VILNIUS GEDIMINAS TECHNICAL UNIVERSITY

Aidas ŠMAIŽYS

A STUDY ON IMPLEMENTATION OF AUTOMATED DECISION PROCESS INTO THE INFORMATION SYSTEMS

DOCTORAL DISSERTATION

TECHNOLOGICAL SCIENCES,
INFORMATICS ENGINEERING (07T)

Vilnius LEIDYKLA TECHNIKA 2011

Doctoral dissertation was prepared at Vilnius Gediminas Technical University in 2007–2011.

Scientific Supervisor

Prof Dr Olegas VASILECAS (Vilnius Gediminas Technical University, Technological Sciences, Informatics Engineering – 07T).

VGTU leidyklos TECHNIKA 1955-M mokslo literatūros knyga
http://leidykla.vgtu.lt

ISBN 978-609-457-037-7

aidas@isl.vgtu.lt

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

Aidas ŠMAIŽYS

SPRENDIMŲ PROCESŲ AUTOMATIZAVIMO INFORMACINĖSE SISTEMOSE TYRIMAS

DAKTARO DISERTACIJA

TECHNOLOGIJOS MOKSLAI,
INFORMATIKOS INŽINERIJA (07T)

VGTU
LEIDYKLA
TECHNIKA
Vilnius 2011

Disertacija rengta 2007–2011 metais Vilniaus Gedimino technikos universitete.

Mokslinis vadovas

prof. dr. Olegas VASILECAS (Vilniaus Gedimino technikos universitetas, technologijos mokslai, informatikos inžinerija – 07T).

Abstract

Many studies on software project failures have identified difficulties in capturing requirements, managing complexity and dynamic changes of the environment influenced by the changes of evolving social and competitive business environment around us. This makes static capture of requirements and long implementation according to the traditional software engineering leading to the problems that can be solved by using models. This is especially important for decision-making, which is such a dynamic sphere of business.

Business people who are first of all directly responsible for the results of automated business processes are left isolated from directly impacting the rules in automated decision-making processes, which are usually hidden in the code. This can be tolerated in case of simple operational decisions; however, it becomes a real problem when trying to automate more abstract tactical and strategic decisions based on complicated and unstable business logic and rules.

In the presented thesis we offer modernisation of information system development methods used for implementation of automated information-, rule-, knowledge- and model-based decision processes assisted by early separation and development of a business logic model and implementation of decision-making and knowledge discovery process models with further support to business people with suitable interfaces for modification of decision-making processes without involvement of software developers in later exploitation stages.

In the analytical part of the dissertation, decision models and methods of intellectualised information systems are analysed. According to the results obtained during the analysis, investigation on the proposed framework and a decision model based method for decision-making process automation is carried out.

In the final chapter several experiments are described in order to evaluate the proposed method, and the general conclusions complete the research.

Reziumė

Eilė informacinių sistemų projektų studijų parodė, kad nesėkmingi projektai dažniausia susiduria su problemomis specifikuojant reikalavimus, valdant sudėtingus ir dinamiškai vykstančius pokyčius verslo aplinkoje. Tai daro reikalavimų surinkimą pakankamai statišku, o tokių reikalavimų pokyčių sąlygoti informacinių sistemų pakeitimai atliekant tradicinius programų sistemų inžinerijos procesus pasidarė pernelyg lėti ir nebeatitinka realių dinamiško verslo poreikių. Atlikti tyrimai parodė, kad tokios problemos tam tikrais atvejais gali būti sprendžiamos vietoje reikalavimų panaudojant modelius.

Verslas jaučia, kad paprastai įdiegus sprendimus informacinėse sistemose yra prarandama tiesioginė tokių automatizuotų sprendimų kontrolė, o aklai pasitikėti sistemos priimamais sprendimais dažnai negalima – atsiranda atsakomybės problema. Šios sąlygos lemia, kad nuolat besikeičiančioje ir dažnai sunkiai prognozuojamoje verslo aplinkoje vykstančių sprendimų automatizavimui reikalingi nauji metodai, kurie atskirtų sprendimų logiką, žinias ir patį sprendimų procesą bei leistų juos vėliau modernizuoti be programų sistemų kūrėjų pagalbos.

Disertacijoje tiriamas sprendimų modelių kūrimas verslo taisyklių ir procesų modelių pagalba bei šių modelių transformacijos į programų sistemų komponentus realizuojančius automatizuotus sprendimus informacinėje sistemoje.

Analitinėje disertacijos dalyje apžvelgiami sprendimų modeliai ir metodai skirti intelektualizuotų informacinių sistemų kūrimui. Tolimesniuose skyriuose pateikiamas remiantis analizės rezultatais sukurtas modelių karkasas ir jo pagrindu sukurtas metodas skirtas sprendimų modelių automatizavimui.

Baigiamajame disertacijos skyriuje aprašomi eksperimentai, kuriuose atliekamos sprendimų modelių transformacijos. Galiausiai pateikiami eksperimentų rezultatai ir galutinės išvados.

Notations

Symbols

Rm – Rule model;
Dm – Decision model;
M_v – Model in the view column of the framework;
M_c – Model in the context row of the framework.

Abbreviations

ADO – ActiveX Data Objects;
AI – Artificial Intelligence;
API – Application Programming Interface;
BI – Business Intelligence;
BEM – Business Event Model;
BDM – Business Metrics Model;
BMM – Business Motivation Model;
BOM – Business Object Model;
BPM – Business Process Model;
BR – Business rule;
BRM – Business Rule Model;
BRsM – Business Resource Model;
BS – Business System;

BSM – Business System Model;
BPMN – Business Process Modelling Notation;
BRG – Business Rules Group;
CDM – Conceptual Data Model;
DMX – Data Mining Extensions to SQL;
DPM – Data Processing Model;
DRL – Decision Representation Language;
DSS – Decision Support System;
EIS – Executive Information System;
IEM – Informational Even Model;
IIM – Informational Infrastructure Model;
IIS – Intellectualised Information System;
IS – Information System;
ISM – Information System Model;
IRM – Information Analysis Rule Model;
IPM – Information Processing Model;
IOM – Information Object Model;
LDM – Logical Data Model;
LOC – Lines of Code;
MDX – Multi-dimensional Expressions;
MIS – Management Information System;
OLAP – On-line Analytical Processing;
OLE DB – Object Linking and Embedding for Databases;
OM – Object Model;
OMG – Object Management Group;
ORM – Object Role Modelling;
PRM – Production Rule Model;
PRR – Production Rule Representation;
UML – Unified Modelling Language;
SBVR – Semantics of Business Vocabulary and Business Rules;
SRML – Simple Rule Markup Language;
PRR – Production Rule Representation;
SEM – Software Event Model;
SSM – Software System Model;
SWRL – Semantic Web Rule Language;
TIM – Technical Infrastructure Model;
RuleML – Rule Markup Language;
SS – Software System;
SSIS – SQL Server Integration Services;
SQL – Standard Query Language.

Terms Adopted by the Author

Decision is a judgement, a conclusion or determination of business object state based on business process measures and knowledge. A decision is also a target value oriented process of cutting off from all alternatives to achieve the best alignment with the business policy and goals. Decisioning covers more specific processes of decision support, decision-making and decision analysis.

Decision support is a process of analytical data processing focusing on discovery of facts and preparation of the aggregated information used to aid decision-making by a human or machine agent.

Decision-making is a process of choosing among alternative courses of action or determination of the objectives for the purpose of attaining goals.

Decision implementation is a process of operational execution of the plan dedicated to assign and communicate the objectives to the corresponding agents. Or it is a pattern of activities executed using predefined procedure required to achieve the objectives.

Decision analysis is a technique to discover the most advantageous alternative under the circumstances. Decision analysis covers statistical analysis methods, development of probabilistic forecasting and mathematical models of the real-world problems that can be used for evaluation of possible decision impact on the system affected.

Contents

INTRODUCTION 1

The Investigated Problem 1
Importance of the Thesis 3
The Object of the Research 3
The Goal of the Thesis 4
The Tasks of the Thesis 4
Research Methodology 4
Importance of Scientific Novelty 5
Practical Significance of the Achieved Results 6
The Defended Statements 7
Approbation of the Results 7
Structure of the Dissertation 8
Acknowledgements 10

1. DECISION MODELS IN INTELLECTUALISED INFORMATION SYSTEMS 11

1.1. Definition of Decision-Making in Business and Information Systems 12
1.2. Decision Classification 15
1.3. Decision-making Models 18

1.4. Business Logic in Enterprise Information Systems 21
1.5. Related Work Analysis 26
1.5.1. Barbara von Hale *et al.* Method 27
1.5.2. OMG (BRG) Approach 28
1.5.3. Avdejenkov and Vasilecas method 30
1.5.4. Asuncion *et al.* Method 31
1.5.5. IBM (ILOG) Approach 32
1.5.6. Microsoft Approach 33
1.5.7. Comparison of the Methods 34
1.6. Conclusions of Chapter 1 and Formulating Tasks for the Dissertation 37

2. THE FRAMEWORK FOR INTELLECTUALISED INFORMATION SYSTEMS ENGINEERING 39

2.1. Frameworks for Enterprise Information Systems Development 40
2.2. Conceptual Modelling Perspectives 41
2.3. The Three Layer Framework 43
2.3.1. Framework Model Transformations 48
2.3.2. Infrastructure Framework for Intellectualised IS development 51
2.3.3. The Lifecycle of Framework Based Engineering 53
2.3.4. Formal Framework Based Method Definition 54
2.4. Principles and Approaches of the Framework Application 56
2.4.1. Separate Development of a Business Logic Model and Integration into the Information System 58
2.4.2. Centralized Business Logic Execution and Business Rule Enforcement in Integrated Systems 59
2.4.3. Intelligent Self Adaptable Systems 60
2.5. Conclusions of Chapter 2 61

3. A METHOD FOR DECISION DESIGN AND IMPLEMENTATION USING THE PROPOSED FRAMEWORK 63

3.1. Model of Decision Process 64
3.1.1. Decision Model of the Guided Business Process 65
3.1.2. Decision Model of the Reactive Business Process 67
3.1.2. Decision Model of the Proactive Business Process 68
3.2. Decision Model 70
3.3. Method Implementation Schema 76
3.4. Model of Business Logic and Decision Process 78
3.5. Conclusions of Chapter 3 81

4. APPLICATION OF THE PROPOSED METHOD 83

4.1. Experimental Investigation on Parameterisation Approach 84
4.1.1. Decision Table Based Business Rule Representation 84

4.1.2. Parameterisation Based Decision Model Implementation 86
4.1.3. Rule Model Transformation and Implementation 88
4.1.4. Results of Experiment 92
4.2. The Experiment on Run-time Transformation Approach 93
4.3. The Experiments on the Use of Declarative Logic Models 96
4.4. Evaluation of the Results 97
4.5. Conclusions of Chapter 4 101

GENERAL CONCLUSIONS 103

REFERENCES 107

LIST OF PUBLICATIONS BY THE AUTHOR ON THE TOPIC OF THE DISSERTATION 113

List of Figures

Fig. 1.1. Relationships between an enterprise and its ISs. Extended EIS oriented schema initially introduced by Montilva *et al.* (2004) 22
Fig. 1.2. Increasing the potential of technical infrastructure to support business decision process 25
Fig. 1.3. A theoretical model of a decision process based on structured information (von Halle and Goldberg 2010) 27
Fig. 1.4. OMG (BRG) approach to decision model (Linehan *et al.* 2011) 29
Fig. 1.5. Business rule and process design model integration (Asuncion *et al.* 2010) 31
Fig. 2.1. The Multiview2 framework (Avison *et al.* 1998) 40
Fig. 2.2. System analysis perspectives - resource, process and function views 44
Fig. 2.3. Owner, designer and builder views to provide IS models 44
Fig. 2.4. Resource, process and functional views to provide BS, IS and SS models 45
Fig. 2.5. The three layer framework for model-driven information system engineering 46
Fig. 2.6. Framework model transformation model adapted from MDA 50
Fig. 2.7. Framework model transformations in the context of MDA 51
Fig. 2.8. The infrastructure framework for integrated intelligent IS for decision automation 52
Fig. 2.9. The lifecycle of continuous decision process design 54
Fig. 3.1. Decision model of the guided business process 65
Fig. 3.2. Sample illustration of the controlled business process model in UML activity diagram 66
Fig. 3.3. Decision model of the reactive business process 67

Fig. 3.4. Decision model of the proactive business process 69
Fig. 3.5. Decision model structure 71
Fig. 3.6. Decision model in the context of framework 73
Fig. 3.7. Decision model for intellectualised decision support system 75
Fig. 3.8. Method schema for implementation of the automated decision process 76
Fig. 4.1. The nodes of DTs metamodel and business rules in decision model 85
Fig. 4.2. DTs subjects, properties, variables and values 85
Fig. 4.3. Business rule based decision model implementation 87
Fig. 4.4. DT model transformation into the ER model 89
Fig. 4.5. ER Model of configuration set transformed from DT 90
Fig. 4.6. Intellectualised interface of port control system 92
Fig. 4.7. Model transformation based implementation model 94

List of Tables

Table 1.1. Comparison of the rule-based methods for decision automation support 35
Table 1.2. Comparison of the rule-based methods for decision automation completeness and flexibility 36
Table 2.1. List of the framework models and model transformations examined 49
Table 3.1. Decision Model Notation. 74
Table 3.2. Comparison of rule and process languages according to the views of conceptual modelling and modelling perspectives 80
Table 4.1. Evaluation of the proposed method application suitability 98
Table 4.2. SWAT analysis of the proposed framework based method 99
Table 4.3. Comparison of the results of decision process automation with the traditional and framework based methods 100

Introduction

The Investigated Problem

Business decisions are usually made by managers based on their own knowledge and information supplied by information workers or specific business information systems, so-called decision support systems. Managers' knowledge usually involves knowledge on legal regulation, corporate rules, skills, personal experience, etc. Such decision-making strictly depends on relevance of the data selected to prepare information supplied to the decision processor – the manager. It is evident that more relevant data and better knowledge lead to a better decision. However, managers rarely consider that they have enough information and that the quality of the information supplied is sufficient. It is also difficult to assess the knowledge and creativity that managers possess and use for decision-making. Usually condition of a decision processor (feelings, emotions), influence of environment, decision support, decision-making, reasoning processes and even decision motivation are only known for the manager himself. The knowledge gained by solving business problems and implementing decisions is an important asset of enterprise, but it is rarely lost for business. This encourages looking for a way to eliminate or reduce a human factor in making decisions as well as to conceptualize the knowledge used for decision-making and its implementation.

In 1994, the Standish Group published the first Chaos report which summarized the findings of Standish research and aimed at investigating causes of software project failure and finding key ways to reduce such failures. In 1994, it reported a shocking 16 % project success rate, another 53% of the projects were challenged, and 31% failed outright. In subsequent reports Standish updated its findings, and in 2009 it demonstrated a marked decrease in project success rates, with 32% of all projects delivered on time, on budget, with the required features and functions, 44% were late, over budget, and/or with less than the required features and functions, and 24% failed, which were cancelled prior to completion or delivered and never used. The results have been improved yet; however, the figures remained troublesome (Eveleens *et al.* 2010). Many studies on software project failures have identified difficulties in capturing requirements, managing complexity and dynamic changes of the environment influenced by the changes of the evolving world around us. Flexibility remains a major challenge related to decision process implementation and alignment in the scope of a constantly changing business system environment. This makes static capture of requirements and long implementation according to the traditional software engineering leading to the problems which can be solved by using models.

Nowadays information system engineering tries to fill this gap by developing a range of agile and heavyweight software development methodologies and approaches to implement better decision-support systems that point to what data might be collected and used for further interpretation and preparation of information presented for decisions made by business managers. Although such an extensive research has been carried out trying to implement business requirements and models into the information systems software and to eliminate misunderstandings between business people and programmers in the development of decision-support oriented information systems, it has led to leaving the other part of decisioning unnoticed - the decision-making, analysis and implementation processes and representation of the knowledge required for decisions being automated. This is still a real challenge for the engineers of modern information systems.

Currently, software architectures introduce separate components representing a business logic layer used to concentrate and simplify implementation of changes in the business logic. However, that is insufficient because a formal and informal heavyweight requirement or even model driven methods of IS engineering still require wide-range involvement of software developers. Moreover, business people who are first of all directly responsible for the results of automated business processes remain isolated from directly impacting the rules in automated decision-making processes, which are usually hidden in the code. This can be tolerated in case of simple operational decisions; however, it be-

comes a real problem when trying to automate more abstract tactical and strategic decisions based on complicated and unstable business logic and rules.

The final target of IS engineering is a software. However, from the business perspective it should be an efficient business system functioning in harmony with a not computerised IS part and the part of IS which will be automated in the final SS. That becomes more important when we start automation of business decisions, which previously was an exclusive prerogative left for humans only. According to the findings of our research, even modern methods lack such business orientation.

In the presented thesis we offer modernisation of information system development methods used for implementation of automated information-, rule-, and model-based decision processes assisted by early separation and development of a business logic model and further support for business people with suitable interfaces for modification of business logic without any involvement of software developers or minimising their persistence in the later exploitation stages.

Importance of the Thesis

In the thesis we propose an approach of integrated use of modern information system development approaches and application of a proposed three layer framework based methodology for development of model transformation based methods of decision process automation and information system software development. Particular attention is given to separation of business logic and business processes at the early stages of information system development and involvement of business people into the adaptation and optimisation of a business logic model used for automated decisions implemented into the running software systems.

The Object of the Research

The object of the present study is the research on implementation of a goal-driven, flexible business process in enterprises, with a special focus on automation of a knowledge-based decision-making process, informational decision support and separation of business logic.

The Goal of the Thesis

The main goal is to investigate the existing business decision-making models and to propose a method for the business rule and decision model based decision automation and its implementation into the IS software.

The Tasks of the Thesis

In order to achieve the goal, the following problems had to be solved:

1. To investigate the existing models and methods used for decision-making automation and its implementation into the IS.
2. To review the existing frameworks used for IS development and propose a model based framework suitable for separation of business logic, decision support and decision-making processes at the business level with further development and implementation of an automated decision model.
3. To investigate and elaborate a method used for framework based decision model development and model transformation into the executable IS models and corresponding software components.
4. To experimentally examine the proposed method by developing the intellectualised IS software prototype used to verify the method suitability and evaluate the efforts gained.

Research Methodology

The research methodology involves theoretical analysis of scientific literature in order to investigate the object of the research, generalisation of the research and synthesis of the knowledge which improve understanding and help to find a solution to the problem. The classification is used to summarise strengths, weaknesses and gaps of the related methods found during literature analysis. An experimental research was carried out to gain experience and find a better new way of achieving the goal of the thesis. The methods of the experimental research were also applied to evaluate the proposed method of flexible information system engineering, the framework and the method proposed.

Importance of Scientific Novelty

Significance of scientific novelty for theoretical and experimental investigation of business oriented information systems software development according to the proposed framework based method is as follows:

1. A new business decision modelling approach is proposed. The approach employs an idea of a flexible automated system control process which is composed according to the models needed for representation of business logic, analytical information processing, decision impact evaluation and implementation.
2. The present study has introduced a framework for the development of business oriented decision models and model transformation based implementation into the executable models and IS software components.
3. The proposed method contributes to the development of a flexible decision process and includes separation of business logic, decision motivation, decision support and decision-making processes with inclusion of implicit knowledge which is a new addition in the field of decision process automation.
4. The present study focuses on flexible knowledge reuse and transparent decision logic representation at the business level used for better business cooperation, including the effect of direct business people knowledge incorporation and solution of the problem of relocated responsibility for automated decisions to the technogenic systems.
5. The proposed method has been applied in several experiments made. The obtained results demonstrate that the use of a separately developed conceptual decision model improves understanding of a decision process by the participants of an engineering process, accelerates its implementation into the IS software and facilitates further changes when necessary.

According to Linehan *et al.* (2011), Decision Model and Notation (DMN) do not exist yet. The new publications in this field by Barbara von Hale *et al.* (2010), Ronald Ross (2010), and Taylor *et al.* (2007) have highlighted the need for new decision modelling methods and techniques which are on the focus of attention in our present study.

Practical Significance of the Achieved Results

The methods developed according to the methodology presented in the thesis have been included into the report of VeTIS project (VGTU project manager: Prof. Dr. O. Vasilecas.). It was initiated seeking to improve the quality of business model-based development of information systems.

While preparing the doctoral thesis a part of the research results have been implemented and tested on the information systems developed at SC Klaipedos Nafta and used for the development of the Pedestrian and Vehicle Access and Cargo Control System at the Klaipeda Seaport. This practical testing provided valuable information and experience used for further improvement of the proposed method and elimination of the drawbacks identified.

Practical evaluation of the methods created according to the proposed framework has demonstrated that it is almost impossible to correctly specify all business logic during the analysis and specification stage by creating requirements. Moreover, it has been determined that the proposed methodology allows minimizing a gap between business logic implemented into the information system software and an evolving business strategy which often makes business logic to become out-dated just having started developing the information system.

We have also understood that a traditional way of IS development leads to early isolation of business people from the software development process leading to low quality of the software, especially in the initial versions. Further improvement of the software is complicated and in many cases it exceeds the project limits and leads to the failure of the project. However, development of a separate business logic model according to the proposed methodology and providing the interface for business people allows keeping business people directly involved into the process of development by letting them to specify business logic and decision rules. Information system software development in such a way allows improvement of the software quality because business logic is transparent from the very beginning, and business people responsible for the results produced by the designed software take responsibility for the rules that are specified by themselves and implemented by using automated transformation without further involvement of software developers. That facilitates further changes of business logic even in a post-production stage of the project.

The Defended Statements

The following statements based on the results of the present investigation may serve as the official hypotheses to be defended:

1. The proposed framework based method allows development of the decision model and its implementation into the IS by using model transformations and improves IS flexibility and adaptability.
2. The proposed method provides separation of the decision logic, decision support and decision-making processes in a decision model, allows transparent representation and ensures better understanding of the automated decision process by business people.

Approbation of the Results

The results of the dissertation are published in 37 scientific articles. 12 articles are in the reviewed scientific periodical publications and 25 articles are published in other scientific editions.
The results of the thesis were presented in 17 international and Lithuanian conferences:

- The 14th Conference for Lithuanian Junior Researchers in Lithuania "Science – Future of Lithuania", 15 April 2011, Lithuania, Vilnius, Vilnius Gediminas Technical University;
- The 13th Conference for Junior Researchers "Fundamental Research and Innovation", 6 May 2011, Klaipeda, Klaipeda University;
- The 12th Conference for Junior Researchers "Fundamental Research and Innovation", 22–23 April 2010, Klaipeda, Klaipeda University;
- The 13th East-European Conference on Advances in Databases and Information Systems (ADBIS-2009), 7–10 September 2009, Riga, Latvia;
- The 11th Conference for Junior Researchers "Fundamental Research and Innovation", 22–23 April 2009, Klaipeda, Klaipeda University;
- The 17th International Conference on Information Systems Development (ISD2008), 25–27 August 2008, Paphos, Cyprus;
- The 8th International Baltic Conference on Databases and Information Systems (Baltic DB&IS 2008), 2–5 June 2008, Tallinn, Estonia, Tallinn University of Technology;

- The 14th Conference on Information and Software Technologies IT 2008, 24–25 April 2008, Lithuania, Kaunas, Kaunas University of Technology;
- The 11th East European Conference on Advances in Databases and Information Systems (ADBIS 2007), 29 September–3 October 2007, Varna, Bulgaria;
- Conference on Information Technologies 2007, 31 January –1 February 2007, Kaunas;
- The 10th Conference of Students' Scientific Society (GMMF SMD), 26–27 April 2007, Lithuania, Klaipeda, Klaipeda University;
- The 10th Conference for Lithuanian Junior Researchers in Lithuania "Science – Future of Lithuania", 13 April 2007, Lithuania, Vilnius, Vilnius Gediminas Technical University;
- The 15th International Conference on Information Systems Development ISD`2006, 31 August–2September, 2006, Budapest, Hungary;
- International Conference on Computer Systems and Technologies CompSysTech'2006, 15–16 June 2006, Bulgaria, Veliko Tarnovo, University of Veliko Tarnovo;
- The 20th International Conference on Systems for Automation of Engineering and Research (SAER-2006), 22–24 September 2006, Varna, Bulgaria;
- The 7th International IEEE Baltic Conference on Databases and Information Systems, 3 July 2006, Vilnius, Lithuania;
- The 19th International Conference on Systems for Automation of Engineering and Research (SAER-2005), 24–25 September 2005, Varna, Bulgaria.

Structure of the Dissertation

The dissertation consists of four main chapters.

Chapter 1 provides a brief introduction of the relevant theoretical concepts, introduces the main problem analysed in the dissertation and its importance. The task of this chapter is to review relevant literature on decision processes and decision models used in business seeking to evaluate the scope of decisions that can be implemented by using currently available reasoning algorithms and knowledge in the field of intellectualised information system engineering. The present research deals with evolving decision-making processes giving special attention to decisions at the operational, tactical and strategic level that could

enable flexible adaptation of the business system to the changes in the business system environment.

Chapter 2 describes our general framework for the development of intellectualised information systems and positions our work with respect to the current research on decision automation in flexible business processes. It examines how conceptual domain models may be created and what relations they have across different levels of abstraction and different modelling perspectives such as rule, process, information, structure, event, and resources. Different uses of the framework for creation of metamodel transformation based methods for model transformations are explained. Metamodel based transformation has been proposed for eliminating manual coding procedures and model transformations from the business level into the final software code. Moreover, this chapter deals with the aspects of conceptual model transformation into the executable models as well as development of an engineering procedure in order to provide a reliable and simple technique.

Chapter 3 presents a method developed by using the proposed framework. The proposed methodology includes a description of the process used for development of an experimental set of methods used to examine and demonstrate its suitability for different tasks of decision-making process development and further implementation into the information system.

Chapter 4 describes a series of experiments in which an application of the method supported by the proposed framework is evaluated. The experiments demonstrate how this method may be used for development of a set of conceptual models and metamodel transformation based model transformations used for rule and decision process model based decision-making automation by implementation into the information system code or run-time execution of rule models.

General conclusions as well as recommendations for further research summarize the present study. Finally, an extensive list of references and a list of publications prepared by the author on the topic of the dissertation are submitted.

Acknowledgements

I am very grateful to my Scientific Supervisor, Prof. Dr. Olegas Vasilecas, whose encouragement, guidance and support from the initial to the final stage of my doctoral studies allowed me to develop an in-depth understanding of the subject and to openly express my ideas. Thank you for a different viewpoint to the world and your concentration on the true values. Thank you for changing me as a person.

Moreover, I would like to say many thanks to my colleagues and friends at Vilnius Gediminas Technical University and Klaipeda University who have always helped me during all my studies. Thank you for all my time I have spent with you. I am especially grateful to Dr. Vitalijus Denisovas who was my first supervisor of my Bachelor's thesis. Thank you for seeding a passion for scientific research and for encouragement to continue my scientific carrier.

I feel particularly indebted to my family who have turned a blind eye to my disregard of the duty that occurred in our family over the past few years. Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of my doctoral dissertation.

Aidas Smaizys

1

Decision Models in Intellectualised Information Systems

This chapter reviews decision processes and decision models used in business regarding the evaluation of the scope of decisions that can be implemented by using currently available reasoning algorithms and knowledge in the field of intellectualised information system engineering. The present research deals with evolving decision-making processes with the special focus on decisions at the operational, tactical and strategic levels that could enable flexible adaptation of the business system to the changes in business system environment. Such business system evolution is usually initiated by changes in business strategy and further need for alignment of business processes and evolving business logic. First of all, this is related to the implementation of new motivation of decision-making and implementation of such changed business logic at operational and tactical levels aiming to support business system relocation into the new desirable future state determined by business strategy and goals by simultaneous adaptation of all related information systems across the enterprise.

1.1. Definition of Decision-Making in Business and Information Systems

There is no unique definition of a notion "decision". The term "decision" is known from the 15th century from Middle French *décider* in turn from Latin *decisionem* (nom. *decisio*), *decidere*, where *de-* means "off" and *ceadere* "cut" - to cut off from all alternatives. This is what is usually needed when you decide. Oxford Dictionary defines a decision as a *noun* meaning a conclusion, judgement or determination and as a *verb* meaning the action or process of deciding something or of resolving a question. Managerial decision-making is synonymous with the whole process of management including decisions that are made at all three levels of decision-making starting from a strategic through tactical to operational level.

A traditional information system (IS) collects, stores, analyses, and disseminates information for a specific purpose. It accepts inputs and processes data to provide information to decision makers and helps them to communicate results. Therefore, in IS the term "decision-making" is more common in the context of "decision support". The term "decision-making" as a cognitive process of reaching a decision was recorded in 1953. We define decision-making as a process of choosing among alternative courses of action for the purpose of attaining a goal or goals. Decision-making in computer and information science literature occurred for the first time in the context of decision support, which has evolved from two main areas of research: theoretical studies of organizational decision-making done at Carnegie Institute of Technology during the late 1950's and early 1960's, and the technical work on interactive computer systems, mainly carried out at the Massachusetts Institute of technology in the 1960's (Keen *et al.* 1978).

There are many approaches to decision-making and a wide range of domains where decisions are made. Decision support systems (DSS) became an area of research of its own in the middle of the 1970's, and according to Sol DSS definition has been migrated from an explicit statement of what DSS does to some ideas how the DSS objective can be accomplished and was described as a computer based system to aid decision-making. Later it started focusing on interactive computer-based systems which help decision-makers utilize data bases and models to solve ill-structured problems. The emphasis lies not so much on the decision process but rather on the support for personal computing. Before gaining in intensity during the middle and late 1980's, DSS was defined as a system "using suitable and available technology to improve effectiveness of managerial and professional activities" (Sol *et al.* 1987). In 1974, Gordon Davis, Professor at the University of Minnesota, published his influential text on Management Information Systems. He defined Management Information System

(MIS) as "an integrated, man/machine system for providing information to support the operations, management, and decision-making functions in an organization" (Davis 1974).

Expert systems have been defined "as a computer system containing organised knowledge, both factual and heuristic, that contains some specific area of human expertise and that is able to produce inferences for the user" (Chang *et al.* 1983). At the end of 1980's DSS faced a new challenge towards the design of intelligent workstations, when executive information systems (EIS), group decision support systems (GDSS), and organizational decision support systems (ODSS) evolved from the single user and model-oriented DSS. Elam has proposed to confine a notion of DSS to "the exploitation of intellectual and computer-related technologies to improve creativity in decisions that really matter" (Elam 1985).

Having started in about 1990, data warehousing and on-line analytical processing (OLAP) began broadening the realm of DSS. As the turn of the millennium approached, new Web-based analytical applications were introduced. DSS was extended by technologies for gathering, storing, analysing, and providing access to the data, query and reporting, OLAP, statistical analysis, forecasting and Data mining combined into the category of applications called Business Intelligence (BI). BI technologies were named for the first time by Howard Dresner in 1989 as an umbrella term to describe "concepts and methods to improve business decision-making by using fact-based support systems" (Power 2007) and have been widespread with newly reborn interest in Data mining based on historical expertise and heuristics provided by Artificial Intelligence (AI) technologies implemented into the information systems with fully or semi-automated decision-making focusing not only on preparation of information used for decisions made by the management staff or evaluation of possible consequences but on decisions implemented as separate automated processes leading to intelligent reaction of such an Intellectualised Information System (IIS).

A decision maker in an enterprise is expected to make decisions which have a positive effect on its future. Information systems should support business activities and adapt to the changes. At present, databases and web-based resources accessed through effective communications make information about the past rapidly available. To project the future the decision maker either has to use intuition or employ other tools, and initialize them with information obtained from an information system to such tools. An effective information system should also support forecasting the future (Wiederhold 2000). Since choices are to be made, including the case of not doing anything, such a system must also support the comparative assessment of the effects of alternate decisions. Further development of intellectualised information systems challenge changes in understanding of the decision maker as an entity of the management staff into the focus on de-

velopment of semi or fully automated components of dedicated information systems or a set of integrated systems forcing automated changes in business logic implemented into the information systems, causing self-adaptation to the changes in business environment and a new way of performing everyday tasks by using such intelligent information systems. That was foreseen by Nobel Laureate Herbert A. Simon (1985) by introducing a provocative discussion on the idea of corporation managed by machines and a new science of management decision.

According to Object Management Group (OMG), decisions are presented in many business processes and process models; however, there is no particular standard notation or model (OMG 2011b). For example, in Business Process Management Notation (BPMN) or UML Activity Diagram decisions are made in process flow forks or dedicated activities represented by a diamond. Such diamonds are usually treated as a simple point for selection of a further activity alternative. However, business logics are often more complicated and require additional informational support and post decision evaluation procedures which need particular modelling and management concerns. We also experience business logic separation from the process. This makes unclear the modelling of such decision processes with separated explicit or implicit logics. There is no Decision Model Notation (DMN) available at the moment.

It is important to note a phenomenon that a decision is a paradox of choice. In many cases the paradox that more choices may lead to a poorer decision or a failure to make a decision at all has been observed. It is sometimes theorized to be caused by analysis paralysis, real or perceived, or perhaps from rational ignorance. A number of researchers, Sheena and Lepper (2000), Schwartz (2004) have published studies on this phenomenon. In addition, we also note another phenomenon of inevitable influence on the result of the decision by the process of making decision itself: a measurement procedure influences the result of measure. This paradox is important when we want to evaluate the actual impact of the decision we make. In most cases we will never know the actual result of the alternative decision which we have not chosen at a particular moment.

To sum up, we define decision-making in the scope of IS as a process of choosing among alternative courses of action for the purpose of attaining business goals. Decision-making is supported by analytical data processing and can be executed by human or machine agents. The analysis of intellectualised IS's evolution lead us to the conclusion that modern IS should support self-adaptation to the rapidly changing business environment and suggest or implement decision alternatives based on automated knowledge discovery and evaluation of a future effect on the system. This is not a trivial task for information system engineers and should be supported by a huge amount of specific knowledge and methods which are partially available but often incompatible with each other

or have important missing parts in the scope of our interest. The comparison of the existing methods is provided in further chapters.

1.2. Decision Classification

The literature survey has identified that there is no existing, well-defined and comprehensive framework for identifying individual factors and specific decision-making strategies; therefore, no existing assessment tool could be used to classify personal preferences for decision-making. The only classification system to index decision-making styles by decision task type and individual factors was found in the paper of Bruyn *et al.* (2006). Bruyn has identified six types of decision-making styles: analytical, naturalistic, avoidant, dependent, spontaneous, and heuristic. He has also determined three high level classes of factors that appear to influence a decision-making style: person-based, social/group and situation/context factors. It considers factors relating to the task (e.g. time pressure) as well as the decision maker (e.g. age), etc. This shows that such common factors as stress, emotional condition, time pressure and personal properties of a decision maker including age are very important in business decision-making. Some decisions should be performed with influence of such human nature; however, a lot of decisions could be performed better in information systems by eliminating such weaknesses.

According to Barbara von Hale and Goldberg (2010), decisions in business are usually grouped by three management levels according to the time, details, questions, and decisions: strategic, tactical and operational.

- **Strategic decisions** (What?) deal with the big picture of business. The focus of strategic decisions is typically external to the business and usually future oriented. Strategic decision-making creates the forward thrust in the business and development of long-term vision.
- **Tactical decisions** (How?) involve the establishment of key initiatives to achieve the overall strategy. Tactical decisions help to define key tactics that will be used to achieve an overall strategy. One of possibilities for tactical decision automation can be selection or arrangement of business rules used in operational decision-making. In the business tactical decision-making is the domain of mission statements.
- **Operational decisions** (How are the resources utilised?) determine how activities actually get done. They are the "grass roots" decisions about who is going to do what and when. Operational decisions are often made in real time and are the results that require making quick adjustments or change to achieve the desired outcome, involve processes and procedures.

Business processes are usually guided by making decisions that are influenced by a set of environmental elements such as political, legal, physical, economical factors and other systems. According to the Deming Cycle (Deming 1992), there are four main necessary steps to align an organization with its processes on the business system level. We propose a structure of decisions along the four phases: plan, do, check, and act. Every step in such a cycle is related to the specific class of decision processes grouped by the phase:

- **Planning decisions** (Plan). The planning process is made up of parts starting from the present position ending at the goal covering determination of the present position, objective, action plan, obstacles, and timeline. Strategic planning usually done by senior management includes planning decisions made on the objectives and committing resources such as money, time, and people. Tactical planning is dedicated for implementation of the strategic plan, combining available resources, considering obstacles, reviewing alternatives and developing a general timetable. Operational planning is a much more detailed level of a strategic and tactical plan. In business it means answering who, what, when, where, how questions, when organisational processes are identified, modelled, and optimized.
- **Operational control decisions** (Do). The operational control decisions are created and implemented into the technical (physical) or social (organisational) system of an enterprise. Such decisions usually are made following business rules and guidelines provided by the upper management staff. Operational control decision processes are created by inevitable employment of creativity used for development or modification of related business processes which can be partly or fully automated and implemented into the IS together with the business process measurement and monitoring processes during the system design phase. At this phase decisions that are left to human agents are implemented into the IS's that support single process steps, and process participants are trained in the organizational rules and regulations as well as the use of the supporting infrastructure. Automated decision-making processes are implemented using decision models stored into the special repositories and executed by dedicated software components of the enterprise IS. The metrics about process performance are collected during the execution of the new processes.
- **Decisions on information processing** (Check). Assessment processes are based on the data collected in the previous phase. Evaluation phase decisions include decisions on selection of transactional data, measurements, analysis dimensions, and methods. Such decision processes include development and implementation of analysis and reporting tasks

to provide results about effectiveness of the new organisational processes implemented in the previous phase to the decision makers on the next reengineering phase.

- **Decisions on process improvement and optimisation** (Act). Reengineering phase decisions are based on the information provided from the previous phase and intuition or objective results of external analysis of the business system, when the attainment of strategic and operative goals is analysed. Such decisions are planned and implemented by execution of new cycle iteration and are usually related to the changes of policy on a strategic level, changes of tactics by selection of a different set of guidelines or modification of the existing requirements. A change at an operational level leads to the changes in business logic and business rules. Decisions at this level include forecasting, impact analysis, and risk assessment.

The analysis of decision modelling methods and decision-making methods provided in the next chapter demonstrates that decisions according to the decision-making process, its behaviour and information used for decision-making can be grouped into:

- **Deterministic – structured:** this group of decisions represents decisions made according to the structured logic according to the determined process, when it is possible to describe all the possible decisions and states of the decisioning system and where the structure of information objects is known. Such decisions are usually made according to the rules specified using horn logic or decision tables;
- **Deterministic – unstructured:** this is a group of decisions where decision-making process can be determined but the structure of rules and facts is unknown and a set of possible solutions is huge but limited. Such decisions can be usually automated by using rule engines;
- **Stochastic:** that is the most complicated group of decisions made under uncertainty. Such decisions are made in behaviour where it is impossible to determine a structure of decision rules or specify a full set of possible solutions. Such decisions are usually made by using fuzzy logic or neural networks and flow engines.

Ronald Ross also defines deterministic, intuitive, and ad hoc decisions which can be repetitive or infrequent and provides several patterns of decisions by inclusion of classification, evaluation, selection, approval, assessment, assignment, allocation, and diagnosis or prediction tasks (Ross 2010).

In the current chapter we have overviewed a classification of decisions and decision processes. Decisions at strategic and tactical levels are usually based on the information provided by IS's used for decision support and deliver the same value as their context has been implemented into the organisational business

processes and decision-making actions directly implemented into the IS software logic. Advanced modern implementations based on execution of decision-making models with employment of specific software components used for model execution are discussed in the next chapter.

1.3. Decision-making Models

According to Linehan *et al.* (2011), Decision Model and Notation (DMN) do not exist yet. The new publications (von Hale and Goldberg 2010; Ross 2010; Taylor *et al.* 2007) in this field highlight the need for new decision modelling methods and techniques, which are on the focus of our present study. The first framework for decision-making models was introduced in 2010 by Barbara von Hale and Goldberg (2010). According to the authors, business decisions are supported by three legs of business decision management: business motivation, business logic, and business metrics:

- **Business motivation**: This is a general business plan, and the specific business objective/s within the business plan that the business decision is meant to implement.
- **Business metrics**: These are the measurements and time periods that are set by the business objectives, and that must be achieved by the business. These metrics are arrived in the business planning phase, and they may be supported by predictive modelling techniques.
- **Business logic**: This is the explicit logic underlying the business decision that is implemented to best deliver the business metrics set by the business objectives. The role of decision model is to maintain a stable, normalized and complete representation of that logic.

Business motivation and the role of business logic for the first time was introduced by Zachman and Sowa (1992). The latest important addition in this field is Business Motivation Model (BMM) introduced by Business Rules Group (BRG) and adapted as a standard by OMG (2008a), and Semantics of Business Vocabulary and Business Rules (SBVR) adapted as a standard by OMG (2008b). Other authors (e.g. Hammer) also agree on the importance of appropriate measurements in order to verify the effectiveness of business processes: "Whatever measures are employed, they must reflect the process as a whole and must be communicated to and used by everyone working on the process. Measures are an enormously important tool for shaping attitudes and behaviours; they play a central role in converting unruly groups into disciplined teams" (Hammer 1996). In addition we define such important additional legs of business decisions as business knowledge and intuition:

- **Knowledge.** Those are the implicit knowledge models which provide additional information about business processes or predict future trends of business metrics allowing a prior reaction and more precise decision-making.
- **Intuition.** This is a compensation mechanism which leads to the conclusion and allows decision-making in uncertain situations, when a part of information and knowledge needed is missing. Rarely intuition based decisions are affected by emotions.

Intuition and emotions belong to the social sphere; therefore, we have left it behind the scope of our research. However, it is important to note that modern science is trying to deal with simulation of such situations by implementing genetic algorithms which provide some creativity to the technogenic systems. Decision-making models and methods based on calculations or logical inference can be grouped into the following groups:

- **Decisions under Structured Logic** include Horn logic based reasoning on structured decision rules involving execution of production rules, decision tables, etc. This approach is widely used in IS engineering and implemented into the software applications.
- **Decision-making under Risk** includes probabilistic methods for Expected Payoff, Expected Opportunity Loss and Most Probable State of Nature Determination, Expected Value of Perfect Information with use of decision trees (Bayesian Approach). Macesker *et al.* (2009) also define a Risk-based Decision-making (RBDM) model composed of risk assessment, risk management, and impact assessment processes.
- **Decision-making under Uncertainty** Max-min criteria (pessimist) evaluation, Max-max (optimist) criteria evaluation and Min-max with the combined use of a loss table and calculation of weighted expected values. It also includes Minimising regret and equal likely strategy (LaPlace). Uncertainty in measurement, in models and in interpretation can be separated.

Other authors also group decision-making models into: Rational decision-making models, Intuitive decision-making models, Recognition primed decision-making models and Ultimate decision-making models.

In a traditional way structured business logic expressed in structural BR can be represented e.g. in OCL (Object Constraint Language) and used for data model design, constraining structure of entities in data model as well as implemented as integrity constraints in the active database management system (ADBMS). Meanwhile, the operational rules are usually expressed as the requirements in the traditional IS engineering process and implemented further into the process models of IS and its software system code as various case scenarios or methods of the objects for implementation and execution of business

logic in the application layer of the software. E.g. Valatkaite proposes a part of operational rules to be implemented directly as triggers or stored procedures for execution of business logic in a data layer of BIS applications (Valatkaite and Vasilecas 2002). Further in this chapter we will discuss business logic and decision-making implementation and support by enterprise information systems.

In this thesis we treat decisions as a business process composed of decision sub-processes, where the input of such a sub-process is the information needed to support decision considerations expressed in declarative business logic terms. Outputs of the decision process are the aggregated conclusion or outcomes enforced from particular activities in the IS software applications or events raised. In effect, decisions operate as programming functions, methods of business objects or SOA services. A separate decision process should be equipped with a sub-process feeding decision process with information represented in facts and business logic represented in terms of production rules.

From business and technological perspective, to make a decision is to assign designated values to one or more decision variables where the assignment is not deterministic. The decision problem according to OMG (2011b) is to mark one or more decision variables for instantiation, where decision-support system proposes "decisions" that are compliant with some body of constraints.

We suppose that this is not necessary in all the cases. There is a huge amount of simple decisions that are made in one step by selecting predefined alternatives. However, we also discuss ability to propose "solutions" in not deterministic way, but we think that an additional decision process on the top is needed in this case. We illustrate such scenario in Chapter 3.1. where we propose to separate decision support, decision-making and decision evaluation models and assign them with separate functions by providing wide range informational and analytical support, evaluation of decision impact on the business process and feed decision process with a set of possible decision alternatives. Separation of support and decision processes allows implementation of more sophisticated decisions based not only on reasoning on rules representing decision logic and facts but also inclusion of decisions based on historical and implicit knowledge, involvement of predictive models, optimization models, risk based analysis models and other models based on AI involvement techniques such as neural networks, regression, etc. into the decision model proposed in Chapter 3.2.

1.4. Business Logic in Enterprise Information Systems

Business involves activity targeted at particular objectives. Foremost among these objectives is the essential purpose of maximising long-term owner value by selling goods and services (Sternberg 1998). The emphasis on value over profit is important, while the profit can be subject to manipulation. In addition, an unrealistic focus on profits over value can push business management to make decisions laden with risk in an effort to reap short-term profit at the expense of long-term business health (Riordan 1997). The thesis show how complicated the nature of business is and why it is necessary to create and manage a complex structure of various decisions to stipulate a business system and provide all necessary infrastructure needed to make adjustments of the business system processes and to achieve so-wanted balance. Productivity is the key to being successful. At present, successful business operates in conjunction with information and communication systems. However, business models today include exploding knowledge amounts, and the main directions for future growth of long-term business value is mainly related to the employment ability of knowledge-oriented information systems including automated knowledge discovery, decision support and knowledge based decision-making systems.

An enterprise is a heterogeneous system in the category of social systems. Being a social system means that the elements are social individuals, i.e. subjects (Dietz 2006). It could be right in the ages before massive expansion of intellectualised information systems that are used nowadays not only for information processing and communication, but are starting to replace humans in "typically" human operations like reasoning and automated decisions. Such systems are not the social individuals, and the enterprises are getting more and more involved in changes dealing with technogenic systems. Such technogenic systems like automated manufacturing equipment and agile intellectualised information systems are getting more and more important comparing with the former social individuals.

Previously, the main value and productivity in the enterprise was delivered by effective business processes and knowledge of human recourses used for that process. It made the change of qualified personnel complicated, while the technogenic and informational resources were treated just as tools that can be easily replaced at any time. Currently, the situation has changed. It is usually easier to replace personnel than make changes in the provided information systems and the embedded business processes.

In particular, business systems are functioning according to a BR approved in a specific business domain. Due to dynamic nature of business environment and accordingly to BR, they are changing frequently reacting to internal and ex-

ternal influences such as changes in law, new competition etc. (von Hale 2001). BRs themselves are relatively new addition in the field of enterprise Information systems development (Bocij *et al.* 2005). It has been accepted since 1988 that BRs are an important element of all types of IS (Van Assche *et al.* 1988; Zachman and Sowa 1988; Loucopoulos and Katsouli 1992) and BIS as a kind of IS therein. SBVR defines a BR as "a rule that is under business jurisdiction" (OMG 2008b). BRs from the database engineering perspective are somewhat similar to integrity constraints (Date 2000) or various programming language constructions used for description of structured logics for making decisions in software systems.

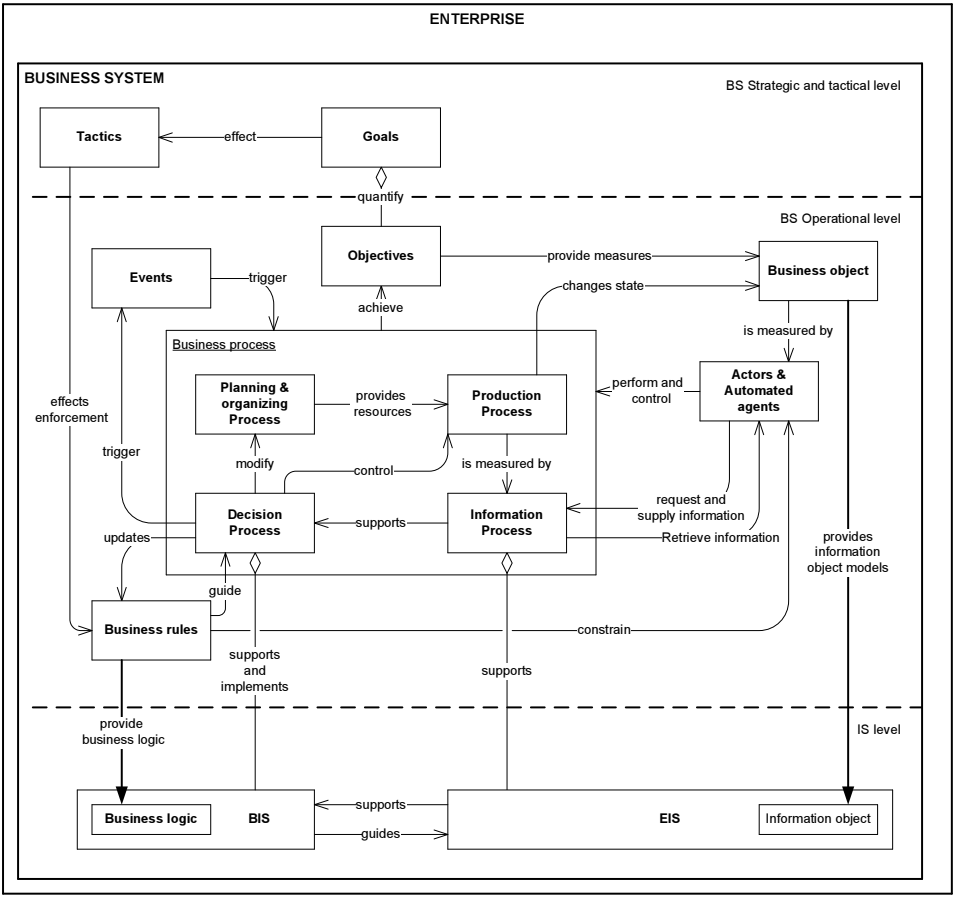


Fig. 1.1. Relationships between an enterprise and its ISs. Extended EIS oriented schema initially introduced by Montilva *et al.* (2004)

Some authors discuss similarity of BRs to requirements (BRG 2000; Hay 2003) or state that BRs can be used to represent both user requirements and conditions to which the system should conform (Valatkaite and Vasilecas 2005). BRs are the main source for business logic in the Enterprise Information Systems (EIS).

EIS is usually defined as an open system which is a part of a wider system, called application domain. The main objective of an EIS is to provide information services to its application domain Montilva *et al.* (2004). Montilva also discusses on what the application domain of the EIS is according to a business system approach and defines the domain of an EIS as a business system, i.e. a human activity system designed to achieve pre-defined goals with the support of technologies. Where a business system is a part of a major system: the enterprise. An enterprise is, therefore, seen as a set of business systems. Each business system has associated one or more EISs.

Gudas *et al.* (2005) propose Enterprise modelling framework for Knowledge-based IS engineering. The authors do not separate planning, production, informational, and decision-making processes and treat them as an instance of a general business process or a separate process activity. However, they propose an interesting formalized model of the Function-Process interaction, where the Function covers a Control System and Feedback Loop used to change a Process state attributes. Further in this study we use similar concepts of Control Theory; however, we focus on decision-making automation and implementation into the automated IS process developed through a specially designed engineering cycle where decision-making and data processing are not instances of informational process, but rather a separate sub-processes of business process, developed separately and implemented into the dedicated infrastructural components of IS (e.g. separate EIS and BIS). From our point of view, we consider those integrated at SS level to provide machine dedicated control on the automated business system process. On the other hand, most EISs in enterprises have already become huge and complex structures which are difficult to maintain, change or develop new functionality needed to provide artificial intelligence features necessary for further decision automation. We expect further evolution of EIS into BIS, which is usually defined as a set of business practices, procedures and processes that are implemented into the software systems. A set of such applications can be combined into the Management Information Systems (MIS) used for a planned system of collecting, processing, storing, and disseminating data in the form of information needed to carry out the functions of management, while EIS system incorporates a set of applications that are not necessarily focused on decision support.

According to BIS definition, it is required to extend a definition of the role of a business system and its relationships with its information systems intro-

duced by Montilva *et al.* (2004). Moreover, by adding BIS within IS context of EIS we find it necessary to introduce additional sub-processes of the business process represented in Fig. 1.1. The modified schema presents an extended business system and EIS relationship schema, which was initially proposed by Montilva *et al.* (2004). Our main addition to this schema was motivated by our orientation towards business rule based decision process automation. To demonstrate the role of production (tangible and service), informational, and control business processes, we have added missing concepts of a business system and their relations to the intellectualised BIS. Here we see the achievement of IS intellectualisation by moving automated business decisions from a business level into the IS level as further EIS evolution with involvement of the intellectualised business information system (BIS), which are on the main focus of the present research.

One of the main widely distributed EISs are ERP systems. According to Chung (2003), ERP system extends a manufacturing resource planning (MRP II) system, which refers to the planning of manufacturing schedules taking into account the time-phased material availabilities, ordering, and the capacities of production facilities, to include engineering, finance, human resources, and other activities in the enterprise. ERP system intends to integrate and automate various back-office functions of management and control activities such as finance, human resources, marketing, and operations. Such integration of former separate software systems into one modular system means great effort for developers, because of the needs for business process standardisation and the use of same object and data model across the enterprise. ERP system is a packaged software solution that addresses the enterprise needs of an organization by tightly integrating various functions of an organization using its process view. The main goal of ERP systems is to integrate data and processes from all areas of an organization and unify them for easy access and work flow. This is the main reason which drives fast ERP systems expansion into former areas of business systems (BS) not dealing with ERP systems and getting functions of BIS. From the historical perspective it is important to highlight expansion of evolving ERP systems to support business processes from operational through tactical to strategic levels. Such expansion enables implementation of a more complex decision and logical derivation processes and requires the use of new technologies and methods to support more intelligent behaviour of ERP systems. ERP systems serve as a core component which expands through different business levels (operational, tactical and strategic), integrates and includes several formerly separate information systems (IS). This leads ERP systems to become a synonymous to IS (though ERP systems like BIS, MIS and executive information systems (EIS) are specialization of a general IS), and a border between ERP systems and other kind of ISs is going to be hard to distinguish.

All kinds of known BRs have found their place in nowadays IS, especially due to the emerged new multidimensional data analysis tools and software based on warehousing techniques and On-Line Analytical Processing (OLAP) technologies (Mahajan 1997). Warehouse data based business analysis tools are widely used in enterprises for collecting and analysing large data sets by using several databases (DB) of such systems as ERP, Data Mining, Expert and Decision Support, Customer Relationship Management (CRM), Production and other systems. The addition of new techniques increases the potential of technical infrastructure to support business decisions as displayed in Fig. 1.2.

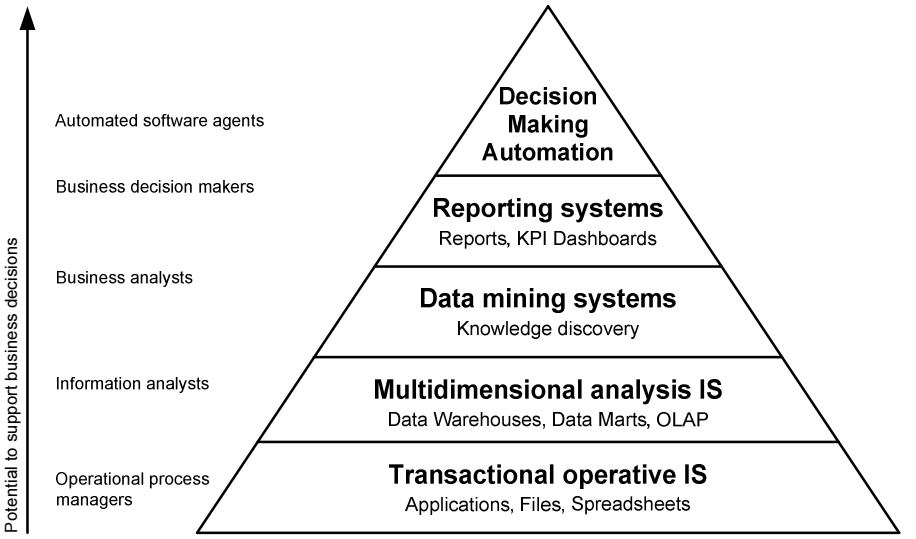


Fig. 1.2. Increasing the potential of technical infrastructure to support business decision process

According to a traditional IS engineering methodology, changes in BRs should be propagated through all the reengineering lifecycle stages starting from the problem domain analysis, requirement specification and design, where BRs are involved in requirements and implemented into the BIS software code as new or modified executable instructions or a kind of production rules used for automated decisions later on. Such changes challenge further inconsistency of the existing BIS and already automated decisions or decision support in MIS and ERP systems with new business situation and have to be adapted to the new business needs and the changed BRs i.e. a subset of the enormous number of BRs enforced by BIS. From a business point of view, nontrivial BRs may be context dependent in space (e.g. different law in different countries) and time

(e.g. law may be changed). Therefore, the rules have to be maintained during the life-cycles of rule-based applications (Herbst 1994).

Increasing maintenance, development costs, security issues, globalisation and business distribution across the different countries with different business rules lead to the opposite process of BIS systems disintegration, when separate business processes for individual functional areas are developed in a fragmented manner. To stop such disintegration, development of new more sophisticated and intelligent techniques and methods of BIS systems development is needed. That might also be provided by involvement of business rules (BRs) approach combined with an agile systems development process based on business process analysis.

Recently, some relevant enhancements of the existing business information systems engineering methods have been introduced, although there are still open issues of how business rules may be used and improve qualitative and quantitative attributes of such kind of information systems. Information systems are getting more and more integrated into the social system of the enterprise. It is not sufficient to change structured business rules and provide them to the personnel expecting business process changes and easy business system adaptation to the changes of the external enterprise environment. A huge amount of informational business processes are fully or partially automated, and all business rules guiding such processes are already implemented in several applications or information systems of the enterprise. This makes rapid changes impossible; and the main barrier for that is inflexible information systems where all the business rules are differently interpreted several times by people responsible for separate roles in the traditional cycle of IS development. Such engineering approach in a traditional way usually hides business rules and disables possibility to make changes of business rules just on demand because it is necessary to process the entire system development cycle again. Therefore, new techniques that allow implementation of automated decision-making based on declarative logic and support for some degree of uncertainty are needed.

1.5. Related Work Analysis

In the following chapters the related methods are analysed, and a comparison of the methods according to the defined criteria is made. The chapter is finished with the results of the comparison.

1.5.1. Barbara von Hale *et al.* Method

Barbara von Hale (2001) proposed to separate the process of business rule discovery into two parts: business rule representation and relation with initial requirement specifications and specification of a business rule itself. Initial requirements are discovered by analysing processes and events in business process model or use cases. Later, in 2010, Barbara von Halle and Larry Goldberg (2010) introduced a Decision model and defined 15 principles of the decision model development. The authors emphasize separation of a business decision from a business process and offer significant improvement by removing the declarative business decision from the procedural process flow. In such a way a new task combines the whole previously sequential set of tasks into one task, denoted as a decision task, behind which a business decision executes in a declarative fusion as displayed in Fig. 1.3.

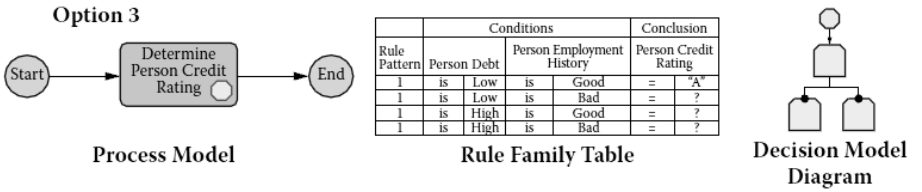


Fig. 1.3. A theoretical model of a decision process based on structured information (von Halle and Goldberg 2010)

Within the process flow, the decision task looks like any other task but contains a decision shape within the task box. It also includes the Decision Model (DM) diagram, which puts the Decision Rule Family in context with its related Rule Families. Neither the rule family table nor the decision model diagram is embedded in the process flow. They are separate deliverables, anchored to the process flow by the decision shape. For the process model the authors use BPMN notation and rules specified as decision tables. According to the proposed method, a decision model represents information structure (a fact model) used for decision-making similar to the decision tree where every node is represented by a corresponding rule table.

The proposed method allows a simpler business process model and highlights all possible combinations of conditions, permitting changes in the Decision Model without changing the business process model. However, the method is suitable only for conceptual modelling of a decision and has no guidelines for further implementation into the software system. Such method is good for the decisions made according to the structured logic; however, it is not suitable for modelling decisions under uncertainty, especially when decisions should be

made taking into account the consequences of such decision implementation. In this case separation of business logic and representation using structured rules is not sufficient, and separate development of additional decision support processes is necessary.

1.5.2. OMG (BRG) Approach

The first discussion on the business rule approach came from a non-commercial peer group of IT professionals working on the business rule approach since 1989. The group called Business Rules Group (BRG) started by introducing discussion in 2000 (BRG 2000) and defining the fundamental principles of the Business Rules Approach in Business Rules Manifesto (BRG 2003). Later, in 2005, they proposed Business Motivation Model (BRG 2005) which was adopted as a standard by OMG (2008a).

The last addition in the field of decision modelling is a discussion provoked by appearance of the book of von Halle and Goldberg (2010) with a discussion on development and use of a decision model and recent releasing a Request for Proposals (RFP) for a Decision Model and Notation (DMN) specification (OMG 2011b). Several authors started discussions with a focus on decision modelling (Ross 2010; Linehan *et al.* 2011).

The authors of BRG focus on the use of decision tables for a structured business logic specification and SBVR (OMG 2008b) as a global semantic model of rules in the enterprise. Some other authors, e.g. Linehan *et al.* (2011), discuss the place of DMN in the context of other existing specifications of BRG the SBVR and BPMN (OMG 2011a). Fig. 1.4 illustrates the proposed relationships among DMN, BPMN, and SBVR, where SBVR formally models the vocabulary and rules found in business sources such as corporate policies, regulations, and contracts. Capturing these as global rules enables businesses to analyse how the rules are applied throughout their operations. Later the SBVR global rules are refined and implemented in DMN decisions that may be applied as activities within BPMN processes, or in other software. The decisions depend upon the SBVR vocabulary concepts. The BPMN processes pass runtime instances of these concepts into the decisions as considerations and receive instances as conclusions. The decisions are to be executed in business rule engines.

Linehan promises that "DMN will define a computation-independent but decision-specific model that bridges the gap between the computation-independent and decision-neutral business vocabulary and business rules modelling framework offered by SBVR, and the process-specific decision model level provided by BPMN" (Linehan *et al.* 2011). In this scope we agree that the discussed DMN can be a missing key component in the field of business modelling

bridging the fields of business vocabulary, business rules, business decisions, and business process modelling into one complete business model.

To sum up, the proposed DMN could provide an integrated model for decision management the same way BPMN does for business processes and offer an integrated process-oriented, information-oriented, and rule-oriented view of a business model.

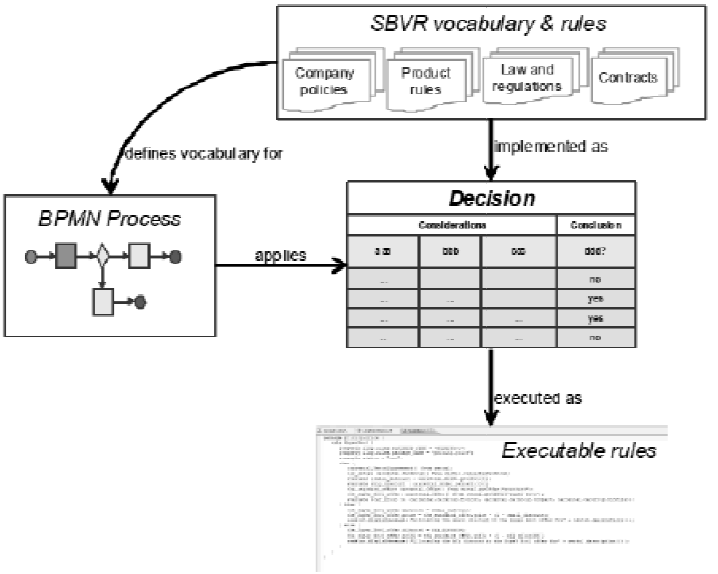


Fig. 1.4. OMG (BRG) approach to decision model (Linehan *et al.* 2011)

OMG adopts specifications by explicit vote on technology-by-technology basis (OMG 2011a). The specifications selected should each satisfy the architectural vision of Model Driven Architecture (MDA), another important OMG standard in a model transformation driven approach of software engineering, with involvement of QVT (Query/View/Transformation) language used for declarative and imperative specification of model transformations in MDA context (Perez-Castillo *et al.* 2010). MDA provides a set of guidelines for structuring specifications expressed as models and mappings between those models. The MDA initiative and the standards that support it allow the same model specifying a business system or application functionality and behaviour to be realized on multiple platforms. MDA enables different applications to be integrated by explicitly relating their models; this facilitates integration and interoperability and should support system evolution (deployment choices) as platform tech-

nologies change. The three primary goals of MDA are portability, interoperability and reusability.

Following the ideas introduced by BRG, we mainly focus on the use of decision tables, widely discussed by Ronald Ross (2010). Decision tables are cooperatively simple for understanding by business people and software developers for specification and communication of explicit business logic. However, here we observe a huge limitation by such approach. Our experiments on decision tables show that this makes implementation of changes in the structure of business logic at run-time very difficult or almost impossible.

However, the proposed approach and specifications already accepted by OMG do not limit the use of declarative rules composed according to the business vocabulary. Although current methods do not include decision support and implementation of decisions based on historical and implicit knowledge and involvement of predictive models, optimization model, and risk based analysis models or other models based on AI involvement techniques such as neural networks, regression, etc., the main drawback of the method is the leaking focus on further implementation of decisions into the IS software.

It may be concluded that OMG in the context of MDA provides architectural vision for model transformation driven IS software development, and the new request for DMN development in the context of the existing specifications such as BPMN and SBVR is the next important proposition made which will change the understanding of modern IS development and will add the missing link between the model of business logic and decisions in the business process.

1.5.3. Avdejenkov and Vasilecas method

Avdejenkov and Vasilecas (2004) propose the use of business rules for business logic separation and centralised implementation. The authors propose specification of initial requirements discovered by analysing processes and events in a business process model or use cases. After such definition of relations between events and actions it is proposed to specify a decision-making process according to the business logic captured and represented as BRs. This is a typical scenario of business logic representation using ECA (event-condition-action) rules and their further implementation with later transformation into the triggers of active DBMS. Business rules according to the proposed method are initially specified in decision tables and stored in the centralised business rule repository. The repository is integrated with an ERP system to extend its functionality by using automated transformations of the business rules into the triggers of active DBMS. Such approach gives some flexibility at design time. However, there are serious limitations and disadvantages at the run-time. The method allows making changes to the logic without direct changes of the system code in the appli-

cation. Moreover, triggers usually cannot be created and implemented into the production system or even changed in a centralized way without complicated verification of system behaviour due to unpredictable interactions at the run-time.

1.5.4. Asuncion *et al.* Method

Asuncion *et al.* (2010) elaborate flexible service integration trough separation of business rules in the context of service mediation to provide an approach for increase of flexibility in integration solutions. The main idea of the proposition is based on combination of goal-based, model-driven and service-oriented approaches. The authors also support an idea that flexibility can be achieved by separation of dynamic parts of the business process from more stable parts.

The authors follow the recommendation of Lapouchian (2005) to treat system objectives or goals as first-class citizens. The proposed solution uses Goal-Oriented Requirements Engineering (GORE) to specify requirements as goal models (Van Lamsweerde 2009) in contrast with early practices where the requirements focus only on the process and data without capturing the rationale of the software systems, making it difficult to relate the requirements to the higher problem domain.

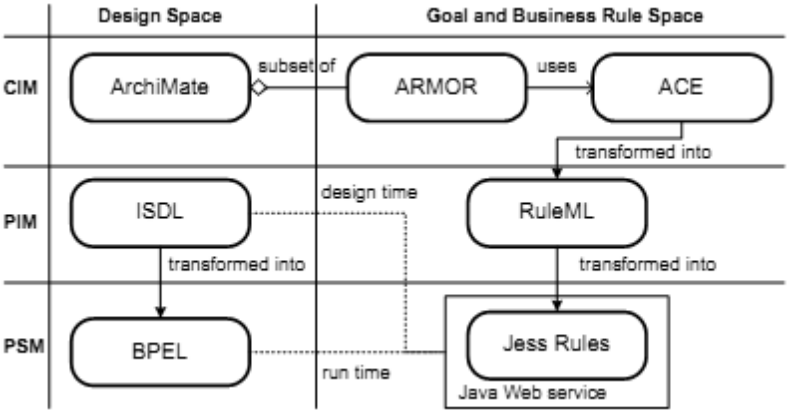


Fig. 1.5. Business rule and process design model integration (Asuncion *et al.* 2010)

The next important aspect of the method related to our research is the use of business rules for specification of business process logic. The authors propose the use of a goal model in ARMOR refinement into business rules and later manual transformation into the ACE at the CIM level and afterwards into the

RuleML language at the PIM level; when the rules in RuleML are later transformed into the Jess Rules using Extensible Stylesheet Language Transformation (XSLT) for later execution at the integrated rule server as displayed in Fig. 1.5.

The proposed method conforms to the criteria of support for functional, organizational, behavioural, informational, and operational perspectives with respect to the change of properties ensuring incremental, permanent, deferred, and planned change.

The method proposed by the authors share similar view on the problem of ensuring business process flexibility and minimisation of the gap in information system software and business alignment. However, the authors concentrate on a different topic of dynamic market demands: on achieving enterprise collaboration facilitated by Enterprise Application integration with a focus on integration of business processes of separate enterprises with different business logic when we focus on the decision process itself wherever it is implemented in the enterprise business information systems.

1.5.5. IBM (ILOG) Approach

In previous chapters we have overviewed several methodological approaches proposed by different authors and organisations. Here we start a discussion focusing on technological support for decision automation provided by IBM (ILOG) and Microsoft.

There is nothing strange that IBM tools are developed with theoretical orientation to the work of BRG and OMG because several leading authors in BRG work in IBM as well. IBM is one of the leaders in implementing flexible business process oriented models into the software systems of an enterprise IS. However, the main missing part in the set of tools provided by IBM was a good and effective business rule engine. This problem was solved when in 2009 IBM completed acquisition of ILOG, of the owner of the leading Business Rule Management System (BRMS) and Business Rule Engine (BRE) JRules based on Forward-Chaining (Data-Driven Inferencing). This missing part filled a gap and provided new possibilities to increase agility of their decision-making by allowing a business process to adopt dynamically.

The main focus of the approach is to provide tools and infrastructure components in IS for rule specification, management and run-time rule execution. Integration of software products is separated into several lines of products: for design-time rule base integration into the Integrated Development Environments (IDE) such as Eclipse or Microsoft Visual Studio for .NET; and run-time inference engine integration with J2SE, J2EE and SOA components (ILOG 2006b).

According to this approach, BRMS facilitate communication, deployment and execution of business policy within an IT infrastructure. Another interesting point of approach is the introduction of a special software component for the run-time rule management and authoring by business people, which allows some degree of implemented business logic transparency at a business level (Duan *et al.* 2010). The technology provides support for business rules using ILOG Business Action Language (ILOG BAL), which is close to the natural language and can be used by non-technical users to express rules in business terms. Business rules expressed in ILOG BAL can be automatically translated into production rules for execution in the rule engine or into the BNF-like format using XML for further translation into the executable rules using XSLT or Java (ILOG 2006a).

BRMS integration into the IBM products dedicated for model based business process execution (Workflow engines) allows additional flexibility by substitution of static and complicated sequences of actions representing decision logic of the business process. Such approach covers a workflow side of BPM tools. Workflow here means a flow of data and activities through the organization; and it supplies support for integration of business logic into the entire automated service where business logic is expressed using languages natural to the business users. Such approach is similar to the approach proposed by OMG (BRG); however, it focuses on a technological side and IS development instead of business modelling as it is in the case of OMG (BRG).

1.5.6. Microsoft Approach

This chapter continues the review of technological support for decision logic separation and decision-making process automation provided by Microsoft. Here we summarise technologies and tools invented for intellectualised IS software development.

The first important invention of Microsoft in this field was Data Mining Extensions to SQL (DMX), which was first introduced in the OLE DB for Data Mining specification authored by Microsoft in conjunction with other vendors in 1999. The goal of this specification was to create a vendor-neutral, programmable interface that leverages concepts already known to the people most able to take advantage of such interfaces – the programmers (MacLennan *et al.* 2009).

Although data mining technologies have existed since the 1960s in various forms, the field is still relatively immature. There were no standard concepts of mining models, training, or predictions. For many people, data mining is just a set of algorithms. The Microsoft introduced data mining tools for decision model design, testing and implementation into the dedicated services of Microsoft Analysis service, which is a part of Microsoft SQL Server. The Analysis services also provide tools for design of multidimensional OLAP cubes and data

consolidation. Here we use another important invention, the language for Multi-dimensional Expressions (MDX).

Until introduction of Windows Workflow Foundation (WWF) Rules Engine there were only two ways of business rule implementation into the IS software according to the traditional way of IS engineering i.e. implementation into the code of DBMS or software code; and some more flexible way based on the use of implicit logic discovered by components functioning according to the AI algorithm based on designed decision models implemented using data mining components.

The introduction of BizTalk server added support for the system workflow implemented by using SOA components for communication between systems servicing automated business activities. The later introduction of WWF provided a foundation for rule engine based decision automation and support for human workflows implemented directly into the application code inside of the software system providing a similar approach to the one proposed by ILOG but in a complicated technology based way without sufficient business orientation and without a focus on decisions in the business process.

The analysis demonstrates strong Microsoft positions on support for historical business information analysis, predictive analysis and mining of implicit knowledge. The new addition in the field of rule based reasoning implementation is built by the introduction of Microsoft WFF and former workflow engine focussed on business communications Microsoft BizTalk. However, Microsoft focuses only on a technological side of the problem and still lacks a separate business rule, and business process engines even rule repository accessed and managed by business people in their language and business terms. We suppose the development of new business oriented methods in this field is especially welcome.

1.5.7. Comparison of the Methods

The present investigation in this chapter is aimed at the comparison of the related business rule-oriented methods and approaches for decision automation in flexible business processes, discussed in the previous chapters. The applied comparison technique is based on the criteria defined at the beginning of this chapter. The results of the comparison for decision automation completeness and flexibility are presented in Table 1.2; and the comparison according to the support of the proposed decision model structure and principles of intellectualised BIS development is in Table 1.1.

The main goal of business logic separation from the business process and isolation of decision support and decision-making processes is business process flexibility. Asuncion *et al.* (2010) define a flexible business process as a process

if one can change only those parts that are affected by the change without affecting those that are not; i.e., the entire business process does not have to be completely replaced as a result of the change.

To evaluate the method from the perspective of business process implementation flexibility, we have added the criteria based on flexibility type spectrum proposed by Schonenberg *et al.* (2008) and selected the criteria according to the taxonomy of business process flexibility proposed by Regev and Wegmann (2006). Here we define functional, organisational, behavioural, informational, and operational perspectives of design time and run-time flexibility.

Table 1.1. Comparison of the rule-based methods for decision automation support

Methods (Approaches) / **Criteria**	**Von Hale *at. al***	**OMG (BRG)**	**Avdejenkov *et al.***	**Asuncion *et al.***	**IBM (ILOG)**	**Microsoft WFF & BI**	**Smaizys *et al.***
Supported decisions							
Deterministic-structured	+	+	+	+	+	+	+
Deterministic-unstructured	-	-	-	-	+	-	+
Stochastic	-	-	-	-	-	+	+
Involvement of historical knowledge	-	-	-	-	-	+	+
Involvement of implicit knowledge	-	-	-	-	-	+	+
Support for the principles of intellectualised BIS development							
Separation of business logic	+	+	+	+	+	-	+
Centralisation of business logic	+	-	+	-	+	-	+
Integration of decision models	+/-	-	-	-	+	+/-	+
Substitution of decision activities by rule-based process	+	-	-	+	+	+	+
Implicit knowledge based intellectualisation	-	-	-	-	-	+	+
Methodological support for decision model implementation							
Engineering methods	+	+	-	+	+	-	+
Formal syntax	+	+	+	+	+	+	+
Formal semantics	-	+/-	-	+/-	-	-	+/-

Table 1.2. Comparison of the rule-based methods for decision automation completeness and flexibility

Methods (Approaches) / **Criteria**	**Von Hale *at. al***	**OMG & BRG**	**Avdejenkov *et al.***	**Asuncion *et al.***	**IBM (ILOG)**	**Microsoft WFF**	**Smaizys *et al.***
Focus	Decision modelling	Business modelling	ECA rule implementation	SOA development	IS infrastructure	IS infrastructure	Decision automation
Approach of engineering							
Goal-based	+	+	-	+	-	+	+
Model-driven	+	+	+	+	+	+	+
Transformation-based	-	+	+	+	-	-	+
Service-oriented	-	-	-	+	+	+	+
Integration	-	-	-	+	-	+	+
Flexibility according to design-time/run-time support for subject of change							
Functional perspective	+/-	+/-	+/-	-/-	+/-	+/-	+/-
Organization perspective	+/-	+/-	-/-	-/-	+/-	+/-	+/-
Behavioural perspective	+/-	+/-	+/-	+/+	+/+	+/+	+/+
Informational perspective	+/-	+/-	+/-	+/-	+/-	+/+	+/+
Operational perspective	+/+	+/-	+/-	+/-	-/-	-/-	+/+
Completeness of decision model							
Decision support model	-	+	-	-	-	+	+
Decision process model	+	-	-	-	+	+	+
Business logic specification	+	+	+	+	+	-	+
Environmental impact model	-	-	-	-	-	-	+
Implementation model	-	-	-	+	-	+	+
Infrastructure model	-	-	-	-	+	+	+

Further we analyse the studied methods according to the applied approaches and completeness of a decision model according to the proposed decision model structure introduced in Chapter 3.2. Moreover, we compare methods according to the support of specific decisions according to the classification described in Chapter 1.2. and the principles of intellectualised BIS development introduced in Chapter 2.4.

The methods proposed by Barbara von Hale *et al.* and OMG (BRG) use a similar approach and involvement of decision table based business logic implementation. Avdejenkov *et al.* and Asuncion *et al.* methods are for specific solution of a separate decision problem such as ECA rule transformation into the DBMS trigger or integration of production rules executed in Jess rule engine integrated into the SOA component for collaboration between directly incompatible services. IBM (ILOG) and Microsoft WFF are technology oriented and do not provide rich scientific background.

The main difference in Microsoft approach from the others discussed is the implementation of models for use of implicit knowledge and a rich set of tools for data mining and integration of multidimensional data analysis based on OLAP and execution instructions in MDX language. However, Microsoft approach lacks support for involvement of business people in the control and authority of decision logic at run-time. The last column represents characteristics of the method proposed in this thesis, and it is discussed in the next chapters.

1.6. Conclusions of Chapter 1 and Formulating Tasks for the Dissertation

This chapter concludes in the formulation of the main objective and tasks of the present research. Being confident about sufficient accuracy of the analysis performed, we raise concerns about technological or methodological limitations of the separate approaches proposed by the authors. Complexity of the issue is indicated by the wide-spread use of different techniques and approaches to the problem of flexible development of information systems and further alignment with rapidly changing business environment. We have also observed that very few reports on the use of decision models in the sphere of information system engineering are available.

However, decision modelling and conceptual process modelling literature-based analysis demonstrates that business domain knowledge represented by rule and process models could be effectively used as a source of knowledge for decision-making process automation using rule based executable business processes implemented into the information system.

Moreover, we have established that current methods reported are either focused to support a technological side of the problem discussed or fragmented and do not provide a complete solution for consistent evolutionary development of flexible decision process implementation including business orientation and comprehensive implementation on the technological side.

The task for further research presented in the thesis is to combine the reviewed opportunities provided by the new emerging technologies and the best characteristics of the existing methods and to create an improved method for development of flexible decision process implementation into the information system discussed in the following chapters.

2

The Framework for Intellectualised Information Systems Engineering

This chapter proposes a framework for intellectualised information systems engineering. In this study we analyse a role of business, information systems and software models as well as their relations across different levels of abstraction and different modelling perspectives and views such as a rule, process, information, structure, event, and resources. This chapter also deals with the aspects of model transformation into the executable models and development of the engineering procedure in order to provide a reliable and simple technique. The framework presented in this chapter was for the first time published in our scientific publication (Vasilecas and Smaizys 2006a). Parts of this chapter have been also published in our scientific publications (Smaizys and Vasilecas 2009a; Vasilecas and Smaizys 2007a; Vasilecas and Smaizys 2007c).

2.1. Frameworks for Enterprise Information Systems Development

This chapter analyses the existing frameworks for Information System development and research. The first framework for the research in computer-based Management Information Systems (MIS) was presented in 1980 by Ives *et al.* (1980). The objective of the study was to define a boundary of MIS and generate a direction for further research. The proposed model consists of three information system environments (user, IS development, and IS operations environments), three information system processes (use, development, and operation processes), and the information subsystem itself (IS content, presentation form, and time of presentation), all of which exist within organizational environment and external environment.

In 1986, D. Avison and A. Wood-Harper presented the Multiview framework submitted in Fig. 2.1, which was extended in 1998 and published as the Multiview2 (Avison *et al.* 1998).

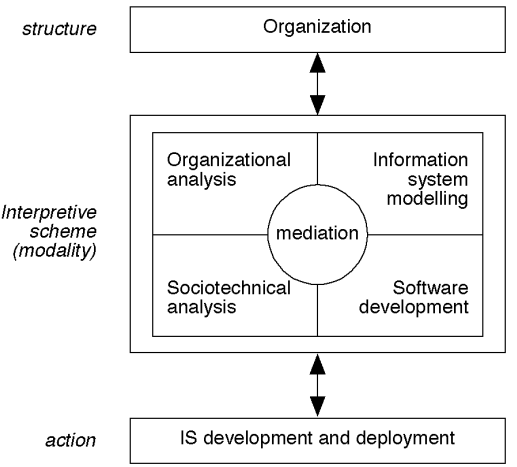


Fig. 2.1. The Multiview2 framework (Avison *et al.* 1998)

The Multiview framework consists of three main elements including organizational behaviour, IS development and technical artefacts which are produced during the five consistent design processes performed according to the methodology: organizational analysis, sociotechnical analysis, information system modelling and software development with a particular focus on mediation.

In 1987, John Zachman defined IS architecture, first "by creating a descriptive framework from disciplines quite independent of information systems, then specifying information systems architecture based upon the neutral, objective framework and view of each participant in the IS engineering lifecycle" (Zachman 1987). This framework has become known as the Zachman framework by the name of its author. The initial Zachman framework consisted of three columns (for data, process, and network aspects descriptions, respectively) and five rows, making a 15-cell grid structure. In 1992, Zachman and Sowa (1992) extended the framework by adding three additional columns: people, time and motivation. This was the first time when BR has been explicitly introduced in the motivation column and found its place in the process of IS engineering. The sixth - motivation - column of the framework represents BR aspect in business systems and IS of the enterprise according to the different views: scope, business model, system model, technology model, components and describing the final implementation of BR in a newly functioning enterprise, starting from vision, mission, goals, business policy and BR until the final implementation in particular software system process executed with respect to the BR, which either prescribe a certain action or constrain a set of possible actions in the final software system. Relations between motivation and other columns of the Zachman framework are explicitly presented by Hay (2003).

All the frameworks overviewed in this chapter are oriented towards the interaction between the social sphere of an organization, technogenic sphere of available technical infrastructure and software itself as well as the IS development team. However, it remains still unclear how to achieve such interaction because business people and IS designers and software developers have different focus and views based on their experience and their role. Summarising the results of the analysis, we arrive at the conclusion that the framework for IS development should cover three separate conceptual levels focussing on the main groups of people participating in the IS development projects: business people as the decision makers, information analysts as the decision supporters, business operation performers and software designers as the technical infrastructure supporters. Further in this chapter we will introduce the three layer framework for effective and flexible intellectualised enterprise IS development.

2.2. Conceptual Modelling Perspectives

Krogstie and Solvberg (2000) have identified seven general modelling perspectives relevant for modelling tasks: structural, functional, behavioural, rule-oriented, object-oriented, language-action-oriented, and role-actor oriented. The Zachman framework (Zachman and Sowa 1992) includes six perspectives, i.e.

data (What), function (How), network (Where), people (who), time (When) and motivation (Why). Other authors interpret a data column of the Zachman framework as a structure, a network column as locations and add different perspectives depending on their own needs, like finances, etc.

In the scientific publication (Airosius and Smaizys 2007) we have proposed to extend the structure of four business process modelling aspects: functional, behavioural, organisational and informational proposed by Giaglis (2001) with the addition of the fifth one – logical, representing business logic in the process. Other authors (Regev and Wegmann 2006) use taxonomy of business process flexibility where they define five subjects of change: functional perspective, organization perspective, behavioural perspective, information perspective, and operation perspective.

Studying the possibilities of model adaptable information systems development, we have combined three main conceptual levels: business system (organisation perspective), information system (information perspective) and software system (physical, technological perspective) with four main modelling views of the intelligent information system: Rule (logical perspective), Process (functional perspective), Object (information perspective) and Data (structural perspective) by introducing a three layer framework for intellectualised information system development. Later on studying ECA rules we have added Event view (behavioural perspective) and Resource view (infrastructural perspective).

Having performed several researches on business process modelling, we have finally decided to employ a list of six views (perspectives) used for later analysis of conceptual modelling languages and specifications used for intensive modelling of a decision process and its logic presented below:

- **Rule (logical)**. The rule view. Logical perspective was firstly introduced in the sixth column (Motivation) of the Zachman framework. This is the most important perspective for representation of knowledge and logic used in decision-making. The examples of languages (specifications) are SRML, RuleML, SBVR, PRR, SWRL, OCL, and DRL.
- **Process (functional)**. The process view, functional perspective. This is a perspective representing process view. A process is defined as a set of activities which based on a set of assigned transformations converts the input flow to the output flow and consumes the assigned resources. The examples of such languages and specifications are DFD, UML, CPN, IDEF3, and BPMN.
- **Object (information)**. The object view, informational perspective. The basic paradigm of object oriented modelling languages includes constructs as an object and a class representing structure and methods representing processes in a functional perspective. This is especially useful for manipulation of business level models where structures of business

objects are not so important. An object can be also related to the business entities and resources used in process models as well as an information object included in the description of business logic or an event in the behavioural model. The examples of such languages are OMT and UML, semantic networks, conceptual graphs.

- **Data (Structural)**. The data view. The structural perspective concentrates on describing the static structure of a system or an object. The main construct of such a language is the entity, attributes, and types. The examples of such languages are ER, UML.
- **Event (behavioural)**. The behavioural perspective. Languages representing a behavioural perspective include constructs such as states and transitions between states. State transitions are usually triggered by events. The examples of such languages are UML (STD) and Petri-Nets.
- **Resource (Infrastructure)**. The actor and role perspective. The main construct of languages within this perspective are an actor and a role. An actor and a role represent business resources required to perform activities on the system. In automated decision-making system resource can be represented as an information system or its structural component (object). The example of such language is ORM. UML with some exceptions can be used as well.

Rule and process perspectives are the most important part of conceptual data models used for decision-making that will be discussed in more detail in Chapter 3.

2.3. The Three Layer Framework

The main ideas of this chapter have already been published in our scientific publications (Smaizys and Vasilecas 2010; Smaizys and Vasilecas 2009a; Vasilecas and Smaizys 2007a; Vasilecas and Smaizys 2006a).

To describe BR model based software development process, first of all we need to discuss a definition of what we call functional and process views. In our opinion, the main difference is in the analysis perspective (Fig. 2.2). If we consider the system as a black box, then from outside of the system a function view reflects system outputs; and a resource view reflects external interfaces. This way process view will be the view into the inside of the box to analyse how resources from the input are processed into the results (services or products) on the system output.

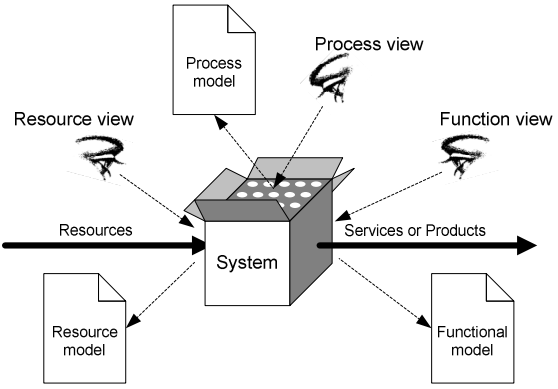


Fig. 2.2. System analysis perspectives - resource, process and function views

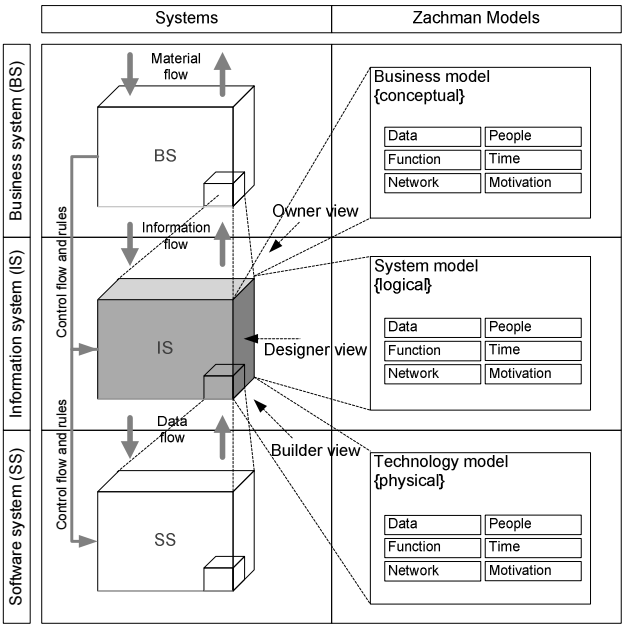


Fig. 2.3. Owner, designer and builder views to provide IS models

Such black box view is very popular in IS engineering methods based on reduction philosophy. Using a black box model we can look at information system and its inputs and outputs from inside of the system. For example, we can look at the information system (IS) from different perspectives as in Fig. 2.3 providing different models interpreted in the context of layers of the proposed

three layer framework. For example, with a focus on the information system as a black box we can map the Zachman framework (Zachman and Sowa 1992) models.

However, such view may be dangerous as it can lead to different descriptions and semantics of the same things at separate context layers. Using such a way of analysis, all attention is given to the information system. A business system together with software systems is treated as information system inputs and outputs. Such a view leaks focus on a domain where the problems of the real world exist. Zachman framework solves this by introducing additional rows to better describe the context of the information system.

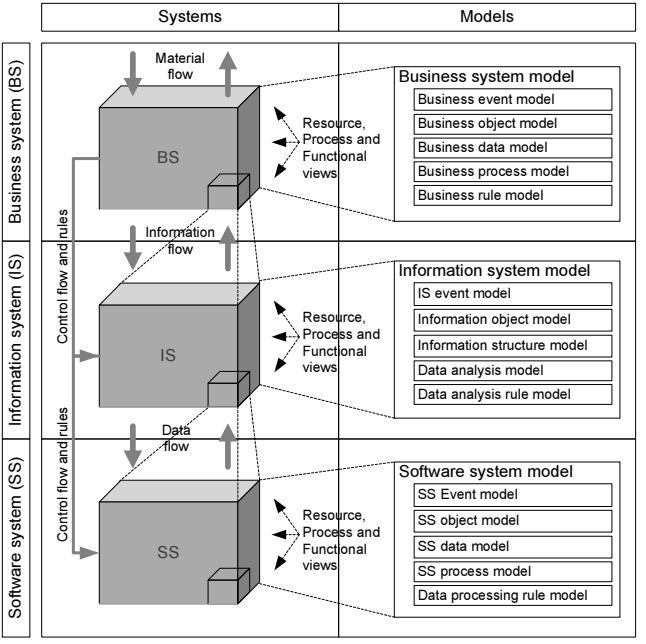


Fig. 2.4. Resource, process and functional views to provide BS, IS and SS models

We propose to treat IS as a part of BS, or as a set of SS's providing its functionality as the resources needed for IS (Fig. 2.4) where all three systems BS, IS and SS are considered as independent deeply integrated systems and their models are made by using white box philosophy, i.e. looking from inside out of every system. We understand that different views and perspectives on separate context layers is the main issue of misunderstanding and confusing definitions of what the models of BS, IS and SS are and how they fit each other going through the layers.

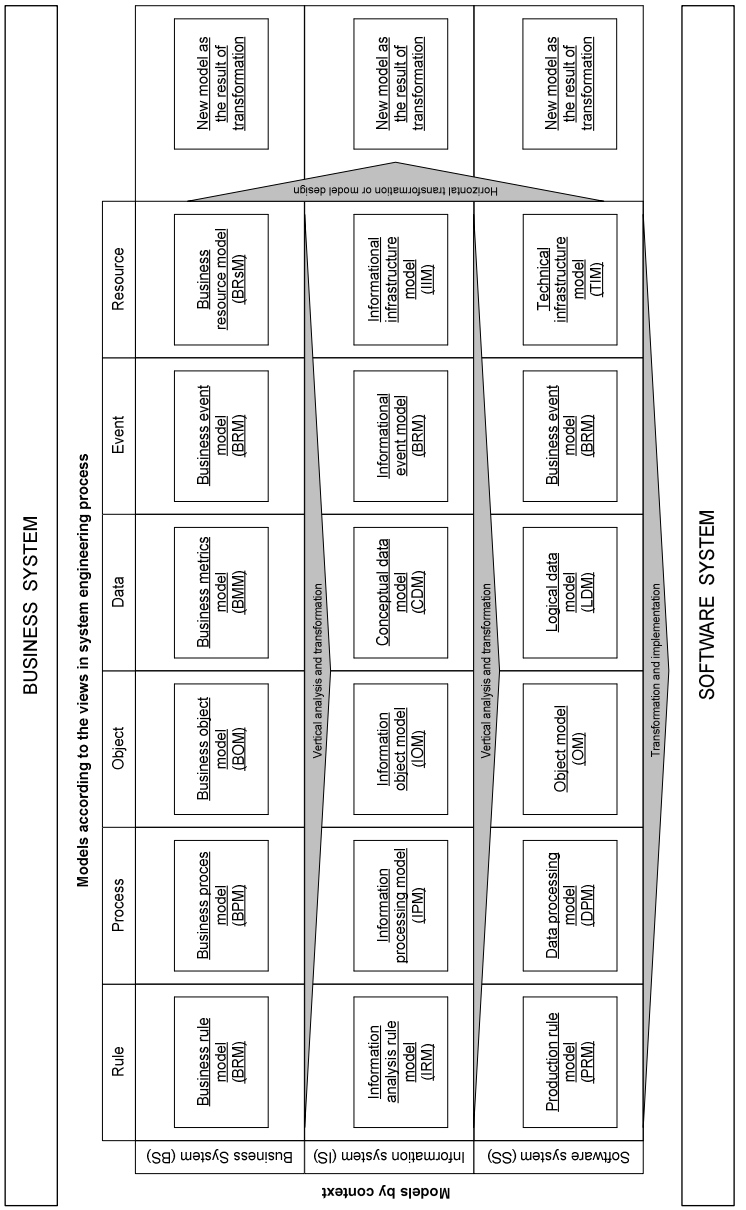


Fig. 2.5. The three layer framework for model-driven information system engineering

To enable better understanding among people working on the separate conceptual levels, we propose to arrange framework models and to group them vertically by the concepts modelled with a focus on the shared views modelled across the conceptual levels represented in the horizontal rows of the proposed three layer framework (Fig. 2.5) according to the context in the systems engineering perspective.

Introducing this framework we suppose that intellectualised BIS is a system threaded through all the levels of BS, IS and SS. First of all, IS is a part of BS providing functionality needed for business processes servicing all BS participants and evolving to automate more and more complex business processes going from operational to strategic levels. SS is a part of IS, where IS is providing data analysis and communication functionality for informational BS processes represented and modelled at IS level. At the same time looking from the bottom, from the developers perspective, BIS consists of several software applications elaborated in SS, which provide resources needed for servicing informational processes of IS and BS.

The proposed way of system analysis allows considering separate systems such as business system, information system and software system to analyse inputs and outputs of each of them treating SS as a part of IS and IS as a part of BS. All of them are integrated and represented as one BIS which represents not only software produced but also shows how such software is integrated into the entire business system and how it services informational business processes according to the business logic captured and represented in the separate BR model of Business system model.

In Fig. 2.5 we present a three layer framework for model driven BIS development with all possible models which can be used in different combinations according to the specific purpose and a specific framework based method.

Horizontally, the models of the framework are composed into three separate models: Business system model (BSM), Information system model (ISM) and Software system model (SSM) at three separate layers (BS, IS, SS) in BIS context.

Vertically, the models are grouped into the columns according to the views on similar objects produced during a system engineering process. Vertical columns represent six models: Rule model, Process model, Object model, Data model, Event model, and Resource model, which have been involved to represent conceptual modelling perspectives discussed in the previous chapter.

There are four main concepts of the proposed model framework, i.e. model, metamodel, model relation, and model transformation in both horizontal and vertical directions. Framework models are described in detail in our publication (Vasilecas and Smaizys 2006a). Metamodels and metamodel based model transformations are discussed in the next chapter. Model relations are discussed in

Chapter 2.4. The fifth important part of the framework is infrastructure necessary to support development of decision automation based intellectualised BIS. The framework for infrastructure support is presented in Chapter 2.5.

Based on the framework, we propose IS engineering process arrangement by developing several related models to reflect all the needed vertical views and perspectives in every horizontal set of models or so-called layer representing engineering context. In further research on decision automation we will mainly focus on development and use of Business rule and Business process models that are a part of Business model and are related to the Business object model which provides vocabulary and terms needed for specification of Business rules represented and stored in the Business model.

2.3.1. Framework Model Transformations

According to Czarnecki and Helsen (2003), at the top level we distinguish between model-to-code and model-to-model transformation approaches. In general, we can view transforming models to a code as a special case of model-to-model transformations; we need only to provide a metamodel for the target programming language. However, for practical reasons a code is often generated simply as a text, which is then fed into a compiler. The researchers also distinguish between visitor-based and template-based approaches in the category of model-to-code transformations. In the model-to-model category, the authors distinguish among direct-manipulation approaches, relational approaches, graph-transformation-based approaches, structure-driven approaches, and hybrid approaches. We use the proposed taxonomy for classification of model transformations used for investigation of the proposed framework.

In the methods based on the proposed framework we do not limit any use of model metamodels and transformations. However, for our experiments on model-to-model transformations we have adapted transformation of MDA-based software development process presented in Fig. 2.6. Such concept of model transformation was initially proposed by Czarnecki (2006) and later adapted by MDA. This transformation is based on mapping of source and destination model metamodels. The result of mapping is a transformation schema, which includes all transformation functions necessary to transform structures determined by the metamodel of a source model and source model data into the structures and data of a destination model. To ensure adequate mapping, source model should be validated according to its metamodel. The destination model should be validated according to the destination model metamodel. Both model metamodels and transformation language used for mapping should be validated according to the shared meta-metamodel. For the purpose of the research, we have made several transformations included in Table 2.1.

Table 2.1. List of the framework models and model transformations examined

Framework model transformations	Meta-models	Complete-ness	Time	Implementation technique, transformation language and tools
BRM-to-CDM	DT, ER	Complete	Design	XSLT
BRM-to-PRM	SRML, predicate	Full	Design	XSLT, for use in Jess
BRM-to-BPM	ECA, BPMN	Partial	Design	Algorithm based tool
BRM-to-BPM	OAO, DFD	Partial	Design	Template
BRM-to-IPM	SRML, MDX	Partial	Run-time	XSLT
BRM and BPM-to-DPM	RDR, BPMN, YAWL	Full	Run-time	XSLT
BRM-to-OM	SRML, XForms	Partial	Run-time	XSLT
LDM-to-IOM	-	Full	Design	LLBLGen tool
IOM-to-OM	-	Full	Design	LLBLGen tool
IOM-to-BOM	-	Full	Design	IBM ILOG for .Net

As an example of transformation, in Fig. 2.6 we have presented transformation from our scientific publication (Rima, Smaizys, Vasilecas 2007). The main purpose of the transformation is to get a model of dimensions in MDX language out of the business rule model in SRML language. First of all, we needed to develop two XSD schemas for SRML and MDX language. Every XSD schema is a metamodel of a corresponding model, e.g. a business rule model in SRML language. While mapping, we mapped corresponding constructs from SRML XSD schema into the constructs of MDX XSD schema by producing a transformation code in XSLT language. For the constructs with non- direct transformation we have created custom conversion functions in X/Query language. All transformation functions have been included into one XSLT file. Now the produced transformation instruction in XSLT file allows transformation of a business rule model into the MDX instruction which can be executed in the analysis server and produce or upgrade dimensions in analysis server. Such implementation allows on demand execution of transformation in run-time after changes of rule conditions in the business rule model.

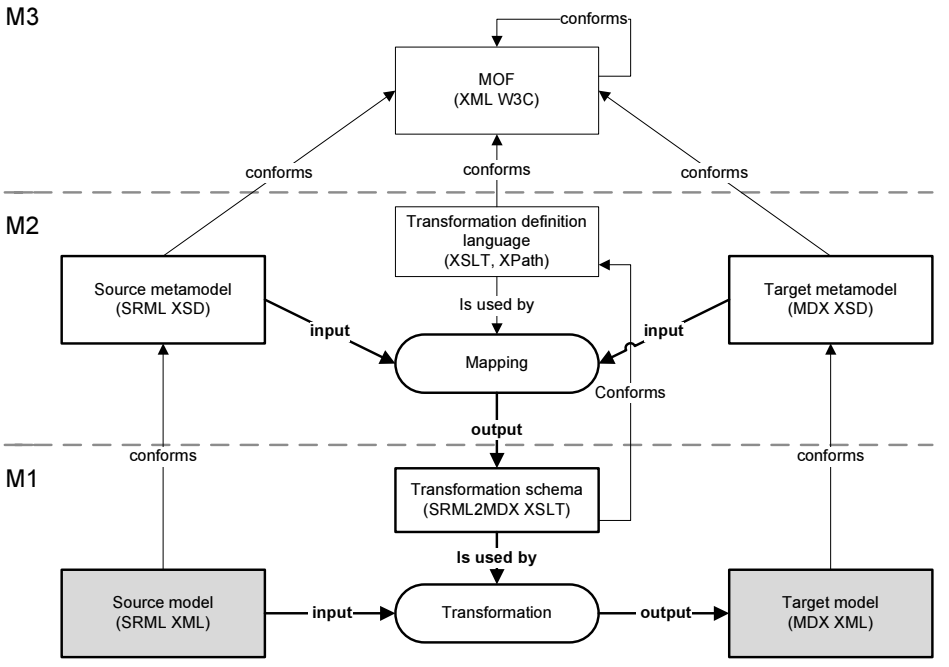


Fig. 2.6. Framework model transformation model adapted from MDA

Initially, MDA-based software development process seems similar to the process proposed by our framework; however, there are significant differences. In Fig. 2.7 we compare MDA model space and the model space provided by our framework.

Model space, the cube, in the picture is limited by three axes. The vertical one represents MDA models (CIM, PIM, PSM) whereas a vertical plane represents MDA model space while models of the proposed framework (BSM, ISM, SSM) are placed on the horizontal plane representing model space of the framework. All framework models are arranged according to the context in diagonal axis and views in horizontal axis. The right vertical plane represents the real world transformation space. The rows represent model transformations and projection of the real world transformation into the MDA transformation space.

The proposed framework uses Business system (BSM), Information system (ISM) and Software system (SSM) models as displayed in the picture instead of Computation Independent Model (CIM), Platform Independent Model (PIM) and Platform Specific Model (PSM) as it is denoted by MDA.

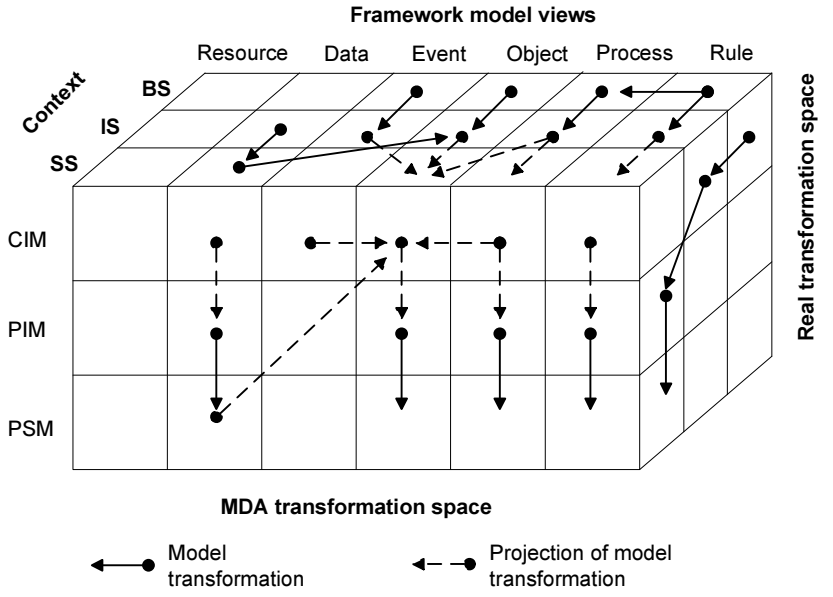


Fig. 2.7. Framework model transformations in the context of MDA

In some of our publications we merge BSM and CIM, ISM and PIM, SSM and PSM. This is a reduction to achieve a compromise with MDA philosophy. However, as we see from the picture, there is a risk that such projection hides important models and transformations as it is depicted in the case of business rule model transformation into the business process model or business rule model transformation from BS context into the IS on the conceptual level of CIM model space.

2.3.2. Infrastructure Framework for Intellectualised IS development

According to the previous research on infrastructure support for intellectualised IS development described in Chapter 1.4, we have identified the increasing potential to support business decision by evolutionary involvement of multidimensional analysis IS, Data mining systems and reporting systems. Further in this chapter we will present an infrastructure framework for integrated intelligent IS for decision automation presented in Fig. 2.8. This framework was introduced in our scientific publication (Vasilecas and Smaizys 2005a) with further improvements published later (Avdejenkov, Vasilecas, Smaizys 2008; Avdejenkov, Vasilecas, Smaizys 2009).

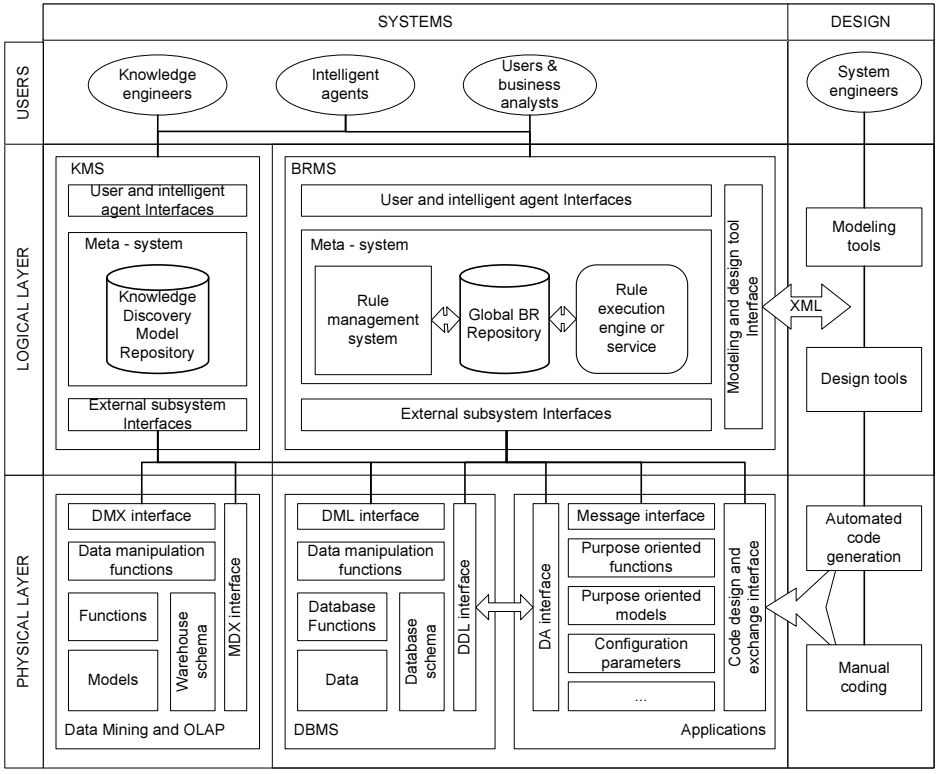


Fig. 2.8. The infrastructure framework for integrated intelligent IS for decision automation

The main advantage of this approach is that a part of software dedicated to business decisions together with the rules describing business logic is separated and implemented into the central component of BRMS. This component is responsible for business decisions and includes a model of decision propagation into the components of the enterprise software through several available information system interfaces. The central part of the proposed framework includes some reasoning processor for business rule execution. For example, rule execution engine, which is also called BRE, carries out various actions based on the reasoning results in the internal inference engine, ensuring data transferred between any two subsystems to conform according to the BR and data entered by users representing facts.

The interface to the external systems builds communication between business users, internal BRMS system components and external software systems. The emerging agent software can be used for automation of BR discovery as well. The interface plays a key role in an open structured BRMS software system:

codify human expertise into the software system in such a way that it can adopt the most creative intelligence and knowledge in decision-making, transform formally represented BR into human language for review and edit by users, communicate with other external intelligent software system agents for more complicated tasks. BR based meta-rules concentration in one BRMS and rule exchange interfaces support intelligent interaction between different subsystems in an enterprise. Moreover, due to a flexible XML language used for BR formalization, it allows introducing an open organisation structure with a new intelligent functionality. Such a BR based meta-knowledge base system creates an open organization structure and allows a new intelligent functionality to enter meta-knowledge and use it in all subsystems with corresponding rule exchange and enforcement interfaces to engage more duties in a decision-making process.

BRMS in this context allows not only integration of new services and business logic into the existing applications at the physical layer but allows indirect integration of applications through BRMS or DBMS as well. According to our proposed framework, the integration possibilities are limited only by architecture of the application and functionality of external communication interfaces of DBMS and applications. DBMS usually include at least two interfaces for message exchange in Data manipulation and description languages represented as Data Manipulations Language (DML) and Data Definition Language (DDL) interfaces in the framework.

The proposed infrastructure framework provides architectural context and is oriented towards the development of sufficient technological environment for support of model driven decision process development and further implementation with maximum reuse of separately stored design time models and their further run-time execution. The framework identifies all the possible interfaces for intervention with dynamically generated executable code generated dynamically by run-time execution decision models guided automated decision implementation procedures.

2.3.3. The Lifecycle of Framework Based Engineering

We arrive at the conclusion that due to the huge complexity of decision-making environment it is impossible to design an efficient IS that could support any type of decisions in any context or state of business environment. To deal with this engineering problem, we propose a framework based lifecycle of incremental and Continuous Decision Process Design (CDPD) displayed in Fig. 2.9.

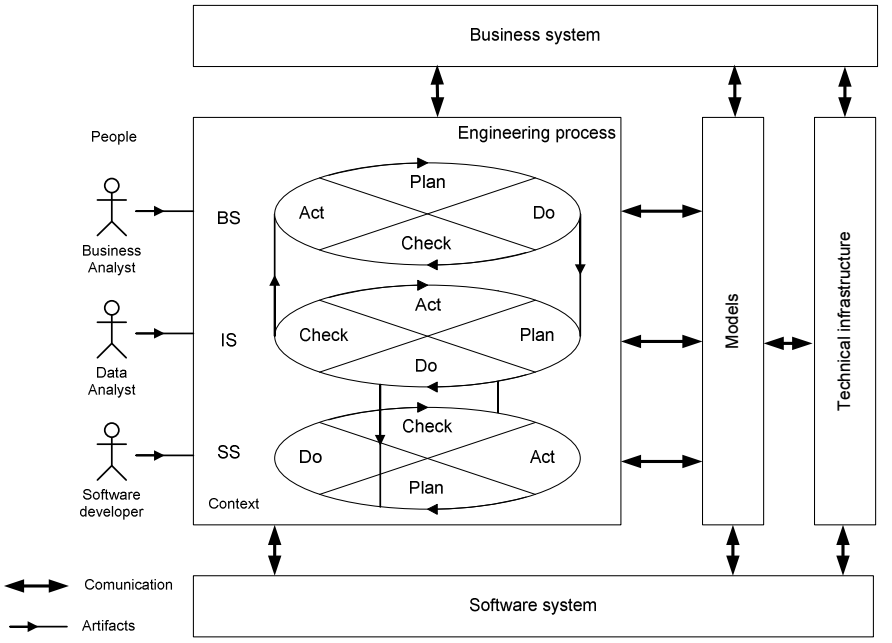


Fig. 2.9. The lifecycle of continuous decision process design

The lifecycle starts at BS level Act part where a business analyst analyses the existing business processes and identifies the need for improvement. The entire engineering process is rotated according to the Deming cycle simultaneously on all three levels BS, IS and SS with some delay which is determined by an entry communication row from the Do stage to the Planning stage at the next level. Every level produces models which are used for transformations or are integrated into the technical infrastructure and prepared for use by the separately developed executable software components used by business system users.

2.3.4. Formal Framework Based Method Definition

This chapter provides a formal definition of the framework and framework based method informally described in Chapter 2.3 and Chapter 3.

Mathematically, we define each model M used in the framework F as a directed labelled graph G:

$$G = (N, E) , \quad (1)$$

where N is a finite set of labelled nodes, representing model elements and E is a finite multiset of labelled edges. Model elements can contain attributes. The

edge E is represented by a tuples (n_1, n_2, α_1), where n_1 and n_2 are members of N and α_1 is the label of the edge, representing relationship between model elements.

All model elements and their relations can be represented as two sets:

$$N = \{n_1, \ldots, n_s\}, \tag{2}$$

$$E = \{(n_1, n_2, \alpha_1), \ldots, (n_{s-1}, n_s, \alpha_k)\}. \tag{3}$$

Model G metamodel MM is also a model of the model G and can be defined the same way as a model G, where every model element n_s correspond one of the metamodel elements h.

Source model G transformation $G \rightarrow G'$ is defined as:

$$T = (G, P), \tag{4}$$

where G is the starting graph representing initial model and P is a set of source and target metamodel element mapping based transformation rules, representing functions used for source H and target metamodel H' mapping:

$$P = \{p_1, \ldots, p_k\},\ p_k = f\left(h, h'\right),\ h \in H,\ h' \in H'. \tag{5}$$

The complete set of framework (Fig. 2.5) models F will be a Cartesian product of all models used in the framework according to:

$$F = M_v \times M_c\ ,\ M_v = \{M_{v1}, \ldots, M_{vc}\},\ M_c = \{M_{1c}, \ldots, M_{vc}\}. \tag{6}$$

The relationship between framework models M_{ij} will be a directed labelled graph, where N is a finite set of labelled nodes, representing framework models and E is a finite multiset of labelled edges. The same way edge E will be represented by a tuples (n_1, n_2, α_1), representing relation r between two models n_1 and n_2.

In general all models and their relationships are being represented as two sets:

$$M = \{M_{ij}\},\ M_{ij} \in N \tag{7}$$

and

$$R = \{r_k\},\ r \in E, \tag{8}$$

where M represents a set of framework models and R represents a set of all framework model relationships.

A framework based method D will contain a set of all framework models m, used for the purpose of a method, its metamodels mm in MM, model relationships r in R and model transformations t in T used to produce the resulting mod-

els during the engineering or runtime model execution according to the method *D*:

$$D = \{m, mm, r, t\},\ m \in M,\ mm \in MM, \mathrm{r} \in R, t \in T\ . \tag{9}$$

2.4. Principles and Approaches of the Framework Application

Modern packaged BIS applications in the enterprise may already have some embedded business rules technology. Business rules in BIS packages usually are covered by complex configuration sets implemented to achieve system adaptation to the changes of basic business rules which represent most vulnerable parts of a business process. The approach discussed in this thesis is focused on implementation of a more flexible approach which covers not only business rule separation but also other important aspects of intellectualised IS concept, e.g. system adaptability to the changes of business behaviour where the system resides and enrichment of decision-making processes by automated discovery of analytical information and implicit knowledge discovery.

Based on our experience gained in developing intellectualised information systems, we could define the following principles of BR approach implementation and strategies for higher BR based BIS development maturity:

- **Separation** of business logic. Declarative business logic should be separated during the design process and implemented according to the transformation driven BR implementation process. Business logic should be used for inference in the separate components of the software system and executed by a static part of the software developed for decision implementation at the design phase or the dynamic implementation process composed by execution of run-time implementation model transformations developed at the design phase. Such dynamic implementation process can be also based on the separated declarative logic model;
- **Centralisation** of business logic execution. For adequate functioning of distributed software components across the enterprise, shared business logic should be centralised and stored in the separate space, where dedicated components for execution and enforcement of BR should be implemented. This principle is typically implemented by involvement of dedicated BR repository and business rule execution engine components performing reasoning tasks and providing solutions based on declarative logics to the integrated software components implementing automated control tasks of the business process;

- **Substitution** of decision related activities by declarative description of the business logic. The dynamic business process parts vulnerable to rapid changes should be substituted by the controlled processes implemented into the information system according to the proposed decision model. Implementation of such principle is based on extraction of business logic and specification of control process using declarative approach instead of the explicit and precise definition of activities in the business process usually modelled according to the imperative approach;
- **Integration** of specialised components for decision-making automation. Main processes of the automated decision process should be developed separately and integrated for parallel and independent execution and implementation of decision support, decision-making, and knowledge discovery. Following this principle, it is necessary to consider the design and development of the final software system for decision process automation in the context of technical enterprise infrastructure; where the software components enabling execution of decision models, informational support and control of the automated business process became an integral part of the enterprise information and business system;
- **Transparency** of business logic. Separated and centralised business logic implemented into the information system should be transparent to the business level and available for direct access and control by business people in the form or language they can understand. The main reason for this principle is the transfer of responsibility for automated decisions from the system to the business entity which gets full control on the business logic and for the results of such automated decisions;
- **Intellectualisation** of a decision process**.** Implementation of intelligent algorithms to allow knowledge discovery in corporate data and later use of such implicit or explicit knowledge in the model of decision logic allows automation of more sophisticated business decisions in BIS, e.g. it enables intellectualised BIS self-adaptation to the changed business conditions. Such approach directly involves automated business rule mining out from the enterprise data and intuition substitution by the analysis based reasoning in the decision-making process.

Moreover, we define a list of strategies for decision automation implementation, experimentally evaluated by applying the proposed framework for the development of flexible intellectualised information systems, listed in the order by growing supported flexibility:

- **Parameterisation**. The strategy when business logic is implemented as a set of configuration parameters guiding behaviour of IS. This approach is discussed in Chapter 4.1.2;

- **Design-time or run-time model-to-model transformation.** Separate development of a business logic model and transformation into the information processing rules at the IS level and model integration for use or execution within a dedicated infrastructure component;
- **Design-time or run-time model-to-code transformation**. The approach is based on model-to-code transformation for production of executable software components. This approach is discussed in Chapter 4.1.3.

The performed experimental studies demonstrate that evolving implementation of the principles defined in this chapter directly result in higher system complexity and difficulties in arrangement of traditional information system engineering processes used. Further in the next chapters we discuss approaches of BR based BIS system development and analyse their advantages and disadvantages according to particular aspects.

2.4.1. Separate Development of a Business Logic Model and Integration into the Information System

Literature studies show a series of ways to implement business rule enforcement and implementation into the final software system proposed: to implement into relational database constraints (Zimbrao *et al.* 2003), triggers (Valatkaite and Vasilecas 2002; Sosunovas and Vasilecas 2004) or executable code in applications using resolution of separately stored BR and facts representing entered value instead of validation code (Tang et al. 2005). The Casati *et al.* (2000) discusses an approach of parameterization. This is an approach, which has been discussed in our publication (Vasilecas and Smaizys 2008c), usually used for the development of a particular component of IS. This enables separate development of a static business logic model applied in IS software for execution of structured decisions using software system parameterisation. The main advantage of such approach is the possibility of software component reuse (Kilov and Simmonds 1996). Another alternative is a model driven service composition (Orriens *et al.* 2003; Rosca *et al.* 2002). This approach is similar and usually does not require the use of rule engine. However, inference processing and involvement of some rule engine is useful for validation of BR combined into the rule sets used for further automated transformations. To sum up our experience, we could define two different alternatives of such BR enforcement and separately developed business logic model implementation in BIS software:

- System parameterisation using BR and business logic usually represented in decision tables;

- Business rule and business process model automated transformation into the executable business processes and business logic exchange between separate parts of the business processes implemented into several software applications later on.

Based on our research, we can define the following two main approaches of BR based IS engineering: Parameter Driven Approach and Model Transformation - driven Approach. Butleris and Kapocius (2002) distinguish the third one, called Independent Process-driven Approach, which is similar to the ILOGs approach of BR based business process execution using BRE.

Considering the analysis performed in our publication (Vasilecas, Avdejenkov and Smaizys 2008), we can draw a conclusion that the business logic exchange is vital for easy and fast IS modification and adaptation to the business changes. ILOG's approach to business logic exchange is achieved by using separate Rule management tools and Rule Execution Server. Such rules or actions according to ILOG are exchanged through BizTalk server where transformations needed are implemented in a special tool. Our Model Transformation - driven Approach is related with the use of XML source to destination transformations based on execution of transformation instructions stored as XSLT schema. Where, the XSLT schema is developed by mapping metamodels of the source and destination models. Another yet approach alternative to the business logic exchange achieved by centralization of business logic execution is presented in the next chapter.

2.4.2. Centralized Business Logic Execution and Business Rule Enforcement in Integrated Systems

There are strong arguments for a non-redundant and centralised implementation of the IS-relevant BR in the database. This approach has an additional advantage that the rules cannot be circumvented by operations on the DBMS because of triggers and stored procedures are used for rule integration (Badawy and Richta 2002). However, business logic used in business processes is distributed through different information systems software and usually differs depending on technology used for each particular software system. This leads to the problems dealing with inadequate automated decision processing and implementation of changes into different software or several parts of the IS. This can be avoided by using centralized business logic execution and BR enforcement in the integrated BIS system according to the framework presented in Chapter 2.3.2.

2.4.3. Intelligent Self Adaptable Systems

Nowadays, information system development is experiencing transition from crisp logic to fuzzy logic. This allows for both algorithmic and heuristic problem solving. Currently, considerable research is being done on fuzzy logic relational databases (FRDB) and fuzzy logic query languages (FSQL). The authors introduce fuzzy attributes, fuzzy constraints and creation of FuzzyEER model with a focus on semantic aspects in the conceptual design (Galindo *et al.* 2006). Fuzzy logic was defined in the late 60's of the 20th century and enabled representation of more complicated rules with implication functions which were impossible using binary logic, e.g. decision systems, where a decision is dependent on a large number of factors.

During our research we have challenged a few not previously discussed problems related to the situations when BR are not known at the beginning or they are incomplete even contradicting but still requiring decisions. Moreover, such BR system is not static and should be changed when a business situation changes. That means the need for the development of the software systems that could be self-adaptable to such changes and ensure information system adaptation to maximise achievement of the goals dedicated by the executive staff. Such situations may be resolved by involvement of fuzzy logic or flow engines implemented in BRMS component called BR mining engine. We have carried out some experiments for solution of such a problem by building a risk model (Vasilecas and Smaizys 2008c). Other authors use statistical methods for simulation of decision influence on the future state of a business system. Such approach allows dynamical adaptation of the user interface of the BIS application to the changes in a business system observed from the collected enterprise data and generation of the business service and results in the extensions or modifications to the workflow logic, business logic or data logic being directly reflected in the operative user front-end and the presentation logic of BIS system.

To sum up the intelligent features of BIS that we can get from the approaches described in previous sections, it may be stated that they usually act by structured rules and do not deal with every possible impact on the business environment or further consequences. That is the main reason why automated decisions based on such rules cannot take responsibility and require involvement or approval of the dedicated business people. This limits decision automation possibilities. The use of fuzzy logic instead or together with crisp logic in BIS systems can be used to simulate a real business environment and evaluate possible impact providing automated heuristic decisions.

2.5. Conclusions of Chapter 2

This chapter presents a research on the proposed framework and discusses framework application approaches. The main reason for the development of the presented framework was an idea that business people should have direct control on all business level models and particularly on business logic used for automated decisions implemented into the information systems. We have presented a description of the technological infrastructure needed to enable the implementation of framework models, including intelligent support for implicit knowledge and use of artificial intelligence. To provide better flexibility, we propose extensive use of framework model transformations.

The proposed framework is created to enable better understanding among people working on different conceptual levels. Therefore, we propose to focus on the same objects by introducing shared views represented in the columns of the framework according to the context in the system engineering perspective. In Chapter 3 we demonstrate how the proposed framework can be applied for the development of decision automation methods for intellectualised IS development.

3

A Method for Decision Design and Implementation Using the Proposed Framework

The current chapter presents a method developed according to the proposed framework. It includes a description of the method for flexible decision design and implementation into the intellectualised IS. The proposed method is dedicated to automation of business decision-making processes supported by the separation and a declarative way of business logic description. Other important aspects of the proposed method are related to involvement of analytical information support and involvement of implicit knowledge for impact prediction of decision implementation and decision model enrichment.

The structure of the method description is structured as follows. It starts with the description of the proposed general decision models, and then it is continued by the description of the structural decision model used in the proposed method. Further chapters present a discussion of model integration and alignment with a special focus on the analysis of the rule and process languages. Further chapters explain transformations developed and introduce a set of tools used to support the proposed method during experimental evaluation. The final chapter summarises the proposed method by discussing the method flexibility.

3.1. Model of Decision Process

In this study we treat a business decision as an automated business process to guide a course of action implemented into the intelligent information system and guided by separately determined business logic, which is aligned with the current business strategy, assessed according to the existing business environment and oriented to the effective achievement of business goals according to the projected future.

We consider a process of decision modelling as a process of the business process analysis. The main analysis tasks involve identification of key decisions in the business process, capture of the corresponding business logic to support decision and decision model refinement to support the achievement of business goals throughout the entire business process. There are three separate inputs for decision-making. Two of them are provided by the dedicated decision support processes which feed decision engine with all necessary facts needed for decisions and a set of decision alternatives to be inferenced according to the business logic expressed in business rules, which are the third input for the decision-making process. The outcome of a decision is an answer or solution of a business problem formulated at the beginning of a decision process definition. The end of the decision analysis is a decision model in the form of structured decision logic specification expressed as a complete and consistent set of business rule statements. Informational decision support model, decision-making and implementation process model specification, optional specifications of a decision evaluation model, and environmental models are discussed in further chapters.

We group decision models into three main groups according to the evolving complexity determined by the additional support needed from the technical infrastructure according to the schema in Fig. 1.2:

- **Guided decision process** models represent a controlled business process, where decision-making is based on business rules representing business policy and is processed according to the actual facts represented by measuring business object attributes (measures) at the particular moment. Such decision process allows guiding a business process according to the business rules;
- **Reactive decision process** models represent business processes, where decision-making is based on facts representing measured facts and facts representing summarised analytical information provided by historical data analysis. Such decision process in addition allows evaluation of historical experience about the decision object;
- **Proactive decision process** models represent a complex decision process with a prediction model involved for what-if analysis or evaluation of future business system state and selection of corresponding corrective

actions on the active business policy used for operational decision-making. Such corrective actions are implemented by selection of a corresponding rule set, changing rule enforcement levels or rule limits (parameters).

Further, we will introduce a set of the proposed decision models and explain them by the example. It is important to note that we will stay on the business system level at this stage. A detailed explanation on implementation of informational and decision-making processes of the model into the information system will be presented in the next chapter.

3.1.1. Decision Model of the Guided Business Process

To represent decision models, we use a set of business processes stereotypes composed of: material or servicing business processes, information processing process, information analysis process, implicit knowledge discovery process, and decision-making. Sample scenarios for such decision processes are order validation, pricing calculation, promotion enforcement, exception process management, and claims adjudication and management or fraud prevention.

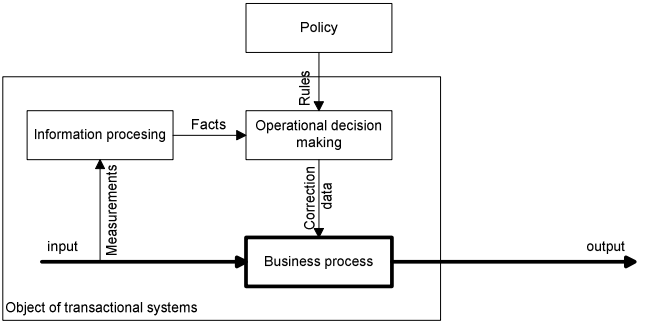


Fig. 3.1. Decision model of the guided business process

The decision model in Fig. 3.1 displays a general schema of the controlled business process, which is guided according to the rules in the policy. Performing informational analysis according to the proposed decision model we can create any composition of the controlled business process model. As an example of such case scenario for illustration we will use a simple business process of sales. We start the design of the controlled decision model with a definition of goals for the designed business process, e.g. “consistent and valid sales data”. Now we will start to follow the data flows in the model and create a sequence of activities starting with measurement and registration of input variables. Firstly, it is necessary to create dedicated activity of type information processing as displayed in

the model, e.g. registration of customer data, selection of goods, etc. Information processing actions can be grouped into separate sub-process, if needed. It is not important what variables will be registered at this stage. The next activity in the business process model will be a type of decision-making. Therefore, we will need to decide what corrective procedures will have to be implemented for the achievement of the defined goals. We start with consistency checking and continue with validation as displayed in Fig. 3.2, where we demonstrate the illustration of the process by providing UML activity diagram. Moreover, we need to decide how to implement corrective actions, e.g. to report a problem if some problem occurs.

We have intentionally skipped material activities which can be included in the business process model at the business level. However, such material activities represented as a business process itself in the decision model are not directly implemented in the information system as well as on further abstraction levels of IS and SS according to our three layer framework.

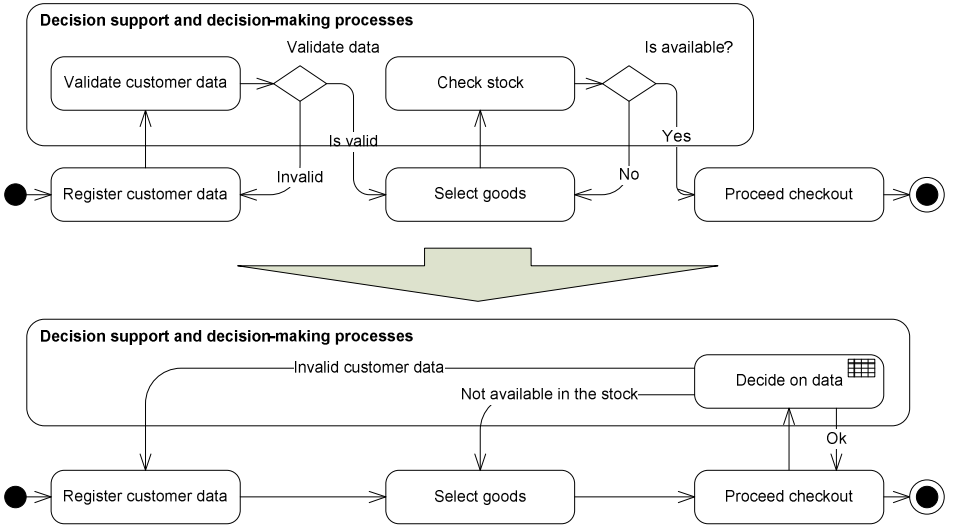


Fig. 3.2. Sample illustration of the controlled business process model in UML activity diagram

The lower variant of UML activity diagram (Fig. 3.2.) represents the modification of the process needed to adjust it according to the proposed method. This variant is more flexible than the upper. All business logics used for decisions on process data will be specified later and stored using business rules in separate repository. So there is no need to modify process schema for implementation of additional business policy represented by the new rules, e.g. “Alcohol

cannot be sold for a person with the age less than 21" or "It is not allowed to sell alcohol after 20 p.m.", which needs to be implemented as the additional diamond in case of upper schema. It is clear that upper model cannot be modified without modification of the software code. Such approach of decision logic separation for reduction of business process complexity is similar to the approach proposed by Barbara von Hale *et al.* (2010). However we also propose additional separation of decision support processes to supply additional information needed to support additional business logic. Following to our method, separation of business rules allows simplified development of initial models and quicker initial implementation of software code. It also allows more flexible business process adjustment to the changes in the business process environment without modification of software code related models at business and information system level.

3.1.2. Decision Model of the Reactive Business Process

Evolving controversial business goals of a business process, e.g. "customer loyalty should be rewarded" and "interest rate should be preserved" require continual change and flexible accommodation of existing business policy guiding decisions in control operations. This leads to a change of controlled business process model into a more sophisticated reactive decision model displayed in Fig. 3.3.

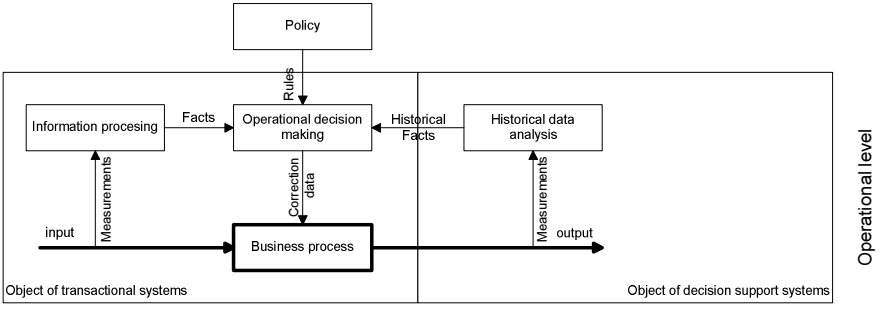


Fig. 3.3. Decision model of the reactive business process

The approach of business logic separation and later specification by using decision tables or decision trees as it was proposed by Barbara von Hale *et al.* (2010) is not sufficient for this purpose. Our proposition for solution of such limitation is separation and development of a separate model for decision support where we include all informational processes needed to feed a decision-making process with facts, including the facts representing analysis on historical

data as it is required in this particular case. However, the information processing model is not an issue of interest in the scope of business; therefore, we move it down according to our three layer framework into the level of IS model.

In rare cases, business does not know the real impact of decisions on business results; and judgement on performance of a business process is based only on indirect measures or indication of some unacceptable side effects. We emphasize the importance of design and integration of a business process performance evaluation mechanism into the business process model on the early design phase. That is the main motivation for the development of a goal model by establishing key performance indicators (KPI) and further implementation of measured variables of the process performance and informational support model at lower models according to the proposed framework. For implementation of control on the managed business process, we propose implementation of an automated e.g. KPI based control process on the top of the controlled business process, where process logic is expressed as a set of business rules whereas goals are stated in terms of the defined KPI model. This approach is supported by development of a measure model, a goal model and models of decision logic and integrated into the historical analysis and tactical decision-making processes for selection of adequate corrective actions and their implementation. It is indicated how to interpret KPIs and arrange business by selection of an adequate business strategy used for decisions and adjust a business process according to the guidelines provided by the management at the tactical level.

3.1.2. Decision Model of the Proactive Business Process

The next important addition to the proposed decision models is the introduction of a proactive business process as displayed in Fig. 3.4. Such addition is usually motivated by the evolved business requirements to dynamically adapt tactics of decision-making according to the current achievement of business goals or the need to evaluate the impact of decision for enrichment of decision logic. Simple control of a business process used in previous decision models by keeping tracked process values in the range limited by directives implemented into the business policy is not sufficient for this purpose.

New addition to the previous models is Impact Prediction Model. We relate this model directly with the use of data mining technologies. Data mining provides the means to make sense of tremendous volumes of data by automating the processes of categorizing and clustering common elements, identifying trends and anomalies in the data, and predicting what will happen given those factors. However, current information system methodologies do not provide sufficient power and tools for easily understandable and effective implementation of data mining into information system processes.

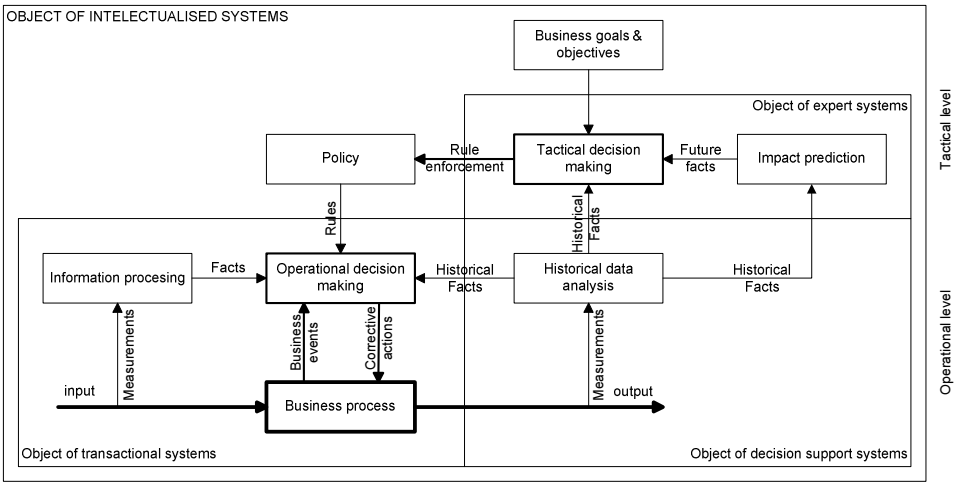


Fig. 3.4. Decision model of the proactive business process

In general terms, data mining algorithms use inputs to learn about outputs. However, each algorithm imbues its own interpretations to what exactly that means in the case of an attribute being both input and output, the convention adopted by Microsoft algorithms is that output attributes are never used to predict themselves. Therefore, the algorithms take steps to ensure the proper separation of information. This is dependent on the algorithm, but it can involve how many networks are created in a neural network, how clusters are computed, etc. DMX is the language that transforms the data you have (i.e. the tables with columns and rows) to the data needed by the data mining algorithms (i.e. the cases and attributes).

The mining structure describes a shape of the problem. Depending on how the structure is created, it can also contain bindings to your data sources so that the structure can be reprocessed without any need to provide the source data information again. Additionally, a mining structure contains all of the models that are used to analyse the source data of the structure. A mining model is the object that transforms rows of data into cases and performs the machine learning using a specified data mining algorithm. Prediction in DMX is somewhat loosely defined. In general, prediction means applying the patterns that were found in the data to estimate unknown information. Examples of prediction might be predicting if a customer will or will not be good for a loan, estimating a credit score, determining to what cluster a case belongs, determining the likelihood of customers cancelling their service, producing a recommended list of products, or predicting future values of a time series. A prediction process is often the final goal in any data mining scenario, providing the ultimate benefit of data collection and machine learning, and can drastically change how business is per-

formed. With traditional data mining systems, predictions can be done in a batch mode with the prediction results being stored in a relational database or some other destination. However, as predictions are efficient, it can also be done in real-time, interactive scenarios. A mining structure defines the domain of a mining problem, whereas a mining model is the application of a mining algorithm to the data in a mining structure.

There are many data mining products on the market, and each of these products has its own proprietary ways of describing and building data mining applications. Most data mining packages include their own algorithms, their own formats to browse and store model patterns, their own data-cleansing tool, and even their own reporting tools. Such approaches isolate the data mining systems from the enterprise operational systems, increasing the difficulty and cost of implementing data mining solutions.

The basic premise of data mining or machine learning is that you show the algorithm some examples and, from those examples, an algorithm can extract patterns (or rules) that can then later be used for inspection or to deduce information about new examples.

To sum up this section, it may be stated that we have identified three main decision models used in business processes: guided (controlled), reactive, and proactive. Having analysed the decision model based business process model composition, we have observed that the proposed model behaviour is similar to the behaviour of state machine but here the model produces activities of a business process model. We also believe that it should be possible to express the proposed decision models in terms of business rules representing restrictions on transfer of activity. Such approach should allow replacement of a business process model by declarative rules. Declarative approach for business process specification could allow even higher flexibility and getting any composition of the process according to the set of rules inference at run-time. This idea is similar to the BPMN (OMG 2011a) model transformation into BPEL but in an opposite way. We have examined such an idea to some extent in other scientific publications (Zaicev, Sliamin, Smaizys 2009).

3.2. Decision Model

This chapter discusses a structure of a decision model and proposes a set of models to be included into the developed decision model to support our three layer framework based decision automation method.

We agree with Linehan *et al.* (2011) on the relation of a decision model and SBVR, which can be used as a business logic model at BS level; and BPMN as a workflow or a decision process model. Such hypothesis was at the beginning of

our study in the field of decision modelling and automation. In 2011, the OMG released a Request for Proposals (RFP) for a Decision Model and Notation (DMN) specification (OMG 2011b): ”Decision Models are developed to define how businesses make decisions, usually as a part of a business process model (covered by the OMG BPMN standard in Business Process Management Solutions). Such models are both business (for example, using business vocabularies per OMG SBVR) and IT (for example, mapping to rule engines per OMG PRR in Business Rule Management Systems)”. It is evident that DMN may relate to the model of business logic and a decision process; however, there are more open questions than answers including the need for additional models related.

A theoretical model of such kind of a business process dedicated for making decisions was inspired by Business Motivation Model (OMG 2008a) discussed in Chapter 1.4.2. We define a decision model as a formal or informal conceptualization of the relationship of the various factors that are relevant for decision-making and planning.

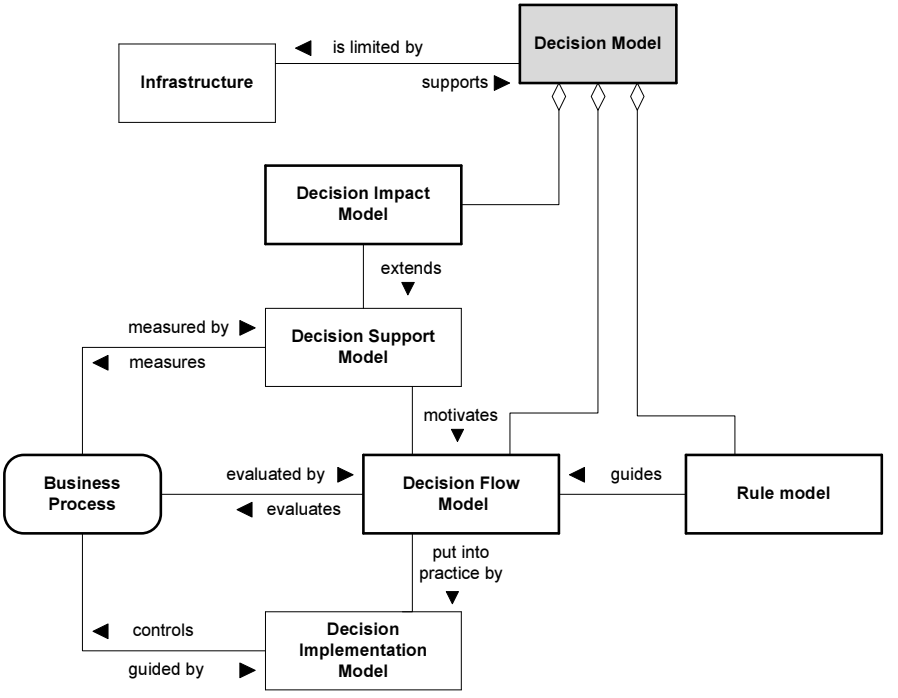


Fig. 3.5. Decision model structure

The proposed decision model presented in Fig. 3.5 includes five models representing important factors participating in the decision-making process:

- **Decision support model** is dedicated for informational decision-making support. The model should cover data structure and processes needed to collect all and interpret data to get information needed to be provided as a set of facts used for inference with rules used for decision-making;
- **Rule model** is a business logic model used as a source of structured or declarative logic used in a decision-making action;
- **Decision flow model** is a model of all decision-making process, which includes all actions needed to arrive at the conclusion. Decision process can be represented as a sequence of decisions made according to the separate sets of rules which after resolution provide facts to further resolution using an additional set of rules in the chain;
- **Decision impact model** employs additional scientific methodology to help understand, predict, or control situations or problems managed by the model and evaluate environmental impact of a decision or enrich the decision-making process by providing facts about the predicted future state of a business system;
- **Implementation model** focuses on implementation of the process of concluding which decisions need to be made and how to find alternatives for each decision;
- **Infrastructure model** determines the infrastructure needed for execution of the decision-making process and implementation of corrective actions.

It is also important to note the role of a goal model which is directly related to the decision logic and a decision support model and can provide additional facts for a particular decision action in decision flow.

A model of decision logic is implemented into Decision-making Model. A goal is the result towards which the effort is directed, an objective that should be achieved. A goal usually expresses what is desired, and in addition, it usually conveys why it is desired, which motivation and rationale can be behind it in a larger context. Goals can be oriented to achieve, maintain, stop, prevent, or optimise an object property or process variables. We support the approach of Navarro *et al.* (2004), where the authors introduce a meta-model for UML based functional and non-functional goal model visualisation and alignment with the requirements.

We treat a decision support model as a model developed at IS level which is used to supply a decision process with all the information needed to get all facts to be provided for inference with decision rules in the rule engine. A decision model is usually created by using UML class diagram at IS level or directly composed at SS level using DML, MDX or DMX language. A decision support model motivates decisions and is restricted by the Business Object Model of the controlled business process.

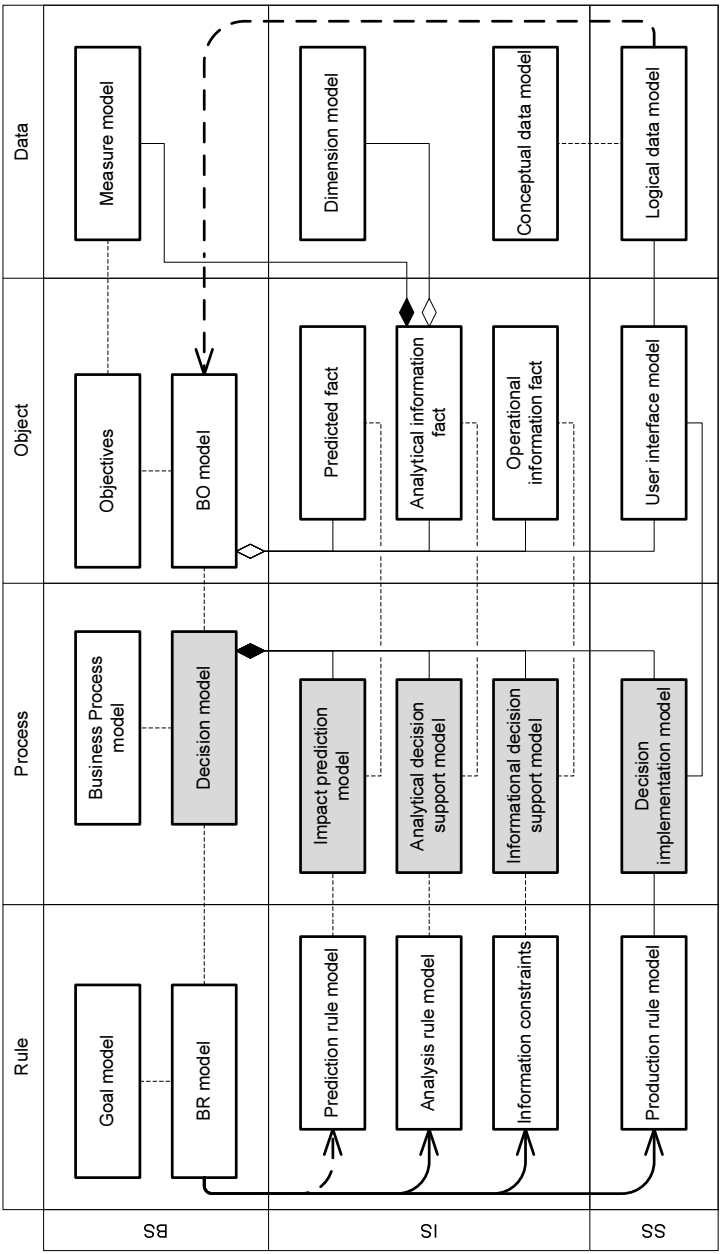
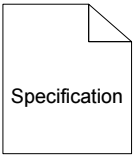
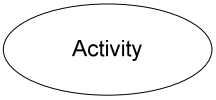
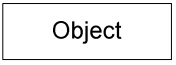
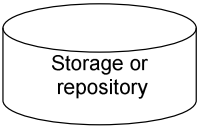


Fig. 3.6. Decision model in the context of framework

For initial visualisation of a decision model at the first stage of development, we create a conceptual decision model adapted to our three layer frame-

work and use specially developed Decision Model Notation (DMN), which was for the first time introduced in our publication (Vasilecas and Smaizys 2006b) and presented in Table 3.1. The example illustration of the model is provided in Fig. 3.7.

Table 3.1. Decision Model Notation.

Notation element	Description
	Specification of rules is used in a decision-making action. It can represent a set of business rules or rules at lower levels according to the context of a three layer framework.
	The activity is related to the automated decision process. It represents any activity later detailed in other models of the complete decision model. It may also mean a decision-making activity performed in the external inference engine of IS.
	It usually represents some data object or an abstract fact used in the inference process.
	It represents a storage to store data, models or knowledge used in a decision process. Depending on the context, it can represent a database, warehouse, or repository.
	Data, specification or object flow.

A decision flow process model is developed at IS level usually by using UML activity diagram and implemented at SS level by creating ILOG rule flow model or Microsoft WFF workflow model depending on the selected architecture and supporting technology. A decision logic model is expressed using business rules. Business rules are captured using rule specification language such as RuleML, SRML, etc. or represented as decision tables. Business rules at run-

time are transformed into the production rules or predicates used for inference in the rule engine.

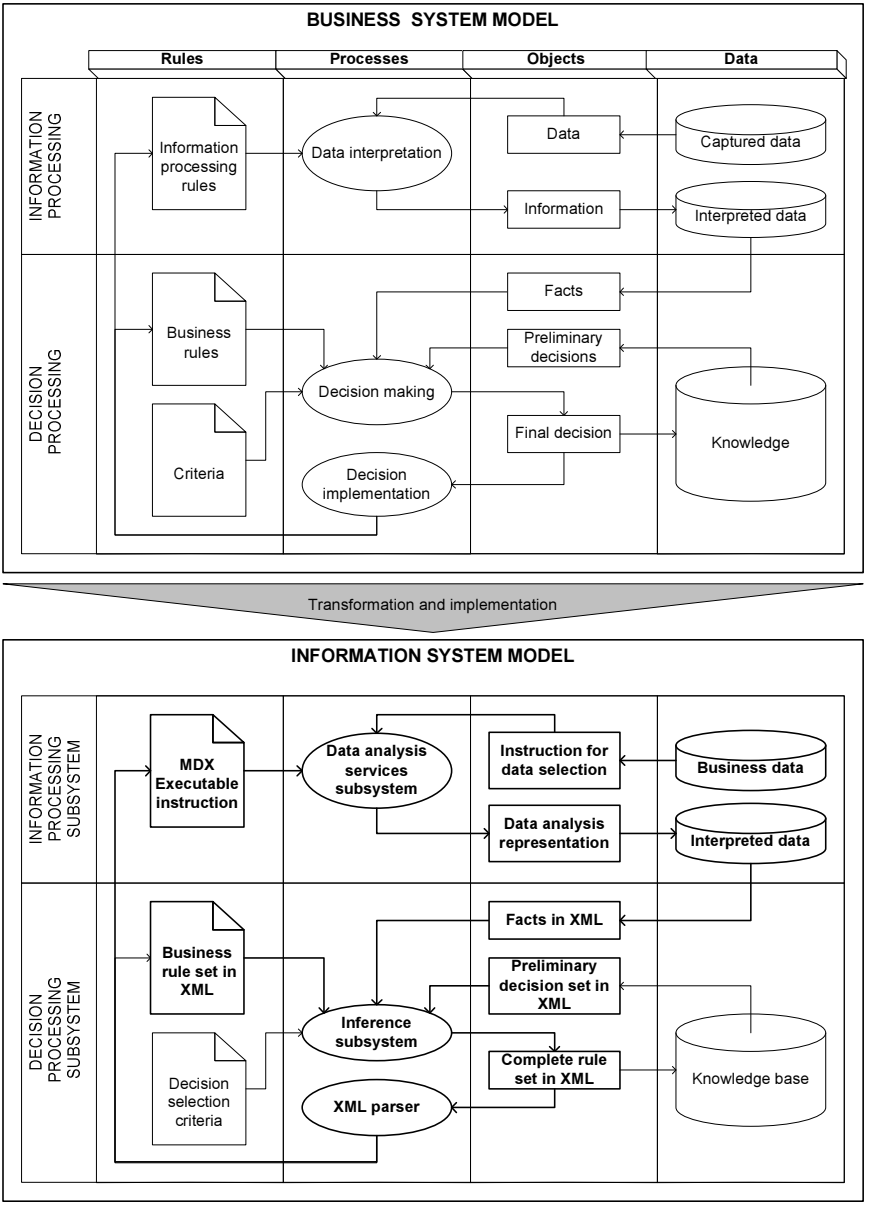


Fig. 3.7. Decision model for intellectualised decision support system

3.3. Method Implementation Schema

This chapter discusses decision modelling and further implementation into the information system. The authors of a scientific publication (Linehan *et al.* 2011) define three main steps involved in decision modelling: considering on what the decision is to be made (e.g. "What is the question to be answered"), a list of the inputs (considerations) required to answer the question, a list of various possible combinations of consideration values and the corresponding outcomes (conclusions) – (e.g. answer the question on "How does the conclusion depend on the considerations?"). We add the additional questions that are important as well: "What is the reason of decision in the scope of all business process?", "How should the solution be implemented to make adjustments needed in the business process?" and "How will it be implemented into the IS?"

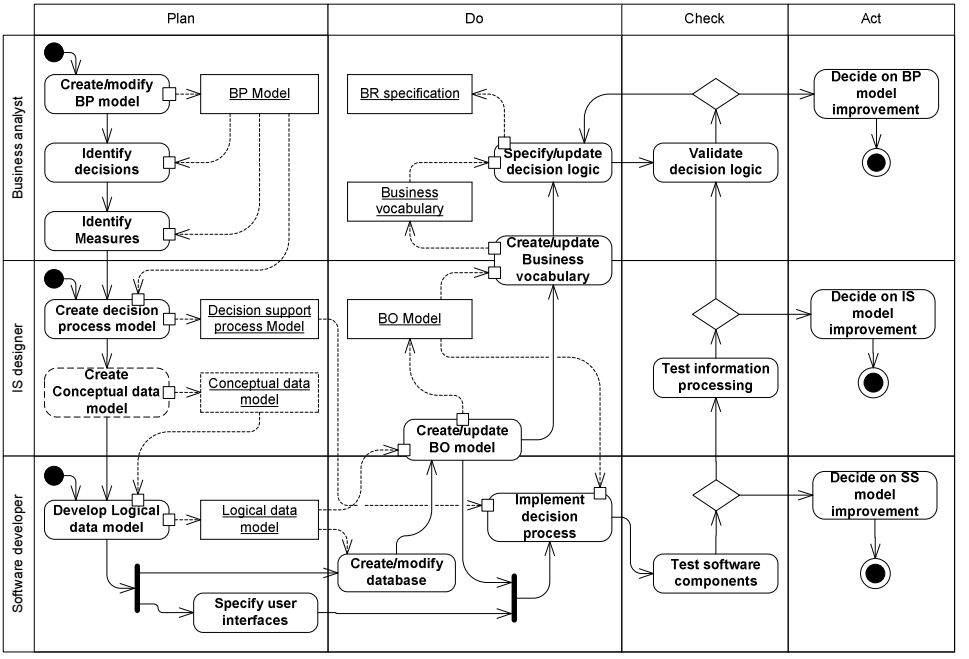


Fig. 3.8. Method schema for implementation of the automated decision process

Further illustration of the proposed method presented in Fig. 3.8 answers the questions given above. The method schema is made with the intention to preserve one of the most appealing properties of decision models: to make

communication more reliable, precise, and unambiguous between different stakeholders, such as business decision owners, policy managers, and business process managers.

It is important to note that implementation of the method requires specific support of the enterprise technological infrastructure used for our experiments and discussed in Chapter 2.3.2. Freely available AdventureWorks database and its warehouse from Microsoft will be used as the source of data for providing examples in this chapter.

An engineering process starts with development of a business process model. A business process model can be formal, developed using BPMN or UML activity diagrams or informal, i.e. a verbal case scenario. Further steps identify core problems necessary to be solved and related decisions to be made to evaluate a core process in the context of business objectives and actions to be taken to adjust a business process by taking corrective actions. Such analysis helps to identify and separate unstable parts of the process. First of all, we usually separate decision logic and express it in a declarative way and replace a set of actions with a single "smart" process guiding further corrective actions in the process according to the business logic implemented in a declarative way.

The step assigned to create a decision process model is directly related to the model of the goals to be achieved by a core business process. Such model should be composed of sub-processes that allow evaluating not immediately-observable data. Sub-processes should be aggregated to produce the consideration as a conclusion of the entire decision process. We consider a decision process model as a workflow composed of sub-processes functioning by a separate isolated fragment of business logic expressed in business rules or a sub-process tree. IBM (ILOG) and OMG (BRG) focus on the use of decision tables for business logic specification. Difficult implementation of business logic and decision process change at run-time is supposed to be a huge limitation of this approach. We prefer declarative rules composed according to the business vocabulary with a direct relation to the business object model. However, the main requirement for a decision process is to be directly executed in the IS at run-time and to use a decision logic equivalent to the business logic specified at a business level and expressed in business rules. The key benefit of decomposing the overall decision into independent sub-decisions is that they can be managed, maintained, and reused independently; including the need of different informational support required. It is important to note for further discussion on the implementation of decisions based on historical and implicit knowledge and involvement of predictive models, an optimization model, and risk based analysis models or other models based on AI involvement techniques such as neural networks, regression, etc.

It is important to mention that such automated decisions should be monitored and adjusted when needed by changing corresponding business rules or starting the process from the beginning.

Some complicated decision models can include rated decision criteria. Such goal can be achieved by assignment of some value meaning importance of the rule. However, for such purpose rule engine which supports weighted rules is necessary. Such assignment of weights can help validating several solution alternatives to derive the optimal decision. We have already discussed such approach in one of our publications (Vasilecas and Smaizys 2006b). However, we have not found any available rule engine yet, which integrates our technological infrastructure.

To integrate the developed information system intuition of a decision, a support model should be separated and provide all the necessary facts for automated decisions executed in the separate rule engine according to the specified declarative logic stored in the centralised business rule repository performed by integrated, decision-making and impact models into the process of a flexible business process design and run-time execution using dedicated software components. The results of the analysis also show that such business decisions are supported by four legs of business decision management: business motivation, business metrics, business logic, and knowledge. In the next chapter we will further examine a role of business rules in IS and decision-making automation.

3.4. Model of Business Logic and Decision Process

In this chapter we perform a representation analysis of conceptual modelling languages and specifications used for decision-making automation based on knowledge (rule) and process model integration. A part of this chapter has been published in one of the latest scientific publication of the author and other researches (Rima, Vasilecas, Smaizys 2011).

BRs are the main source for business logic. In general, a BR was defined in Guide project (Hay *et al.* 1997) as "a statement that defines or constraints some aspect of the business". According to the OMG, the BR resides at the borderline between business engineering and software engineering. The Business Rules Group (BRG) gives another definition using business and information system perspectives: "From the business perspective a business rule is a directive, which is intended to influence or guide business behaviour, in support of business policy that is formulated in response to an opportunity or threat. From the information system perspective a business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure, or to control or influence the behaviour of the business" (BRG 2003). In this thesis we

consider BR as a well-structured statement in a natural language and good vocabulary, which can be used for communication with business people and business analysts. BR is something we can deal with as a business proposition and something that has a meaning in business. Directives are composed of policies, standards and guidelines, where relationship between them can be described as follows: guidelines describe best practices on how to do something; standards describe what the baseline requirements are for various technologies and configurations that are supported in the enterprise; and policies describe the required actions to take and why.

A decision process belongs to a kind of business processes. Various authors consider a decision process as a guided workflow of separate processes executed until the business process reaches the goal. Other authors treat a decision process as an activity where some decision are made and some further action executed or an event raised. According to Ciuksys and Caplinskas (2007), a business process is a partially ordered set of linked activities that create value by transforming an input into a more valuable output. Both input and output can be artefacts and/or information, and the transformation can be performed by human actors, machines, or both.

Gudas and Skersys (2005) proved that business rules can be represented using a special set of natural language templates and used for development of the UML class diagram supporting the informational model used for the related business process. Such approach is widely adopted by other authors as well. Moreover, we consider that some formal or semi- formal language which could be more suitable for later automated meta-model based model transformations is necessary.

Further in this chapter we have adopted and extended the research made by Muehlen *et al.* (2010) to compare a set of the existing rule and process modelling languages used for our investigation. To group the comparison criteria initially defined in Bunge-Wand-Weber representation theory, we selected six perspectives described in Chapter 2.2. The same conceptual modelling perspectives are used for the development of three layer framework introduced further in Chapter 4.

For comparison we have included such rule modelling languages as: Simple Rule Markup Language (SRML), the Semantic Web Rules Language (SWRL), the Production Rule Representation (PRR), and the Semantics of Business Vocabulary and Business Rules (SBVR) specification. We have also included some process languages that appear as graph-based languages (e.g. BPMN, EPC), net-based languages (e.g. Petri nets, flow nets), and workflow programming languages (e.g. Business Process Execution Language (BPEL)).

Table 3.2. Comparison of rule and process languages according to the views of conceptual modelling and modelling perspectives

	Rule specification (language)					Process specification (language)					
Language	SRML	SBVR	PRR	SWRL	OCL	UML	DFD	CPN	EPC	IDEF3	BPMN
Year/version	2001	2006	2007 v1.0	2006 v0.6	2008 v2.0	2005 v2.0	1968	1981	1992	1995	2004 v1.0
Process											
Action (process)	X			X	X	X	X	X		X	X
Sequence						X	X	X		X	X
Information (Object)											
Thing	X	X		X	X	X	X	X		X	X
Kind						X					X
Structure											
Property	X	X	X	X	X	X		X	X	X	X
Class		X	X	X	X	X		X			X
Events											
Event	X	X			X	X		X	X	X	X
Conceivable event space											
Lawful event space											
External event						X					X
Internal event						X					X
Well-defined event											X
Poorly defined event											X
Transformation	X			X						X	X
Lawful transformation	X		X	X							X
Coupling										X	X
Acts on	X			X	X			X			X
States											
State	X				X	X		X	X	X	
Conceivable state space	X				X						
State law	X	X	X	X				X	X		
Lawful state space	X		X					X			
Stable state									X		
Unstable state								X			
History											
Resources											
System	X	X	X		X	X				X	X
System Composition		X	X			X				X	X
System environment											X
System structure					X	X	X			X	
System decomposition		X								X	X
Level structure									X	X	X
Sub-system						X	X				X

The analysis results are presented in Table 3.2. The results demonstrate that most languages support different aspects of previously defined perspectives. There is no single language to cover all the related perspectives necessary for the rule and process combination. The main task on implementing business decisions in IS is the alignment and integration of business logic and business processes. From the analysis of the comparison results we expect that the most common combination for our purpose is SRML + UML (CD, AD, STD) combination. We assume that UML has corresponding XML schema specification and can be used together with SRML for XML based transformations described in further chapters.

3.5. Conclusions of Chapter 3

The proposed method meets flexibility criteria defined in Chapter 1.4.7. The experience gained on experimental study described in the next chapter leads to the conclusion that the proposed three layer framework allows improvement of the method by involvement of additional models and separate development of additional parts related to decisions in the business process and later implementation into the information system. However, we have experienced that every additional model implemented leads to the increasing complexity of the system and results in difficulties in testing and management of unexpected behaviours of the integrated system components at run-time. Management of complex infrastructure and reliability required to support decision models according to the method proposed after deployment of a decision process into the production environment of the information system should be also at the concern by expecting run-time problems, which should be tested and validated on the check phase according to the introduced method.

A clear positioning of the models involved in the process of a decision design and implementation process according to the proposed intellectualised IS engineering lifecycle creates clear directives on development of the required structure of the models necessary to achieve the required complexity and maturity of the selected decision process implementation into the BIS. The proposed evolutionary approach of a development process minimises development time and effort, and allows maximum reuse of separately developed models according to the specific purpose of the expected flexibility in the final implementation of the automated decision process in the information system.

4

Application of the Proposed Method

This chapter presents a set of experiments on the application of the method developed according to the introduced three layer framework to examine and demonstrate its suitability for different tasks of decision-making process development and further implementation into the information system. Parts of the chapter have already been published in scientific publications of the author and other researchers (Vasilecas and Smaizys 2008a; Vasilecas and Smaizys 2007b; Vasilecas and Smaizys 2005d; Vasilecas, Smaizys *et al.* 2009; Vasilecas and Smaizys 2010).

The first experiment demonstrates application of the method applying a parameterisation approach. For this reason we express business logic in rules using decision tables. Business rules from decision tables are later implemented as software parameters stored in the relational database. Applying the proposed approach, we finally get software where parameters are used to implement and manage business logic in the software system as the result of the engineering process.

A second set of experiments reviews design time and run-time model transformations for the implementation of business rules at a business system level models and later use of declarative logic models on information system level

models and a software system to allow adaptation of software system behaviour to the changed business rules.

4.1. Experimental Investigation on Parameterisation Approach

The experiment was conducted participating in the real project for development of the information system used to control and manage access of people and transport vehicles with cargo entering or leaving Klaipeda Seaport.

For the illustration of a parameter driven approach presented in our publications (Smaizys and Vasilecas 2009b; Smaizys and Vasilecas 2008), we have analysed decision automation based on BR represented as DT for the use by separate components servicing in a business logic layer dedicated for selection of the action and execution of the related executable code in the application layer or database layer of software systems.

The idea of the experiment was to investigate design-time metamodel-based transformations of decision tables (DTs) used for representation of a decision model expressed as business rules into a logical model of the relational database schema used as a parameterised model of business logic in the functioning software system. A parameterisation task of the SS is solved by automatically building SS configuration parameter set and executable components of n-Tier architecture according to the set of rules in DTs, previously captured by a business domain analyst.

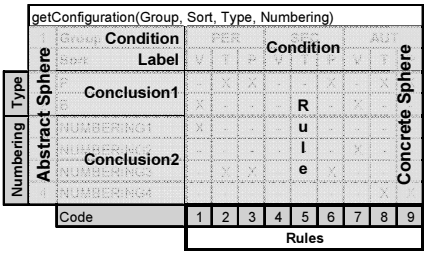
4.1.1. Decision Table Based Business Rule Representation

The experiment was performed according to the method discussed in the previous chapters. The experiment started with the representation of a particular set of requirements captured by the business analyst with extraction of business logic into the separate model using decision tables. Decision tables can quickly highlight where an outcome or a decision is missing. We have limited each decision table to make one particular type of determination.

The extracted rule set representing a decision model based on MOF/UML metamodel is represented as a decision table in Fig. 4.1. The business rules here represent logic of a business process, which will be later implemented into the configuration parameters stored in the final software system.

A rule set in a decision table consists of subjects, properties and variables, which are represented by subject atoms and property atoms. Furthermore, a rule set consists of an ordered composition of condition labels. These condition la-

bels group the conditions of the rule set in the sets of exhaustive and mutually exclusive conditions.



getConfiguration(Group, Sort, Type, Numbering)

			Group	PER			SEC			AUT		
		2	Sort	V	T	P	V	T	P	V	T	P
Type	1	B		X	-	-	-	-	-	X	-	-
	2	P		-	X	X	-	-	X	-	X	X
Numbering	1	NUMBERING1		X	-	-	-	-	-	-	-	-
	2	NUMBERING2		-	-	-	-	-	-	X	-	-
	3	NUMBERING3		-	X	X	-	-	X	-	-	-
	4	NUMBERING4		-	-	-	-	-	-	-	X	X
		Code		1	2	3	4	5	6	7	8	9
				Rules								

Fig. 4.1. The nodes of DTs metamodel and business rules in decision model

Each condition is a logical formula that refers to the domain atoms and variables of the rule set. The actual rules of the rule set are an ordered conjunction of conditions, such that a rule contains at most one condition of each condition label. It should be noted that not every conjunction of conditions is necessarily meaningful. In other words, it might be the case that a specific condition is only meaningful in combination with other specific conditions. In addition to conditions, a rule refers to one or more conclusions.

Subjects, properties, variables and values:

	ID	Code	Name	Description, processing requirements
Group	1	PER	Person	Entrance through pedestrian gates allowed
	2	SEC	Security	All security operations allowed
	3	AUT	Auto	Entrance through automobile gates allowed
Sort	1	V	Visitor	Pass used for one entrance and exit.
	2	T	Temporary	Pass used for multiple acces for temporary period
	3	P	Permanent	Pass used for unlimited access
Type	1	B	BarCode	Pass issued using BarCode card
	2	P	Proximity	Pass issued using Proximity card

	ID	Code	StartFrom	FinishOn	Current
Numbering	1	NUMBERING1	PB0001	PB9999	PB0000
	2	NUMBERING2	AB0001	AB9999	AB0000
	3	NUMBERING3	PP0001	PP9999	PP0000
	4	NUMBERING4	AP0001	AP9999	AP0000

Fig. 4.2. DTs subjects, properties, variables and values

An example of a set of rules is presented in Fig. 4.1; and a list of subjects, properties, variables, and values (Fig. 4.2) used for security system pass issue process logic representation in form of DT. According to the rules displayed, the selection of numbering and type of the pass can be processed according to the group and sort of the pass. As it is seen from the example, DTs based configura-

tion management not only provides better understanding of the logics, but it should also allow addition of the attributes for condition labels and conclusions. This is important for a later generation of the configuration data tables and filling them with configuration values, after the logics represented by the rules has changed. A set of rules for better understanding by business people can be expressed as a set of sentences in Horn logic.

1 IF Pass Group IS Person AND Pass Sort IS Visitor THEN issue Pass Type BarCode AND use NUMBERING1
2 IF Pass Group IS Person AND Pass Sort IS Temporary THEN issue Pass Type Proximity AND use NUMBERING3
3 IF Pass Group IS Person AND Pass Sort IS Permanent THEN issue Pass Type Proximity AND use NUMBERING3
4 IF Pass Group IS Security AND Pass Sort IS Visitor THEN issue Pass is Not Available
5 IF Pass Group IS Security AND Pass Sort IS Temporary THEN issue Pass is Not Available
6 IF Pass Group IS Security AND Pass Sort IS Permanent THEN issue Pass Type Proximity AND use NUMBERING3
7 IF Pass Group IS Auto AND Pass Sort IS Visitor THEN issue Pass Type BarCode AND use NUMBERING2
8 IF Pass Group IS Auto AND Pass Sort IS Temporary THEN issue Pass Type Proximity AND use NUMBERING4
9 IF Pass Group IS Auto AND Pass Sort IS Permanent THEN issue Pass Type Proximity AND use NUMBERING4

The next section describes a method applied for decision process implementation into the software system using parameterisation and decision logic model transformation into the logical data model and a set of configuration parameters.

4.1.2. Parameterisation Based Decision Model Implementation

The method applied for implementation of business logics into the software system configuration is based on the method schema developed according to the proposed framework presented in Fig. 4.3. The schema displays a general structure of models used for business rule based information system parameterisation.

According to the schema at the first step, business rules should be captured in a business system usually interviewing business people, analyzing business processes, studying documentation and legal regulations. Captured informal business rules are placed into repository and represented in a form of decision tables for later representation of configuration rules. One part of a decision table (DTs) contains conditions that can be linked to create a rule, and the other part contains actions that are related to the rules. Once the rules are defined in DTs tool, it is necessary to perform automated analysis, addition of missing rules, and to remove the ones that are redundant or contradictory.

Later the decision tables are directly implemented into the software system database, using metamodel based transformations and automated logical data model generator, and prepared for further transformation into the information system rules. Information system rules are implemented according to the architectural context into the software system components of an information system (business object classes) in a business logic tier of n-Tier architecture.

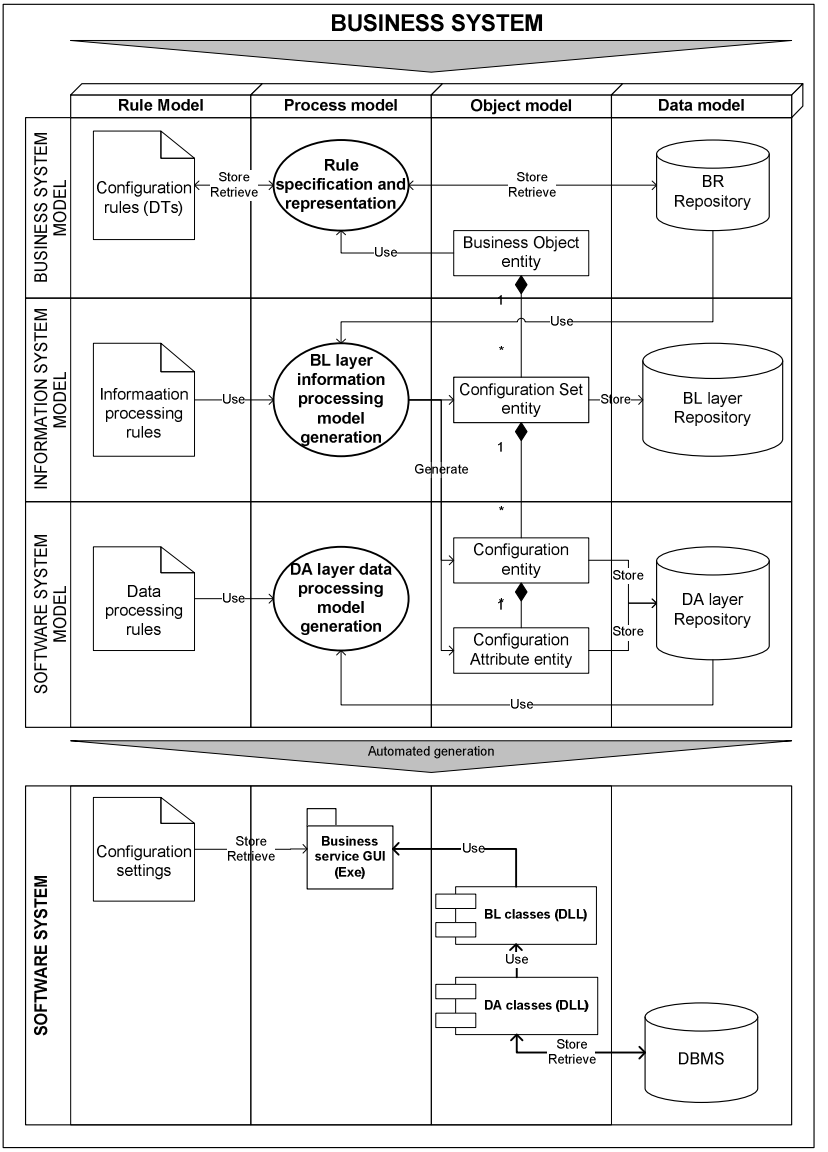


Fig. 4.3. Business rule based decision model implementation

After deployment of the rules into the database, a logical data model is to be reverse engineered using a special design tool back into the software system model and later used for building a business layer of a software system model. This allows automated access and manipulation from information or a business

system model, implementing business objects automatically into the separate software system module later on.

Summarising the method, shortly presented in this section, we define three main consequent tasks necessary to be performed: DT based BR representation, metamodel based transformation of DTs into the configuration tables of the software system database and automated software system code generation of the components represented by n-Tier architecture of the software system.

4.1.3. Rule Model Transformation and Implementation

For further business logic implementation into the configuration of software system, we have used modified MOF/UML DTs metamodel presented by Goedertier and Vanthienen (2005).

Moreover, we needed to create ER metamodel. The metamodels were used for specification of transformation instructions required to produce ER model from DTs.

The models presented in typed graphs were transformed into XSD schemas and mapped accordingly. All the mapping functions were collected into XSLT transformation schema and the transformation executed using XML parser according to the schema presented in Fig. 4.4.

An example of the result of the presented transformation into the ER model is provided in Fig. 4.5. Such ER model was used for the creation of the database and filling it with the data entered in DTs.

The rule values from DT are inserted into the database using generated sentences of SQL data manipulation language (DML); and the physical relational database is created according to the produced SQL data description language (DDL) constructs. Having executed the generated CREATE TABLE, JOIN and INSERT instructions, we get a database schema according to the ER model with configuration tables filled with the data from DT's. The generated database is used for automation of guided decisions on a business object in the scope of the analysed business process.

Further, we will continue on automated software system code generation for implementation of the automated decision process. As the result from previous steps we have business process logics represented as DTs, a data model in ER and data values filled into the database. This allows generation of IS process code in the SS according to the information processing rules presented in Fig. 4.3. At the next step we create GUI interface component on the presentation tier of SS represented as a business service (presentation) tier and supporting classes in a business logic (BL) tier used for automation of the collection and entity operations like save, get, delete, filter, etc.

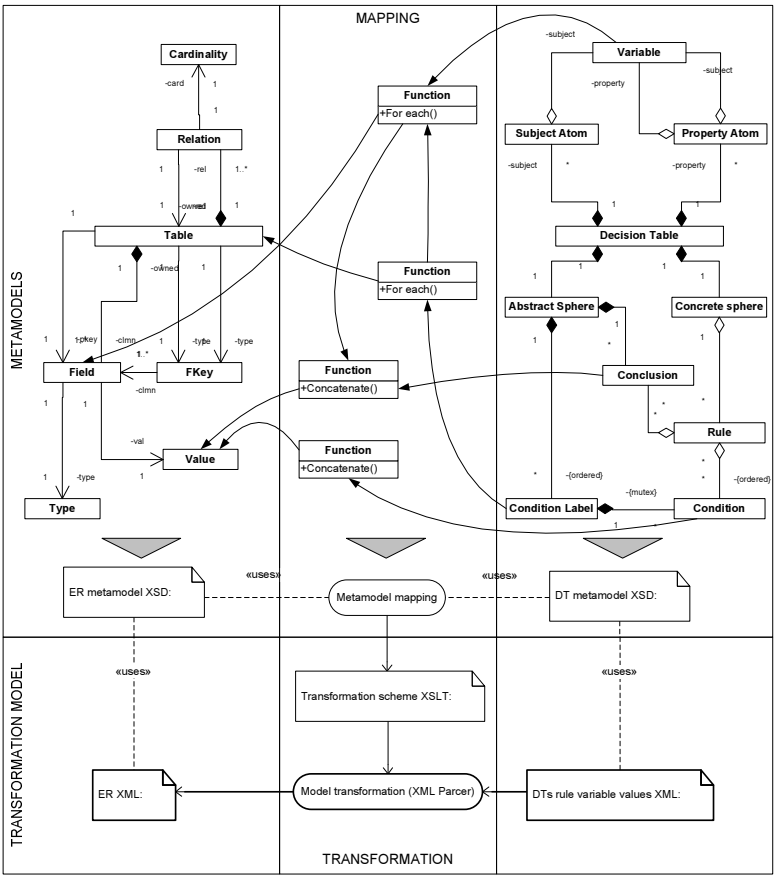


Fig. 4.4. DT model transformation into the ER model

The access to the database is performed through the generated DAO classes in a data access (DA) tier according to the data processing rules stored in a predefined template. To produce separate parts of the software system code, we reverse an engineer configuration model and enrich it with the missing information structures to produce a model of information structures in business terms. The result is a business object model which is later implemented as a separate software class and used for additional business logic specification and implementation of the business logic already implemented into the set of configuration tables.

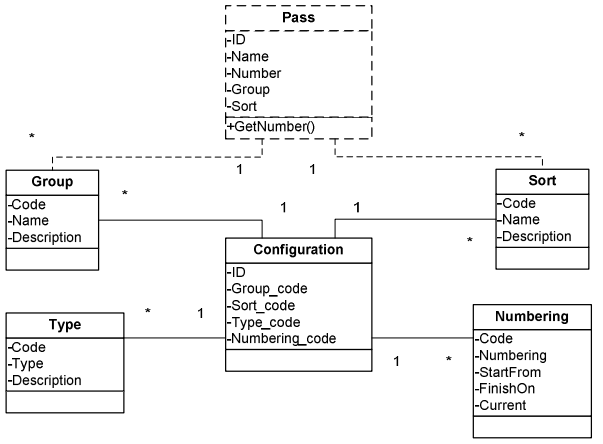


Fig. 4.5. ER Model of configuration set transformed from DT

Such approach of software engineering allowed semi-automated implementation of the business system logics into the software system by automatically building a decision model acting according to the parameters filled into the previously built database used as configuration storage. Below we give an example of automatically generated software system code used to get numbering schema for the "Pass", depending on the Group, Type and Sort values according to the configuration values delivered as Configuration object entity.

```
001  Imports BL.CollectionClasses
002  Imports BL.EntityClasses
003  Imports BL.FieldClasses
004  Public Class frmPassEdit
005      Inherits System.Windows.Forms.Form
006      Private _PassEntity As PassEntity
         ...
007          Me.cbGroup.DataSource = GroupCollection()
008          Me.cbGroup.ValueMember = GroupFields.Code.Alias.ToString()
009          Me.cbGroup.DisplayMember = GroupFields.Name.Alias.ToString()
         ...
010      Private Sub btnSave_Click _
011              (ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSave.Click
012          Dim Pass As PassEntity
013          Dim Numbering As String
014          Dim Err As Boolean = ConfigurationEntity.getConfiguration _
015              (cbGroup.SelectedValue, cbSort.SelectedValue, Pass.Type, Numbering)
016          If Err then Exit Sub
017          lblType.Text = Pass.Type.Name          'Get Pass Type Name through Entity relations in BL
018          Pass.Number = NumberingEntity.getNumber(Numbering)
019          Pass.Save
020      End Sub
021  End Class
```

The automatically generated method getConfiguration is stored in a business logic (BL) tier and applies a method getConfigurationEntity of ConfigurationEntity class, which uses the generated DAO classes from a data access (DA) tier to get the data from the database (line 27). In the same way a software system code for creation and edition of the "PassEdit" object was generated, providing input selection of the Group and Sort combo boxes in the form, etc.

```
021  Imports DA.DaoClasses
022  Namespace BL.EntityClasses
023      Public Class ConfigurationEntity
             ...
024          Public Shared Function getConfiguration _
025                  (ByVal Group As String, ByVal Sort As String, _
026                   ByRef Type As String, ByRef Numbering As String) As Boolean
027              Dim Configuration As ConfigurationEntity = GetConfigurationEntity(Group, Sort)
028              If IsNothing(Configuration) Then Return False
029              Type = Configuration.Type
030              Sort = Configuration.Sort
031              Return True
032          End Function
033      End Class
034  End Namespace
```

The main difference of such approach is that for example ILOG uses separate Rule Execution Server, and here we dedicate this functionality to the separate software system component by automated generation of business layer objects and direct deployment of BR represented in DT to the Application database. This approach is simpler and can be easily understood by IT people; and, moreover, it does not require the use of separate rule execution server and uses automatically generated dll's with business layer objects providing execution of business logic according to the rules stored in DBMS. The same shared dll's in a software system layer are used for integration of several applications in the enterprise to implement shared logics. We argue that structured rules in DTs representing decisions required are simple, and it is sufficient to automate development of a querying process in the software system according to the BR managed at the business system level. Although for more sophisticated and intelligent decision-making methods based on statistical analysis (e.g. clustering), neural networks, risk models, fuzzy logic, etc. the use of separate rule execution server is unavoidable.

Summarising the parameterisation approach, we arrive at the conclusion that the approach to the engineering of a guided decision process implementation into the information system is possible. The main steps to be performed are the specification of decision logics expressed as the rules represented in decision tables, decision table conversion into the logical data model and implementation into the relational database by filling with configuration values from the decision

table; and generation of a business object model for later implementation as a class library component on the separate tier of a software system, where decision processes are implemented as separate dedicated methods of the business object class.

4.1.4. Results of Experiment

Summarising the experience acquired, it may be stated that decision tables are very common and well understood by system engineers; however, they are difficult to read for some business users. Therefore, we have designed transformations used to extract the business rules implemented into Software system Configuration according to the currently entered configuration parameter values and represent them in a plain text. Similarly, the decision tables were transformed into the Horn clauses and used as the logical control code uploaded into the security system controllers.

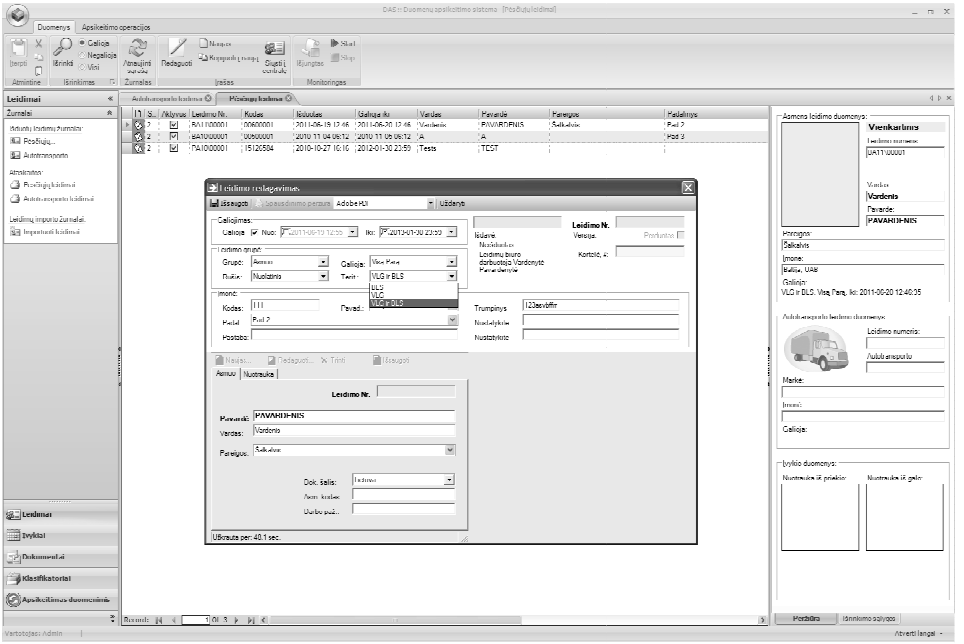


Fig. 4.6. Intellectualised interface of port control system

However, the main disadvantage of the parameterisation approach is tight implementation of information structures used for representation of business logic into the logical model of the relational database. This dramatically limits flexibility of the system at run-time and causes problems in later modernisation

of the system when changes in business logic structures are needed. Anyway, such approach is common for simple projects especially based on integration with PLC controllers where the structure of the controller logic and its functionality is stable and provides the main limitation during implementation of business logic into the information system.

4.2. The Experiment on Run-time Transformation Approach

The goal of the experiment is to examine suitability of the proposed method for the design and implementation of run-time transformations that enable dynamic change of the process according to the business logic expressed in business rules.

Model transformation-driven approach has already been presented in our publication (Vasilecas and Smaizys 2007a), where we introduced a method for the development of XML based web applications, using BR model transformations into the design specifications used for further development and/or validation, interpretation and transformation of BR sets into the executable code.

According to the method, first of all, we define a business process. For illustration we will use a simple analysis process and create a verbal analysis process model by describing a case scenario. It is necessary to analyse production process of sales to determine product categories that are sold without achieving objectives set by the management. The current objective is to have sales not less than 5000 units per quarter. Firstly, the business problem is to identify "bad sales" by product category and provide such information for later decisions to manage the situation. Therefore, we see a solution of the business problem by developing and implementing a flexible analytical process and providing the results as the report.

Further, we need to identify dynamic and static parts of the process. The most vulnerable part for change in business environment is a business rule derived from the objective that "If units by category sold are less than 5000 per quarter, then these are bad sales". It is possible that the amount set by the objective can be changed at any time. We determine that it is necessary to separate and develop a part constraining further analytical process by separation of process logic and using run-time transformations to enable additional flexibility.

A less dynamic part is related to the analysis rule itself "Select all sales by product category from the last quarter, where sales are bad" and can be directly translated into the analysis rule at IS level, which uses OLAP model to get the data needed directly from the warehouse. At this stage we assume that we have working warehouse and OLAP model implemented and will skip some steps

needed to perform according to the method schema. The report preparation process will be created at SS level and is not discussed in this study.

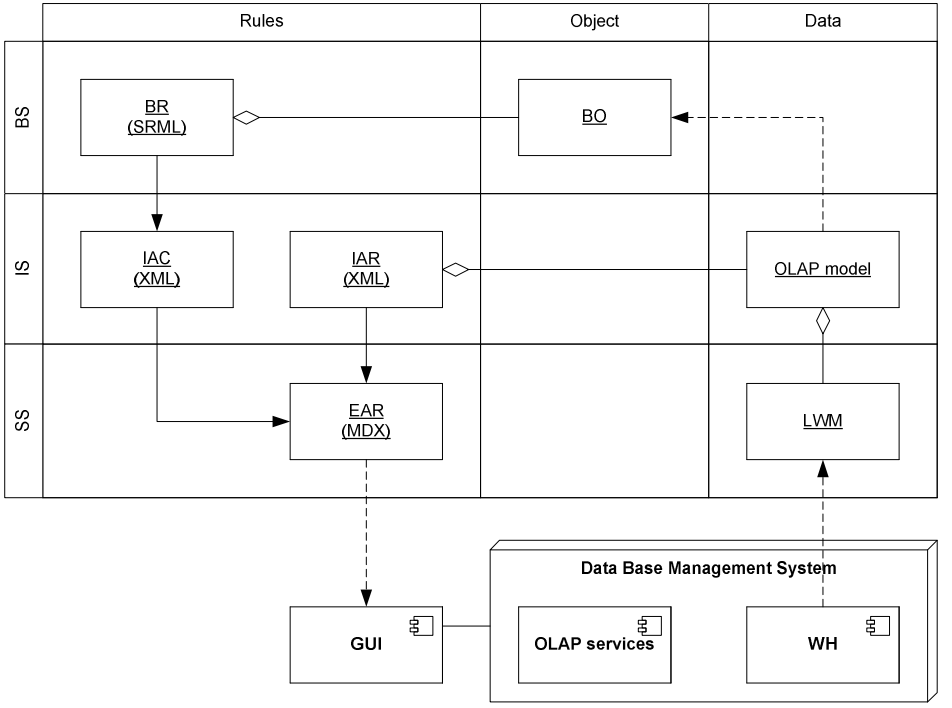


Fig. 4.7. Model transformation based implementation model

At the next step, we will develop the analysis process implementation model presented in Fig. 4.7, which is composed according to the framework. A set of models organised into the rule, object, and data model views is used. Arrows represent model transformations and dotted arrows represent implementation processes of software engineering or reverse engineering as displayed in the case of OLAP model reengineering in the data column.

We have identified a set of models necessary to implement informal business rule (BRi) and informal analysis rule (ARi), which are not presented in the schema:

- **BR (SRML) is a** formal business rule model in SRML language. The model should be validated according to the corresponding SRML metamodel XSD schema. The terms and vocabulary used in the model are determined by the business object model (BOM), which is reengineered from the warehouse OLAP schema. A warehouse data model should be

composed using concepts in business terms so that the terms used here are similar to the business terms in a natural language;

- **IAC (XML) is** formal information analysis constraint (MDX model) in XML language. The model should be validated according to the corresponding MDX metamodel XSD schema;
- **IAR (XML)** is a formal information analysis rule (MDX model) in XML language. The model should be validated according to the corresponding MDX metamodel XSD schema. The model is restricted to the use data structures implemented in the schema of corresponding OLAP cube;
- **EAR (MDX)** is an executable analysis rule in MDX language.

We also need to use a set of manual and metamodel mapping based transformations: BRi → BR(SRML), ARi → IAR(XML), BR(SRML) → IAC(XML), IAC(XML) → IAR(XML) and join them producing executable EAR in MDX language. The executable MDX instruction will be used in GUI which is engineered separately and is not in the field of interest in the description of this experiment. For the development of transformations we use a method described in Chapter 2.3.1. Metamodels of the models in SRML and MDX with the corresponding transformation in XSLT language from the model in SRML language into the MDX was presented in our scientific publications (Rima and Smaizys 2007a; Rima and Smaizys 2007b).

Here we present MDX code which is the result of run-time transformation executed at GUI to get all the data from DBBMS needed and provide for the composition of the report.

```
With Member [Measures].[BadSales] as
'iif (([Product].[Category].CurrentMember,
       [Category Sales])<5000, "Bad sales", "")'
Select {[ Category sales], [BadSales]} on columns,
        [Product].[Category].members on rows
from [sales] where [Time].[Quarter].[Q3, 2010]
```

Evaluation of the results of the experiment demonstrate that by separation of a dynamic part of the analytical business process we can achieve additional flexibility avoiding involvement of engineering process in the case of change in business logic. For example, a current business problem solved by the development of the analysis process allows change of a business rule separately stored in the rule repository directly accessed by business people. The change of such a kind of objective will not require any IS modification. All the analytical processes that are constrained or guided by this particular rule will adapt to the change automatically and give corresponding results in the reports or wherever used.

Similarly, we have made experiments to generate XForms and YAWL language code as described in our scientific publications (Martisius, Vasilecas, Smaizys 2010; Vasilecas, Smaizys *et al.* 2009). The latest publication also discusses the implementation of the Risk analysis based decision impact model.

4.3. The Experiments on the Use of Declarative Logic Models

The series of experiments were conducted participating in the Project "Business Rules Solutions for Information Systems Development (VeTIS)" at the Faculty of Fundamental Sciences of Vilnius Gediminas Technical University and the Faculty of Mathematics and Natural Sciences of Klaipeda University to examine the use of declarative logic models and create implementation models to enable business rule enforcement and reactive or proactive decision process implementation into the IS and decision model implementation or execution at run-time.

The following objectives of the experimental program have been pursued: 1) examination of run-time transformations of decision models into the software code or a kind of executable model; 2) exploring whether the implementation models allow direct execution of a decision process without development and execution of transformations that produce an executable code.

The results of the experiments are published in the scientific publications prepared by the author of the thesis (Vasilecas and Smaizys 2005c; Vasilecas and Smaizys 2005d; Vasilecas, Smaizys *et al.* 2009; Vasilecas and Smaizys 2010; Smaizys *et al*. 2010).

The results of the conducted experiments demonstrate that development of IS with the use of decision models based on declarative logic requires separate components for business rules storage, management, and execution. For the experimental purposes we used several rule engines based on forward chaining algorithms. We started our experiments with Jess rule engine and finally stayed with the rule engine provided by IBM (ILOG). During the first experiments much time was allocated for trying to specify business rules in SRML language and translate them into the language used by rule engines. Having transformed the results of inference back into SRML, execution of transformations was performed according to the implementation model expressed as a rule transformation schema.

We have developed several transformations to illustrate the use of the Rule model. The example transformation of a business rule set specified in SRML language into the MDX instruction to allow analysis model adaptation according to the business behaviour examined by business rules is presented in our scientific publications (Rima and Smaizys 2007a; Rima and Smaizys 2007b). The

example transformation of a business rule set specified in SRML language into the XForms specifications for dynamic generation of the adaptable user interface, providing business rule based form data evaluation, is presented in our scientific publications (Vasilecas and Smaizys 2010; Brazinskas and Smaizys 2007). The last experiment described as the example in Chapter 3 is provided to illustrate how following the proposed method we can use separate infrastructure for rule model development and execution and simplify IS engineering process by focusing on development of information support and implementation models with the involvement of additional decision impact or evaluation models if required. The use of predictive power of AI algorithms and involvement of implicit knowledge extends the possibilities of a decision-making process.

4.4. Evaluation of the Results

In the previous chapters we have described series of experiments performed for the research of the proposed framework and method for decision design in flexible business processes and automated decision-making implementation into the related BIS. As it has already been mentioned, the main purpose of the experiments was to prove that the proposed framework based method and models with their transformations allow flexible development and implementation of decision models according to the introduced framework application approaches. It is important to notice that the techniques used in the design of the proposed methods for decision-making process modelling and implementation are based on different assumptions and approaches such as spontaneous change of decision-making culture and responsibility distribution across the enterprise.

The proposed method covers three separate approaches of decision automation implementation: parameterisation, run-time transformation, and the use of declarative logic. Such segregation enables better flexibility at design time due to availability to choose tools and development tools to suite the specific situation. The experience gained during the experiments according to the selection of the best suitable method is presented in Table 4.1.

It was determined that in cases when it is required to have high flexibility during run-time and short time is available for changing implementation, the declarative logic based approach should be applied. On the contrary, if there is a static rule structure, it is better to stay on the parameterisation approach, which does not require complicated infrastructure. It is difficult to track history of business logic used for a specific decision case if transformation or declarative logic approaches are used. If the reason is important, parameterisation should be used instead. Parameterisation allows uncomplicated implementation of keeping historical records for the particular decision logic used for the particular decision

case or per time period. Tracking versions of declarative rules is more complicated but still available.

Table 4.1. Evaluation of the proposed method application suitability

Evaluation criteria	Value	The approach of application of the proposed method		
		Parameterisation	Real-time transformation	Declarative logic
Structure of business Rules applied	Static	+	+	+/-
	Dynamic	-	+/-	+
	Unknown	-	+	-
Automated decision process model by complexity	Controlled	+	+	+/-
	Reactive	+/-	+	+
	Proactive	-	+/-	+
Time available for change implementation	Short	+/-	+	+/-
	Limited	+/-	+	+
	Unrestricted	+	+	+
Importance of version tracking	Unimportant	+	+	+/-
	Important	+	-	+/-
Decision error toleration and resistance	None	+	+/-	-
	Low	+	+	+
	High	+	+	+
Number of rules used for decision	Few	+	+	-
	Average	+	+/-	+
	High	-	+/-	+

However, it is not a trivial task to determine which particular rule determined a particular decision. In case of transformation, such control is complicated because of hidden logic implemented into transformation rules. The selection of the most appropriate method is also related to the number of business rules used for a particular decision problem. Structured rules using DT can be effectively controlled if there are less than 10-20 rules. In case the amount of rules increases, there is the only way of using declarative rules executed in rule engines at the run-time.

To sum up the experience gained, it may be noted that there is no universal method or tool for decision automation available. That is the main reason why we needed to compose strong technological inventions from both IBM (ILOG) and Microsoft to cover decision models discussed in Chapter 3. The method is not limited to support decision types according to the reviewed classification

schema analysed in Chapter 1. For different classes of decision tasks a specific set of the models included in the decision model structure is necessary. Involvement of implicit knowledge in the method can be supported by using Microsoft tools for the data mining model development, adjustment, and testing. Data mining and multidimensional analysis models according to the proposed method use DMX and MDX language constructs. The proposed method is also adjusted with the principles of the intellectualised BIS development and supports the approaches defined in Chapter 2.4.

Advantages and disadvantages of the proposed framework based method are summarised by providing SWAT analysis matrix in Table 4.2.

Table 4.2. SWAT analysis of the proposed framework based method

Strengths	Weaknesses
• Separation of rapidly changing business system models at an early design phase. • Detailed and flexible model composition according to the proposed framework. • Availability to extend the proposed method by development of new automated and semi-automated transformations and models.	• A large amount of required models and unspecified additional engineering tasks not covered by the method. • Weak support for model design and transformation by integrated software development tools. • Complicated testing of complete automated decision model on runtime.
Opportunities	**Threats**
• Decisions implemented applying the proposed method allow immediate changes in business logic by direct modification of business rules. • Development and integration of the selected models can be performed applying other methods. • Use of metamodel based model design-time and run-time transformations.	• There is a risk of unmanaged runtime interaction of separate models implemented in dedicated architectural components. • Incompatibility of infrastructural components used for execution of separate models. • Limited responsibility for automated decisions by business people responsible for the rules guiding a business process.

To evaluate the advantage of the proposed approach, during the last experiment we performed an alternative engineering process based on traditional requirement engineering. The results of the detailed evaluation of the method are presented as a comparison with the alternative traditional methods and submitted in Table 4.2. Cases 1, 3, 5 in the Table represent traditional engineering whereas

Cases 2, 4, 6 represent framework based development. Cases 5 and 6 represent simple business logic change without involvement of new measurements or logical constructs, and Cases 3 and 4 represent change of business logic which includes modification of the underlying data model and related structural changes in the models.

Table 4.3. Comparison of the results of decision process automation with the traditional and framework based methods

Scenarios / Activities	Initial decision automation		Implementation of business logic change		Simple business logic change	
	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
Requirement specification	90	48	3	2	1	0
Business process modelling	0	12	0	1	0	0
Decision support model development	0	8	0	1	0	0
Data model development and deployment	31	26	2	1	0	0
Model transformation development	0	29	0	2	0	0
Business vocabulary and business rule specification	0	20	0	1	0	1
Software interface design	8	8	0	0	0	0
Coding and implementation	64	21	21	4	1	0
LOC written	1439	332	256	17	12	0
LOC generated	236	2467	37	65	0	0
Testing	4	4	2	2	0.5	0.5
Total time spent	193	172	26	12	2	1

Comparison of the results demonstrate that the automated decisions developed according to the proposed method are modelled and implemented into the IS software at least about 10% faster; however, the main advantage was gained by faster modification and reuse of previously implemented decision models which allow about 40% acceleration compared to the traditional requirement driven IS software engineering methods. Moreover, the changes of decision

logic, which do not require introduction of new constructs, necessary for guided automated process, are allowed by changing business rules without running any engineering process at all. The results also identify that the code created by transformations usually provides more lines of code (LOC). However, such parts of the code are better structured and usually changed by repeating transformations without involvement of programmers.

4.5. Conclusions of Chapter 4

The present experimental research demonstrates several advantages for using the parameterisation approach which, however, misses flexibility related to the fixed structure of decision logic implemented directly into the configuration tables of the relational database. The main advantage of such approach is a traditional approach to software engineering, which is usual for developers. The proposed improvement of the engineering process related to the use of decision tables and a set of transformations provides tools and processes for easier understanding of a decision process and decision process automation. Testing and validation of such decision process are easy to perform and are supported by traditional tools implemented in the existing IDE environments.

The use of executable declarative logic allows development and implementation of more sophisticated reactive and proactive decision processes. The decision logic according to the approach is separated and stored into the separate repository. This allows direct involvement of business people into the process of decision logic management and real-time information system adaptation to the changes in business rules stored in the central business rule repository. Applying this approach business people are provided with direct control on the logic of automated decisions; however, there is a big disadvantage in complicated testing and business logic evaluation procedures needed.

General Conclusions

The present study aims at contributing to a better understanding of the business decision process, continuous knowledge reuse and adaptation of the decision logic according to the information collected from running business processes. The following results were obtained during the research and the following scientific and practical conclusions were formulated:

1. The analysis of the existing decision automation methods has identified that such model implementation is usually based on the business logic separation from the decision-making process where the decisions are usually treated as an act of deciding made by a human actor on the basis of information provided by the decision support systems. However, the overall process of deciding is usually left without proper attention, and typical sub-processes, usually accompanying the automated decision process, e.g. decision support, evaluation of decision impact, decision implementation, etc., are not identified or developed separately. Such approach prevents overall understanding of the automated decision process and does not consider the knowledge acquired by making and implementing decisions.
2. Having analysed rule, business object, and business process models of the proposed framework and model transformations into the software components, we conclude that rule models (SRML, DT, PRR) together

with process models (UML, BPMN) are suitable for decision process modelling and should be selected by evaluating particular needs depending on the complexity of a decision-making process and the already existing IS infrastructure in the particular business environment.

3. The experience acquired after having modelled a decision-making process by using framework models demonstrates that the main advantage of the proposed method is gained by separation of business logic and decision processes at early engineering phase and later integration during the IS engineering process. That allows minimising a gap between evolving business strategy and guidance of the automated business processes implemented into the IS software. The decision logic expressed in rules and exposed at a business level allows keeping control of an automated decision process in the hands of business users.
4. The conducted experiments have shown increasing complexity of the decision model requires employment of additional infrastructure used for management, transformation, and execution of the involved models. This requires additional time and tangible resources. However, once developed, it can be reused for development and implementation of similar decision processes without significant utilisation of system engineering resources.
5. The experience gained by applying the proposed method and the performed experiments have demonstrated that automated decisions developed according to the proposed approach are modelled and implemented into the IS software at least about 10% faster. Nevertheless, the main advantage is gained by faster modification and reuse of previously implemented decision models, which allows about 40% acceleration, in comparison with the traditional requirement driven IS software engineering methods. Moreover, the changes of decision logic which do not require introduction of new constructs needed for a guided automated process are allowed by changing business rules without running any engineering process at all.
6. It is important to take into account that by employment of the proposed executable decision process, decision-making is not a separate function of the management anymore. In fact, such decision-making is intertwined within the other functions of several integrated enterprise information systems and software components including rule management systems, inference engines, business analysis, data mining, and decision support systems. Although management keeps automated decision process control with possibility to influence and modify knowledge used by technical systems, but the question of responsibility dedication to the

software system may arise. This aspect could be an interesting suggestion for further socio-technical system research.

References

Asuncion, C. H.; Iacob, M. E.; Sinderen, M. 2010. Towards a Flexible Service Integration Through Separation of Business Rrules, in *Proceedings of the 14th IEEE International Enterprise Distributed Object Computing Conference*. IEEE, 184–193.

Avdejenkov, V.; Vasilecas, O. 2004. Business Rules Systems Modelling With UML and Rules Enforcing in Active Relational Databases. *Lithuanian Mathematical Collection* 44: 383–387.

Avison, D. E.; Wood-Harper, T.; Vidgen, R.; Wood, B. 1998. A Further Exploration into Information Systems Development: the Evolution of Multiview2. *Information Technology & People* 11(2): 124–139.

Badawy, M.; Richta, K. 2002. Deriving Triggers from UML/OCL Specification, in *M. Kirikova et al. (Eds.). Proc. of ISD 2002 conference on Information Systems Development: Advances in Methodologies, Components and Management.* Kluwer Academic/Plenum Publishers, 305–316.

Bocij, P.; Chaffey, D.; Hickie, S.; Greasley, A. 2005. *Business Information Systems*. Pearson Education.

BRG. 2000. *Defining Business Rules – What are They Really?* Business Rules Group (BRG). Available from Internet: <http://www.businessrulesgroup.org/first_paper/br01c0.htm>.

BRG. 2003. *The Business Rule Manifesto - The Principles of Rule Independence*. Business Rules Group (BRG). Available from Internet: <http://www.businessrulesgroup.org/brmanifesto/BRManifesto>.

BRG. 2005. The Business Motivation Model - Business Governance in a Volatile World. Version 1.2, *Ross, G. R. (Eds.). Business Rules Group (BRG).* Available from Internet: <http://www.businessrulesgroup.org/bmm.shtml>.

Bruyn, M. L.; Bandali, F.; Lamoureux, T. 2006. *Decision Making Styles: Classification System, Contextual Analysis and Validation of Classification System*. *Report to* Department of National Defence, Contract No. W7711-047911/001/TOR, Call-up No. 7911-03. (November, 2010). Available from Internet: <http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA472965&Location=U2&doc=GetTRDoc.pdf>.

Butleris, R.; Kapočius, K. 2002. The Business Rules Repository for Information Systems Design, in *Proceedings of the 6th East-European Conference ADBIS'2002. Research Communications*, Bratislava: STU. 2: 64–77.

Casati, F.; Ilnicki, S.; Jin, L.; Krishnamoorthy, V.; Shan, M. . C. 2000. Adaptive and *Dynamic Service Composition in eFlow*. HP Lab. Techn. Report, HPL-2000-39. Available from Internet: <http://www.hpl.hp.com/techreports/2000/HPL-2000-39.html>.

Chang, C.; Melamud, Y.; Seabrook, D. 1983. *Expert Systems*.

Chung, C. H. 2003. Operation management. *Encyclopedia of Information Systems* 3: 391–402.

Ciuksys, D.; Caplinskas, A. 2007. Reusing Ontological Knowledge About Business Processes in IS Engineering: Process Configuration Problem. *Informatica* 18(4): 585–602.

Czarnecki, K.; Helsen, S. 2006. Feature-based Survey of Model Transformation Approaches. *IBM Systems Journal – Model-driven Software Development* 45(3).

Czarnecki, K.; Helsen, S. 2003. Classification of Model Transformation Approaches, in *2nd OOPSLA'03 Workshop on Generative Techniques in the Context of Model-Driven Architecture*. Anaheim, CA, USA.

Date, C. J. 2000. *What Not How: The Business Rules Approach to Application Development*. Addison-Wesley.

Davis, G. 1974. *Management Information Systems: Conceptual Foundations, Structure, and Development*. New York: McGraw-Hill, Inc.

Deming, W. E. 1992. *Out of the Crisis: Quality, Productivity and Competitive Position*. Cambridge.

Dietz, J. L. G. 2006. *Enterprise Ontology. Theory and Methodology*. New York: Springer-Verlag Berlin Heidelberg.

Duan, Z. H.; Xie, Z. Q.; Cheng, H. 2010. Overview of ILOG JRules and WebSphere Process Server integration. *developerWorks*. IBM. Available from Internet: <http://public.dhe.ibm.com/software/dw/wes/1002_duan/1002_duan-pdf.pdf>.

Elam, J. 1985. A vision for DSS, in *Fick, G; Sprague, R.H. (Eds.). DSS: Issues and Challenges*. Pergamon Press, Oxford.

Eveleens, J. L.; Verhoef, C. 2010. The Rise and Fall of the Chaos Report Figures. *IEEE Software* 27(1): 30–36.

Galindo, J.; Urrutia, A.; Piattini, M. 2006. *Fuzzy Databases: Modeling, Design, and Implementation*. IGI Publishing, Hershey, PA.

Giaglis, G. M. 2001. A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques. *The International Journal of Flexible Manufacturing Systems* 13: 209–228.

Goedertier, S.; Vanthienen, J. 2005. Rule-based Business Process Modeling and Execution, in *Proceedings of the International IEEE EDOC Workshop on Vocabularies, Ontologies and Rules for The Enterprise (VORTE 2005)*, 67–74.

Gudas, S.; Skersys, T. 2005. The Enhancement of Class Model Developement Using Business Rules. *LNCS* 3746: 480–490.

Gudas, S.; Skersys, T.; Lopata, A. 2005. Approach to Enterprise Modelling for Information Systems engineering. *Informatica* 16(2): 175–192. ISSN 0868-4952.

Hay, D. C. 2003. *Requirement Analysis: From Business View to Architecture*. New Jersey: Prentice Hall.

Hay, D. C.; Kolber, A.; Anderson, H. K. 1997. *Guide Business Rule Project: Final Report*. Available from Internet: <http://grace.evergreen.edu/businessRules>.

Hammer, M. 1996. *Beyond Reengineering. How the Process-Centred Organization is Changing our Work and our Lives*. New York.

Herbst, H. 1994. The Specification of Business Rules: A Comparison of Selected Methodologies. *Methods and Associated Tools for the Information System Life Cycle,* 29–46.

ILOG. 2006a. *ILOG Business Rule Components: Business Rule Management, Deployment & Execution*. ILOG.

ILOG. 2006b. *Decision Services: The Next SOA Challenges White Paper*. ILOG.

Ives, B.; Hamilton, S.; Davis, G. 1980. A Framework for Research in Computer-based Management Information Systems. *Management Science - Management* 26(9): 910–934.

Keen, P. G. W.; Morton, M. S. S. 1978. *Decision Support Systems: an Organizational Perspective*. Reading, Mass.: Addison-Wesley Pub. Co. ISBN 0-201-03667-3.

Kilov, H.; Simmonds, I. 1996. Business Patterns: Reusable Abstract Constructs for Business Specification, in *Humphreys, P. et al. (Eds.). Implementing Systems for Supporting Management Decisions: Concepts, methods and experiences*. Chapman and Hall, 225–248.

Krogstie, J.; Solvberg, A. 2003. *Information System Engineering: Conceptual Modelling in a Quality Perspective*. Trondheim, Norway: Kompendiumforlaget.

Lapouchian, A. 2005. *Goal-oriented Requirements Engineering: An Overview of the Current Research*. University of Toronto.

Linehan, M. H.; de Sainte Marie, C. 2011. The Relationship of Decision Model and Notation (DMN) to SBVR and BPMN. *Business Rules Journal* 12(6). Available from Internet: <http://www.BRCommunity.com/a2011/b597.html>.

Loucopoulos, P.; Katsouli, E. 1992. Modelling Business Rules in an Office Environment. *SIGOIS Bulletin* 13(2): 28–37.

Macesker, B.; Myers, J.; Guthrie, V. H.; Walker, D. A. 2009. Quick-reference Guide to Risk-based Decision Making (RBDM): A Step-by-step Example of the RBDM Process in the Field. Available from Internet: <http://www.au.af.mil/au/awc/ awcgate/uscg/risk-qrg.pdf>.

MacLennan, J.; Crivat, B.; Tang, Z. 2009. *Data Mining with Microsoft SQL server 2008*. WileyPublishing, Inc. ISBN: 978-0-470-27774-4.

Mahajan, S. 1997. Building a Data Warehouse using Oracle OLAP Tools. *Oracle Technical Report, ACTA Journal*. Available from Internet: <http://www.oracle.com>.

Montilva, J. C.; Barrios, J. A. 2004. BMM: A Business Modeling Method for Information Systems Development. *Clei Electronic Journal* 7(2). Paper 3.

Muehlen, M.; Indulska, M. 2010. Modelling Languages for Business Processes and Business Rules: A Representational Analysis. *Information Systems* 35(4): 379–390. ISSN 0306-4379.

Navarro, E.; Letelier, P.; Ramos, I. 2004. UML Visualization for an Aspect and Goal Oriented Approach. *The 5th Aspect-Oriented Modeling Workshop (AOM'04)*. Available from Internet: <http://www.cs.iit.edu/%7Eoaldawud/AOM/AcceptedPapers/index.htm>.

OMG. 2008a. *Business Motivation Model*. Object Management Group (OMG). (December, 2010). Available from Internet: <http://www.omg.org/spec/ BMM/1.0/PDF/>.

OMG. 2008b. *Semantics of Business Vocabulary and Business Rules (SBVR)*. Version 1.0. (December, 2008). Available from Internet: <http://www.omg.org/docs/formal/08-01-02.pdf>.

OMG. 2011a. *Business Process Model and Notation (BPMN)*. Object Management Group (OMG). Available from Internet: <http//www.omg.org/spec/BPMN/2.0/>.

OMG. 2011b. *Decision Model and Notation RFP*. Object Management Group. Available from Internet: <http://www.omg.org/cgi-bin/doc?bmi/11-03-04>.

Orriens, B.; Yang, J.; Papazoglou, M. P. 2003. A Framework for Business Rule Driven Service Composition. *LNCS* 2819: 14–27.

Perez-Castillo, R.; de Guzman, R. I. G.; Piattini, M. 2010. Implementing Business Process Patterns through QVT Transformations. *LNCS* 6142: 168–183.

Power, D. J. 2007. A Brief History of Decision Support Systems. *DSSResources.COM*. (December, 2010). Available from Internet: <(http://dssresources.com/history/dsshistory.html>.

Regev, G.; Wegmann, A. 2006. Taxonomy of Flexibility in Business Processes, in *Proceedings of 7th Workshop on Business Process Modeling, Development and Support*.

Riordan, S. J. P. 1997. *The Purpose of Business and the Human Good*. University of Saint Thomas. Available from Internet: <http://www.stthomas.edu/cathstudies/cstm/antwerp/p6.htm>.

Rosca, D.; Greenspan, S.; Wild, C. 2002. Enterprise Modelling and Decision-support for Automating the Business Rules Lifecycle. *Automated Software Engineering* 9(4): 361–404.

Ross, R. G. 2010. Decision Analysis Using Decision Tables and Business Rules. A Business Rule Solutions White Paper. *Business Rule Solutions, LLC*. Available from Internet: <http://www.brsolutions.com/b_decision.php>.

Schonenberg, M. H.; Mans, R. S.; Russell, N. C.; Mulyar, N. A. 2008. Towards a Taxonomy of Process Flexibility, in *Proceedings of CAiSE'08 Forum*, 81-84. (April 15, 2011). Available from Internet: <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-344/paper21.pdf>. ISSN 1613-0073.

Schwartz, B. 2004. The Paradox of Choice: Why More Is Less. *Ecco*. ISBN 0-06-000568-8.

Sheena, I. S.; Lepper, M. R. 2000. When Choice is Demotivating: Can One Desire Too Much of a Good Thing? *Journal of Personality and Social Psychology* 79(6): 995–1006.

Simon, H. A. 1985. *The shape of automation for Men and management*. New York: Harper & Row.

Sol, H. G.; Takkenberg, C. A. T. 1987. Expert Systems and Artificial Intelligence in Decision Support Systems, in *Second Mini Euroconference, Lunteren, The Netherlands*. Springer, 1-9. ISBN 9027724377.

Sosunovas, S.; Vasilecas, O. 2004. Transformation of Business Rules Models in Information Systems Development Process, in *Proceedings of Databases and Information Systems Doctoral Consortium: Sixth International Baltic Conference BalticDB&IS 2004*, *Riga*, Latvia: University of Latvia, 672: 373–384.

Sternberg, E. 1998. Corporate Governance - Accountability in the Marketplace. *Hobart Paper* 137.

Taylor, J.; Raden, N. 2007. *Smart (Enough) Systems: How to Deliver Competitive Advantage by Automating Hidden Decisions*. Prentice-Hall. ISBN: 978-0132347969.

Tang, Z.; Maclennan, J.; Kim, P. P. 2005. Building Data Mining Solutions with OLE DB for BM and XML for Analysis. *SIGMOD* 34(2): 80–85.

Valatkaite, I.; Vasilecas, O. 2002. Deriving Active Database Triggers from Business Rules Model with Conceptual Graphs. *Lithuanian Mathematical Collection* 42: 289–293.

Valatkaite, I.; Vasilecas, O. 2005. On Business Rules Automation: the BR-centric IS Development Framework*, LNCS* 363*:* 350–365.

Van Assche, F.; Layzell, P.; Loucopoulos, P.; Speltincx, G. 1988. Information Systems Development: a Rule-based Approach. *Journal of Knowledge Based Systems* 1(4): 227–234.

Van Lamsweerde, A. 2009. *Requirements Engineering - From System Goals to UML Models to Software Specification.* Wiley.

Von Hale, B. 2001. *Business Rules Applied: Building Better Systems Using the Business Rules Approach*. New York: John Wiley & Sons, Inc.

Von Halle, B.; Goldberg, L. 2010. *The Decision Model: a Framework for Business Logic and Business-driven SOA*. USA: Auerbach Publications, Taylor & Francis Group.

Wiederhold, G. 2000. Information Systems that Really Support Decision-making. *Journal of Intelligent Information Systems* 14(2): 56-66.

Zachman, J. 1987. A Framework for Informationsystems Architecture. *IBM Systems Journal* 26(3): 276.

Zachman, J.; Sowa, J. 1992. Extending and Formalizing the Framework for Information Systems Architecture. *IBM Systems Journal* 31(3): 590.

Zimbrao, G.; Miranda, R.; Souza, J. M.; Estolano, M. H.; Neto, F. P. 2003. Enforcement of Business Rules in Relational Databases Using Constraints, in *Proceedings of the XVIII Simposio Brasileiro de Bancos de Dados / SBBD 2003.* UFAM, 129-141.

List of Publications by the Author on the Topic of the Dissertation

In the Reviewed Scientific Journals

Avdejenkov, V.; Vasilecas, O.; Smaizys, A. 2009. Business Rule Management in Enterprise Resource Planning Systems, *Frontiers in Artificial Intelligence and Applications, Databases and Information Systems V*. Amsterdam: IOS Press, Vol. 187: 255–266. ISSN 0922-6389.

Martišius, M.; Vasilecas, O.; Šmaižys, A. 2010. Verslo taisyklių integravimas vykdomuose verslo procesų modeliuose, *Technologijos mokslo darbai Vakarų Lietuvoje.* Vol. VII: 204–209. ISSN 1822-4652.

Pareigis, G.; Vasilecas, O.; Šmaižys, A. 2010. Neuroninių tinklų panaudojimas intelektualizuotose informacinėse sistemose, *Technologijos mokslo darbai Vakarų Lietuvoje.* Vol. VII: 220–225. ISSN 1822-4652.

Smaizys., A.; Vasilecas, O. 2010. Business Rule Model Integration into the Model of Transformation Driven Software Development, *LNCS,* Berlin Heidelberg: Springer-Verlag, Vol. 5968, 153–160. ISSN 0302-9743. (Thomson ISI Master Journal List, SpringerLink)

Smaizys, A.; Vasilecas, O. 2009a. Business Rule Based Agile ERP Systems Development, *INFORMATICA,* Vol. 20(3): 439–460. ISSN 0868-4952. (Thomson ISI Web of Science)

Smaizys, A.; Vasilecas, O. 2009b. Agile Software System Development and Customisation using Business Rules, *Frontiers in Artificial Intelligence and Applications, Databases and Information Systems V*. Amsterdam: IOS Press, Vol. 187: 243–254. ISSN 0922-6389.

Vasilecas, O.; Smaizys, A. 2008a. Business Rule Based Software System Configuration Management and Implementation Using Decision Tables, *Advances in Databases and Information Systems Research Communications*. CEUR-WS.org, Vol. 325. ISSN 1613-0073.

Vasilecas, O.; Smaizys, A. 2007a. The Framework for Business Rule Based Software Modelling: An Approach for Data Analysis Models Integration, *Frontiers in Artificial Intelligence and Applications, Databases and Information Systems IV*. Amsterdam: IOS Press, Vol. 155: 175–188. ISSN 0922-6389.

Vasilecas, O.; Smaizys, A. 2005a. Business Rule Based Knowledge Integrated Intelligent System Framework, *Informacijos mokslai.* Vol. 34: 195–200. ISSN 1392-0561.

Vasilecas, O.; Šmaižys; A.; Rima, A. 2010. Trijų lygmenų karkaso taikymas daugiamatės duomenų analizės informacinėms sistemoms kurti, *Technologijos mokslo darbai Vakarų Lietuvoje.* Vol. VII: 259–263. ISSN 1822-4652.

Zikaras, S.; Šmaižys, A.; Vasilecas, O. 2010. Informacinių sistemų modernizavimas. *Jūra ir aplinka*, Vol. 2 (19): 65-68. ISSN 1392-785X.

Other Papers

Airošius, N.; Šmaižys, A. 2007. Verslo procesų reinžinerija: Penkių aspektų modeliavimo metodas, *Informacinės technologijos 2007*, Kaunas: Technologija, 3–7. ISSN 1822-6337.

Avdejenkov, V.; Vasilecas, O.; Smaizys, A. 2008. Problems and Solutions with Business Rule Management and Enterprise Resource Planning System Integration, in *Haav, H.M.; Kalja, A. (Eds.). Proc. of 8th International Baltic Conference on Databases and Information Systems, June 2-5, 2008, Tallinn, Estonia*. Tallinn University of Technology Press, 315–325. ISBN 978-9985-59-789-7. (Thomson ISI Proceedings)

Bražinskas, R.; Šmaižys, A.; Vasilecas, O. 2007. Verslo taisyklių su privalomumo lygmenimis taikymas informacinių sistemų kūrime, *10-osios Lietuvos jaunųjų mokslininkų konferencijos „Mokslas – Lietuvos ateitis" pranešimų rinkinys, Vilnius, Lietuva.* Vilnius: Technika, 277–282. ISBN 978-9955-28-138-2.

Rima, A.; Šmaižys, A; Vasilecas, O. 2011. Sprendimų rėmimo sistemose naudojamų duomenų saugyklų projektavimo metodų analizė. *14-osios Lietuvos jaunųjų mokslininkų konferencijos „Mokslas – Lietuvos ateitis" 2011 metų teminės konferencijos Informatikos sekcijos straipsnių rinkinys*, Vilnius: Technika, JMK 14. ISSN 2029-7149.

Rima, A.; Šmaižys, A.; Vasilecas, O. 2007a. Intelektualizuota duomenų analizė verslo taisyklių transformacijų pagrindu, *Informacinės technologijos 2007*, Kaunas: Technologija, 202–206. ISSN 1822-6337.

Rima, A.; Šmaižys, A.; Vasilecas, O. 2007b. Verslo taisyklių panaudojimas duomenų analizės metamodelių transformacijų pagrindu, *10-osios Lietuvos jaunųjų mokslininkų konferencijos „Mokslas – Lietuvos ateitis“ pranešimų rinkinys, Vilnius, Lietuva.* Vilnius: Technika, 264–269. ISBN 978-9955-28-138-2.

Šliamin, A.; Zaicev, A.; Šmaižys, A. 2009. ECA taisyklių transformacijų panaudojimas verslo procesų modeliavime, *Respublikinės Klaipėdos universiteto Gamtos ir matematikos mokslų fakulteto Studentų mokslinės draugijos konferencijos „Fundamentiniai tyrimai ir inovacijos mokslų sandūroje“ medžiaga, Klaipėda, Lietuva*. Klaipėdos universiteto leidykla, 108–116. ISBN 978-995-18-329-7.

Smaizys, A. 2004. XML-based Representation of Business Rules, *7-osios Lietuvos jaunųjų mokslininkų konferencijos „Lietuva be mokslo – Lietuva be ateities“ pranešimų rinkinys, Vilnius, Lietuva*. Vilnius: Technika, 174–179. ISBN 9986-05-775-2.

Smaizys, A.; Vasilecas, O. 2008c. Business Rule Based Software System Customisation and Configuration Management, in *Haav, H. M., Kalja, A. (Eds.). Proc. of 8th International Baltic Conference on Databases and Information Systems, June 2-5, 2008, Tallinn, Estonia*. Tallinn University of Technology Press, 303–314. ISBN 978-9985-59-789-7. (Thomson ISI Proceedings)

Strigūnas, R.; Šmaižys, A.; Vasilecas, O. 2007a. Verslo taisyklių rinkinio darnos užtikrinimas loginio išvedimo mašina, *Informacinės technologijos 2007*, Kaunas: Technologija, 207–211. ISSN 1822-6337.

Strigūnas, R.; Šmaižys, A.; Vasilecas, O. 2007b. Verslo taisyklių rinkinio darnos užtikrinimas, *10-osios Lietuvos jaunųjų mokslininkų konferencijos „Mokslas – Lietuvos ateitis“ pranešimų rinkinys, Vilnius, Lietuva.* Vilnius: Technika, 283–288. ISBN 978-9955-28-138-2.

Vasilecas, O.; Avdejenkov, V.; Smaizys, A. 2008. Business Rule Management System Integration into the Enterprise Resource Planning Systems Using Triggers, in *Butleris, R. et al. (Eds.). Proc. of the 14th conference on Information and Software Technologies*. Kaunas: Technologija, 370–376. ISSN 2029-0020. (Thomson ISI Proceedings)

Vasilecas, O.; Rataitė, B.; Šmaižys, A. 2005. Verslo taisyklų panaudojimas programų sistemų reinžinerijoje, *Informacinės technologijos 2005*, Kaunas: Technologija, 664–668. ISBN 995-09-788-4.

Vasilecas, O.; Smaizys, A. 2008b. Business Rule Enforcement and Conflict Resolution Using Risk Analysis, in *Targamadzė, A.; Butleris, R.; Butkienė, R. (Eds.). Proc. of the 14th International Conference on Information and Software Technologies, Research Communications, Kaunas University of Technology, Kaunas, Lithuania,* 92–98. ISSN 2029-0039.

Vasilecas, O.; Smaizys, A. 2007b. Business Rule Based Configuration Management and Software System Implementation Using Decision Tables, in *Ionnidis, Y.; Novikov, B.; Rachev, B. (Eds.). Local Proc. of the 11th East European Conference on Advances in Databases and Information Systems, September 29 – October 3, 2007, Varna, Bulgaria,* 27–37. ISBN 978-954-20-0384-7.

Vasilecas, O.; Smaizys, A. 2007c. The Framework for Adaptable Data Analysis System Design, in *Magyar, G.; Knapp, G. (Eds.). Proc. of the 15th International Conference on Information Systems Development 2006, Budapest, Hungary*. New York: Springer, 101–109. ISBN 978-0-387-70760-0. (Thomson ISI Proceedings, SpringerLink)

Vasilecas, O.; Smaizys, A. 2006a. The Framework: an Approach to Support Business Rule Based Data Analysis, in *Vasilecas, O. et al. (Eds.). Proc. of 7th International IEEE Baltic Conference on Databases and Information Systems, July 3, Vilnius, Lithuania.* Vilnius: Technika, 141–147. ISBN 1-4244-0345-6. (Thomson ISI Proceedings, Xplore, Inspec, IEEE/IEE)

Vasilecas, O.; Smaizys, A. 2006b. Business Rule Based Data Analysis for Decision Support and Automation, in *Rachev, B.; Smirkarov, A. (Eds.). Proc. of the International Conference on Computer Systems and Technologies "CompSysTech'06", 15-16 June, 2006*, *Varna, Bulgariai,* II.9-1–II.9-6. ISBN 978-954-9641-46-2. (Inspec)

Vasilecas, O.; Smaizys, A. 2006c. Business Rules Based Enterprise Knowledge Representation for Business Process and Application Development, in *Romansky, R. (Eds.). Proc. of the 20th International Conference on Systems for Automation of Engineering and Research, 22-24 September 2006, Varna, Bulgaria,* 160–164. ISBN 954-438-575-4.

Vasilecas, O.; Smaizys, A. 2005b. Business Rule Specification and Transformation for Rule Based Data Analysis, in *Romansky, R. (Eds). Proc. of the 19th International Conference on Systems for Automation of Engineering and Research. 24-25 September, 2005, Varna, Bulgaria,* 35–40. ISBN 954-438-501-0.

Vasilecas, O.; Smaizys, A. 2005c. Business Rule Based Data Analysis, in *Simutis, R. (Eds.). Proc. of the International Conference Information Technologies for Business 2005. Vilnius University, Kaunas Faculty of Humanities, Kaunas, Lithuania,* 71–76. ISBN 9955-09-871-6.

Vasilecas, O.; Šmaižys, A. 2005d. Verslo taisyklų panaudojimas duomenų analizei ir informacijos pateikimui, *Informacinės technologijos 2005*, Kaunas: Technologija, 655–663. ISBN 995-09-788-4.

Vasilecas, O.; Šmaižys, A. 2005e. Verslo taisyklų panaudojimas duomenų analizei OLAP sistemose, *8-osios Lietuvos jaunųjų mokslininkų konferencija „Lietuva be mokslo – Lietuva be ateities" pranešimų rinkinys, Vilnius, Lietuva*. Vilnius: Technika, 174–179. ISBN: 9986-05-856-2.

Vasilecas, O.; Smaizys, A.; Brazinskas, R. 2009. Risk Analysis Based Business Rule Enforcement for Intelligent Decision Support, in *Papadopoulus, G. A. et al. (Eds.). Information Systems Development. Towards a Service Provision Society. Proc. of the 17th International Conference on Information Systems Development 2008*, *Paphos, Cyprus*. New York: Springer Verlag, 403–411. ISBN 978-0-387-84809-9. (SpringerLink)

Vasilecas, O.; Šmaižys, A.; Rataitė, B. 2005. Verslo taisyklės programų sistemų reinžinerijos procese, *8-osios Lietuvos jaunųjų mokslininkų konferencija „Lietuva be mokslo – Lietuva be ateities" pranešimų rinkinys, Vilnius, Lietuva.* Vilnius: Technika, 664–668. ISBN: 9986-05-856-2.

Zaicev, A.; Šliamin, A.; Šmaižys, A. 2009. Verslo taisyklėmis grindžiamas verslo procesų modeliavimo metodas, *Respublikinės Klaipėdos universiteto Gamtos ir matematikos mokslų fakulteto Studentų mokslinės draugijos konferencijos „Fundamentiniai tyrimai ir inovacijos mokslų sandūroje“ medžiaga, Klaipėda, Lietuva.* Klaipėdos universiteto leidykla, 150–159. ISBN 978-995-18-329-7.

Aidas ŠMAIŽYS

A STUDY ON IMPLEMENTATION OF AUTOMATED DECISION PROCESS INTO THE INFORMATION SYSTEMS

Doctoral Dissertation

Technological Sciences,
Informatics Engineering (07T)

Aidas ŠMAIŽYS

SPRENDIMŲ PROCESŲ AUTOMATIZAVIMO INFORMACINĖSE SISTEMOSE TYRIMAS

Daktaro disertacija

Technologijos mokslai,
informatikos inžinerija (07T)

2011 12 16. 11,25 sp. l. Tiražas 20 egz.
Vilniaus Gedimino technikos universiteto
leidykla „Technika“,
Saulėtekio al. 11, 10223 Vilnius,
http://leidykla.vgtu.lt
Spausdino UAB „Ciklonas“,
J. Jasinskio g. 15, 01112 Vilnius