nets,

SLA (Service Level Agreement) - formal agreement between a Service Provider and customers to provide a certain level of service,

SOA (Service Oriented Architecture) - a collection of services that communicate with each other,

UML (Unified Modelling Language) – non-proprietary, object modelling and specification language used in software engineering,

VAN (Value-Added Network) - is a private network provider that is hired by a company to facilitate electronic data interchange or provide other network services,

VPN (Virtual Private Network) - a private data network that makes use of the public telecommunication infrastructure,

WAPI (Workflow API) – a set of routines provided in libraries that extends a language's functionality of workflow,

WfMC (Workflow Management Coalition) – non-profit organisation creating workflow management standards together with scientific and commercial organisations,

WfXML (Workflow XML) – XML notation specialized for workflow,

WMS (Workflow Management System) - an autonomous or embedded workflow schema, according to the context of it's is usage, interpretation system,

WofBPEL – a tool (built on Woflan) for formal analysis of BPEL processes,

WS-CDL (Web Services Choreography Description Language) - is an XML-based language that describes peer-to-peer collaborations of parties,

WSCI (The Web Service Choreography Interface) - an XML-based interface description language that describes the flow of messages exchanged by a Web Service participating in choreographed interactions with other services,

WSCL (Web Services Conversation Language) - the business level conversations or public processes supported by a Web service,

WSFL (Web Services Flow Language) - an XML language for the description of Web Services compositions as part of a business process definition,

WWW (World Wide Web) – global information space which people can access via Internet connected devices,

XLANG - is an extension of WSDL that defines the behaviour and orchestration of business processes based on Web Services,

XML (Extensible Mark-up Language) – a W3C recommended general purpose mark-up language for creating special purpose mark-up languages,

XPDL (XML Process Definition Language) - is the language proposed the WfMC to interchange process definitions between different workflow products,

YAWL (Yet Another Workflow Language) – Petri-net based workflow language formally defined for syntax and semantics.

Introduction

The Relevance of the research

Through e-business has emerged not that long time ago, it has already evolved very much and to this day is a key element in business, government and citizen or employee life. This proves the number of branches of e-business which are dedicated for various business environments and held goals and methods dedicated to reach them. Government and business societies agree that e-business has its uses in effectively combining connected business and government processes. Citizens and employees agree that e-business is a very attractive and convenient form of performing regular task as of social, as of business area.

To this day already exists quite a number of non-profit societies (WfMC, BPMI.org, BPMG.org, OMG), which mostly are created from leading commercial and scientific representatives working at the same goal to create and improve the standards for reliable and flexible business management environments and supporting tools. To this day, already there are more than 200 of commercially available BPM solutions. While quite an amount of them do not correspond to high requirements held for such solutions, still a big part of them are dedicated to high level flexible and inter-operability features realising business solution implementation. What is still a problem with e-business solutions and its tools; it is the lack of reliability and ability to reflect real world changes. The situations of which nor the business representatives, nor e-business applications were aware occur more often than expected. These situations cause can be of most unimaginable nature: organisational, legislation changes, fast evolution of leading technologies, even human mind. Regretfully, to this day, there are no e-business supporting applications or instruments which would help to deal or at least manage the damage of such unexpected and unpredicted situations, which further will be addressed as exceptions.

Such lacks of e-business technologies impede with usage and implementation of many e-business solutions, which would save people's time and minimise the effort needed to fulfil particular procedures. The other obstacle is the lack of user satisfaction and trust to existing e-business technologies.

Many world wide commercial and independent organisations, such as Gartner, Capgemini, Forrester, Bruse Silver Associates and many others, delivers the annual reports and surveys on this subject, evaluating the progress of improves of e-business technologies and evaluating current commercial products. And only 62% of citizens are satisfied with e-government services, while average time saved for performing e-services could save up to 81 of person's minutes [36]. Various results show that there are still many spaces for improvements of e-business

technologies and gaining trust in using them while performing critical business operations and managing simple user accounts.

The Object of the Research

The object of the research is business process exception handling modelling. The research is performed through the analysis of business process characteristics, business process mapping using workflow, business process and workflow exception appearance sources and causes, also possible exception handling methods with a help of workflows.

The Tasks of the Research

Tasks held for this research achievement are:

- To perform the analysis of business process method usage, taking into account all the relevant business process exception scope.
- To research and to adapt business process exception handling models in business-to-client and business-to-business environments.
- To create and practically implement the business process exception handling model in business-to-client environment.
- To create and to test practically the business process exception handling model in business-to-business environment.

Methodology of the Research

The research was performed through these steps:

- Analysis of scientific technical literature on workflow and business process exceptions,
- Analysis of several most popular commercial business process management and workflow management technologies/software,
- Analysis of most existing scientific workflow and exception management tools,
- Analysis of current business demand and market in business process as workflow management area,
- Definition of assumptions for flexible workflow exception management and their testing in several commercial software environment,
- Definition of achieved results and their effectiveness and also usefulness evaluation.

During this research the achieved results were also tested in real business environment. Monitored results were analysed with an aim to identify still weak spots of the solution and to produce fixes for non-flexible solution spots.

The scientific novelty

Regardless the great possibilities of BPM technologies, they still lack the true exception, which are totally unpredicted and unknown to implemented business process, handling instruments. The demand for such instruments is great, as the same nature of business process is totally unpredictable, and having in mind that human decisions are not always interpretable by machines, exceptions are situations, which can raise critical consequences to consumer, business process and its managing company.

During this research the classification and taxonomy of exceptions were created, which basically separate them into three separately manageable branches: exception, change and failure. For full solution support, the taxonomies of exception diagnosis and handler were also created and discussed. And finally, for intention of creating the better result of selected solution, the deep analysis of currently available technologies and methodologies were performed, which resulted in comparable study.

The methods of exception handling created during this research, for B2C and B2B environments, combine some of already invented methods for exception handling and integrate the best practices of managing business processes. In such a way the created solutions gains flexibility of their usage and support, possess features of inter-operability, which insures independence from BPM platforms and can be implemented as integrated or totally self-standing collaborative solution. The results of research can also be used in e-government solution realisation.

Practical value

The results of this research and the knowledge gained while achieving it was used and adapted in real world implementations.

The achievements are following:

- The G2G and G2B realisation feasibility study and exception handling issue discussion was performed at UAB ERP company's projects. Such like Lithuanian e-government platform creation feasibility study and requirement analysis for e-services realisation.
- The B2B solution implementation in the biggest Latvian pharmaceutical company Grindeks. The solution is totally based on business process management using workflow.
- The G2C portal implementation in Lithuanian Ministry of Finance, based on flexible platform, which realises e-services delivery possibilities using workflows.

Approbations

The research results were accepted, presented and discussed in 15 republican and international conferences.

1. 6th WSEAS International Conference on Applied Informatics and Communications (AIC'06), 2006, Crete, Greece,
2. CAiSE 2005 Doctoral Consortium on Advanced Information Systems Engineering, 2005, Porto, Portugal,
3. ISDA'05 5th International Conference on Intelligent Systems Design and Applications, 2005, Wroclaw, Poland,
4. 5th WSEAS International Conference on Simulation, Modelling and Optimisation (SMO'05) (Corfu, Greece, 2005),
5. 2005 WSEAS International Conference on Dynamical Systems and Control (CONTROL'05), 2005, Venice, Italy,
6. 13th International Conference on Information Systems Development (ISD'2004), Doctoral Consortium, 2004, Vilnius, Lithuania,
7. CAiSE'04 Workshops, in connection with the 16th conference on Advanced Information Systems Engineering: „The fifth Workshop on Business Process Modelling, Development, and Support", 2004, Riga, Latvia,
8. Information technologies 2004 conference, 2004, Kaunas, Lithuania,
9. RTU 44th International Scientific Conference Section for Electronics and Telecommunications, 2003, Riga, Latvia,
10. Information technologies for business conference, 2003, 2004, Kaunas, Lithuania,
11. XXV Lithuanian Catholics scientific reunion, 2003, Šiauliai, Lithuania,
12. Lithuanian mathematics XLIV, XLVI conferences, 2003, 2005, Vilnius, Lithuania,
13. Conference on Information Technologies: Theory, Practices and Innovations, 2003, Alytus, Lithuania.

Publications

Publications in foreign and international organisations reviewed periodical editions (included into referred/indexed databases):

1. D. Vojevodina, G. Kulvietis. Towards e-Business Application Flexibility: Avoiding Human Factor Exceptions. WSEAS Transactions on Systems, Issue 11, Volume 5, ISSN: 1109-2777, 2006, pp. 2554-2562.

2. D. Vojevodina, G. Kulvietis. From Critical Exceptions to Reliable Process Cases in E-business Models. WSEAS Transaction on Systems, Issue 1, Volume 5, ISSN: 1109-2777, 2005, pp. 248-256.
3. D. Vojevodina, G. Kulvietis, Change and Exception in Business Processes: Handling Differences, WSEAS Transactions on Systems, Issue 10, Volume 4, ISSN-1109-2777, 2005, pp. 1658-1668.

Publications in foreign and international organisations reviewed periodical editions:

4. D. Vojevodina, G. Kulvietis. The Differences of B2B and B2C Workflow Exception Handling. Lithuanian Mathematical Journal, Volume 45, Special Issue, ISSN: 0132-2818, 2005, pp. 180-190.
5. D. Vojevodina. Business Process Exception Handling Using Workflow Methods. Lithuanian Mathematical Journal, Volume 43, Special Issue, ISSN: 0132-2818, p.310-315, 2003.
6. D. Vojevodina, G. Kulvietis. Business Process Unpredictable Exception Handling Using Workflow Method. ISD‘2004 Doctoral Consortium. Scientific Proceedings of Riga Technical University, Series 5, Vol. 22, ISSN 1407-7493, 2005, pp. 68-75.
7. D. Vojevodina, G. Kulvietis. E-business workflow: requirements and realization. Scientific Proceedings of Riga Technical University in Series 7, Volume 4, ISSN: 1407-8880, 2003, p. 25-28.
8. D. Vojevodina. Workflow Exception Handling. Collective Works of XIX Reunion , Book III, Lithuanian Catholics Scientific Academy, ISSN 1392-0499, 2003, pp. 1191-1197.

Publications in other conference proceedings:

9. D. Vojevodina, G. Kulvietis. Butterfly Effect in Exception Management: Preventing Cause and Stopping Reaction. Proceedings of the 6th WSEAS International Conference on Applied Informatics and Communications (AIC’06), ISBN: 960-8457-51-3, ISSN: 1790-5117, 2006, pp.200-207.
10. D. Vojevodina, G. Kulvietis. B2B and B2C Process Nature Influence in Their Exception Handling. Proceedings of the 5th WSEAS International Conference on Dynamical Systems and Control (CONRTOL’05), ISBN: 960-8457-37-8, 2005, pp.97-102.
11. D. Vojevodina, G. Kulvietis, P. Bindokas. The Method for E-Business Exception Handling. IEEE Computer Press, ISDA’05, ISBN: 0-7695-2286-6, 2005, pp. 203-208
12. D. Vojevodina, G. Kulvietis. Business Process Unpredictable Exception Handling using Workflow Based Method. Proceedings of the WSEAS International Conference: 5th

International Conference on Simulation, Modelling and Optimisation (SMO'05), Issue 1, Volume 5, ISSN: 1109-2777, 2005, pp. 248-256.

13. D. Vojevodina. Exception Handling Automation in E-business Workflow Processes. Proceedings of CAiSE'05 Doctoral Consortium, 2005, pp. 42-54.
14. D. Vojevodina, G. Kulvietis. Office Activity Procedure Exception Handling Realization Difficulties. Proceedings of CAiSE'04 Workshops, Vol. 2, 2004, pp.194-201.
15. D. Vojevodina, G. Kulvietis. Computer Decomposed workflow method for run-time office activity procedure exception handling. Proceedings of Conference on Information Technologies'2004, Technology, UDK 004(474.5)(06) Li 301, 2004, p. 400-405.
16. D. Vojevodina, G. Kulvietis. The Algorithm and logic architecture for exception handling in the workflow processes. Proceedings of Conference on Information Technologies for business' 2004, Technology, ISBN 9955-09-649-7, 2004, pp. 237-243.
17. D. Vojevodina, G. Kulvietis. Groupware for Intelligent Business Model Implementation. Proceedings of Conference on Information Technologies for Business' 2003, Technology, ISBN 9955-09-649-7, 2003.
18. D. Vojevodina. Groupware technologies and workflow usage. Scientific Conference on Information Technologies: Theory, Practices and Innovations, Alytus, 2003, p.124-128.

1 The Analysis of Business Process Modelling Methods

1.1 Introduction

Though these days' computer technologies are the main tool used and developed in purpose to help business, the primal purpose of rise of these technologies wasn't such at all. Though electronic business processes realisation is done using Internet (and its technologies), the rise of Internet was based on military goals. In the year of 1952 SAGE (Semi-Automatic Ground Environment) was created on the order of the American air defence. Later this technology was started to develop wider, in the industrial sectors and of course, for the business development.

Already in the year of 1968 was discovered electronic data exchange way (EDI – Electronic Data Interchange), which gave the ability for companies to commit transactions, deals in electronic environment. Later, in the year of 1984, ASC X12 standard helped to process big amounts of transactions and data. This was the beginning of business-to-business (B2B) connections.

In 1972 the first electronic mail system was made. This was the first step towards computer based business technologies evolution. After several years, in 1982, the world was presented with so called "killer application", Lotus 1-2-3 application. This was the first step towards groupware technologies, which later successfully started to realise electronic workflow management. Though workflow were used even for building pyramids, scientifically described and technologically realised they were quite late.

Finally, in the year of 1998, the Amazon.com was registered which realised electronic books marked of business-to-client type for the first type.

To this day, business process modelling and realisation technologies are developed with a challenge of concurrency and sometimes scientific goals.

This research is also a challenge to enhance existing computer technologies. Though existing technologies are capable of processing big amounts of information, to make business critical decisions and connect separated in distance people, still, they lack human intuition, logic independent decision making or process execution direction to non-standard path possibilities. Such features are quite vital for electronic business management systems.

The business itself is of a much unpredicted nature and the technologies are not quite reliable. So the human goal is to assure that business goal would be reached in time in one or the other way.

The purpose of the first chapter is to analyse the business process modelling ways, to evaluate their management technologies and to classify identifiable and non-predicted exception

groups in business processes in purpose later to use this analysis for the flexible, business process encountered exception handling model creation using workflow methods.

1.2 Analysis of Usage of Workflow Methods in Business Process Modelling

The term of business process is quite often heard in daily life, especially in business companies. Business community tend to think of business process as of a phenomenon which brings value to the company and makes money. But what is left behind is the essence of the business process, what it is really is what does it consists of and how it works. Business people are more familiar with business processes from the expected and achieved results perspective. This research is dedicated to the essence and ways of existence of business process and to achieve this, one should go behind it.

Definition 1 *Business Process is a set of one or more linked procedures or activities which collectively realise a business objective or policy goal, normally within the context of an organisational structure defining functional roles and relationships [68].*

Other glossaries define business process mostly as a collection of activities or business rules which are meant to produce the value and output for the customer or the organisation. These definitions are also correct, but as this research focuses on business process background aspects, the Definition 1 will be used. This definition was formulated by Workflow Management Coalition (WfMC) which has over 300 member organisations worldwide, representing all facets of workflow, from vendors to users and from academics to consultants. The mission of this coalition is to:

- Increase the value of customers investment in workflow technology,
- Decrease the risk of using workflow products,
- Expand the workflow market through increasing awareness for workflow [89].

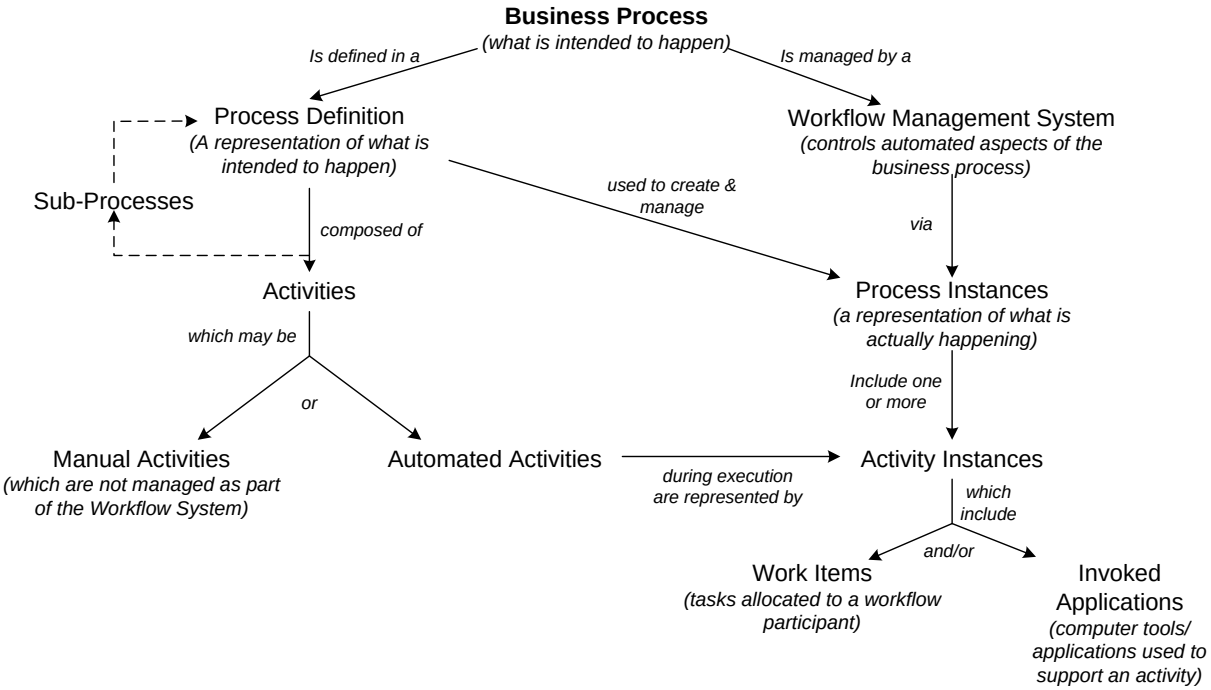
The coalition has defined the reference model for workflow and written quite all specifications for workflow interfaces, which will be analysed later.

Returning to the essence of business process it is necessary to define what it is used for:

- To ensure one particular line of organisation existence. In other words business process can be used to achieve not only the final result of the company's mission, but also serve to commit middle objectives and goals on the way to this grand goal. Such processes can be interpreted as not only in the context of the organisation, but also span outside the organisation.

- A business process also can serve as a trigger for initiating other processes; in that case, the goal of this process is to check for the opportunities of initiating these other goals.
- A business process can be used to arrange the everyday tasks in the company (which usually are staff management, document management, client relationships management or even collaboration processes), which serve as subsidiary tasks for achieving final organisational goals.

Looking from behind the business process we can see that it covers much more and nothing like achieving goals. From the WfMC point of view, the business process is a set of intentions, which are driven by formal representations of these intentions and controls needed to achieve it. The formal representations are composed of various activities and controls are used to observe the actual results of these formal representations (see Picture 1).



Picture 1. Business process structure.

As it seen in the business process structure, the controls are achieved through the workflow management system, so the process definition is described using workflow.

Definition 2 *Workflow is the automation of 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 [68].*

In other words, we see that workflow is a strict sequence of jobs or even algorithm for the workflow management system, how to execute business process.

Definition 3 *Workflow Management System is a system that defines and manages the execution of workflows through the use of software, running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participants and, where required, invoke the use of IT tools and applications [68].*

Workflow is an extremely valuable technology which usefulness was tested through ages. If correct used, workflow can be understandable for all its users (end-users, business users, system engineers) leaving no chance for misunderstandings.

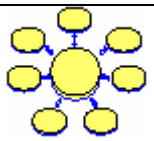
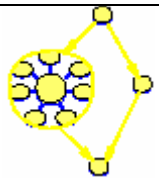
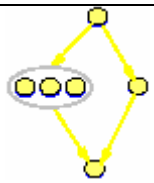
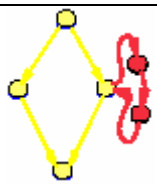
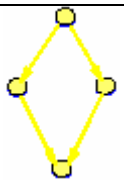
As business processes are meant to achieve some particular goal in vertical or horizontal industry, the workflows used to describe these processes are also modelled according to these industries demands or the results the process must to achieve.

Basically, there exist five workflow types:

- Production workflow – is modelled for back-office data processing, which is highly structured, transaction oriented and clerical in style. The purpose is to automate paper driven processes (like accounts or loans) and to increase productivity of these procedures,
- Enterprise workflow – covers usually department processes in some cases company-wide processes (like HR, procurement or specific company process). Usually they involve data as in electronic (using e-forms), as in paper format and are meant to replace paper-based processing. These processes can involve many various employees,
- Collaborative workflow – is a way for managing present/momentarily knowledge between many employees dedicated to the same goal achievement or task, but separated in distance. It is often used for work team or project management and helps to reduce cycle-times,
- Customer-focused workflow – are quite the only workflow type which is dedicated to the external company's activities (like CRM), which involve company's employees and customers which are connected in the same business case, while having slightly different goals,
- Ad-Hoc workflow – is the only type of workflow which is possible to reflect real life situations. It a combination of all upper four workflow types as it is meant to realise e-business processes, where the main actor is a customer and the rest of the processes is presented as highly structured back-office procedures, with a possibility for a deviation with a help of customer and employee collaboration.

The next table defines all ways of workflow usage for business process construction according to its nature (see Table 1) [14].

Table 1. Workflow process classification due to the degree of their structure.

Unstructured		**Semi-structured**		**Structured**	
Ad hoc	*Cooperative team*	*Integrated team*	*Chained workflows*	*Ad hoc exceptions*	*Standard workflow*
					
Information gathering and processing according to real-world circumstances and rules, helping to exchange knowledge between application and individual.	Processes and their results which are achieved step-by-step, like budget planning. In such processes is vital proper knowledge sharing and management of the task structure.	Is a combination of cooperative and standard workflows, when many people work on one common task while some members are responsible for the task completion.	Strictly predefined workflows which can interact with each one in some particular steps (like employee hiring, interviewing).	Real-world situations when the company client initiates deviation from standard process execution and back-office workers are involved (mostly e-business transactions).	Completely standardised, identical in usage and recurrent workflows (like office activities or documentation management) executed by the strict rules and timing.

According to the context and expected goals/results the process should serve, processes can be modelled in various degree of structure. Usually it depends on information the process is processing. If we have a process, which fully uses only electronic data (like e-forms, meta-data, e-documents or various renditions) and process is capable of processing all this data itself, without human interaction, then it is structured process. Such processes usually are standard, recurrent procedures, like bank or insurance services.

Semi-structured processes are a bit more complex as they involve human decision tasks, which define the next steps of the process. In such workflows, not only structured data can be used, but also unstructured (paper documents or human assumptions). These processes are usually met in project management or execution parts (like creating blue print).

Finally, we have totally unstructured processes, which have the defined basic path of its execution, but rarely follows it to the end. Unstructured processes uses various data, from structured to unstructured, which can not be predefined at the process build-time. Unstructured processes tend to live their life cycle according to real world environment triggers and fully depends on external factors. Unstructured processes are mostly of ad-hoc type and met in e-business environment.

Of course the right chosen model of workflow is very important while constructing any business process, but the element which is vital for good result is workflow management system (WMS). Workflow itself is meta-data which is passed to WMS for processing.

In business process context workflow management systems can be defined as autonomous or embedded according to the context it is usage:

- autonomous workflow management systems can be functional without any additional software, with the exception of database management systems and message queuing services,
- embedded workflow management system is only functional if it is employed with the surrounding (embedding) system, such as ERP system [75].

Autonomous workflow management systems are basically unique and are results of development, using particular interfaces, developed just for this particular business case. While autonomous workflow management systems can exist separately from any other business management systems and integrate with them if need be. Such systems are never depended on any business context and can be configured to execute any type of business process. Still, these autonomous workflow management systems can have different level of accessibility:

- Proprietary solution – which exclusively supports only build-in functionality and does not allow to invoke any other external applications during the run-time,
- Semi-open solution – which has proprietary interfaces for integration of external systems (like office or client applications). In such way the process is executed with a help of additional software components which are configured during the build-time,
- Open solution – which offers interfaces for integration of external systems as server applications (used for inter-enterprise solutions) and usually gives interfaces for invocation services (like starting, stopping or suspending procedures),
- Standardised solutions – defined standardised solutions for the integration of other applications, the interoperability with other workflow enactment services and the client control of the workflow engine.

Usually these solutions are used according to the demands of the business process and its expected results achievement difficulty. Proprietary solutions are perfect for internal supporting procedures processing applications (like documentation management), while open solutions are used for the combined or interactive business processes possessing, when one process must exchange data or even actions with the other. Standardised solutions are commercial solutions which are aimed to some particular application integration or particular marked/industry goals achievement.

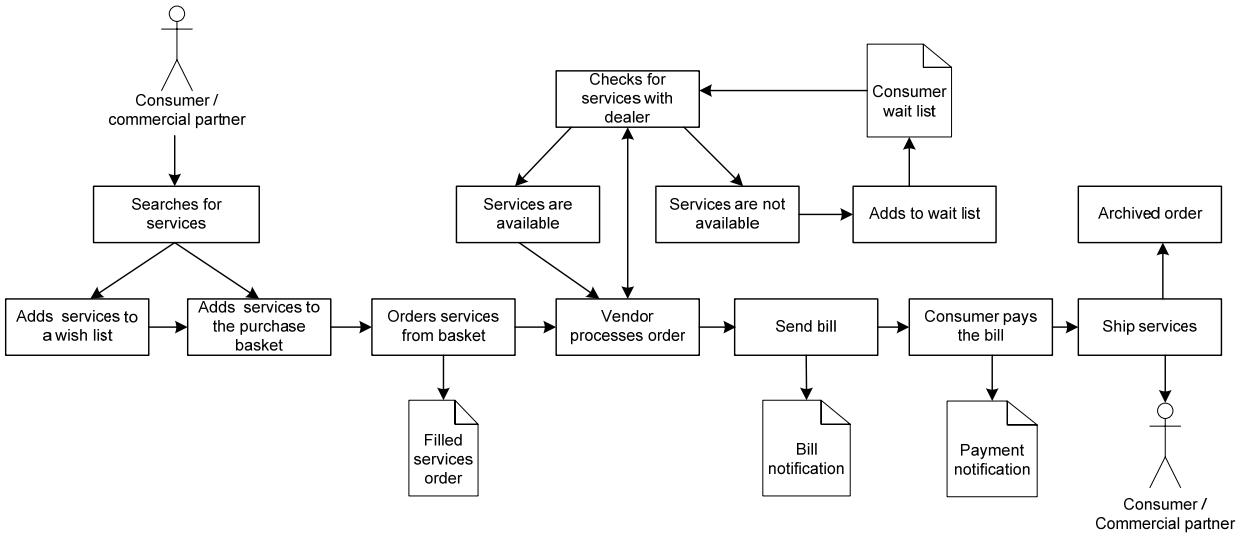
As this research is dedicated to e-business processes context, it is necessary to understand its nature and demand.

Definition 4 *E-Business refers to the usage of Internet technologies, to improve and transform key enterprise processes [30].*

The goal of any e-business is to connect customer, business partners, suppliers and the company itself into the common medium and in so doing make the business more flexible. The difference between traditional business and e-business is Internet characteristics and communication using it. In this case Internet brings many opportunities for the company and for the customer, to help to unify the business context, gives possibility to structure the needed for business data, to optimise the process and the most important to give the customer the freedom of actions or even the sense of importance.

In this research two types of e-business will be discussed:

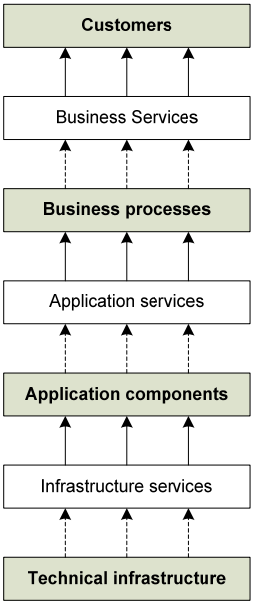
- Business-to-client (B2C) - describes activities of commercial organizations serving the end consumer with products and/or services [90]. In B2C activities are driven by the consumer, which is outside the company, which enables the customer to connect to their information systems and provide a single gateway to the personalized information, needed to make business decisions [15, 38],
- Business-to-business (B2B) describes relations of commercial partners, without serving end consumer [90]. As internet, as extranet can be used to accomplish this type of business and basically transforms inter-organizational relationships. It can be used to exchange information between business and for collaboration, sharing information and knowledge between businesses for mutual benefit [15].



Picture 2. Basic e-business process example.

Both of these types of e-business are of inter-organisational applications types involving only different end users (customers or business partners). Both of these types are usually modelled using the combination of structured and ad-hoc workflow.

What is also very important to mention while trying to understand the nature of e-business processes is the environment in which it resides, the enterprise architecture (see Picture 3). E-business process, especially B2B processes span all these layers. Workflow model is made and mapped to business rules only in Business process layer, while other left layers are responsible for execution of this mapping, which is done through infrastructure (inter-organisational relationships), application (workflow management system) and business services (interfaces for communication).



Picture 3. The layers of enterprise architecture model.

Here it is also necessary to mention one more property of business process, which will be also extremely important in reaching the goal of this research, the workflow inter-operability. It is the possibility of business process, constructed using workflow to interact with external business applications and actors.

Workflow Management Coalition defines five principles of workflow inter-operability which should be met in any flexible e-business workflow. They are:

- Ability to support chained, nested and parallel-synchronised models of interoperability,
- Ability to provide synchronous and asynchronous interactions,
- Possibility to support individual and batch operations,
- Necessity to remain implementation independent,
- Possibility of light, easy to implement protocol usage [67].

As it can be seen, all of these principles are quite related to the nature and properties of workflow itself, as workflow also has the properties of sequential and parallel execution, chained, nested and parallel synchronised task management. What is most important in inter-operability, it is independency of used technologies. Workflow has the property of inter-operability only if it is able to interact with standard external applications and business units, without the need of adaptation.

1.3 Analysis of Business Process Management Technologies

Before starting the analysis of business process management (BPM) technologies, it is important to notice that they are divided into those of commercial and scientific nature. Some of commercial tools are produced together with scientific organisations or research communities, in order to combine the scientific knowledge and best business practices in order to produce a product, which would be technologically advanced and reliable and would fulfil business needs. To this day, there are more than several hundreds of commercial tools available for business process management, while scientific ones are much less.

Essentially important is, if these technologies/tools really can be called business process management tools. First of all, it is important to define, what business process management technology is and what does it consists of.

Definition 5 *Business Process Management Technology – is the convergence of workflow, enterprise application integration (EAI) and unstructured or ad-hoc processes [35].*

Definition 6 *Business Process Management Technology – is a blending of process management/workflow with application integration technology to support rich human interaction and deep application connectivity [34].*

Definition 7 *Business Process Management Technology – is a managerial discipline which target is to improve the efficiency across the systematical management of the business process, which must model oneself, automate, integrate, monitor and optimise it's continuity [90].*

As we can see, business world representatives define BPM technology as one which is able to integrate ad-hoc processes and human interaction, while more scientific sources define BMP technology as a set of specific tools for insuring process efficiency. Basically, all these definitions are true, but for the further research Definition 6 will be used, as it mostly corresponds to this research context.

In more technical context, BPM technology consists of services and hardware that support the explicit administration of processes: analysis, definition, execution and monitoring. To fulfil the tasks raised for these processes, the corresponding engines or tools are created, which are mandatory parts of any BPM technology:

- Business Process Management,
- Business Process Modelling,
- Business Activity Monitoring,
- Business Operations Management.

To fully understand the problem, we also must state the goals and tasks raised for any BPM. The goals are the sources which initiate the creation of corresponding tools to fulfil them, but on the other hand, the goals are also the cause of the business nature exceptions, as they place business terms into goals terms, which can not correspond to the raised business goals.

Basically, the BPM technology fulfilment goals are [35, 90]:

- Extension of the institutional program of quality,
- Fulfilment of legislations,
- The automation of rule-based processes,
- Process execution validity, does the process execution correspond the legislation and real world situation,
- To document processes for outsourcing and definition of SLA (Service Level Agreement),
- Process automation (information, documents and tasks routing),
- Creation and support of the chains of value,
- Integration of legacy systems.

The true workflow automation system must provide quite many mandatory features and capabilities, so that it would be able to automate business process. The next table is the extension of the list of mandatory workflow automation features described in [45].

These are some basic features which must comply with any commercial BPM technology, but scientific ones has some additional features, which should be met. As corresponding scientific technologies and tools are created for innovation, research and analysis purposes and do not execute any business goals, further in this research context, it will be called inter-enterprise workflow technology.

Table 2. Workflow automation features table.

No.	Feature name	Feature description	BPM tool
1.	Graphical workflow maps	Separate user interface for workflow constructions in graphical maps format.	Business process/workflow modelling tools
2.	Simulation for testing workflow applications	Separate or incorporated into graphical workflow maps tool, capable of graphically simulating constructed workflow and presenting simulations results report.	BPM engine
3.	Role-based routing	Possibility of creation of tasks prescribed to users according to roles, not user names.	BPM engine
4.	Workflow metrics and monitoring	Separate tools and interfaces for workflow instances execution and workflow engine work monitoring and reports generation.	BPM engine administration/supporting tools
5.	Conditioning routing	Possibility to define a complex condition (triggers, actions, external events and etc.) for task routing.	BPM engine
6.	Workflow statistics	The possibility to generate detail workflow instance execution report, including actors, information, history logs, current status of instance and graphical workflow map simulation photo for current status representation.	BPM user interface and BPM engine
7.	Automation agents for using third party applications in workflow	Defined interfaces for third party application integration into workflow, usually used for realising inter-enterprise workflows.	BPM integration tools
8.	Automatic workflow installation and maintenance	Corporate installation wizards must be used for installing workflow as for maintenance procedures execution, when common user dialogs are presented to user with informative instructions of execution.	
9.	Seamless interface with databases	Technological interface for the insurance of the communication between databases and business rules engines.	BPM integration tools
10.	Task Queues	The part of the business engine, which is responsible for the evaluation and distribution of tasks to the executors according to the defined business rules.	BPM engine, BPM integration tools
11.	Exception handling	Fully automated engine for identification of executed processes validity (including business and technological aspects) and non-defined/unexpected real-world situations.	BPM engine, BPM exception handling engine

Table 3. Specific functions and features of scientific workflow technology.

No.	Feature name	Feature description	Tools
1.	Verification function	The process of evaluating a system or component to determine whether the products of the given development phase satisfy the conditions imposed at the start of that phase.	ExSpect, Woflan, Fluent, COSA, FileNet, Tibco/InConcert
2.	Petri-Net based	A Petri net (also known as a place/transition net or P/T net) is one of several mathematical representations of discrete distributed systems. As a modelling language, it graphically depicts the structure of a distributed system as a directed bipartite graph with annotations. As such, a Petri net has place nodes, transition nodes, and directed arcs connecting places with transitions.	GRMS, I-Lab, COSA BPM, ExSpect, FLOWer, Income Suite, Kontinuum, PROTOS, Renew, WoPeD, WFL/Flower, YAWL
3.	Vendor independent	It is the form of application independency, when the industrial standards are used for building it with a goal not to rely on any vendor specific technologies or standards. This makes the application integration independent.	Fluent, Fiswidgets
4.	Process Mining	The term *process mining* is used for the method of distilling a structured process description from a set of real executions [64].	Mostly met in Petri-Net based systems, Chimera, D2K
5.	Soundness property	Property of logic system that every sentence derived from a set of sentences is also a valid consequence of that set of sentences. The ability of the components of an aggregate to withstand the effects of various weathering processes and agents.	WoPeD
6.	Schemas import	This feature is related to the vendor independent feature, as schemas import is the ability to import any semantics workflow schema, with a possibility to convert it into the format, understandable to the vendor independent application.	Woflan, ARIS, BusinessSpecs
7.	Interoperability	Is the ability to transfer and use information in a uniform and efficient manner across multiple organisations and information technology systems. It underpins the level of benefits accruing to enterprises, government and the wider economy through e-commerce.	WorkFlo, FloWare, CrossFlow, JOpera for Eclipse
8.	Lifecycle inheritance	In computer science, the term inheritance may be	Woflan, ARIS,

		applied to a variety of situations in which certain characteristics are passed on from one context to another. The term originates with the biological concept of a parent passing on certain traits to a child.	BusinessSpecs, Staffware, META Software Workflow
9.	Ontology support	Ontology resembles faceted taxonomies but use richer semantic relationships among terms and attributes, as well as strict rules about how to specify terms and relationships. Because ontology does more than just control a vocabulary, they are thought of as knowledge representation. The oft-quoted definition of ontology is "the specification of one's conceptualization of a knowledge domain."	METEOR

Existing workflow exception management tools are often related to the verification and validation stages and are more of a scientific nature than of real world situation management. Verification techniques are mainly technical and check the design of communication protocols, multiprocessor systems, and consumer electronics. They are rarely used for business process analysis.

Successful exception handling requires an exception management system with strong verification, validation and performance analysis tools. Woflan is a tool, which can import workflow-process definitions from several workflow management systems and can analyse realistic and complex workflows [42]. Woflan is capable of transforming any workflow management system definition language into Petri nets, localising the exception and fully executing detection and diagnosis procedures.

Also, good exception management systems should have access to knowledge of business processes, their instances, and their historical data. Such systems should also have simulation capabilities. Some of the workflow management systems support simulation or provide gateways to existing simulation tools, like ExSpect, which can simulate COSA workflows; business processing re-engineering tools ARIS and BusinessSpecs can import and export Staffware procedures; and Meta Software's workflow can interface with Visual WorkFlo and FloWare [42].

Another kind of exception management system is related to particular industries and is often ontology-based [43]. An ontology-based exception management system is exhaustive, but it is very tightly related to particular industry context-dynamic exceptions. Such a system usually consists of two steps: ontology-based identification of migration rules (such as profile change and workflow condition change, rule change or automatic exception handling, which is achieved

by comparing actual and expected input and output data) and the application of migration rules (which can consist of compensate, redo, order, insert or delete operations).

A third type of existing exception management systems is constructed using an integrated perspective of exceptions and exception handling, considering organisational levels, which are affected by exceptions, and respective human roles in recovery. Recovery is usually done using a set of WfMS components to help and support these roles [24, 28]. This kind of exception management system can be used for the real-world business process exception handling.

Table 4. Scientific workflow tools features evaluation.

Nr.	System	Description	Advantages	Limitations
1.	CrossFlow [78]	Aims at providing high-level support for workflows in dynamically formed virtual organizations. High-level support is obtained by abstracting services and offering advanced cooperation support. Virtual organizations are dynamically formed by contract-based matchmaking between service providers and consumers. CrossFlow developments are driven by requirements from real-world scenarios.	The architecture is based on commercial workflow management system technology, shielded from the CrossFlow technology by an interface layer. In the project, IBM's MQSeries Workflow (formerly FlowMark) workflow product is used.	Is designed for contract management and is meant to connect existing inter-organisational commercial products.
2.	Adome [13]	ADOME (Advanced Object Modelling Environment) is an active OODBMS extended with role facilities, built by integrating ITASCA (OODB) and CLIPS (inference engine). It provides a good supporting environment for the dynamic features required for a WFMS, especially for online exception handling, which can be classified along the three dimensions: exception source, exception type, exception handling mode.	In ADOME-WFMS, exception handlers can be procedural (extra branches of existing activity decomposition for exception handling) or declarative (in the form of ECA rules that can be specified within the scope of different activity and sub-activity levels). Both of them can be added, deleted and modified during activity definition time before execution, or during exception occurrence at run-time.	Mainly dedicated to handle change, workflow and schema evolution, but lacks unpredictable and unexpected workflows management features.
3.	Mariflow [84]	In particular, the aim of the project is to develop and adaptable	A prototype of architecture for automating and monitoring the	Mainly developed for usage in marine

		workflow management engine through which the activities of the participants in the maritime industry can be harmonized, combined, and expended through better tracking functional dependencies and documents, improved data access and handling, and lower administrative overheads.	flow of control and data over the Internet among different organizations. This is "electronic medium", capable of delivering value-added services to the participants, encompasses many different technological areas: from communication to security, databases, transaction support and agents.	organisations, shipyards and their supply management chain support.
4.	Mentor	The Mentor project aims to reconcile a rigorous workflow specification method with distributed middleware architecture as a step towards enterprise-wide solutions. A first prototype of Mentor has been built which allows executing specifications in a distributed manner.	Mentorlite provides advanced workflow features like work list management, history management, and monitoring as extensions, implemented as workflows on top of this core. The project uses the formalism of state and activity charts and a commercial tool, Statemate, for workflow specification. Addresses fault tolerance issues.	Has limitation of current monolithic, heavyweight workflow architectures.
5.	WorCraft	Objective: Conception and realization of a development and management environment for business processes.	A special design phase for workflow patterns extends in addition the present process of development by the description of aspects relevant for execution independent of WFMS, the check on contradictions, ambiguities and mistakes with the specification as well as production and assessment of different alternatives to the commercial process support.	Designed for specific industry requirements.
6.	WorCos	Objective: Design and implementation of a novel architecture for workflow management systems.	The WorCOS project demonstrates how a fully distributed workflow management system can be realized in harmony with the Object Management Architecture. That is not only exploitation of CORBA service,	New paradigm of viewing workflow is presented, which will have support for fine-grained calls of object-services, business objects and arbitrary types of other services

			but also integration with the Meta-Object Facility.	in the network. Currently is not supported widely.
7.	FreeFlow [81]	The Xerox FreeFlow Workflow Model provides the framework for describing key processes within an entire production workflow - from order entry through fulfilment.	Uses workflow to create connected business process among clients and transforms these processes in order to optimise process execution and support.	Dedicated strictly to printing processes management across client with some features of CRM.
8.	METEOR [71]	The objective of the original METEOR (Managing End-To-End Application) project at Bellcore was to facilitate and support the automation of enterprise-wide operations (workflows) that execute on a heterogeneous, autonomous and distributed operating environment.	Offers a well-defined model and an intermediate language for specifying the workflows and the tasks, a compiler for the language, and a run-time enactment system that provides functionality to monitor the progress of the workflow execution, enforce the inter-task dependencies and manage the execution of each individual task.	Mainly developed for medical sector, and has ontology based encapsulated exception management for predefined cases.
9.	TRAMs [88]	The aim of the TRAMs project (Transactions and Active Database Mechanisms for Workflow Management) is to develop a workflow management system (WFMS) supporting the modelling and enactment of processes.	TRAMs is on enactment of workflows using advanced database technology, i.e. the controlled execution of workflow specifications. Support graphical modelling of workflows, simulation and planning, evolution of process models even for enacted workflows, reusability of "old" process models and agent implementations.	Has all prerequisites but still lacks inter-operability features and exception management instruments.
10.	PantaRhei	PANTARHEI is a special purpose description language for specifying business processes (called WDL: workflow description language). Specifications done with WDL can then be transformed automatically in a set of rules which provides the required functionality.	Extend the functionality of conventional ("passive") databases with a rule mechanism. After creating rules in the active database, the system can coordinate and schedule the specified processes directly.	As processes are specified only WDL in real world situations it can lacks process integration features.

11.	WASA	WASA2 is an application that supports the creation and execution of workflows from CORBA components.	It includes a GUI workflow modeller as well as controls to modify a workflow while it is running. WASA2 has been used in the domains of geo-processing and molecular biology as well as for modelling business processes.	WASA2 uses a strictly object orient approach where workflows are represented as CORBA objects and can be displayed as UML diagrams.
12.	MOBILE	Supporting workflow enactment.	Mobile is a workflow model defined in terms of abstract data types. It includes detailed models from five major perspectives: functional, behavioural, organizational, informational and operational, as well as brief descriptions of some optional perspectives.	Developed for mobile applications only.
13.	Polyflow	Supporting workflow enactment, designed as a goal-based workflow system that uses the power of goal-based knowledge representation to assist in planning, monitoring, and repairing workflows with emphasis on the users and their workgroups.	Polyflow employs a goal-based approach to modelling workflows. Planning techniques from AI research are employed to handle exceptions.	Mostly suitable for simple office procedure automation.
14.	OVAL [86]	The Opel Virtual Aero Lab (OVAL), an automated workflow tool that combines commercial and free software, has been created to speed up the numerical simulation process. The tool provides a better understanding of the flow field. Aerodynamic optimization through simulation can be realized.	A detailed and complex tool covering all simulation steps analysis, performance and result mining, gives better understanding of flow field. Tool was developed using modern programming languages, has instruments for error checking.	Was produced mainly for OPEN factory production testing and expected results simulation.

The next table gives the results of comparing study, described in detail in [62] on vendor workflow systems, how they realise most spread and needed for workflow patterns. The table is supplemented with Oracle and EMC Documentum product evaluation, as at the moment of comparing they were not included into the table. According to market surveys the leader places of workflow management systems is shared by EMC Documentum and IBM, while production workflow leader place is taken by SAP and Oracle workflows.

Table 5. Commercial workflow management tools evaluation/comparison [91, 62].

Pattern	Staffware	COSA	In Concert	Eastman	FLOWer	Domino	Meteor	Mobile	MQSeries	Forte	Verve	Oracle	EMC Documentum	I_Flows	SAP R/3
Sequence	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Parallel Split	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Synchronisation	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Exclusive Choice	+	+	+/-	+	+	+	+	+	+	+	+	+	+	+	+
Simple Merge	+	+	+/-	+	+	+	+	+	+	+	+	+	+	+	+
Multi Choice	-	+	+/-	+/-	-	+	+	+	+	+	+	+	+	+	+
Synchronising Merge	-	+/-	+	+	-	+	-	-	+	+	+	+	+	+	+
Multi merge	-	-	-	+	+/-	+/-	+	-	-	+	+	-	+	-	-
Discriminator	-	-	-	+	+/-	-	+/-	+	-	+	+	-	+	-	+
Arbitrary cycles	+	+	-	+	-	+	+	-	-	+	+	+/-	+	+	-
Implicit termination	+	-	+	+	-	+	-	-	+	-	-	+/-	+	-	-
MI without synchronisation	-	+/-	-	+	+	+/-	+	-	-	+	+	+	+	+	-
MI with Priori design time knowledge	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
MI with a Priori runtime knowledge	-	-	-	-	+	-	-	-	+/-	-	-	-	+/-	-	+/-
MI without a Priori runtime knowledge	-	-	-	-	+	-	-	-	-	-	-	-	+/-	-	-
Differed choice	-	+	-	-	+/-	-	-	-	-	-	-	-	+	-	-
Interleaved parallel routing	-	+	-	-	+/-	-	-	+	-	-	-	-	+/-	-	-
Milestone	-	+	-	-	+/-	-	-	-	-	-	-	+	+	-	-
Cancel activity	+	+	-	-	+/-	-	-	-	-	-	-	+/	+	-	+
Cancel case	-	-	-	-	+/-	+	-	-	-	+	+	+/	+	-	+

1.4 Business Process Exceptions Classification

The analysis of this section was mostly produced inspired by the works and researches of such scientists as Zong Wei Luo, Wil van der Aalst, Crisantos Dellarocas and Mark Clein, Fabio Casati, H. Saastamonen.

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 into account 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.

That leads to the assumption that change in business processes is vital to its goal. The change itself can raise an exception during the execution of business process and in so doing delay the finish of the business process and raise many trouble for reaching the final business goal.

Therefore, while making exception classification, it is important to exclude change and failures, which are of other nature than exception. The definition of failure, change and exception should make the clear distinction between these three phenomenons's, which from the first look are quite related.

Definition 8 *Change – is dealing with unanticipated events, resulting from an incomplete specification and handling modification of specification because of changing conditions [61].*

Taking into account the importance of change, the constructed workflows must take into account the five perspectives of workflow itself:

- Process perspective,
- Organisation perspective,
- Information perspective,
- Operation perspective,
- Integration perspective [61].

Every of these perspectives can raise or be related to a change, as it usually is a result of the outside factors (like business goal changes, legal changes, economic changes and etc.). Sometimes a change can impact some of the perspectives at one time. For example, reorganisation of company can impact the organisation and process perspectives (sometimes even integration perspective) as the existing workflows must correspond and mirror the change.

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, implemented and running workflow processes are rather difficult to support and are usually not tolerant of any kind of exceptions and changes. They can be very diverse, with some of them being predictable and addressable in advance, some unpredictable. Unpredictable changes and exceptions can change processes radically, decreasing the processes' chances of delivering the desired business outcome [72].

Definition 9 *Business activities, which cannot be executed in a predefined manner and cannot reach their desired outcomes, are referred to as exceptions [50].*

The other related type of undefined and unpredicted situations in workflow, can be identified as failure. The main distinction of failure from exception and change is that failure usually points to particular software element or its fault. Failures are more of hardware and infrastructure (like network or DBMS) nature and usually are handled with a help of standard administration tools. Still there is slight possibility that failure can be business related and impact the final goal of business process. In e-business context it can be encountered only in inter-enterprise transactional cases. Therefore, failures in business perspective can result only in task failures when submitted task fails because of not committed transaction [10]. The open issue is, should we treat the failure further as a failure or specify it as an exception, which mostly depends again from the context of failure and further business goals.

Definition 10 *Failure – refers to the state or condition of not meeting a desirable or intended objective [90].*

In other words, it is the inability of a system or system component to perform a required function within specified limits. Basically, a failure may be produced when a system execution fault is encountered.

Workflow requires specification of both normal process flow and possible variations due to exceptional situations that can be anticipated and monitored [59].

Non-predicted situations can be divided into:

- Change, which can be identified as:
    - Incomplete specification and its change– corresponding to lack of specification details,
    - Outside development change – related to business, legal, technological, design or technical problems,
    - Ad-hoc/momentary changes – individual or selective cases,
    - Dynamic/evolutionary changes – new cases in system [61, 59],

- Failures, which can also be divided into [10]:
  - 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 exceptions, which are [69, 10]:
    - 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 exceptions can be classified as [40]:
    - Useful exceptions – a "key to effective and flexible" processes, usually easily handled;
    - Unanticipated exceptions – the result of an unexpected, infrequent and non-repetitive event.
  - Human nature exceptions – which are the combination of expected and unexpected factors:
    - casual environmental changes,
    - incorrect decisions in business process execution,
    - application usage knowledge,
    - unpredicted turn of business rule execution.

Described above types of 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.

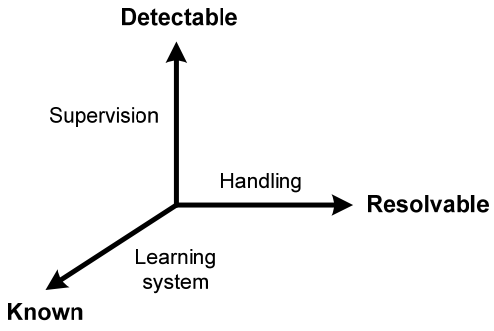
In essence, expected exceptions cannot be called true exceptions, simply because they do not impact the final goal of the business process – a method of handling them is already defined, which allows achieving the final goal.

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 impact the work model, so the final goal is reached, though it can be slightly influenced or changed [40]. Unexpected and unknown exceptions usually occur asynchronously to the flow and 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.

One more thing which very important before starting to handle the exceptions is to specify it. Seeking universal exception handling model it is necessary to dissociate from e-business context and specify the exception only as workflow exception in the first place. Also it is important to notice, that effective exception specification can be reached dividing it into business and technological specifications.

The most basic business workflow exception specification is defined in [72] as a three dimensional graph, which, in part, by identifying the answers, points to the systems elements which are involved in the exception handling. There are three main primary specification unknowns (see Picture 4):

- Known – mostly unknown of business knowledge nature, which judge upon the time and resources needed to solve exception,
- Detectable – the main unknown, which helps to fill in the left two unknowns,
- Resolvable – related to business goals, as exception is resolvable only and only when it reaches the raised business goal to the process.



Picture 4. The dimensions for basic exception specification.

Though these dimensions are truly connected with each other and influence each other, while experimenting with real life situations, it is possible to make a conclusion, that there is dependency and sequence connecting these dimensions into one life cycle. Basically, usually

exception is encountered while executing a process. After we encounter it, the most important property of exception is delectability. If we can detect it, then we can classify it as exception and think of its handling ways, which can be reached only through supervision of encountered situation. As soon as we define the handling ways, this exception becomes known and adds knowledge to our business case.

Picture 5. The sequence of exception lifecycle.

Answering these three primary business unknowns of exception, it is possible to specify the exception from a technological point of view. Seeing as exceptions in business processes are context dependent (in technological specification differently from business specification, context must be evaluated), there is good reason to specify them using JECA (Justified Event-Condition-Action) rules. Each JECA rule [r(j, e, c, a)] consist of these four parts [72]:

- Justification (j) – the context where the rule will be performed,
- Event (e) – the event, upon which the rule is triggered,
- Condition (c) – logical constraints, that must be satisfied so that the action in the rule would be taken,
- Action (a) – that should be done if the rule is true.

These rules are prerequisites of exception handling consistency, gathering exception context information, localising exception occurrence in the process, deciding whether an exception is truly an exception, and eventually solving the exception. The latter rule will be used as the basis in the exception handling models.

1.5 Research Object and Tasks

After performing the detailed analysis of business process and exception management literature, and reviewing of business process characteristics, mapping using workflow, exception appearance sources, causes and handling methods, the following tasks are raised in the frame of this research:

1. To perform an analysis on business process modelling methods. While analysing, to evaluate workflow method usage for business process modelling, also summarise the workflow modelling technologies, and classify exceptions in business processes,

2. To analyse in detail the existing business process exception handling models, separately the ones used to handle exception in B2C and B2B environment, also evaluate the possibility of workflow method usage in business process exception handling,
3. After performing the detailed analysis of existing technologies, present B2C exception handling model, which would consist of exception handling algorithm and method realisation architecture discussed in detail,
4. Create B2B exception handling method using workflow technologies by evaluating existing B2B technologies and present B2B exception handling algorithms together with its realisation schemes and architecture descriptions,
5. To summarise research results and to conclude the research.

2 Business Process Exception Models

2.1 Introduction

Business process is a combination of workflow, business rules and organisational structure. Therefore, exceptions occurring in business processes are classified according to these three elements, as it already was noted in previous chapter. That makes business process exception management tools quite unique, as they must face several aspects of non-defined situation while performing detection, diagnosis and handling of exception. Such environment conditions the appearance of several types of models for handling exceptions in business processes. These models can also be built according to the chosen type of modelling techniques used to models the business process itself. There are several of them:

- Petri nets – which map tasks to transitions, resources to places, and cases to tokens,
- Flowcharts – most popular technique in business to show pictorial flow of a process from its beginning to its completion,
- Activity diagrams – which are similar to flowcharts but focus on the flows driven by internal processing as opposed to external events [18].

It is important to have in mind that the business process is analysed in runtime and build time environments, where build time environment provides a modelling language and appropriate design tools, which allow specification of the processes, and runtime environment, which provides necessary services for process automation and monitoring. This leads to the assumption that build time environment is friendlier to exception cases but also is more complex for solving them in runtime. It is because of the build time environment nature. As it was noted above, there exists several ways of constructing business process. All of them generate different process depiction and description results (such as UML diagrams, XML or YAWL sheets). Such results can of initiate the appearance of several exception handling models, as different build time results require different runtime environment interpretation and description.

These sections discusses the exception handling models for runtime and build-time environments, which are achieved using various techniques, which can be based on different formulas usage, description languages and methods.

2.2 Basic Exception Handling Methodology

The basic method for any exception handling is simple, it is a three step method, beginning from exception detection, going through its diagnosis and finishing with handling phase. This

method can be applied in build time and run time, as it is more an algorithm or exception handling process lifecycle which does not point to any particular handling tools but defined basic rules and concepts for its achievement.

Detection phase

As exception can occur in any place of running workflow instance the most critical is to design the failure models, which are described as:

$$F \text{ in } (W, F) \rightarrow E \qquad (1)$$

In this formula (1) F is a failure condition, W is workflow instance and E – is exception handling process). Exception conditions can be identified in three levels [40]:

- Task level – the exceptions which occur in specific tasks,
- Block level – the exceptions which occur in group of workflow tasks,
- Process level – the exceptions which occur in any place of the process at any time of its execution.

Therefore it is mostly important to specify:

$$W \text{ in } (W, F) \rightarrow E \qquad (2)$$

Formula (2) can represent any of these levels but anyway provide sufficient data for the evaluation of exception.

Detection phase always starts from the model of correct behaviour, both for entire system and for each individual component. This model must be drawn such, which achieves business process goal and is considered ideal process model. In case to find an exception in this ideal model one has to have taxonomy of elementary process elements, which can be annotated with failure modes. In [12] this taxonomy consists of exceptions related to goals and assumptions, exceptions related to activities and exceptions related to resources. To find an exact place of an occurred exception, this taxonomy must be broken up into smaller specified units.

To specify the possible exception mode, the system can be instrumented with additional sentinel components, which would look for appropriate patterns in the behaviour of base components [41]. We also can use similar sentinels to collect information during all the process execution time, in order to collect information about its work, so that it would be later used for inspection of occurred exception. Such method would help to step over the detection phase and go straight to the diagnosis phase.

Detection is the most critical phase which can be also achieved by supervising the workflow system components external inputs and output and comparing their behaviour with the specified behaviour of the system. For that several techniques can be used [72]:

- User specified supervision – it is the construction of perfect graph of behaviour which includes all possible branches of behaviour. This technique is called non-deterministic system specification, which advantage is that it defines system behaviour as mandatory, but leaves always a possibility of incompletion,
- Hierarchical supervision – is used when the case is split into part in order individually to examine external and internal behaviour. In such a way, an examination can be started from whole process, moving to task groups and finishing with individual tasks,
- Time supervision – is related to states of workflow and time moment when it must finish. These exceptions can be detected in run time and easily detected the occurrence spot.

Diagnosis phase

After identification of the cause of the exception, the next step is to make a diagnosis of it, deciding what type of handler should be used afterwards. This is "→" in (W, F) → E [40].

This can be achieved through:

- Control flow – when process flow conditions are modelled to diagnose exception. The disadvantage of such method is, that diagnosis can be achieved only in complete process state, thus it limits to the task level of detection phase. But the exception handling process can be modelled as single task, group of tasks or even nested process,
- Exception rules – it is most popular method because of its structural nature. It is commonly used in programming environment and can be defined for any level. Exception rules define the simple formula *if then*, where *if* identifies condition and *then* defines the rule. The rule itself can point to the other exception rule or be a sub-rule of the primary exception rule. This method is based on simple deduction and is mostly reliable when human factor exceptions are eliminated,
- External triggers – when the workflow process itself can not provide enough data for failure condition evaluation of the particular exception case, the exception workflow can be triggered through external triggers. The difference between exception rule and external trigger is that in latter case an exception workflow is instantiated instead of exception sub-process. Usually this workflow is managed by human and does not require system evaluation as the nature of the failure is human factor nature.

Diagnosis phase is used to determine which methods or tools should be used to solve unacceptable situation. In this phase exception specifications can be processed, which are generally made in the detection phase. The results later are used to make the exception handler specification. For the systems efficiency exception handler should be generated by the

autonomous process or system, which could have an exception and exception handlers' knowledge base. In this way, information and solutions are collected all the time and reused if possible in the future. This would shorten exceptions handling time and cost, which are critical for every business process model. Diagnosis phase can be achieved using simulation tools [44]. It would be effective but expensive.

Handling phase

The effectiveness of any exception handling lies on properly designed handlers. This is:

$$E \text{ in } (W, F) \rightarrow E \quad (3)$$

As was noted above, the design of exception handling process can be very unique as it is domain dependent and meets build time requirements. The design can be achieved in various ways and are not that important to the exception as its resuming after completing the handling. Only in one case the design of exception handler is important, it is when the sub-process is invoked and the running workflow instance is suspended till it finishes.

For example, if for the running workflow instance is ***i*** of workflow ***W***, an emerged failure ***f*** $\in$ ***F*** evaluates to ***True***. Also assume that ***e*** $\in$ ***E*** is diagnosed. As instance ***i*** is in a current state of workflow at the time when ***f*** is evaluated, there will exist at least one ***n*** $\in$ ***N***, such that ***node state (n, i)*** will be active state or scheduled state (see Picture 21). The invocation of ***e*** can be predicted if ***i*** was suspended. In that case, ***n*** will be put into suspension state only if it was previously active, or it will be terminated if it was previously scheduled. After finishing ***e***, running ***i*** may not be resumed, it will depend on the state of ***n*** after ***f*** evaluation, which will point to various options [40].

Table 6 The possibilities of instance resuming after exception handling.

Case	Node state (n,i) = Active	Node state (n,i) = Scheduled
A	n is aborted and control never return to i	Control never returns to i
B	Control returns to i and n is reactivated	Control returns to i and n is rescheduled
C	Control returns to i, but execution is resumed at a different point	Control returns to i, but execution is resumed at different point
D	Control is not diverted from n and n executes in parallel with e	

Exception handlers are usually modelled as autonomous processes and are always enacted in the place of exception occurrence, not appearance. These processes can implement three different types of actions: masking, propagation and recording [72]. These actions can be used separately or together. The exception process can ignore the exception, modify the process

according to changed data and record the situation for the future reuse. Also exception process can retry corrupted task, execute appropriate workflow recovery operation or modifications.

In order to successfully complete the handling phase it is necessary that exception handling system would satisfy some essential requirements, such as [72]:

- Support of specification for exception handling – specification of various types of user defined exceptions in advance, in order to catch the unpredicted cases,
- Support task specific exception handling – it is the black box method and does not concern about the realisation of task itself. The only critical data is input and output data and how they will interact with the task,
- Localise exceptions – the most important feature, which must isolate and mask exception preventing it to affect other parts of the system,
- Support exception handling by forward recovery – it is a method of repeating of finishing the process with an aim to discover the missing data and relationships between tasks, which would point to the causes of exception,
- Support human assisted exception handling – the possibility to involve human into handling phase, as not always all situations can be solved automatically due to the unpredictable nature of the internal and external factors.

In generally, exception detection and diagnosis needs a lot of precise and timely information, so this information can be also stored using ontology. It can be used to describe the ideal business process model in the terms of the ontology, so that it clears communication between system developers and domain experts. In the ontology-based case management there are used three phases: abstraction, retrieval and adaptation. Each phase uses different specialized ontology to represent exception case, or to retrieve similar cases or to adapt the existing cases. Use of ontology clarify and frame business process terminology and logic, so that the exception handling can be achieved more precise.

2.3 The Models of Business Process Exceptions in Business-to-Client environment

First of all, before starting handling exceptions, it is necessary to define environments in which the handling of non-defined situations will be performed. In this research, the main focus is on these e-business types: business-to-consumer and business-to-business. The exception models discussed in this section were collected and experienced while using a real e-business application.

Definition 11 *(B2C) B2C application is for serving customer needs as they exchange information, goods and services over the Internet [15].*

B2C is publicly accessible Web site offering goods for sale. In B2C applications users' access needed information simply through the Web pages, accessible through Web browser and Internet access. Any person who wishes to use B2C services needs to provide some general information about them, like name, address, credit card. Most B2C sites also encourages users to register and become members, which provide a possibility to build customer relationships with a customer, provide better service and fulfil customer loyalty functions.

Most usually met types of B2C are these [90]:

- Direct sellers – companies providing products or services directly to customers,
  - E-tailors – shipping a product directly to consumer upon receiving an order (like Amazon.com),
  - Manufacturers – who directly sell to consumer via the Internet, removing the intermediaries (like dell.com),
- Online intermediaries – companies facilitating transactions between buyers and sellers and receiving a percentage of the transactions value (usually brokers and infomediaries):
  - Types of brokers include eTrade (buy/sell fulfilment), virtual mall (like Yahoo), metamediary (like Amazon zShops), bounty, which offers a fee to locate a person, place or idea, search agent, helping a customer to compare different stores (like MySimon),
  - Types of infomediares include advertising based models, using high-traffic and niche practices, community based models, which is a community based system, fee-based models, when a person has to pay for viewing content.

B2C Web site main components are these [16]:

- User profile – maintaining information about user. This profile is used for marketing purposes and serves as user environment personalisation profile,
- Product catalogue – organised in logical groups electronic product catalogue used for product search by users,
- Shopping flow – it is the process where customers browse the catalogue, select products and purchase them (usually accomplished as workflow),
- Shopping cart – widely used term in around Web representing an online order basket.

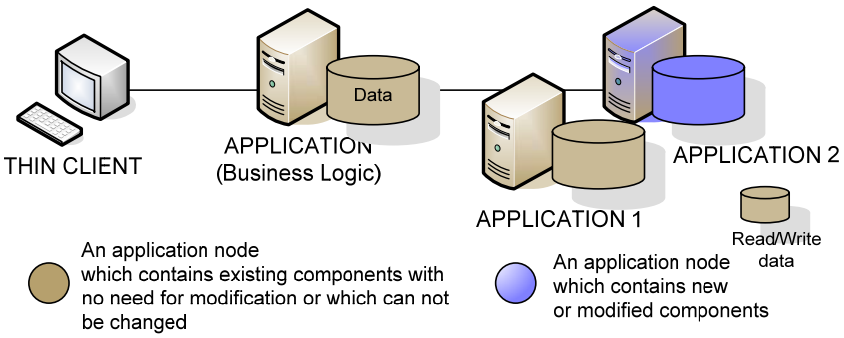
B2C applications are for consumers only and based around the Internet. The consumers are outside the organization so they are using organization services, which can be goods or information.

B2C applications are frequently modelled using workflow because these tasks are frequent and accomplished according to set of rules and input data. Workflow makes it possible to process raised process instance quickly, using correct data and present user with the correct and effective solution.

Business process workflow always has an opportunity to extend. There is always a possibility to change the chain of tasks and activities or add a sub-process into existing process. But that applies only to the future business process instances. As for the old instances, they must be left as they are, or they may be altered using other workflow processes, which alters old instance to meet new instance requirements and supply with necessary data.

B2C and B2B serve the same purpose – to make the business more successful, still they have slight differences which point to different projections of them on technologies [49, 53].

B2C applications are accessed through the thin client (usually internet page) and user can directly reach needed information from the business logics application (see Picture 6). This business logic application can present information to the client/user from different applications, which information transmission is processed in the backend of the system. Due to the information demand, it can be transmitted to the business logic application constantly/online or use time to time synchronisation mode. In such cases all participating applications have their own databases for temporal or constant data storage.

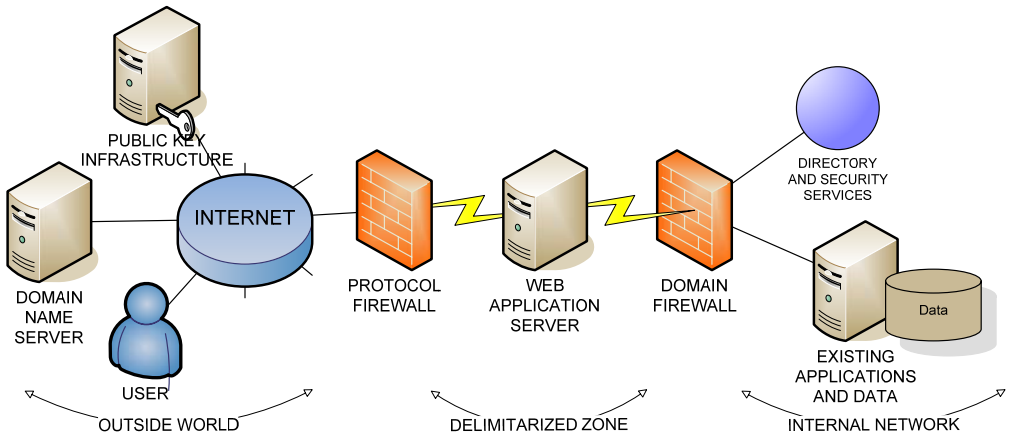


Picture 6. Business to Consumer model conception [15].

Before speaking of the key problems, rising in B2C, it is necessary to get acquainted with applications architecture.

B2C application architecture does not differ from simple client-server application architecture; where client is internet user, connecting to the system thought the web using secure connection. As B2C applications can involve third party applications for sensitive data exchange

(like internet payments using transactions to bank systems), the customer in this case can use various security features for protecting its personal data (like PKI). The e-business application usually resides on company side and presents needed information and services thought the web application server, which should reside in demilitarised zone, with a purpose to protect company's applications from intruders.



Picture 7. Business-to-Consumer Application Framework for e-business architecture [15].

Picture 7 represents run time B2C application framework, which consists of three main parts [15]:

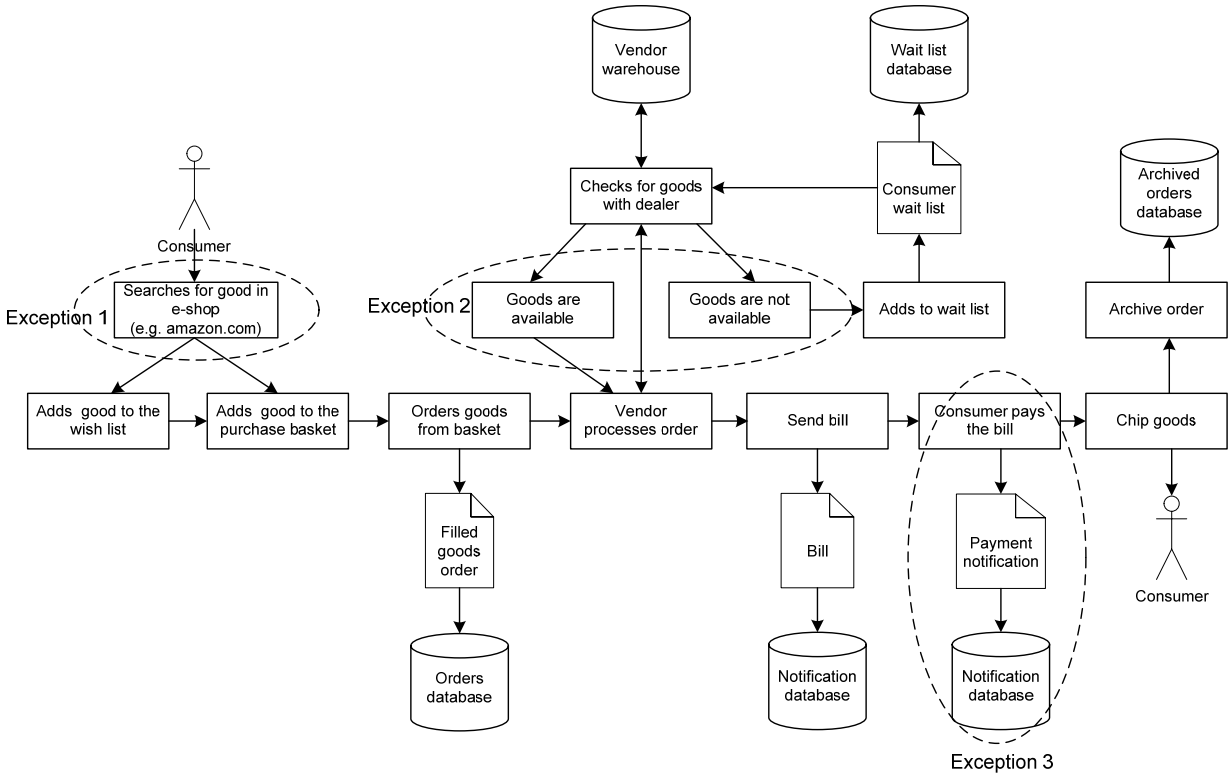
- Outside world – customer environment, which is accessible through the Internet, which can include and use PKI infrastructure for certified user identification and user domain name servers, which provide information necessary for e-mailing and is essential for Internet usage,
- DMZ – is a protected area containing B2C Web application server, where Web server and application servers are running and which contain business logic implementation,
- Internal Network – stores user information, needed for authentification and authorisation, which is stored in the directory and security services node behind the firewall. Also the data accessed from business logic is behind the domain firewall in the internal network.

B2C environment exception models

As it was already noted above, B2C environment is dedicated to Web purchases, directly involving a customer as a physical person. Therefore, it is necessary to define the most often met exception models, which will later be analysed for handling construction purposes.

The exceptional situations must be grouped according to two their main lifecycle stages [12, 10, 61]:

- Build-time – exceptions related to process construction, validation and verification:
  - Exceptions related to goals and assumptions: workflow exceptions,
  - Exceptions related to activities: workflow and data exceptions,
- Run-time - exceptions related to process enactment, management and processing, including monitoring and administration issues:
  - Exceptions related to activities: workflow, data and temporal exceptions,
  - Exceptions related to resources: external exceptions.



Picture 8. Purchase order workflow.

While identifying these types of exceptions, unpredicted cases should answer main dimensions of exceptionality taxonomy [39]:

- Exceptionality – difference between an exception and a normal event based on rules,
- Handling delay – the time between the appearance of an exception and when it can be handled,
- Amount of work – the amount of extra work caused by an exception when compared to a normal event,
- Organisational influence – the number of people the exception involves,
- Cause – the reason for an exception,

- Rule impact – the change an exception causes to an organisation rules.

For visualisation of main exception models a purchase order execution workflow schema was created and will be used, which is described in Picture 8.

The picture identifies three spots in workflow, which are possible change and unexpected exception sources. The main amount of exceptions in B2C applications can occur in Exception 1 (see Picture 8) spot, as it is related to good performance of the system to the user. Suppose the consumer fills an order for an item which is the last in store, but obviously the consumer can not know that. By the same time the other consumer can try to order the same last item. If the system performance on recalculating the remainder of the product is slow it can result in a situation that one of the consumers will not be shipped the selected product though he was expecting it. Obviously if such situation repeats quite often in the same online shop, this shop can loose many clients due to inefficiency of the performance.

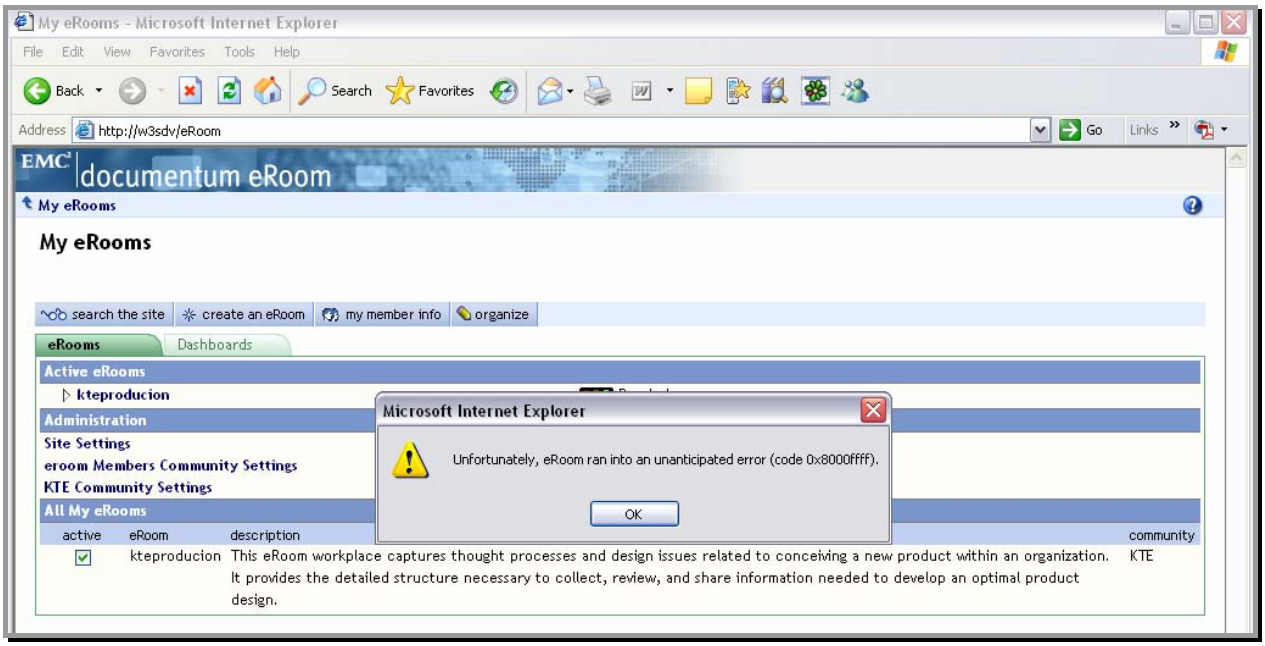
The described situation can lead to the dynamic change, which result would be to optimise the purchase ordering form running data refreshing capabilities and result in workflow exception.

The second example, Exception 2 (see Picture 8), is related to more complicated issue; when the situation completely relies on real world situations. Assume the purchase order was filled for several items and business application starts collecting items from the store or inquire for them from the vendors. Here can happen that some items can not be shipped at the promised time because these items will be available only after the shipment deadline. In this situation, first of all it is necessary to identify business case change. It is a change as the system itself is working without exceptions and the change is a result of factor of outside the system. It is an ad-hoc change, which can actually result in a temporal or external exception.

There are some ways how to deal with this possible exception. One of them is to slightly change the delivery phase and deliver the items in two packages: one of them in time with available goods and the other later without any shipment tax. In this situation company has some losses but retains customer's loyalty and trust. The other way is to delay shipment and ship goods in one package. This situation is very unreliable as the company risks customers' loyalty and trust twice: (1) delaying the shipment and (2) risking not shipping the delayed items at all, as it will turn out that this good will not be available at all. In that way, it is very possible that company will loose a customer.

Evaluating the possible result of the described situation we can ascribe this change to the exception, which should be managed in an intelligent way. The goods order form should have extra procedure which would check for availability of goods already at order submission time.

The third situation, Exception 3 (see Picture 8), is rather unpredicted exception than change as it more depends on reliability of the third parties. The payment submission, verification and confirmation functions are usually executed by banking systems and only exchanging data with business application. In this situation there are possible data exchange deviations (when data is submitted too late) or exception (when data is lost during submission). These situations are difficult to support and handle as they involve inter-organisational relationship management. In such situations the workflow management system should initiate the data transmission checking procedure from the beginning of the transmission to its end and generate the report. This procedure should be included into the working workflow model as a part of it and no longer correspond to the unexpected exceptions.



Picture 9. The example of exceptional situation in B2C application usage.

Picture 9 represents the unexpected situation, which occurred during the search of goods in application. As the system identifies itself, it is an unanticipated error occurred in operational level, which means that it is actually a failure, according to integrated perspective approach, described in [28]. Usually such failures can be handled using traditional transaction processing techniques without any human interventions. Unfortunately, in described situation the system itself could not handle this deviation. In such case, the problem is propagated to the tactical (application or workflow) level and failure becomes an external exception. The human intervention is necessary as the changes in application procedures must be achieved.

2.4 The Models of Business Process Exceptions in Business-to-Business environment

In B2B, the special internet applications are build for business partners, which gives special and selected information for every partner separately (sometimes such applications can even be installed as local/offline systems at partner place).

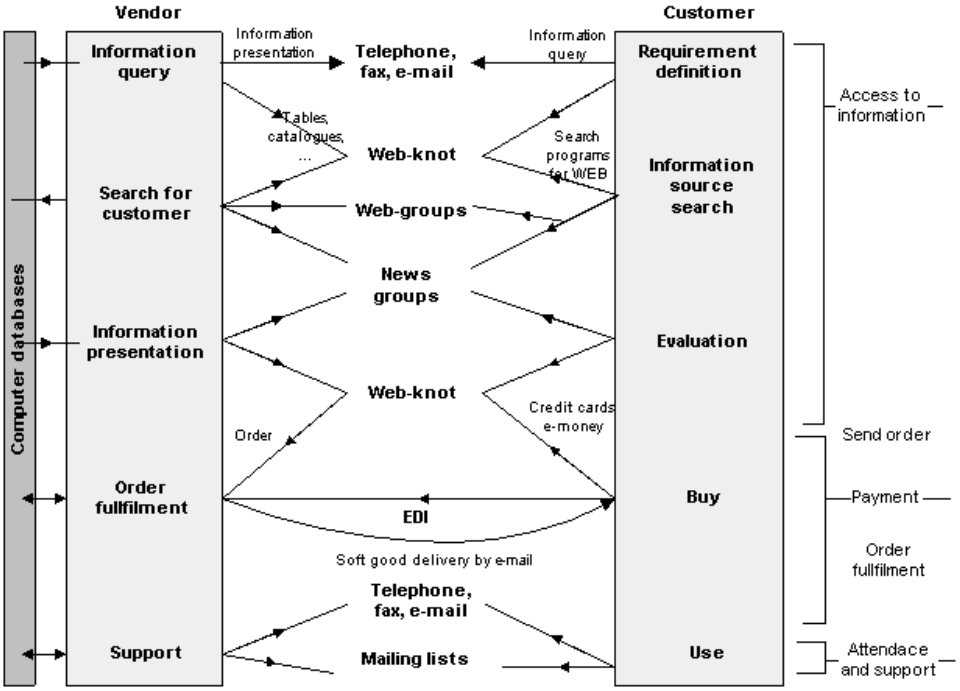
Definition 12 *(B2B) B2B application focus on using the Internet and/or extranet to improve business-to-business partnerships and transform inter-organisational relationships [30].*

Typically, B2B is constructed with a help of automated processes which connect trading partners and is performed in much higher volumes than B2C applications. Besides pure purchases B2B also involves marketing activities between businesses and to identify sales transactions between them.

Picture 10 identifies all possible activities and their interaction concerning customer and vendor. As it is obvious, vendor's main responsibility is information provision to all possible channels suitable for customer. While customer performs most e-business process tasks, starting accessing the information, sending an order, performing the payments and finalising with support tasks, when the most data is also provided by the customer. The picture clearly shows, that the main actor in e-business process is the customer, and the vendor is the one, who helps the customer to achieve its goals using commercial, marketing and customer relationship management traditions.

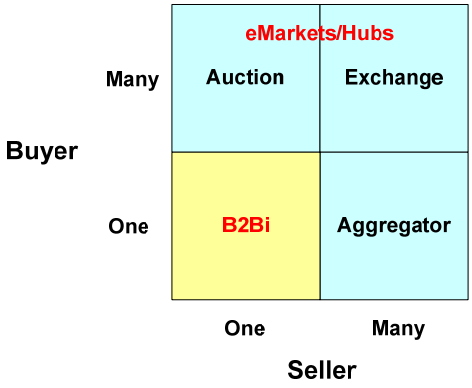
B2B is a sub-class of e-business inter-business class and describes two styles of such business [30]:

- eMarketPlace – a relationship one-to-many, many-to-many and many-to-one between buyer and seller, sometimes also noted as B2M2B. The buying function in such application can be performed programmatically or online. This type of applications integrates B2B and market specifics using internet technologies (example Chemdex, VerticalNet).
- B2Bi – usually is one-to-one relationship between buyer and seller and covers programmatic links between arms length businesses. Usually, it is supply/chain applications and initially was EDI application, but was replaced by far advanced technologies because of cost and inflexible structure limitations (for example GNT.lt).



Picture 10. E-business and business process [26].

B2B is based on dynamic business process integration, which definition incorporates common language. This language is used by specific partners as means of interaction, and electronic contract empowers this language to enforce the interaction protocols which will perform business. Finally, it is obvious that B2B is a convergence of A2A (application-to-application), EAI and B2Bi.

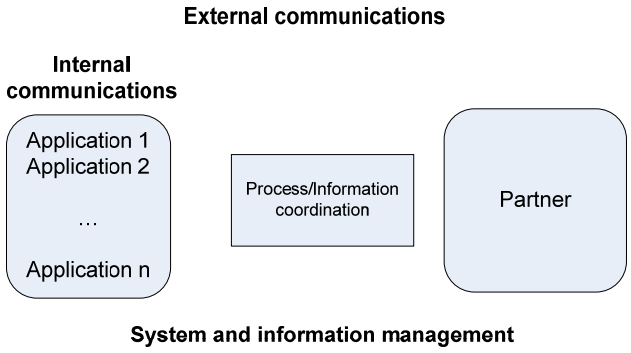


Picture 11. Separation between Markets and Hubs.

The components composing B2B solution are these, the graphic description of which is presented in Picture 12 [30]:

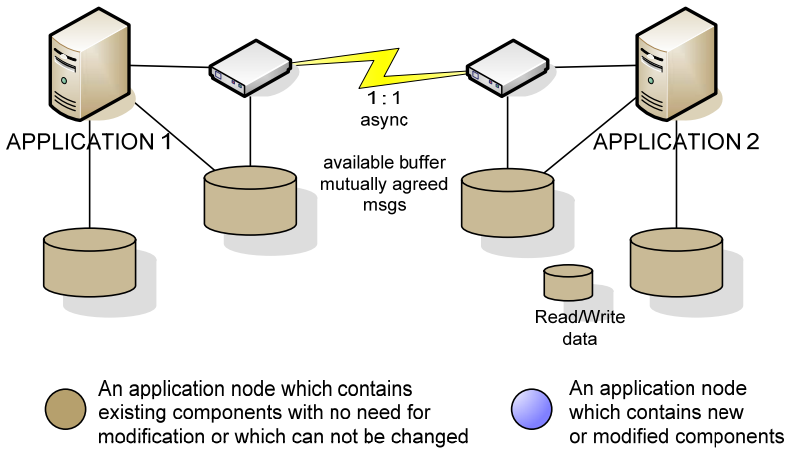
- External communications – communication between trading external partners having different requirements,

- Internal communications – meant for the organisations with existing infrastructure. Here a mechanism to communicate with existing applications is a requirement and usually very similar to EAI,
- Process/information coordination – it is the heart of logic of interaction between internal and external communications, in other words, it is information brokering function,
- System and information management – it is a critical success factor as solution require managing several information components, including partner profile information, data and process definitions, communications and security settings and user information.



Picture 12. B2B solution components.

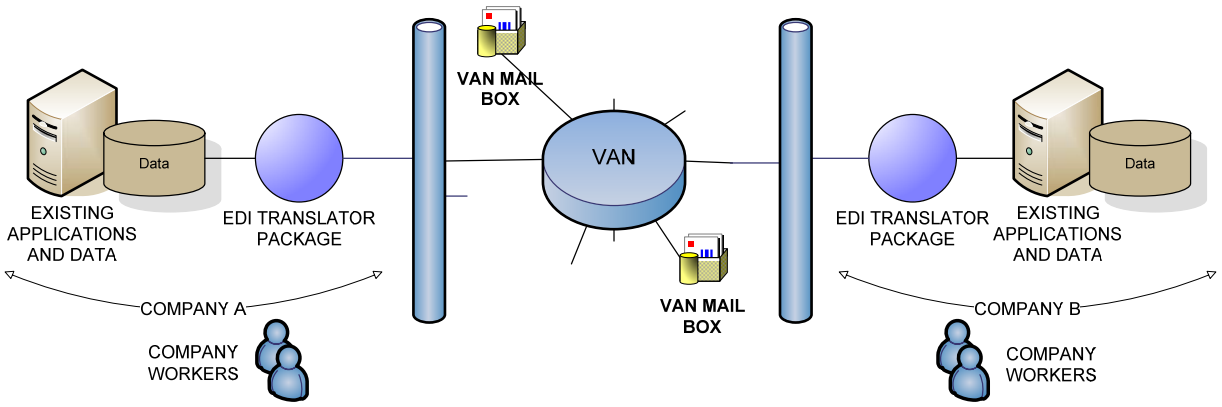
Speaking of B2B applications, information is reached through the application on both nodes (partner and company/consumer) (Picture 13). These applications can be used using Internet connection or be installed as separate applications on the partner computer. In such case, information needed for offline using must be stored on local disk (or data storage device) and transmitted for merging from time to time to the main company/consumer server.



Picture 13. Business-to-Business model conception.

During the replication of this data, information is checked for validity and uniqueness, so that replication data would be completely new or changed with a purpose to minimise the replication time and amount and to lessen network loads.

One of the main differences between B2C and B2B are purchase cost, frequency and number of customers. These factors are more of human factor nature and points to real world/ad-hoc decision making subtlety. B2B is of wholesale nature so here we have cost and client number as main business factors, which dictate business demands and supply. B2C is simple one person shopping, who concentrates on small items and performs it very often. In this case, cost is not always the risk factor, but quality, satisfaction or even delivery time.



Picture 14. Business-to-Business Application Framework for e-business architecture [15].

B2B application framework includes also three or more communicating spaces which include:

- Each of companies have their own applications and data, which actually can be referred as legacy systems and must be integrated into B2B solution using EDI translation package, which function is to convert EDI transaction sets to and from flat files into a format needed by the enterprise applications. The legacy applications also provide security and directory services, needed for successful internet connection, queue manager is required also, for request prioritization and management according to business rules using existing application data,
- VAN is a space where users send and receive messages using mailboxes containing service requests,
- Each of companies is protected by their own domain firewalls and protocol firewalls. The security also can be enhanced using VPN connection which require addition credentials during connection time and limit the chances to intruders,

- The VAN is surrounded by each company DMZ, which is like a threshold to insecure public network,
- Domain name servers and PKI infrastructure resides on Internet between protocol firewalls as it must be reachable for both communicating companies.

B2B applications transform inter-organizational relationships and are dedicated for partnership. They can be used to exchange information between businesses and for collaboration, sharing information and knowledge between businesses for mutual benefit [15]. In other words, B2B application architecture is more of external communication mechanism nature. As in this case information is exchanged between two similar systems which serve the same purpose and are of inter-organisational relationship type. They exchange information using messages, which carries needed data for process execution. B2B applications also use secure connection to the main business application server and usually does not limit with encrypted usernames and passwords (usually e-signature or e-certificate technologies are used).

It is also necessary to point that the main B2B exception lot rises from inter-organisational relationships and deals with different manufacturer workflow management systems, which exchanges information using only messages. In B2C the main exception source, rising from architecture level, is network access performance and application data refresh problems.

Table 7 Main B2C and B2B distinctions.

Involved parties	**B2C (business – to – consumer)**	**B2B (business – to - business)**
Buyer/Seller Relation	1 – 1	1 - many
Product Source	Non – Intermediary - Based	Non – Intermediary - Based
Evolution Level	Publishing	Triggering
Functional Integration	Single Function, no integration	Multiple, partially integrated functions

Table 1 depicts main factors, which influence the used technologies and identify system differences. As it can be seen, main differences are in information presentation and gathering points [27].

B2B environment exception models

B2B models correspond to multiple relationships between multiple various actors. Most common scenarios, as it was already discussed, are marketplaces and supply chain applications. One more branch of B2B must be mentioned in this place, it is e-banking systems (like https://ebankas.seb.lt). Although, the end-user of such system usually is a person, he/she can

represent either own or company's interests. The application can be prescribed to many-to-many relationships type, as it acts as mediator between a bank client and various institutions, to whom the banking application delivers payments and funds transfers. The process example of such B2B possible exceptions scenario will be discussed later.

The most common exception models met in B2B applications should be also grouped according to main lifecycle stages and three classes of exceptions according to [74]:

- Build-time – exceptions related to process orchestration and choreography (see Definition 13 and Definition 14) and exceptions related to business logic,
- Run-time – non-predicted situations should be treated as failures and exceptions, as especially in this e-business type, there is only slim line between these two definitions. The possible failures and confirmed exceptions should be classified as:
    - Workflow – related to workflow engine and business logic exceptions, which also can include workflow management system and human nature exceptions,
    - Application – inter-operability or cross-organisational nature exceptions,
    - Infrastructure – totally related to public usage software and hardware, capable to raise non-desirable deviations from ideal working expectations.

Definition 13 *Process orchestration – Orchestration tends to imply a single co-ordinating force, whereas choreography also applies to shared co-ordination across multiple autonomous systems [83].*

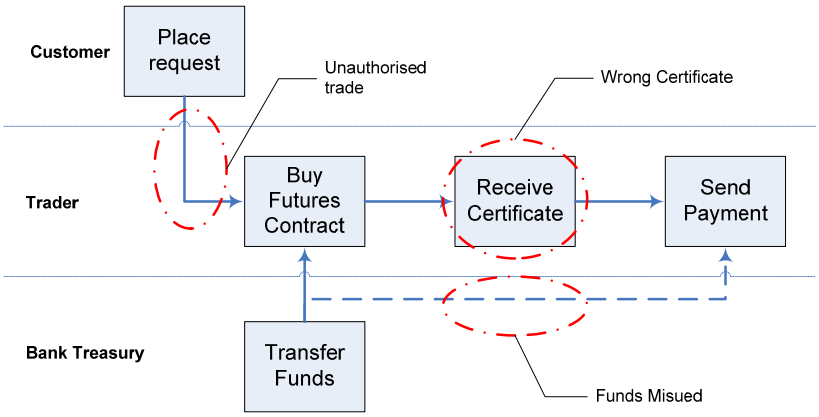
Definition 14 *Process choreography - Overlapping with the related concept of orchestration, choreography directs and manages the on-demand assembly of multiple component services, to create a composite application or business process [83].*

Unfortunately, the exceptionality dimensions discussed in B2C exception models are not quite appropriate for B2B exception evaluation cases as they lack complexity ant B2B anatomy specifics. Therefore, B2B exceptions should be evaluated first of using JECA (described 1.4 Business Process Exceptions Classification) and if a formula returns exception, it should be useful to evaluate it, first of all, as a change. Such evaluation would answer many questions about the nature of exception and lead to its source and handling methods. According to [61], the change should answer these main questions:

- The reason of change – does it change business, technological or legal context and what triggers initiated exception (internal or external),
- What will be the effect of change – momentary change, affecting only current case, or evolutionary change, affecting the whose process,

- Which perspectives could be effected – process, organisational, information, operation or integration. The most critical affect would be on integration as it acts as insurance of flexible and continuous behaviour,
- What kind of change would it be – extension, reducing, replacement or re-linking of the process,
- When the change would be allowed – entry time, right after the commitment of change, or on the fly – at any point,
- What must be done with existing cases – the main question how to solve the unpredicted case, what type of handling method to apply: recovery, proceeding (handling old cases with old logic and new ones with new logic), transfer (migrate the old cases to new ones) or detour, using temporary solutions.

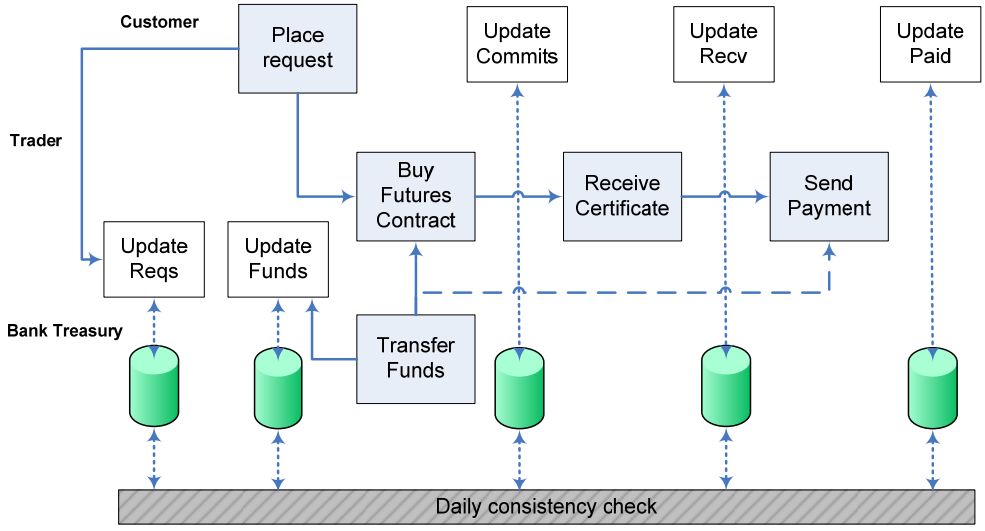
A common build-time exception example will be provided using the schema described in [12]. The Barings bank of more than 200 years old went bankrupt because of simple undetected exceptions in bank's secondary processes. One of them representing the simplified future trading process is described in Picture 15. The three, possible to take the responsibility of the bankrupt, exceptions are: unauthorised trading, unwanted security receipts and unnecessary payments, which should exactly match the funds forwarded to the trader by the bank headquarters. It is obvious, that these three critical exceptions lacked just a simple verification function and timely system information renewal.



Picture 15. Trading process with associated exceptions [12].

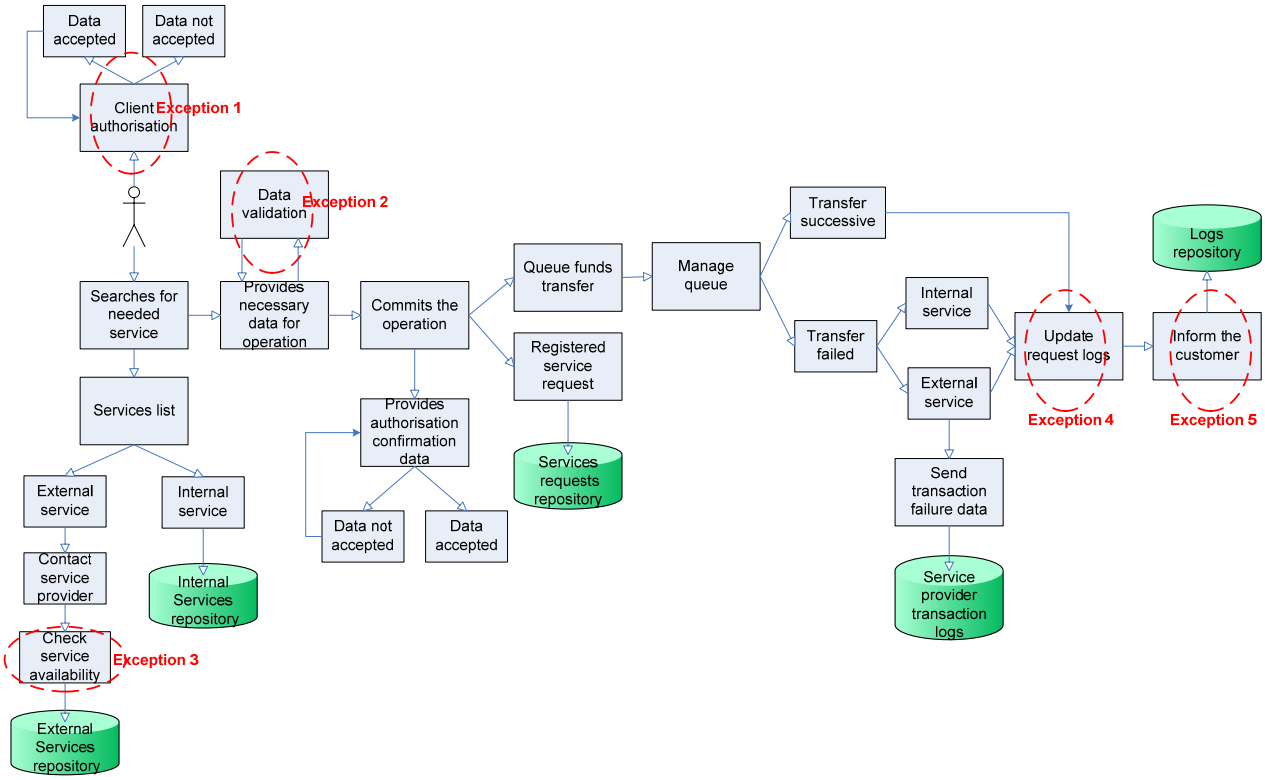
The upper picture (see Picture 16) describes the process model with a properly instrumented logging. One the most critical functions which were missed in previous process description, was information logging feature, which should be performed in every step. The other lack of business logic was transferred funds comparison against funds used for client transaction, which actually was far greater than should. And the last and very important exception, which is of build-time

nature, was that the bank failed to protect the application from violations by giving too many privileges to the employees.



Picture 16. Trading process properly instrumented with logging processes [12].

The examples of run-time exceptions can be defined using simple e-banking process, which covers the main e-banking production functions. The following picture was created for better depiction and analysis of e-banking process.



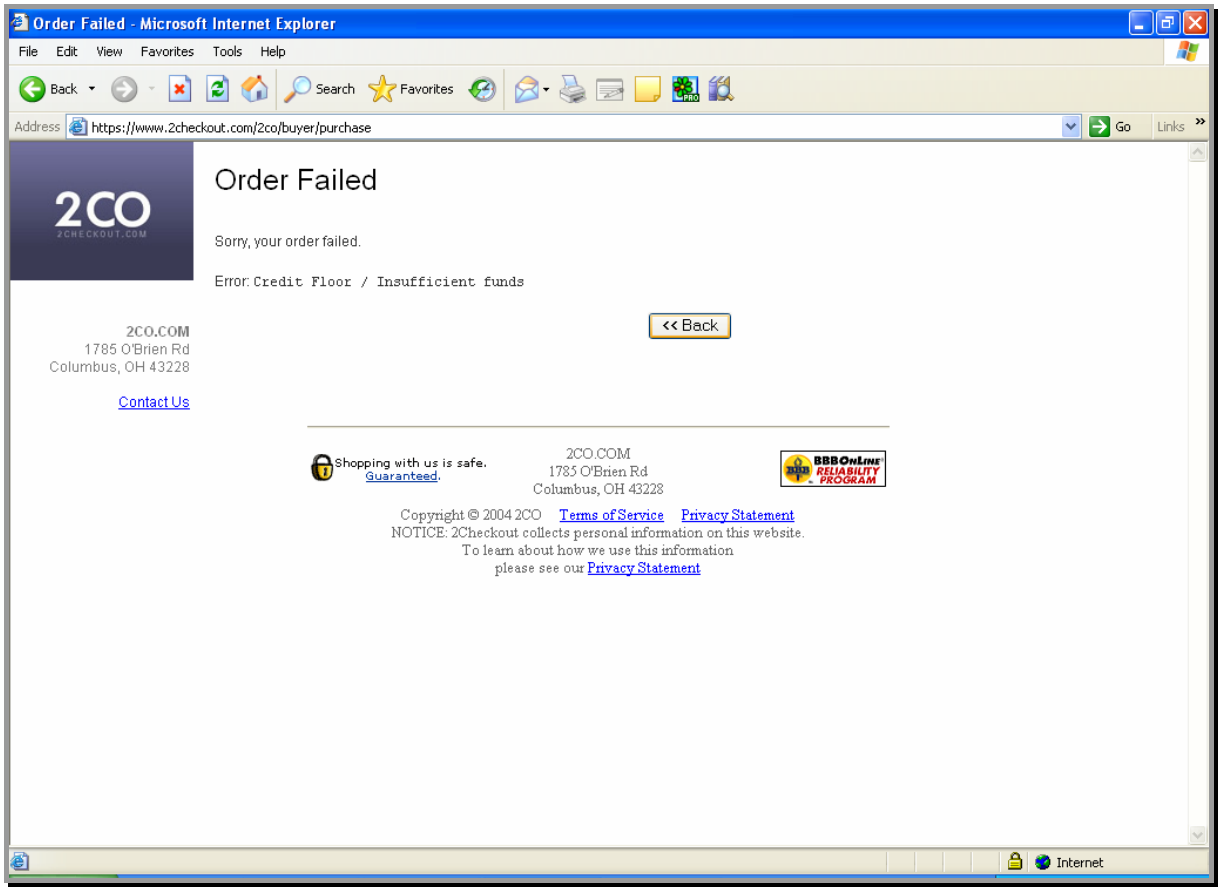
Picture 17. E-banking application main execution process.

Creating this simple example and analysing its behaviour in practive, at least five major unexpected exception sources were identified:

- Exception 1 (see Picture 17, client authorisation) – the required software is usually kept between firewalls freely accessible on Internet. The exception can arise in several cases. One of them would be credentials verification fail, which can of course lead to unauthorised access for intruders or for the true customer, who will be provided with other customer information. The other possible to register exception would be of infrastructure type, when the software itself fails to properly respond or does not respond at all. In this case, any banking system, should forbid any further access to the system, by providing the customer with appropriate information on the case.
- Exception 2 (see Picture 17, data verification) – this is of course the responsibility of the application to properly operate with provided data. This exception is closely related also with workflow and human factor exception possibilities. As if not constructed properly and given wrong data, the process can turn to unexpected for the customer way and result in totally unexpected results (such as transfer of funds to unwanted account, transfer of incorrect total of funds, performance of the other than wanted service). Data verification task can be called the most critical, as even with the most ideal process model and accurately executing application, the customer must be protected from itself, from making human nature exceptions (like choosing wrong account or service, providing not correct format of the data, providing the data in wrong places, choosing wrong actions and etc.),
- Exception 3 (see Picture 17, service availability check) – the most common B2B application function, the involvement of third parties, like other service providers. The exception possibility here lies in time related exceptions, as the trick lies in timely connections and connection existence confirmation in time, during customer procedures completion. The e-banking application should carefully search for requested service, evaluate its availability and possibility fully to complete it, if not the exception can be very tricky, as its source would be especially difficult to identify and solve. The problem would appear as this is the external service provision, that means, the e-banking system must insure inter-operability requirements and contact third-party application according to defined procedures, by logging every operation step and informing both operation actors if a deviation occur according also to predefined procedures,
- Exception 4 (see Picture 17, request logs update) – the improper logs registration can lead to initiation of any kind of exception mentioned above. Proper logs are the guarantee that

any possible deviation, fault or exception will be encountered on time and in accurate place. As it was noted in previous example, unevaluated logs can lead to most crucial results. Additionally, system should use logs for constant comparison of actual data to ideal data model described using taxonomies in order to prevent possible unpredicted situations and solve them before they are allowed to arise,

- Exception 5 (see Picture 17, informing the customer) – if such a task is not achieved in time, it can result not just in a human nature exception, but in a damage to a business application or its data. Seems a simple gesture of courtesy, but the lack of it can ruin client and bank relationships, shatter the reliability concept for the bank and bring financial losses to a customer or even worse raise legal problems for him/her. That is why any B2B application procedure should result in informative message of the achieved results for the customer. Keeping in touch with a customer, also gives him/her the illusion of importance and that client himself can manage his own banking procedures.

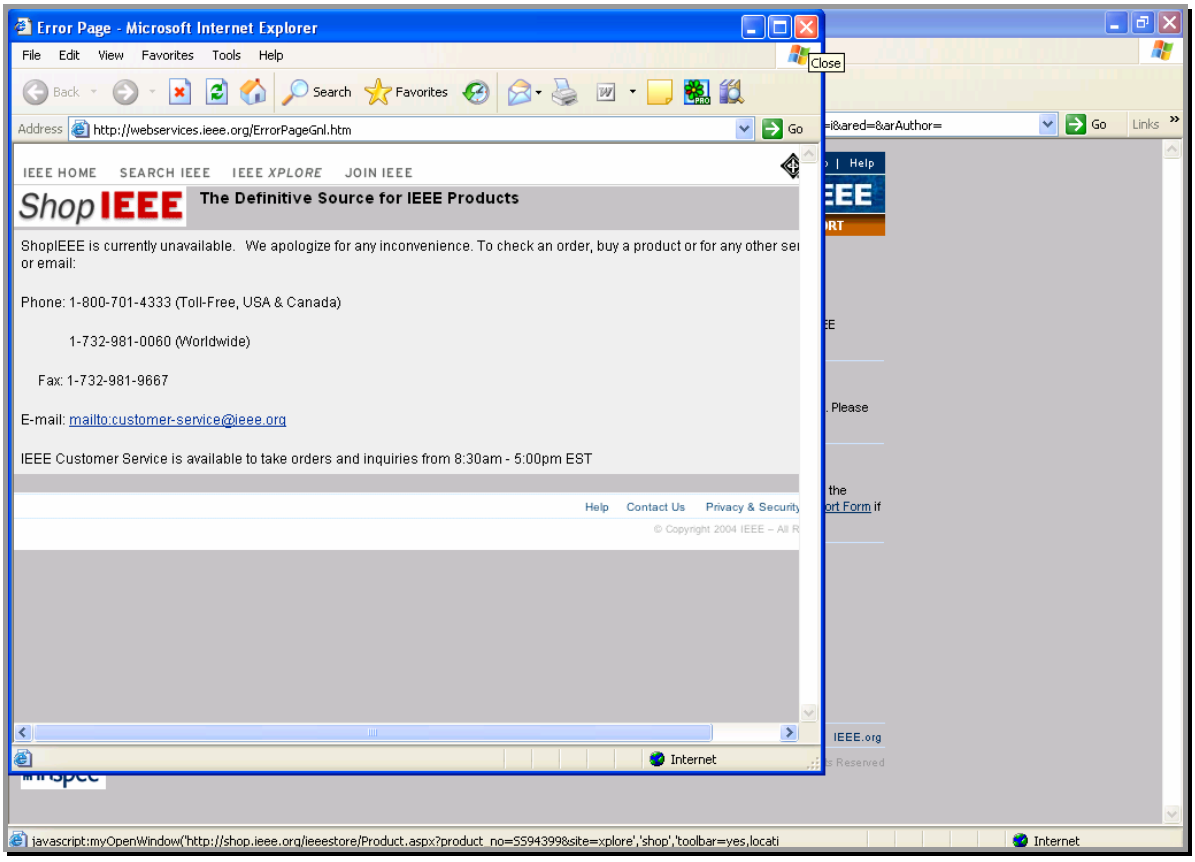


Picture 18. B2B type exception, encountered performing credit card payment.

The above picture represents the result of described above Exception 2 and Exception 5 (see Picture 17), when data verification and proper client information tasks failed. The customer was asked to provide payment related data and firstly failed to evaluate a simple thing, that a chosen

account was not suitable for current operation execution and responded as order fail because of the lack of funds in the account, while the account had more money than needed. Secondly, the application failed properly to inform the customer about the problem, that he/she should choose other, appropriate account. The result was quite tricky. The customer tried to repeat the operation, hoping that it was a temporal application failure and resulted in multiple transfers of funds. Every time, he/she tried to complete the procedure the application completed the payment but informed the customer about fail, who later had to put efforts in proving this situation and returning lost funds.

And the last case (see Picture 19) is related to the Exception 1 and Exception 5, when the system failed to even initialise and complete the authorisation task. The result, described in the picture, of course was achieved properly but failed to properly inform the customer about the exceptional situation. If the result would have been achieved in the first customer authentification, but not in the middle of procedure completion, the message would fit, but in this case, the customer should have been informed about the next steps and results which were already achieved (like: was the request submitted or funds transferred, or does he/she needs to complete something to finalise the procedure).



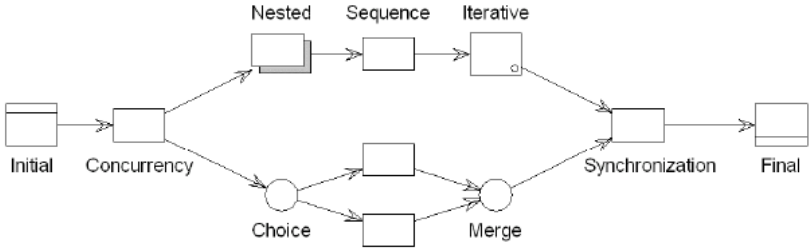
Picture 19. B2B exception example encountered trying to connect to business application.

2.5 The Usage of Workflow Methods in Business Process Exception Handling

Starting to define the exceptions occurring in one or the other environment it is always useful to define notations and rules of description of theses exceptions. There are several different ways to describe workflow and its exceptions in build-time (such as UML, flowcharts, Petri-nets), but there is also most common to all of them notation which will be used in this chapter.

As it was noted above, workflow is the automation of business process in order to achieve information migration according to predefined rules and actors. Also, the workflow management system is a set of tools used to execute and coordinate the defined workflow. Usually any workflow consists of two basic parts. It is process model (actually workflow model schema) and process instance (a particular occurrence of the process). These two elements are mostly important while dealing with exceptions, as process model is build-time result, and process instance is run-time result (run-time exception handling is discussed in detail in [51]). Therefore they raise different nature exceptions and require different methods of handling. What unite these two notations, it is node states and their ways/rules of interaction.

Further introduced workflow modelling language closely conforms to WfMC standards and gives most basic notations of workflow graph and its lifecycle depiction [40].



Picture 20. Widely used process notations.

Basically, workflow is a Directed Acyclic Graph (DAG):

$$W = <N, F> \quad (4)$$

DAG is such that ***N:*** finite set of nodes and ***F:*** flow relation:

$$F \subseteq N \times N \quad (5)$$

The nodes can be of two types. They are types, which are noted as rectangles and coordinators, choices are noted as circles. Each task of a workflow also contains a set of predefined and dynamic properties, related to data, time, related applications, resources, and actors and other.

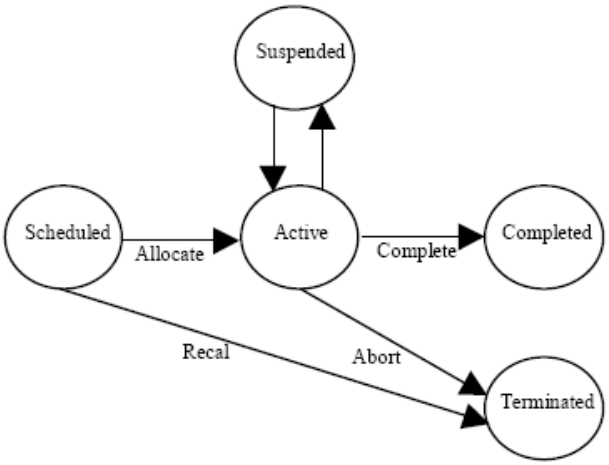
Each node $n \in N$ in workflow W has an execution structure of predefined states and transitions between them, which are modelled using Finite State Machine and can be these [68, 66]:

- *Scheduled:* scheduled for execution, but not allocated for any workflow actor,
- *Active:* allocated for an actor and being performed,
- *Suspended:* the process instance is quiescent, no further activities are started until it is resumed,
- *Complete:* completion conditions are successfully achieved,
- *Terminated:* the execution of the process has been stopped due to an error or user request.

Graphically, the finite state transitions are described in [Picture 21]. It is clear that completed state always is finite and the only two states which have dual directions are suspended and active, that means that these two states are dynamic and the others are finite results.

The picture clearly shows that if INS is a set of instances for W, then:

- ***node state:*** $N \times INS \rightarrow$ *{Scheduled, Active, Completed , Terminated, Suspended}*.
- ***node state (n,i):*** ***N*** state of node ***n*** for instance ***i***, where $n \in N, i \in INS$.



Picture 21. Node state interaction basic model [66].

If the time factor is not critical, then the following states for the instance would be [68, 66]:

- *Ready:* an instance ***i*** is in *Ready* state when $node\ state\ (n_0, i) = Scheduled$,
- *Current:* an instance ***i*** is in *Current* state when $\exists n \in N$, $node\ state\ (n, i) = Active \vee (\exists n \in N, node\ state\ (n,i) = Scheduled \wedge \exists m \in N, node\ state\ (m, i) = Completed)$,
- *Complete:* an instance ***i*** is in *Complete* state when $node\ state\ (n_f, i) = Completed$.

n_0 is an *Initial task* and n_f is the *Final task* of ***W***.

Exception Process Modelling and Enactment

Business process model is composed of the core process and a set of exception processes. It is described as [40]:

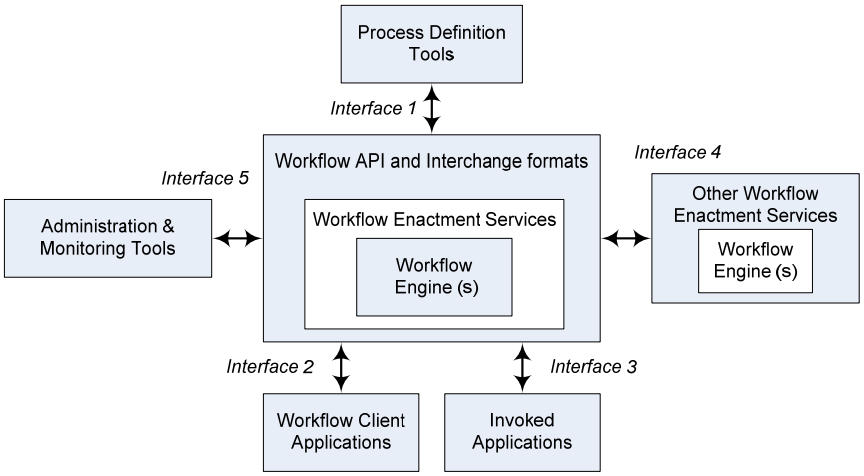
$$BP=\{CP, P_1, P_2, P_3, \ldots P_k\} \qquad (6)$$

In this formula:

- ***CP*** is a core business process, encapsulating all necessary business logic,
- ***P_i'(i:1, 2, …k)*** is the exception handling component of the business process,
- and ***P_i*** is ***(n, t, f, e, p)***, where:
  - ***n*** is the name of exception,
  - ***t*** is a set of triggering objects, which can include task, block of tasks, all the process or external triggers,
  - ***f*** is a failure condition,
  - ***e*** is an exception handling process,
  - ***p*** is a type of the exception.

The ***P_i*** processes can be modelled as sub-processes of core process if an exception is of temporal nature and the process can always return to the normal flow. And ***P_i*** can be modelled as an independent core process, if a raised exception requires core process substitution.

Any core process task or exception handling task is managed by workflow management system, the de facto standard for which is defined by WfMC. Still the proper exception handling would require some additional instruments or components added to the workflow management system. The core element of which is workflow engine, performing all process execution.



Picture 22. WfMC Reference Model [21, 80].

WfMC reference model defines the main components which are necessary for any business process lifecycle, starting its modelling and finishing with the result archival or analysis. Therefore, coalition defines workflow engine as the core element, which, given the appropriate format input, returns strictly defined output. To achieve that, required API's and interchange formats must be defined as inseparable elements of core workflow management system part.

The rest elements of the model are the tools of result achievement, by modelling, administering it, presenting it's running instance to the actors and integrating with other workflow management systems.

The five defined interfaces are for communication insurance between all the elements of workflow management system. The fifth interface is the critical element for inter-organisational communication (like in B2B applications) and does not always exist is simple workflow management systems, used for simple application realisation (like document management system). But this interface together with interface 4 (see Picture 22) is critical for flexible exception handling achievement. The later performs detection, diagnosis and executes exception handling procedures, while interface 5 (see Picture 22) is responsible for timely data delivery to the core workflow management system element in an appropriate format.

By receiving necessary triggers from exception handling engine, the main engine defines in what manner the handling procedure should be executed: as sub-process or independent process, which is also called process view.

For an unanticipated exception $\boldsymbol{P_i}$ would be defined as *(unknown, [W], NULL, empty-process, exception type)*. As the type of exception is not known and the handling ways are not defined, the main goal of the process still remains to achieve business goal. To complete that, the whole exception detection, diagnosis and handling process should constantly concentrate on business goal.

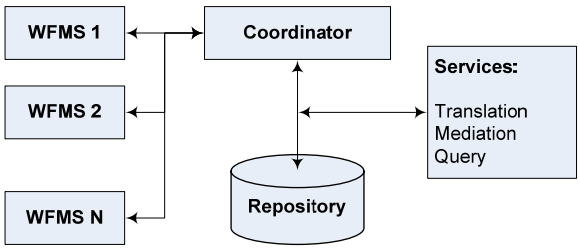
The handling process of course should start from identifying failure conditions and its effects on business goal, after that ***t*** is defined which allows defining ***p***. The last steps remain to identify the exception source, which would be ***t***, and select appropriate handler ***e***.

Knowledge Based Exception Handling

Every unpredicted and unknown situation handling is about knowledge, how to detect, how to make diagnosis and how to solve. Knowledge is the essential component in any goals achieving application. Sometime this knowledge is also called "best practice" or "know-how". But what really distinct these later definitions from knowledge dedicated for exception handling is the changing and unpredictable nature of surrounding environment. It is always difficult to

predict what exceptions can be raised. The same difficulty is to predict, what knowledge will be required at different points to solve situations.

That leads to a conclusion what human factor is inseparable from exception handling application knowledge and is a part of it. The knowledge based exception handling model is described in detail in [74]. The main idea is to gather various workflow management system exceptions and collect them into one repository. The same is done with handling methods of these exceptions, with a possibility to extract needed information later and apply at any required moment. Such reuse of information is considered a possible way of continuous process improvement.



Picture 23. Exception knowledge sharing model [74].

Exception knowledge sharing includes:

- Exception patterns – defined and stored as exception specifications,
- Exception handler patterns – defined and stored as exception handler specifications,
- Exception handling experience – specification evaluation and improvement.

The patterns are described using attribute collections and stored in repository, which can be accessible for any business process, including inter-organisational processes. The repository itself is an additional element of workflow management system and interacts with latter with a help of Interface 5, discussed earlier. The interface is constructed so, that it could access such information:

- The structural information about workflow specifications,
- Exceptions that occur in inter-organisational processes,
- Exception handlers for specific exceptions,
- Responsible party for the abnormal situations,
- Interaction points among cooperating business processes.

Exception specification is the first thing that is received at exception handling application and should be strictly defined (including format, delivery and its receiving). One of the most important things is the classification of exception, which help to diagnose exception faster. Usually, exception classification is defined using taxonomies, which can be renewed any time. Most important thing as in classification, as in overall exception handling is inter-organisational

exception specification definition. This kind of exceptions should be defined using wider collection of attributes than other exceptions, as more critical information is needed in such cases.

Most researchers come to a conclusion that exception handling is a possibility for any application to learn and evolve. The approach is correct, but is acceptable only if exception handling application is smart enough to realise it.

Traditional workflow management systems already have some necessary handlers for simple exception handling, usually it is predicted exceptions handling. The handlers are:

- Ignore - no measures are taken if the business goal is achieved anyway,
- Record - the exception is recorded using specification as soon as it happens,
- Retry - this is relevant only to repeatable exceptions. In such specifications time factor is necessary to define,
- Compensation – if the effect can be compensated with the other action,
- Alternative task – instead of running task, the alternative task is enacted,
- Backwards recovery – the method of rolling back the workflow to a last consistent state,
- Forward recovery – proposed for long running workflows, then it is rolled back to the last consistent state, resumed and proceeded,
- Propagation – if native exception handlers does not fit, the exception is propagated to exception coordinator which is aware of more handling schemes,
- Termination, suspension and resumption – basic functionality for supporting workflow handling,
- Procedural exception handler – involves series of steps to handle the exception.

The named exception handlers are suitable only for simple workflow handling systems. As for inter-organisational exceptions, these handlers must be enhanced with additional elements, like compensation proceeding rework (CPR). CPR works on the basis of agreement between two parties and is the extension of compensation handler. The CPR template is defined like this [74]:

- Compensation mode – automatic, manual or none,
- Compensation type – denotes the compensation scheme type (class name),
- Compensation action - denotes the compensation scheme type (method name),
- Compensation parameter number – number of parameters needed for compensation scheme,
- Compensation parameter string – parameters past to the compensation scheme,
- Rework mode - automatic, manual or none,
- Rework type – type of rework: retry, alternative task, etc.,

- Rework task name – task name to be retried or activated,
- Rework state string – state string, that the correct parameters and data could be found,
- Rework parameter string – task parameter string, that the correct parameters and data could be found,
- Rework host – denoted the host the task will reside,
- CPR mode – user and automatic,
- CPR name – denotes the adaptability of this CPR.

It is clear that this CPR method is the extension of $P_i\ (n, t, f, e, p)$ definition, added with additional collection of attributes.

The control flow semantics of CPR is defined in such ways [74]:

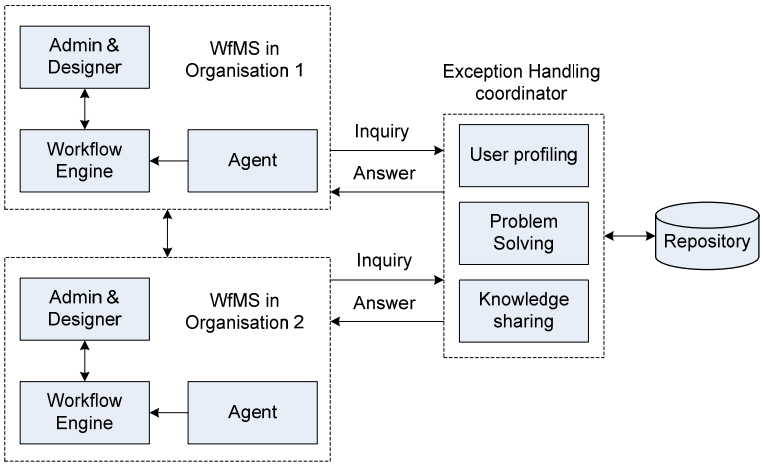
- If (compensation mode is automatic) the compensation scheme is automatically executed,
- If (compensation mode is manual) a human needs to be involved to execute the compensation scheme,
- If (compensation mode is none) no compensation is necessary for this CPR,
- If (rework mode is automatic) the rework scheme is automatically executed,
- If (rework is manual) a human needs to be involved to execute the rework scheme,
- If (rework is none) no rework is necessary for this CPR.

As it is clear the CPR method is based on simple compensation type using simple if then constructions. Such approach has its limitations, as compensation type is not always suitable, especially in inter-organisational exceptions. Also control flow types lacks complex compensation and rework types, involving as automatic and human involvement, which is most casual phenomenon.

Coordinated Exception Handling

As it was obvious in previous section, knowledge without coordination can be useless, especially, if it is about cross-organisational exceptions. Coordinated exception handling is presented in several ways: as extended knowledge sharing model enhances with coordination functions [73, 74] and using the coordinated atomic actions (AC) framework [70].

Firstly, coordinated exception handling is defined having in mind cross-organisation process interaction, which can be one-way (process send a message to another process, after which no interaction exists between these processes) or two-way interaction (process send a message to another process, it can either receive response immediately or at later time).



Picture 24. Conceptual model of coordinated exception handling in cross-organisational environment [73].

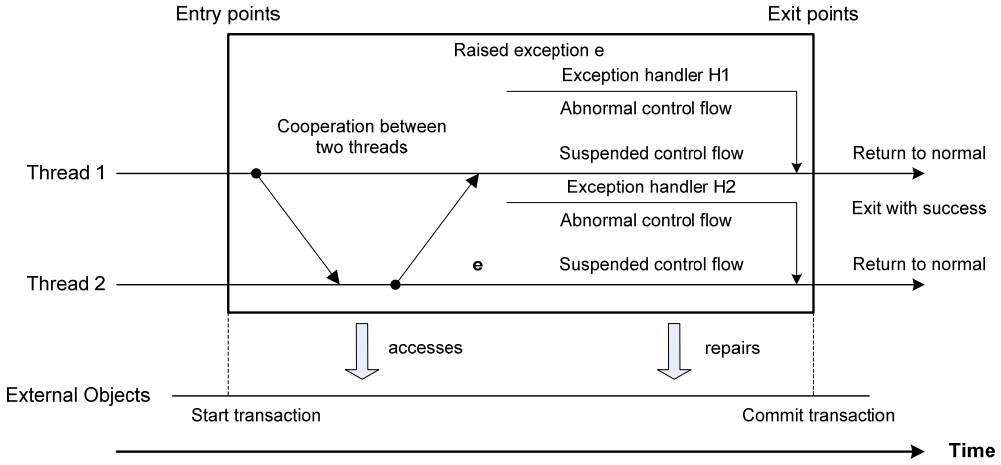
This approach also solves the issues about operation flexibility and process privacy using five coordinated exception handling patterns [73, 74]:

- Immediate pattern – suitable for single two-way interaction, which also shares information on exception and uses CPR handler template:
  - If (mode is immediate) signal the exception directly,
  - If (compensation is auto/manual) execute the compensation scheme,
  - If (rework is auto/manual) execute rework scheme,
- Deferred pattern – suitable for a single two-way interaction, but the exception information is not reported immediately. This patterns uses such CPR handler template:
  - If (mode is differed) fulfil the request and add the exception object to the normal flow,
  - Signal exception,
  - If (compensation is necessary) execute the compensation scheme,
  - If (rework is necessary) execute rework scheme,
- De-coupled pattern – used for single one-way interaction and exception information is passed through exception handling coordinator:
  - If (mode is de-coupled) find the contract party in cooperating process and the interaction point,
  - Signal exception through the exception handling coordinator,
  - If (compensation is necessary) execute the compensation scheme,
  - If (rework is necessary) execute rework scheme

- Free pattern – is suitable when there is a several interaction points between processes, such that two-way interaction is possible through more than one interaction point:
    - If (mode is free) find the contract party in cooperating process and the interaction point,
    - Add an exception object to the normal flow,
    - Signal the exception,
    - If (compensation is necessary) execute the compensation scheme,
    - If (rework is necessary) execute rework scheme,
- Close pattern – is used when there is no interaction points between processes such that no interactions are possible and exception are passed through coordinator for a record.

The second way to achieve the coordinated exception handling is to involve atomic activity framework.

Definition 15 *The activity of a group of components or objects constitutes an atomic action if there are no interactions between that group and the rest of the system for the duration of the activity [70].*



Picture 25. Coordinated exception recovery [70].

Picture 25 is the depiction of two threads entering a ***CA action*** synchronously through different entry points having a common goal. An exception is raised and one of the threads is informed of it, then both threads transfer control to their respective handlers ***H1*** and ***H2*** which attempt to perform forward recovery. External object are put into correct states so that ***CA action*** is able to complete successfully.

In ***CA action*** framework exception declaration is a bit different and defined only as:

- Totally internal to the ***CA*** and declared as part of the action definition and if raised are entirely handled with this action. These exception are declared as:

$$e = \{e_1, e_2, e_3, \ldots\} \quad (7)$$

- Those that are known and can be signalled to the enclosing environment and are depicted as:

$$\varepsilon = \{ \varepsilon_1 , \varepsilon_2 , \varepsilon_3 , \ldots\} \quad (8)$$

The definition of e and ε are totally recursive:

$$\varepsilon_{nested} \subseteq e_{enclosing} \quad (9)$$

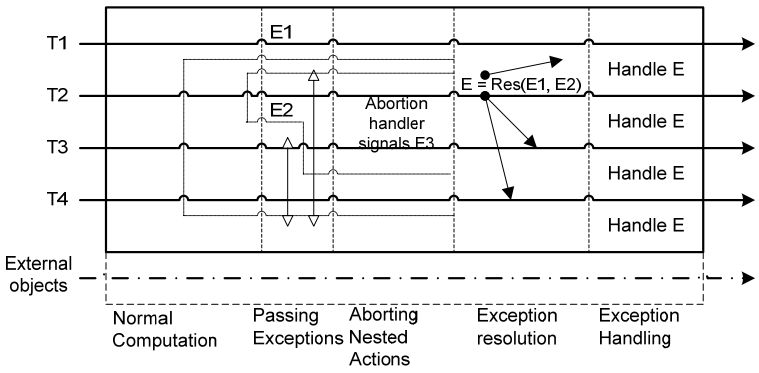
There exists only two special exceptions ***μ*** and ***f*** in ***ε***. An undo exception ***μ*** implies that the action has been aborted and all its damage has been undone. As undo is not always possible, failure exception ***f*** indicates action abortion, but the damage can be not completely undone.

The authors of [70] uses the following set of definitions for coordinated exception handling algorithm:

- The innermost action of specified thread is called active ***CA***,
- The group of participating threads are:

$$G_{CA\ actions} = \{T1, T2, \ldots, Ti, \ldots, Tj, \ldots\} \quad (10)$$

  In (10) formula each thread has unique identifier and the threads are ordered,
- The participating thread ***Ti*** can be in any of the next states (defined as ***S(Ti)):*** *N-normal*, *E – exceptional* (if any ***Ti*** is raised), *S – suspended* (if ***Ti*** has to stop normal execution flow),
- ***A*** is the active action of ***Ti*** and ***GA*** is a corresponding set of participating threads,
- ***Ti*** can keep such data structures as ***LEi*** (records exceptions that have been raised and has ***S*** state) or ***SAi*** (stores the name of nested actions ***Ti*** in which is currently is),
- Application related messages are of these types:
  - *Exception(A, Ti, E)* is sent by thread ***Ti*** to all the other threads of action ***A*** when an exception ***E*** is raised by ***Ti,***
  - *Suspended(A, Ti, S)* is sent by each thread ***Ti*** that does not raise an exception but has received an *Exception* or *Suspended* message from another thread, where ***S*** indicates ***Ti*** is in the *Suspended* state,
  - *Commit(A, E)* is sent by a chosen thread in action ***A*** to all the other threads after it completes resolution of exceptions, where ***E*** is the resolving exception. A corresponding handler for ***E*** will be called by each thread once it receives this ***`Commit`*** message.



Picture 26. Exception handling and resolution [70].

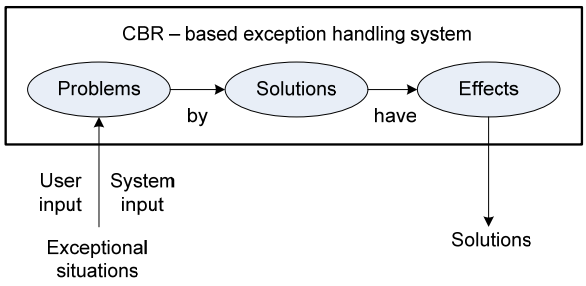
The algorithm of coordinated exception handling is based on FIFO messaging and calls to abortion handlers. Picture 26 shows how the algorithm works when exceptions ***E1*** and ***E2*** are raised concurrently in several nested ***CA*** actions. The algorithm informs all the participating threads of the exceptions with a help of messages. After that, it aborts two nested actions because of ***E1*** that occurred outside the nested action. Finally, the algorithm determines a resolving exception ***E*** that covers ***E1*** and ***E3*** and starts the corresponding handler ***E***.

Intelligent Problem Solving

The method of intelligent problem solving enhances knowledge sharing by incorporating CBR (case-based reasoning) approach and in so doing improves exception handling capabilities. Occurred exception is propagated to intelligent problem solver, which consults the case repository. Similar cases are extracted and analysed; if some of them are adaptable it is automatically derived. The most important in this method is similarity retrieval which similarity measure must take into account as semantic, as structural similarities.

The similarity measure is achieved by obtaining next [73]:

- Exception similarity – based on relationship in exception hierarchy,
- Workflow similarity – workflow structural and building block similarity,
- Context similarity - is obtained by computing nearest neighbourhood function of the quantified degrees of semantic similarities over workflow application data.



Picture 27. CBR based exception handling.

The CBR based exception handling method is used to derive the applicable exception handler. A human intervention is needed if the selection can not be achieved automatically. Effect of exception handler candidate is evaluated upon the effect it will bring on workflow and application. If need be, some correction can be made to improve the handler. The whole exception resolution must perform following actions [73]:

- Exception handling coordination mode must be determined, according to the process interaction types,
- The contracting party and interaction point must be determined,
- The compensation scheme must be found if necessary,
- The rework scheme must be found if necessary.

Obvious that CBR based exception handling uses main coordinated exception handling framework, enhanced with a similarity measure procedure, which actually includes not only workflow, exception and context similarity measure, but also computing similarities between targeted (new) case and old case for three components: exception information block, case information block and action information block. Each of these blocks are context dependent and has its own collection of attributes, therefore the following formula is used:

$$S = \sqrt{\sum_{n} w_i \times (a_i - b_i) \times (a_i - b_i)} \qquad (11)$$

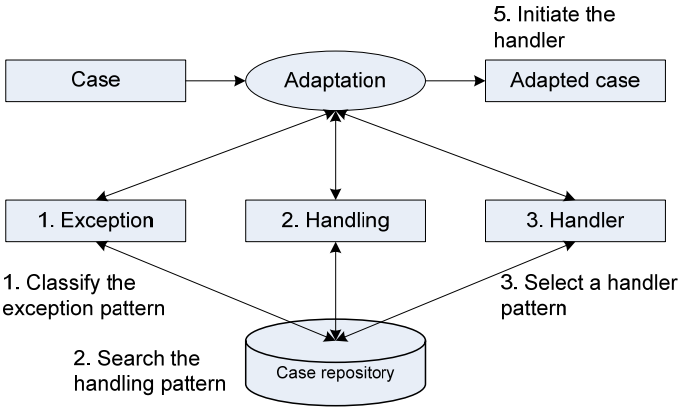
- $\boldsymbol{w}_i$ is the weight for $\boldsymbol{i}^{th}$ attribute,
- $\boldsymbol{a}_i$ is $\boldsymbol{i}^{th}$ attribute in case $\boldsymbol{a}$,
- $\boldsymbol{b}_i$ is $\boldsymbol{i}^{th}$ attribute in case $\boldsymbol{b}$,
- $\boldsymbol{(a_i - b_i)}$ is the similarity between attribute $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$.

In case two exceptions cases do not have the same number of attributes, then a default reasoning scheme is used. It is when default values are assigned to these attributes which doesn't exist before the similarity calculation. After that the concept trees, built using ontology, are used to calculate the similarities. And finally, similarity between two concepts is determined by considering the sibling difference and the level of difference between the two concepts in the concept tree.

The successful case retrieval of course leads to the case adaptation and final exception handling. There are two ways to achieve that [73]:

- Problem adaptation – adapting the case by enhancing the partial description of the problem or substituting conception realisation in a case,

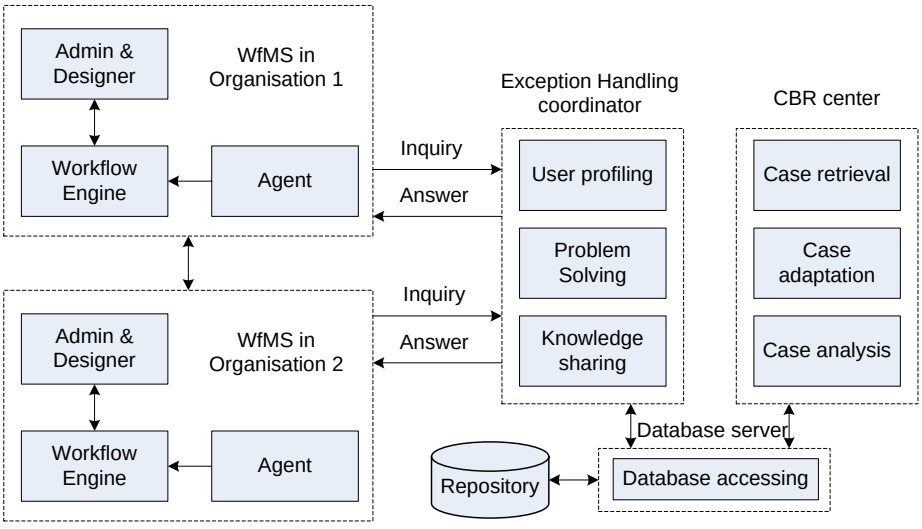
- Solution adaptation – reusing the past case solution adaptation or reusing the past methods that constructed the solution.



Picture 28. Case adaptation model [73].

As it is pictured in Picture 28, the pattern adaptation scheme must complete four steps:

- Classify the exception pattern – pattern identification step. If a new patterns is encountered, it is added to the pattern repository,
- Handling pattern search – after exception pattern determination, the handler must be found. This step also identifies exception handling coordination mode,
- Handler pattern selection – the most appropriate handler is selected from all received patterns in the repository,
- Handler initialisation – CBR handler template is populated and an adapted case is created.



Picture 29. Intelligent problem solving framework architecture [73].

At it is seen in Picture 29, intelligent problem solving is a combination of knowledge sharing and coordinated exception handling. The approach is advanced in a way, that it is a decomposed application, consisting of organisational workflow management systems, exception handling coordinator, repository accessing client and CBR centre. The approach brings flexibility into whole application support and access, as also insures process confidentiality, as there are no direct interaction with repository, but only through the coordination client, which is a Interface 5 type described in WfMC reference model and completes all held requirements for it.

2.6 Conclusions

The analysis of business process exception handling methodologies and existing handling models leads to the following conclusions:

1. Workflows can be modelled using three basic methods; it is Petri-nets, Flowcharts and activity diagrams, for which certain tools already exist. As analysis showed, analytically and from the scientific point of view, the best method is Petri-Net. For business users it is too complex (modelling and understanding), that is why it is not used widely in commercial workflow management systems. The most friendly and understandable way to model workflow is to use Flowcharts, which are modelled using graphical tools and are very informative. Activity diagrams are more useful in process analysis stages, wherefore rarely used in commercial systems.
2. Basic workflow exception methodology, consisting of detection, diagnosis and handling phases, gives a good framework for starting to model a workflow handling system and does not depend on any technical method used in this framework. Also it provides a good background for composing the workflow based exception handling, as the framework itself is of workflow nature.
3. B2C and B2B exception differences are mostly identified because of their different architecture and environment in which they are used, and different data management policies. In B2C application exception scale, the biggest impact has the human-factor exceptions, while in B2B inter-organisational and flexibility concerning exceptions.
4. Currently existing exception handling methods, such as compensation proceeding rework (CPR), case-based reasoning (CBR), knowledge based and intelligent problem solving methods, has separate strong exception management functions and procedures. Still, as self-standing exception handling application they could be used only in B2C environment. In B2B applications, the combination of the strongest functions of these

methods (knowledge management and reuse) would be more useful. Some of these methods like CPR and CBR are quite complex in their achievement therefore can be difficult to support. While intelligent problem solving method is the most suitable in it handling logic and achievement procedure for B2C and B2B applications.

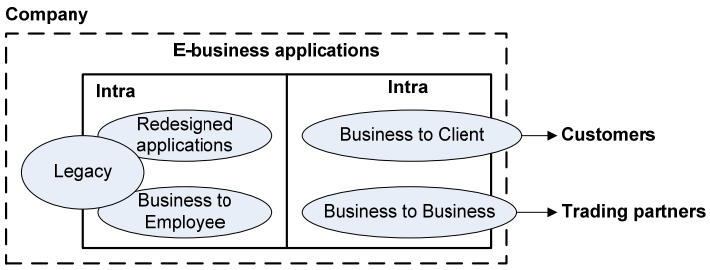
3 Business Process Business-to-Client Exception Handling

3.1 Introduction

The expected result of any e-business application is improved client management, which includes client satisfaction and constant return, also possibilities to manipulate client using customer loyalty programs and the most important – flexibility and privacy, client managed e-business procedures (raised requirements are discussed in [52]).

The company must be constantly involved in monitoring client performed operations – in case to avoid company and client misunderstandings. To achieve that, e-business application must be constantly improved and optimised according to the changing business processes.

This section is dedicated to the analysis and description of B2C based application build-time and run-time subtleties and its exception management patterns. The described further example was implemented in one of the telecommunication companies. Besides B2C operations the application manages many more functions (like document, HR and project management) which of course make the application very complex and versatile. The nature of such application does not lack various internal and external integrations. The system is also unique as it changes the classification proposed in Picture 30 as it spans intra and inter application types and can be prescribed as to B2E as to B2C application, both of which integrate legacy system.



Picture 30. E-business application classification scheme [30].

To avoid problems which bring application complexity, the application was broken into separate modules which can operate independently and covers one of the B2C or B2E applications function collections. The exception handling in this application was implemented as internally (into each application module), as externally (the separate exception handling module was also designed for complex and unpredicted exception handling). In such a way, designers separated expected and unexpected exception handling [54].

Two techniques were used for this exception handling realisation. One of them was the integration of conditional sentences into business process workflow. The other was knowledge

sharing model with a tips of coordination model, both of them were described in previous sections. The whole solution was implemented using IBM Lotus Domino R6 platform.

3.2 Business Process Business-to-Client Exception Handling Algorithms

Exception handling using conditional sentences

Conditional sentences are the most usual way to model expected deviations and exceptions. Unluckily, such method is totally not appropriate for change management, which usually ends in unexpected exception handling. That is why in this research conditional sentences method was chosen only for every separate module internal business logic deviation modelling. The positive side of such method is that a human is constantly involved into any expected exception handling as this handling is a part of a business process and delivered to a business actor as a simple workflow task.

The further research results of B2C exception handling using conditional sentences were achieved and will be delivered using example of the business process implemented in the telecommunication company and which was designed for complex business tasks execution.

The process actors were defined as:

- Task creator – the single actor, which is responsible for primary task creation and distribution,
- Approvers – the single actor, which is responsible for a final task execution and mid-execution approvals or declines,
- Task managers – mid-level managers, who are the main coordinators of the process, possessing all necessary information about the task and capable of acting as exception managers,
- Exception manager – a single actor, the mid-level manager, who is aware of solving complex predicted but not planned for current moment business process situations,
- Executors – multiple actors, main task performers, who acts according to task manager instructions and can not make any task acceptance related decisions and can not perform exception management,
- Administrator – application supporting person, who is not informed about business process details, but is the main business process supporting person.

According to the role prescribed to the person, the separate task collection is presented to the actor together with the business process management functions, necessary for that particular task

collection achievement. The task collection can vary in the course of the process execution and depend on approver or task/exception manager decisions.

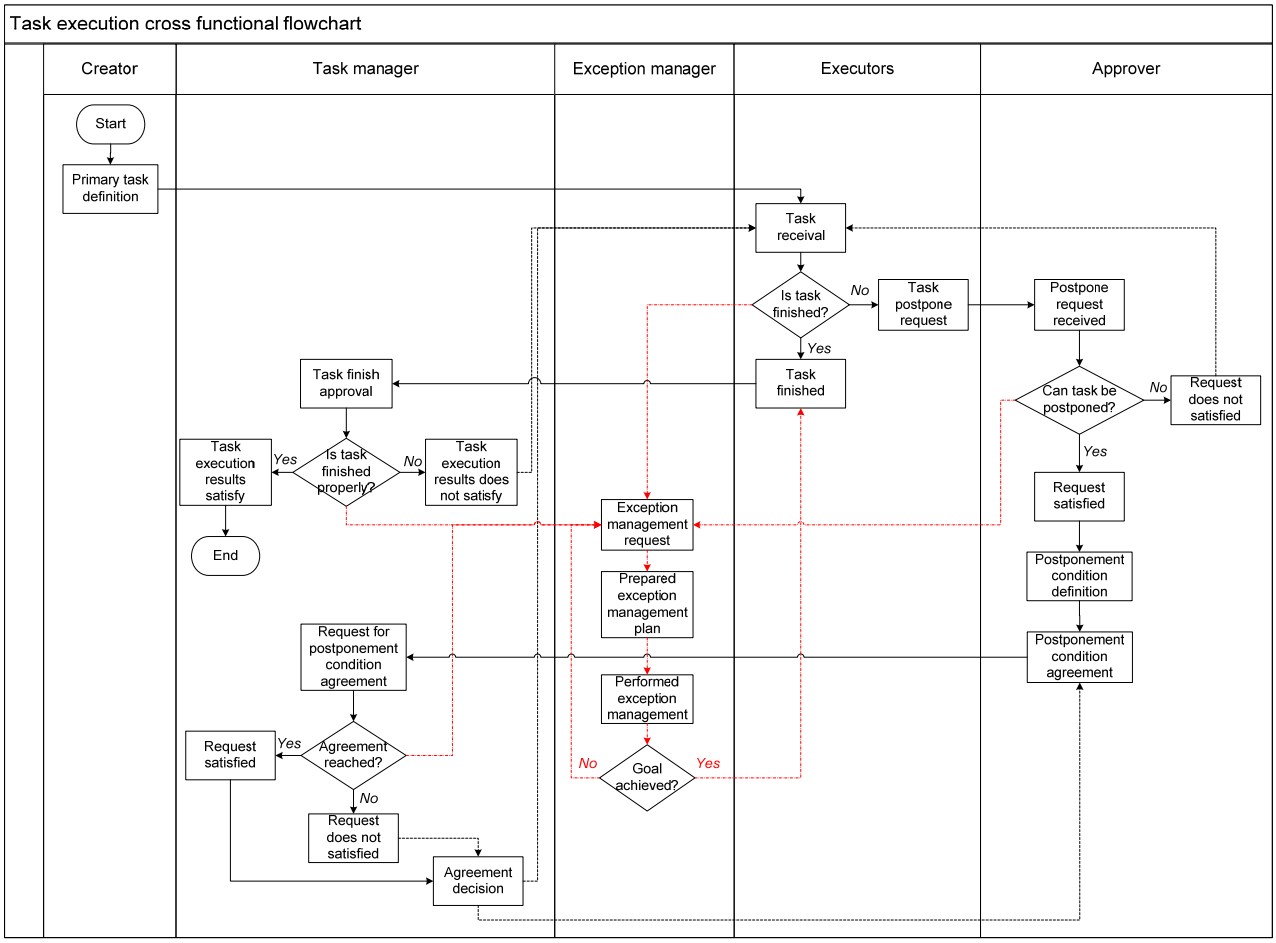
Also exist some cases of process actors overlapping and this depends on task complexity level:

- Approver is the same person as task manager – the same actor. In that case the task execution approval is done by a single person,
- Approver is not the same as task manager – separate actors. Such case denotes that final task execution will be approved in two steps. The first step will be performed by task manager, and the second – final, by approver,
- Task manager is the same person as exception manager – the same actor. In such case process execution delivery time can be shorter, as it does not involve additional decision makers, but can lack situation objectivity and impact final results quality,
- Task manager is not the same as exception manager – separate actors, which share responsibility for the task execution quality and knowledge for solving exceptional situations,
- Task creator is the same person as task manager – the same actor, which is responsible for exhaustive task definition and all actors selection,
- Task creator is not the same as task manager – separate actors; task creator is totally not involved into task execution just primary task issues distributions. In such case, total task conditions description is passed to a task manager defined by a task creator,
- Task creator is the same as approver – the same actor, initiates the task execution and can define task conditions, but the task management is passed to task manager defined by approver.

For overall task execution business model definition the cross functional flowchart was prepared, which included task execution and control mechanism.

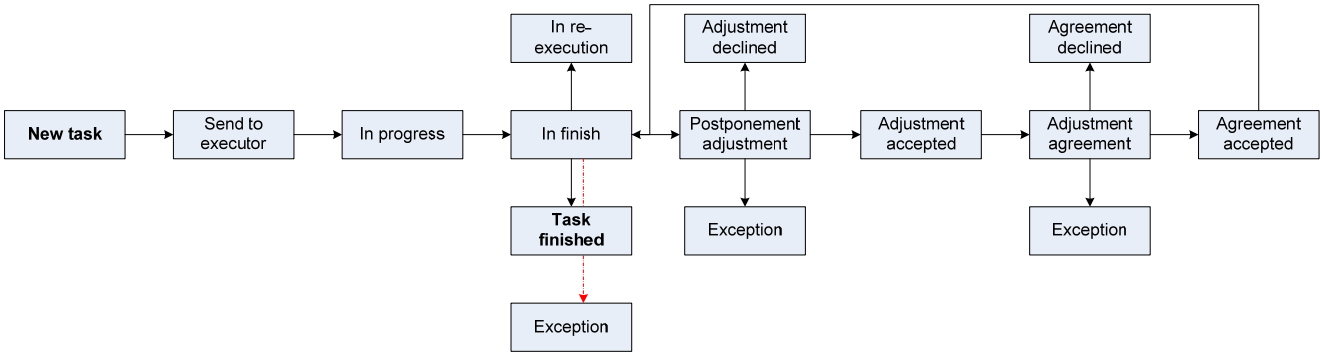
The business process workflow was constructed on the basis of cross functional flowchart. The workflow was defined describing every task, prescribing it a state definition, which in execution process is used as one of the main identifiers. Every defined business task is farther addressed as process instance, composed of activity collection, which is automated or manual. The automated activity instance list further is interpreted as activity instances list, which will be selected and executed according to process instance state.

Process definition includes one manual activity, which is performed if task manager is approved to manually change the process state. This case is further addresses as typical, known and easily solved exception.



Picture 31. Task execution process flowchart.

The final workflow, which was constructed using flowchart (see Picture 31) includes four conditional routes, which conditions further workflow execution. The choices are performed explicitly by human decisions. Workflow itself can end only in one place that is the ideal workflow requirements, which leaves no possibilities for expected deviations.



Picture 32. Task execution lifecycle.

The lifecycle of the task process consist of three stages, each of them include a state package, defining every step of the process execution. The exception is not detailed in states just in one state *Exception*. Exception lifecycle will be discussed in detail later.

Table 8 Task execution lifecycle description.

Stage	State	Involved actors	Description
Task creation	New task	Creator/task manager/approver	Task is created, execution details and actors defined.
	Sent to executor	Executor	Executor receives a task and starts execution.
Task execution	In progress	Executor	Executor performs instructions defined in task description. At the end, executor sends the task to the task manager for approval.
	In finish	Task manager	Task manager evaluates task execution.
	In re-execution	Executor	Task manager finds unfinished issues and return requests task re-execution.
	Task finished	Task manager	Task manager finds the execution of the task complete.
Task postponement	Postponement adjustment	Approver	Task executor requests the postponement of these task executions due to defined reasons. It is the duty of the approver to evaluate such request.
	Adjustment declined	Executor	Approver does not agree with task execution date postponement and return the request to executor with appropriate explanation.
	Adjustment accepted	Approver	Approver finds the postponement request reasonable and acceptable.
	Adjustment agreement	Task manager	After admitting to postpone the task execution date, approver must seek agreement for postponement with task manager.
	Agreement accepted	Task manager	Task manager finds postponement acceptable.
	Agreement declined	Executor/Approver	Task manager does not agree to postpone task execution due to defined reasons and returns the request to executor and approver.

From the workflow and its lifecycle described above, it is possible to predict some critical process points:

- Execution date definition,

- Task prescription to proper executor,
- Execution date monitoring,
- Task finish acceptance (performed by one or several managers).

These critical process points can be prescribed to possible exceptions. Every of these exception can be specified using specification language defined in Picture 23 and an individual handling plan can be prepared for every of them. Two of these exceptions can be handled programmatically in automated way depending on particular identifiers.

In research process context were identified and declared such expected exceptions:

- Late task execution date,
- Missing any of workflow execution actors,
- Task state change by human person on demand.

These expected exceptions can be specified using the format of P_i is ***(n, t, f, e, p)***:

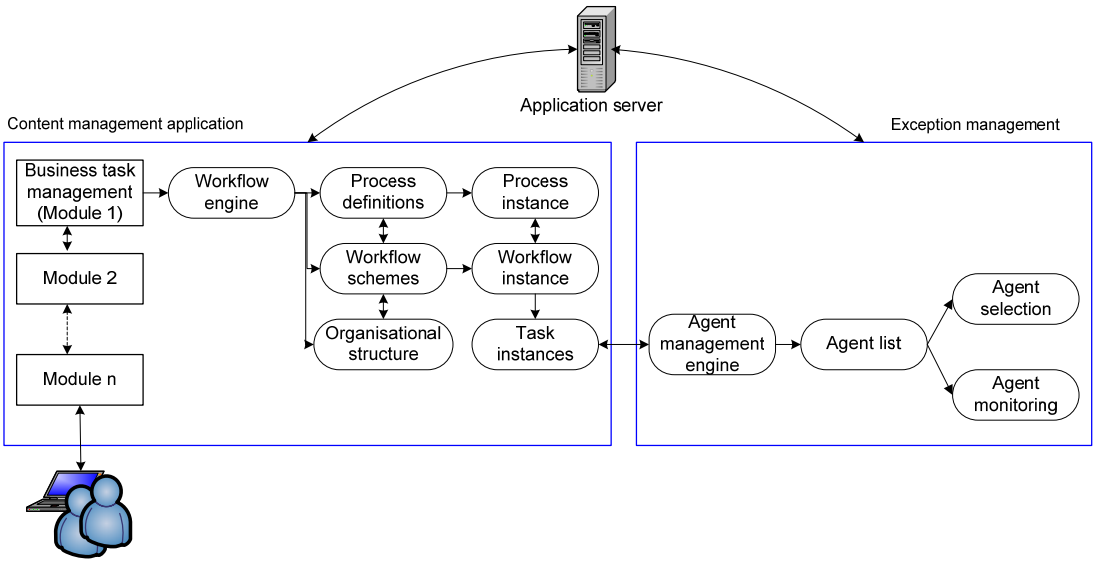
- ***Late execution date (n, t, f, e, p)***, where
  - ***n*** – execution date overdue,
  - ***t*** – the defined date of execution,
  - ***f*** – today's date,
  - ***e*** – exception handler (date evaluation and decision making agent),
  - ***p*** – process execution can be renewed from the same point,
- ***Substitution of missing actor (n, t, f, e, p)***, where
  - ***n*** – missing any of process actors (possible vacations, sickness, mission),
  - ***t*** – system returned identifier indicating missing actor,
  - ***f*** – missing actor profile availability identification return True,
  - ***e*** - substitution evaluation and performing agent, redistributing tasks to new actors,
  - ***p*** - process execution can be renewed from the same point.
- ***Process state change (n, t, f, e, p)***, where
  - ***n*** – process execution deviation pointing to state change,
  - ***t*** - current state,
  - ***f*** – current state does not correspond to real situation,
  - ***e*** – handling performed by human decision using human intervention,
  - ***p*** – process is suspended and resumes from human defined state.

Main execution process data for any task is execution date, task execution conditions and actors. All of these components influence task execution process and the final goal statement.

After task execution date changes, the process can be not suspended, only responsible actors are informed using e-message format and execution resumes from the same point.

If any of execution conditions changes, the process can be terminated and new task can be created defining new execution terms and conditions. In such a case, the old task is terminated and the execution is renewed using sub-process (which is a newly created task). If the conditions of the old task can be corrected, then the task is suspended and resumes after the condition re-definition. The actors are also informed using predefined e-message format.

Overall task execution function collection is managed by independent task execution module, which is a part of complex application, which was constructed for basic exception management achievement (see Picture 33).



Picture 33. Task module in overall application conception.

Overall created application conception is quite simple and is separated only to content management and exception handling parts. The discussed business task management module has its own workflow engine which is supplied by stored process definitions, workflow schemas and actor roles information. Expected exception management was created and achieved at task instance level using predefined handling agents, which are managed by agent engine. The approach is simple to construct and suitable for not complex business process execution, but can not handle unexpected exceptions.

The usage of conditional sentences together with exception management

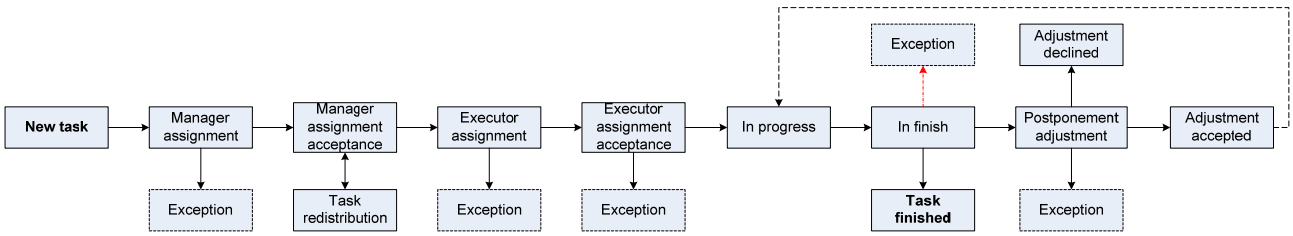
Next, the example of combining conditional sentences and true exception management features will be delivered, which was created also during this research. The difference, the

advantages and limitations, are very easy to identify, as the presented example is the same as presented in the previous section and was implemented in a big Lithuanian production company.

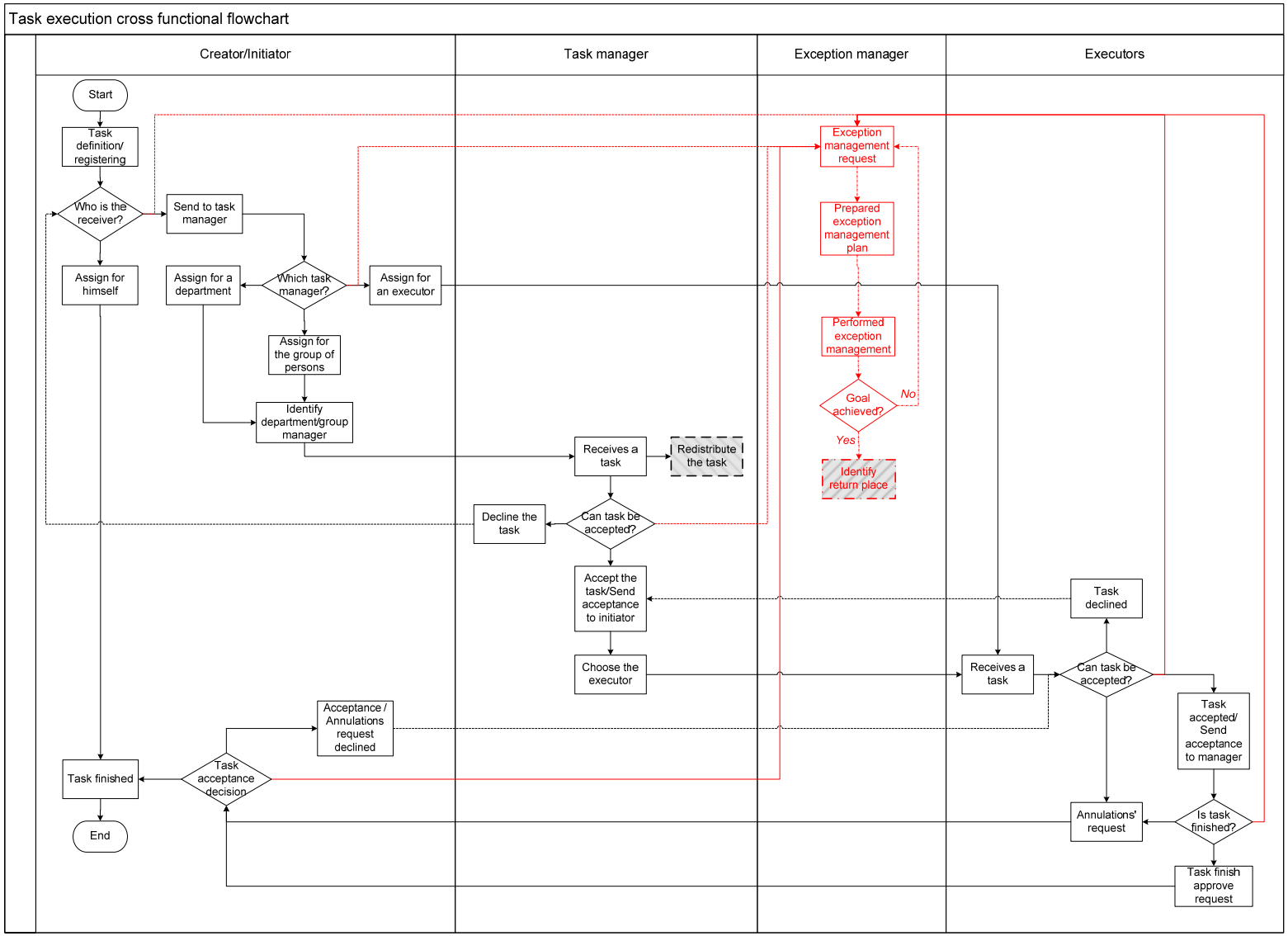
The advanced task management cross functional flowchart is presented below, which describes the business model of this example (see Picture 35).

The task management module, which process flow example is presented in Picture 35, uses quite advanced and complex workflow scheme. The mechanism of task management is complex enough itself and besides that it was enhanced with exception management possibilities, which increased the level of complexity even more. Workflow scheme originally uses several conditional sentences, which define the further flow of the scheme and the next task list. The sequence also uses two sub-processes, which point to other stand alone process, which can be called with a help of sub-process, or even can point to some earlier place of current scheme for repeating. The sub-process usage already can be called as exception management, but besides that, every sequence conditional sentence has a branch leading to exception management request task, devoted to basic exception management process execution initiation. So here, it is possible to make a conclusion that discussed task management scheme is realised with a help of embedded workflow into running workflow instance algorithm. While using embedded workflow method, it is not mandatory to stop running workflow instance, because the flow execution can be preceded using recovery and resume from the same place methods of exception handling.

Here discussed task execution example differs from the first one, especial in task assignment, which is especially complex. The *Exception* state in this situation is possible in almost every step. The task acceptance is a bit simpler, but the whole lifecycle management is stronger, as there is a clear exception states management.



Picture 34. Advanced task execution lifecycle.



Picture 35. Advanced task process execution flowchart.

Table 9 Task execution lifecycle definition.

Stage	State	Involved actors	Description
Task creation	New task	Creator/Initiator	Task is created, only execution details are defined.
Task assignment	Manager assignment	Initiator	The responsible person for a task execution is defined, the one who will coordinate the process. In this step the initiator can decide to leave the task for him to perform it, or to choose other responsible person.
	Manager assignment acceptance	Task manager	Assigned by the initiator task manager accepts or declines the prescribed task. If accept, he is responsible for the further task executor selection.
	Executor assignment	Task manager	Task manager selects the concrete executor. He also can become an executor himself.
	Executor assignment acceptance	Executor	Assigned by the task manager executor accepts or declined the prescribed task. If declined the task returns to the task manager for re-assignment.
Task execution	In progress	Executor	Executor performs instructions defined in task description. At the end, executor sends the task to the initiator for approval.
	In finish	Initiator	Initiator evaluates task execution.
	In re-execution	Executor	Initiator finds unfinished issues and returns request task re-execution.
	Task finished	Initiator	Initiator finds the execution of the task complete.
Task acceptance	Postponement adjustment	Executor	Task executor requests the postponement of this task execution due to defined reasons. It is the duty of the initiator to evaluate such request.
	Adjustment declined	Initiator	Initiator does not agree with task execution date postponement and returns the request to executor with appropriate explanation.
	Adjustment accepted	Initiator	Initiator finds the postponement request reasonable and acceptable.
Exception management	Exception	Exception manager	Exception manager receives a possible deviation or exception notice and starts handling process.

In this created workflow scheme there exists these main factors influencing the possibilities of exception emergence:

- Task performance date (as also its overdue),
- Correctly selected task executor (in what way he is selected and if correctly selected),
- The finish of the task (the reflection of real-world task state situation).

This leads to the conclusion that these main exception classes arise (identifying the application module exceptions, the identification was made excluding technical or political nature exceptions):

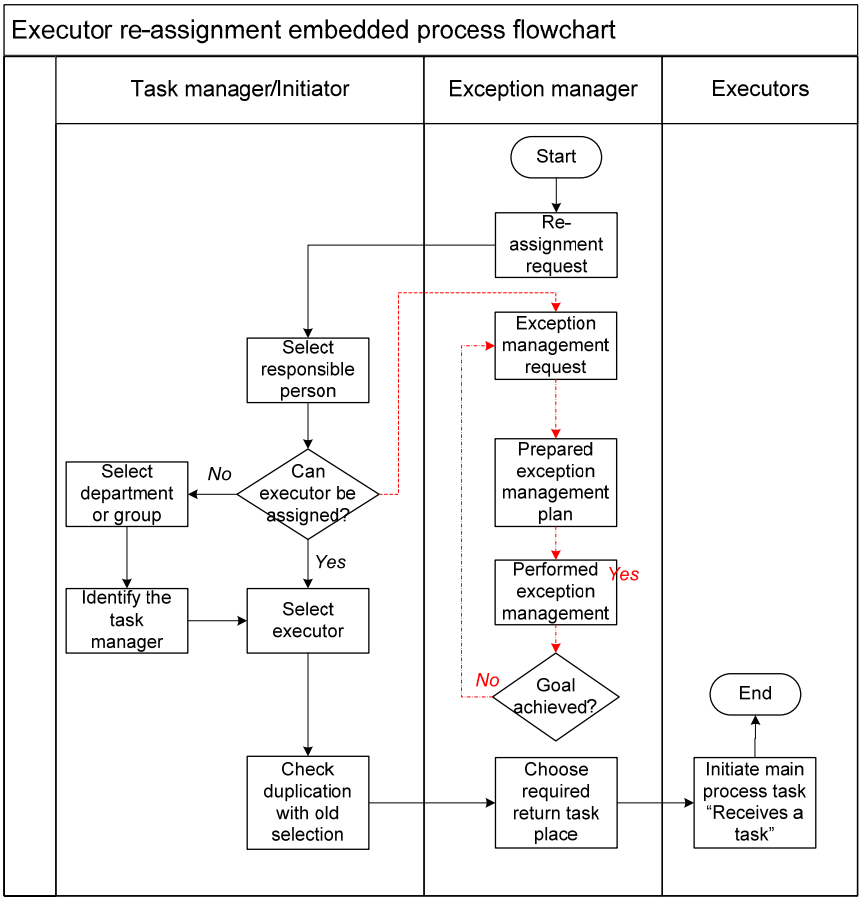
- The execution overdue,
  Possible handling – the exception is handled automatically by the system,
- The repeated executor selection,
  Possible handling – the reassignment of the executor is performed manually by the user, using embedded processes,
- The task becomes impossible to solve,
  Possible handling – the possibility to terminate the task in the current state and place is performed manually by the user or system.

The embedded process dedicated for the executor re-assignment must have its own start and finish tasks and be initiated manually by the user; using dedicated decision procedure functions in the running workflow instance (see Picture 36).

The executor re-assignment embedded process starts in a place where it is called out and finishes in a place which is indicated by the exception manager. Task manager or initiator, depending on the situation, is required to lead the re-assignment process, which goal is to detect the correct person. First of all it is required to decide how the selection will be performed, directly selecting concrete executor, or the other way, through the task manager identification, which would be competent enough to select the correct executor. After executor selection, the process itself should check for newly selected executor correctness (for example, not the same person, as already has been chosen earlier). After re-assignment, the exception manager should decide into which workflow scheme task the embedded process should resume the normal flow and inform newly selected executor.

The workflow management system itself, who will deal with such a workflow scheme described above, must have the abilities to store and interpret process and workflow definitions, create and manage their instances, also control the interactions between application and the user. The application also must posses task re-assignment to the other workflow actor mechanism, be scalable in application, process and workflow terms (also for already running instances),

maintain monitoring functions, as also administration of all the application modules in separate and together function and tools package.



Picture 36. Executor re-assignment embedded process flowchart.

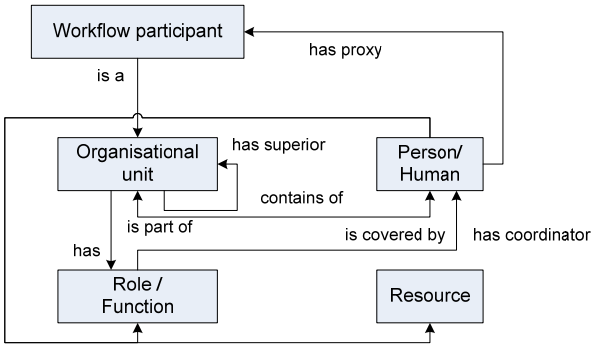
While describing the processes, it is necessary to prescribe the unique identifiers list, which later will be used for the running process activities performance identification. The described above workflow scheme already possess two unique identifiers, they are activity state and actor role.

Any workflow user can gain one of these roles according to the process context described above:

- Initiator – the person who can create the new task and also can accept or decline its executions; initiator also is capable of terminating the task if need be,
- Task manager – decides, if the assigned task executor is chosen properly for the current task and also can re-assign the user,
- Executor – is responsible for the task execution and is directly accountable to the initiator of the task.

The meta-model of workflow participants, roles, functions and resources interaction is defined by WfMC (see Picture 37). Using this model, the description of described case

organisational model was made, where organisation is defined as hierarchically accountable departmental structure. Each of departments has its own manager, which can gain initiator or task manager role. The rest of the department personnel can gain task manager or executor roles.



Picture 37. Organisational structure, used for actor roles identification [68].

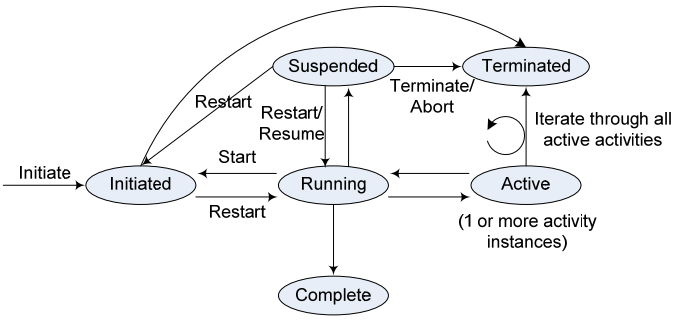
The described enhanced task management process covers these main sub-processes:

- New task registration,
- Assignment of task to task manager or executor,
- Task manager decision,
- Executor decision,
- Execution finish,
- Execution acceptance/annulment.

Correspondingly, any of these sub-processes can have these task lists:

- The declaration of execution date, the configuration of reminders/notifications sending,
- The assignment of task manager/executor, additional information declaration,
- Assigned task acceptance/re-sending/decline/re-assignment, declaration of information needed for this step,
- Assigned task acceptance/decline, declaration of information needed for this step,
- The request of task finish/annulment confirmation, declaration of information needed for this request,
- The request of task finish/annulment confirmation/decline, declaration of information about the decision.

All the task management process, after its initiation, is managed as process instance and for ideal model should reside in *Running* state (see Picture 38). The task lists, described above are treated as activity instances and reside in *Active* state and can be terminated (also iteratively) at any time. The process itself also can be suspended, if any exception of organisational or political nature occurs. If exception is of application or infrastructure nature, they should be registered handled at activity instances level.



Picture 38. Process instance state transitions meta-model [68].

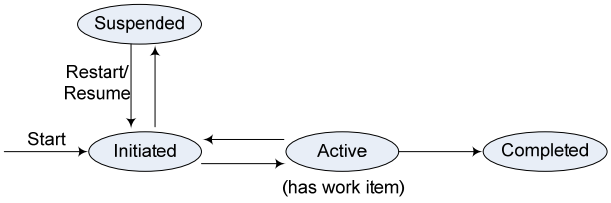
The process activities can gain various states and change each other during process instance *Active* state.

Table 10 Table of state list projected on tasks described above in task execution flowchart.

No.	Activity instance state	Workflow participant	Projected tasks
1.	Assigned for himself	Initiator	Task definition/registering, Assign for himself
2.	Send to task manager	Initiator	Send to task manager, Assign for a department, Assign for a group of persons, Identify department/group manager
3.	Task manager accepted	Task manager	Receives a task, Accept the task/Send acceptance to initiator, Choose the executor
4.	Task manager declined	Task manager	Receives a task, Decline task, Redistribute task
5.	Send to executor	Initiator, Task manager	Task definition/registering, Assign for an executor, Choose the executor
6.	Executor accepted	Executor	Receives a task, Task accepted/Send acceptance to manager
7.	Executor declined	Executor	Receives a task, Task declined
8.	Executor finished task	Executor	Task finish approve request
9.	Executor requested annulment	Executor	Annulment request
10.	Initiator declined	Initiator	Request declined
11.	Task finished	Executor	Task finish approve request
12.	Task finally finished	Initiator, task executor	Task finished
13.	Task annulled	Initiator, task executor	Task finished

The meta-model of any activity instance management is defined also by WfMC (see Picture 39). The model clearly notes that any activity during its initiation can become active or suspended. The active state usually leads to a completed state, while suspended would usually note the possible exceptional situation. If the activity is declared *Suspended*, the exception

management system should consider that particular task as the exception source and diagnosis all relative to this activities.



Picture 39. Activity instance state transitions meta-model [68].

The predicted and known exceptions in this task management scenario can be execution date overdue or wrongly assigned task executor. Each of these exceptions can be specified using P_i in ***(n, t, f, e, p)*** format:

- ***Execution date overdue (n, t, f, e, p)***, where:
  - ***n*** – current date,
  - ***t*** – defined execution date,
  - ***f*** – result: today's date is older than defined execution date,
  - ***e*** – reminder/notification sending mechanism,
  - ***p*** – the execution is resumed from the same workflow instance,
- ***Wrong executor (n, t, f, e, p)***, where:
  - ***n*** – wrongly selected executor,
  - ***t*** – defined executor,
  - ***f*** – defined executor can not perform/accept the task,
  - ***e*** – embedded sub-process,
  - ***p*** – the execution can be resumed from the same workflow instance or roll back to selected activity would be needed.

The deviations/exceptions of the process instance can be also the change of task definitions, the change of task execution date, the change of executor (sickness, mission, does not work for a company anymore).

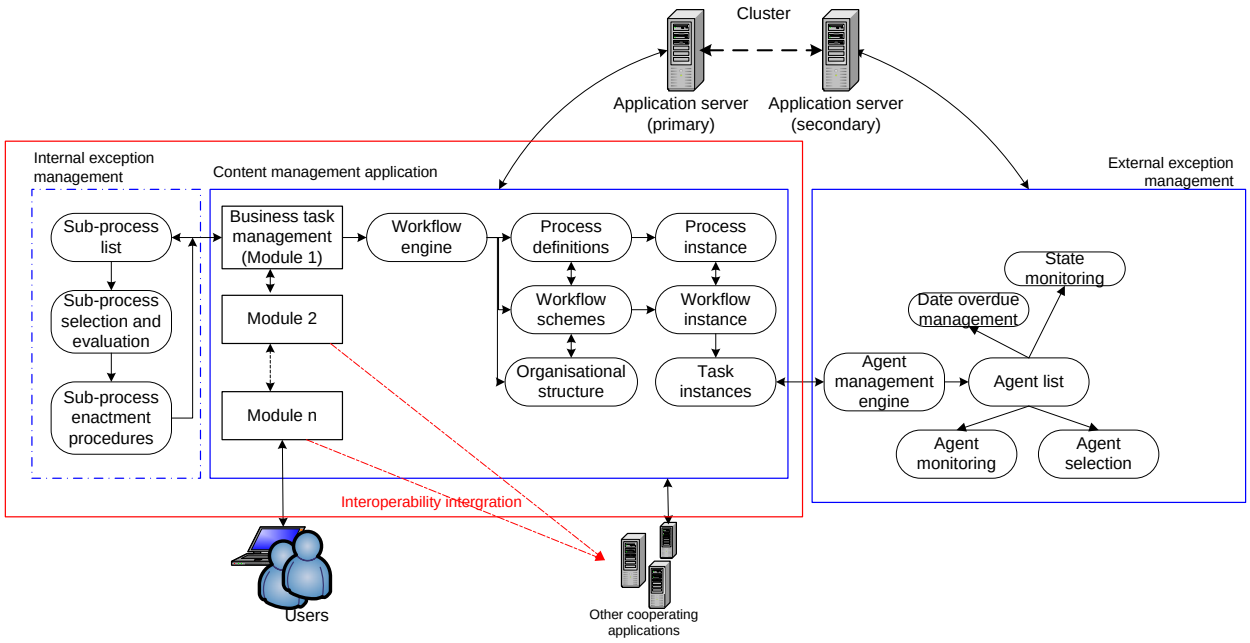
After the change of task execution date, the process can be resumed from the same place with appropriate information notices for related executors, or the task can be terminated and the new task process instance can be registered with new task and date definitions. The task process instance also can be rolled back to the certain place with a help of embedded sub-process, if a task has not yet been accepted by the prescribed executor.

After the change of task definition, are possible the same handling scenarios as in previous, execution date change, case.

If the task executor changes, is missing or sick, the other, substituting person, is assigned for current task. In such case, the task can be suspended and resumed from the same process instance place, right after the return of the original executor.

If the task executor is no longer a worker of the company, his former task should be terminated, but later a new ones can be initiated on the basis of these tasks, describing new task definition, execution date and executors. The other choice is to suspend the task until the executor re-assignment embedded sub-process finishes, and to resume only when the sub-process returns results to the main process instance.

Enhanced task management module was created on the basis of architecture presented in Picture 40. The architecture is also three-tier type but supplemented with interoperability features.



Picture 40. Enhanced application architecture.

The enhanced task management architecture has two exception management modules: internal and external. Internal exception management is fully workflow based and manages embedded processes library. Sub-process evaluation and enactment procedures uses content management application workflow engine, therefore these system parts have reliable communication and data exchange between them. The external exception management module is fully self standing and cooperates with main application using interoperability integrations (inbox/outbox message exchange scheme). Also main application can have integrations with other self standing applications but these applications can not communicate with external exception management module. For better performance and reliability, application is stored in

two application servers, one of which is primary and replicates all the data to the secondary, which is used in case of primary server crash.

3.3 Business Process Business-to-Client Exception Handling Realisation

Exception management model combining knowledge management and intelligent problem solving methods

The analysis of existing exception handling methods and its testing in real environment leads to a conclusion that each of them is suitable and effective in particular situations and in limited exception range. The assumption rises that it would be reasonable to use every single positive aspects of every method, and combining them to model and create general purpose exception handling method.

The example, which was realised in this research, shows how such methods as knowledge management and intelligent problem solving were combined. The last one is especially effective in a context of fully automated systems, but sometimes inevitable is also the external factors interruption, to handle the problem in a rational way. The first of mentioned methods must be used to fill up the gap of intelligent problem solving method, as firstly mentioned method allows coordinating executed actions, which is a very valuable function. The system coordinator would help very much in the interface between the user and the exception handling mechanism [48].

As earlier was mentioned, it is possible to exclude application, workflow and infrastructure level exception classes.

As infrastructure level exceptions are of totally different nature than the left two, in this example, infrastructure level exceptions will not be discussed. The concentration will be focused on workflow and application level exceptions. The effective and rational solution realisation for application and workflow exceptions management would also lead to a possible flexible infrastructure exception handling.

Further, for a flexible exception management solution finding, it is important to divide exceptions into solvable and non-solvable. As real-world situations analysis showed, these two groups are closely related, as exception is usually handled if it is detected. The moment of detection is mostly important in overall exception handling process. The detection moment can be divided into such types:

- The exception was noticed and detected by the user,
- The exception was detected and registered by the system at any place,

- The exception was detected and registered by the system at real exception occurrence place (this is the to-be model for a solution described here).

The good exception handling system should be able to handle all types of exception detection and incorporate best practices of exceptional situation detection. The moment when exception truly occurs is always mostly important, because in such a way it is possible to avoid some consequences made on running working workflow instance. That is why for such requirement realisation, there should be a very strong system, running monitoring and unexpected situation registering tools. From the other point of view, when the exception is noticed by the user, system can validate the ordinary normal flow and execution; in that case it is business logic, political or organisational nature exception (such as wrongly assigned task executor) [55, 47].

The modelling of the exception management system

While modelling a flexible and self standing, reliable exception handling solutions, it is necessary to define main steps of actions which will also be delivered in this research:

- To create the table of goals (which will indicate to-be results),
- To create the table of problems and possible their solutions (indicating what should be avoided and how to solve avoided situations),
- To create the algorithm of exception handling (the process model how the handling should be achieved),
- To define possible exception handlers models,
- To model the system architecture (technical reflection of conceptual model).

Common goals and tasks

Firstly, it is possible to define these common goals, which are raised for a flexible and self-standing exception handling system:

- The possibility of exception management coordination – there should be always a possibility to assist a user and monitor the system executed flow of tasks,
- The possibility of storage of the exception cases – with a purpose of extraction and reuse from exception specifications repository,
- The possibility of storing exception handling cases – with a purpose to reuse and improve existing handlers,
- The ability of the system to make decisions for itself on exception handling - creating the possibility for a system to learn and evolution,

- Usage of flexible user interface – the possibility to empower the user to make decisions on exception handling for himself,
- Events (of the process flow) results storage – the possibility to monitor and analyse the entire systematically executed step of process execution.

Having the goals of system modelling, it is possible to list them in detail using a matrices of goals and issues. This help to decide which of the raised goals are most important and which are critical to the system modelling expected results.

Table 11 The matrices of goals and critical issues.

The goals	Tolerance for small exceptions/ deviations	Detection	Avoidance	Possible model changes
Interactive execution (meta-data, behaviour, processes)	**X**	**X**	**X**	
Flexible resource management	**X**	**X**	**X**	
Process logical decomposition	**X**	**X**	**X**	
Process evolution				**X**
Reusable specification repositories				**X**
Monitoring, audit and analysis interactive instruments		**X**	**X**	
Change management possibilities	**X**			**X**
User and system collaboration and cooperation	**X**	**X**	**X**	
Coordination	**X**	**X**		**X**
Interoperability		**X**	**X**	**X**

The table indicates that the biggest possibilities for running workflow model changes are influenced especially by the features which are specific and important to exception management. That means the design of these features requires considering the balance of achieving flexibility. Here is very important qualitative achievement of interoperability which should act as an instrument for balance adjustment.

It is also obvious that the tolerance of small exceptions/deviations is quite easily achievable through most of the goals, which means, the goals are preferable. Also it is noticeable that detection and avoidance features mostly comes together and can be achieved through the same goals. That leads to the assumption, that goals are raised in right direction.

The problems and their elimination possibilities

Having the goals and their priorities it is possible and useful to analyse, what effects will be observed while implementing constructed solution for exception handling. That will help to avoid system limitations and need for complexity. Also this analysis can help to solve the questions, which are already raised before constructing solution, and addressed as the bottlenecks of solution.

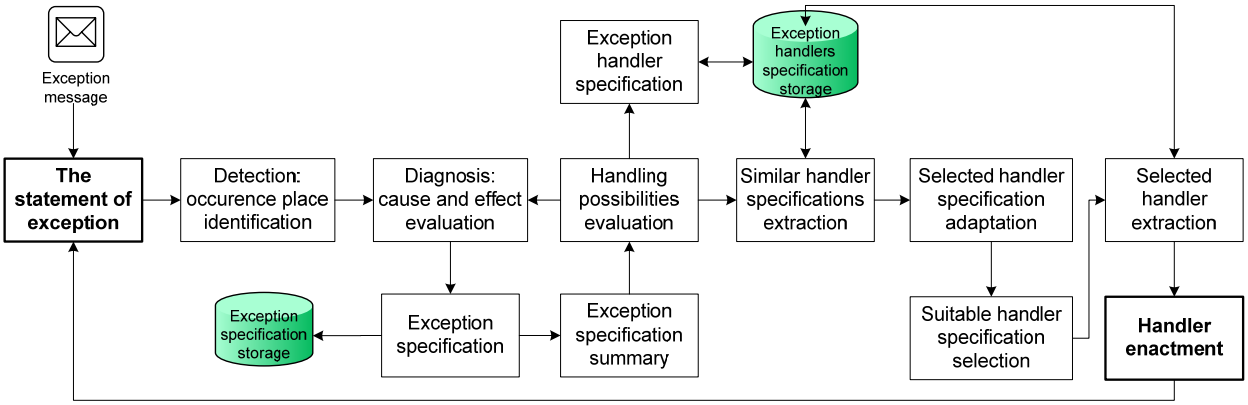
Table 12 The problems, their solutions and limitations.

No.	Problem	Solution	Effects on
1.	Events archival in repositories	Overall application (preferable including business process system) decomposition into functional modules	System performance and integrity
2.	Exception events register	Flexible user interface and integrations, coordination possibilities	User work performance and data precision
3.	Specifications (the difference of formats, syntax and interpretation)	The template for data gathering, description and presentation (including formats and style)	Specification precision
4.	The exception detection	The monitoring of events, its analysis and control functions	System performance
5.	The handling of the exception	Knowledge repository management	System performance and integrity
6.	Solution adaptation	Flexibility of user interface and coordination possibilities	System performance and correctness
7.	Flexible user interface	System scalability and interoperability	The price
8.	The price of the solution	System standardisation	Upgrade possibility

It is obvious that the main and critical effects of problems solution can be observed on the overall system performance. Before applying possible solutions, it would be valuable firstly to evaluate their effect on every system module and overall execution effectiveness against all application contexts. The other intriguing factor is the price, which basically does not influence the correction and effectiveness of the solution, but brings to the conclusion, that the realisation of the solution will require quite many resources and budget, which is not undesirable, as it later influences especially the price of support and upgrades.

The exception handling scheme

The scheme of exception handling should not rely on any technologies, but can be fulfilled using different technologies. The offered scheme is constructed using meta-model of exception handling: detection, diagnosis and handling, supplementing it with intermediate steps, dedicated to information gathering and approbation.



Picture 41. B2C exception handling scheme.

The statement of the exception is a place in which unavoidably certain decisions and flow changes must be made. After the statement of such fact, mostly important step becomes the identification of exception emergence causes, which later will determine the identification of the appropriate handling solution. The other important moment in this scheme, indicating why the identification of causes must be performed right after the statement of the problem, is the data, which later unavoidably will change and possibly make heavier the identification of exception causes. In this place, the most helpful would be events data gathering repository.

After cause's identification that is gathering of sufficiently exception raise explanatory data, it is possible to create an exception handling specification, which further on will be saved in a specialised repository.

The next step is to make the analysis of the specification and define the handling possibilities list. Any of these methods should be analysed in detail, deciding if it conforms to the requirements raised by the exception specification and if this method will be supplied with enough data for constructing the appropriate handler. As it is observed in the scheme (see Picture 41), this step can have a return connection to exception identification step. This can happen if during the handling solution searching process the requirement for additional data raises.

In that case, the next step is to create the specification of the handler, corresponding to the requirements held by the exception specification and suggesting the idea on handling scheme. This specification also should be included into specialised repository.

While adjusting the selected existing solution, system must further monitor its execution, as handler by itself can also raise the other exception. That is why the first and the last steps of the scheme are connected and this makes the exception handling scheme recurrent. If the last step indicates to the first step instead of its retiring, that means, that mistakes and misjudgements were made in exception handler selection and adaptation. Constant business application and integrated exception handling monitoring should ensure overall accurate execution and effective exception detection.

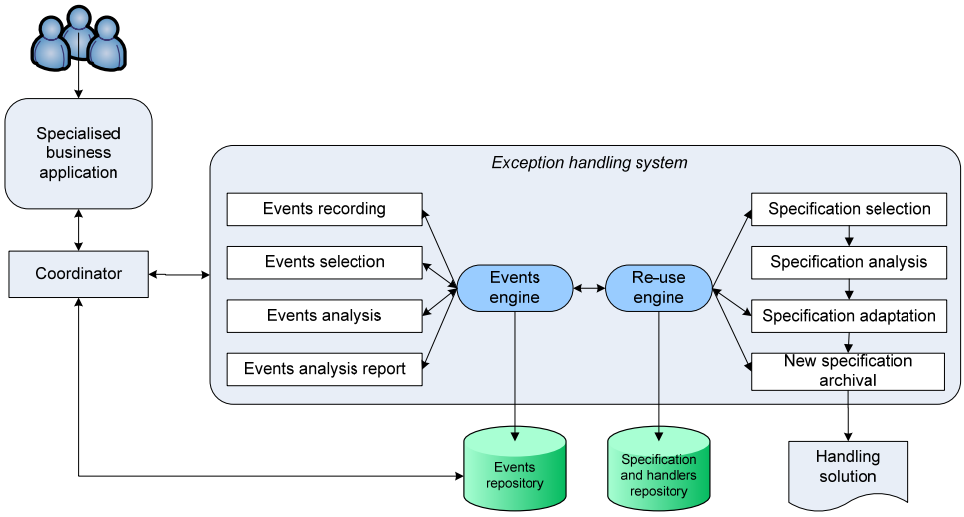
Conceptual model of the architecture

Any business application is a place where exceptions can arise and, for their solution, the last components of the architecture are dedicated. All application events are of course recorded into events repository.

The event repository purpose is:

- To register every single step made in the application,
- To gather the most accurate information about every of the steps and events,
- To analyse the gathered information and report their limitations,
- On demand, use this information for the recovering and simulating the exception moment.

Events repository should collaborate with business application thought the predefined interfaces and be totally automated, that means the user should have no clue about such a repository functioning existence. With a help of this database, later on the while handling the exception, it would be possible to concentrate on exception emergence moment, but not on its detection moment. In such a way, the recorded information of the exception would be more precise and useful.



Picture 42. B2C application exception handling conceptual architecture.

The selection component of events repository should be initiated by the coordinator or by the exception handling system. The selection component purpose is to select all possible stored and related to emerged exception data, which later would be possible to analyse and present the report of their summary. That leads to the conclusion that the engine of the repository should be constructed of events selection, analysis and report creation components.

The engine of the exception management should consist of four elements collaborating with each other:

- *Exception specification selection*. This component, according to newly encountered exception specification, should select all similar exception descriptions, which would satisfy the degree or criteria of given specification, from the specialised repository,
- *Exception specification analysis*. After the selection of group of similar exception descriptions, further it is necessary to perform the analysis of them in detail, which should identify each of these similar exceptions similarity degree. The similarity degree identification process should be performed using these criteria:
    - The similarity of each exception description,
    - The similarity of used workflow,
    - The similarity of the context,
    - The similarity of observed results of exception.

  The degree of exception similarity, which is calculated selecting particular attributes collection according to given context, is calculated using a formula, already discussed in previous section:

$$S = \sqrt{\sum_{n} wi \times (ai - bi) \times (ai - bi)} \qquad (12)$$

  In this formula:

  w_i is i^{th} the weight of the attribute,

  a_i is i^{th} attribute in an example a,

  b_i is i^{th} attribute in an example b,

  $(a_i\text{-}b_i)$ is a similarity of attributes a_i and b_i.

- *Exception specification adaptation.* Further on, the most similar to the given case exception example, which data and information mostly corresponds to required, is selected to serve as a basis for handling the raised exception. This exception example should be modified, using specialised exception specification editor, and adapted to the current case. In such a way the similar, but new exception specification is created.

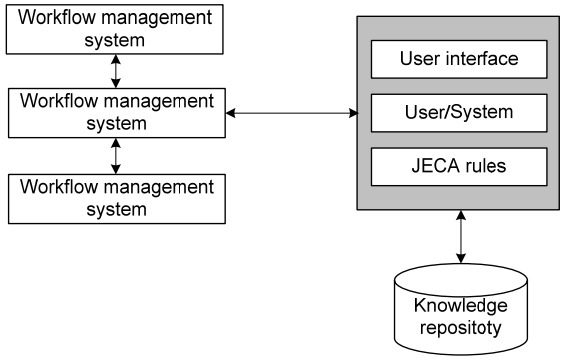
- *Exception specification archival.* The newly created exception specification should be archived into specialised repository for future re-use and included into the common taxonomy.

The overall exception management engine should present the coordinator with a complete exception specification with a list of possible handling methods, which also is a part of exception specification. This specification from the gathered and modified data should be generated by engine managed analyser and present it to the handling mechanism as input data. The analyser's response in exception recording and analysis module should be considered as the output of this module.

Coordinator role

Coordinator can be totally automated or semi-automated, which makes decisions only in part and the most important decisions are made by human. The automated coordinator is an ideal conception in overall exception handling and would be the most wanted solution, but due to the exception real world influence nature and unpredicted human mind results, it is always more optimised solution to use semi-automated coordinator and leave a possibility for a human to make the last review of the handling case before it is enacted. The coordinator conceptual architecture is presented in Picture 43.

Actually, it is an interface between the user and the exception handling system together with events recording mechanism. The coordinator itself could be semi-automated system managed by the user himself. In such a way, the user could call out the interface, which connects him to the coordinator, and would present the user all the possible and needed information at the current moment. With a help of this interface, the user could define, in what manner the system should further behave and manage the recorded exception.



Picture 43. The conceptual model of system coordinator.

The coordinator should have the direct connection to the exception specifications and their handlers' specialised repository also. The logic and behaviour of the coordinator can be created

and realised using JECA rules, which allows, according to the given situation, to manage the exception as for the system, as for the user himself.

JECA rules primary purpose is to model the logics of the business process task according to given context; in such a way the possibility to manage the concrete organisation events is given [72].

ECA rules would not satisfy the raised requirements as they act not on the basis of the context evaluation but on the conditions, which must be satisfied, in order the defined action of the rule should be performed. After enabling of ECA rule, it can be suspended only if any of its conditions is not met. That is why these rules have limitations for unexpected business environment.

The attribute ***J*** of JECA rule describes the context information, according to which the ECA rule is selected, which is capable of supporting the context dependable processes.

JECA rule is defined as ***r(j, e, c, a)***, where [72]:

j – is a context, according to which the JECA is evaluated. If ***j*** is evaluated to *true*, then JECA will not be performed,

e – the event, on which occurrence the JECA is evaluated,

c – the logical string, which should be met in order to complete the rule, if the rule itself is disqualified,

a – the necessary actions, which must be taken, if a rule is not disqualified.

That means that the coordinator in fact should be a JECA rules collection, which defines the further system work. The coordinator's conception using JECA rules, can be defined in such a manner:

As long as there is a possibility to activate JECA rules:

1. *find the activated JECA rule* ***r,***
2. *evaluate the condition of* ***r*** *and its context,*
3. *if the condition of* ***r*** *is true and the context is not true, the rule* ***r*** *must be performed.*

For example:

The event: the exception encountered in the system,

The condition: the factor of risks – the damaged process or its data,

The action: to specify the exception and define the handler,

The context: the data of the context correspond to the encountered exception.

In fact, there can exist by default defined conditions and actions, which would be enacted after occurrence of recognisable exception. That leads to the conclusion that JECA rules can

make decision: in what cases the user should be involved, and in what cases the system can manage exception itself.

The conceptual model of the handler

The handler is a self standing procedure or mechanism capable of returning the workflow to its normal flow and handling the damage made by the exception.

It is possible to define these common handler behaviour types:

- rollback (to the process or the task, from which the flow should be repeated),
- resume (continue from the place, where the flow was stopped),
- terminate (to cancel all the instance of the process and start the new workflow instance).

According to the intelligent problem handling method, the exception management system should have a specialised repository of the exception handlers, which would be freely available as to coordinator, as to exception specification mechanism.

In order to create a proper handler for an occurred exception handling, it is necessary to (see Picture 44):

- to create a specification of the handler,

the primary data for specification creation is exception specification report summary data, which is presented by the analysis tool as the result and output of the exception specification selection and adaptation result – the output of the system,

- to select the existing similar handler specifications from the repository,

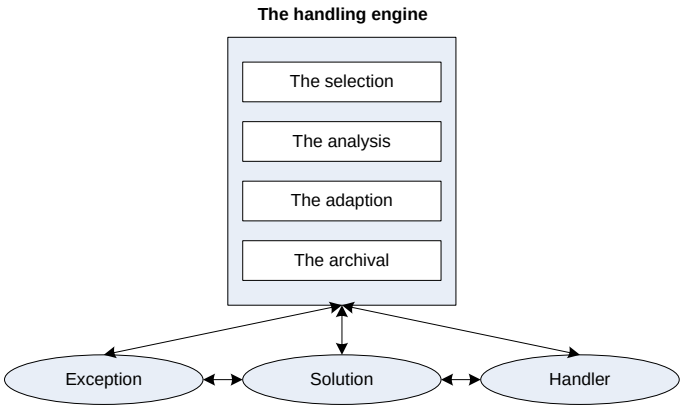
the handler is used in the same environment as in which the exception was encountered, the similarities should be evaluated the same as in exception case: workflow, context and handler,

- to adapt the most similar existing handler specification to the given case,

the adaptation of the existing handler specification should be automated as much as it is possible and managed by the coordinator. For the adaptation of the handler, the specialised editor should be modelled and used, which could be available for the user as a separate interface,

- to pass the handler to the exception management system for enactment.

the execution of the handler should be monitored by events analyser, as the adapted solution can raise the new exception. The execution should be documented by recording every event corresponding to exception handling process.



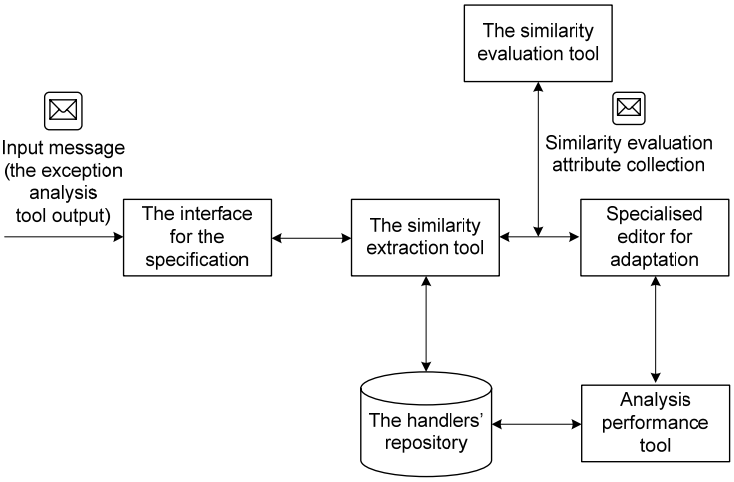
Picture 44. The conceptual model of the handler mechanism.

The physical model of the handler

The physical model of the handler architecture is quite similar to exception specification management engine model. The only difference is that handler management model requires managing not only exception handler specifications but also the handlers itself, that means procedures, agents, functions, fixing workflow and other elements.

As the exception handling engine is separated from exception specification engine at least logically, they should exchange information and data using input and output messages through dedicated inboxes. The most important data for exception handler management engine is exception specification generated by exception specification management engine. This specification data report is passed to handler engine as input data using predefined message format. Using this data the primary exception handler specification is generated, which later is finished having required specific data.

Handler management engine transforms received exception and handler specification into query, used for similar existing handlers search in specialised handlers' repository.



Picture 45. The physical model of the handler.

The received search results are then passed to similarity extraction tool, which evaluates received handlers specification according to selected priority, attributes collection, which can change each search. Usually similarity attributes include at least workflow type, expected results, type of exception and results of exception attributes, which are supplemented with additional context definition attributes. The most similar existing example in repository is selected and modified to fulfil raised requirements for current case. Newly created handler specification and the handler itself are stored in the repository. As exception handler specification as the handler itself is modified using predefined embedded into the system interfaces. The interfaces are used also by humans if handling process is semi-automated and involves human decisions. The requirement of human involvement is announced using mail messages to key persons.

It is also very important to notice, that exception handling specification also includes information on handlers enactment place and the overall handling procedure itself and after handler's enactment it is used as execution logic for the handler processing.

The exception handling architecture

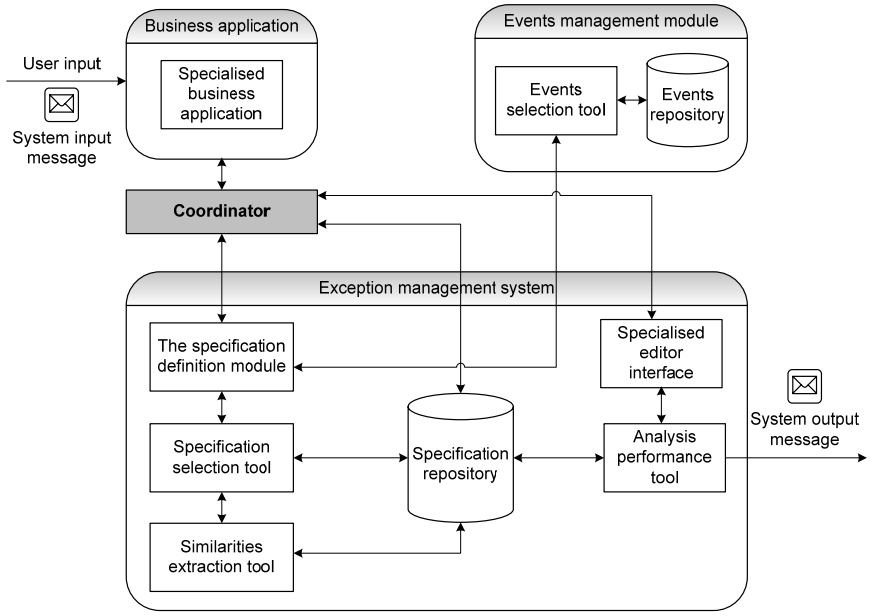
The architecture of the exception handling system should be based on three tier model and be constructed of three self-standing modules, which should collaborate between each other with a help of a coordinator. The component modules should be these:

- Specialised business application, dedicated to the user for the specific business actions and task performance,
- The events repository, supplied with additional interfaces for communication with business application and coordinator,
- Exception handling management module, a self-standing system, which can handle exceptions by itself, automated way, and also possesses the interfaces with a business application and coordinator for semi-automated handling fulfilment.

The coordinator is also a module of the exception handling system and is presented as a separate interface. The main communication method of system coordinator and human coordinator, the key person responsible for making final handling decisions, is specialised messages, received into human and system possessed inboxes. The messages already contain main informative information on occurred situation and links to exception specification, handler specification, handler itself and of course the action plan of handling.

Each of specification aggregation process closely relies on events repository contents and its analysis. After each exception encountering the exception statement must be validated and

diagnosed against facts stored in events logs. The analysis of these logs makes it possible to approve or re-define the place of exception occurrence, the damage and expected results.



Picture 46. Exception handling system architecture.

Each exception handling system module and element was already discussed above. Therefore, it is still necessary to note that coordinator plays main role in overall handling process, as every decision accepted for specification approval or handling method applying is performed in by the coordinator. Also coordinator is in part responsible for exception handling process monitoring and evaluation, in case to avoid recurrent exception phenomenon, when one exception raised other exception.

Research results analysis

The following results were collected during the implemented production system usage. The results were collected for about the half a year and classified according to exception types and their handling phases. The results summary was generated using the exception specifications, which were automatically, collected when exception occurred in business application, or when the users reported them manually.

The collected results were useful not only for analysis purposes, but also were used as a very detailed monitoring reports, which delivered the results of exception handling according to various selected exception or handler specification attributes.

Table 13 The summary of research results in percentage.

Exception type	Human/system detectable	Handled using existing handlers	Changes working process model
Non predicted execution of workflow branch	80 %	5 %	50 %
Wrong execution of process lifecycle	95 %	90 %	5 %
Not correct input or output data for execution	50 %	95 %	5 %
Human resource lack for process tasks execution	80 %	70 %	20 %
The lack of system usage knowledge	60 %	10 %	5 %

The implemented in real environment and tested B2C model results are delivered in Table 13. The table depicts that according to exception type, the possibilities of its detection diagnosis and handling are different, and the requirement to change the process model can vary from 5 to50 percent. The results shows that created exception handling module usage can noticeably increase the possibility to handle the human factor exception, which is a very important feature of the whole exception handling process..

The performed analysis results proves, that exception detection becomes possible in 60-95% of exception cases, their management possibilities, using knowledge management tools, grows to 95%, and the impact to the working business process model can be minimised from 5% to 50% depending on exception type.

3.4 Conclusions

1. The three discussed examples of exception handling, which were realised during this research, depicture the different types of organisations, having different infrastructure, which in part dictates possible deviations/exceptions classes. Therefore, the conclusion is made, that the environment of process execution plays a great role in exception types scale creation and variation.
2. Conditional sentences usage in B2C application exception handling is possible, but carries quite enough of risks in its fulfilment, as production processes (and B2C workflow is prescribed to production workflow type) tend to raise faults and troubles, which can not be predicted.
3. The second of examples of handling exceptions grants more freedom to the user, and in part, makes possible the coordinated exception management. However, the process

exception handling is possible only with a help of conditional sentences in this case. So the programmer's involvement becomes unavoidable.

4. The third example is the most universal, as it possesses the knowledge, stored in knowledge repository, which is managed by the coordinator (in critical cases by simple user) and is organised as self-standing application. The advantage of self-standing exception handling application is its resistance to possible business application logics and user working ways change and redesign.

4 Business Process Business-to-Business Exception Handling

4.1 Introduction

Business processes are market-centred descriptions of an organisation's activities, implemented as information processes and/or material processes [17]. E-business realisation is based on strategic and leadership goals, which define the company's place in the market and its ability 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. B2B applications 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 [23, 4]. The level of mutual benefit can also depend on the type of B2B marketplace. Vertical e-marketplace is designed to expand and share knowledge of every segment of one specific industry and in such a way can increase the collaboration across sellers and buyers, while horizontal e-marketplace connects buyers and sellers across many industries and supply them with maintenance, repair and operations materials. Both of these marketplaces serve one purpose (to unite the buyer and the seller), while realisation of the goal is achieved in different complex ways.

Not less important in this area of discussion is e-government sector, as it is closely related to e-business nature and specifics, only realised in different environment, which once in a while can cross. Most related e-government types to current topic is G2B (government-to-business) and G2C (government-to-citizen). G2B is a relative branch of e-business and shares a collection of B2B common goals. eEurope initiated initiatives on creating the benchmarking of European public services has shown that integration of e-services improves the level of interaction between government and citizen/business and shortens the completion of service delivery lifecycle [9, 36].

What also connect e-government and e-business is also the requirement to improve instead of replacing the current processes, which the aim to achieve [1]:

- The support of required tools for the process complexity realisation simplification,
- The possibility to extend the margins of system and process integration,
- The improvement of the level of complexity of process supporting information,
- The demand for higher volume of transactions,
- The ability to increase the duration of relationship between parties,

- The demand of the process evolution realisation and support,
- The possibility to bid higher transaction value,
- The need to comply with the legal nature of transactions.

Besides these first level aims, there is one other not less important aim – to support the process exception handling. This aim is usually raised from the back of the e-business of e-government application, but plays the most significant role in achieving the first level aims.

The exception handling in B2B and G2B application has not been addressed as an important issue yet, because the applications and realisation tools are quite young and are still improving to gain the main functionality in order to provide the possibility to improve B2B and G2B environments. The paradox is that the improvement of the technologies raises the need for exception, raised by these technologies, handling.

In this final section, the possible exception handling model for B2B and G2B environments will be presented and discussed, which was achieved while combining the best practises of process management methods and strong sides of different Web technologies.

4.2 Business Process Business-to-Business technologies

Using Petri-Nets and Work Flow Nets

Any process which is managed electronically consists of process definition and its procedure for execution. Workflow management system has a set of tools for interpretation of the definition according to a given procedure. For that purpose various workflow management systems can use various languages or technologies. One of them is Petri-Nets, which describes the process in terms of places, transitions and arcs. While in commercial systems Petri Nets are rarely used as a method, there are several strong reasons for using it as workflow modelling language [58]:

- Formal semantics – which helps to build a clear and precise definition of the process,
- Graphical nature – that helps to make the process intuitive and easy to learn, also insures the communication with end-users,
- Expressiveness – support all primitives needed to model a workflow process,
- Properties – the mathematical foundation makes it possible to be interpreted by many technologies,
- Analysis – can be used to prove properties and calculate performance measures,

- Vendor independent – is not based on specific software package.

Definition 16 ***Petri Net -*** *is a triple* ***(P, T, F)****, where* ***P*** *is a finite set of places,* ***T*** *is a finite set of transitions* ***($P \cap T = \emptyset$)*** *and* ***$F \subseteq (P \times T) \cup (T \times P)$*** *is a set of arcs (flow relation) [57].*

The nature of Petri Nets is full of advantages and has a strong set of formulations and procedures in order to construct any business process, which also will fulfil the main requirement soundness (discussed in [59] and [6]). The author of [59, 58, 56] proves that Petri-Nets can even be used for handling change in processes, but also discusses the main problems which arise while using Petri-Nets in order to construct business processes:

- The complexity of deciding soundness,
- The problem of calculation the differences/commonalities of processes,
- Which of workflow patterns are essential,
- The problem while comparing different vendor workflow management systems,
- Real world features integration into process schema or extraction from its logs,
- The migration of changes into production process instances,
- The possibility to predict with a help of knowledge reuse,
- The possibility to cope with parallelism and non-phase type distributions,
- How to optimise performance according to recourse allocation.

An optimised version of Petri-Net, to suite business process construction need, while being understood by workflow management systems, was created and named WF-Net.

Definition 17 *A Petri net* ***PN = (P; T; F)*** *is a WF-net (Workflow net) if and only if (i)* ***PN*** *has two special places:* ***i*** *and* ***o****. Place* ***i*** *is a source place:* ***$\bullet i = \emptyset$****. Place* ***o*** *is a sink place:* ***$o \bullet = \emptyset$****; (ii) If we add a transition* ***t**** *to* ***PN*** *which connects place* ***o*** *with* ***i****, then the resulting Petri net is strongly connected [57].*

While WF-Net is highly qualified for constructions of strongly sound processes which can also be verified using the same language, only few workflow management systems uses this language as a basis. Most of these systems are of scientific nature and lack the real world features integration possibility, and only COSA (commercial system) constructions can be comparable to WF-Net.

Still the main weakness of this net is its construction and usage complexity. Any designer who would like to use it for constructing the processes or handling exceptions, should learn to use it before in detail, also the constructions of WF-Nets are not presented to the user in the readable and understandable form. For this purpose many commercial products are built on workflow schemas, which are easy to use for both, designer and the end user.

Service Oriented Architecture

The described above Petri-Net method is high level and complex language for business process schema and procedure creation, but it lacks physical environment for its execution. For that purpose the new technology called Service Oriented Architecture (SOA) was created and still being improved.

Definition 18 *Service Oriented Architecture (SOA) – is a software design approach that allows enterprises to focus on business processes in their application development, rather than focusing at a lower level on integration or application issues [7].*

SOA evolved from procedural/unstructured code to object oriented/structured code. The technology itself is based on server/client at least 3 tier architecture, with an ability to add N(th) tier for communication insurance between applications. This solution is highly valuable as is leaves the open space for any application or technology integration and multi-communication insurance. Also SOA is ubiquitous and heterogeneous system of a set of distributed objects, which are aggregated into logical components, which achieve specific functionality and create interfaces to those components [32, 46].

SOA platform main characteristics are also much related to business process execution core requirements [63]:

- Loosely coupled – provides ability to redesign business processes in response to raised demand, insures better flexibility increasing IT adaptability, while lowering implementation costs,
- Modular approach – enables incremental development, deployment and maintenance,
- Non-intrusive – lowers risk and development effort, avoids the need to rewrite and test existing applications,
- Standards-based – platform independent, allowing to use the software of the choice,
- General purpose technology – delivers economies of scale, can address a broad range of business problems.

Business integration solution is one of the key solutions, allowing companies to fulfil B2B requirements and migrate their legacy systems to SOA, which downstream business integration initiatives through reusability and reduced cost and complexity resulting from standardization.

The same SOA platform can be used for business process management, which also involves business process design, description and execution procedure construction, simulation and execution management, which can also be supplemented with business activity monitoring solution.

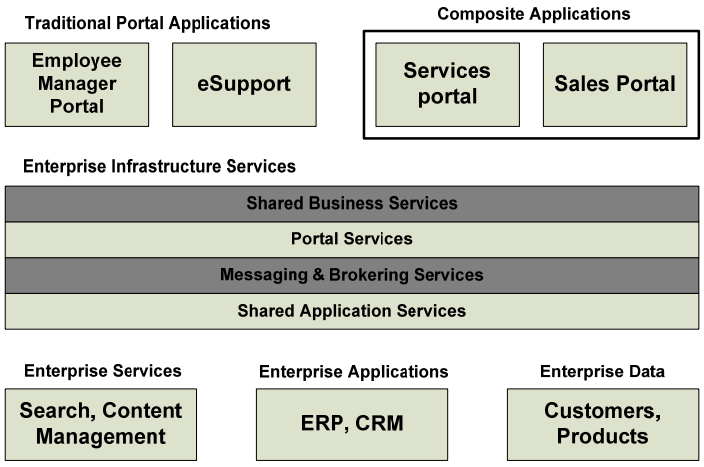
Reference model

The reference model of SOA is a template for representing system services, design requirements and principles. The core SOA platform services consist of [37]:

- Deployment services – to host and manage the operational , runtime functions,
- Integrations services – data and process integration across operating environments and platforms with inter-operability addressing issues,
- Process orchestration – service organization into flows to automate system and business process,
- Policy – business rules conception appliance,
- State management – lifecycle management capabilities, for state recognition, support and management.

The supporting SOA platform services are following [37]:

- Access support – people-based access to services and system artefacts,
- Development facilities – full life cycle support and versioning of services and messages, including all other required development functions,
- Security and management services – service and process monitoring, management, security and ID management,
- Application and data services – services built around the supporting data.



Picture 47. High level SOA solution model [7].

As it is seen in Picture 47, SOA architecture covers all three traditional 3 tier architecture layers, spreading the key business management addressing elements through all of them. Also it

can be noted, that SOA tends to split traditional infrastructure level into two part, managed separately, infrastructure and integration, which makes this platform more flexible for changes.

Table 14 Six Challenges of SOA [46].

No.	Challenge	Dimension
1.	Alignment	Business versus technology
2.	Flexibility	Adaptation versus adaptability
3.	Granularity	Large versus small
4.	Design Orientation	Top-down versus bottom-up
5.	Use/Reuse	Supply versus demand
6.	Geometry	Differentiation versus integration

Besides the great advantages which are already realised in SOA, it also raises future challenges as for the platform, as for the business process design methods (see

Table 14).

Gartner researches also identify some benefits, which in real are not realised yet. It would be simple software engineering, free integration or interoperability, technology independence, vendor independence and ultimate architecture for the modern enterprise [31].

And finally, one more advantage to SOA was gained through legislation correspondence. It is corresponding with the most spread and required regulation standards in world, such as Sarbanes-Oxley, Basel II, HIPAA and Patriot Act [37].

Business process modelling industrial standards

Business process initially is meant to cover several organisations, which can be heterogeneous in business and in technological nature. The most important feature of any business process is inter-operability and it can be achieved only through standards usage, as we need a way to make different technologies and businesses comprehend each other.

Table 15 BPM standards [66].

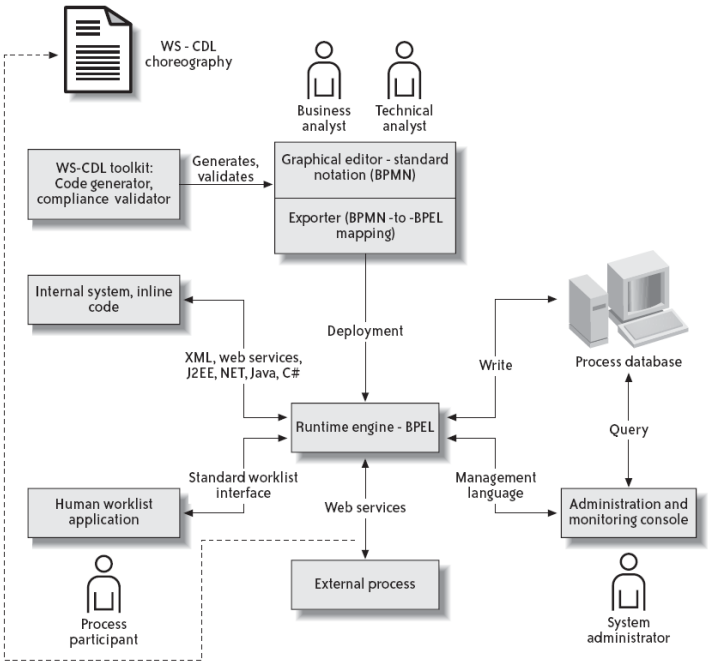
No.	Name	Organization	Type
1.	Workflow Reference Model	WfMC	Architectural model
2.	Business Process Modelling Notation (BPMN)	BPMI	Notation language
3.	UML Activity Diagram	OMG	Notation language
4.	Business Process Execution Language	OASIS	Execution language
5.	Business Process Modelling Language	BPMI	Execution language
6.	XML Process Definition Language (XPDL)	WfMC	Execution language
7.	XLANG	Microsoft	Execution language
8.	Web Service Flow Language (WSFL)	IBM	Execution language
9.	Business Process Definition Meta-model (BPDM)	OMG	Execution language ad/or notation language
10.	Business Process Runtime Interface (BPRI)	OMG	Administration and monitoring Human interaction System interaction

11.	Workflow API (WAPI)	WfMC	Administration and monitoring Human interaction System interaction
12.	Business Process Query Language (BPQL)	BPMI	Administration and Monitoring
13.	Web Services Choreography Interface (WSCI)	W3C	Choreography
14.	Web Services Choreography Description Language (WS-CDL)	W3C	Choreography
15.	Web Services Conversation Language (WSCL)	W3C	Choreography
16.	Workflow XML (WfXML)	WfMC	Choreography
17.	Business Process Schema Specification (BPSS)	OASIS	Choreography (and Collaboration)

The standards which are used for business process management to this day are described in Table 15. Most of these standards today are unified under one standard called BPEL4WS, which covers every needed step of business process modelling: architectural model, notation, execution, administration and interactions, also choreography.

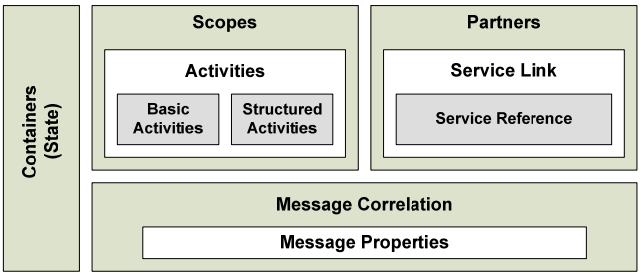
The exception management in B2B model in the frame of this research will be constructed on the basis of the WfMC architectural model, the processes will be described using BPMN language and executed using BPEL for orchestration enablement. The process choreography will be achieved using WXML, which usually is managed in its own manner by every commercial business process management engine.

The next picture describes the high level architecture of standards usage. No matter which commercial product is selected, the core function of process management will depend on the engine, which is responsible for timely and precise process notation and execution procedure interpretation.



Picture 48. Main pieces of BPM architecture and their relationships [20].

As the numbers of standards, which are needed to cover each part of BPM architecture, are quite wide, the requirement for deriving the new broader standard was raised. BPEL4WS was developed by IBM, BEA and Microsoft in the year of 2002, which main purpose was to deliver result through web services. It was assumed, as the business itself is a sum of its process, the orchestration and process refinement, it was critical to make it possible to achieve this using one standard, which also had to ensure BPM flexibility.



Picture 49. BPEL4WS logical view [5].

BPEL4WS and BPML to this day shares strongest features of PBM and that is why most spread are across commercial companies. They are related to BPMI.org standards, BPML is a superset of BPEL4WS, also they share identical set of idioms and similar syntaxes for convergence, provides rich and mature expression language, are block structured languages and are based on logical process model that can fully express concurrent, repeating and dynamic tasks. Also they are built on foundation of WSCI for expressing public interfaces and choreographies [8, 77].

Some other tools, such as BPEL2PNML, WofBPEL and Postprocessor must be mentioned here for their scientific nature and value, as they are capable of translating BPEL to Petri-Nets. The PNML (Petri-net Markup Language) code can be loaded into WofBPEL tool, which is for formal analysis of BPEL processes. Postprocessor is used for PNML and WofBPEL output interpretation [76].

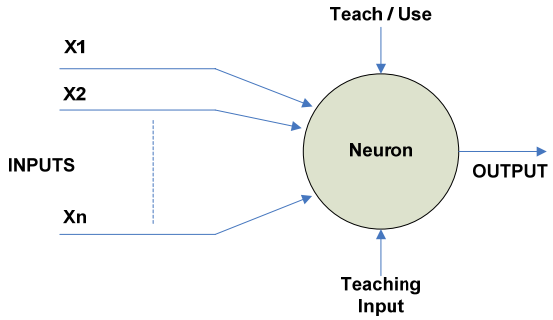
Neural networks usage

The main problem in BPM exception handling is the lack of the ability to predict the possible problem, to decide how to respond to it and to learn to handle the problem in the most suitable manner. In this place the science of neural networks can be applied to gain great value of analysing the information which is constantly collected during business process execution.

Definition 19 *An Artificial Neural Network (ANN) – is an information processing paradigm that is inspired by the way biological nervous systems, such as the human brain, process information [87].*

The key is in using the brain mechanism of reacting to input impulses to solve specific problems, while also learning during this process. The main functioning principle in artificial networks is the ability to extract patterns and detect trends that are too complex to notice by humans or simple applications. The key secret of the success of neural learning is training, which is gained through the information classification, in such a way it is possible to answer the "what if" questions, which are mostly problematic in any exception handling mechanism. Using the ability of ANN to learn and combining it with the computer algorithm approach, it would be possible to create the exception handling mechanism, which would minimise the need of human interaction by copying business world (ANN is widely used in business for forecasting and prediction) and human brain functioning.

An artificial neuron is a device with many inputs and one output. The neuron has two modes of operation: the training and the using mode.



Picture 50. Artificial neuron structure [87].

During the training mode, the neuron is trained to react to given input and fire the output. During the using mode the neuron is giving and output only to recognise during learning input patterns. If the input pattern is not recognised, the firing rule is used to determine the output result. Here again we can remember "if then" rules or even taxonomies, which also classifies information according to its inputs and outputs. Adding the ANN firing rules principle, which also is known for high flexibility (the key desired element in exception handling), it is possible to gain the intelligent approach for handling unpredicted exceptions.

The other useful property of ANN is its architecture, which is usually layered as in enterprise or BPM architectures. This gives an opportunity to use the advantage of simple three layers ANN architecture and to integrate it into each 3-tier BPM architecture to supplement the BPM elements for better flexibility and information analysis issues. Simple 3-tier ANN architecture consists of the input layer, which is connected to the hidden layer and that, is connected to the output layer. Analogically, in 3-tier architecture, the input layer is a presentation layer, the

hidden layer is the application layer and the output layer can be presentation or infrastructure layer, depending on the current business process execution process.

There are quite many learning process paradigms of ANN, but the most corresponding to the needed features to realise intelligent learning exception handling mechanism is supervised learning paradigm. It would allow the exception handling mechanism to learn from the collected data and determine the output with a help of external teacher, which also can be human in case to receive the desired and mostly needed for the business process to result response. The expected result of such exception handling mechanism would be the usage of the ability of the neurons, which are taught to respond to the patterns not seen during training – this is equal to the BPM ad-hoc requirement and unexpected and unpredicted exception handling main sense.

Integration of the "Black box" principle

The discussed neural network approach as itself is referred to the "black-box" algorithm, as it does not reveal the functioning process to the system or user and does not provide the explanation of its decisions. Therefore it is important to discuss the "black-box" principle as it is a part of exception management process.

Definition 20 *"Black-box" - is a technical jargon for a device or system or objects when it is viewed primarily in terms of its input and output characteristics. Almost anything might occasionally be referred to as a black box: a transistor, an algorithm, the Internet [90].*

Picture 51. "Black-Box" system conception.

The author of [33] stresses, that a CBR method (discussed in previous sections) is not that valuable and returns not precise decisions, as the nearest retrieved case in CBR system, may not be the best case for the current issue. Author also notes, that combining the CBR and ANN methods it is possible to construct the decision making system which would return most valuable results. Using such approach, the black box is treated as an oracle for predictions and an artificial data set is constructed around the point of inquiry and then a model is built on this data.

Such method proves in validating the ANN result while using the knowledge reuse and their weight method elements which leads to the most desired and suitable solution for handled situation.

The "black-box" method also is valuable in other places, than selecting the most appropriate handler for the case, in exception handling. This method should be used as a basis for

constructing exception handling end-user interface, as it would not reveal the essence of the exception and the handling details to the user. This would make the user interface more friendly in its simplicity and would not burden the user with a handling data, which is necessary only for the system designer or administrator. The user interface constructed in such a way, would have more use to the end-user providing detailed data on current issue and results of achieved current issue, which would inform the current status of executed process and provide information on the next steps, in which a user is involved or should be provided on necessary information.

Existing Exception Management Technologies

Today's workflow management systems are built using two main technologies, workflow schemas or Petri nets. Some of them support ad-hoc change (like FileNet and Tibco/InConcert) some of them use versioning mechanism and in part ad-hoc changes (like IBM Lotus Workflow or EMC Documentum Workflow), but all of them are not capable of handling real world changes and exceptions.

Commercial tools' purpose is to provide the workflow architect with such an instruments, which would let to construct such a workflow, which could be used as a template to initiate many workflow instances and manage them with the same creation instruments. Still, none of the vendors takes into the account the need of flexible ad-hoc change or exception change, but we have to admit that dynamic changes can be handled quite in an intelligent way. The existing workflow change and exception management tools are often related to verification and validation stages and are more of scientific nature than of real world situation management. One of such tools, Woflan, can import workflow process definitions from several workflow management systems and can analyze realistic and complex workflows [42]. For a successful change and exception handling, the analysis capabilities are critical in workflow support systems, where should be strong verification, validation and performance analysis tools. Woflan is capable to make transformation to Petri nets from any workflow management system definition language and can exactly find and exception spot and fully execute detection and diagnosis phases. The weakness of Woflan is that it lacks the handling phase execution instruments and requires usage knowledge. Also, as much as Woflan is capable of interpreting various transformations of various languages, it is not built on inter-operability principles, and in such way looses flexibility feature.

Good workflow support system also should have knowledge of business processes, their instances, and their history data and simulation capabilities. Some of the workflow management systems support simulation or provide gateway to existing simulation tools, like ExSpect, which can simulate COSA workflows, the business processing reengineering tools ARIS and

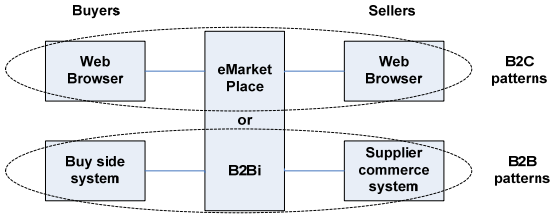
BusinessSpecs can import and export Staffware procedures, and Meta Software's workflow can interface with Visual WorkFlo and FloWare [42]. It is obvious that many vendors do not leave exception handling problems behind their developed tools, but try to integrate various functions and capabilities to help to solve unknown and undesired issues. The problem persists, that these functions and capabilities cover only some exception handling elements and in that way can not fulfil the desired scope management.

The other kind of exception management systems are related to particular industries and are often created in ontology-based manner [43]. The ontology based exception management system is exhaustive, but it is very tightly related to the particular industry context dynamic exceptions. It usually consists of two steps, like ontology-based identification of migration rules (like profile change and workflow condition change, rule change or automatic exception handling, which is achieved by comparing actual and expected input and output data) and applying migration rules (which can consist of compensate, redo, order, insert or delete operations).

Some of the industrial workflow management system use versioning mechanism as a base for change and exception handling. If a change or exception is raised, a new version of workflow schema is constructed and installed. The already running workflow instances are left to complete with the old schema and the new ones are started using new workflow schema. This method is the easiest but not the best, as many process instances can be long lasting (like loan process) and a lot changes should be made according to the external changes during that period. Here the need of flexibility feature integration into the solution still persists.

4.3 Business Process Business-to-Business Exception Handling Models Using Workflow Method

The main types of B2B applications are eMarketPlaces, enhanced e-shops, and B2Bi, the e-service providers. Both of them are usually modelled in the same manner extinguishing only business process description and procedures. The tools for completing the process are usually quite the same.



Picture 52. B2B application high-level conception [28].

The main identified issues of B2B application is inter-operability and flexibility issues, which if solved, would help to solve flexible exception handling problem.

Definition 21 *Interoperability - is the ability of products, systems, or business processes to work together to accomplish a common task [90].*

The goals of achieving inter-operability can be addressed like [65]:

- Support chained, nested and parallel-synchronised models of inter-operability,
- Provide for both synchronous and asynchronous interactions,
- Support for individual and batch operations,
- Remain implementation independent,
- Define a light, easy-to implement protocol.

Flexibility has always been an object of discussion. Many researches are curious of the true nature and influence to the process of flexibility. The different means of flexibility has been noted.

Definition 22 *Flexibility - refers to the ability to easily bend an object or the ability to adapt to different circumstances [90].*

The different ways and places of achieving flexibility have been found. In this paper we think of flexibility as of [22]:

- 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.

Such definition of flexibility should be the essence of any application architect as it points to the ways of coming up to the truly supportable and exception non-receptive business environment.

The following solution on handling exceptions in B2Bi (or eMarketPlaces) will be based using these two arguments as goals in mind: flexibility and inter-operability. As these issues are managed deep in the back-end of BPM, where also real exception management should be addressed, the possibility of combining these features for better BPM performance is quite realistic. The further example of exception handling will be also based on CBR method and using “black-box” principle for end-user interface realisation.

The modelling of B2Bi exception management system

The main tasks and steps for achieving in creating the flexible and inter-operability supporting exception handling mechanism during this research are these:

- To define main B2B exception taxonomy,
- To create the taxonomy of exception diagnosis and their handlers,
- To define specifications for exceptions, diagnosis and handlers,
- To define the exception handling model architecture,
- To model the exception handling procedure using particular example.

Creating B2Bi exception taxonomy

The important question is why we should take exceptions into account so seriously and why we should try to avoid them. There are many business cases which illustrate how unnoticed exception in workflow (which were not modelled and noticed in time) made the biggest and oldest companies bankrupt by injuring their, stable at a glance, enterprise architecture [41]. The source of exception initiation is always human-user or a system, in both cases the seriousness off exception damage must be taken into account. As it was stated above, the most important thing is to avoid the echo of exception in other application layers than it was encountered.

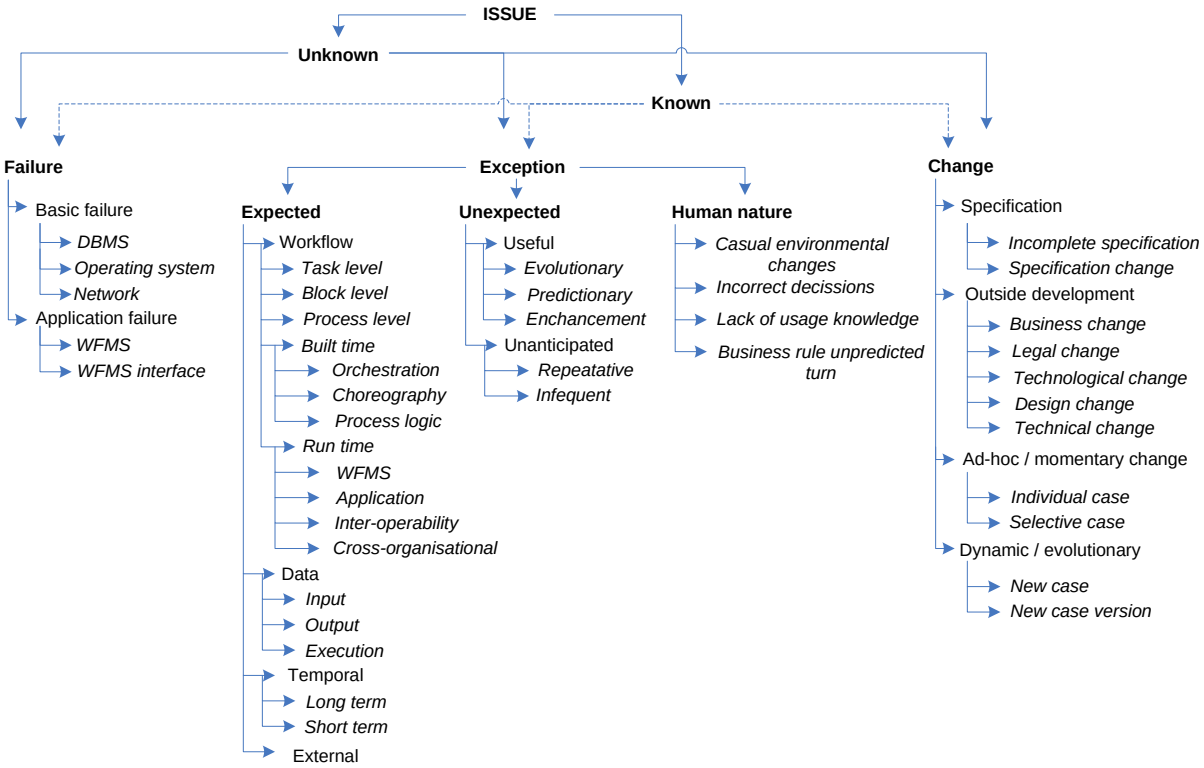
Though many researches does not exclude the human factor in the basic exception taxonomy, quite a number of exceptions arise exactly because of unpredicted human brain result and only later can result in application, business or infrastructure exceptions. From the analysis of previous sections, it is obvious that human factor exceptions are unavoidable due to the surrounding environment changeable nature. It is wise to state that exception is always a part of any application, as it starts changing from the very first day of application implementation. That leads to the assumption that exception is already a part of the application, but not yet activated and thus must be regarded as mere deviation from the current business state and must be dealt with using "if *condition* then *conclusion* " method [3]. Still, such method can be not enough between distributed processes, of which B2B application is constructed, when processes interact with external applications and have side effects [19].

Some authors suggest using an integrated perspective when exception specification is projected on several layers of evaluation, like strategic, tactical and operational level, each of them pointing to particular side of exception properties [28]. Basically this points to the application ability to support ad-hoc processes and ad-hoc processing. As every time the distinction in properties is found, we must consider changing the particular workflow instance or the whole process.

Here we come to the conclusion, that ad-hoc property is the greatest e-business application advantage seeking flexibility in managing human factor exceptions. Still, this property can be enhanced and generate better results if we could manage the knowledge of the whole exception lifecycle, like the cause of appearance, processes, tools or even actors usually involved, screenshot of current situation and data log of processed data at the exception time. Surely this information is nothing without proper analysing tools, so here we once again face the proper workflow management system features, such as workflow metrics and monitoring, workflow statistics, exception handling.

In this decision rises the great need of having the dynamic issue taxonomy, which is a result of this research also (see Picture 53). The issue term in this place is selected because it can represent exception, failure or change at the moment when we can not prescribe the particular unexpected and unknown event to any of these classes. The exception handling system of course should be loaded with the starting exception taxonomy, which can already be generated from the architects experience and most spread best practices. During the usage of the B2B application this taxonomy will hopefully be filled with new branches.

The most important thing in properly using the advantages of exception taxonomy is the specification of the issue, which include issue attributes, needed for weight calculation and supporting attributes needed for human decision achievement (see Table 15).



Picture 53. Issue taxonomy, containing exception, change and failure types.

The proper method for calculating exception weight is CBR similarity calculation formula. This formula is useful, when a feature of self-standing automated exception specification and selection mechanism is implemented in exception handling solution. That takes advantage of using the knowledge of already possessed knowledge about experienced issues and deciding if current issue can be prescribed to the know issue. In such case, the specification of current issue is created and the attributes are weighted. The next step is to select the possible similar issue specifications from knowledge repository, weight them and identify similarity level. The less similarity result is achieved the more similarity is found in two issue specifications.

$$S = \sqrt{\sum_{n} w_i * (a_i - b_i) * (a_i - b_i)}\,, \quad \text{discussed in [73]}$$

The known and unknown branches are on the first level of the issue taxonomy as any of the 2nd level issue can be unknown, till the repository is not search and the similarity is identified. The issue remains unknown if the similarity result not zero, otherwise it is classified as known, in part or in whole. The highest similarity will be found if ***S*** evaluates to zero.

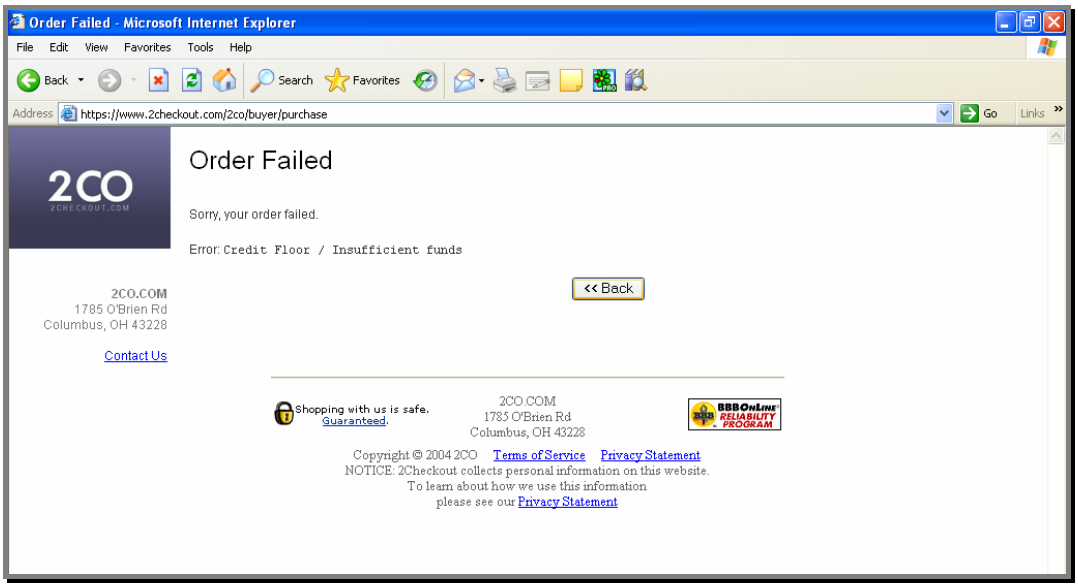
Table 16 Attributes and weights for exception, change and failure.

Exception		**Failure**		**Change**	
Attribute	*Weight*[1]	*Attribute*	*Weight*	*Attribute*	*Weight*
Detectable	5	Failure place	5	Preconditions	9
Exception place	4	Failure type	3	Change place	4
Detection place	3	Damage	10	Change type	3
Known	6	Current state	8	Change conditions	7
Exception type	3	Handling type	3	Change procedure	3
Resolvable	8	Handling results	7	Complications on BP	9
Handling place	9	Action list	2	Change execution type	3
Handler type	3			Change execution place	5
Handling method	10			Execution test result	8
Damage	7			Expected result	10
Diagnosis type	3			Influence on BP level	10

Creating B2Bi exception diagnosis and handlers taxonomy

While analysing the nature of exceptions, the possible their cure and avoiding ways, we still must not forget that exceptions can not be fully avoided and handled. This shows everyday practice while using business applications. To this day, many people are using e-business technologies for paying bills, shopping, getting certified or mostly important managing their own accounts. It is obvious that the processes which involved our own money are mostly important for us and we are always doubling cautious in such situations.

[1] Weights for each attributes are prescribed using the numbers of the scale from 1 (lowest) to 10 (highest).



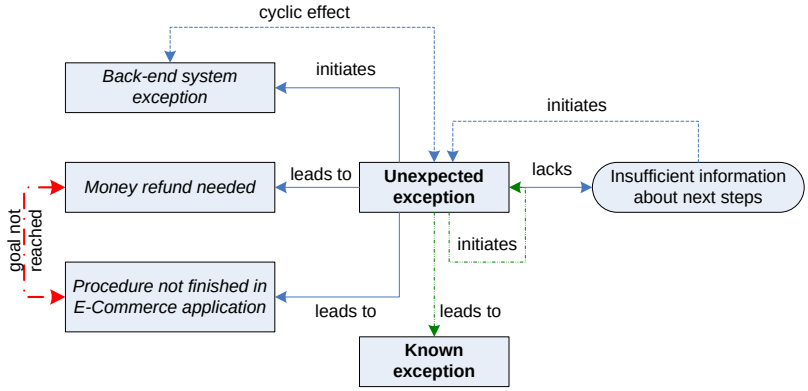
Picture 54. E-Commerce application exception, received after failed money payment.

The Picture 54 shows the most common exception result in e-business application. The exception was achieved after basic human mistake – wrong data entry. It would seem that such a case should be predicted in such an application. Still the user gets the most uninformative error message, which points to only two diagnoses in this case:

- the user account lacks money,
- some serious problem was raised in the application.

Unfortunately, user does not get any information nor from the system directly, nor from the application support team, while the transaction was actually made, but the process is suspended as it does not finished registration and informative procedures.

Picture 55 clearly describes that such unexpected exception can initiate other back-end system exceptions, which are not usually noticed immediately, but maybe after some time. In response, such left and unnoticed back-end system exceptions can again raise the same or other exceptions in user level and initiate cyclic exception occurrence effect. That is why it is always vital to remember to start handling any exception case from going to the back of the system and starting checking from the very roots of whole application logic.



Picture 55. "Butterfly effect" initiated by unexpected exception.

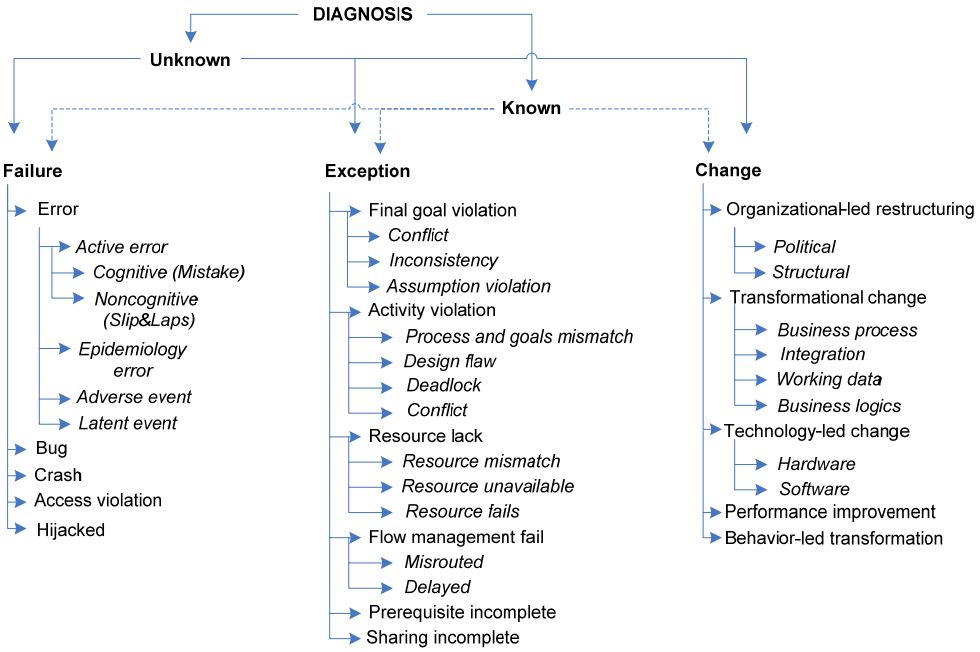
The same raised unpredicted exception can lead to two more uncertain conclusions: the account balance should be checked and the business procedure itself should be finished. If we leave the current procedure as it is, the result would be, that some data can be left open for intruders (if sessions were not finished properly), the user profile can be messed up and contain wrong information which would prevent him/her from making other transactions in the future, and current transaction will be questionable (was it achieved as it should).

Many of these questions could be avoided if the user would be supplied with proper information and help at the moment. While the user didn't get it, he/she initiates other doubtful cases, which can later raise unknown exceptions.

The bright side of this situation is only one, if we manage to detect the roots of current case and find a proper handling way; we gain knew valuable knowledge of this business process and related technologies. This unpredicted business case becomes a known exception.

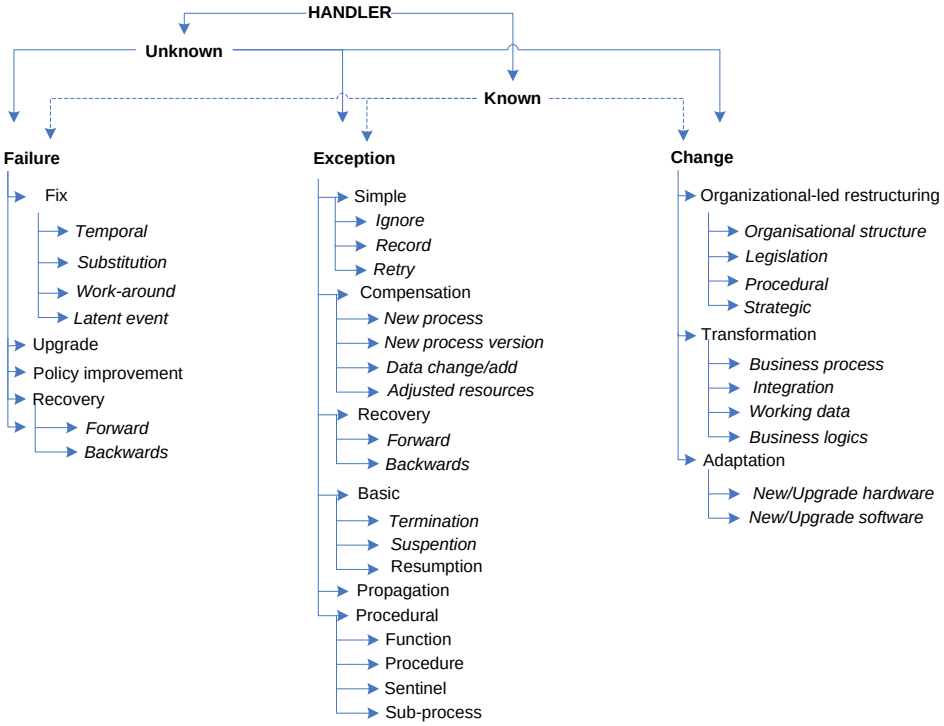
This example shows very clearly why a proper diagnosis stage execution result is such vital in any exception handling situation. Basically, the diagnosis is what a user gets as a result in the computer screen and also diagnosis is an instruction for the application performed next steps. That is why it is also valuable and helpful to create the dynamic diagnosis taxonomy, which would have as much informative specifications as it is possible. It is also very important to notice that these specifications should be filled with more care as some of the text will be published as an answer in the problematic situation.

The diagnosis taxonomy (see Picture 56) must be created to correspond with the context of the B2B application and B2B technologies specifics, but not necessarily should have the same number of branches and number of levels, as an exception specification can point to several diagnoses, or vice versa. The following diagnosis taxonomy was produced during this research.



Picture 56. Diagnosis taxonomy, containing failure, exception and change types.

The decision on which diagnosis specification should be selected is made the same way as in exception case, by analysing current case (in this place exception specification), already possessed diagnosis specification and calculating the similarity level. If the corresponding diagnosis is not found, the new one must be specified and included into the taxonomy.

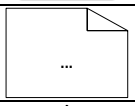
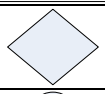
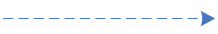


Picture 57. Handlers' taxonomy, including failure, exception and change types.

The last taxonomy needed for the successful and result achieving exception handling mechanism functioning is the exception handlers taxonomy, as one of the results of this research. Selection of handler is a result to diagnosis phase and therefore diagnosis taxonomy branches should always point to at least one branch in handlers' taxonomy (in the ideal model). If the branch in handlers taxonomy corresponding to any diagnosis branch does not exists, this exception remains unknown and its diagnosis remains unknown as the solution does not exists (at least in the frame of exception handling mechanism).

The result of the competent diagnosis is the pointer to the exception handling taxonomy branch in the ideal model and the best similarity result in the real world situation. The usage of similar handler should be approved by human, as in this place the final business process result is what matters. If the similar handler can not achieve expected or satisfactory result, the current handler is modified to meet these needs and saved into repository as new handler. Also the totally new handler can be created, if it turns that the reengineering costs are too high for the case.

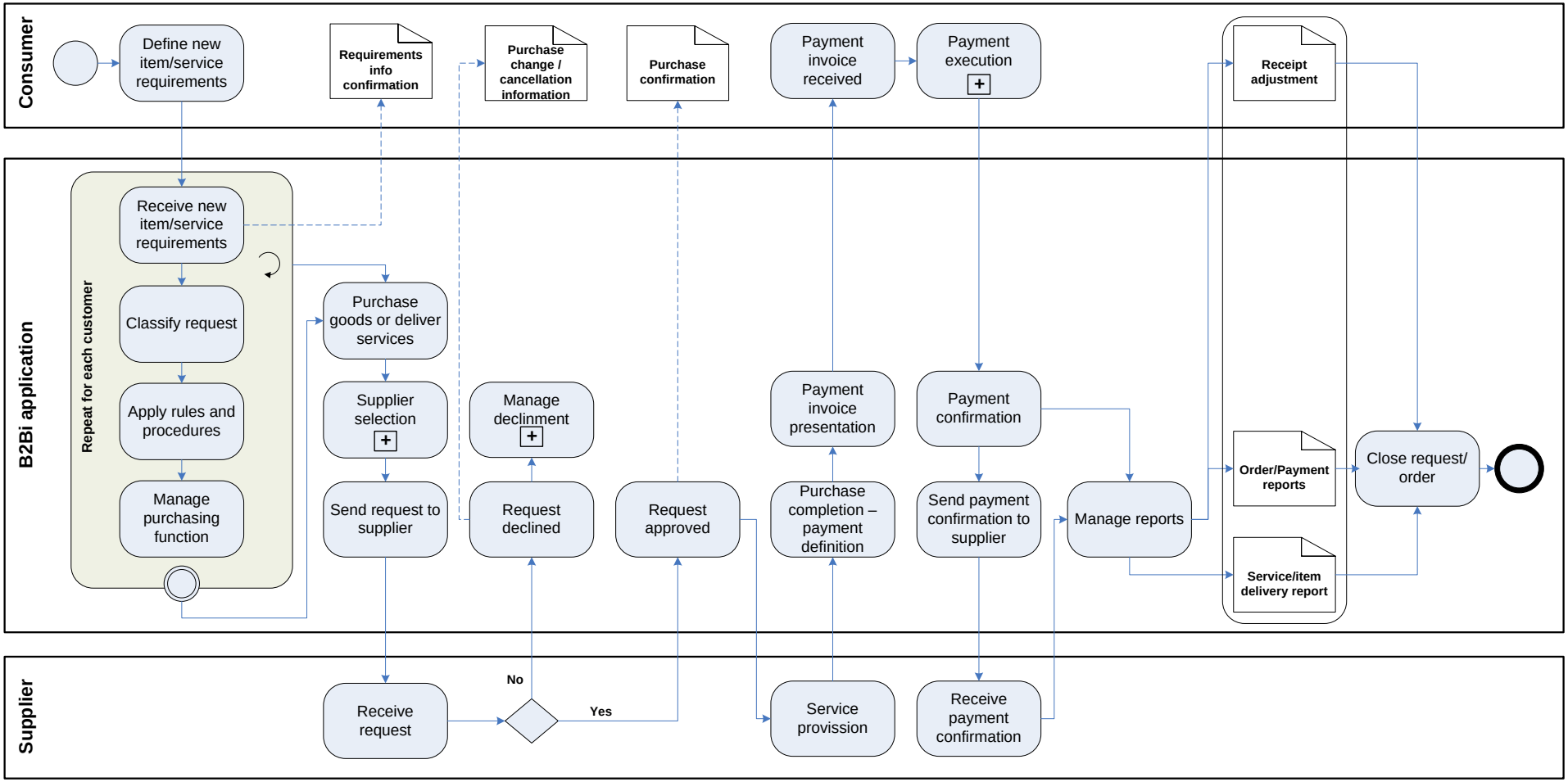
Table 17 Legend of notations used in Picture 60.

No.	Element name	Element depiction	Element description
1.	Activity: Task		Generic term for work that company performs. It can be atomic or non-atomic (compound).
2.	Activity: Sub-Process		Have the same characteristics as activity task, only sub-process can have no end-point or initiate self-standing different process.
3.	Data object		It is a mechanism to show how data is required or produced by activities. They are connected to activities through associations.
4.	Gateway		Used to control the divergence and convergence of sequence flow. It is also used to depict decisions.
5.	Event: Start		Events affect flow and have cause or effect. This one indicates the start of the process.
6.	Event: Intermediate		Events affect flow and have cause or effect. This one indicates the start of the process.
7.	Event: End		Events affect flow and have cause or effect. This one indicates the end of the process.
8.	Loop		Loop indicates the repeat of the group of the tasks.
9.	Sequence flow		Used to show the order that activities will be performed in a process.
10.	Message flow		Used to show the flow of messages between two separate process participants that send and receive them.

For the further B2B exception handling mechanism description of the B2Bi type process for supply chain management will be used (see Picture 58). The process is created and described using BPMN [86]. That makes this process description more convertible into process execution

procedure and lessens the level of misunderstanding between process architect and its user. As all B2Bi processes, supply chain management process covers three parties: customer, supplier and the broker. The communication between them is achieved using inter-operability enabling instruments.

Further the architecture for the B2Bi exception management will be discussed using this process as B2Bi business process and its description for input and possible execution simulation.

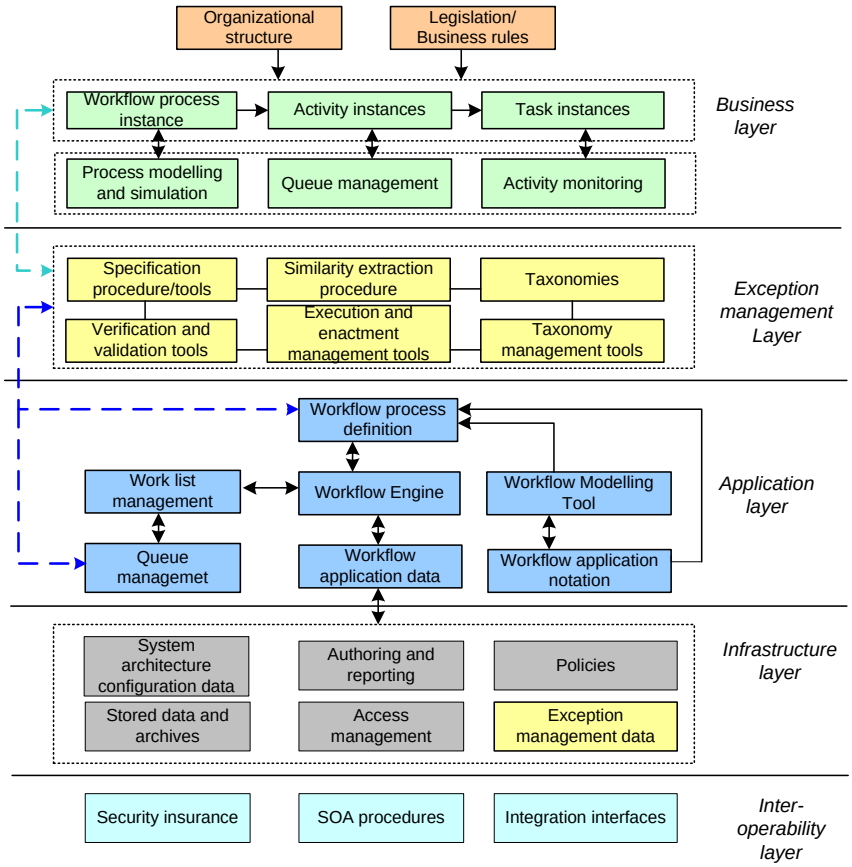


Picture 58. Supply Chain Management process using BPMN.

Defining exception handling model architecture

Seeking to create a workflow based exception handling model, which would be suitable for managing inter-organisational exception, it is necessary to take into account the influence of widely spread standards and methods for creating inter-operability supporting methods. For this purpose the architecture which is represented below is based on SOA architecture and WfMC workflow management system meta-model.

The described e-business architecture (see Picture 59) is a model for achieving flexibility in system and human driven exception handling. The architecture is based on mostly spread layered architecture, which also correspond with SOA requirements. The enhancement of this architecture was achieved during this research through the two additional layers: exception handling and inter-operability layer. The exception handling layer is like glue between human (system user) and business logics and is located between business and application layers. In so doing it is possible to achieve the direct communication and human decision transfer to business logic from application layer, through the exception layer which helps to evaluate the properties of current and possible next exception/deviation. This is achieved by using current situation specification and projection on predefined taxonomies.



Picture 59. Flexible exception detection and handling e-business architecture, incorporating mid-layer between business and application layer.

The inter-operability layer is added as the extension to infrastructure layer in order to separate workflow, business and exception management from their integration with external corresponding applications. As inter-operability main purpose is to securely exchange business data, therefore related application procedures should not be allowed to directly access upper layers of described architecture.

Realisation of Proposed Architecture

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 adaptable to business process management systems and other applications. Multi-layered architecture has all required specifications possible to comply with flexibility and easy support requirements.

Constructing architectures it is vital to remember the great influence of humans to BPM. Users should access the business management system using their distinct user permissions and roles profiles which define what different access and responsibilities the user has to different application modules. These profiles must be managed in business layer 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 inputting their requests. Once the request is filed, it is passed to the e-business process logic, which defines processing sequence for that request. Deviations quite 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 and applications layers.

The exception handling application, which is integrated into BPM architecture as one of its layers, must also have a dedicated schematic of all business logics, represented as possible exception tree [40] 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 this example the taxonomies are used. 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 must look for the appropriate patterns in the behaviour of the basic components, can be used for this purpose [17, 11].

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 exception management environment.

The architecture, created during this research and described in the Picture 59 consists of:

- Business layer – which mainly is responsible for business activities representation to the user and supply of necessary instruments required for completion of these activities. The layer also has input data, which is received with a help of the user or through supplied organizational and supporting business specific legislations (like organisational structure, business rules, standards and etc.). Business layer is responsible for whole workflow instance, its activities and their tasks representation and execution. Therefore it supplies a set of procedures and functions which help to monitor the progress of activities and manage a queue of tasks, by defining priorities and estimating resource overloads to needed to complete tasks. This layer also is the place where specialised interfaces or applications for modelling and simulating the business processes as workflows reside. This eliminates the possibility of misunderstanding between system designers, users and system itself, as these tools are designed to model the workflow in simple graphical representation form and to interpret by system understandable languages in the backend of BPM by it.
- Exception management layer – which is basically exception handling module and interacts with the workflow management module through the similarity extraction and evaluation tools. Additionally this module gets required data through exception specification. Its main function is to decide whether a running workflow schema instance has exceptional, changeable or failure situations. For that purpose, the similarity extraction tool compares workflow schema instance historical data with the appropriate taxonomy data. The taxonomies are changed during this process as in this example exception, diagnosis and handler taxonomies are separated with the purpose to create most flexible and precise descriptions of all of these phases. The separation of taxonomies is valuable as it does not requires to posses all sequential information about some exception, its diagnosis and handler, as in most cases some of these elements can be missing or unknown. To fill these gaps is possible through separately specifying the all three stages of the exception handling procedure and using these specifications to identify the possible similarities with already detected and solved situations. 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. Of course newly created specifications and handlers must be verified and validated before their true appliance.

- Application layer – is basically a workflow management module. The core of this layer is a workflow engine, which is usually used for the running workflow instance processing. The workflow engine acts according to the predefined workflow schemas (which are stored partly in infrastructure and partly in an application layers). 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 layer. The additional data needed for workflow schema instance processing is accessed from the workflow application data (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.
- Infrastructure layer – consist of knowledge repositories, knowledge management, security, authoring and policy management tools, also include specific infrastructure configuration instruments. This layer 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 these types of repositories:
    - Exception, diagnosis and handler specifications,
    - Workflow application data repository;
    - Business process management system repository;
    - Long term data repository.

  The main requirements for these repositories are scalability, flexibility and extensibility.
- Inter-operability layer – enhances the abilities of infrastructure layer with additional SOA corresponding procedures, which main purpose is to exchange information with other, and not necessarily identical or homogeny BPM. For required BPM security insurance, the

layer can be totally separated from the upper layers and exchange information with them only with a help of infrastructure layer which validates and verifies exchanged information against defined policies. Through this layer is totally separated from the rest, still application layer can have add-ons to business process modelling interface, which would supply the architect with the instruments for modelling inter-operable business process extensions.

If the BPM architecture, enhanced with exception handling model, proposed in this paper would be implemented in a particular industry business process context, there would be a good reason to have exception ontology rather than taxonomy (referring to the [25], 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 compare actual data with an ontology tree. If data does not match, a sentinel could pass this data to the exception registration agent.

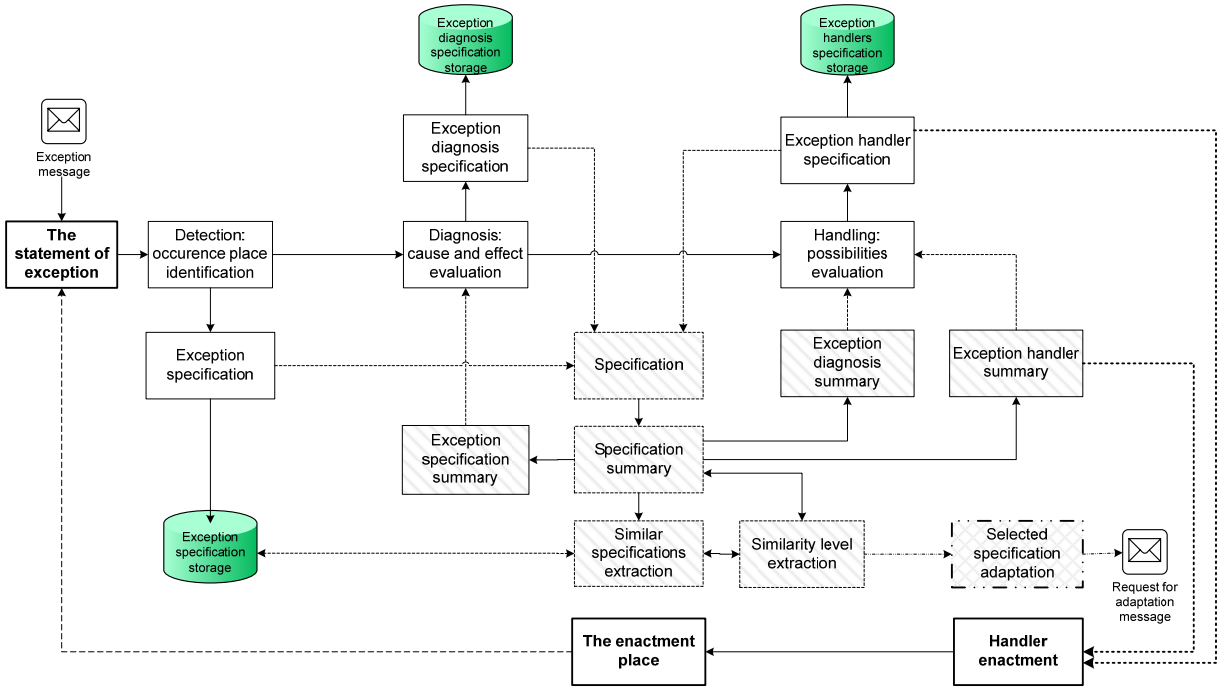
Any business process flow processing can involve many people with various roles, who have different tasks and responsibilities. The exception detection sentinels must take account of that - if the ontology itself provides or links to information on roles and task lists.

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

Using Workflows for Business Process Exception Handling

Detection, diagnosis and handling must be related to business process scope, the expected process result and interacting resources. Business processes are usually modelled using workflow methods; therefore, it is logical to handle exceptional situations in a similar manner. This research paper models exception management using workflows. 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 one of the three phases (detection, diagnosis and handling), to confirm the automatically made decisions. Confirmation is usually needed due to the unpredictability of real world situations. The business process human executor is almost eliminated from the exceptional situations handling, excluding rare situations, when the exceptional situation must be evaluated immediately from the user's point of view. The

automatically formed exception, diagnosis or handler specification is sent to the exception handling engine for processing and/or to the system administrator for the approval.



Picture 60. Exception handling procedure meta-model (algorithm) using workflow method.

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.

Table 18 The explanations of notations used to describe exception handling algorithm.

No.	Notation	Description
1.	...	External message, managed by workflow management system messaging module and responsible for interaction with business process and exception management layers.
2.	...	Workflow management start and ending describing stage.
3.	...	A stage of workflow management algorithm representing its core activities, also can be referred as decision point.
4.	...	The stages of workflow management algorithm which activities should be included into separate sub-process, as they are repeatable for detection, diagnosis and handling stages.
5.	...	External stage, which mostly require human interaction and can return result to main exception handling stages or supporting/sub-process stages.
6.	...	Knowledge and specification and their summary repository.

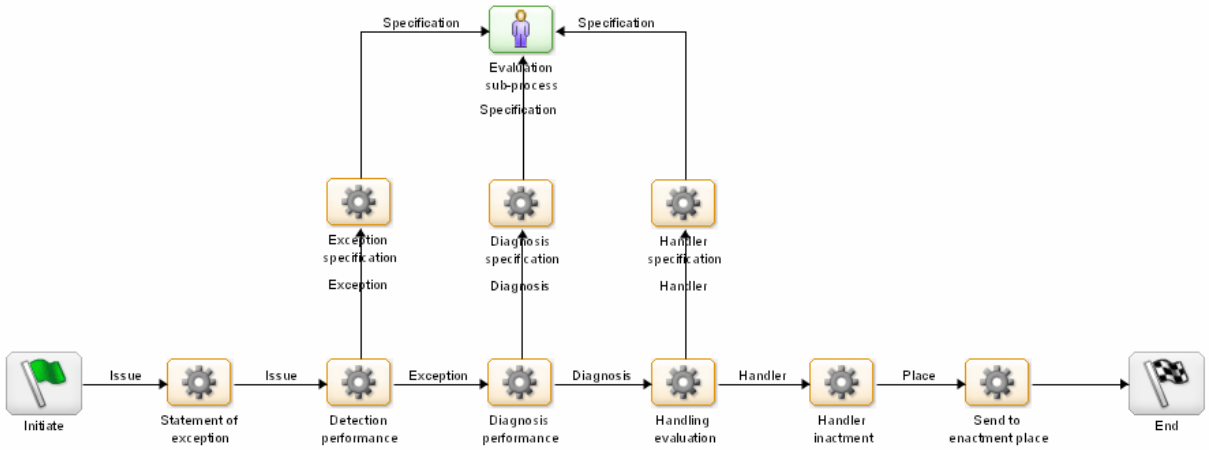
The most important step of the exception handling process is exception detection. This phase can be achieved in many ways, such as making the ideal graph of workflow and comparing every

run-time step to it and if we find a distinction, we register an exception. The other way is to build sentinels, specific services, which would be active all the time and gather information/data on workflow instance run-time and compare it to predefined ontology or taxonomy. [60] always stress that the one who wants to deal with exceptions must go behind the system. The detection stage should be related to going to the back of workflow management system and gathering information there, which is usually a cause of unpredicted and unknown exception.

After identifying the right spot of exception occurrence and gathering all possible data, which will help to deal with raised situation we can pass to the second stage: the exception diagnosis stage. Where the detection phase tends to be similar for various contexts of e-business architecture, the diagnosis phase is highly dependent on context and (therefore) differs from case to case.

This is particularly true if we care to catch not only infrastructure and application exceptions but also business logics exceptions. The best way here is taxonomies, which defines the context in detail and predefined hierarchy according to specifics of the context.

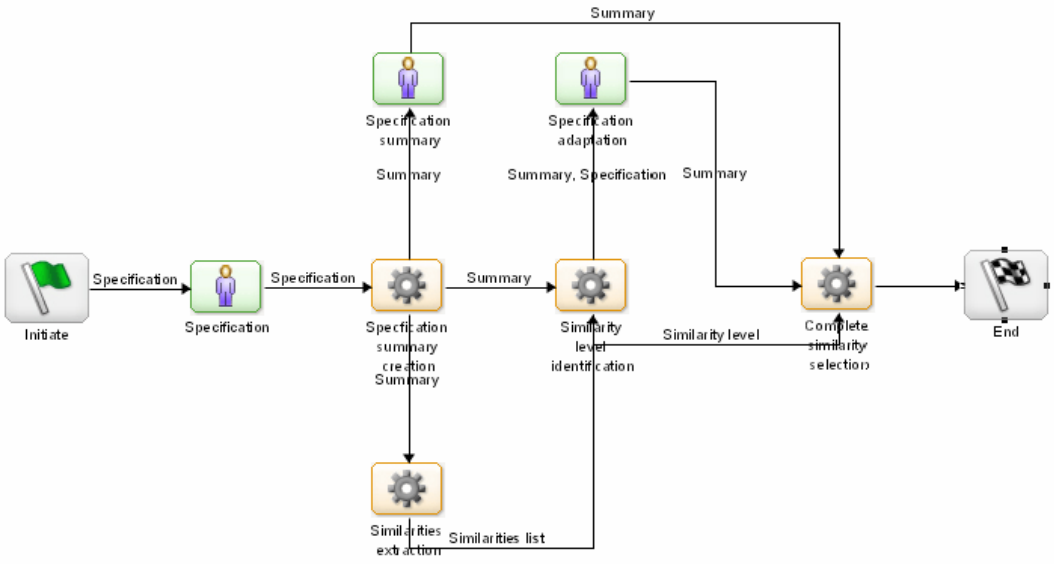
The result and actions taken to deal with the exception depends on how precise the diagnosis was made. So if detection of exception is a projection on ideal workflow model graph, then diagnosis is a method to identify the distinction of this projection and the reasons/sources why they occurred. After approval of the exception diagnosis, the last step of the cycle is to select the appropriate handler. This phase is the most simple as system or human has guidelines, specification and just need to execute them correctly and finish the case. The fulfilment of business goal is not at stake during this stage though it may seem so. The decision of the impact on business goal must also be made in diagnosis phase, leaving the handling phase only to confirm the correctness of the diagnosis.



Picture 61. The exception handling workflow model (designed during this research using EMC Documentum Business Process Manager).

The handling phase can, none the less, be tricky as other unpredicted exceptions can be raised if diagnosis or detection phases failed. However, knowing the reasons and the cause of the exception, one only needs to select the right procedure, method, function or modification and enact it on the predefined spot, which is also identified in diagnosis phase. Note that one should not forget that the spot of exception occurrence will not always be the spot of fixing if we care about the fulfilment of business goal. The whole exception handling cycle is illustrated in Picture 60.

Picture 60 clearly shows that for any unpredicted case we should find more or less predictable handling case. The handling of exception is almost always achieved through the additional procedures or functions or other program elements which are newly created for this particular case or reused older handling procedure elements. Once again we should notice that if we bring into the e-business application new elements, which were not natively included, we bring a new area for new exception emergence. Therefore, we should always avoid new elements in any part of e-business application, as one change in business logic can not always be prepared to meet this change in application or infrastructure level.



Picture 62. Similarity level extraction workflow (created during this research), modelled as sub-process for the main exception handling workflow process.

The Picture 61 is a projection of Picture 60 using EMC Documentum workflow management system design interface. The selected software is fully complies with SOA and is built according to WfMC meta-model. Therefore the modelled workflow can fully realise all intentions raised in exception management algorithm above. The modelled workflow has one start and one end task, which eliminates the possibilities of not fulfilling all predefined exception handling workflow

instance by taking a wrong branch of workflow schema. Workflow also has one task (the one with a human picture) which is repeatable and fully described in Picture 62. This task is responsible for current issue specification verification and similarity level calculation for all similar specification, found in the specification repositories for current case.

As such sequence of tasks, like creation of specification summary for search in repository, the process of found similar specifications aggregation and finally similarity level evaluation, are repeatable for detection, diagnosis and handler identification phases, these tasks are totally independent of any specific data related to these three stages. The specific data is loaded into this sub-workflow only as input data and fully returned to main process as output data.

The advantage of exception handling algorithm realisation using workflows is that even if all tasks are performed by automated task with a help of workflow engine, even the smallest system functioning or task performance deviations or issues are encountered by the required knowledge possessing human, who is able to make a human nature decision for the raised situation.

4.4 Conclusions

1. Business process procedures are complex and unpredictable by their nature, involving the interaction of humans and multiple system resources. Business process outcomes are functions of time needed to reach the desired business goal and the precision of process specifications. There fore human interaction can not be fully eliminated from exception handling process.
2. This research has resulted in exception, its diagnosis and handlers' classification, also the workflow management system analysis and comparison together with a feasibility study was performed.
3. The exception handing mechanism created during this research consists of exception handling workflow schemas, and its execution algorithm, also prerequisites for exception, diagnosis and handler specifications. The created solution fully realises data interchange between traditional workflow management system and exception management system, also integrated workflow management and simulation tools. This fully complies with raised inter-operability requirements. The flexibility of the solution is gained through the enhanced SOA architecture, which can be fully decomposed if need be.
4. The value of this approach is in its very nature – that workflow exceptions are handled using workflow methods, leading to a flexible and integrated handling of these situations. Testing of this method on real-world situations has been done. These tests confirmed that the source of the exception can be detected at the place of its occurrence, and the

problems which arise at the exception handling procedure or tool activation place can be monitored and eliminated.

Conclusions

1. The performed analysis of business process modelling leads to the assumption, that currently existing methods (like Petri-nets, flowcharts and activity diagrams) will be improved with an aim to create more efficient workflow management systems and notations, reflecting real world situations. Existing BPM technologies are already adapting to evolving process notations and BPM demands, and enhance them with exception management instruments. To this day it possible to perform only come sort of compensation, which truly is applied in rare cases. The exceptions itself can be described and stored in quite informative way and leaves space for their definitions' additions. Unfortunately, current BPM technologies and exception descriptions lack communication interface with each other.
2. The exception handling models, which were created during this research, for B2C and B2B environments, uses tools, which are supplemented with knowledge repository, exception handling engine and inter-operability interface, which is dedicated for exception specification, diagnosis and handler selection activity management. It made possible to realise exception handling flexibility requirement, to shorten the exception detection and handling duration, to minimise the amount of non-detectable exceptions and in such way to give the end user a proper exception description and further steps list. The system inter-operability requirement was realised, which was described using currently existing architecture creation and process creation/management standards.
3. The method for B2C exception handling was created, which connects knowledge management and intelligent problem handling methods. It was achieved by identifying raised goals to exception handling and completion of the projection of these goals on selected technologies and methods. The results of achievement were successfully implemented in practice, in the production and telecommunication companies for workflow based system support. The created model made possible to noticeably minimise the occurring exception number. The main advantage of this model is the possibility to use and manage it as totally self-standing module, non-dependent from workflow engine and business management system. This makes possible to re-use this module for distinct business management system support without any additional changes. The performed analysis results proves, that exception detection becomes possible in 60-95% of exception cases, their management possibilities, using knowledge management tools, grows to 95%, and the impact to the working business process model can be minimised from 5% to 50% depending on exception type. The results were implemented using IBM Lotus Domino environment.

4. The method for B2B exception handling using workflow method was created. It was achieved combining the strongest sides of WfMC reference model and SOA architecture. The full classification of e-business exceptions was created and transformed into exception taxonomy. For better flexibility achievement in performing detection, diagnosis and handling phases, the taxonomies on diagnosis and handlers were also created. This helped to realise the links between exception, its diagnosis and its handler's specification. With a help of these links, the exception handling process noticeably shortens, the number of possible to initiate mistakes or new exception minimises, while performing exception handling lifecycle. The new SOA levels improve the possibility to detect exception between technologically different business management systems, allocated in distance. The recommended method fully performs exception handling using workflow, which also has supportive workflow for repetitive tasks. Created method results are already being used in creation of e-business and e-government realisation strategies, addressing selective issues. The created method was tested in test environment using EMC Dcumentum Process Suite.

References

1. Accela. Government-to-Business (G2B) Web Services for Online Permitting: Vertical Applications that Drive Long-Term GDP Growth. Whitepaper, 2002.
2. Adams J., Koushik S., Vasudeva G., Galambos G. Patterns for e-business: A Strategy for Reuse. IBM Press, 2002, ISBN: 1-931182-02-7.
3. Adams, M., ter Hofstede, H. M., Edmond, D., van der Aalst, W.M.P. Facilitating Flexibility and Dynamic Exception Handling in Workflows through Worklets. In The 17th Conference on Advanced Information Systems Engineering Forum, CAiSE05 Forum, June 2005, Porto, Portugal (2005).
4. Alonso, G., Gunthor, R., Agrawal, D., El Addadi, A., Mohan, C. Exotica/FMDC: Handling Disconnected Clients in a Workflow Management System. Proceedings of the 3rd Conference on Cooperative Information Systems, Vienna, May, 1995.
5. Arjuna. Introducing BPEL4WS. An Arjuna technologies report, 2003.
6. Basten, T. In Terms of Nets: System Design with Petri Nets and Process Algebra. Cip-Data Library Technische Universiteit Eindhoven, ISBN 90-386-0631-1, 1998.
7. Bea Systems. Service-Oriented Architecture. Solution Accelerator Guide: Platform Solution Accelerators. Whitepaper, version 1.0, revision 2.12, 2004.
8. BPMI.org. BPML and BPEL4WS: A Convergence Path Toward a Standard BPM Stack. Position Paper, April 15, 2002.
9. Capgemini. E-Europe. Web Based Survey on Electronic Public Service: Report of the Fifth Measurment. 2005.
10. Casati, F. Models, Semantics and Formal Methods for the Design of Workflows and Their Exceptions. PhD Thesis. Politecnico de Milano, Department of Informatics and Electronics, 1998.
11. 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.
12. Dellarocas, C., Klein, M.: A Knowledge-based Approach for Handling Exceptions In Business Process. Information Technology and Management, Vol. 1, 3, January 2000, pages 155-169.
13. Dickson, K.W. CHIU, Karlapalem, K., Li, Q. Poster on 'Exception Handling in ADOME Workflow Management System'. Proceedings of the 2nd IFCIS International Conference on Cooperative Information Systems (CoopIS '97), ISSN: 0-8186-7946-8/97, 1997, IEEE.

14. Domino Workflow: Automating Real - World Business Processes. White Paper. 1999.
15. 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. 2002.
16. Ganci, J., Banik, S., Boaglio, F., Cowlagi, A., Holecy, M. E-commerce patterns for building B2C Web sites: Using IBM WebSphere Commerce Suite V 5.1. IBM redbooks, Internation Technical Support Organisation, 2001, SG24-6180-00.
17. 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), pp.119-153.
18. Gonzalez, A. E., Smari, W. W., Debnath, N. Towards Flexibility of Workflow Management Systems Based on Task Requirement Classification. Publication of The Society for Modeling and Simulation International.
19. Hagen, C., Alonso, G. Exception Handling in Workflow Management Systems. IEEE Transactions on Software Engineering, Vol. 26, No. 10, October 2000, 0098-5589/00 (2000).
20. Havey, M. Essential Business Process Modelling: Theory in Practice. ISBN: 0-596-00843-0, August 2005.
21. Holligsworth, D. WFMC: The Workflow Reference Model. TC00-1003, Issue 1.1, 1995.
22. Hu, J., Grefen, P. Conceptual framework and architecture for service mediating workflow management. Information and Software Technology, Number 45, 2003: pp 929-939.
23. IBM Corp.: DB Magazine: Strategies & Solutions for Database Professional. Volume 8, Number 2, Quarter 2, 2003, pages: 16-19.
24. Kappel, G., Lang, P., Rausch-Schott, S., Retschitzegger, W. Workflow Management Based on Objects, Rules and Roles, Bulletin of the Technical Committee on Data Engineering, Volume 18, Number 1, March 1995: pp 11-19.
25. 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.
26. Kosiur, D. Электронная коммерция: стратегические технологии. Microsoft Press, 1999, ISBN 5-7502-0104.
27. Kuhn, H., Junginger, S., Bayer, F. How Business Models Influence the Development of E-Business Applications. *Proceedings of eBusiness and eWork*, Madrid, Spain.
28. Maurao, R. H., Antuanes, P. Supporting Direct User Interventions in Exception Handling in Workflow Management Systems. Sistemas de Informação, vol. 17, pp. 39-51. ISSN: 0872-7031 (2005).

29. Moore, B., Fielding, M., Wilson, C., Avanesian, A., Viljoen, P. e-MarketPlace Pattern. Using WebShpere Commerce Suite, MarketPlace Edition: Patterns for e-Business Series. IBM, SG24-6158-00, November 2000.
30. Naick, I., Berkhoff, M., Verdugo Bosnich, D. IBM Redbooks: Business-to-Business Integration Using MQSeries and MQSI: Paterns for E-business Series. ITSO, IBM Corp. 2002.
31. Natis, Y. V. Service Oriented Architecture Scenario. Gartner research, ID number: AV-19-7651, April 2003.
32. Nickull, D. Service Oriented Architecture. Adobe Whitepaper, 2005.
33. Nugent, C., Cunningham, P. A Case-Based Explanation System for "Black-Box" Systems. The proceedings of Seventh European Conference on Case-Based Reasoning (ECCBR-04), 2004.
34. Prior, C. Workflow and Process Management. WFMC, Workflow Handbook 2003, ISBN: 0-9703509-4-5.
35. Pyke, J. BPM in Context: Future Now and in the Future. WFMC, Workflow Handbook 2006, ISBN: 0-9777527-0-4.
36. Ramboll Management. Top of the Web: User Satisfaction and Usage Survey pf eGovernment Services. eEurope 2005. 2004.
37. Rogers, S., Hedrick, S. D. Oracle Builds Comprehensive SOA Platform. White paper, IDC, January 2005.
38. Ruthledge, K. The Business Case for e-business. IBM Press, 2000, ISBN: 1-931182-00-0.
39. Saastamoinen, H. Exception-Based Approach for Information Systems Evaluation: The method and its Benefits to Information Systems Management. The Electronic Journal of Information Systems Evaluation, Vol. 8, Issue 1, 2005, ISSN: 1566-6379, pp. 51-60.
40. Sadig, W. S., Orlowska, M. E. On Capturing Exceptions in Workflow Process Models. Proceedings of 14th International Conference on Business Information Systems, Poznan, Poland. Springer-Verlag, April 12-13, 2000.
41. 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.
42. Sheth, A. P., Wil van der Aalst, Arpinar I. B.: Process Driving the Networked Economy. IEEE Concurrency, 1092-3063, 1999.

43. Soon Ae Chun, Vijayalakshimi Atluri: Ontology-based Workflow Change Management for Flexible eGovernment Service Delivery. MSIS Department and CIMIC, Rutgers University, Newark, NJ 07102.
44. Tarumi, H., Matsuyama, T., Kambayashi, Y.: Group Activity Database for Groupware Evolution. Department of Information Science, Kyoto university.
45. Ultimus Whitepaper. Workflow for the Information Age: 10 Myths About Workflow Automation. 2001.
46. Veryard, R. Business-Driven SOA 2: How Business Governs the SOA process. CBDI Journal, June 2004.
47. Vojevodina, D. Business process exception handling using workflow methods. Lithuanian Mathematical Journal, Special Issue 43. Mathematics and Informatics Institute, Lithuania (2003), pp.210-315.
48. Vojevodina, D. Exception Handling Automation in E-business Workflow Processes. Proceedings of CAiSE'05 Doctoral Consortium, pp. 42-54, 2005.
49. Vojevodina, D., Kulvietis G. B2B and B2C Process Nature Influence in Their Exception Handling, Proceedings of the WSEAS International Conference: 5th international Conference on Dynamical Systems and Control (CONRTOL'05), ISBN: 960-8457-37-8, pp.97-102.
50. Vojevodina, D., Kulvietis G. Change and Exception in Business Processes: Handling Differences, WSEAS Transactions on Systems, Issue 10, Volume 4, ISSN-1109-2777, pp. 1658-1668, 2005.
51. Vojevodina, D., Kulvietis G. Computer Decomposed workflow method for run-time office activity procedure exception handling, Proceedings of Conference on Information Technologies'2004, Technology, 2004, UDK 004(474.5)(06) Li 301, ISBN, p. 400-405.
52. Vojevodina, D., Kulvietis G. E-business workflow: requirements and realization. Scientific Proceedings of Riga Technical University in series 7, ISSN 1407 – 8880, vol. 4, 2003, p. 25-28.
53. Vojevodina, D., Kulvietis G. From Critical Exceptions to Reliable Process Cases in E-business Models, WSEAS Transactions on Systems, Issue 1, Volume 5, ISSN: 1109-2777, pp. 248-256, 2005.
54. Vojevodina, D., Kulvietis G. Office Activity Procedure Exception Handling Realization Difficulties, Proceedings of CAiSE'04 Workshops, Vol. 2 pp.194-201, 2004.
55. Vojevodina, D., Kulvietis G., Bindokas P. The Method for E-Business Exception Handling. IEEE Computer Press, ISDA'05, pp. 203-208, ISBN 0-7695-2286-6, Wroclaw, Poland, 2005.

56. W.M.P. van der Aalst. Making Work Flow: On the Application of Petri nets to Business Process Management. Application and Theory of Petri Nets 2002, vol. 2360 of Lecture Notes in Computer Science, pp. 1-22. Springer-Verlag, Berlin, 2002.
57. W.M.P. van der Aalst. Verification of Workflow Nets. Application and Theory of Petri Nets 1997, vol. 1248 of Lecture Notes in Computer Science, pp. 407-426. Springer-Verlag, Berlin, 1997.
58. W.M.P. van der Aalst. Designing workflows based on product structures. Proceedings of the ninth IASTED International Conference on Parallel and Distributed Computing Systems, pp. 337-342. IASTED/Acta Press, Anaheim, 1997.
59. W.M.P. van der Aalst. Exterminating the Dynamic Change Bug: A Concrete Approach to Support Workflow Change. Beta working paper series, WP 51, Eindhoven University of Technology, Eindhoven, 2000.
60. W.M.P. van der Aalst. Loosely Coupled Interorganisational Workflows: modeling and analyzing workflows crossing organizational boundaries. Information and Management, 37(2):67-75, March 2000.
61. W.M.P. van der Aalst., Jablonski S. Dealing with workflow change: identification of issues and solutions. International Journal of Computer Systems & Engineering (2000) 5, pp.267-276.
62. W.M.P. van der Aalst., ter Hofstede, A.H.M., Kiepuszewski, B., Barros, A.P. Workflow Patterns. BETA Working Paper Series, WP 47, Eindhoven University of Technology, Eindhoven, 2000.
63. WebMethods. Making the Transformation to Service-Oriented Architecture: Capitalising on the Next Revolusion in IT. Whitepaper, AV-19-6751, January 2005.
64. Weijters A.J.M.M., Aalst W. M. P.v. Process Mining: Discovering Workflow Models froom Event Based Data. Proceedings of the 13th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC 2001), pages 283-290, 2001.
65. WfMC. WfMC Standard – Interoperability. Wf-XML Binding. Document number WFMC-TC-1023, Final Draft, v.1.1, 14 November 2001.
66. WfMC. Workflow Management Coalition Standard – Interface I: Process Definition Interchange Process Model. WFMC-TC-1016-P, Draft 7.0 beta, 1998.
67. WfMC. Workflow Management Coalition Standard – Interoperability: Wf-XML Binding. WFMC-TC-1023, version 1.1, 2001.
68. WfMC. Workflow management coalition terminology & glossary. Workflow Management Coalition. Document status - Issue 3.0, 1999.

69. 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.
70. Xu, J., Romanovsky, A. and Randell, B. Coordinated Exception Handling in Distributed Object Systems: From Model to System Implementation. In Proceedings of the 18th IEEE International Conference on Distributed Computing Systems (ICDCS '98), IEEE Computer Society Press, Amsterdam, The Netherlands, 1998, pp. 12-21. 1998.
71. Zhongqiao Li. Using Collaboration Task In Orbwork Enactment System for METEOR Workflow Management System. Master Thesis, Shanghai Jiao Tong University, China, 1996.
72. Zongwei Luo, A. Sheth, K. Kochut and J. Miller.: Exception Handling in Workflow Systems. Applied Intelligence, Volume 13, Number 2. September/October, 2000: pp125-147.
73. Zongwei Luo. Exception Handling for Conflict Resolution in Cross-Organizational Workflows. Distributed and Parallel Databases, Kluwer Academic Publishers. Manufactured in The Netherlands, Volume 13, 2003, pp. 271–306,
74. Zongwei Luo. Knowledge Sharing, Coordinated exception Handling, and Intelligent Problem Solving for Cross-Organisational Business Processes, PhD Thesis, 2001, Computer Science Department, The University of Georgia.
75. zur Muehlen, M., Allen, R.. Workflow Classification. Embedded & Autonomous workflow Management Systems. WFMC. 2000.

Internet Resources

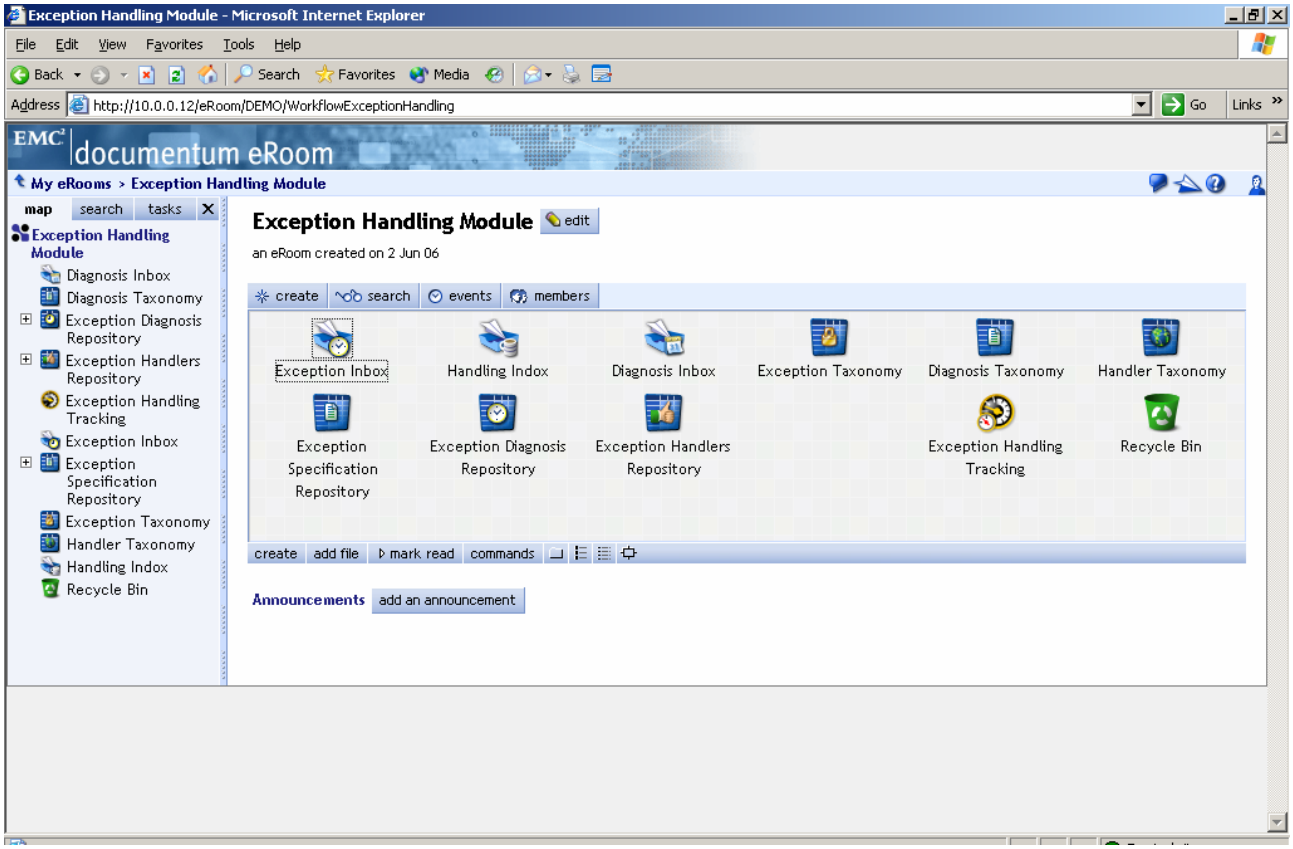
76. BABEL – Tools. http://www.bpm.fit.qut.edu.au/projects/babel/tools/
77. BPMG - Business Process Management Group. http://www.bpmg.org
78. CrossFlow. http://www.crossflow.org/
79. Elektronika. http://www.elektronika.lt
80. E-Workflow – the Workflow Portal. http://e-workflow.org/
81. FreeFlow. http://www.xerox.com/go/xrx/portal/STServlet?projectID=ST_FreeFlow&pageID=FreeFlow_Workflow&Xcntry=USA&Xlang=en_US
82. Klase. http://www.klase.lt
83. Loosely Coupled. http://www.looselycoupled.com
84. Mariflow. http://www.srdc.metu.edu.tr/webpage/projects/mariflow
85. OMG – Object Management Group. http://www.bpmi.org/
86. OVAL. http://www.fluent.com/about/news/newsletters/05v14i2/a5.htm

87. Stergiou, C., Siganos, D. Neural Networks. http://wwwhomes.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html.
88. TRAMs project. http://www.ifi.unizh.ch/dbtg/Projects/TRAMs/trams.html#description
89. WfMC - Workflow Management Coalition. http://www.wfmc.org
90. Wikipedia Encyclopedia. http://en.wikipedia.org/wiki/
91. Workflow Patterns. http://is.tm.tue.nl/research/patterns/index.htm

Annex No. 1 – B2B exception handling system overview

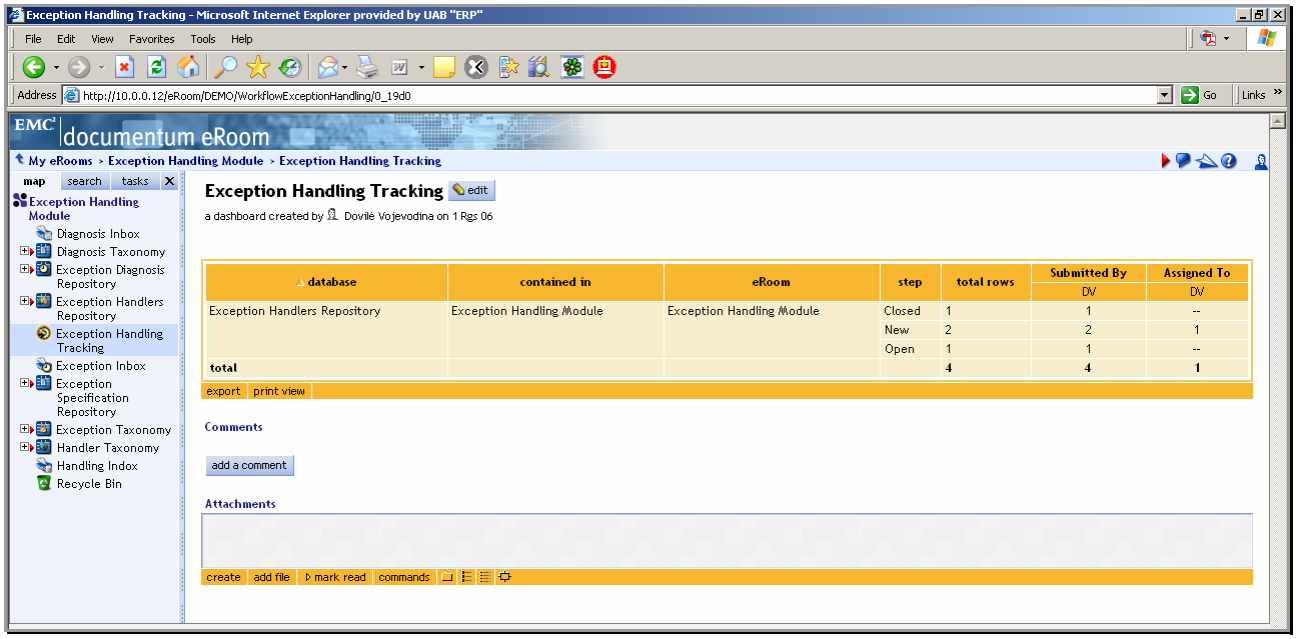
This section is dedicated to the overview of the created and tested B2B exception handling system, which was created during this research. The overall architecture and methods of creation of this system were discussed in detail in section 4 of this research.

The system uses all advantages of SOA, therefore is very flexible as is can be used as self standing application and as integrated into B2B application module. The advantage of SOA separated architectural layers brings possibilities to manage data, workflow, module administration and integrations separately.



Picture 63. Main interface of B2B exception handling system.

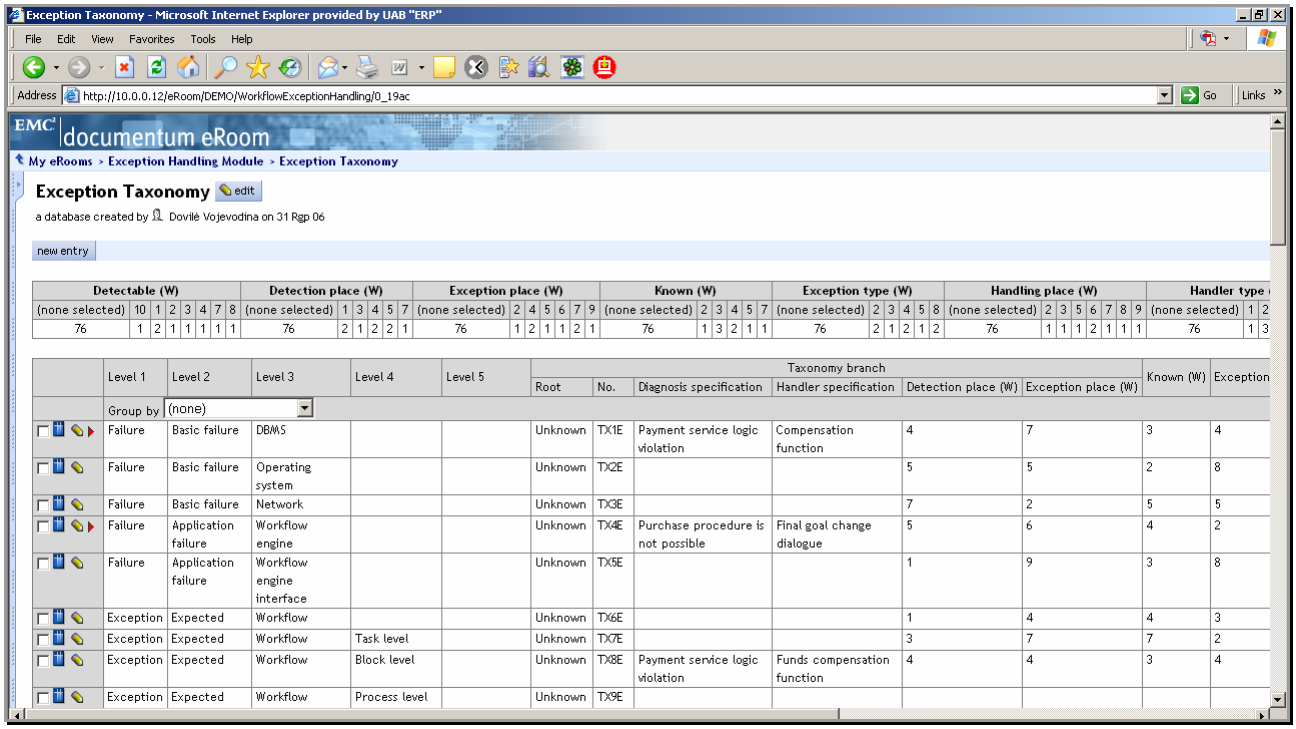
Picture 63 depicts the main interface of exception handling system, which has 6 logically separated repositories: 3 taxonomy and 3 specification repositories, managed separately but closely related with each other. In addition each of specification repositories has its own e-mail box, which help to sort the received (manually or automatically) exception handling requests. The main monitoring of exception handling can be viewed using Exception Handling Tracking dashboard (see Picture 64).



Picture 64. Exception handling process monitoring dashboard.

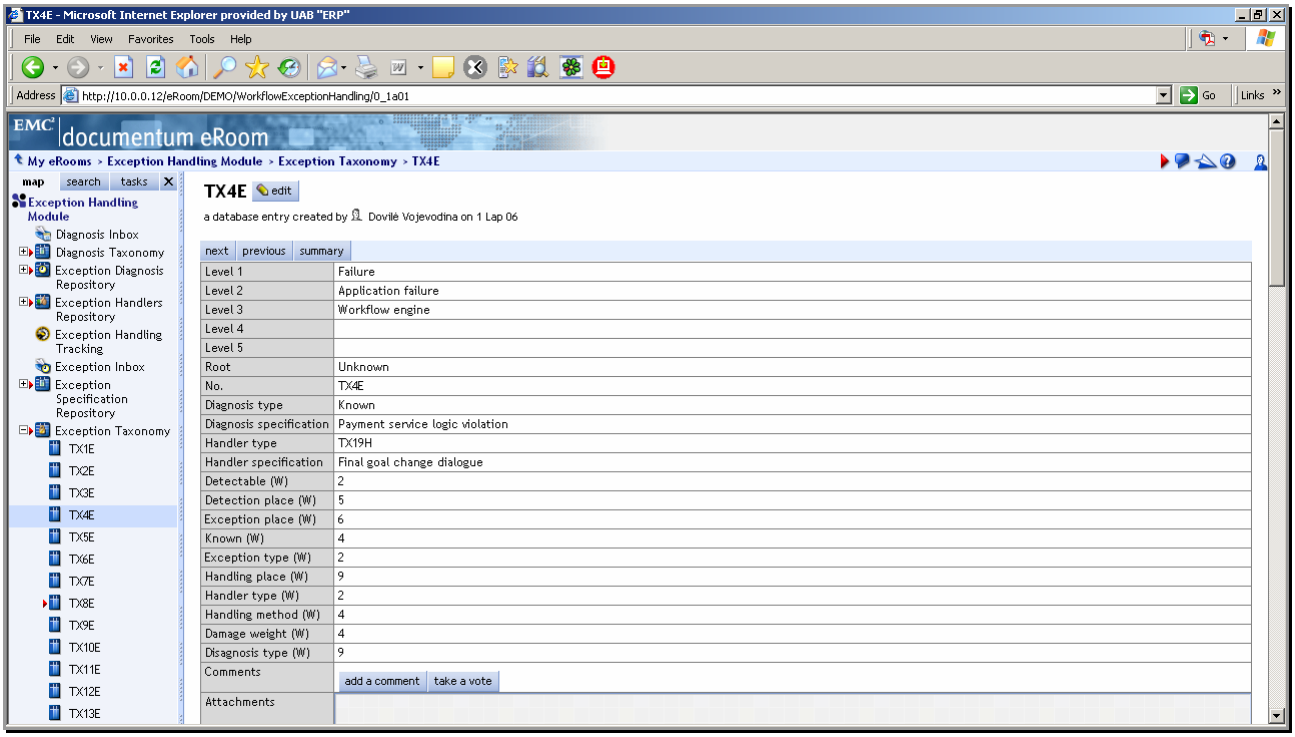
The level of exception taxonomy branches and their depth is unlimited therefore the taxonomy can be supplemented with new branches any time and give the wanted level of details on any branch.

Each of taxonomy branches has linking related information on specifications, which are used for current branch description.



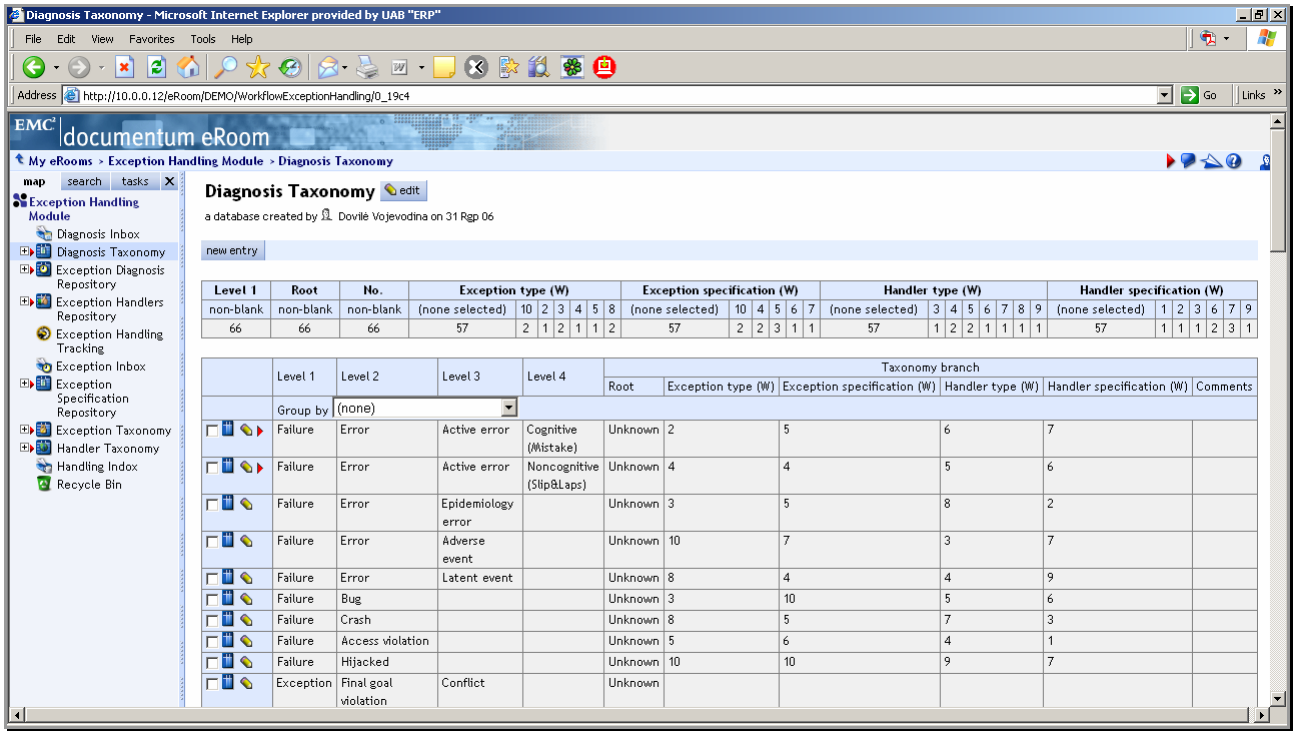
Picture 65. Exception taxonomy module, which has all possible branches and can be supplement with new ones at any moment.

Each record of taxonomy branch (see Picture 66) can be described with as many attributes as it is needed. The collection of attributes can be supplemented with new ones at any moment. Also each record has a set of pre-described weights which are used for similarity level calculation. The overall weights statistics for all taxonomy records can be monitored at the mini-dashboard above taxonomy records table (see Picture 65).



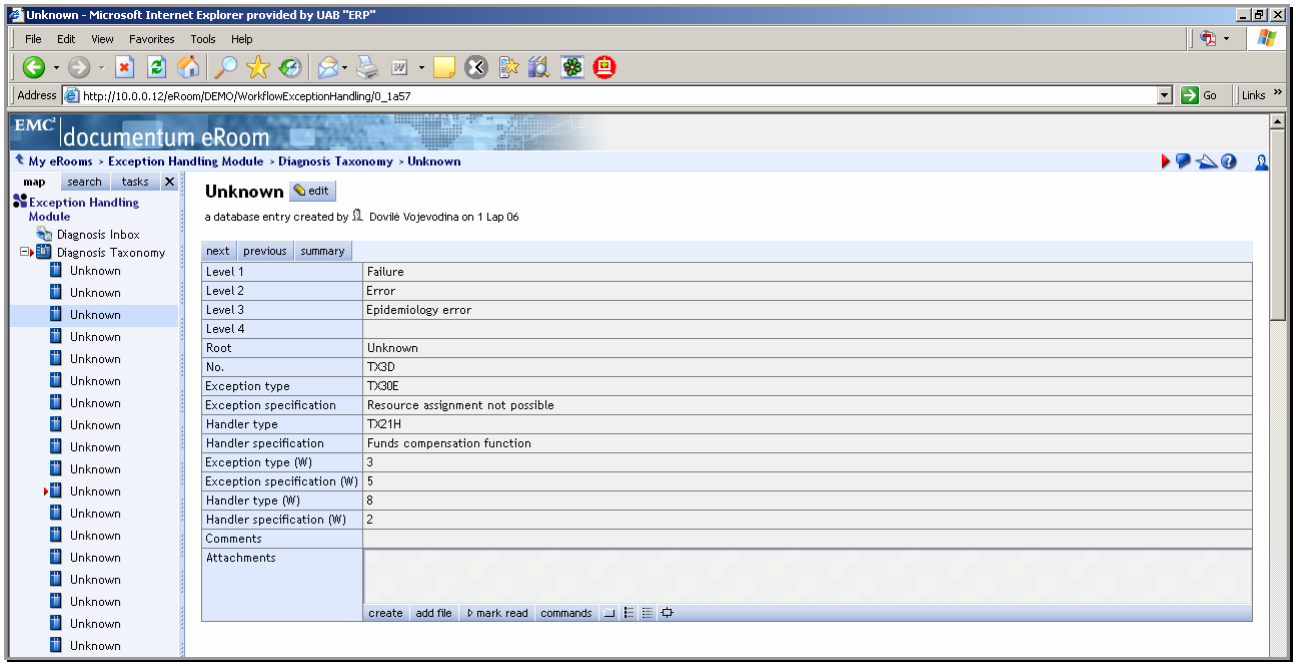
Picture 66. Exception taxonomy branch record.

The diagnosis taxonomy is created and managed in the way as exception taxonomy, only it has the different set of description attributes and weights collection.

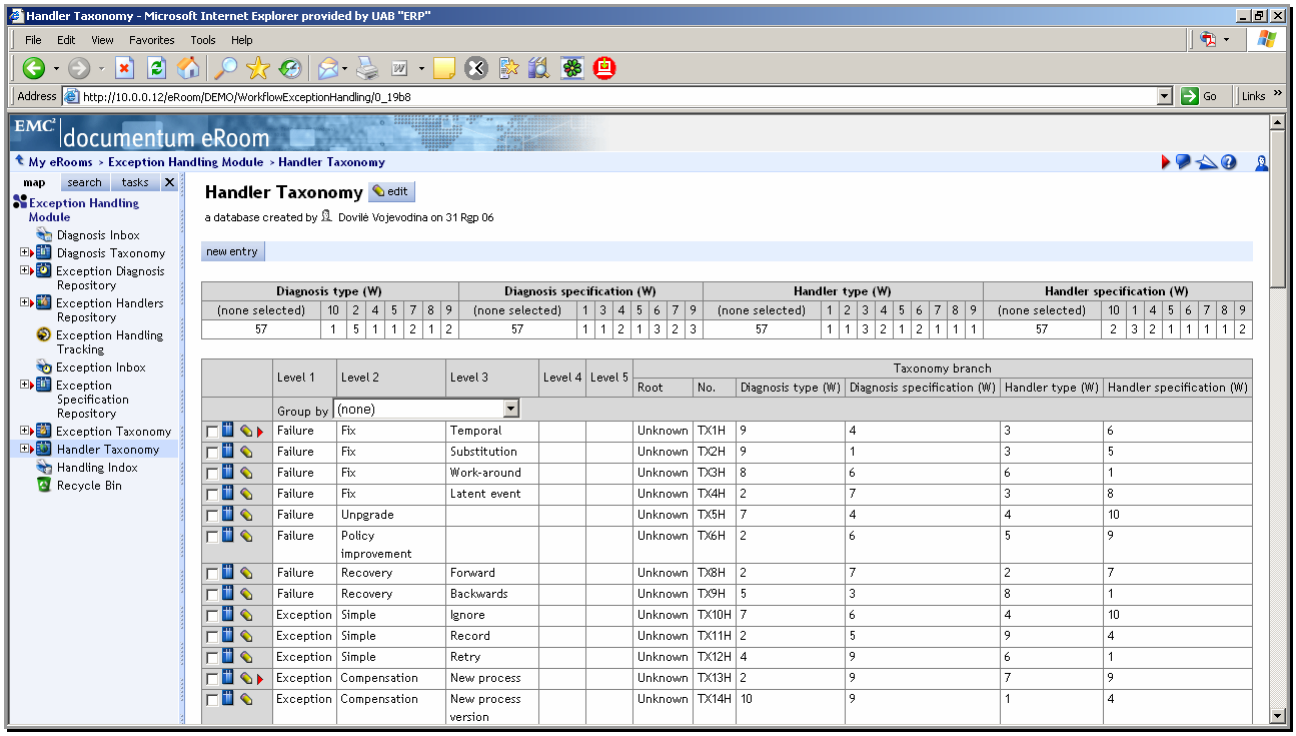


Picture 67. Diagnosis taxonomy module, which has all possible branches and can be supplement with new ones at any moment.

The diagnosis taxonomy branch is also created and managed the same way as exception taxonomy branch, collecting only different set of attributes and weights.

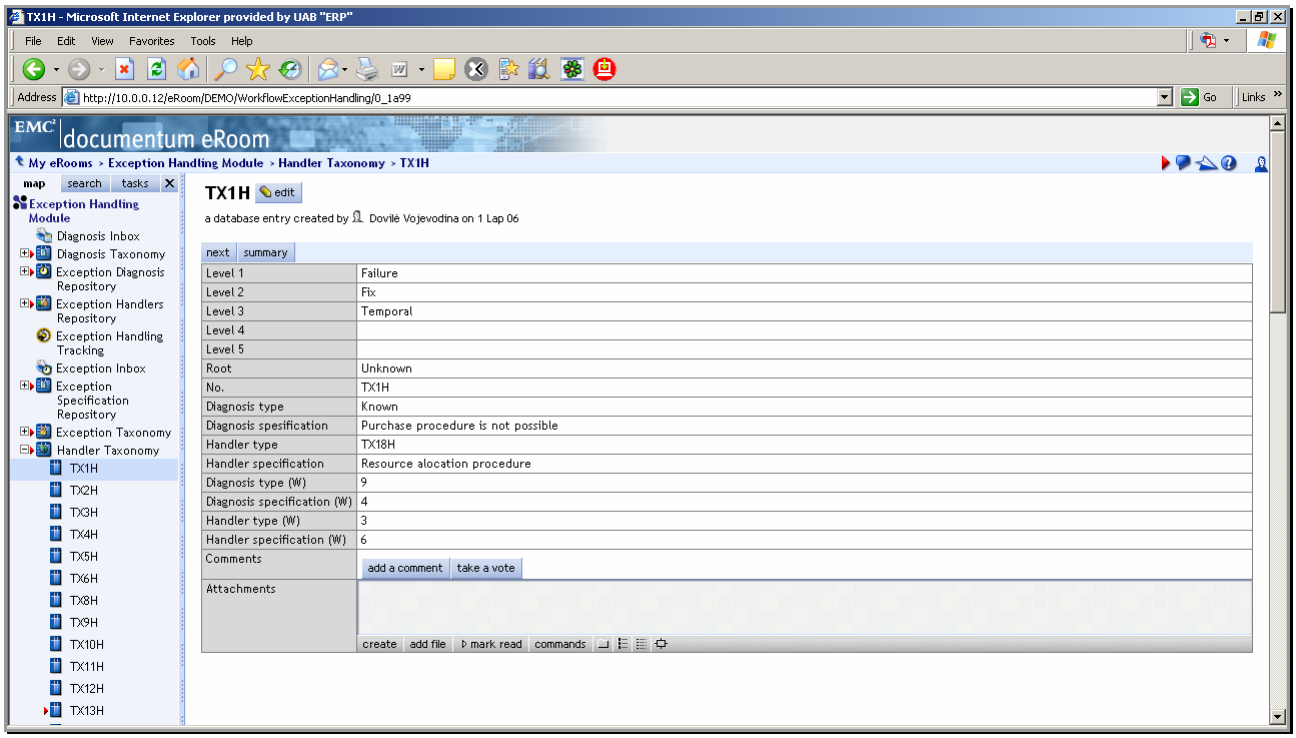

Unknown - Microsoft Internet Explorer provided by UAB "ERP"
File Edit View Favorites Tools Help
Address http://10.0.0.12/eRoom/DEMO/WorkflowExceptionHandling/0_1a57
documentum eRoom
My eRooms › Exception Handling Module › Diagnosis Taxonomy › Unknown
map search tasks
Exception Handling Module
Diagnosis Inbox
Diagnosis Taxonomy
Unknown
Unknown
Unknown

The exception handlers' taxonomy is created and managed in the way as exception taxonomies, only it has the different set of description attributes and weights collection.



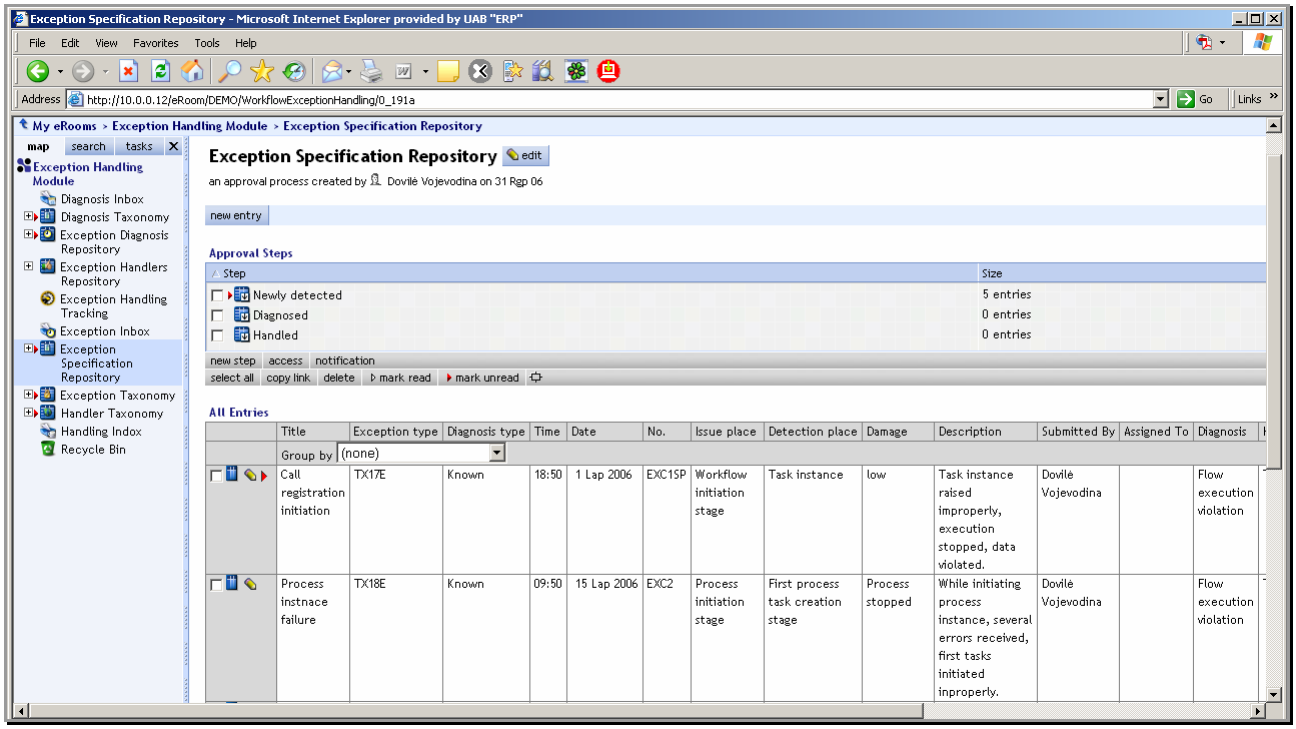
Picture 69. Handler taxonomy module, which has all possible branches and can be supplement with new ones at any moment.

The exception handlers' taxonomy branch is also created and managed the same way as exception taxonomy branch, collecting only different set of attributes and weights.



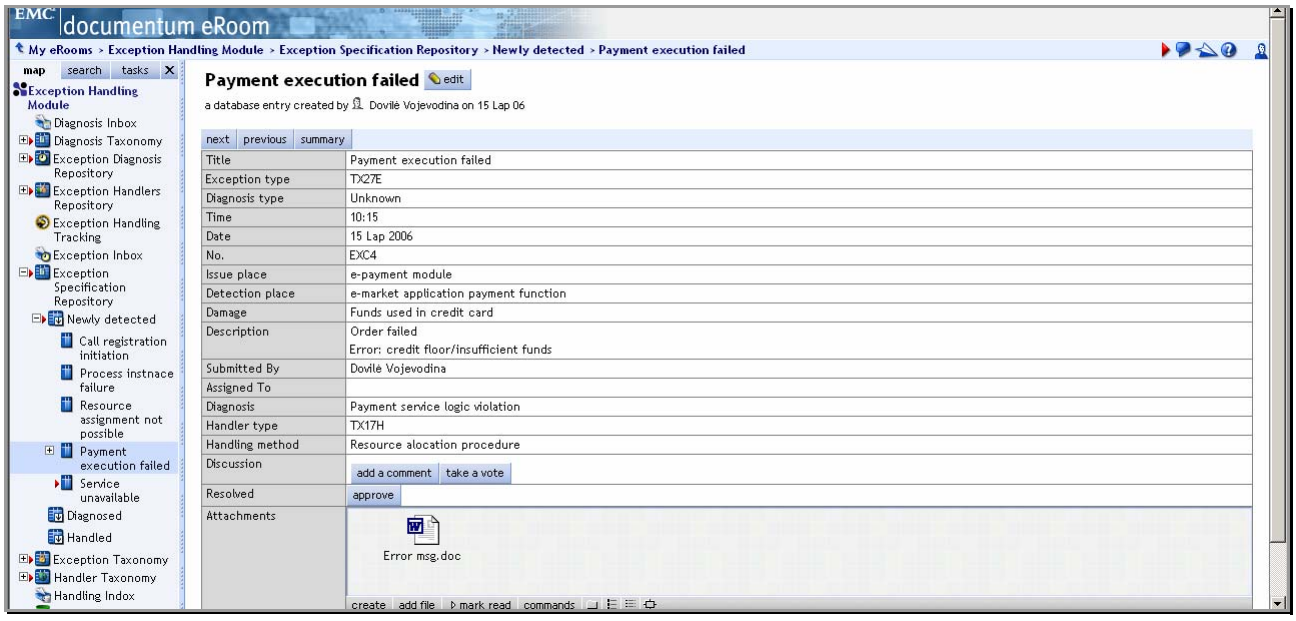
Picture 70. Exception handler taxonomy branch record.

Exception specification repository is dedicated for description in detail of each exception taxonomy branch. Each specification has supplemented information from other taxonomies and repositories, on the matter which diagnosis or handler refer to each exception specification.



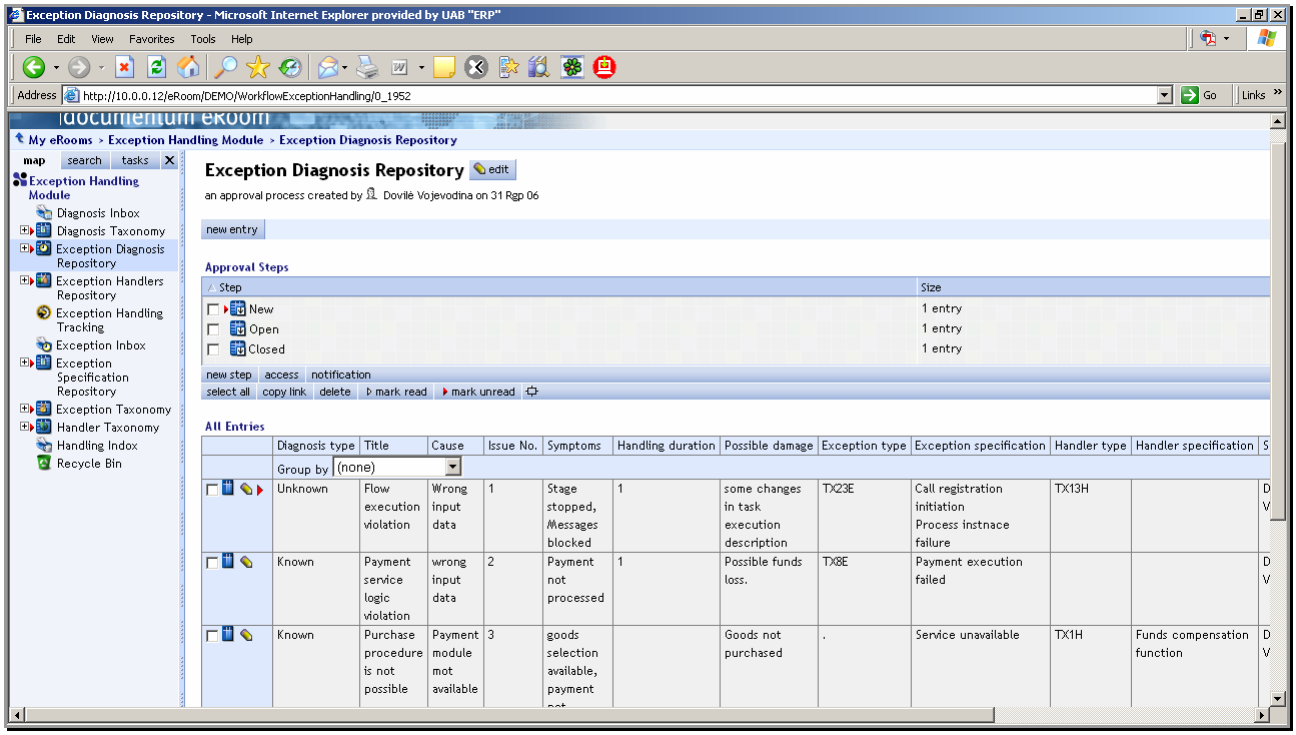
Picture 71. Exception specification repository with main lifecycle steps and collected records.

The new specification can be created any moment using manual or automated way. The initiation of new specification creations can be e-message from exception mailbox.



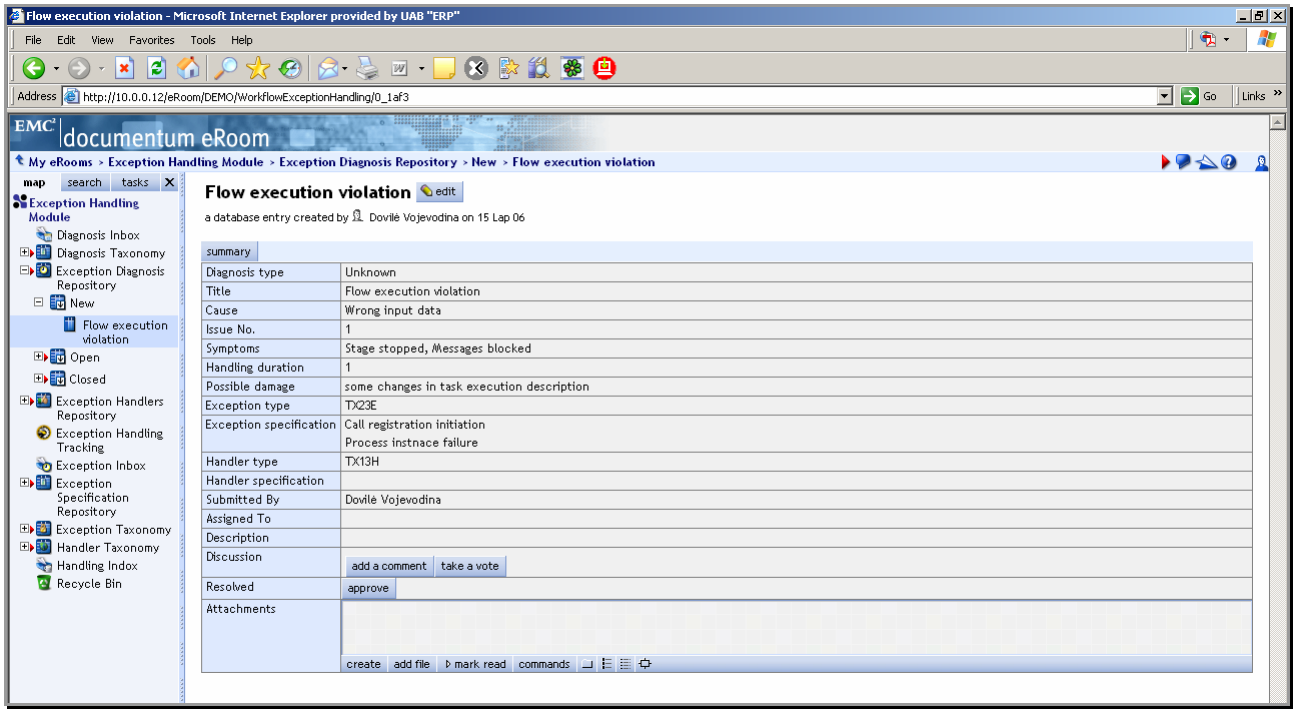
Picture 72. Exception repository record, containing full description of attributes and descriptive information in attachment box.

The exception diagnosis specification repository is created and managed the same way as exception specification repository. The method of creation of the specifications and their management was described in detail in chapter 4 of this research.



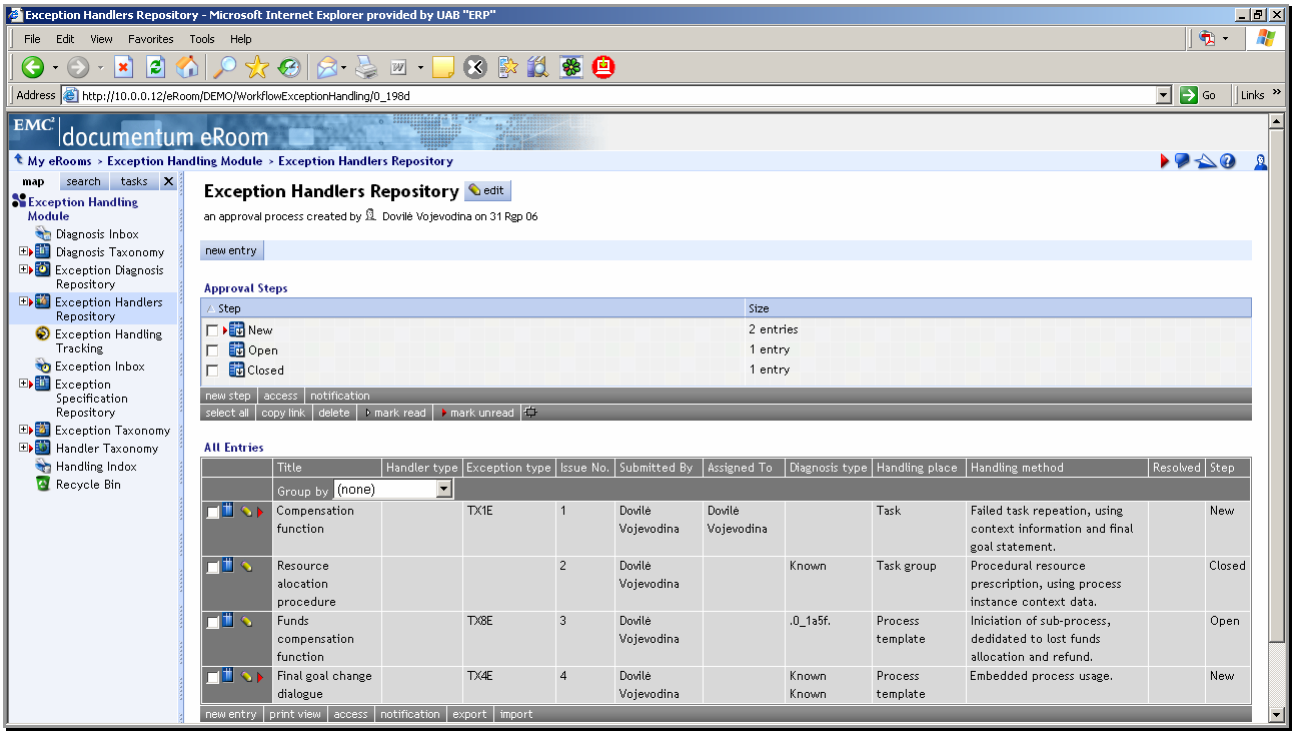
Picture 73. Diagnosis specification repository with main lifecycle steps and collected records.

The record of diagnosis specification is also managed the same way as exception specification and fully approbated using workflow.



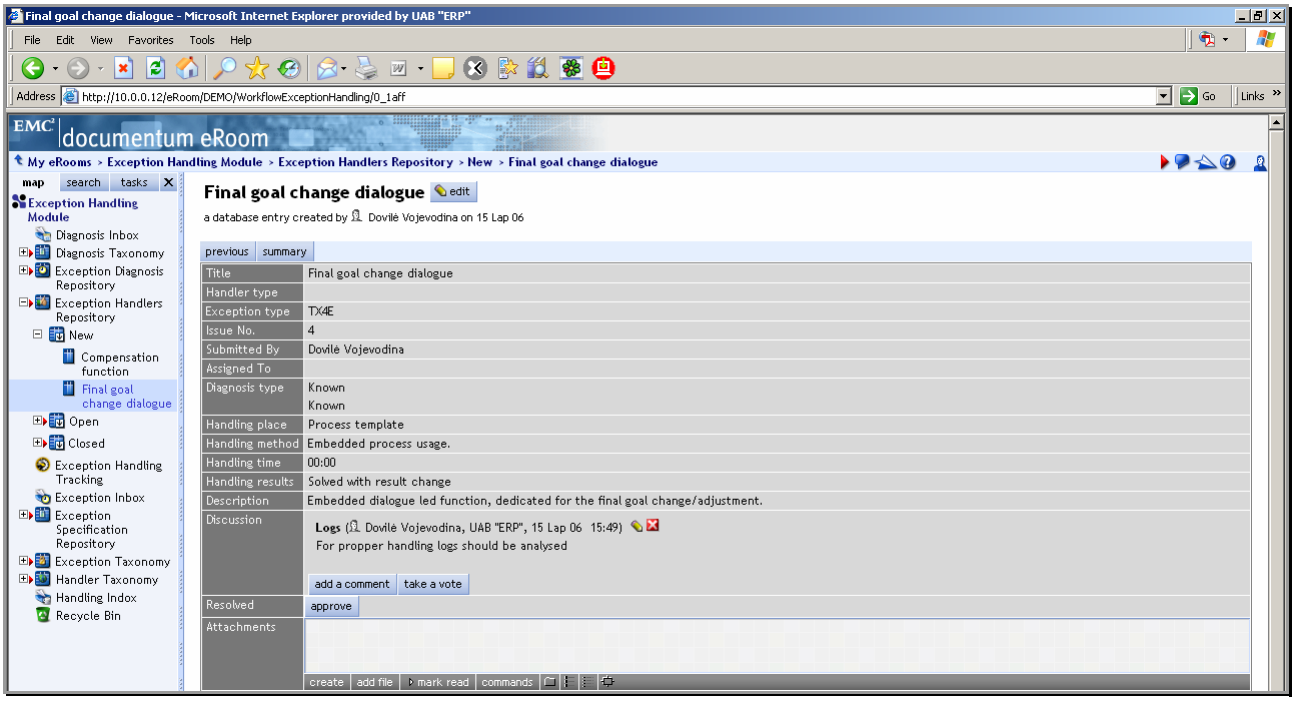
Picture 74. Diagnosis repository record, containing full description of attributes.

The exception handler specification repository is created and managed the same way as exception and its diagnosis specification repository. The method of creation of the specifications and their management was described in detail in chapter 4 of this research.



Picture 75. Exception handlers' specification repository with main lifecycle steps and collected records.

The record of hander specification is also managed the same way as exception and its diagnosis specification and fully approbated using workflow.



Picture 76. Exception handler repository record, containing full description of attributes.