

MASTER OF SCIENCE THESIS

Business Process Simulation by Management Consultants?

**Introduction of a new approach for business process
modeling and simulation by management consultants**

I.J. Rust B.Sc.

30 August 2011

PUBLIC VERSION

Faculty of Technology, Policy and Management - Delft University of Technology



Delft University of Technology

Business Process Simulation by Management Consultants?

**Introduction of a new approach for business process
modeling and simulation by management consultants**

MASTER OF SCIENCE THESIS

For obtaining the degree of Master of Science in
Systems Engineering, Policy Analysis and Management
at Delft University of Technology

I.J. Rust B.Sc.

30 August 2011



Delft University of Technology

Copyright © I.J. Rust B.Sc.
All rights reserved.

DELFT UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF
SYSTEMS ENGINEERING

The undersigned hereby certify that they have read and recommend to the Faculty of Technology, Policy and Management for acceptance a thesis entitled “**Business Process Simulation by Management Consultants?**” by **I.J. Rust B.Sc.** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: 30 August 2011

Chairman:

prof.dr.ir. A. Verbraeck

First supervisor:

dr. M.D. Seck

Second supervisor:

dr. V. Dignum

External advisor:

dr.ir. I. Wenzler

External advisor:

dr.ir. G.J. Pasman

Preface and Acknowledgments

In front of you lies the result of the graduation research I performed during the past 10 months, namely my Master Thesis to partially fulfill the requirements of obtaining an M.Sc. degree in Systems Engineering, Policy Analysis and Management from Delft University of Technology. This research was conducted as an external research internship at Accenture Nederland. The purpose of this research thesis is threefold:

1. It is a description of my graduation research and the outcomes,
2. it is a starting point for continuing the development of a business process simulation solution for management consultants, and
3. it provides the basic knowledge, as well as references, for management consultants who are interested in knowing more about business process simulation.

The main question of this research may be relatively simple to understand. However, the concepts behind it proved to be much more difficult to understand, let alone to apply in order to find an answer. As my first supervisor, Mamadou Seck, once said to me: “For some graduation projects, you know what the outcomes and deliverables will be. But with this research, nobody exactly knew what the expected outcomes and deliverables were.” I am glad that at last, I can present to you a possible answer to the question: *how can management consultants be supported to model and simulate business processes in a new and easy way*; as well as an actual working prototype. Maybe one day, some of the concepts proposed in this study will be implemented in a final and mature software program, which will be used by management consultants on projects world-wide. My main contribution is hopefully to bring computer simulation of business processes a step closer to a larger audience.

Whenever I was asked during the past months by family, friends, colleagues, or others, to explain what my research was about, I explained them enthusiastically as much as they wanted to know. And every time I finished my story, I thought: “Although there is so much more I still need to understand about this subject, and so much more still needs to be done. . . it is so great that I got the possibility to work on this project.” Of course because of the interesting subject and the possible contribution of the outcomes of this research, but also because all the new things I was learning, the new people I got to know, and the people who I already knew, but got to know better.

Therefore, I would like to thank those people who contributed to this project by sharing their knowledge and experience, who supported me and who have shown their trust and confidence in me. Firstly, I want to thank Mamadou Seck, my first supervisor, for his continuous support and patience during the whole project. I thank him especially for always challenging my concepts and deliverables, and making me aim at something more difficult. Furthermore, I want to thank my second supervisor, Virginia Dignum, for sharing her views on the project, as well as her helpful comments and suggestions on the written deliverables. I thank my third supervisor, Gert Pasman, for adding his design-perspective view to this project. I am also thankful to the chairman of the graduation committee,

Alexander Verbraeck, for his enthusiasm and inspiration, as well as his confidence in me and the final results.

I thank Ivo Wenzler greatly, for offering me the possibility to undertake this research within Accenture, but also for the time he took and his contributions during the design-sessions and the meetings we had. Furthermore, I want to thank my daily supervisor in Accenture, Rutger Deenen, for his guidance and support throughout the whole project, and all the valuable meetings we had. Also, I want to thank all the other Accenture colleagues, who contributed by sharing their knowledge and experience with me.

Furthermore, I also want to thank Deniz Cetinkaya for all her contributions to my research. Her efforts enabled me to finally deliver a functional and working prototype of the solution. I think our collaboration was a stimulation and a catalysis for both our researches.

Next, I thank my parents for always having supported me and provided me with the opportunity to finish this study at Delft University of Technology. I also thank my sister Ida, who has always believed in me and my capabilities, and supported me significantly, several times throughout the project. And finally, I want to thank everybody else who I have not mentioned explicitly, but who know they contributed to this research, to my study-time, and to my life.

While sitting on the balcony of my home, the sun shining brightly, and my notebook in front of me, I just realized that not only my graduation research project has almost come to an end, but also my time of being a student. A new phase of my life is about to start, with probably even greater challenges than performing a graduation research project and writing this thesis. However, by challenging my own capabilities once again during this project, I have come to realize that almost anything is possibly, as long as you believe in yourself, trust yourself, and go for it.

Igor Jan Rust

Delft, August, 2011

Executive Summary

To stay competitive and to operate effectively, an organization needs to improve its process efficiency and its quality by adapting its strategy, structure, management and operations to its changing business environment. Achieving a good understanding of the organization's business processes is therefore essential. A business processes depict the tasks which are performed by resources within an organization, in order to produce a product or deliver a service. Business process simulation (BPS) provides the possibility to simulate and evaluate the dynamic behavior of business processes. It has proven its usefulness in the past decades to analyze business processes and experiment with these, without directly influencing or changing the processes.

To apply BPS, first a conceptual model (the business process model) is developed which is a representation of the business processes. Based on this conceptual model, a simulation expert develops a computer simulation model. After the simulation model is validated for being an adequate representation of reality, it can be executed by computer simulation software. The main problem is that developing a simulation model is being considered as difficult, time-consuming and expensive.

For a management consulting company like Accenture, BPS could be a useful addition to their current practices. Business performance analysis and improvement of client organizations through business processes modeling and (re)design, are part of many projects undertaken within Accenture. However, applying best-practices based on previous projects and experience, is not a proof for optimal results. As BPS requires advanced skills to develop a simulation model, a new approach is required to enable business process simulation by management consultants. In order to support Accenture, the main objective of this research is:

To find new way how a support tool can help management consultants to easily model and simulate business processes.

To accomplish this research objective, first the current business process modeling and simulation practices are examined as how they are currently performed within Accenture's services lines, as well as how they are described in literature. The main goal is to get a general understanding of business process modeling within Accenture, and to identify limitations of the current approach. Next, design requirements are stated which a solution should meet in order to support management consultants. Based on these requirements, a concept design is developed. To increase the likelihood that a designed solution is finally found useful and usable by the end-user, the management consultants, a user-centered design approach is applied. A group of management consultants is actively involved through-out the design processes, in design and evaluation workshops. After the concept design is finalized, a prototype is developed based on the design. As a proof of concept, the prototype is used to model and simulate a sample business process case. Finally, evaluation sessions are held with management consultants to test the usability of the

solution.

The main design requirements for the solution are: 1) a management consultants should be able to easily model business processes as how he sees the business processes; 2) the solution should be able to transform a conceptual model into an executable simulation model; and 3) the solution should be able to simulate business processes.

The new designed approach for modeling and simulation of business processes by management consultants, is based on Model-Driven Development (MDD) principles. MDD allows to (semi)automatically transform a conceptual model into a simulation model. To enable this for management consultants, we defined a conceptual modeling language based on the Business Process Modeling Notation (BPMN) and we adapted it to connect with the consultant's mental model. This modeling language consists of several modeling elements which represent the different elements in a business process (like for instance resources, decisions and activities). By drawing and connecting the modeling elements, the management consultant can construct a model representing the business process as how he sees them. The main concept of the language is that entities (for instance phone calls, orders or insurance claims) arrive in an organization through a Start Event-element; enter a Swimlane-element which can contain activities done and decisions made by resources; and finally leave the organization through an End Event-element.

For each modeling element, a corresponding DEVS-simulation component is defined, including the internal states of the component. A simulation model can be constructed by coupling these simulation components. To support the transformation from conceptual model to simulation model, a set of model-transformation rules is defined. And finally, several output statistics were specified, related to for instance resource utilization and processing times of entities in a business process.

As a proof of concept of our designed solution for management consultants, we applied the Model-Driven Development framework for Modeling and Simulation (MDD4MS). The final prototype includes a visual model builder (which allows to draw business process models using the modeling elements), as well as a set of formally defined model-transformation rules (which allow to transform the conceptual model into a simulation model), and the simulation components implemented in Java programming language. The Distributed Simulation Object Library (DSOL) was selected to provide the simulation and execution functionalities of the prototype.

The prototype is used to model and simulate part of the real business process of a Dutch telecom operator. This business process contains 3 groups of resources, 26 activities performed by these resources, and different types of entities that are processed. Based on this evaluation, we concluded that the solution was indeed able to model and simulate business processes as how consultants would do it. The second evaluation was a usability test with several management consultants, who were asked to use the prototype and perform a series of modeling activities with it.

Based on the usability evaluation and the sample-case, as well as the questionnaires that were received back from the consultants, we conclude that **the proposed design is a fruitful possibility to enable business process simulation by management**

consultants.

However, as this research is performed in a relative short amount of time and with limited means, several recommendation are made to direct future research and development activities. The main recommendations for further scientific research are the following.

- ▷ The applicability of the modeling language should be verified in a more general sense, for instance by business analysts and management consultants from other companies.
- ▷ Research on a Model-Driven Development implementation for Modeling and Simulation should continue.
- ▷ A design framework could be developed to guide the design and development of a conceptual modeling language with simulation purposes in a rigorous manner.

Several recommendations can be formulated for Accenture.

- ▷ To see whether the proposed modeling elements are adequate to express different business processes and thus to increase its credibility, the business process modeling language should be evaluated by more management consultants and preferably in an actual consulting project.
- ▷ The simulation components need to be extended to include the full design specification, and should be validated by an external party to confirm their correct implementation.
- ▷ Further research should clarify for what purposes business process simulation can be usefully applied within Accenture, like for instance operational excellence projects or sizing of teams.
- ▷ Based on the limitations study on simulation projects undertaken by Accenture in the past, we recommend as a short-term plan that a permanent team of management consultants is set-up who are specialized in modeling and simulation. This, in order to improve the efficiency in and effectiveness of future simulation-related projects.

Contents

Preface and Acknowledgments	v
Executive Summary	vii
I Main Thesis	1
1 Introduction	3
1.1 Background	4
1.2 Research problem	5
1.3 Research objective	7
1.4 Research questions	8
1.4.1 Sub research questions	8
1.5 Research approach	9
1.5.1 Research methodologies	12
2 Accenture and Business Process Modeling	15
2.1 Background on Accenture	15
2.2 Business process modeling in Accenture	16
2.2.1 Accenture's service lines and business process modeling	17
2.2.2 Accenture's decision support tools and projects	20
2.3 What are the limitations of the current approach?	24
2.4 Summary and Conclusions	26
3 Background on Business Process Modeling and Simulation	29
3.1 Motivation for business processes modeling and analysis	29
3.2 Business process modeling in practice	32
3.2.1 Modeling of business processes	33
3.2.2 Business process analysis and simulation	38
3.3 Limitations of current approaches and tools	43
3.4 Comparison of literature and Accenture's case	46
3.4.1 Comparison	47
3.4.2 Problem space	48
3.5 Summary and conclusions	49
4 Design Requirements	51
4.1 Stakeholders and needs	52
4.2 Identifying requirements	54
4.2.1 Requirements related to: Usefulness	54
4.2.2 Requirements related to: Usability	55
4.2.3 Requirements related to: Usage	56
4.3 Overview requirements and evaluation criteria	57

4.4	Conclusions	60
5	Solution Design	63
5.1	End-user involvement through workshops	65
5.1.1	Example: First design workshop	66
5.2	The mental model of a consultant	69
5.3	From business processes to simulation model	72
5.3.1	Formalization of conceptual model: BPMN	72
5.3.2	Proposed business process modeling elements	73
5.3.3	Components supporting simulation model construction	76
5.3.4	Overview of simulation components	79
5.3.5	Conceptual model to simulation model transformation	86
5.4	Statistics and outcomes	91
5.4.1	What to collect?	91
5.4.2	How to present?	96
5.5	Interaction and Process	98
5.5.1	Interaction with a tool and visual appearance	99
5.5.2	Implementation in consultants' process	101
5.6	Summary and conclusions	102
6	Proof of Concept	105
6.1	DSOL and MDD4MS-framework	106
6.2	Development of a prototype	110
6.2.1	Usage of DEVSDSOL	110
6.2.2	Implementation of simulation components in Java	111
6.2.3	ATL-Transformation rules	114
6.2.4	The final prototype	116
6.3	Verification of the prototype and components	118
6.3.1	Verification: model transformation	120
6.3.2	Verification: correctness of behavior and outcomes	122
6.4	Case study: A Telecom Operator	127
6.4.1	Case description	127
6.4.2	Model development and transformation	128
6.4.3	Conclusions	128
6.5	Usability evaluation with consultants	130
6.5.1	Evaluation workshop and questionnaires	130
6.5.2	Outcomes and findings	131
6.6	Design requirements evaluation	133
6.7	Summary and conclusions	136
7	Conclusions, Reflection and Recommendations	137
7.1	Conclusions	137
7.2	Research reflection	141
7.2.1	Reflection on design science research approach	141
7.2.2	Reflection on research stages	142

7.3	Recommendations	146
7.3.1	Recommendations for further research	146
7.3.2	Recommendations for Accenture	147
	Bibliography	149
II	Appendix	155
A	Responsibility assignment matrix for BPDST projects	157
B	Expert-interviews	159
C	Background on simulation	161
D	Minutes of design sessions	163
D.1	Design session 1: March 28, 2011	163
D.2	Design session 2: May 10, 2011	166
D.3	Design session 3: May 24, 2011	171
E	Component descriptions	177
E.1	Modeling elements and components	177
E.1.1	Start Event	177
E.1.2	End Event	179
E.1.3	Task	181
E.1.4	Sub Process	183
E.1.5	Sequence Flow	184
E.1.6	Exclusive Gateway	184
E.1.7	Parallel Gateway	186
E.1.8	Swimlane	189
E.1.9	Swimlane Entry	190
E.1.10	Swimlane Exit	192
E.1.11	Resource Manager	193
E.1.12	Entity	196
E.1.13	Signal	196
E.1.14	BPMNCoupledModel	197
F	Transformation rules descriptions	199
F.1	Pseudo-code description	199
G	Sample BPMM and DEVS models	203
H	Verification of model translation	209

I	Verification of prototype behavior	213
I.1	Test 1: Single-server queuing system	213
I.2	Test 2: Single-server queuing system with Exclusive Gateway	220
I.3	Test 3: Parallel activities	221
I.4	Test 4: Parallel activities extended	225
J	Modeling and Simulation prototype	227
J.1	Specification of components	227
K	Case study: Telecom Provider	229
L	Usability evaluation	231
L.1	Material and preparations	231
L.2	Minutes evaluation with I. Wenzler: June 23, 2011	233
L.3	Minutes evaluation with R. Deenen: June 24, 2011	236
L.4	Received questionnaires	239

List of Figures

1.1	Simulation study life cycle	5
1.2	Design Science Research Framework (Adapted from: [HC10])	10
1.3	Research approach: Transformation from research questions to thesis chapters	11
2.1	Outline Chapter 2	15
2.2	Best practices [Accenture, 2011]	18
2.3	Generic BPDST components	21
2.4	Current business planning decisions support tool development process	22
3.1	Outline Chapter 3	29
3.2	Levels of business processes [Wes07]	32
3.3	Levels of abstraction [Wes07]	34
3.4	Modeling and understanding [GW04]	36
3.5	Representing basic control flow patterns in UML	37
3.6	Representing basic control flow patterns in YAWL	38
3.7	Representing basic control flow patterns in BPMN	39
3.8	Simulation study life cycle	41
3.9	Categorization of limitations of business process simulation studies	44
3.10	Problem space	50
4.1	Outline Chapter 4	52
5.1	Overview of design choices	64
5.2	Outline Chapter 5	65
5.3	Sample selection paper prototype modeling elements	67
5.4	Example how participants were demonstrated to use modeling elements	68
5.5	Workshop participant uses proposed modeling elements to model sample process (workshop no. 3 - May 24, 2011)	68
5.6	White board sample sheet 1	69
5.7	White board sample sheet 2	69
5.8	Overview selected BPMN modeling elements	74
5.9	Metamodel proposed business process modeling language	74
5.10	A spectrum of reuse (adopted from: [Pidd2002])	76
5.11	DEVS atomic and coupled model decomposition	78
5.12	Simplified DEVS Metamodel (adopted from [CVS11])	78
5.13	Sample atomic model State Diagram	80
5.14	State diagram Start Event	82
5.15	State diagram End Event	82
5.16	State diagram Task	83
5.17	State diagram Swimlane Entry	85
5.18	State diagram Resource Manager	87
5.19	Transformation from Sequence Flows to Couplings	88

5.20	Sample Business Process Model: One Task	89
5.21	Sample DEVS model: One Task	89
5.22	Sample Business Process Model: Two Swimlanes and Exclusive Gateway	90
5.23	Organization Performance Pyramid	91
5.24	Screenshot DSOL statistics output	98
5.25	Screenshot Arena statistics output	98
5.26	Screenshot of Microsoft Visio 2007 with BPMN template	99
5.27	Screenshot of Arena 13	100
5.28	Screenshot of Arena 13 specification windows	101
6.1	Outline Chapter 6	106
6.2	Simplified DEVSDSOL Metamodel (adopted from [CVS11])	109
6.3	Metamodels and Model transformations in MDD4MS (adopted from [CVS11])	109
6.4	Screenshot MDD4MS model builder prototype	117
6.5	Verification and validation (Adopted from [Sar04])	118
6.6	Test 1: conceptual model	121
6.7	Test 2: conceptual model	121
6.8	A single-server queueing system	123
6.9	DEVSDSOL model single-server queueing system	124
6.10	Test 2: Single-server queueing system with Exclusive Gateway	125
6.11	Test 3: Business process model as drawn with prototype model builder	125
6.12	Case study: Output statistics example	129
6.13	Arena example run set-up	135
A.1	Responsibility assignment matrix of	157
C.1	Steps in a simulation study according to [Ban98]	161
C.2	Steps in a simulation study according to [Sha75] (adopted from [Cet10])	162
D.1	Workshop participant uses proposed modeling elements to model sample process (workshop no. 3 - May 24, 2011)	173
E.1	Graphical representation of a Start Event	178
E.2	State diagram Start Event	178
E.3	Graphical representation of an End Event	180
E.4	State diagram End Event	180
E.5	Graphical representation of a Task	181
E.6	State diagram Task	182
E.7	Graphical representation of a Sub Process when folded	183
E.8	Graphical representation of a Sub Process when unfolded	183
E.9	Graphical representation of a Sequence Flow	184
E.10	Graphical representation of an Exclusive Gateway	184
E.11	State diagram Exclusive Gateway	185
E.12	Graphical representation of a Parallel Gateway	186
E.13	State diagram Parallel Gateway Split	187

E.14	State diagram Parallel Gateway Join	187
E.15	Graphical representation of a Swimlane	189
E.16	State diagram Swimlane Entry	191
E.17	State diagram Swimlane Exit	192
E.18	State diagram Resource Manager	194
G.1	Sample BPM: Two Swimlanes and Exclusive Gateway	203
G.2	Sample BPM: Two Swimlanes and Exclusive Gateway (DEVS)	204
G.3	Sample BPM: 2 Swimlanes and Parallel Activity	205
G.4	Sample BPM: Two Swimlanes and Parallel Activity (DEVS)	206
G.5	Sample BPM: Two Swimlanes and Parallel Activity (Arena)	207
I.1	Arena representation of a single-server queueing system	213
I.2	Entities in comparison Arena - DEVSDSOL model	215
I.3	Server utilization comparison DEVSDSOL model - expected value	216
I.4	Utilization comparison Arena - DEVSDSOL model	216
I.5	Entities in queue comparison Arena - DEVSDSOL model	217
I.6	Waiting time comparison Arena - DEVSDSOL model	217
I.7	Entities out comparison Arena - DEVSDSOL model	218
I.8	Test 2: Single-server queueing system with Exclusive Gateway	221
I.9	Entities in queue comparison Arena - DEVSDSOL model	221
I.10	Entities in queue comparison Arena - DEVSDSOL model	222
I.11	Test 3: conceptual model prototype	223
J.1	Properties Window: Start Event	227
J.2	Properties Window: End Event	227
J.3	Properties Window: Swimlane	227
J.4	Properties Window: Root model	227
J.5	Properties Window: Exclusive Gateway - Probability	228
J.6	Properties Window: Exclusive Gateway - Attribute	228
J.7	Properties Window: Parallel Gateway Split	228
J.8	Properties Window: Parallel Gateway Join	228
J.9	Properties Window: Task	228
L.1	Sample model usability evaluation: step 1	231
L.2	Sample model usability evaluation: step 2	232
L.3	Sample model usability evaluation: step 3	232

List of Tables

2.1	Overview Accenture's operating groups and industries	16
2.2	Past BPDST projects	22
4.1	Usefulness requirements and evaluation criteria	57
4.2	Usability requirements and evaluation criteria	59
4.3	Usage requirements and evaluation criteria	60
6.1	Prototype statistics generation	118
6.2	Test4: Resource utilization	126
6.3	Test4: Total Time in System	126
6.4	Test4: Total Process Time	127
6.5	Test4: Total Waiting Time	127
1.1	Specification of Single-server queueing system model	214
1.2	Outcomes 1 execution as presented in [LK00]	214
1.3	Average results of 20 executions (DSOL)	214
1.4	Average results of 20 executions (Arena)	214
1.5	Percentage outcomes of 20 model executions	221
1.6	Specification of parallel activities model	222
1.7	Test3: Resource utilization	222
1.8	Test3: Total Time in System	222
1.9	Test3: Total Process Time	224
1.10	Test3: Total Waiting Time	224
1.11	Specification of extended parallel activities model	225

Part I

Main Thesis

Introduction

"I almost forgot!", Christina thought out-loud when she saw the subject of an e-mail in her inbox. She just arrived home after a weekend sailing with her colleagues and forgot that tomorrow she would begin with a new project. Several months ago she started working for a large management consulting firm and recently rolled-of her first project as the project was finished. For the past three weeks she 'sat on the couch' in the main office building, as her colleagues normally call it when you're not on a project and need to fill-up your time by helping others with their projects. It was actually quite a nice time, as she finally got to meet some new colleagues while they were working on their projects. During her last project (which was actually also her first one since she got hired) she was almost never in the main office building, so in her first months she only met the colleagues that were on the same project as she was.

The new project she would start with tomorrow, was at a large telecom operator and the e-mail she received contained a general description about the project and what she would be doing for the next several months. She opened the e-mail and she realized quite quickly that both the project as well as her role, were quite similar to the previous project. The only difference was, that she would be staying in a hotel during the next couple of weeks, as the location of the client's office building was about 200 miles away from her house. However, living in a hotel for several weeks was not the only thing she was worrying about. During her last project she encountered some frustrating things and if this time it would be the same, well, it would be a hard time.

The main frustration during the last project related to the interactions with her own colleagues and the managers of the client firm. In the beginning, everything went smooth. She was assigned with the task to map the business processes of some of the departments of the client. These descriptions were then given to someone of her own firm at the programming department, who translated these into some kind of computer simulation model. How this happened was a big mistery for everybody, but finally she would receive a program back that she could demonstrate to the managers of the client.

This program showed the business processes that she had mapped some weeks before and simulated the behavior of the business processes as time passes by. She understood the usefulness of such a program, as it would allow someone to analyze the performance of an organization and you could even experiment with it, without the need to change something in the real organization. The problem was however, that each time she demonstrated the program, the managers almost immediately noticed mistakes in it. Each time, they had to validate the mapped processes again, and often within a few minutes, mistakes were found and corrected. Then, Christina would send back the updated model to the programming department, and after several days or even weeks, she would receive an updated program. And again, mistakes were found so quickly, that it had to be send back, again, and again, and again.

"If only, I could in some way model the business processes myself, and demonstrate a simulation to the client's managers immediately. That would be so much more convenient", Christina thought time and time again. "That way, they could immediately provide feedback if they see some incorrect behavior, and I could change it right there, with them." But now, so much time was wasted as most of the time while the programmers were performing their magic, namely transforming her mapped processes into a computer simulation model, she was doing only some side-projects. However, she had no experience with complicated software, let alone with software programming.

"Like if that would ever happen. . . ", she thought, after which she shut-down her notebook and went to bed.

1.1 Background

Good understanding of business processes of an organization is essential when making organizational decisions or redesigning these business processes. Business processes are related to the concept that a product or service is the outcome of one or - more often - a series of activities. The business process serves as an instrument to organize these activities, and business process modeling is used to map and analyze these business processes [Wes07].

A distinction can be made between static modeling of business processes and dynamic modeling [BVCH07]. Static modeling tools often provide a graphical representation of the business processes, for example simple flowcharts, IDEF0 or BPMN diagrams. For dynamic modeling there exist business process simulation (BPS) tools, which provide the possibility to simulate and evaluate the dynamic behavior of business processes and the random behavior of resources [BVCH07].

BPS is an application of computer simulation, which itself is defined in [Ma01] as "a set of iterative activities to abstract, build and experiment with a computer-based model which mimics the dynamic behavior of a system". As this definition illustrates, the process

of analyzing business processes using simulation techniques involves several activities to be performed in an iterative manner. Through BPS a business analyst or management consultant is able to conduct 'what if' analyses. BPS allows namely to create insight in the dynamics of business processes and enables the evaluation and experimentation of business process modifications on the overall performance of the organization before actually implementing these changes.

In Figure 1.1 a simplified simulation project life cycle is shown which depicts the main activities and products of a simulation study. The 'real world' depicts the organization of which its business processes are analyzed. Through conceptual modeling a conceptual model (or in our case a business process model) is developed based on the business processes. This model is used as the input for the construction of a computer simulation model, as it contains a description of what happens within the organization. Often through many iterative steps, a simulation model is developed, verified and validated. Finally, an analyst can experiment with the model by adjusting model parameters, executing it with some simulation software and interpreting the output results as presented by the software. When after various experimentation and analysis sessions a solution is found for some organizational problem, the changes can be implemented in the real world.

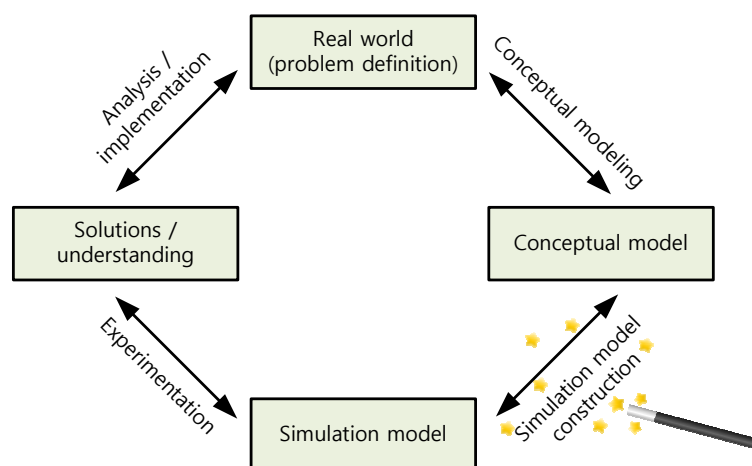


Figure 1.1: Simulation study life cycle

1.2 Research problem

An attentive reader probably noticed that in Figure 1.1 not only a generic simulation life cycle is drawn, but also a magic wand, next to the 'simulation model construction'-step. This wand depicts the main area of our research and the main problem which also initiated this research.

Many management consultants and business analysts rely currently on simple static process mapping methods ([BVCH07]; [MaP03]). This is not a problem on itself, as static modeling has often proven to be adequate for consulting projects. The usefulness of business process simulation (BPS) however, is confirmed by many researchers and

several experts within management consulting companies are as well aware of this. It offers namely various advantages compared to the more static methods. As mentioned before, an advantage of BPS is that it allows to experiment with business processes without influencing (and possibly jeopardizing) the real business processes and thus the organizational performance.

The reason for a lack of adoption and application of BPS within management consulting firms is not related to limited availability of simulation software. There are many tools available instead, both commercially licensed (sometimes however rather expensive), as well as free and open-source licensed, and often offering many features and possibilities for modeling, experimentation and analysis of business processes. The main reason for a lack of adoption is however that to develop a valid executable simulation model, much experience is needed. And even if someone is available with the adequate experience, the whole simulation model development process is often time consuming and costly [VD98].

A consulting firm which also noticed this problem and initiated this research, is Accenture. Accenture is an international management consulting company, specialized in IT consulting. The role of Accenture management consulting activities is generally speaking to advise and support clients with their business decisions and support possible business transformations. In the past years, Accenture Nederland undertook several BPS projects, but noticed that the projects were sometimes over-time, often labor intensive and several other issues arose during the simulation model development process.

To clarify their issues, we will briefly describe the development process. Business process models are developed by management consultants with adequate experience to understand and map the business processes of client organizations. After finishing these models, the consultants send these to the software programming department, named Accenture Technology Services (ATS), where software programmers develop the computer simulation models based on the process models. The development of the simulation model by the ATS-department is considered partly to be a 'black box': a business process model is send in. . . something happens, but what exactly is rather vague. . . and finally, after several days or weeks, a simulation model is send back to the consultants.

The received simulation model can now be *validated*. In simulation studies validation is considered to be the process of determining whether 1) a conceptual model is a reasonable representation of the problem (conceptual model validity), and 2) whether the outcomes produced by a corresponding simulation model are sufficiently accurate to be applicable on the real world (operational validity) [Sar04]. During the past projects, it often happened that when evaluating the correctness of the operational validity of the developed simulation models, together with client-side specialists, mistakes were found. This was sometimes due to misinterpretation of the business process models by the programming specialist, or mistakes were found in the original business process model itself. In both cases, these models were then updated by the consultants and sent back to the ATS-department, where an updated simulation model was developed. This process repeated itself many times and was according to involved consultants rather frustrating, due to the delays and the lack of knowing what happened inside the 'black box'. However, this is the only way how simulation models can be developed, as there are currently no tools available which support management consultants to easily develop a simulation

model of business processes themselves and directly experiment with these and analyze the outcomes.

However, this problem of Accenture is not a unique example. As we mentioned before, the development of a simulation model based on a conceptual model in general is considered difficult and time-intensive. This is partly due to the fact that conceptual modeling techniques are at best semi-formal and that there is a lack of continuity between conceptual and simulation model [CVS10]. This means that based on a certain conceptual modeling techniques, like for instance UML-diagrams, IDEF0-diagrams or BPMN-models, different simulation models can be developed. It depends largely on the simulation model developer and his interpretation of the conceptual model, how he specifies the simulation model. In other words, there is a large semantic gap between conceptual models and simulation models, which may finally lead to inaccuracies in developed simulation models [CVS10].

Currently undertaken academic research investigates the possibility of applying a Model-Driven Development (MDD) approach to overcome this semantic problem. The Model-Driven Development framework for Modeling and Simulation (MDD4MS) was developed to support the life cycle of modeling and simulation [CVS11]. Through the use of formally defined metamodels of the conceptual modeling language and the simulation language, a conceptual model can be (semi-)automatically transformed into an executable simulation model.

To conclude this section, we define the main problem which is covered in this research:

Research problem: *Management consultants are currently unable to model and simulate business processes, due to the complexity of simulation model development.*

1.3 Research objective

The main objective of this research is to find a solution for the problem mentioned in the previous section, namely a new way how a support tool can help management consultants to easily model and simulate business processes. As a proof of concept, our goal is to also develop a working prototype based on the developed design. Because this research is initiated by Accenture and performed as an external research project within Accenture, we have the possibility to evaluate the prototype with management consultants from Accenture. This will allow us to evaluate whether the design is indeed a possible solution for management consultants and feasible to develop.

With this research, we also want to approach the problem mentioned in [CVS10] regarding the semantic gap between conceptual models and simulation models and the inconsistencies that may arise when translating the first into the latter. The problem for consultants being currently unable to develop simulation models, relates to the difficulty of developing simulation models based on conceptual models. By defining a method which allows for the translation of a business process model into an executable simulation models in a rigorous manner and implementing this in a prototype, we also contribute to

solving the more general problem, namely how the semantic gap between the conceptual and simulation models may be overcome.

1.4 Research questions

Following the description provided in the previous sections about the research problem, we can now formulate the main research question:

Research question: *How can a support tool help management consultants in a new way to model and simulate business processes?*

1.4.1 Sub research questions

Several sub-questions can be derived from this main research question. Because this research is undertaken as an external research project within Accenture, this provides the possibility to study the problem more elaborately within Accenture.

Research question 1. *What are the limitations of Accenture's current business process modeling and simulation approach and how does this relate to limitations mentioned in literature?*

After the limitations are identified of the current modeling approach, we can involve management consultants to define design requirements that a solution should meet.

Research question 2. *What requirements should a new solution meet in order to support modeling and simulation of business processes by management consultants?*

The next sub question relates to the design of a solution for management consultants.

Research question 3. *What could be a new approach which enables business process modeling and simulation by management consultants?*

To answer this question, most aspects depicted in the simulation life cycle (shown in Figure 1.1) should be analyzed. To support management consultants with conceptually modeling the business processes of an organization, we should first understand how a consultant sees the business processes (which is considered to be his mental model). We then need to define a way how the business processes can be mapped in a conceptual model. This way of conceptually modeling the business process also allows us to define a method to support the transform a conceptual model into a simulation model. However, the main purpose of a business process simulation model is for a consultant to be able to experiment with it and analyze the generated output statistics. Due to the extent of sub question 3, we split it further more up in several sub-sub questions.

Sub question 3.1. *How does a management consultant see organizations and business processes?*

Sub question 3.2. *How can a management consultant's view on business processes be represented in a conceptual model?*

Sub question 3.3. *How can the translation of a conceptual model into an executable simulation model be supported?*

Sub question 3.4. *How can relevant output statistics of a simulation model execution be presented?*

Sub question 3.5. *How can a management consultant interact with a solution to develop a business process model and perform experiments with it?*

Finally, the following question should be answered as a proof of concept, namely whether the designed solution is indeed an adequate solution for our initially stated problem.

Research question 4. *Does the proposed solution enable modeling and simulation of business processes by management consultants?*

1.5 Research approach

Designing artifacts such as information systems, is a complex process which requires knowledge from two distinct paradigms, namely design science and behavioral science [HMPR04]. The design-science paradigm (which has its roots in engineering) focuses on the creation of artifacts through theories and methodologies, whereas the behavioral science paradigm focuses on explaining and predicting organizational and human aspects surrounding the analysis and design of artifacts. The Design Science Research framework, as proposed by Hevner et al. (2004) combines these two paradigms and will form the basis of this research [HMPR04].

Applying the Design Science Research framework will ensure that a designed solution is based on both academic rigor, as well as practical relevance. It also supports that both the design research process and the outcomes of the process contribute to the environment for which it is developed, as well as that it provides an academical contribution. In Figure 1.2 the framework is shown with the research cycles made visible. These cycles provide both the input for the design process and support the feedback loop to the environment and academical knowledge base.

Environment and Business needs The main identified need of the management consultants from Accenture is to have support for modeling and simulation of business processes. A solution for the consultants also contributes to Accenture in a more general sense. By being able to simulate business processes and experimenting with redesign suggestions, management consultants from Accenture are enabled to evaluate and improve the performance of client organizations in a rigorous manner.

Knowledge Base As modeling and simulation has been studied for several decades, much literature and studies can be found on it. Recent research, like for instance on a Model-Driven Development framework for Modeling and Simulation (MDD4MS), is

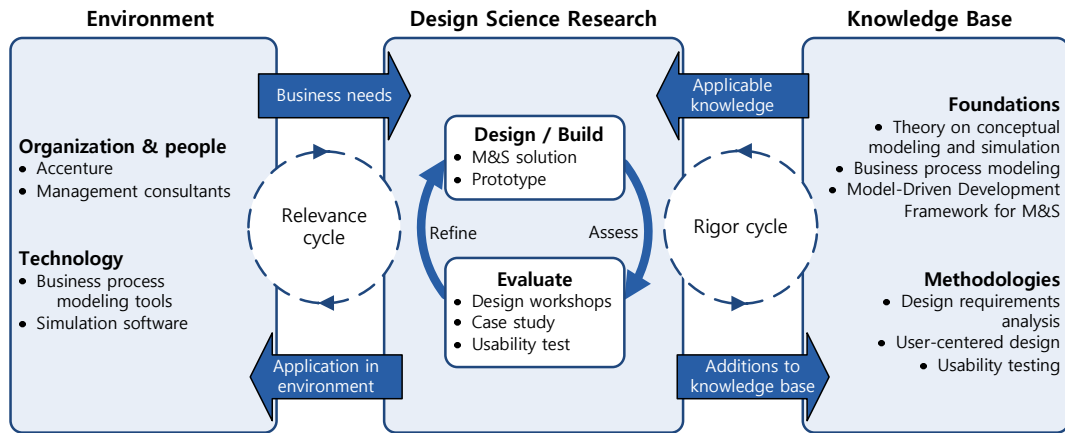


Figure 1.2: Design Science Research Framework (Adapted from: [HC10])

aiming at improving the applicability of simulation analysis. Our research will be based on academic foundations. It will however also contribute to the scientific domain of modeling and simulation, as it studies ways which allow users with limited simulation modeling experience to apply simulation.

Design Science Research The design process aims at developing a modeling and simulation solution for management consultants, as well as a working prototype. Several design and evaluation workshops with management consultants will secure that a solution connects with the management consultant's way of thinking and working.

Seven clear guidelines are formulated for effective design-science research [HC10]. These guidelines provide the foundation of this research and each will be evaluated at the end of this research. The guidelines are:

1. *Design as an Artifact*: Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation
2. *Problem Relevance*: The objective is to develop technology-based solutions to relevant business problems
3. *Design Evaluation*: The utility, quality and efficacy of a design artifact must be demonstrated through well-executed evaluation methods
4. *Research Contributions*: Design-science research must provide clear contributions to design foundations
5. *Research Rigor*: Design-science research relies on application of rigorous methods for construction and evaluation
6. *Design as a Search Process*: Search for an effective artifact requires utilizing available means to reach desired ends
7. *Communication of Research*: The research must be presented effectively to all involved stakeholders

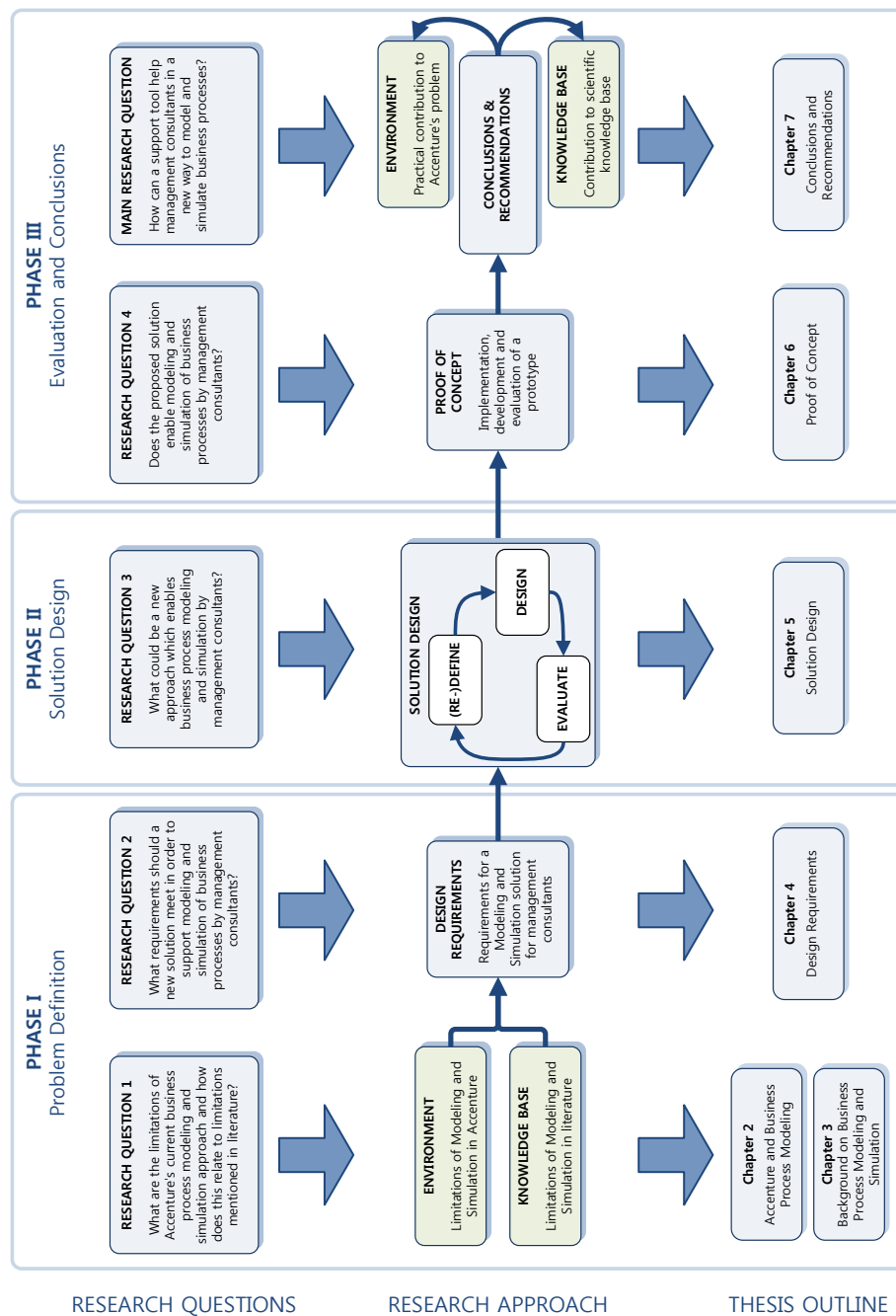


Figure 1.3: Research approach: Transformation from research questions to thesis chapters

In Figure 1.3 the research approach is shown how we suggest to transform the formulated research questions into thesis chapters through our design approach. Generally, our research consists of three main phases, namely the problem definition phase, the solution design phase and the evaluation phase.

During the problem definitions phase, we will elaborate on the problem, namely

how business process modeling and simulation is currently undertaken in Accenture and what the noticed limitations and issues are. The main findings will be discussed in Chapter 2. We will then move on to scientific literature where we try to define and elaborate on business process modeling and simulation as how it is discussed in scientific works. Based on the limitations also mentioned in literature, we will make a comparison between Accenture's case and the general scientific knowledge, to underline the academic relevance and applicability of our research within Accenture. This allows us to answer research question 1 in Chapter 3.

Next, design requirements need to be defined to support the design of a solution. In Chapter 4 we will elaborate on research question 2 by involving management consultants to state requirements that a final solution should meet. We will relate these requirements to three more general defined requirements which a solution like the one proposed in this research should meet, namely that of usability, usefulness and usage [KS08].

In the following phase we will answer research question 3 by designing a solution which allows management consultants to easily model and simulation business processes. The solution will however not be designed in isolation, as we suggest to apply a user-centered design (UCD) approach. The main goal of a UCD approach is to increase the likelihood that a designed and developed artifact is found usable by its end-users [Mag01]. We will do this by actively involving management consultants from Accenture in the design and evaluation process. The findings and outcomes of the design process will be discussed in Chapter 5.

In the last phase, research question 4 will be answered by evaluating the developed design. As a proof of concept and to support the evaluation, we will develop a working prototype. Using this prototype a case-study will be performed, to confirm that a solution is able to model and simulate real business processes. Next, based on a usability-test with management consultants from Accenture we will assess whether the solution is found useful and usable by the end-user. The development of a prototype, as well as the evaluation, will be discussed in Chapter 6.

1.5.1 Research methodologies

For this research we propose the application of several research methodologies or instruments, namely: *Desk research*, *Interviews*, *Design Workshops*, *Prototyping*, *Usability testing* and finally a *Case Study*. A motivation for each methodology is given below.

Desk research Desk research relates to summarizing, collating and synthesis of existing research. We will mostly apply desk research during the first phase of this research to answer research question 1. More specifically, we will first study available material from Accenture (e.g., presentations, internal web-resources, ... etc), to understand the current way of business process modeling. Based on articles and books about business process modeling and simulation, we continue and try to understand the basics of business process modeling and the general limitations of business process simulation. To find related articles, we will search in research databases such as: Google Scholar, Elsevier's Scopus, IEEE explorer and Science Direct.

Interviews We will have interviews with management consultants from Accenture, to understand the current way of business process modeling and simulation better. It will also provide us with a better understanding of the limitations and issues of Accenture's current approach. An interview is considered to be an efficient technique to acquire information about a specific topic. Interviews will be held with management consultants who have experience with business process modeling or have also participated in simulation projects. Preferably, these consultants are from different Accenture's service lines. This will provide us with a broader image of the corporation and give more insight whether there are differences within the firm with regard to how business processes are modeled. Interviews will also be held to define the design requirements, in order to answer research question 2.

Design workshops To increase the likelihood that a designed artifact is found usable by its end-users, we apply a user-centered design approach by actively involving management consultants from Accenture (as they are the end-user of a solution) in the design process. Active end-user involvement will be facilitated through several design workshops. Goals of these design workshops is to get a better understanding of the consultants, to understand how they see business processes, to evaluate design proposals and to make design decisions. These design workshops will contribute to answering research question 3.

Prototyping We will develop a working prototype as a proof of concept of our designed solution and to answer research question 4. Opposed to for instance written design specifications, a prototype enables evaluation of a proposed design in context. As the form of a designed artifact is the solution to a problem, the context *is* the problem [NJ82]. Evaluation of the prototype within the real context (i.e., a management consulting project) is not feasible due to limited amount of time available and a lack of business process simulation-related projects within Accenture. However, evaluation of the usability of a solution with management consultants and examining a sample case with it, allows us to proof the suitability of the solution for management consultants, as well as the feasibility to continue research on developing a final solution.

Usability testing A wide range of usability evaluation techniques have been proposed, some of which can be applied in the early stages of the design process, whereas other can only be performed after a design or prototype has been implemented [IH01]. A solution should be considered usable by its end-users, namely management consultants. We will apply usability testing to evaluate the usability of the solution through a working prototype. The thinking-aloud protocol will be applied for the usability testing during which the end-user will interact with the prototype, performs a series of activities and mentions out loud what he is currently thinking. This allows us to get a better understanding of what can be improved and to what extent a user is able to understand and use the solution.

Case study Finally, as a proof of concept, we will use the solution based prototype to model and simulate real business processes. This allows us to partly evaluate the objective of this research, namely whether business processes can be modeled and simulated in an easy manner. The objective of our research includes the use of the solution *by*

management consultants. However, this seems impossible due to the limited amount of time available, as well as the early-development state the prototype will be in. Regarding the sample case, we will use a business process model made by management consultants which represents the business processes of a telecom operator. We use the prototype to model the represented business processes and simulate these.

After a few days working on the new project, Christina received an email. The sender was an M.Sc.-student from Delft University of Technology who recently started working on his final M.Sc. graduation research project. He introduced himself politely and asked her if she could help him with his research during the following months.

"My research", as the student introduced in his e-mail, "is about designing a new way that allows management consultants to model business processes and simulate these themselves, without the direct need of a software programmer." Apparently, she was not the only one who noticed that there was a problem with the current modeling and simulation approach. "Because the end-users of a possible solution are management consultants like you", the student continued, "I would like to involve you during the whole research, design and testing process." He explained that for a solution to be finally successful and usable, end-user involvement during the design process is crucial. In the remainder of the email he briefly described what he expected from here, if of course she wanted to cooperate. He suggested that it would be nice to meet in the following days, so she could explain in more detail what the issues are with the current way of working. Later, they would meet to discuss what she thought would be useful characteristics and features of a new solution.

Although she knew that she would be quite busy these following months on her new project, Christina got excited and replied to him that she would be more than willing to help him out. This was exactly what she was thinking about during the past months and finally someone seemed to be able to read her mind.

Accenture and Business Process Modeling

2

Goal of this chapter is to identify the main issues and limitations of the current business process modeling (BPM) and simulation approach, as how it is currently undertaken within Accenture. This in order to clarify the problem and the context of the problem better. We will do this, by first introducing Accenture briefly in Section 2.1. Then, we will look in Section 2.2 at Accenture's current BPM projects. First, we will explain how different service lines perform BPM in Section 2.2.1. Next, we will elaborate in Section 2.2.2 on the past simulation-related projects that were undertaken. Finally, in Section 2.3 we will present an overview of the identified limitations of the current approach.

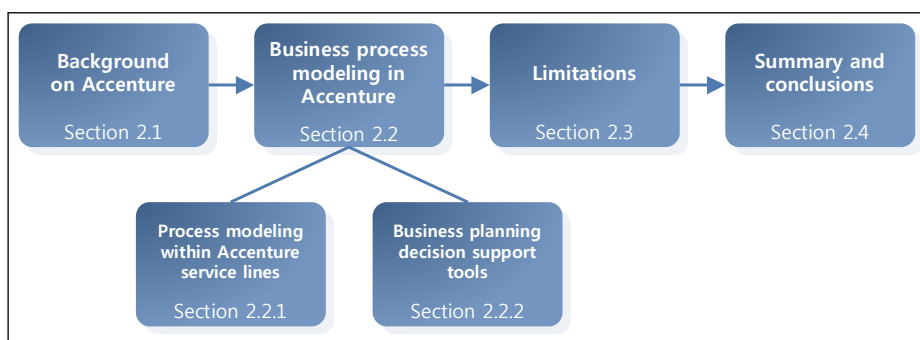


Figure 2.1: Outline Chapter 2

2.1 Background on Accenture

Accenture was formed in 1989 after a division split from the accounting firm Arthur Andersen and was first known as *Andersen Consulting*. In 2001, the name of the firm was changed to its current name. Accenture is currently active in the fields of management consulting, outsourcing of services and technology consulting, development and integration. At the moment of writing, Accenture has approximately 211 000 employees and serves clients in more than 120 countries [Acc11a].

The role of Accenture management consulting activities is generally speaking to advise and support clients with their business decisions and support possible business transformations. Accenture is specialized in delivering its services to almost any type of industry. The different industries are grouped by Accenture into five main operating groups and an overview is given in Table 2.1. Besides the subdivision in industries, Accenture's management consultancy activities are also subdivided in service lines. Some service lines

are Finance & Performance Management (F&PM), Process & Innovation Performance (P&IP), Talent & Organization Performance (T&OP), Supply Chain Management (SCM) and Strategy.

Table 2.1: Overview Accenture's operating groups and industries

Operating group	Industries
Communications and High Tech	Aerospace and Defense, Communications, Electronics and High Tech, Media and Entertainment, Life Sciences, and Public Safety
Financial Services	Banking, Capital Markets, Insurance
Products	Automotive, Building Materials, Consumer Goods and Services, Retail, Chemicals, Freight and Logistics, Travel, Infrastructure and Transportation, Industrial, Equipment, and Airline
Resources	Energy, Forest Products, Utilities, Metals, and Mining
Government	Defense, Customs, Health, Health & Public Service, Border Management, Non Profit, Postal, Human Services, and Public Transportation

2.2 Business process modeling in Accenture

Every management consulting project has different characteristics and is focused on different business aspects. To accommodate these projects, service lines offer different techniques to approach business problems, but there are also common techniques which are used throughout the company. Accenture teaches for instance its consultants how to model and analyze business processes through participative training courses and online learning courses. Two types of process models that are developed by consultants are so called *as-is* business process models and *to-be* business process models. [Acc11b].

The purpose of an *as-is* model is to understand a clients' *current* situation and to give insight in how an organization currently operates. This is done by collecting information about the business processes and mapping these in a graphical model. When analyzing developed *as-is* models, consultants may look for possible bottlenecks in business activities, such as places where work backs up and may cause delays, or inefficient areas within processes that could be removed or relocated to improve the overall performance of an organization.

To-be models are models of business processes that do not exist yet, but depict how the processes will be or could be in the future. Business processes changes may occur due to for instance the integration of a new business information system within an organization, or due to an performance optimization project within an organization. A *to-be* model supports these changes by defining exactly how the processes are intended to look in the future, and thus providing a way to communicate and debate the changes with all involved stakeholders. When a *to-be* model is developed, attention is paid for instance to which activities and responsibilities can move between business process' participants,

which process parts could be automated, or how the processes can be reassigned or reordered.

The *as-is* and *to-be* business process models can be placed under the header of *descriptive* or *static* models. Descriptive models are meant to portray business activities, their dependencies and business rules of an organization in a static manner, i.e., without including the time-related aspects and business uncertainties which may have an influence on an organization and its performance. Active models, on the other hand, capture the dynamic characteristics of business processes by adding real-world uncertainty and effects of time on the processes and flows of entities. Active modeling is meant for business processes which are too complex to be captured in static process models [Acc11c]. In the following two subsections, we will look more closely at how business processes are modeled in two different service lines, providing also some background on these service lines, and on to what extent these active models are currently developed by Accenture.

2.2.1 Accenture's service lines and business process modeling

As mentioned, there are some differences between the service lines with respect to applied techniques and how business processes are modeled. To get a better general understanding of how management consulting projects are organized and how business process modeling is currently undertaken, two expert-interviews were conducted with management consultants from different services lines. The two service lines in which business process modeling is frequently applied as part of management consulting projects, are the Talent & Organization Performance (T&OP) service line and the Process & Innovation Performance (P&IP) service line. For this reason, from both service lines an expert was interviewed.

Talent & Organization Performance (T&OP) service line Projects undertaken within the T&OP service line focus on improving the performance of Accenture's clients related to their workforce productivity [Jan11]. The projects that are undertaken within T&OP are subdivided into four categories, namely *Change Management*, *Human Capital and Organization Effectiveness*, *Human Resource and Talent Management* and *Learning and Collaboration*. Typical projects may variate from training client's workforce to adopt a new Knowledge Management System, or providing means to a client of training future organizational leaders.

Regarding process modeling within T&OP, a distinction is made between "process design" and "process improvement". Process design focuses on the design of new business processes when for instance an information system is introduced, while the latter focuses on analyzing and improving the current business processes. To support the design or re-design of business processes, *as-is* models are considered useful but are not always made. This depends namely largely on the actual client's situation, on the time available for the project and on the objective of the project. Sometimes full insight in the current business situation is not available which limits also the possibilities of *as-is* process modeling. Due to these uncertainties, consultants largely rely on *best practices* or *common practice*. These are examples of processes which were proven to be successful in similar projects in the past and are thus re-used. Knowledge about these best practices is acquired

through a database called the “Accenture Business Process Repository”, but also by contacting other management consultants and specialists who have participated in these past projects. Figure 2.2 shows the concept of the use of reference models. It also shows that *as-is* processes are only modeled in 15 to 20% of the projects. In Figure 2.2 a distinction is made between strategic processes (for which *as-is* models are developed), and other processes (for which no *as-is* models are developed).

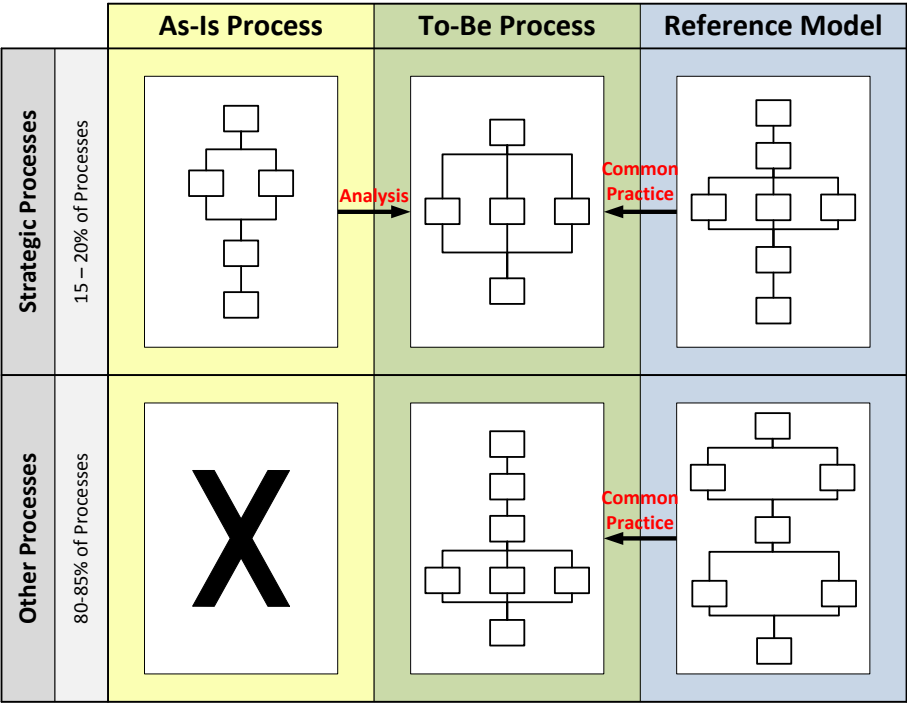


Figure 2.2: Best practices [Accenture, 2011]

An organization structure design project consists of five phases, namely the organization analysis phase, the design phase, the building phase, the testing phase and finally the deploy phase, where the designed structure is implemented in an organization. There are two approaches used for the design of an organization structure, namely the top-down approach and the bottom-up approach. The top-down approach focuses on the overall business strategy, and works down to the required capabilities and design criteria for the new organization structure, to finally come to a new organization structure.

top-down
organization design
approach

bottom-up
organization design
approach

The second approach, the bottom-up approach, focuses on first designing the new processes and activities on the lowest, most detailed level, and then move up to finally design the organization structure, regarding the teams of employees and deciding upon number of employees per team. The actual design of the lowest level of organizational processes occurs in a top-down manner. First the highest level (“level 1”) process models are developed and moves down to the lowest, most detailed level (“level 5”), which consists of the activities and processes. Process modeling is done differently depending on the project and the availability of process descriptions by the client. Sometimes, full level 5 process models are available in a detailed format (for instance developed using

tools like for instance ARIS or Microsoft Visio). In other cases, only levels 1, 2 and 3 process models are available by the client. The lower-level models need to be developed by Accenture consultants.

After these detailed process models are completed, the organization is designed in a bottom-up manner, starting with designing organizational roles by grouping several activities into distinctive roles. Based on these roles, jobs are then designed to understand the organizational impact of the implementation of the process activities. Then the actual team structure is described per business unit, based on the designed jobs. And finally, the number of employees (full time equivalents) per team is calculated, which is known as the *workload size*. The workload size is calculated using spreadsheets based on the estimated activity times and some other factors. The estimated activity times follow from a comparison of the current *as-is* processes and the designed *to-be* processes. To what extent the calculations fit the final business structure, and not cause issues like bottlenecks, can only be seen after the processes are implemented. According to the interviewed expert, a useful purpose of active modeling within organization design projects, could be to use simulation to analyze whether the estimations are correct and evaluate how choices for workload size will workout on the organization. However, this is not done yet. The uncertainty whether the workload size estimations are correct, will remain until a new organization design is implemented and evaluated.

Possible use of
simulation in T&OP

Process & Innovation Performance (P&IP) service line Projects that are undertaken as part of the P&IP service line focus on developing and sustaining an organizations operational, process, and innovation effectiveness. A division is made of the activities within the P&IP service line, in *Operational Performance*, *Process Performance* and *Innovation Performance*. The main methodology (in approximately 80% of the cases [dJ11]) used within the P&IP service line is Lean Six Sigma, which is a combination of the *Six Sigma* business management strategy, and Lean manufacturing. The main process improvement activities are based on three characteristics, namely simplicity, flow and discipline. With *simplicity* is meant that business processes ought to be simple by removing complexity; the *flow* characteristic focuses on reducing waste in the business processes (so making the processes more lean); and *discipline* relates to improving the attitude towards business process changes, for instance with regard to the behavior of the employees.

There are five distinct phases which describe a general P&IP project, namely the Define phase, the Measure phase, the Analyze phase, the Improve phase and finally the Control phase. The first three phases relate to the current situation, during which current business processes are mapped in *as-is* process models. The last two phases relate to the designed future situation, whereby *to-be* process models are developed and changes are implemented. During the define phase of a project, the client decides on its ambitions and targets. A client may for instance want to realize a high customer-satisfaction ratio by minimizing the delivery times of a service or product. During the measure phase, the business processes are mapped in value stream maps using brown-paper and Post itTM notes and activity times are then determined. A process is then broken-down several times into smaller activities (and the activity times are determined as well for these processes), until a level of detail is realized which is considered adequate to make suggestions for

improvement and which clarify the main reasons for organization inefficiencies. Next, the causes of the inefficiencies are set out against the effort needed to solve these. Solutions which require the least amount of effort and have a high gain on improving the efficiency (the so called “quick wins”) are then identified, documented and possibly implemented.

The use of best practices is also common in P&IP to re-design the current processes. This is done by using the business process repository and by contacting experienced consultants. Although process models are developed as part of P&IP projects, the use of more advanced business process modeling tools (e.g., ARIS) besides the use of brown-paper and post-its, does not occur often. The possibility to use active modeling to evaluate what suggested business process changes may actually have, is also not used in P&IP. It was mentioned, that although new designed processes should be an improvement of the current situation, until the actual implementation it is unknown whether they actually lead to improvements.

Possible use of simulation in P&IP

2.2.2 Accenture’s decision support tools and projects

The use of business process simulation, as part of the business process analysis phases of for instance T&OP and P&IP service line projects, is not widely adopted within Accenture. There were however several projects in the past that relate to business process simulation, namely those that involved the development of Business Planning Decision Support Tools (BPDST). Since 2008 Accenture developed decision support tools for several organizations which gave the organizations the possibility to analyze what impact certain business decisions have on for instance their performance. The organizations received a tool with which they got control over the analysis of their business processes and performances, and thus their own decisions and operations.

One project was the development of decision support tool for a large telecom operator. The tool supported the sales and operations planning of this organization. It provided the telecom operator with the capability to forecast for a two-year time horizon what impact certain changes (e.g., market conditions, marketing promotions, resource availability, project, . . . etc) would cause on the business workload within the 21 different departments. Another project Accenture undertook was also for a telecom operator to support decisions with regard to the roll-out of fiber-optic cables in The Netherlands of over 100 administrative areas. Through this tool the telecom operator could gain insight in the impact of certain factors (e.g., growth in demand, capacity, delays through weather conditions, . . . etc) on specific indicators such as financial and operational performance (e.g., lead times, inventories, . . . etc). The purpose of this tool was to help the organization develop robust operational and tactical business plans and continuously improve the speed and quality of analysis and decision making.

A third project was the development of a tool to simulate the supply chain relationships of both Dutch and other European parties. This tool enabled the client of Accenture to forecast – for a period of five years – the expected demand and behavior of the other supply chain parties (e.g., raw material suppliers, manufacturers, distributors and retailers). A fourth project was the development of a workforce planning tool for the Human Resource (HR) department of Accenture itself, to gain, more control over the planning of Accenture’s workforce, a tool was developed to support workforce related decisions over a five-year

period. Aspects like for instance recruiting new employees and transfer and attrition of current employees, and financial aspects of these decisions were included in the tool. Based on the expected or planned growth figures, or possible other changes in corporate strategy, scenarios can be executed to forecast the impact of these changes or decisions on predefined criteria. The fifth project was for a cigarette manufacturer and is similar to the workforce planning tool developed for the HR department of Accenture.

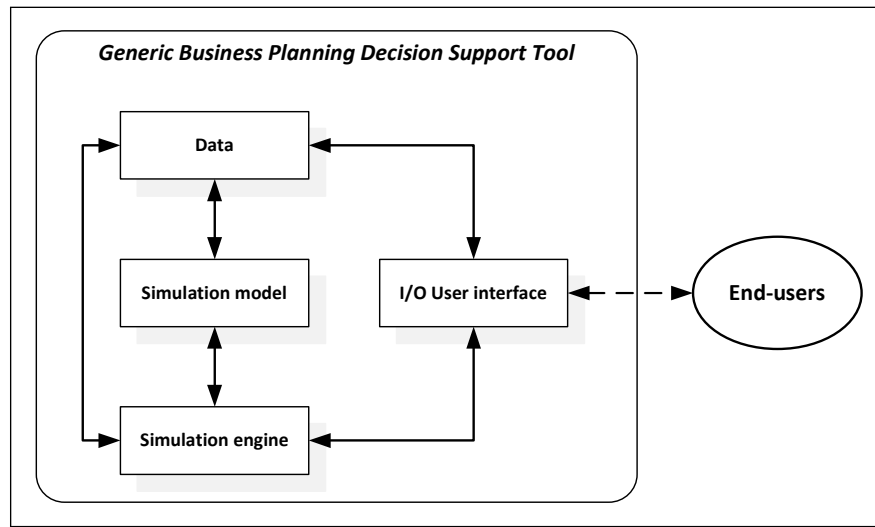


Figure 2.3: Generic BPDST components

The decision support tools as developed by Accenture generally consist of four main components, namely the simulation engine, the simulation model, the interface for user and data input and output, and the actual process data. In Figure 2.3 an overview is given of the relations of these main components. A customized user-interface is provided to support the interaction between the end-user and the decision support tool, for instance to specify scenarios which the user wants to analyze, and to present the results from an analysis. The simulation model is the main component of the decision support tool which describes the operations and logic within an organization. The parameters of the simulation model are acquired from this user data, as well as data gathered about the business processes during the development of the model. Through the user-interface the end-user can execute the simulator engine to analyze a scenario. The results following from the execution of the simulation model are finally presented to the end-user in the form of tables, histograms, geographical maps, . . . etc.

An overview of the software which was used to accommodate the functionality of the decision support tool is shown in Table 2.2. As can be seen in this table, DSOL was used for all the projects by providing the forecasting capabilities to the tools through the use of simulation models. DSOL is a simulation tool developed at the Delft University of Technology and allowed Accenture to develop a customer specific support tool. Microsoft Excel was used to provide the means for the customer to enter specific parameters regarding the scenario's they are interested in analyzing, and to return the results of the analysis.

Table 2.2: Past BPDST projects

Project	Simulation Engine	User-input interface	Results interface
Telecom - Sales & Operations	DSOL	Excel	Excel
Telecom - Fiber roll-out	DSOL	Excel / Java	Excel
Mushroom Supply Chain	DSOL	Excel	Excel
Workforce Planning - Accenture	DSOL	Access / Java	Excel
Workforce Planning - Cigarettes	DSOL	Excel	Excel

2.2.2.1 Accenture's process of developing decision support tools

To gain more insight in the actual development process of the decision support tools, several expert-interviews were held within Accenture with consultants who participated in these projects. Figure 2.4 provides an overview of this development process. The figure contains the main roles (at Accenture's side), the main process activities, the main inputs of information and the feedback-loops during this iterative process.

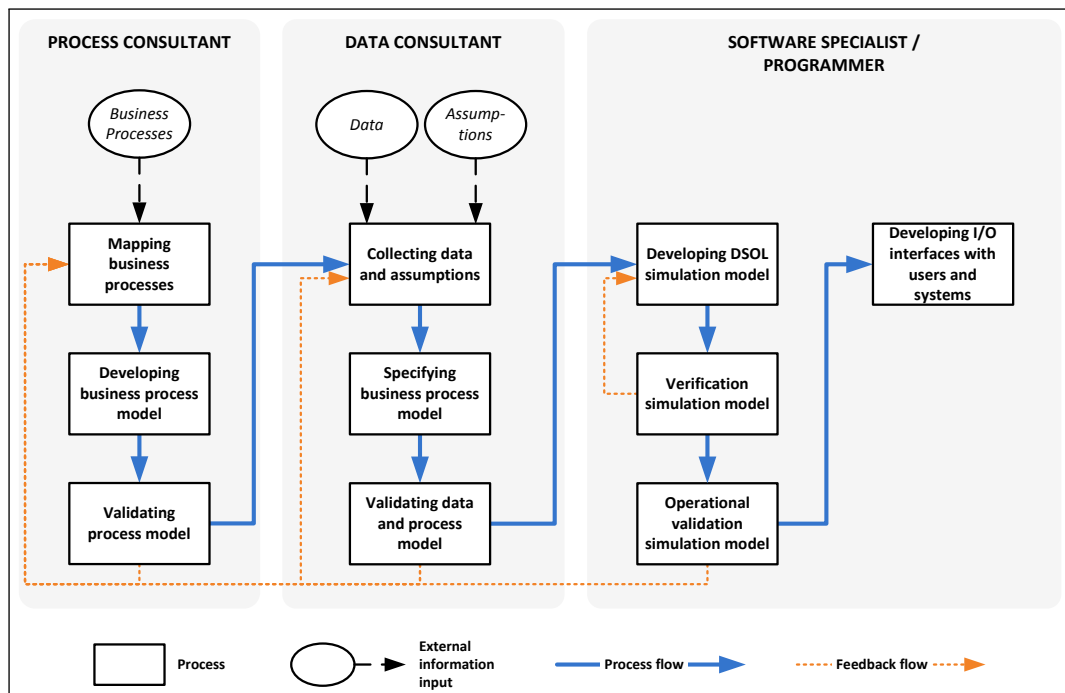


Figure 2.4: Current business planning decisions support tool development process

Several main roles can be identified within a decision support tool development project team. Besides a general project manager, there are the process consultant role, the data consultant role and the software specialist role. The reader should note that these roles can be fulfilled by multiple employees. For instance, there can be two consultants

who both have the same role, and one of them can also have several roles, or only the capability to do part of a role.

The simulation model is the main component of the decision support tool which describes the operations and logic within an organization. To be able to develop this model, information about the business processes is first collected and documented by a process consultant, together with the Business Unit Manager and Sub-Unit Managers of the organization. They provide the descriptions of the organization processes which will be documented using for instance spreadsheets. After the processes are mapped, graphical business process models are then developed, using some model representation. For instance for the development of the decision support tool for the sales forecasting of a telecom operator, the Visual Paradigm software was used. Visual Paradigm uses the Business Process Modeling Notation (BPMN) for modeling of business processes. When these models are finished, they are presented to the Business Unit Manager and Sub-Unit Managers, to validate whether these models represent the actual business processes. The models are often inaccurate and more details are needed about the processes. Developing the business process models is an iterative process and so this process may repeat several times until a valid business process model is developed.

The next step in the development process of a simulation model is specification of the processes. This is done by acquiring actual data about the processes and assumptions in case when business decisions are made. Examples of data are the durations of the identified activities or the arrival times of for instance products. Business decisions are preferably specified based on a known parameter, for instance: "in case of the arrival of a product of type A: perform activity X. else: perform activity Y". However, this information is not always available. In those cases decisions are specified using an assumption, like for instance: "for 70% of the arriving products, activity X is performed. activity Y is performed for the other products". The data is requested by the data consultant and employees on the client-side of the project will support this phase by collecting the data. Sources for this data may be estimations from data dumps of SAP systems or process reports. In case the data is not available, estimations are made by a specialist with enough knowledge about the process. This phase is also an iterative process. During the process of data and assumption gathering, the information and the specified business process model are validated. New information may become available during the gathering process, which may lead to redesigning part of the business process model.

After several iteration rounds, the simulation model can be developed. Because of the choice for DSOL by Accenture for its decision support tools, a computer specialist with Java programming experience is then involved. DSOL does not have a graphical user interface (GUI) to develop and specify the simulation model, so the Java-syntax of the simulation model is written by the computer specialist. Based on the information provided in the business process model and the acquired data and assumptions, the programmer develops an executable simulation model. The model is verified continuously by the programmer for correct functionality and behavior. Finally, when the model is finished, the process and data consultant evaluate the behavior and outcomes of the model with the client-stakeholders, to validate whether the model and dynamic behavior of the model correspond with the actual business processes. Often, the model appears

not to be valid and more accurate information is needed, or re-adjustments need to be made in the business process model, after which the development process is repeated. When finally a working and valid simulation model is developed, the input and output interfaces are developed between the tool and the end-user.

The previous paragraphs provide a brief overview of the development process of business planning decision support tools. For a detailed overview of the assignment of responsibilities in a BPDST project, see Appendix A.

2.3 What are the limitations of the current approach?

Based on interviews held with consultants that took part in previous BPDST development projects, several issues were identified that caused delays and problems in the development process of the decision support tools. These issues are categorized based on whether they relates to the *business process model development*, *specification of the business process model* or the *simulation model construction*. An overview of these inconveniences is given below.

Issues related to business process model development

- 1. Different process modeling practices within the service lines**
Project teams are formed by consultants from different service lines of Accenture. Within each service line, different process mapping and modeling practices are considered as standard. To guarantee consistency of the final business process model, the process modeling techniques and interaction process between consultants from different service lines should be streamlined during the project, which may cause delays in the project planning.
- 2. No common and agreed understanding of business processes at client-side**
Different client-side specialists (e.g., Business Unit Managers, Sub-unit business managers, etc) may describe processes differently, based on their understanding of the overall process flows, and based on the subjective importance they give to their own processes.
- 3. Unknown on beforehand whether full process descriptions are available**
During some projects the client originally stated that the business processes are already documented, thus no time is needed to map these. Later on, it was found by the consultants that these process descriptions were not documented at all, or inadequate to develop a simulation model from it.

Issues related to specification of business process model

- 1. Project team members interpret tasks differently due to different backgrounds**
Due to a background to different from other team members, with regard to the different service lines of Accenture, as well as a lack of understanding of simulation modeling and analysis, some management consultants assigned with the task of

acquiring data sometimes aggregated this data in formulas and models. To them it seemed more useful to do this because it was a common practice within their service line. For the purpose of developing a simulation model however, these aggregations are often useless, and thus new data had to be re-acquired.

2. Data specialist at client-side understand processes differently than business managers

Static invalidity of the process model often arises when business processes are being mapped for the development of the conceptual process model, and when data is acquired about these processes. Frequently, after the business processes are being mapped by the business process consultant, they appear to be different or even non-existing when concrete data about these processes is collected by the data acquiring consultant.

3. Client-side specialist often have difficulties estimating figures

Client-side specialists often have difficulties with estimating figures with regard to for instance process duration times and assumptions (i.e., in case of decisions that are made, what percentage of entities move to different processes). This is also partly due to the subjective importance they give to their own processes.

4. Unknown on beforehand which data for business process specification is available

It is unknown on beforehand whether certain information and data to specify the business processes is available by the client, or should be all acquired during the BPDST development process.

Issues related to simulation model development

1. Software specialists sometimes need to be educated with simulation and DSOL related knowledge

To develop a simulation model, the software programmer needs to have an understanding of the concept of modeling for simulation and simulation analysis. In case a programmer lacks this knowledge, he needs to acquire a common understanding of simulation practices, as well as how to use the DSOL simulation library. Depending on the level and skills of the programmer, this can consume a certain amount of time.

2. Time consuming to translate business process models into simulation models

Operational validation of the process model can only be undertaken, after the process model is translated into an executable simulation model. Developing a simulation model is a time consuming process, and the speed and quality this can be done depends largely on the expertise and understanding of the process models of the programming specialist. Many adjustment rounds are needed to come to a dynamic valid process model.

2.4 Summary and Conclusions

As was mentioned in the introduction of this chapter, goal of this chapter was to shape an image of Accenture as a management consulting firm and to better understand what kind of project it undertakes. The second goal was to provide insight into how and to what extent business process modeling and business process simulation is performed within Accenture.

Several expert-interviews were conducted and it became clear that for many projects modeling of business processes is a common part. The models that are developed by consultants can be representations of the current processes of an organization (*as-is* models), or designs of new business processes (*to-be* models). For the design of new business processes, consultants often rely on best practices or the experience of other consultants. Although this appeared to be successful in many cases, it is unknown to what extent and on what scale adjustments had to be made after the processes were implemented, or how many processes had to be completely redesigned due to wrong estimations of bad assumptions. A consultant from the Talent & Organization Performance service line mentioned in an interview, that dynamic analysis through simulation seems very useful to decide for instance on the sizes of teams in a newly designed organization structure. He also mentioned that simulation could be useful to evaluate the effects that business process changes might cause on the overall performance of an organization.

The second part of this chapter focused on the analysis of projects involving the development of decision support tools for clients. These are the only projects within Accenture Nederland that involve the development of business process simulation models and that use simulation tools to mimic the dynamics and uncertainties within business processes. Several interviews were conducted with consultants that were part of these projects, and various issues were identified that arose during these projects. For instance, because modeling for simulation is not common for Accenture consultants and because project teams consisted of members from different service lines, the tasks assigned to project members were not always clear to them. Another aspect that was mentioned, was that the development of the simulation model was too time consuming. The consultants had to rely on a software specialist with both programming and simulation experience for the development of the simulation model. At the start of some projects, the software specialists had adequate programming experience, but no knowledge about simulation. This resulted in that he first had to be educated to understand the concepts behind simulation and to understand what was expected from him. Due to this distribution of tasks, regarding the consultants who modeled the business processes, and the software specialist who developed the actual simulation model, the efficiency of business process modeling and validation of the simulation model was low.

Due to the time available for undertaking this research, not all issues mentioned in Section 2.3 can be tackled. Some issues relate strongly to the client situation and are considered outside the boundaries of this research. Other issues are more process related, for instance regarding the project team composition. Although this will be partly covered later on in this thesis, most attention will be on the last two issues, namely related to the development of simulation models: How can this be improved in a new and innovative

way. In the following section we will elaborate on how business process modeling and simulation is discussed in literature and the issues mentioned.

Background on Business Process Modeling and Simulation

3

The previous chapter proposed an overview of business process modeling and simulation practices in Accenture, as well as the main limitations of the current approach. To guide this research to a solution which enables business process simulation by management consultants, this chapter will elaborate on academic literature regarding modeling and simulation.

This will be done by first looking at the general motivation for simulation and modeling in Section 3.1. Next, in Section 3.2 we will look at how business process modeling is generally applies, as well as at a generic modeling and simulation life cycle. This will be followed by an overview of limitations of the current modeling approaches. A link will be made between generic modeling and simulation approaches, and Accenture's modeling approaches in Section 3.3. This will finally lead to a definition of the problem space.

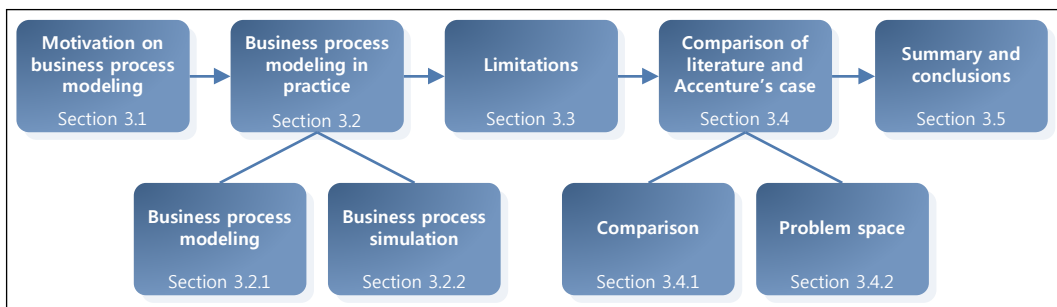


Figure 3.1: Outline Chapter 3

3.1 Motivation for business processes modeling and analysis

Most business organization can be considered being a part of a larger environment containing similar organizations. These organizations may vary from being competitors of each other, or partners, or may be related in any other way to each other. However, these environments are not static. Due to the ongoing changes within an environment, an organization needs adopt itself to these changes. Decisions need to be made continuously in order for an organization to remain competitive or at least operate effectively [DVvE03]. However, what an organization decides and how that decision is actually made, can be considered as solving a problem. To clarify this statement, we look at the definition of problem. Ackoff (1981, page 20) defined a problem as following: *“By a problem we mean a situation that satisfies three conditions: First, a decision-making individual or group has alternative courses of action available; second, the choice made can have a*

significant effect; and third, the decision maker has some doubt as to which alternative should be selected." In other words, a problem is a situation where there is a problem owner who needs to make a decision among various alternatives, but is unsure which alternative to choose. We consider this being the case for an business organization which needs to cope with the changing environment.

A distinction can be made between structured and ill-structured problems. Soll (1982), considers a problem to be structured when it meets the following three conditions: 1) the set of alternative courses of action or solutions is finite and limited; 2) the solutions are consistently derived from a model of the problem situation that has been validated empirically and shows good correspondence with reality; and 3) the effectiveness and/or the efficiency of the courses of action can be numerically evaluated [Sol82]. A problem is considered to be an ill-structured problem on the other hand, depending on how much is known about the following three basic components, as defined by Young (1984)(adopted from [Bot87]):

1. Objectives: in ill-structured problems not all objectives may be known at the outset; multiple objectives exist rather than one, and the trade-offs or relative utilities of the objectives are largely unknown.
2. Outcome-affecting variables: in ill-structured problems the identity of all of the important variables (both controllable and uncontrollable) that affect the outcomes may not be known at the outset of the decision process, and therefore complete models cannot be specified in advance.
3. Relations between affecting variables and outcomes: in ill-structured problems these relations are not all well-known in advance, or they may vary according to different plausible assumptions.

Bots (1987) states that no problem is actually *de facto* structured or ill-structured. To solve a problem however, it is necessary to structurize that problem [Bot87]. Bots also noticed that both Soll and Young used the terms *model* when they discussed the structuredness of a problem. A generic definition of model is provided by Shannon, namely a model is "*a representation of an object, system, or idea in some form other than that of the entity itself*" [Sha75]. Pidd defined a model in the context of operations research and management sciences as "*an external and explicit representation of part of reality as seen by the people who wish to use that model to understand, to change, to manage and to control that part of reality*". In other words, a model can be used as a representation of reality (like for instance an object, idea or an organization), to support someone who wants to understand that part of reality, and possibly wants to make decisions which will influence reality.

This brings us closer to understand what a business process model is. However, we still did not define what a business process actually is. In literature there is however no clear and agreed definition available for what a business process is. Hammer and Champy (1993) define a business process as "*a set of activities that, taken together, produces a result of value to a customer*", whereas Davenport and Short (1990) define a business process as "*a set of logically related tasks performed to achieve a defined business outcome*[HC93][DS90]". In their research, Hlupic and Robinson (1998) conclude

generic definition of
model

definition of model
related to OR / MS

that although there is no agreed definition, some common elements appear to be in a majority of definitions [HR98]. These elements relate to the process itself, and to the input and output of the process. The definition provided by Harrington (1991) elaborated on the definition by Davenport and Short, and incorporates the three elements as described by Hlupic. He defines a business process as *“a group of logically related tasks that use the resources of the organization to provide defined results in support of the organization’s objective”* [Har91]. In the remainder of this thesis, we will use this definition for business process. When we join the definitions of a model and of a business process, we conclude that a business process model is *a representation of business processes within an organization and which can be used to understand, to change, to manage and to control these processes*.

definition of business process

definition of business process model

To elaborate on what a business process model actually represents at a business organization, we look at the different levels of an organization, which are shown in Figure 3.2. Figure 3.2 shows five levels of business processes. At the top level, the business strategy is stated, like for instance product quality leadership for products in a certain domain. The business strategy is broken down into goals which are placed at the second level. These goals contribute to the realization of the business strategy. At the third level, high-level organizational processes are defined, which realize the goals stated at the second level. If for instance a goal is reducing the costs for supplied materials, then the process which manages the incoming materials is informally specified at this level. The activities and relationships of the business processes are specified at the fourth level. And at the fifth level the business processes are implemented, which may be done through written procedures or process enactment platforms. Business process models are developed for the processes at the fourth level, namely of the operational business processes [Wes07].

The process models developed through business process modeling provide a modeler with several possibilities to support decisions and process improvements. Melão (2001) grouped the possibilities that process models may provide into four categories, namely the possibility [Ma01]:

1. to capture and record knowledge in order to promote understanding and communication about the business processes;
2. to improve operating performance by investigating which performance measure like time, costs, quality can be improved;
3. to assess alternative scenarios to investigate the impact of possible process changes; and
4. to support the implementation of information systems, namely by supporting the design, implementation, management and control of a business information system (like for instance an ERP-system).

The development of business process models offer the possibility to give insight in complex business processes and interaction between the processes within an organization. These models also provide the possibility to show what effects organizational changes may cause on the structure and performance indicators of an organization, or to support decisions during process execution related to the operating performance. Instead of

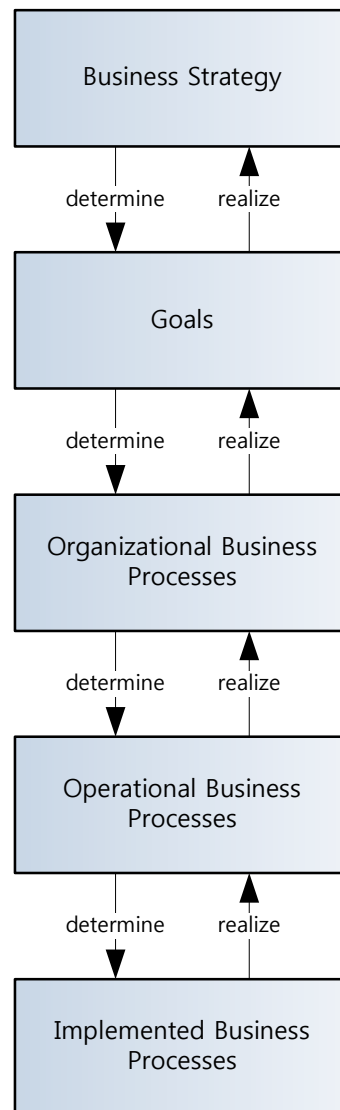


Figure 3.2: Levels of business processes [Wes07]

directly implementing changes in an organization which may be costly and risky (when for instance the processes are not well designed and integrated into the organization), a cheaper and more safe environment is created than the real-world to allow experimentation with possible designs and scenarios [Ma01].

3.2 Business process modeling in practice

In the previous section we discussed the main concept behind business process modeling and why an organizational processes may be modeled. Based on categorization as provided in [Ma01] (with regard to the possibilities the business processes modeling

(BPM) provides), we continue in this section with discussing how this is put into practice. In section Section 3.2.1 we provide an overview of various notations that are widely used to represent business processes in models. As points 2 and 3 in the list relate to the purpose of modeling business processes, namely improving operating performance by possibly redesigning business processes, we will discuss in section Section 3.2.2 several methods that are currently applied for realizing this. A distinction that we make is between heuristic methods to analyze business processes and change these, for instance based on best practices, and business process simulation to involve the uncertainties and time dependences of business processes. The fourth point, related to the implementation of information systems within organizations, is put outside the scope for this research.

3.2.1 Modeling of business processes

In order to be able to represent business processes in a consistent manner, we first look at the lowest level of a business process, namely that of entities on the instance level. An instance of an entity is in this context considered to be a distinct part of a real business process, like for instance activities that take place in a business process, resources, humans, work items, . . . etc. However, due to the complexity of most business processes, it is practically impossible to represent every entity instance as a model. Therefore, it is useful to make some abstraction of the entity instances. Similar instances of entities can be grouped together into a model, which we call a *modeling element* if it has an associated graphical notation. This can be done for all similar entities. All these models grouped together will form for instance a business process model, being a representation of the real business processes [Wes07].

definition modeling element

These models can itself also be expressed in a model, called the *metamodel*. A metamodel describes all the models that can be used to represent all distinctive entities of a business process (or in general any other process, situation or thing) as well as the relations between these models. For the purpose of enabling communication about these models between stakeholders in the real world, a graphical notation may be associated to each model. Weske (2007) formulated the relation between metamodel and notation as follows: “A notation associated with a metamodel allows expressing the concepts of that particular metamodel. Each model is described by a metamodel, and is expressed in a notation associated with the metamodel” [Wes07]. However, it is important to mention that it is not necessary that there is a one-to-one correspondence between a model and a notation. A model can be represented graphically through various notations, while still representing the same model. Finally, the metamodel itself is again an instance of a higher-level model, namely a meta-metamodel. A meta-metamodel allows describing meta-models. Figure 3.3 shows an overview of all levels of abstraction of models, from entity instances, to a model, to finally the meta-metamodel describing the metamodel [Wes07].

In general there are several basic elements which allow a modeler to define a business process [FMS09]. These are the *Start* (an initial node to depict the start of a process), *End* (an end node to depict the final node of a process), *Task* (an activity or process step), *Decision* (to model choices in a process flow), *Split* (to model parallel branches in a process), *Merge* (to consolidate different flow paths), *Join* (to synchronize parallel

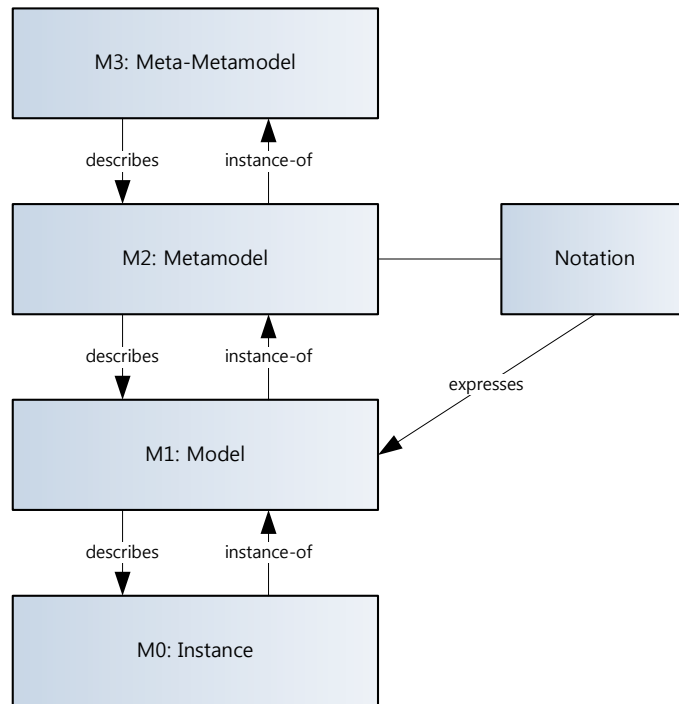


Figure 3.3: Levels of abstraction [Wes07]

sub-processes) and *Event* (to define the occurrence of a certain type of ‘incident’ in a process). As some of these generic elements may make clear, the purpose of business process models is not only to represent the different entities within an organization and the relationships, but also to specify the process orchestration within business processes [Wes07]. The process orchestration depicts how the processes are actually executed within an organization, in other words the internal behavior of a business process. This provides means to analyze the operations within an organization and allowing one to make adjustments and re-design the process.

To describe the control and flow of work in a business process, a series of control flow patterns were identified and defined using these generic (and sometime modeling language specific) modeling elements [RHVM06]. These patterns allow to describe the behavior (i.e., the process orchestrations) of most business processes. The most apparent patterns within business processes are:

- the Sequence pattern;
- the Parallel Split pattern;
- the Synchronization pattern;
- the Exclusive Choice pattern; and
- the Simple Merge pattern

White (2004) provides a clear description of each of these patterns [Whi04]: “The *Sequence* pattern is defined as being an ordered series of activities, with one activity

starting after a previous activity has completed. The *Parallel Split* pattern is defined as being a mechanism that will allow activities to be performed concurrently, rather than serially. A single path through the process is split into two or more paths so that two or more activities will start at the same time. The *Synchronization* pattern combines the paths that were generated by a Parallel Split pattern. The final set of activities within the flows must be completed before the process can continue. This is the “synchronization” of the parallel paths. The *Exclusive Choice* pattern is defined as being a location in a process where the flow is split into two or more exclusive alternative paths. The pattern is exclusive in that only one of the alternative paths may be chosen for the Process to continue. The *Simple Merge* pattern is defined as being a location in a process where a set of alternative paths is joined into a single path.” For more information on these and many other defined patterns, the reader is suggested to see [RHVM06].

The flow patterns are independent of the modeling notation which is used. There exist several modeling notations or modeling languages to model business processes, like for instance BPMN, Flow Chart, Gantt Chart, IDEF0, IDEF3, Petri Nets, RAD and UML [AS04]. Each of these languages has different characteristics (semantics, representation notation, . . . etc) and there are various ways to evaluate these modeling notations. For instance, [WDDR06] compares the UML and BPMN notation (among others) based on their meta-models, graphical notation, serial representation (file format in which a model is saved), extensibility (ability to customize or extend a model) and tool support. [FMS09] compare various notations based on the usability of the models. Usability is defined as an expression of how easy to learn something is, how efficient to use, how easy to remember, number of errors made by users and how subjectively pleasing something is [Nie93]. Based on the framework provided by [GW04] Figl *et al.* state that in general there are two main aspects regarding modeling that can be distinguished, which are [FMS09]:

Creating models The activity of developing a model. To evaluate the usability of a modeling language, the time needed to develop a model is considered, as well as the ease-of-learning and the ease-of-use.

Understanding and interpreting models The cognitive activity involved to be able to obtain and interpret the information that is represented through the model.

The relation between these two aspects is shown in Figure 3.4. It is important to notice that the modeling notation used to depict a certain situation, is the same for both the activity of drawing the model, as well as the interpretation of a model. A modeling notation can be easy to use by a designer to describe a situation in a model, but the resulting model may be not understandable to a viewer (stakeholder). It is thus important that both aspects are considered when evaluation current and a possible new modeling notation.

In the remainder of this section, we will discuss briefly the Unified Modeling Language (UML) activity diagrams, Yet Another Workflow Language (YAWL) and Business Process Modeling Notation (BPMN). The selection of these languages was made due to their wide application for business process modeling by the industry. For a more elaborate comparison on all of the before mentioned languages, see [AS04]. For each of the selected

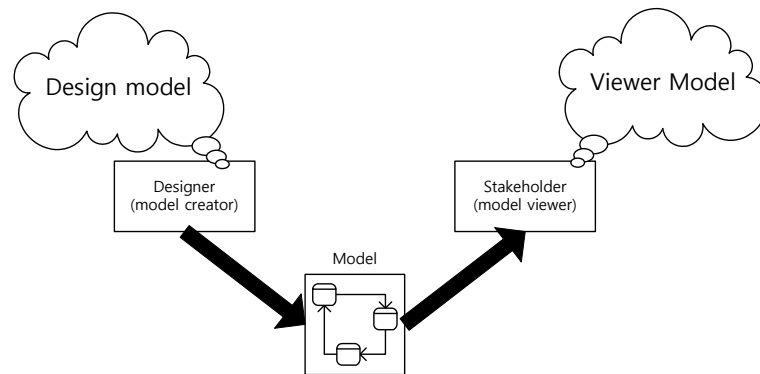


Figure 3.4: Modeling and understanding [GW04]

languages, we discuss the background, demonstrate how each of the control flow patterns is visually modeled using the modeling elements provided by the language, and discuss briefly their usability based on the research by [FMS09]. Figl *et al.* evaluated BPMN, UML and YAWL based on five principles of cognitive effectiveness, namely the representational clarity (the fit between the graphical symbols used in a modeling notation and the concept they refer to), perceptual discriminability (the ease for a user to distinguish the different modeling elements), perceptual immediacy (the ease or difficulty for a user to understand the meaning of a symbol and representation), visual expressiveness (to what extent a modeling notation exploits visual variables like for instance shape, size and colors) and graphic parsimony (complexity of a notation and graphics may impair understanding of a model) [FMS09].

UML activity diagrams The Unified Modeling Language (UML) is a standardized general-purpose modeling language created by the Object Management Group (OMG) [Obj11]. UML is widely adopted and provides specifications to visualize various kinds of models, like for instance application structures, behavior, architecture and business processes through activity diagrams [WDDR06]. Activity diagrams are able to represent the dynamics of a business process to some extent, by depicting the sequence of activities of a business process. An activity is the main element and represents a process that consists of actions and different type of control nodes [FMS09]. It is also possible to represent particular stakeholder who undertake specific activities in an activity diagram, namely by drawing a *swimlane* (in the form of a thick black line between the activities).

In Figure 3.5 an overview is given how the basic control flow patterns are expressed in an UML activity diagram. Figl *et al.* concluded that although the UML notation violates some of the criteria as mentioned above (like for instance the representational clarity of some modeling elements), the overall notation is both quite parsimonious as well as quite expressive [FMS09].

YAWL Yet Another Workflow Language (YAWL) is a workflow language based on Petri nets. Petri nets are a modeling technique for the description of systems, based on the changes of the local environment of a system. The development of YAWL was motivated

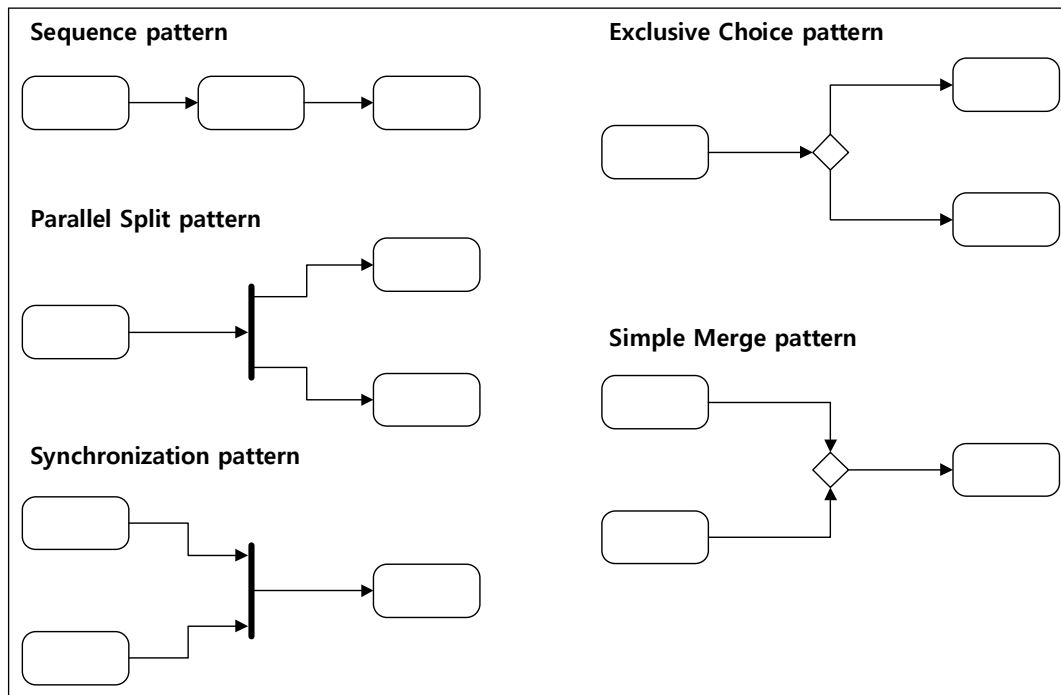


Figure 3.5: Representing basic control flow patterns in UML

by the insights gained from the identification of the various workflow patterns and the inability of Petri nets to express some of these. YAWL is however not merely an extension of Petri nets, but a completely new language with independent semantics [VADH03]. In Figure 3.6 the basic patterns are expressed using YAWL. As can be seen in these models, due to the little variations in shape and size of several modeling elements, YAWL offers little visual expressiveness.

BPMN The Business Process Modeling Notation (BPMN) is developed by the Business Process Management Initiative, which merged with the Object Management Group (OMG) in 2005 [Obj08]. BPMN was developed to support the representation of complex executable business processes as well as to support more general or conceptual business modeling activities [MR08]. To make BPMN models understandable by business analysts, consultants, as well as technical specialists, an attempt was made to develop a set of recognizable and understandable graphical elements.

In Figure 3.7 the basic patterns are expressed using the BPMN language. The perceptual discrimination of some symbols (like for instance used for the parallel split and exclusive choice pattern) is not high. Also, the simple merge pattern can be expressed in various ways which may lead to increased unclarity to a user (both a modeler and a viewer).

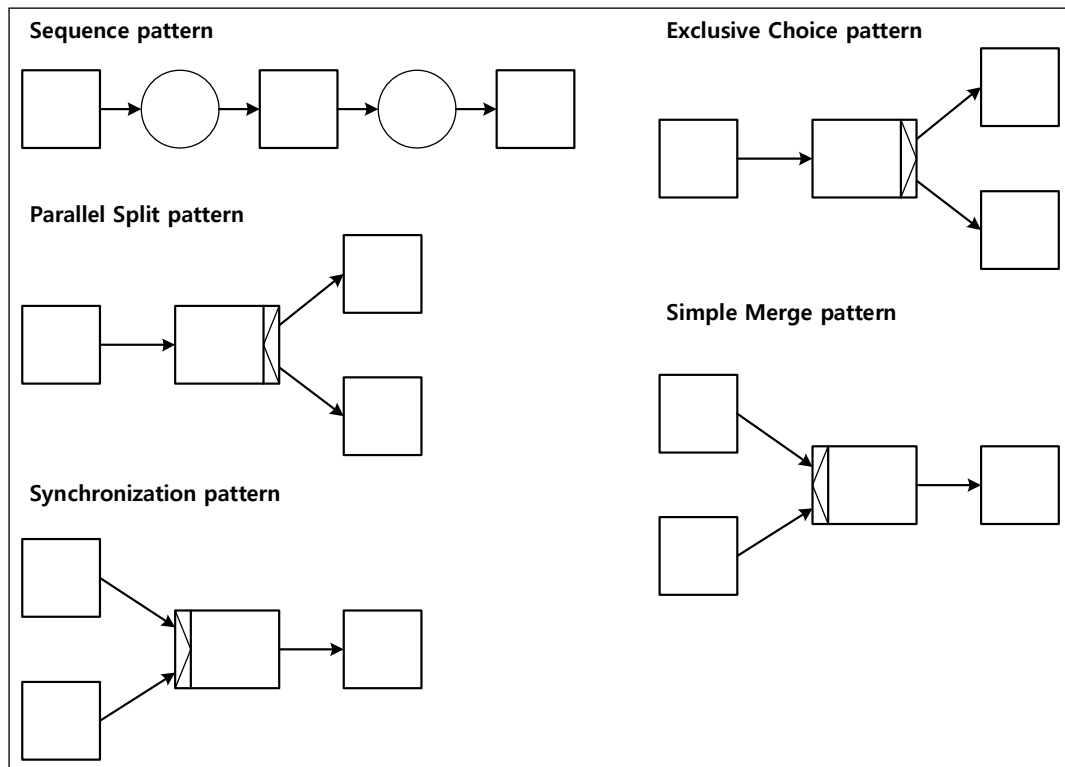


Figure 3.6: Representing basic control flow patterns in YAWL

3.2.2 Business process analysis and simulation

In the first section of this chapter we discussed the purpose of business processes modeling (BPM) (see Section 3.1). We concluded that BPM is used to create insight and understanding of business processes, as well as it provides means for analyzing business processes in order to (re-)design the processes or to support decisions during execution of the processes. This section will elaborate on the use of business process simulation by clarifying the role that simulation can have for the redesign of business processes or during the execution of the processes.

Four main dimensions can be distinguished on which business processes redesign may have an effect, namely *time*, *cost*, *quality* and *flexibility* [RL04]. The main goal of many business process redesign projects are to decrease the throughput time of an order, to decrease the costs required to execute a business process, to improve the quality of a delivered service, to improve the ability of a business process to react on certain variations, or a combination of these goals. However, implementing a change which focuses on improving one dimension, often has a weakening effect on another. Decreasing the throughput time of handling an order for instance, may deteriorate the service level (i.e., quality) of an organization.

The use of best practices is a widely applied approach to redesign business processes. A best practice (or heuristic) is considered to be a way to modify a business process that has proven to be successful (i.e., leading to an improvement of one or more of

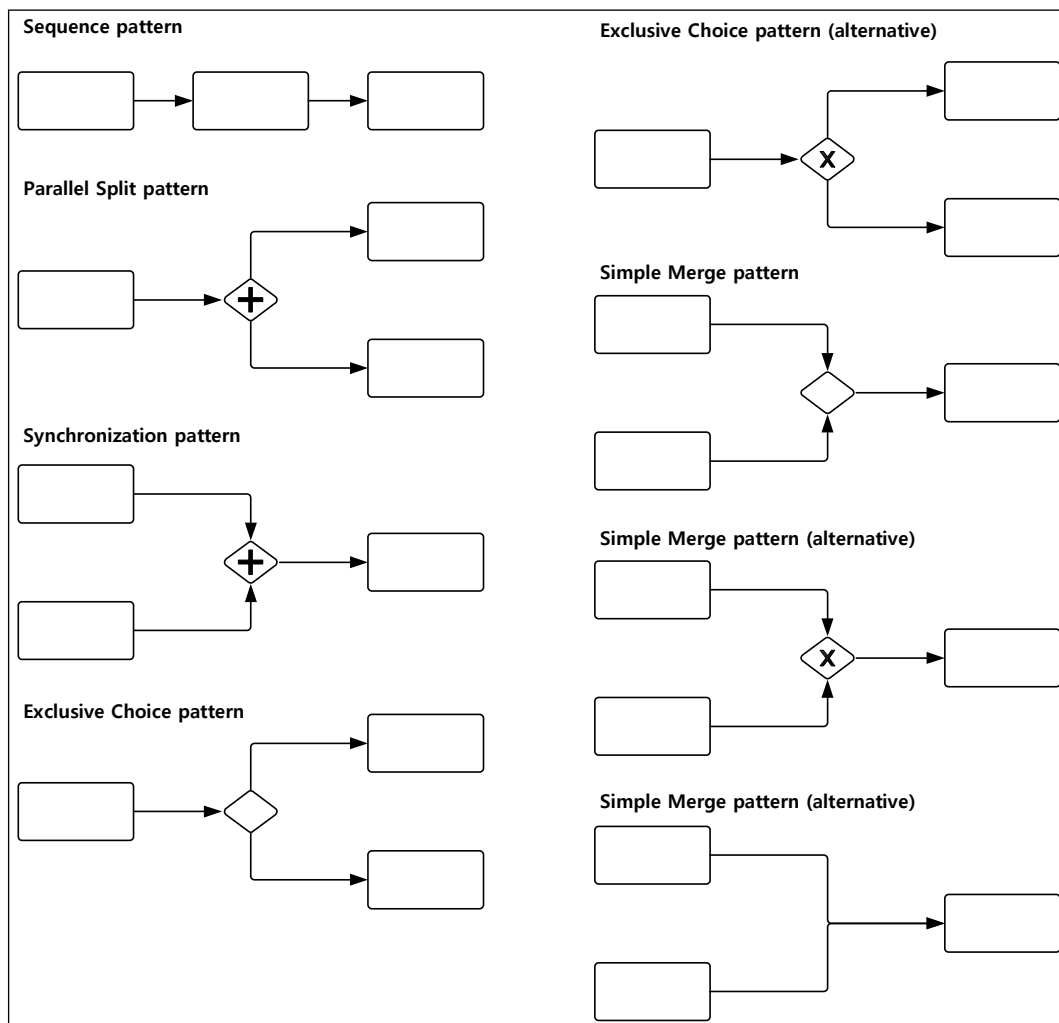


Figure 3.7: Representing basic control flow patterns in BPMN

the identified dimensions) in an earlier redesign project. [RL04] present a framework to evaluate best practices and apply this on in total 29 best practices. These 29 best practices are adopted from various academic literature sources. We will not mention all of these best practices here, but merely a selection to create a general understanding of these concepts.

- **Task elimination: eliminate unnecessary tasks from a business process.** Task elimination is often applied to increase the speed of business processes as well as to reduce costs to handle an order. When a task adds no value (from a customer point of view) it is commonly considered to be unnecessary. However, some tasks in business processes add no direct value to a product or service, but have for instance a quality control function. Eliminating such tasks may lead to decreased quality.
- **Resequencing: move tasks to more appropriate places.** Resequencing is also known as process order optimization and is based on the idea that the ordering of

task does not necessarily reveal the dependencies between tasks. Reordering tasks may lead to decreased costs, because of possible insight that a task is superfluous. Another effect that may arise is that due to resequencing tasks to the proximity of similar tasks, setup times of tasks (the time needed for a resource to start a task) may diminish, thus leading to an overall shorter process time.

- **Extra resources: if capacity is not sufficient, consider increasing the number of resources.** In case the number of resource seem not sufficient to handle all orders, an option it may be to increase the number of resources responsible for doing that task. By increasing the capacity the time that an order needs to wait may be decreased, thus leading again to an overall shorter process time. However, in case additional resources need to be bought or hired, has an effect on the total costs.

Although redesigning business processes based on best practices may seem quite straight forward, the importance is that when implemented a redesign effort gains the results as was originally planned for. However, as was mentioned by explaining the drawback that focusing on one dimension may lead to a decrease of another dimension (e.g., decreased time may lead to decreased quality), it may be difficult to foresee all effects that a business process change may cause. Research performed by [HC93] makes clear that redesign efforts are often less successful than originally planned. Some frequently mentioned problems related to business process design are the inability to accurately predict the outcome of redesign efforts; the difficulty to capture existing processes in a structured way; the level of costs incurred by implementing a new process; or the inability to recognize the dynamic nature of business processes [HD05]. Errors in business processes designs are only recognized once these redesigned processes are implemented, and often when it is too late, costly and difficult to correct wrong decisions [Tum95]. Business process simulation (BPS) provides the possibility to model business processes, analyze the processes, and evaluate the effects that changes in the business process may cause [HD05]. This also limits the risk that redesigned processes finally fail to meet the expectations.

definition computer
simulation

In [Ma01] computer simulation is defined as “*a set of iterative activities to abstract, build and experiment with a computer-based model which mimics the dynamic behavior of a system*”. As this definition makes clear, the process of analyzing business processes using simulation involves performing several activities in an iterative manner. Simulation allows a business analysts to conduct ‘what if’ analysis, it shows dynamic changes of business processes and allows to evaluate the effects of stochastic events and random behavior of resources [HD05]. In Figure 3.8, a simplified simulation project life cycle is shown which depicts the main activities and products of a simulation study.

An abstraction of a real-world situation is made by *conceptual modeling* this situation, which results in a conceptual model. *Simulation model construction* relates to the development of a computer-based simulation model based on the conceptual model which will be executed by some simulation software. *Experimentation* relates to the execution of the simulation model in order to achieve an understanding of or to provide a solution for the real-world situation. What is important to mention, is that the life cycle of a simulation study consists of various *iterative* steps, which is depicted by the bidirectional arrows in

Figure 3.8. Several authors have discussed similar, but slightly different simulation study outlines (see in Appendix C Figure C.1 and Figure C.2). One of the differences between the two outlines is that Banks considers the model conceptualization and data collection to be performed in parallel [Ban98], whereas Shannon considers it to be sequential [Sha75]. However, regarding the scope and purpose of our research, we consider the simplified simulation life cycle in Figure 3.8 to be adequate.

The following subsections will discuss briefly Figure 3.8 and its contents. For more information on simulation, the reader is referred to texts such as [Ban98], [LK00] and [Pid04a].

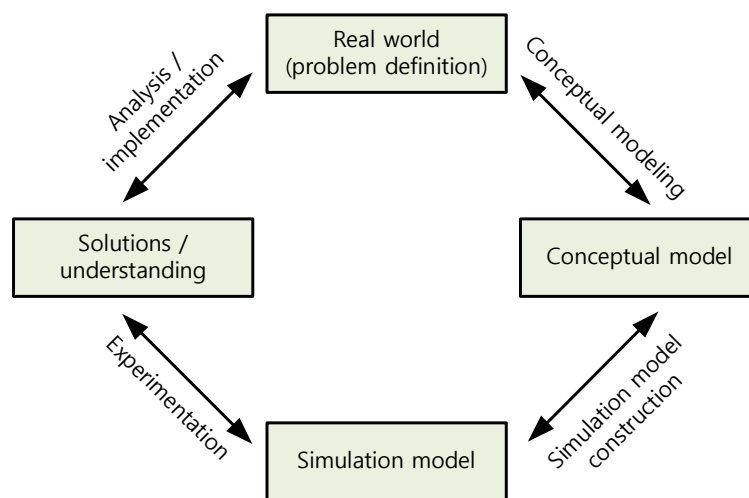


Figure 3.8: Simulation study life cycle

Conceptual model development There is no general agreed definition to be found in literature what a conceptual model is. Based on various literature sources, Robinson (2011) defined a conceptual model as *“a non-software-specific description of the computer simulation model (that will be, is or has been developed), describing the objectives, inputs, outputs, content, assumptions, and simplifications of the model”* [RBKZ11]. A conceptual model is developed to build credibility for and guides the development of a simulation model. The conceptualization phase is considered to be an important part of a simulation study. In Section 3.1 we defined of business process model as *a representation of business processes within an organization and which can be used to understand, to change, to manage and to control these processes*. Because computer simulation relates not necessarily to simulation of business processes but also for instance to simulation of production processes, the more generic term conceptual model is used in computer simulation literature. We consider a business process model (as in ‘a model of business processes’) to be a specification of a conceptual model.

definition of
conceptual model

Simulation model construction After a conceptual model is developed, a simulation model of the business processes can then be developed based on the conceptual model. As

mentioned before, undertaking a simulation study is an iterative process and a conceptual model may be revised several times during a study.

A distinction can be made in a simulation study whether the behavior of a system (e.g., an organization and its business processes) is modeled and simulated as a discrete or continuous system. This depends on how changes in a system occur. Pidd (2004) clarifies this by stating four possibilities when changes may occur, namely: 1) continuously at any point of time; 2) continuously but only at discrete time points; 3) discretely at any point of time; and 4) discretely only at discrete points of time [Pid04a]. Discrete event simulation is considered to be a suitable way to model business processes, because of the behavior of the business process entities (e.g., resources are busy or idle, or entities like for instance orders enter or leave an organization system) and the discrete events (e.g., begin task and end task) [Ma01].

A discrete event simulation model is executed by simulation software which generally consists of a simulation engine (or simulation executive) and an application program [Pid04b]. The engine keeps track of the state changes which occur at some moment in time and reminds the application when a state change is due. This interaction between engine and application are defined by a set of rules, which define the dynamic state transitions that occur over time. In general there are three definitive sets of these rules, known as “world-views”: 1) Event Scheduling world-view; 2) Process Interaction world-view; and 3) Activity Scanning world-view [Van02]. In the Event Scheduling world-view a system is viewed as “a series of instantaneous events that change the state of a system over time” [Peg10]. A simulator engine advances the simulator clock until the moment that the earliest event occurs and only execute this event (except if other events are scheduled at this moment, then these are also executed). In the Process Interaction world-view, a time-ordered sequence of events is used to describe the flow of an entity through a system. Finally, the Activity Scanning world-view focuses on the activities performed in a system, as well as the conditions that control the begin and end of such activities [Pra10].

Various specification formalisms exist to support the formal representation of a simulation model, like for instance Discrete Event System Specification (DEVS) [ZPK00] and Petri Nets [Pet81]. A formalization of a simulation model provides a standardized way of how to describe states and time of a model and its entities. To get an executable model, the conceptual model should be translated into an executable simulation model based on a formalism. This can be done manually (i.e., programmed) by the modeler, or constructed through a visual interface [Pid04b]. Tools that provide a visual interface often rely on a library of formally specified general-purpose or domain-specific components which the modeler can drag-and-drop on a virtual worksheet. By connecting these components the relations and sequence order are specified. To allow a simulation engine to translate this into an executable model, the components need to be specified with data. Data is collected from the business situation (e.g., arrival rates, task durations, number of resources, . . . etc).

Two aspects which are of importance during both the conceptual modeling and simulation model construction phase, are validation and verification of the developed models (see Figure C.1). The purpose of validating a conceptual and simulation model is to validate that these models represent the modeled situation adequately for the purpose

of the model [Sar04]. The output that a simulation model produces should be consistent with expected results based on information describing the real world situation. Verification of a simulation model relates to whether the computer programming and implementation of the conceptual model is done correctly.

Experimentation and analysis After a simulation model is developed, validated and verified, analyses can be performed. The modeler may set-up one or more experiments which contain information about how the simulation engine should be executed (e.g., duration of a simulation run, number of replications, . . . etc) and model parameters depicting different scenarios. Execution of the simulation engine results in output data related to the parameters and performance measures as were defined by the modeler. It depends on the business case which parameters are of interest for the modeler. Several generic statistics are defined which are generally of interest to a modeler, e.g., total process cycle times, average waiting times, resource utilization [GT94]. There are different ways to present the output of a simulation execution, like for instance in raw data format, in histograms or in data plots. Animation of the simulation is useful for debugging, understanding and communication purposes [LK00]. Depending on the output interpretation by the modeler, the modeler can decide to perform more experiments or make changes in the simulation model to evaluate the performance of different process redesign scenarios.

3.3 Limitations of current approaches and tools

The description on business process analysis and simulation in Section 3.2.2 is merely a brief overview of how a generic simulation study is described in literature and does not cover (by far) all aspects related to it. However, it provides a generic background on undertaking a simulation study and a starting point to discuss some of the limitations that are mentioned in literature about Business Process Simulation (BPS) and BPS projects. These limitations clarify why BPS is currently not widely adopted in business analysts practices.

In this section, we will highlight the most common limitations mentioned in literature regarding business process modeling and simulation. This overview will provide a starting point to later define the problem space. Figure 3.9 depicts how the limitations are categorized in the remainder of this section. This figure is not an all-embracing representation of a simulation study. However, it provides the means to assess to what aspects the limitations relate. The input variables of a modeling and simulation study is the current organizational situation, including the business processes and the data describing the business processes (1), and the objectives of the modeling and simulation study (2). Supported by the modeler and domain experts describing the business processes (3) and business process modeling and simulation tools (4), business process models and simulation models are developed and executed (5) which will finally produce some output (6).

1. Issues related to the input of a study: Business processes and data Melão & Pidd (2003) held a survey under practitioners of business process modeling and simulation

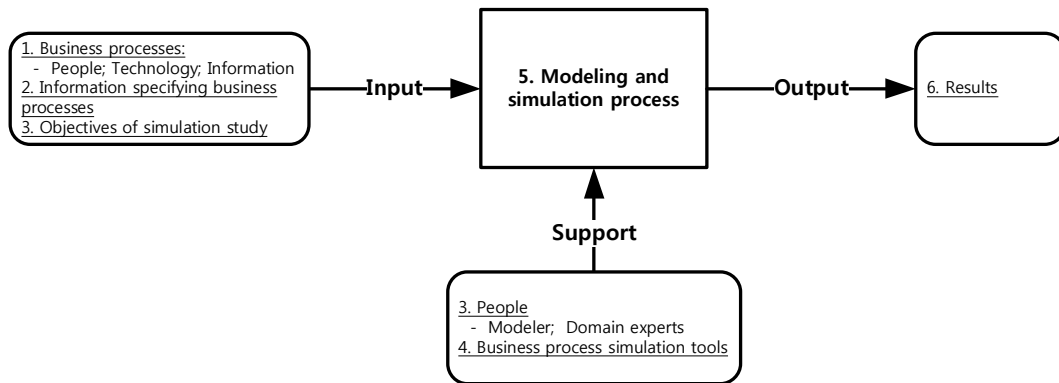


Figure 3.9: Categorization of limitations of business process simulation studies

which resulted in several interesting statements regarding limitations of current tools, and why BPM and BPS is possible not widely adopted [MaP03]. A reason which was given was that stakeholders feel that BPS is likely to be unproductive because business processes are too ill-defined. In other words, BPS is not suited to all applications or to all circumstances. These stakeholders often rely on simple static process mapping methods and possibly use spreadsheets to extend the details. A quote from one respondent was: "... some of the business processes are so loosely defined that BPM is a pointless undertaking".

A possible explanation was provided by [MaP03] that some BPS tools are not flexible enough or it may be too difficult to model the ill-defined nature of many business processes. This also because the human behavior may be regarded as so unpredictable and tacit that it is very difficult to model. Or possibly because current BPS tools may be too simple to adequately represent the logic resulting from complex human interactions. Another finding why it is difficult to model business processes, is because 'reality' may change because of subjective construction of peoples' mind [Ma01].

A possible limitation of current BPS tools is that these tools may not provide the possibility to model and analyze interrelations among different business process dimensions, such as people, organization and technology [Ma01]. These dimensions are interdependent: changes occurring in one dimension may have an effect on the other dimensions. BPS tools should support this multi-faceted nature of business processes, but this may raise more issues regarding the integration and consistency between different modeling requirements and paradigms.

Besides the limitations related to representing process participants in a simulation model, another problem is identified which relates to data acquiring describing the business processes. This data may simply not exist or is missing, or it may be too expensive to gather all data, or the system that is being modeled does not exist at all [Ban98]. It also happens that although domain experts mention that they have all required data, they actually only have abstractions or averages of the data. This is often useless for business processes simulation. Data collection is by some experts considered to be the most challenging and time consuming phase of a simulation project, and may cause the study objectives to be revised due to unavailability of data [Ma01].

2. Issues related to the input of a study: Objectives of modeling and simulation study Problems within organizations often have a high “sense of urgency” which results in that decisions need to be made quickly [DVvE03]. However, BPS is a time consuming undertaking and this may result in that business process decisions can not always be supported on time. It may also occur that a problem already changes or diminishes before a simulation study is finished [VV02].

3. Issues related to the support of the process: People (modeler and domain expert) The people that are involved in business processes are often also the ones describing the activities that they and others undertake, which leads to several issues. If different people have the same role in a business process, they may describe the process differently, due to different values or perspectives, or how they find that a process is or should be carried out. It is also possible that some participants have hidden agendas [Ma01].

To understand a business process adequately, in order to develop a good representation of it, many participants and domain experts should be interviewed. This results sometimes in conflicting information and the modeler should confront the participants with this to finally obtain a good and agreed representation of the processes [Ma01]. This whole process is time consuming, as well as costly: interviewing a large group of domain experts requires more resources to conduct the interviews [Ban98]. Another issue that is mentioned in literature is that simulation experts and domain experts speak a different “language”, which makes it hard to effectively develop an adequate understanding of the business processes [VV02].

4. Issues related to the support of the process: Modeling and simulation tools Besides that modeling the complexity of most organizations is time consuming, Hlupic & De Vreede (2005) mention that it is also expensive [HD05]. Professional BPS tools are often expensive and simulation-experts with the experience to develop good models are also well-paid. Another issue which is mentioned by Melão (2001) is that BPS models tend not to provide the multiple levels of abstraction for different involved stakeholders, as well as integration and consistency between the different levels [Ma01]. Senior management for instance, is more interested in a simplified high-level view of business processes, whereas operational staff is more interested in the details of how activities are or should be carried out.

De Vreede & Verbraeck (2003) provide an explanation of why simulation model development is time consuming, namely because direct mapping of conceptual model elements and simulation code is rarely possible [DVvE03]. As Harrel (1996) also pointed out, a modeler must often start from scratch when developing a simulation model based on a process map [HF96]. There are namely large incompatibilities between process mapping tools and simulation tools regarding both purpose and paradigm. Because the structure of a conceptual model is often quite different from the structure of the simulation model, the result is that a model will be hard to recognize by domain experts and hard to verify by the model builders. Cetinkaya (2010) noticed in recent research that there is a large semantic gap between the conceptual modeling stage and the simulation model construction stage [Cet10].

5. Issues related to the modeling and simulation process Based on literature research, Melão (2001) noticed that information about practical use ("how-to") of BPS is mainly absent [Ma01]. Most of the literature about simulation projects describe projects that were successful, however projects that failed or were less successful are rarely made explicit. Determining the appropriate level of detail for a simulation model is also not straight forward. Not enough detail results in less costs because less resources are used, but will likely produce not enough insight in a business organization. And too much detail in a simulation model will require extensive resources in building, running and validating it, and will not provide the guarantee that results will provide useful insights.

Banks (1998) also mentions as a disadvantage of BPS that the process of modeling and simulation requires special training that is learned over time and through experience [Ban98].

One other aspect mentioned in literature relates to the continuity of conceptual and simulation models. The development of a simulation model based on a conceptual model in general is considered difficult and time-intensive. This is partly due to the fact that conceptual modeling techniques are at best semi-formal [CVS10]. This means that based on a certain conceptual modeling techniques, different simulation models can be developed. It depends largely on the simulation model developer and his interpretation of the conceptual model, how he specifies the simulation model. In other words, there is a large semantic gap between conceptual models and simulation models, which may finally lead to inaccuracies in developed simulation models [CVS10].

6. Issues related to the output and results Banks (1998) mentions that results of simulation studies are often difficult to interpret due to their random nature [Ban98]. Simulation outputs are essentially random variables based on random inputs (e.g., arrival rates of e-mails) which makes it hard to determine whether an observation is a result of system interrelationships or randomness.

Support of decisions through the use of BPS is often inadequate, according to many decision makers. Undertaking a simulation study is too time consuming which leads to results not being available on time [VV02]. Also, the quality of results is sometimes considered being too low. Due to this, decision makers tend to doubt the quantitative results from a BPS study, if they understand the models at all [DVvE03].

3.4 Comparison of literature and Accenture's case

In Section 2.3 we introduced a set of limitations and issues within Accenture related to their current business process modeling and simulation approach. Now that we discussed in Section 3.3 various issues that are identified in scientific works about modeling and simulation, we will provide a comparison between both domains in this section. In Section 3.4.1 we will first compare the issues identified in Accenture against the ones identified in this chapter. Based on this comparison, we will define the *problem space* in Section 3.4.2.

3.4.1 Comparison

In this section, we will try to match the issues identified in Accenture (see Section 2.3) with the issues identified in literature (see Section 3.3). The first finding is that of the nine identified issues within Accenture, only four can be matched directly with the ones found in literature. Two issues that cannot be matched directly relate to the composition of Accenture project teams, two relate to assumptions that are made based on project information that was provided at the start of projects and the fifth relates to data acquiring by consultants for the specification of business processes.

Related to project composition:

- "Different process modeling practices within the service lines";
- "Project team members interpret tasks differently due to different backgrounds".

Related to project information at the start of projects:

- "It is unknown on beforehand whether full process descriptions are available";
- "It is unknown on beforehand which data for business process specification is available".

Related to specification of business processes:

- "Client-side specialist often have difficulties estimating figures"

Although the above mentioned issues were not identified directly in literature, we do not want to suggest that these are Accenture specific issues and did not occur in projects outside Accenture. The first four issues however all relate to the management of a project, namely to the initial phase of a project. For now, we leave issues related to project management outside the scope of our problem and solution finding research, but will come back on it in the recommendations chapter. Regarding the fifth point, we did not find elaborate discussions in literature about problems with estimating figures by domain experts or business managers. However, as mentioned under point 1, data collection is by some experts considered to be the most challenging and time consuming phase of a simulation project. Banks (1998) also mentions that if no exact data is available to specify a model, so-called 'guesstimates' should be obtained, but no explicit issues are mentioned about this (with regard to the person who need to provide these guesstimates) [Ban98].

The issues identified in Accenture projects which could be matched with the ones identified in literature, are presented and briefly discussed below.

No common and agreed understanding of business processes at client-side This issue was also mentioned in Section 3.3 under list-item 3: '*Issues related to the support of the process: People*'. Melão (2001) mentioned in his research that descriptions of business processes may be different among various stakeholders [Ma01].

Data specialist at client-side understand processes differently then business managers This issue is not explicitly mentioned, but is according to us similar to the above mentioned issue and also falls under the scope of list-item 3. In literature we did not come across a clear separation of data specialists and business managers at the client-side. However, inconsistencies do arise when business process descriptions are collected and mapped provided by different stakeholders.

Software specialist sometimes need to be educated with simulation and DSOL related knowledge This issue is mentioned in [Ban98] and falls under the scope of list-item 5: '*Issues related to the modeling and simulation process*'. Modeling and simulation requires namely special training that is learned over time and through experience.

Time consuming to translate business process models into simulation models This issue is mentioned in [DVvE03] and falls under the scope of list-item 4: '*Issues related to the support of the process: Modeling and simulation tools*'. Both in [HF96] and in [DVvE03] this issue is elaborately discussed.

Issues falling under the scope of list-item 2 and 6 were not mentioned as being current limitations of Accenture's simulation projects. However, we do not want to suggest that issues regarding objectives of a simulation study, or the output and results of a study, did not occur within Accenture's projects. It is also possible that these issues appeared but were not considered to be of too much importance to be mentioned by the interviewed consultants, or were simply forgotten to be mentioned.

Our main goal of first describing the limitations within Accenture, then the limitations within literature, and then making a comparison between these, is to provide the foundation that if a solution is found for the problem of Accenture, this can be deducted to a much wider problem area. In other words, Accenture's problem with regard to modeling and simulation does not stand on its own, but is similar to problems which our found within scientific literature. By executing our research within Accenture, we hope to be able to solve a problem which occurs also outside Accenture.

*motivation limitations
study and comparison*

3.4.2 Problem space

In [Ven06] the concept of the problem space is explained as being the representation of the researcher's understanding of the problem(s) which will be addressed by a proposed (technological) solution. It is placed in context by relationships with other identified problems and problem aspects. According to Venable (2006) the relationships between the concepts may be aggregation, generalization or some other kind of relationship. Causal links however are mostly used.

Based on the insights acquired from the background literature study, as well as the limitation studies and comparison between both studies, we abstracted the *problem space*, which is shown in Figure 3.10. To understand the meaning of a relationship drawn in Figure 3.10, we explain it as following. Let us consider the relationship between "Accuracy of business processes and data" and "Conceptual model development process". We read this as following: "The *accuracy of business processes and data* (provided by the

client) has a certain influence on the *conceptual model development process*, which can be positive or negative. For instance, if the business process descriptions are inaccurate (related to the real situation), the conceptual model development process will take longer to finally realize an adequate and valid conceptual model".

Generally, this diagram depicts aspects which influence the development of conceptual and simulation models, as well as the quality of these models. This influences ultimately the overall successfulness of a BPS project. The influences depicted within the diagram (e.g., the choice of a conceptual modeling language) on the model development stages and analyses stage, as well as the performance indicators (e.g., the successfulness of a BPS project) and the relations between these, should be considered more as qualitative indications of what can be influenced which leads to an overall improvement of BPS projects.

3.5 Summary and conclusions

In the previous chapter we elaborated on Accenture management consulting activities related to business process modeling and simulation. In this chapter we provided an overview based on literature research with the main question why and how business processes are modeled and what the limitations of current approaches are. First, we defined what a business process and a business process model is and elaborated on the different levels of business processes. We clarified how business processes contribute to the goals and strategy of an organization. In order to improve or maintain a certain level of performance of an organization (because of the changing environment most organizations are part of), we argued that business process modeling provides means to understand and improve business processes, and to assess alternative scenarios.

We then discussed business process models in more detail and elaborated on the concept of metamodels and graphical notations associated to these metamodels. Regarding the development of models, two activities are of much importance, namely *creating* a model and *interpreting* a model. Several control flow patterns were then discussed which appear in most business processes and we provided three examples of how these control flow patterns are expressed using different modeling languages (namely in YAWL, BPMN and UML). To allow different scenarios of business process changes to be evaluated, we noticed in literature that business process simulation (BPS) is considered useful. BPS is a set of iterative activities to abstract, build and experiment with a computer-based model which mimics the dynamic behavior of a system.

Several issues were identified, related to the input of a BPS study (i.e., the real world business processes and data), the support of a BPS study (i.e., the simulation environment and modeler), the BPS process itself, and the usefulness of the output and results. After comparing the insights of the literature research with the limitation found in Accenture projects, we constructed a model depicting the problem space. This model provides us with support to direct our next research steps of possibly finding a solution for the right problem.

To conclude, the purpose of this and the previous chapter is to make clear that there is an evident relation between Accenture's modeling and simulation (M&S) issues and M&S issues found outside Accenture. This allows us to finally deduct the results

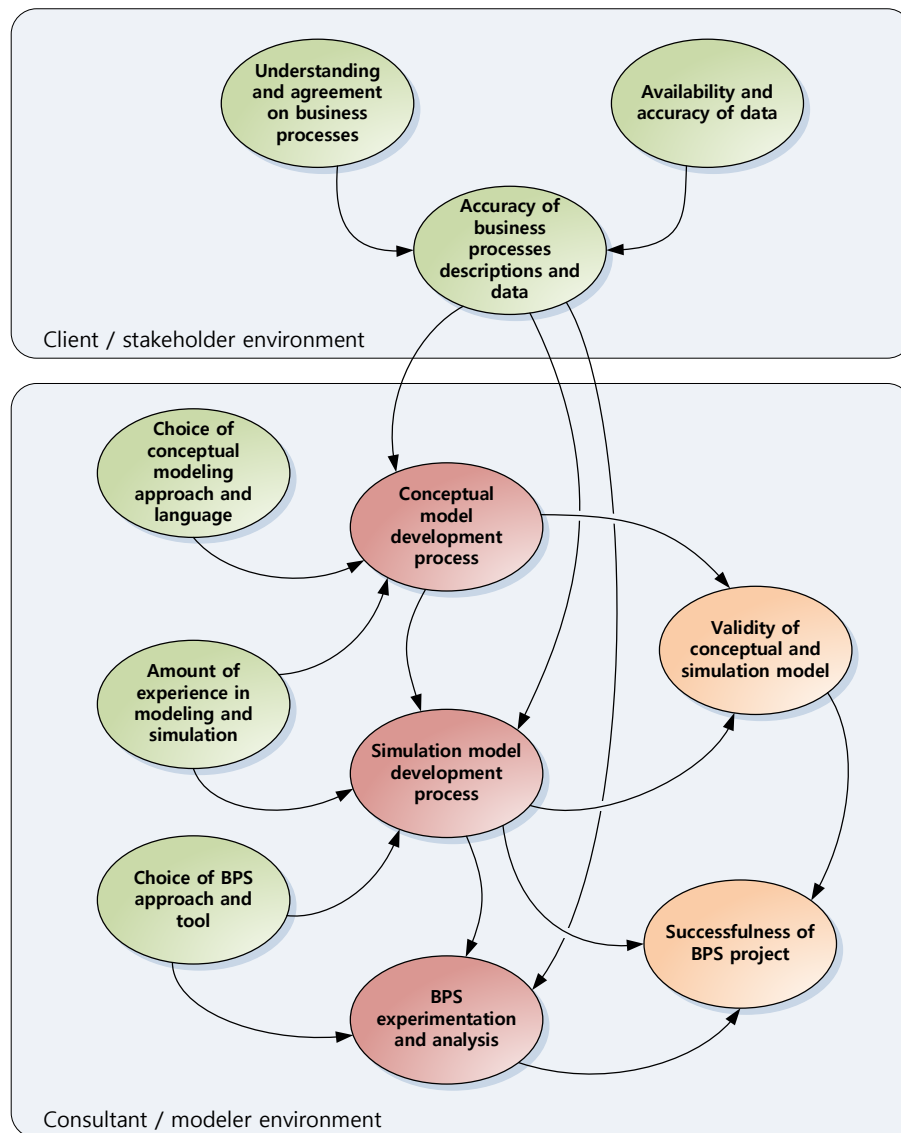


Figure 3.10: Problem space

and solutions from our research (focused on Accenture), to the more general problem regarding the difficulty of simulation model development.

Design Requirements

During the past 2 months, Christina was working on her project and encountered similar issues as during her previous project. She also met several times with the student, to explain the way how and for what reasons business processes are modeled within the company and within her department.

Yesterday, she met again with the student and he asked her what a solution for consultants should be able to do, or should look like. "Well, I would like to have a program, that allows me to develop a business process model in the way how I see business processes," she started with. She continued: "But the way the processes are modeled, should also be understandable by for instance the managers of the client organization. To validate the correctness of models, I often sit down with one or more managers and we look at what we've developed. So, it would be nice if it can be used together with several persons."

Christina mentioned several other aspects, that she saw as essential to be included in the solution. "It would be great if the solution could do the work of the computer programmer. I guess that it should be able to automatically develop the simulation model, based on what I draw. And of course, once such a model is generated, I want to experiment with it. After I press on a 'play'-button, it should produce understandable graphs and numbers, relating to the characteristics and performance of the business process. And it would be very useful, if it would actually visualize the processes and what is actually happening within them."

Again, the student thanked her for her time. He told her that he would make an overview, based on what she said, as well as some other experts who are involved in the project, and send it back to her. Later, they would meet again, to verify whether what he wrote down, was correct.

In Chapter 2 and Chapter 3 we discussed the problem by elaborating on the business process modeling and simulation (M&S) approach as currently performed within Accenture and as described in literature respectively. The next step is to identify requirements that a solution to the problem should meet, which will be done in this chapter.

The process of identifying good requirements is a difficult but important step in a design process, to finally obtain an appropriate solution for the right problem. A requirement is defined by Bahill (2009) as "a statement that identifies a capability or function that is needed by a system in order to satisfy its customer's needs". We will

follow in main line the approach as mentioned in [You04], namely:

1. Identify the stakeholders;
2. understanding the stakeholder's needs and expectations;
3. identify, clarify and restate the requirements;
4. define evaluation criteria;
5. test and verify the requirements;
6. validate the requirements.

Although these steps are numbered and appear to be sequential, they are in fact performed in an iterative and cyclical manner. During the research project the requirements are validated and restated several times to connect as closely as possible with the needs and expectations of the main stakeholder (our customer). This section is the outcome of the complete requirement acquiring process.

Before we can identify capabilities and functions which are needed by a solution, we first need to define in a clear and unambiguous manner who the main stakeholder customer is and what his needs are. In Section 4.1 we will elaborate on list-items 1 and 2, namely the identification of the customer and other stakeholders. In Section 4.2 we will elaborate on list-item 3 and present the design requirements. The design requirements are based both on expert-interviews with management consultants from Accenture, as well as on literature related to modeling and simulation (most of which is discussed in the previous section). Finally, in Section 4.3 we will discuss list-item 4, namely the evaluation criteria for each requirement. List-item 5 and 6, related to testing and validating the requirements, is done during the interviews with the management consultants of Accenture.

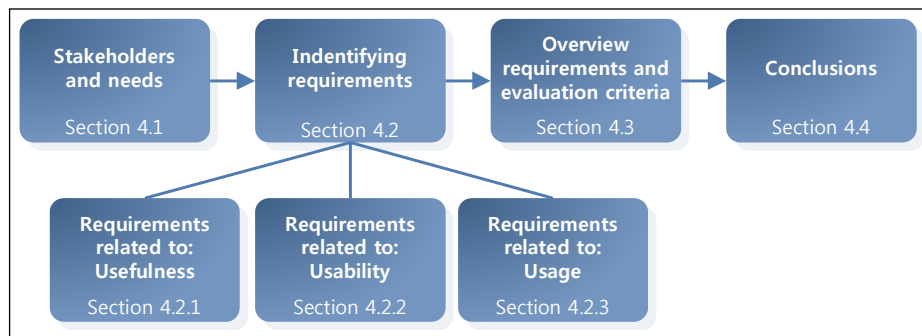


Figure 4.1: Outline Chapter 4

4.1 Stakeholders and needs

We identify two stakeholders in the process of modeling and analyzing business processes, namely the *management consulting firm* and the *client organization*. The client organization faces some problem, like for instance low business performance or inadequate

knowledge how to make certain business decisions, and requests the help from a consulting firm and its consultants to understand the problem and to get advise how a problem may be solved or the situation can be improved. The consultants wants to provide this help, by having the ability to model and analyze the current business processes of a client in a rigorous manner and possibly evaluate redesign options.

The consulting firm is the main customer of this project and its management consultants are the end-user of the final solution. In general, a consultant's role consists of "*analyzing and bridging the gap between their (the consultant's) body of knowledge and skills and the requirements of the client organization*" [LG86] (adopted from [FW96]). More explicitly and related to business process modeling, a consultant is entailed with the task (a project) to model business processes of a client firm and possibly analyze, evaluate and suggest redesign alternatives. With regard to the body of knowledge of a consultant, most consultants have experience with modeling business processes, but only few of them know about (and even less have actual experience with) business process simulation (BPS) (based on information from Accenture).

The consulting firm is the main customer of this project and its management consultants are the end-user of the final solution

Consultants of Accenture have in general a university master degree, but their backgrounds differ from engineering sciences, to social sciences, to almost any kind of study. Depending on their background and qualities, consultants are part of a certain service lines or industry group. The duration of a project for a client organization can vary from less than a month, to several years, depending on the scope of the project. During a project, a consultant is assigned with one or several tasks which he performs in a team of other consultants and with the support of client-side specialist (depending on the project).

The client organization can be *any kind of organization that has requested and receives help from a consulting firm*. This definition of a client is by some academics considered to be too shallow, but we will use this definition for the understandability of this concept (for more information on "the client", see [FW96]). The client is considered to be the *problem owner* and although the business processes are part of its organization, the client may only have a partial understanding of how the business processes actually take place. The client may have some assumptions of what is causing organization problems, but the role of the consultant with its knowledge and experience is to create a meaningful representation of what is actually happening in the organization and be able to identify the real causes of a problem or processes that can be improved. To make it more explicit, the general needs of a client organization are: 1) To get insight in its business processes, its organizational situation and its possible problems; 2) to get support for business decisions; and 3) to get suggestions for business process improvement.

Although the needs of the client are important to understand what help a consultant may provide, the main 'customer' of this research is the consultant and more specifically Accenture's management consultants. In general, the consultant wants to support the client and provide help, thus wants to be able to:

1. ...develop 'as-is' and 'to-be' business processes models of a client's organization easily; and
2. ...to execute these models using some software tool, in order to simulate the business processes and analyze specified performance indicators and redesign scenarios.

We elaborated on these needs with some experts from Accenture (who have simulation experience) and several aspects were mentioned to be included in a solution. A proposed modeling language should for instance be as easily constructible and interpretable as a simple flow diagram, but it should also preferably be based on a formally specified business process modeling language. Regarding the modeling and analyzing activities itself, it should all be as simple as possible, both to draw a business process model, as well as to experiment and analyze it. In essence, it should be possible to go for a first meeting to a client organization, and be able to map some business processes in a model and execute this model directly. Regarding this research, two hard requirements were made, namely: 1) in the time available for this research a working prototype should be developed which incorporates the basic requirements (as will be discussed in the following section); and 2) the modeling language is mostly based on the business process modeling experience of experts within Accenture.

... it should be possible to go for a first meeting to a client organization, and be able to map some business processes in a model and execute this model directly

4.2 Identifying requirements

Based on the needs identified in the previous section as well as the discussion in the preceding chapters, we conclude that the main purpose of a solution is to give insight into the client's business processes and support decisions related to these business processes. Keen and Sol state that there are three factors that add value to the effectiveness of a decision support system (DSS), namely Usefulness, Usability and Usage (also known as the three U's) [KS08].

Three factors that add value to the effectiveness of a DSS: Usefulness, Usability and Usage

- **Usefulness** relates to the analytical methods and information resources of a DSS and is commonly acquired through domain specific libraries and examples [Jacobs2005].
- **Usability** depends largely on the interface between the user and the tool. In other words, the way users think and work and the ease of interaction and collaboration with the tool.
- **Usage** expresses flexibility, adaptivity and suitability and the main question concerned is *how the system is embedded in a decision process*.

To guarantee that a solution in the form of a software tool meets the characteristics of an effective DSS, we will use these three U's to categorize the requirements in the following subsections.

4.2.1 Requirements related to: Usefulness

The main requirement for a solution to support the analysis of business processes by consultants, is that it can simulate business processes over time. More specifically, it

should be able to simulate the dynamic behavior of events and interactions within business processes.

Requirement 4.1. *The tool should be able to simulate the dynamic behavior of events and interactions within business processes.*

In order for a tool to be able to simulate business processes (or other discrete events), the tool should provide certain features. As formulated in [Jac05], a simulation tool is required to provide: pseudo-random number generation, statistical distribution functions, time-flow mechanisms and statistical analysis routines. In [LK00] these features are discussed more elaborately. The duration of activities within processes can be and are often abstracted and described using statistical distribution functions. Pseudo-random number generation is required to draw random values from these specified distribution functions. A time-flow mechanism or timing executive is needed to mimic the course of time. The statistical analysis routines provide a descriptive summary of the simulation model behavior [Nan95]. It is beyond the scope of this research to develop a new simulation environment and thus these features are not explicitly mentioned as design requirements. However, they should be included in the final solution.

4.2.2 Requirements related to: Usability

Usability is defined in Section 3.2.1 as an expression of how easy to learn something is, how efficient to use, how easy to remember, number of errors made by a user and how subjectively pleasing something is. The usability of a solution relates to how the user (the consultant) will interact with a tool and to what extent the consultant considers the tool usable.

In Figure 3.4 we depicted two important activities related to business process modeling, which are *creating* a model and *understanding* a created model. Three requirements relate to this. Firstly, the interaction with the tool depicts the actual activities that a consultant performs to develop a model (for instance the tool allows the management consultant to 'drag-and-drop' a modeling element onto a virtual worksheet) and specifies the model (entering parameters of for instance a task duration). Secondly, the modeling language should connect with the *mental model* of a consultant (how the real business processes are perceived by the consultant). And thirdly, a business process model developed using the modeling language should also be understandable by the client in order for him to validate the correctness of the business process model.

Requirement 4.2. *The tool should accommodate an intuitive approach for management consultants to develop and specify a business process model.*

Requirement 4.3. *The tool should provide a modeling language to represent business processes which connects with the mental model of a management consultant.*

Requirement 4.4. *The tool should provide a modeling language to represent business processes which is understandable to all stakeholders.*

In Section 3.2.2 we distinguished four main dimensions on which business process redesign may have an influence, namely *time*, *cost*, *quality* and *flexibility*. A consultant

wants to be able to evaluate what effect a redesign effort causes on one or more of these dimensions, but it depends on the business case in what parameter a consultant is specifically interested in. There are however some general output statistics, like for instance 'the average waiting time' of a queue (a place where entities need to wait) or 'the total time in system' of an entity. Based on these generic output statistics, the influence of a business process change can be evaluated. It is however important that the output is relevant and presented in an understandable way.

Requirement 4.5. *The tool should present relevant statistics of a simulation model execution in an understandable manner.*

For a simulation engine to be able to execute a simulation model, some parameters should be defined. These parameters describe the run-setup of a simulation model, like for instance the run-length (e.g., one day, one month, . . . etc) and default time unit (e.g., seconds, minutes, . . . etc).

Requirement 4.6. *The tool should allow specification of the simulation run set-up in a manner understandable to management consultants.*

In Section 3.3 we mentioned the limitation of current BPS tools. Especially the transformation of a conceptual model into an executable simulation model is inadequate in most BPS tools. For a solution to be usable by consultants it should support or provide some mechanism which transforms a conceptual business process model into an executable simulation model.

Requirement 4.7. *The tool should be able to transform a conceptual business process model into an executable simulation model.*

4.2.3 Requirements related to: Usage

Business process models are often developed at the location of a client using the notebook of a management consultant. In order to support a consultant in developing and executing a business process model, the tool should run on a default consultant notebook. A default notebook is considered to be not a high-end notebook or a notebook with specifications which are too outdated.¹

Requirement 4.8. *The tool should be executable on a default management consultant's notebook.*

Business process of different organizations can differ a lot from each other and it is possible that a proposed modeling language is too limited to adequately represent a certain business process. To support flexibility for modeling of any future business process, it is important that a proposed modeling language is extendible with additional modeling elements (when needed).

¹The default notebook of Accenture consultants at the moment of writing is a Dell Latitude E5410, equipped with an Intel-i5 2.40 GHz quad-core processor, 4.0 GB working memory and running Microsoft Windows 7 operating system

Requirement 4.9. *The modeling language should be extendible with additional modeling elements.*

Business process redesign decisions are often made by multiple actors (people that directly involved in the decision process) and affect or can influence multiple stakeholders. It is thus preferred that a tool is operatable in a multi-actor and multi-stakeholder context [KS08]. This relates to both conceptual model development as well as model execution and output analysis.

Requirement 4.10. *The tool should enable conceptual model development and model execution in a multi-actor and multi-stakeholder context.*

4.3 Overview requirements and evaluation criteria

The purpose of stating design requirements is to guide the design and development process of a solution, and to be able to evaluate whether the requirements are met. For that purpose, the requirements mentioned in Section 4.2 are specified with a certain performance measure, the target performance and the way of measurement. The performance measure depicts on basis of what characteristic the requirement can be evaluated. In other words: *how can we test whether the requirement is met?* The target performance depicts what the target performance is of the performance indicator to evaluate the requirement as being met. Finally, the way of measurement depicts how we suggest to evaluate each requirement and its performance regarding the final designed solution.

The evaluation criteria for the requirements related to the usefulness of a solution are shown in Table 4.1, the requirements related to usability are shown in Table 4.2 and the requirements related to usage are shown in Table 4.3.

Performance measurement usefulness requirement

An important aspect of a simulation tool is that it is able to simulate the dynamics of business processes and is included through Requirement 4.1. Based on a proposed design (see Chapter 5), a prototype will be developed (see Chapter 6). The evaluation of the requirement will be done by evaluating the prototype, namely by developing several business process simulation models and executing these. The outputs of these models will clarify whether the solution is able to simulate business processes.

ID	Requirement	Performance measure	Target performance	Way of measurement
4.1	The tool should be able to simulate the dynamic behavior of events and interactions within business processes.	Ability of the tool to simulate dynamic behavior of business processes.	The tool can simulate the dynamic behavior of events and interactions within business processes.	Evaluation of several differently specified sample cases.

Table 4.1: Usefulness requirements and evaluation criteria

Performance measurement usability requirements

Requirement 4.2 relates to the user-computer interaction with the solution, namely the way a consultant interacts with the tool itself to model business processes. To evaluate to what extent the requirement is met, usability testing is elaborately discussed in literature and provides various methods to apply (see for instance [Bar02]). The think-aloud procedure, where a participant interacts with the tool in a testing environment and mentions out-loud any choices he or she makes, is considered useful for evaluating this requirement. In Section 6.5 we will elaborate on this.

Requirement 4.3 relates to the modeling language used by a consultant to describe business processes and to what extent the consultant's mental model (how he or she perceives a business process) can be mapped using the language. This can be evaluated by showing various business process models and validating whether these models are understandable by the consultant. By asking follow-up questions we can evaluate whether consultants consider the models and modeling language connects with their mental model.

Requirement 4.4 is a strong requirement, as it prescribes that a proposed modeling language should be understandable by *all* stakeholders. This includes both the management consultants who make the models and who need to interpret the models, as well as the client-side specialists. Although this requirement is difficult to fully meet in this research (due to the limited available time and resources), it provides an ultimate goal. Follow-up research may focus on evaluating and improving a suggested modeling language. What is important, is that the models should provide as much visual clarity to be easily interpretable. More information on understandability of process models can be found in [Lar08].

Because it is unknown on beforehand in what specific information a management consultant is interested, the evaluation criterion of Requirement 4.5 is that the presentation of simulation outputs should be customizable by the end-user. Depending on the parameters of interest, the end-user should be able to define how he or she wants to have the outputs of interest presented. Based on the implementation in the prototype, this will be evaluated with several management consultants.

The parameters to specify the simulation run set-up as well as the way how these parameters are entered, should be easily understandable and executable by consultants. The target performance of Requirement 4.6 states that a consultant should himself (or herself) be able to perform this task. It can be evaluated through an usability evaluation session with consultants and monitor their actions while specifying the simulation run set-up.

One of the main issues in Accenture's past simulation projects (as was discussed in Chapter 2) relates to the interaction between the consultants and the programming specialist. More specifically, the conceptual business processes models were not adequate to be fully translated to a simulation model: the developed simulation models were partly based on the interpretation of the programming specialist. Regarding the usability of the final solution, Requirement 4.7 is considered to be an important requirement as it promotes direct transformation possibilities from conceptual model into an executable simulation model. The designed solution (and finally the prototype) should incorporate this. The implementation will be assessed to validate whether the requirement is met:

whether based on a conceptual business process model an executable simulation model can be generated.

ID	Requirement	Performance measure	Target performance	Way of measurement
4.2	The tool should accommodate an intuitive approach for management consultants to develop and specify a business process model.	Simplicity of developing a conceptual model.	A consultant should be able to learn developing business processes models using the tool easily.	Usability evaluation and follow-up interview and questionnaire with participants.
4.3	The tool should provide a modeling language to represent business processes which connects with the mental model of a management consultant.	Compatibility of modeling elements with mental model of consultant.	A consultant should be able to express his view on business processes in a model using the supplied modeling language.	Evaluation workshop and follow-up qualitative interview with stakeholders.
4.4	The tool should provide a modeling language to represent business processes which is understandable to all stakeholders.	The simplicity and clarity of a business processes model.	Stakeholder should be able to understand the business processes represented in a business process model.	Evaluation with stakeholders and follow-up (qualitative) interview with the participants.
4.5	The tool should present relevant statistics of a simulation model execution in an understandable manner.	Understandability of output statistics.	Output statistics are presented to the stakeholders in a customizable manner (data table, histogram, etc).	Assessment on the implementation, as well as evaluation with stakeholders.
4.6	The tool should support the specification of the simulation run set-up in manner understandable by management consultants.	Simplicity of specifying the simulation run set-up.	A consultant should be able to specify the simulation run set-up.	Evaluation and follow-up qualitative interview with stakeholders.
4.7	The tool should be able to transform a conceptual business process model into an executable simulation model.	Implementation of a translation method.	Based on the conceptual model, an executable simulation model should be generated.	Assessment on the implementation.

Table 4.2: Usability requirements and evaluation criteria

Performance measurement usage requirements

The solution should be used on a default consultant's notebook (Requirement 4.8). By developing a prototype using a default consultant's notebook and modeling and executing several sample cases, we validate whether this requirement is met.

Requirement 4.9 is important as it promotes extendibility for possible future modeling elements (which are not included in the proposed design). Continuity will thus be guaranteed by providing flexibility. To evaluate whether this requirement is met, the final design and prototype will be assessed to check whether it is possible to extend it with one or more modeling elements.

Because management consultants work in project teams and often work together with one or more client-side specialists, a preferred property is that the solution can be used in a multi-actor and multi-stakeholder context (Requirement 4.10). Although evaluation by testing this in an actual multi-actor context is not feasible (due to the limited amount of time available), we can discuss the final solution and suggest future research possibilities.

ID	Requirement	Performance measure	Target performance	Way of measurement
4.8	The tool should be able to be operated and executed on a default management consultant's notebook.	Execution performance evaluation.	The tool can be used on a standard notebook, which has a hardware configuration comparable with a consultant's notebook.	Testing on a default consultant's notebook.
4.9	The modeling language should be extendible with additional business process elements.	Possibility to add custom elements to the business process modeling language.	The tool supports the addition and reuse of an additional component.	Assessment on the implementation.
4.10	The tool should support model development and model execution in a multi-actor and multi-stakeholder context.	Number of stakeholders that can be involved in the model development process.	At least 2 stakeholders should be able to work together in developing a business process model.	Evaluation and follow-up qualitative interview with experts.

Table 4.3: Usage requirements and evaluation criteria

4.4 Conclusions

In this chapter the design requirements are defined which a solution to the problem should meet. These requirements are based on the inputs provided by management consultants, as well as on background literature. The requirements are categorized based on three main requirements for decision support systems, namely usefulness, usability and usage. Requirements that are considered as most important to fulfill within this research, are:

1) whether the tool is actually able to simulate the dynamics of events and interactions within business processes; 2) the understandability of a proposed modeling language and the output statistics; and 3) the direct relation between a conceptual business process model and an executable simulation model based on the conceptual model.

performance measures are identified for each requirement, as well as the target performance and the way of measurement. These evaluation criteria are more qualitative than quantitative: it is possible that they are not fully met within this research, but it is important to consider to what extent they are met. This will propose a basis for future research.

Christina arrives at the office of her client, a week after her last meeting with the student, when her phone rings. She answers the call and finds out that it's the student. He asks her whether it would be possible to meet next week. Apparently, he started to make a design of the solution and wanted to ask her some questions about business processes and evaluate some design choices.

The next week, she arrives at the room that the student reserved. Another consultant is already in the room, as well as the student. After getting a cup of coffee for everybody, the student begins his presentation. He explains them why they are here. "I would like to know more about how you work in a real project with the client. Unfortunately it was not possible to join you for a day," the student tells. "However, I've prepared a sample business process case, which is actually based on one of your previous projects. I would like you to see me as a client-manager, and try to make a model of my business processes. I've developed a set of modeling elements, which you can use to draw the model."

Quite soon after Christina asks her first question to understand the client's situation, a discussion starts. The discussions is about many different things. What is actually happening inside a business process? Is there something that arrives and gets processed? How do you call such a thing? An entity? And who performs the task in a business processes? A resource? And how can we visualize that in a model? Everybody actively uses the white-board to draw models to support their arguments. The 2 hours scheduled for the workshop fly by and many things are discussed. They decided for instance that a 'swimlane' is very useful to represent resources in a business process. A swimlane is a drawn rectangle in which you can place other modeling elements, like for instance a task. This depicts that some resource can perform a certain task. Although the meeting is already over, many things were discussed.

During the next month, they all meet several times for follow-up design workshops. Each time, they discuss different things and make decisions what would be good to be included. For instance, how are activities performed when two persons are involved? And how can you make this visible in a model? And what are you in general interested in to know about a business processes? And how do you see yourself using the solution. Finally, the student collected enough information to prepare his final design.

To design a solution which can support consultants in their business process modeling and simulation activities, we first need to determine what a solution should consist of. In other words, what needs to be designed, what can be based on academically founded knowledge and research and what can be adopted from already developed implementations? In Figure 5.1 the life cycle of a simulation study is shown again, but now we included a specification of the objects, actions or methods that need to be considered during the design phase (for the original figure, see Figure 3.8).

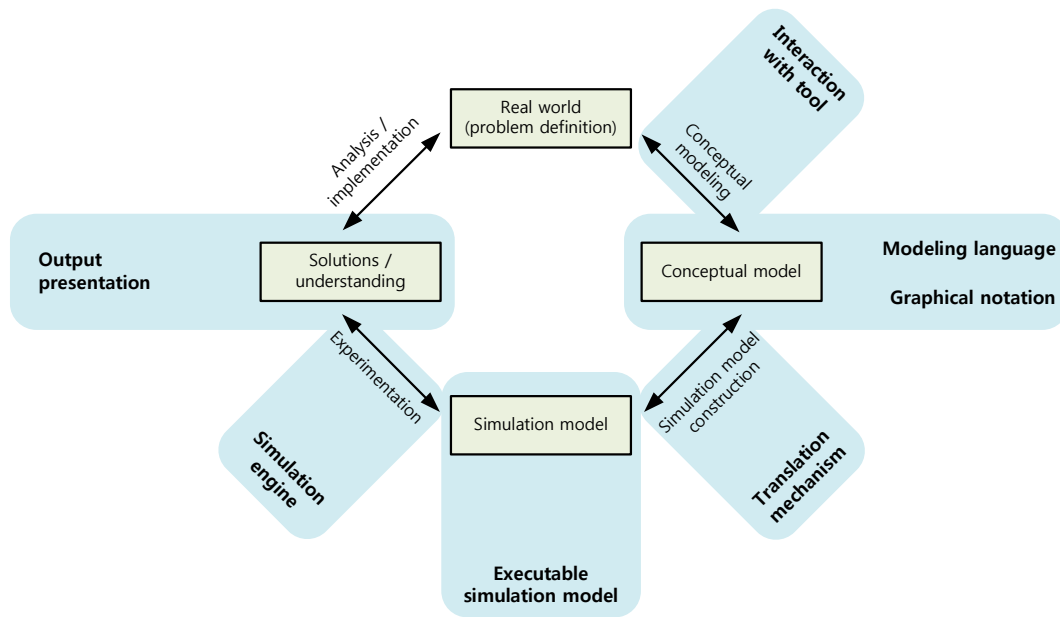


Figure 5.1: Overview of design choices

In Figure 5.1 two parts were left outside the scope of our solution design, namely the 'real world' and the 'analysis and implementation' part. Although the real world is clearly of importance to a consultant, the intention of this research and design is not to propose a new way of seeing and understanding an organization and its business processes (as in to propose a new *world view* to the consultants), but rather to support the current view of consultants on organizations and business processes. The analysis and implementation phase is considered outside our scope of research and design, because it involves the actual implementation of business redesign decisions. This depends among other things on the consultant and what he / she considers to be the best solution. However, the designed solution should support the consultant and the organization by offering the possibility to get an understanding what could be a good solution to some organizational problem or situation.

When we consider business process simulation from a consultant's point of view, the three main activities that he / she is interested in performing, are: 1) developing the business process model (conceptual model) and specifying the model parameters; 2) experimenting with a simulation model; and 3) interpreting the results. The focus of the design approach will be on these three activities. How the transformation of a

conceptual model into an executable simulation model can be done should be defined, but it is irrelevant to the consultant (as long as the simulation model leads to results as how the consultant intends it to do).

The outline of this chapter is as follows. In Section 5.1 we will elaborate on how the end-user (i.e., the consultant) is involved in the design process. Next, in Section 5.2 we will elaborate on the question *How does a consultant see organizations and business processes?* To support conceptual modeling by a consultant with a (new) modeling language, we first need to understand the mental model of a consultant. The mental model relates to the abstraction that a consultant makes when he analyses and maps business processes of an organization. To enable experimentation with a simulation model and interpretation of the outcomes, the mental model of a consultant should first be mapped by a consultant in a conceptual model which then needs to be translated into an executable simulation model.

In Section 5.3 we will propose a conceptual modeling language, based on the end-user input, and discuss how a conceptual model using the proposed modeling language can be transformed into an executable simulation model. We will elaborate on the use of simulation components, corresponding to each modeling element and on the actual transformation rules, allowing for manual or automatic transformation of the conceptual model into a simulation model. We then show several examples how conceptual models are transformed into simulation models, thus clarifying the execution of the stated transformation rules.

We continue this chapter with a discussion about the statistics collection and presentation aspects in Section 5.4. In Section 5.5, we elaborate on the interaction of the user with a tool and the inclusion of the tool in the actual process of consultants. This section will also discuss how the conceptual modeling language, simulation model transformation and execution can be implemented in an actual tool.

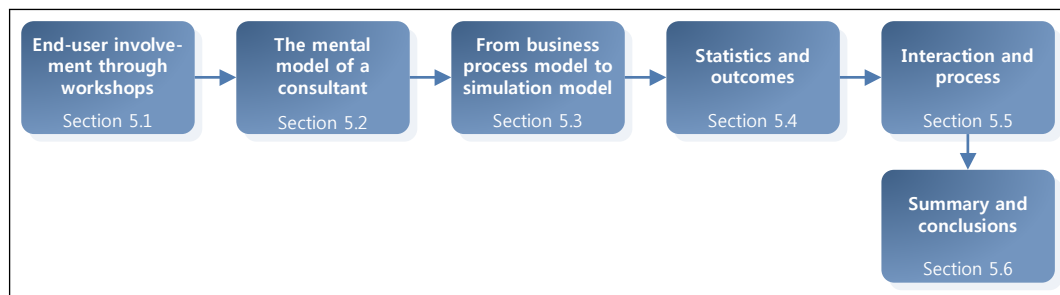


Figure 5.2: Outline Chapter 5

5.1 End-user involvement through workshops

The main goal of a user-centered design (UCD) approach is to increase the likelihood that a designed and developed artifact is found usable by its end-users. Various usability methods are available that can be used as part of a UCD approach and are elaborately discussed in [Mag01]. According to Maguire (2001), the user's perspective should be

incorporated into the software development process to achieve a usable system. As stated in Chapter 1, the main framework chosen for this research is the design science research framework. This framework promotes that a design should meet the requirements of relevance. A UCD approach is considered most suitable as the underlying design approach for the design part of this research. In our consideration, it is of the utmost importance that a chosen solution is considered practical and usable by the end users, and as such, the design should be tailored around the consultants needs and expectations.

We adopted a UCD approach by actively involving the end-user from the start until the end of the design process through a series of interactive discussion and design sessions, as well as expert-interviews. In total, three sessions (each with an average duration of 2 hours) were held with management consultants from Accenture. These sessions had several goals, namely:

- To understand the consultant's view on business processes;
- to propose and evaluate design options; and
- to make design choices.

The participants of the workshops all had some experience with and knowledge about business processes and simulation. It could have been useful to also include participants with no simulation experience, as they might pose a different view on business processes, on the behavior of the processes and on the usability requirements of a solution. However, to enable discussions about concepts related of modeling and simulation, participants would at least need a basic knowledge of this. Due to the limited time available, the choice was made to only include management consultants with some simulation experience, because it was considered not feasible to include management consultants with no experience and to first educate them with the basics of simulation.

We will now briefly discuss the first held evaluation session to give the reader an idea of how it was organized, what the goals were and what the results were. For more information on the first and the other sessions, the user is referred to Appendix D.

5.1.1 Example: First design workshop

It was advised by several experts (both academic and industry design specialists) that it is suggestible to observe a management consultant when they perform activities in their normal working environment, namely at a client firm's location. By applying the principle of "fly-on-the-wall" it would be possible to see how a consultant interacts with a client-side stakeholder and see what actions he actually performs. Observing how a consultant *actually* models business processes would provide a better idea for the end goal of the design process. It would also provide a better understanding of the context in which the final solution will be used. However, the problem was that it is difficult to plan when consultants would actually map business processes, and when they are performing other activities. This would mean they had to be observed several times, in the hope that some interesting (for this research) activities would be performed.

An alternative approach is to mimic a client's modeling case. The consultant is assigned with the task to get an understanding of the client's business processes and map

these in a business process model. This would still create some understanding of how a consultant works, thinks and how he asks questions. This concept would also give us a clearer understanding of what possible complications are for consultants to model and simulate business processes.

A workshop was finally scheduled with a total duration of 2 hours. The goals were to let a consultant model a sample case using a set of proposed modeling elements in order to understand his way of working; to understand to what extent the modeling elements are adequate and understandable; and to discuss several other modeling and simulation aspects. The sample case was based on a business process model developed as part of an earlier modeling and simulation project of Accenture. The modeling elements were based on the BPMN notation and some supplementary elements adapted from the Arena simulation software¹. In Figure 5.3 some of the selected modeling elements are presented. The practical execution of the modeling process was to use A5-sized sheets of paper with the different modeling elements printed on. The consultants sticks these sheets in a preferred sequence on a magnetic wall and connect these using an ink-marker to create a model (see Figure 5.4). They could specify the modeling elements to include information about the process itself.

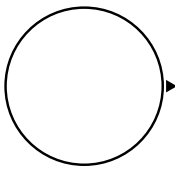
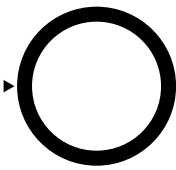

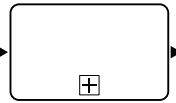
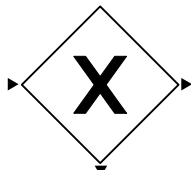
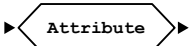
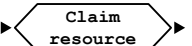
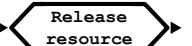
Start event	End event	Task	Sub-process																																
Name: 	Name: 	Name: 	Name: 																																
Specification of component parameters <table border="1"> <tr><td>Entity type</td><td></td></tr> <tr><td>Time between arrivals</td><td></td></tr> <tr><td># entities that arrive</td><td></td></tr> <tr><td>Maximum # entities</td><td></td></tr> <tr><td>Custom parameter #1</td><td></td></tr> </table>	Entity type		Time between arrivals		# entities that arrive		Maximum # entities		Custom parameter #1		Specification of component parameters <table border="1"> <tr><td>Collect data?</td><td>Yes (default) / No</td></tr> <tr><td>Custom parameter #1</td><td></td></tr> <tr><td>Custom parameter #2</td><td></td></tr> </table>	Collect data?	Yes (default) / No	Custom parameter #1		Custom parameter #2		Specification of component parameters <table border="1"> <tr><td>Delay duration</td><td></td></tr> <tr><td>Use resource?</td><td>Yes / No</td></tr> <tr><td>Resource name</td><td></td></tr> <tr><td>Custom parameter #1</td><td></td></tr> <tr><td>Custom parameter #2</td><td></td></tr> </table>	Delay duration		Use resource?	Yes / No	Resource name		Custom parameter #1		Custom parameter #2		Specification of component parameters <table border="1"> <tr><td>Custom parameter #1</td><td></td></tr> <tr><td>Custom parameter #2</td><td></td></tr> </table>	Custom parameter #1		Custom parameter #2			
Entity type																																			
Time between arrivals																																			
# entities that arrive																																			
Maximum # entities																																			
Custom parameter #1																																			
Collect data?	Yes (default) / No																																		
Custom parameter #1																																			
Custom parameter #2																																			
Delay duration																																			
Use resource?	Yes / No																																		
Resource name																																			
Custom parameter #1																																			
Custom parameter #2																																			
Custom parameter #1																																			
Custom parameter #2																																			
Exclusive gateway	Modify entity attribute	Claim resource	Release resource																																
Name: 	Name: 	Name: 	Name: 																																
Specification of component parameters <table border="1"> <tr><td>Number of exit ports</td><td></td></tr> <tr><td>Expression</td><td></td></tr> <tr><td>Custom parameter #1</td><td></td></tr> </table>	Number of exit ports		Expression		Custom parameter #1		Specification of component parameters <table border="1"> <tr><td>Name Attribute #1</td><td></td></tr> <tr><td>New value</td><td></td></tr> <tr><td>Custom parameter #1</td><td></td></tr> <tr><td>Custom parameter #2</td><td></td></tr> </table>	Name Attribute #1		New value		Custom parameter #1		Custom parameter #2		Specification of component parameters <table border="1"> <tr><td>Name resource</td><td></td></tr> <tr><td>Number to claim</td><td></td></tr> <tr><td>Priority</td><td>High / Normal (default) / Low</td></tr> <tr><td>Custom parameter #1</td><td></td></tr> <tr><td>Custom parameter #2</td><td></td></tr> </table>	Name resource		Number to claim		Priority	High / Normal (default) / Low	Custom parameter #1		Custom parameter #2		Specification of component parameters <table border="1"> <tr><td>Name resource</td><td></td></tr> <tr><td>Number to release</td><td></td></tr> <tr><td>Custom parameter #1</td><td></td></tr> <tr><td>Custom parameter #2</td><td></td></tr> </table>	Name resource		Number to release		Custom parameter #1		Custom parameter #2	
Number of exit ports																																			
Expression																																			
Custom parameter #1																																			
Name Attribute #1																																			
New value																																			
Custom parameter #1																																			
Custom parameter #2																																			
Name resource																																			
Number to claim																																			
Priority	High / Normal (default) / Low																																		
Custom parameter #1																																			
Custom parameter #2																																			
Name resource																																			
Number to release																																			
Custom parameter #1																																			
Custom parameter #2																																			

Figure 5.3: Sample selection paper prototype modeling elements

To conclude: the workshop itself, as well as the outcomes of the workshop, were very interesting. Although the original plan of consultants modeling the complete sample case

¹The reason for choosing BPMN as the modeling notation during the first workshop will be elaborated on in Section 5.3.1.

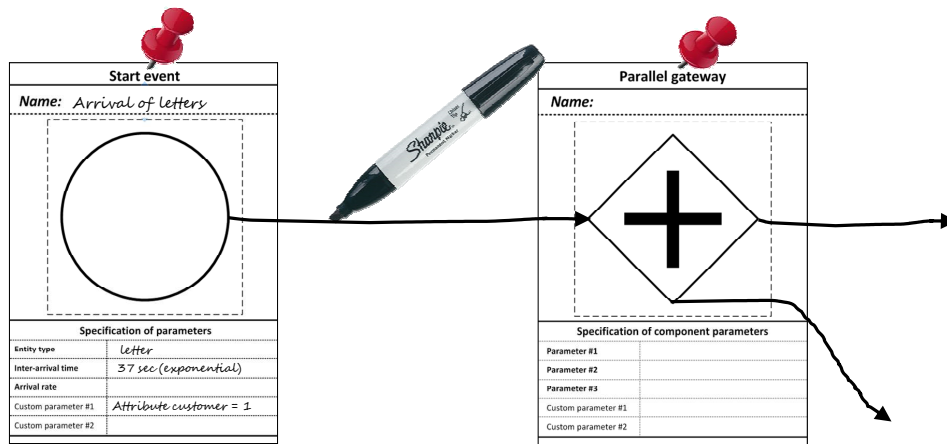


Figure 5.4: Example how participants were demonstrated to use modeling elements



Figure 5.5: Workshop participant uses proposed modeling elements to model sample process (workshop no. 3 - May 24, 2011)

was by far not reached (Figure 5.5 shows a photo taken during a later workshop), but it triggered some useful discussions. During the workshop discussions immediately started about several modeling elements and actual business processes. Supported by white paper (see as an example Figure 5.6 and Figure 5.7) the participants were triggered to think about concepts and discuss these. One of the main contributions of this workshop was the design choice that was made to use Swimlanes to depict the resources in a

business process and to use the crossing of the boundaries as the places where resources are claimed and released (this concept will be explained more elaborately later on in this chapter).

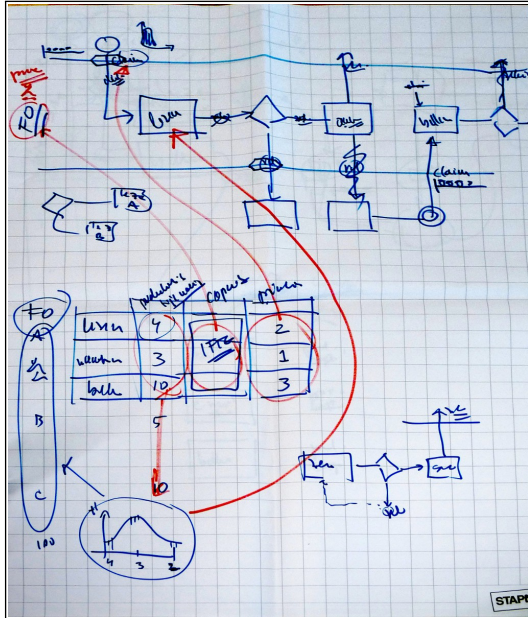


Figure 5.6: White board sample sheet 1

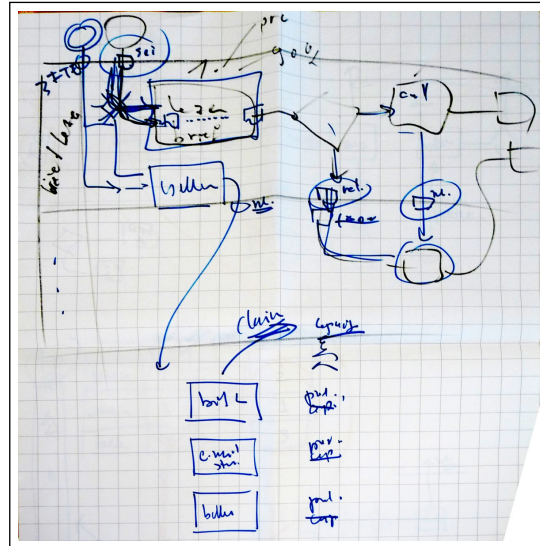


Figure 5.7: White board sample sheet 2

5.2 The mental model of a consultant

Entities

A business process is seen by consultants as a *series of activities and decisions which are performed by resources and which influences the flow and state of entities*. An **entity** (or passive entity) is an abstract object and can represent anything that undergoes activities in a business process. The entity arrives at an organization, 'flows' through the business process(es), and then leaves the organization. What the entity actually represents, is called the **entity type**. Examples are: an order that is received by a company and needs to be processed; an insurance claim; a phone call with a customer question; or a contract cancellation e-mail.

During a business process, an entity undergoes state changes as a result of the activities that are performed by resources. To be more strict, an activity *always* changes the **state of an entity**. To explain the concept of an entity state, consider the following example. When an entity of type 'order' arrives at a company, its state is 'to-be processed'. After the order is being processed, the state changes to 'processed'. The type of an entity can also change due to an activity. Consider for instance again an order that arrives at a company. After an order is being processed, an invoice should be created and send to a customer. To generate an invoice a new document may be created by a resource, but this invoice is associated directly to the order. In this case, the entity type changes from

'order' to 'invoice'.

Entities can arrive separately at a business process, or in batches. To explain, it may occur that one department of an organization collects certain entities (like for instance insurance claims). After a certain amount of claims are collected, these claims are forwarded all at once to another department which will handle these insurance claims. The arrival of entities may be constant over time (e.g., every 10 minutes an entity arrives), or it can be more random.

Resources and entity queues

resources are typified by their capability, role, capacity, availability and state

A **resource** is seen as the leading part of a business process who is responsible for making decisions and performing activities on entities. A resource can be a human or a machine. Resources are typified by their capability, role, capacity, availability and state. The capability of a resource depicts whether a resource is able or allowed to perform a certain activity. Based on for instance the experience that a resource has, the resource may be able to perform more complicated activities. The collection of all capabilities of a resource is called the resource's role. It is possible that within an organization, multiple resources have the same capabilities and thus share the same role. In that case, a role has a certain capacity: the number of resources that are available to perform a determined set of activities. A role can also be considered as a 'container' of resources.

The availability of a resource depicts when or for how long a resource is available to commence activities. For instance, a full time employee has an availability of 1 FTE (Full Time Equivalent), which depicts that during a complete working day the resource is available to perform activities. The state of a resource relates to whether a resource is currently busy (or active) with performing a certain activity on an entity, or is available. In case the resource is available, he or she will look for queues that hold entities that need to be processed.

A queue occurs in several cases, namely when an entity arrives and waits to be processed by a resource; when a resource hands over an entity to another resource, which is not available yet; when an entity needs to wait for some time before an activity can be repeated or until an event occurs. There are two possible cases that relate to the resource selecting an entity to process. The first is that there are one or more entities waiting to undergo the same series of activities. These entities are held at the same place, namely in a queue. The second case is that there are multiple queues where entities are held and it differs per process which activity is first undertaken.

In the first case, a resource may select an entity from a queue that has the longest waiting time, following the so-called 'First In - First Out' (FIFO) principle, or the resource may select an entity that with the shortest waiting time (Last In - First Out, or LIFO). In the second case there are multiple queues which all hold one or more entities. The way how a resource chooses a queue from which an entity is taken, can be different per organization and business process. However, there are some frequently identified manners how a resource selects a queue to take a waiting entity from:

- **Random:** A resource selects a queue with waiting entities on a random manner
- **Cyclical:** A resource selects a queue in a cyclical order. To clarify, first an entity is taken from the first queue and the resource performs certain activities. Next,

an entity from the second queue is taken, and so forth. When from every queue an entity is taken, the resource repeats the cyclical procedure, by taking again an entity from the first queue.

- **Cyclical - repeat:** In this case, a resource also selects queues in a cyclical order, but instead of taking only one entity from a queue before continuing to the next queue, he repeatedly for a certain number of times takes entities from a queue, before continuing to the next queue.
- **Priority:** A resource can select a queue based on the priority of a certain activity, like for instance almost finished entities will be processed before newly arrived entities.
- **Queue characteristics:** A resource may select a queue based on the characteristics of that queue, for instance the number of entities waiting to be processed, or the current average waiting time of the entities that are waiting to be processed.

Activities and decisions

An activity is a piece of work performed by one or more resources which requires a certain amount of time. An activity can be a task or a grouping of task, called a sub-process. A **task** is a piece of work that cannot be subdivided into smaller pieces of work to be performed by a resource, or is not crucial for the purpose of describing and analyzing a business process. Consider for instance the task 'write letter to customer'. This task could be subdivided into 'start up word processor', 'write letter to customer', 'perform spelling and grammar check', 'store document on hard drive', ... etc. However, often when a task duration is too short (compared to other tasks), it is considered insignificant for the overall business process and are grouped as being one identifiable task. When several tasks contribute to a larger activity and all are of relevance to the process, they can be grouped into a **sub-process**. A sub-processes itself can also be part of a larger process, containing other sub-processes and task. The concept of tasks and sub-processes provide the means to decompose a business process hierarchically.

There are three options how an activity can be performed, namely 1) one resource starts and finishes an activity on its own; 2) a resource hands over the entity to another resource who will perform one or more activities; or 3) the amount of work is divided over two or more resources. In the first case, the flow of an entity through activities is called sequential. In the second case, a resource will give the entity to another resource who will perform his activities. The third case is called a parallel activity. After the work is split up, two or more resources will perform their activities independently. At some moment the work is synchronized again and some resource will continue performing activities. Because often one resource finishes his work before the other resources, the question is whether he will wait until the other parts are finished and work can be synchronized, or he leaves his work somewhere, and continue performing other tasks. In real business processes it is most often the case that a resource will look for new work (i.e., a waiting entity), instead of waiting until all other resources are finished and work can be synchronized. After all partial activities are finished, the entity waits until a resource is available with adequate capabilities to continue the process.

In a business process **decisions** are made that influence the choice and order of activities to be undertaken by a resource. A decision can depend on the attribute of an entity (e.g., entity type or state), or the state of the system (e.g., what are other resources doing, how many entities are waiting to be processed, . . . etc). As an example, consider an 'insurance claim' entity. The size of the damage for which insurance money is claimed, may influence the checks that are performed by employees and thus the activities that are performed by a resource. In a business process it is however sometimes unclear on what a decision depends, but is seen as a probability that suggests a different flow of entities.

5.3 From business processes to simulation model

In the previous section we elaborated on the main characteristics of business processes how they are recognized by consultants. In this section we will make the step from how consultants perceive business processes, to how they can model these processes conceptually, to finally how an executable simulation model can be made based on this conceptual model. First we will elaborate in Section 5.3.1 on the choice for BPMN as the basis for the proposed modeling notation for business processes. Next, in Section 5.3.2 we will present a set of business process modeling elements and the corresponding metamodel. In Section 5.3.3 we will provide the argumentation why we chose for the DEVS simulation formalism and why we formally define the modeling elements as simulation model components. In Section 5.3.4 we will elaborate on the formalization of the modeling elements as simulation components. In Section 5.3.5 we will elaborate on how a conceptual business process model can be translated into an executable simulation model and several transformation examples.

5.3.1 Formalization of conceptual model: BPMN

In Section 3.2.1 we discussed several modeling language that are widely used to represent business processes, like UML activity diagrams, BPMN and YAWL. We suggest a modeling language based on the BPMN specification as the main modeling language for consultants to model and simulate business processes, because of several reasons.

Firstly, the way how consultants see business processes, as described in Section 5.2, corresponds closely to how business processes are described in the BPMN specification [Obj10]. The BPMN specification provides a formal representation of business processes and although sometimes different names are used for business process elements compared with how consultants name the elements (e.g., token instead of entity, participant instead of resource), the BPMN specification provide means to represent the earlier mentioned concepts.

Secondly, many management consultants of Accenture have experience with modeling of business processes through the use of "swimlane diagrams". This method resembles BPMN to some extent, but is more simplified. Swimlane diagrams show what is done, by whom, and in what sequence - "who does what, and when" [SM01]. It uses a modeling element called a swimlane to represent the resources of a business process. Activities and decisions are placed within a swimlane to depict the capability of a resource to perform

A modeling language based on the BPMN specification is suggested for consultants to model and simulate business processes

certain activity and to make certain decisions. The BPMN also uses the concept of swimlanes to model resources (called 'participants' in BPMN) and to assign activities to a participant. Because of the experience of many consultants with modeling through swimlanes diagrams, we consider the use of swimlanes through the formal representation specified by BPMN as most appropriate.

Thirdly, the BPMN was used for conceptual modeling of business process in past projects for the development of Accenture's business planning decision support tools. Although the BPMN specification was not fully respected (some BPMN modeling elements were used with a different meaning than is specified in the BPMN specification), Accenture's management consultants accepted BPMN as a consistent and understandable notation for business processes.

And finally, the BPMN is becoming a standard language to model business processes through out industries [MR08]. Adopting a standard modeling language that is known across industries increases the likelihood that clients of Accenture are familiar with the notation and the models. If a client is known with the notation, development of a business process model and communication about developed models will become easier.

In Section 4.1 we stated (based on interviews with experts from Accenture) that a conceptual modeling language is preferably based on a formally specified business process modeling language. We consider BPMN as such a formal language. BPMN's basic modeling elements, which allow to represent relatively simple business processes, are however quite similar to elements used in simple informal flow diagrams. Our goal is to establish a set of modeling elements, based on BPMN elements, which are easily understandable by consultants and other stakeholders and have a rigorous and well-known foundation (and can thus also be understood by other stakeholders, not known with the in this research proposed modeling elements).

5.3.2 Proposed business process modeling elements

From the BPMN specification a subset of modeling elements was taken that allows to represent business processes and characteristics as were described in Section 5.2 [Obj10]. In Figure 5.8 an overview is given of the selected BPMN modeling elements. The metamodel of the proposed modeling language is shown in Figure 5.9. As can be distinguished in both figures, a categorization is made between Flow Objects (Events, Activities and Gateways), Sequence Flow, and Collaboration element (Swimlane). An Event represents something that "happens" during the course of a business process and influence the flow in the model. An Activity represents the work that is being performed in an organization. A Gateway is used to control the divergence and convergence of Sequence Flows in a business process. A Sequence Flow is used to show the order in which activities are performed. A collaboration element links activities in an organization with actors who perform the activities.

We will now briefly discuss each modeling element. For a more elaborate description of each modeling element, we refer the reader to Appendix E.

Start Event A Start Event represents the start of a business process. It depicts where entities arrive into an organization or a specific business process and it triggers activities

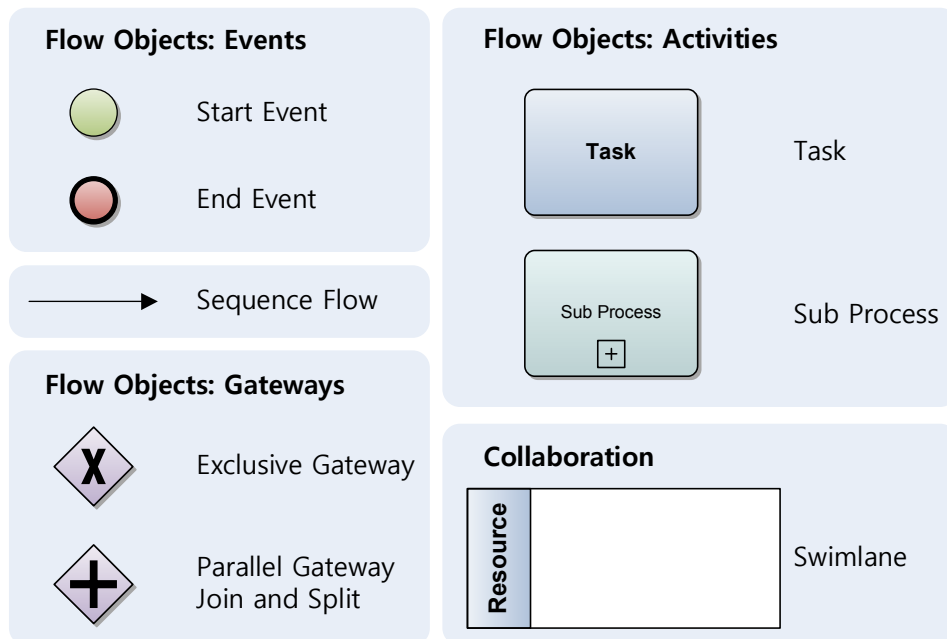


Figure 5.8: Overview selected BPMN modeling elements

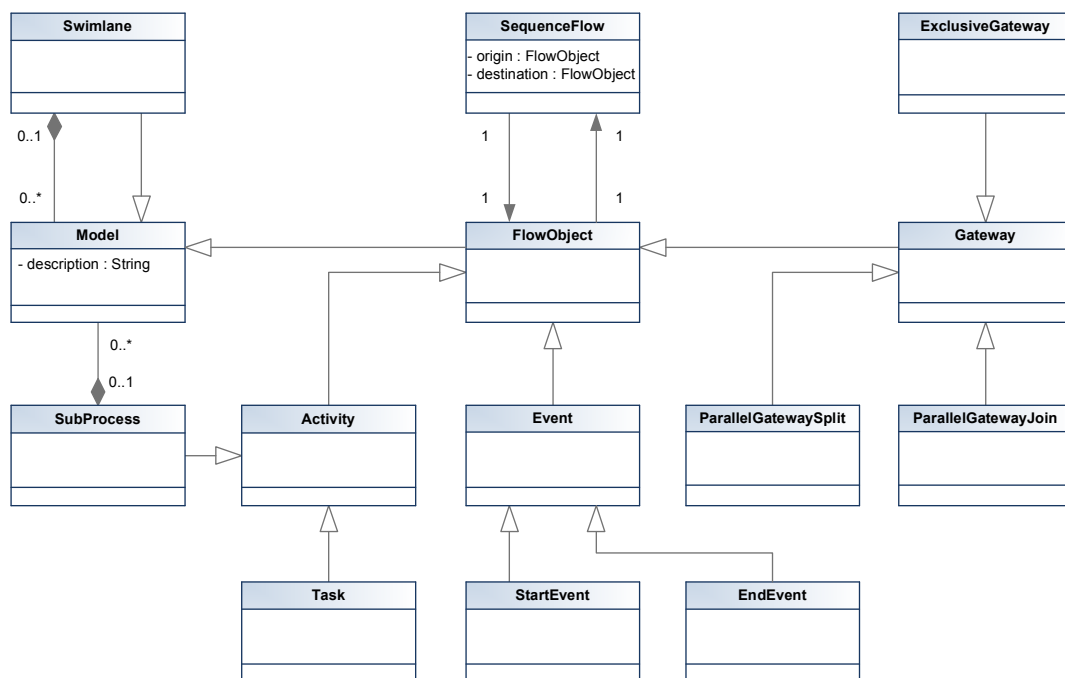


Figure 5.9: Metamodel proposed business process modeling language

to be performed. A Start Event can for instance represent the arrival of customer orders or phone calls. The graphical representation of the Start Event is a circle with a single

thin line. A description of the entities that arrive can be placed anywhere around, but in the vicinity of the element (as long as it can not cause confusion to which element the description relates). A Start Event can only have an outgoing Sequence Flow.

End Event An End Event represents the end of a business process, namely when an entity leaves the organization. It depicts the boundary of interest of a modeled organization. An End Event is graphically represented as a circle with a single thick line. A description of the entities that leave and / or the target of the entities that leave, can be placed anywhere around the element. An End Event can only have one or more incoming Sequence Flows.

Task A Task represents work that is performed by a resource and consumes a certain amount of time. The graphical representation of the Task is a rectangle with possibly rounded corners and a single thin line. The description of the Task is placed inside the rectangle. A Task can have multiple incoming Sequence Flows and only one outgoing Sequence Flow.

Exclusive Gateway An Exclusive Gateway is used to represent decisions made in a business process. The flow of an Entity is influenced when it arrives at an Exclusive Gateway, as it is directed in a certain direction, based on the decision that is made. The graphical notation of an Exclusive Gateway is a diamond and it is drawn with a single thin line. An "X"-marker may be placed inside the diamond. The Exclusive Gateway can have one or more incoming Sequence Flows and one or more outgoing Sequence Flows.

Parallel Gateway (Split / Join) A Parallel Gateway is used to model parallel activities in a business process. Parallel Gateways are used in pairs: one gateway is used to represent the start of a parallel activity (Parallel Gateway Split). A second Parallel Gateway is used later on in the process to synchronize a parallel activity again after all activities are completed (Parallel Gateway Join). A Parallel Gateway Split and Join modeling element are both identically graphically represented, namely as diamond shape and drawn by a thin single line. A "+"-marker is placed inside the diamond. The Parallel Gateway Split can have one or more incoming Sequence Flows and one or more outgoing Sequence Flows. The Parallel Gateway Join can have one or more incoming Sequence Flows, and only one outgoing Sequence Flow.

Sequence Flow A Sequence Flow (or Flow) is used to create a path between Flow Objects to depict the sequential order in which activities are undertaken by a resource on an entity. A Sequence Flow can only link one output-port of a modeling element with one input-port of another modeling element. A Sequence Flow is graphically represented as an arrow pointing in the direction in which an Entity flows.

Sub Process A Sub Process element is an Activity that can contain other Activities as well as Gateways. It is used to visually organize the business process model by hiding details. It provides a way of hierarchical modeling, and it may also contain other Sub

Processes. A Sub Process has two visual states: 'folded' and 'unfolded'. When a Sub Process is folded, it hides the internal details. When the Sub Process is unfolded, the internal modeling elements and Sequence Flows are visible. A Sub Process is drawn as a rectangle with possibly rounded corners and a single thin line. The description of the Sub Process is placed inside the rectangle. A \boxplus or \boxminus symbol placed in it depicts whether the Sub Process is respectively folded or unfolded. A Sub Process can have one or more incoming Sequence Flows and one or more outgoing Sequence Flows. These Sequence Flows are connected to the Flow Objects within the Sub Process, and not to the boundaries of the Sub Process.

Swimlane A Swimlane represents a resource or a group of resources that share the same capabilities of performing certain activities. A Swimlane can be horizontally or vertically oriented. It is graphically represented by a rectangle with on the left side or on the top (depending whether the Swimlane is horizontally or vertically oriented) a description of the resource or resource group it represents. The swimlane can contain Activity and Gateway modeling elements, as well as Sequence Flows to connect the elements. Events are however not allowed to be placed inside the Swimlane.

5.3.3 Components supporting simulation model construction

The next step in this research is to provide means how a conceptual model, based on the proposed modeling notation, can easily be transformed into an executable simulation model. To decrease the complexity and time needed to develop simulation models, reusing parts of simulation models or reusing even complete simulation models is suggested to be fruitful [Pid02]. There are four different types of model reuse and are shown in Figure 5.10. This figure shows four points on a non-linear scale with two axes: frequency and complexity. Code scavenging (reusing existing code of a simulation model) and function reuse (reusing predefined functions that provide specific functionalities) are frequently applied ways of model reuse and are relatively easy to apply, but have only a limited influence on the total development time of a simulation model. Applying model reuse, as in 'reusing a complete simulation model for a different situation', is found to be very complex to apply. The development of a generic and valid simulation model usable for different cases is also considered rather time consuming. Lastly, component reuse (reusing an encapsulated simulation module with well-defined interfaces) is considered to be fruitful concept to increase the efficiency of hierarchical model construction [CVS10].

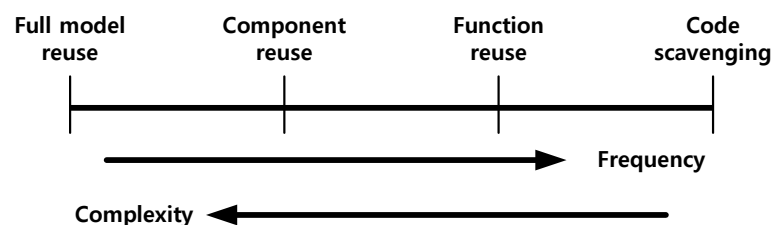


Figure 5.10: A spectrum of reuse (adopted from: [Pidd2002])

be coupled to form a composed model that represents a system. In [VV02] a list is presented that describe the characteristics of a modeling component (or building block). According to Valentin et al. (2002) a component should be: 1) self-contained (nearly-independent), 2) interoperable (independent of underlying technology), 3) reusable, 4) replaceable, 5) encapsulating its internal structure, 6) providing useful services or functionality to its environment through precisely defined interfaces, and 7) customizable in order to match any specific requirements of the environment in which it is used. These characteristics relate to domain specific building blocks (or components). If we consider a container terminal in a port for instance, we can represent this in a model as a combination of various domain specific simulation components, like ships, quay cranes, containers, Automated guided vehicles (AGVs) and Trucks.

Business processes as seen by consultants (see Section 5.2) can according to us also be represented in a simulation model using the concept of simulation components. These components are less considered to be domain specific building blocks (as they don't represent unique entities in a system, like for instance a truck or a quay crane), but they do represent specific element of the business process it self: so called low-level components. If we define aspects such as clear interfaces for each component, we can connect several components together and thus create a coupled model which represents a business process or part of a business process. Such a part of the business process could then also form a reusable domain specific component.

Defining low-level simulation components also allow us to apply the principles of Model-Driven Development (MDD) for the transformation of the conceptual model into an executable simulation model. MDD is a software development approach and is based on the essential concept to shift attention from programming software code to the use and transformation of models [LM08]. The idea is that a model expressed using some domain-specific modeling language (DSML) (which is described using a metamodel) can be transformed (semi-)automatically into another model which uses another modeling language (called model-to-model transformation). If for the target modeling language programming code is specified for each model element, this new model can then be translated into actual programming code (model-to-code transformation) [CVS11].

... Model-Driven
Development

For each modeling element which is depicted in Figure 5.8 we need to develop a representative simulation component. In order to avoid any ambiguity, the simulation components should be formalized using a well-founded technique [VHAR10]. We then also need to define exact transformation rules how a conceptual model (using the business process modeling elements) can be transformed into a simulation model. However, in order to apply MDD, we need to specify a simulation modeling language where a conceptual model is transformed into. This target modeling language should incorporate the concepts that allows us to finally implement it using some programming language and execute it using a simulation engine.

Different formalizations exist which allow to formalize simulation models, like for instance Petri Nets, DEVS, State Charts, Finite State Automata,... etc[Van00]. Because of several reasons we chose DEVS as being an appropriate formalism to express the simulation components in. In DEVS, a system is represented by two types of models: atomic and coupled models. Atomic DEVS models describe the behavior of a model, whereas coupled models describe the structure of a model. Atomic models can be

... choice for DEVS
formalism

integrated into coupled models, and coupled models itself can also be integrated in higher level coupled models (see Figure 5.11). This way, DEVS allows us to decompose a model in a hierarchical manner. DEVS is also component based, which makes it suitable to map conceptual modeling elements with simulation components. Furthermore, DEVS includes a well-defined formal specification [ZPK00]. And finally, one of the main reason for not choosing any of the other before-mentioned simulation formalisms, is that all other mentioned formalisms can be transformed to DEVS (see the “Formalism Transformation Graph” figure in [VLM02]), which makes DEVS a generic but powerful formalism.

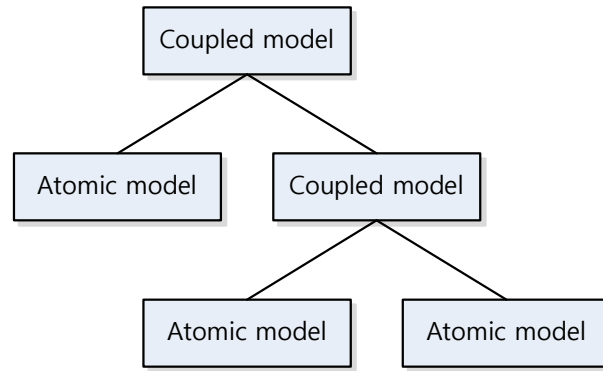


Figure 5.11: DEVS atomic and coupled model decomposition

A simplified metamodel of the DEVS formalism (adopted from [CVS11]) is shown in Figure 5.12. The combination of this metamodel together with the metamodel of the proposed business process modeling language (Figure 5.9) allows us to specify transformation rules, thus enabling the concept of conceptual to simulation model transformation.

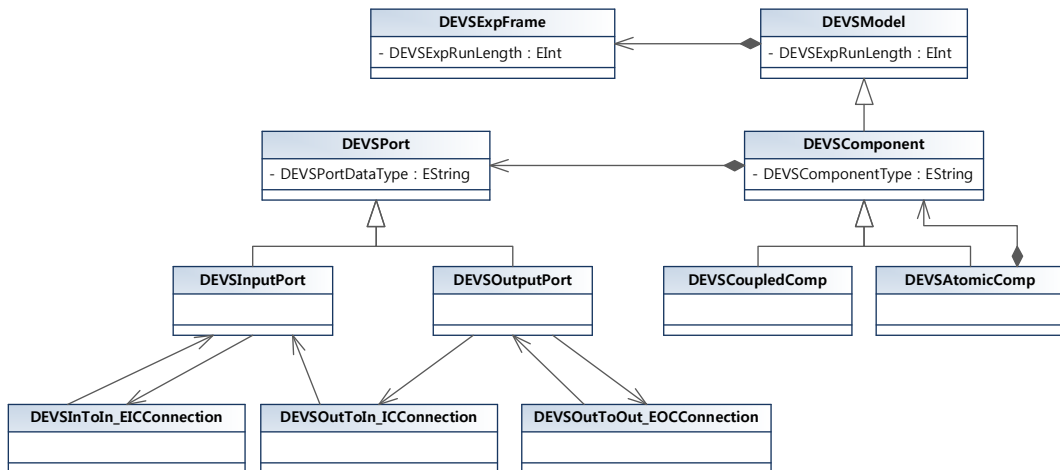


Figure 5.12: Simplified DEVS Metamodel (adopted from [CVS11])

To conclude, by specifying the selection of business process modeling elements as simulation components, we incorporate the advantages of developing a conceptual model

visually, with the advantages of providing possibilities to transform this conceptual model into an executable simulation model. This will provide the means for consultants to draw a conceptual model, which can then relatively easy be translated into a simulation model through simulation components. This transformation can be done by hand, if the transformation rules are specified unambiguously, or – more preferably – automatically.

5.3.4 Overview of simulation components

As mentioned in the previous section, a system in DEVS is represented by atomic and coupled models. A basic atomic DEVS model is defined by the following tuple [ZPK00]:

$$M = (X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta)$$

where:

X	Set of <i>input ports</i> through which <i>external events</i> are received
S	Set of <i>sequential states</i>
Y	Set of <i>output ports</i> through which external events are sent
δ_{int}	<i>Internal transition</i> function, which specifies to which next state the system will transit after the time given by the time advance function has elapsed
δ_{ext}	<i>External transition</i> function, which specifies how the system changes state when an input is received
λ	<i>Output</i> function which generates an external output just before an internal transition takes place
ta	<i>Time advance</i> function which controls the timing of internal transitions

An atomic model remains in state $s \in S$ for a time $t_s = ta(s)$. After t_s elapses, an *internal event* is triggered and the state of the atomic model changes to $s_{new} = \delta_{int}(s)$. Just before this internal transition occurs, an output can be generated based on the state previous to the event and the *output function* ($output = \lambda(s)$). Following the internal transition, a new internal event is scheduled at time $t_{s_{new}} = ta(s_{new}) + time$. The time left in a state is defined by $\sigma = ta(s) - e$.

If an input is received on one of the defined input ports, an *external event* is triggered. The state changes to $s_{new2} = \delta_{ext}(s, e, x)$, where s is the current state, e is the elapsed time since the last transition and x is the external input causing the event. The time left in a state (before the internal transition would have occurred, if no external event was received) is defined by $\sigma = ta(s) - e$.

A coupled model is defined by the following tuple [ZPK00]:

$$M = (X_{self}, Y_{self}, D, \{M_d\}, EIC, EOC, IC)$$

where:

X	Set of <i>input ports</i> through which <i>external events</i> are received
Y	Set of <i>input ports</i> through which <i>external events</i> are sent
D	Set of <i>component names</i>
$\{M_i\}$	Set of components: each must be an atomic or coupled DEVS model
EIC	<i>External Input Coupling</i> : connections between the input ports of the coupled model and its internal components
EOC	<i>External Output Coupling</i> : connections between the internal components and the output ports of the coupled model
IC	<i>Internal Coupling</i> : connections between the internal components

Within a coupled model, messages can only be sent between atomic and coupled models through external input, external output and internal couplings. A message can contain any type of amount of information, depending on the modeled situation. These messages are generated through the external output functions, as defined by the atomic model, and are received by other models as external inputs. It is allowed to connect models in a coupled model in the form of 1-to-1, 1-to-many, and many-to-1.

For the formal specification of a simulation component, we use a graphical representation to depict the states and state transitions, of which a generic sample is shown in Figure 5.13. An explanation of the meaning of the used symbols is also provided.

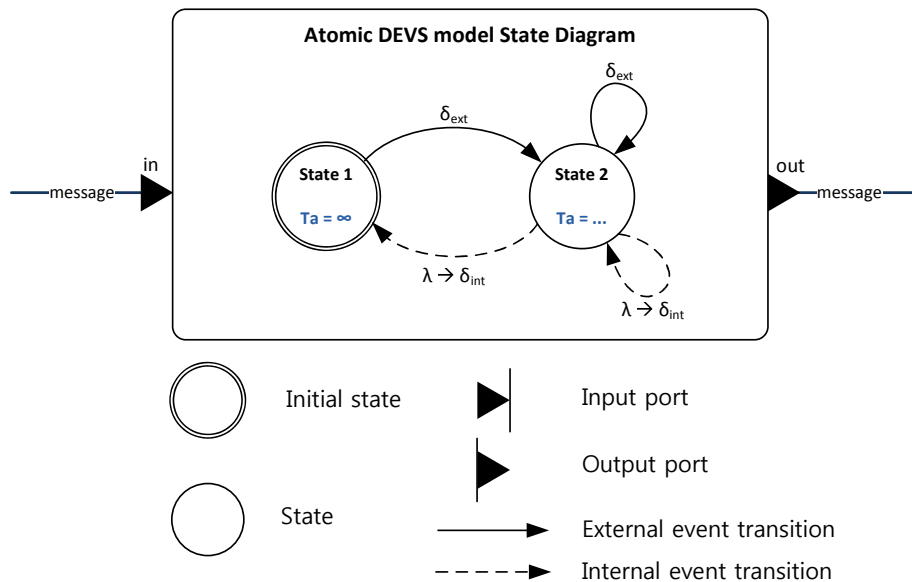


Figure 5.13: Sample atomic model State Diagram

The remainder of this section will elaborate on the components representing the

modeling elements, as were proposed in Section 5.3.2. Only the Start Event component, the End Event component, the Task component are discussed (all represented as atomic DEVS models), as well as the Swimlane component (represented as a coupled DEVS model). The reader is referred to Appendix E for a complete overview. To enable simulation however we also had to design three additional components which are not visually modeled. These components are needed in a simulation model to fulfill functions with regard to resource operations (such as requesting, assigning and releasing resources). These are the Swimlane Entry component, the Swimlane Exit component and the Resource Manager component (of which the first and the last will be discussed in this section). For an elaboration on how the components are coupled, see the following section (Section 5.3.5).

Start Event simulation component

The Start Event represents the arrival of entities into an organization or specific business process. The corresponding simulation component mimics this behavior by generating entities based on a certain specification. It is similar to the 'Create-block' as can be found in the simulation software Arena. The parameters describing its specification, are:

- **Inter-arrival time** Time between the arrivals of two entities or two batches of entities
- **Entity Type** Type of the entity that arrives
- **Batch Size** Number of entities that arrive at once
- **Time first arrival** Point in time at which the first entity (or entity batch) arrives

The formal description of the Start Event component is given by the state diagram as shown in Figure 5.14. This component has one output-port through which a newly created entity leaves, and one state, namely the 'Passive' state. The time duration after which an output function (λ) takes place is equal to the inter-arrival time as specified by the modeler. This inter-arrival time may be constant (e.g., every 10 minutes an entity is created), or following a statistical distribution (e.g., an entity is created on average after 5 to 10 minutes, following a uniform distribution). In case batches of entities arrive, the component generates the specified number of entities, and sends them to the output-port.

End Event simulation component

The End Event represents the end of a business process, namely when an entity leaves the organization. The End Event depicts the boundary of interest of a modeled organization. To mimic the behavior of an entity leaving the organization, the End Event is formalized as an atomic DEVS model with two main states, namely 'Passive' and 'Active' (see Figure 5.15). The component will remain in 'Passive' state, until an entity arrives at its input-port, triggering an external transition. Before the state changes back through an internal transition from 'Active' to 'Passive', the entity is disposed (i.e., destroyed).

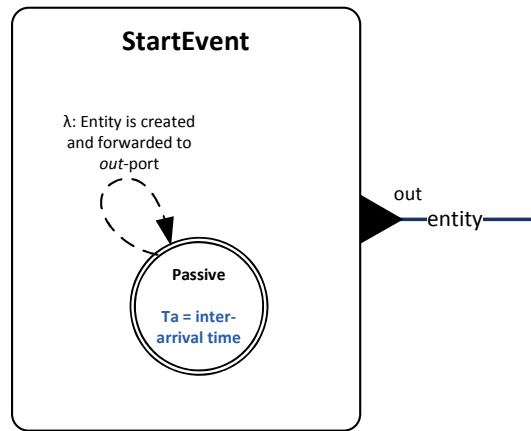


Figure 5.14: State diagram Start Event

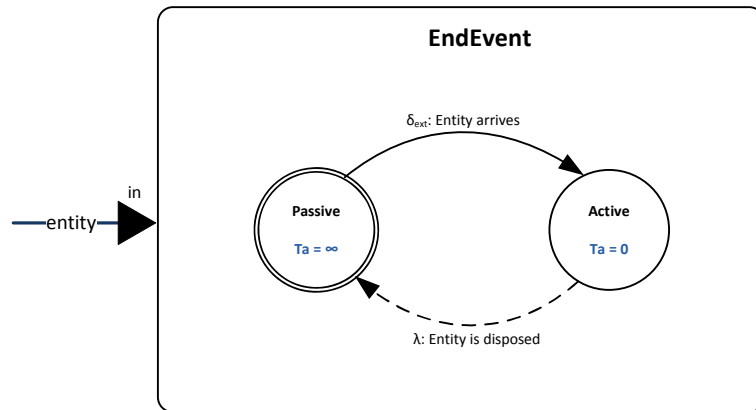


Figure 5.15: State diagram End Event

Task simulation component

A Task represents work that is performed by a resource and consumes a certain amount of time. To mimic behavior of a resource performing an activity for some time, the Task simulation component will delay an entity arriving at the input-port for a specified duration, before sending it to the output-port (see Figure 5.16). Because one kind of task can be performed by multiple resources at the same time (work performed on different entities), the Task component may contain multiple entities at a same instance.

When a Task is in 'Passive' state it means that no resource is currently performing that task. Due to an external event of an arriving entity, the state changes to 'Active'. If an entity arrives while the state is 'Active', the state remains 'Active' but the time advance is reset depending on which resource will be first finished with the task. If the duration of the newly arrived entity is shorter than the current shortest remaining duration of entities already being processed, the time advance function is set to the duration of this newly arrived entity. Else, the entity will be added to a list containing all entities that are currently processed.

When an internal transition occurs due to the elapsed duration of an entity, the state of the Task changes to 'Passive' if no more resources are currently performing that task, or remains in 'Active' state if one or more resources are performing that same task. The next internal transition event is scheduled for when the next entity with the shortest remaining duration. A Task is specified by the following parameters:

- **Task duration** Time that it takes to perform the task by a resource on an entity.
- **Entity Type Change** In case the task changes the entity type, the new type can be specified through this parameter.
- **Entity State Change** After a task is performed, the state of an entity changes which is specified by the parameter.

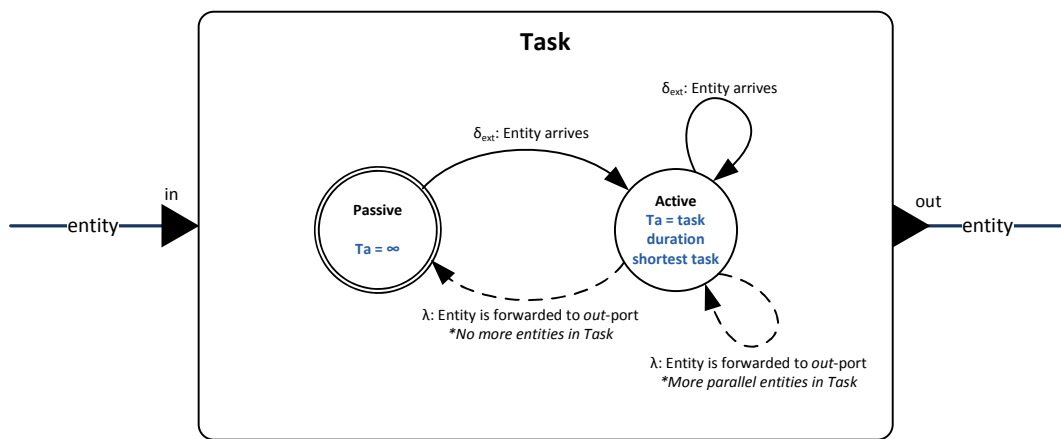


Figure 5.16: State diagram Task

Swimlane component

A Swimlane represents a resource or a group of resources that share the same capabilities of performing certain activities. The Swimlane is represented in DEVS as a coupled model. An entity arrives in a Swimlane after it is created by a Start Event which is placed outside the Swimlane. The point where the Sequence Flow crosses the boundary of a Swimlane, depicts the place where an entity will wait in a queue until a resource is available and can be assigned to work on it. After all activities in a Swimlane are performed, the entity can leave to another Swimlane, or enter an End Event. The point where the entity crosses the boundary to leave the swimlane depicts the place where the resource that was originally working on the entity becomes available to start working on another waiting entity. These points are modeled using Swimlane Entry and Swimlane Exit components.

The choice to have Swimlane Entry and Swimlane Exit components was made in a design workshop with involved consultants. Although some business process simulation tools (e.g., Rockwell Arena) use process components (similar to our Task component) which include the possibility to claim per process a resource and release him after the

process is finished, it made more sense to the consultants that entities which enter a Swimlane should claim a resource from that point on. During a series of activities namely, the resource remains associated with the entity, until he hands it over to another resource, or the entity leaves the swimlane and leaves the organization (through an End Event).

To enable these characteristics (of a Swimlane Entry queue; claiming a resource; and releasing a resource) in a simulation model, choices had to be made how to organize the assignment of resources to entities that are waiting to be processed. A possibility is to chose for a central approach, whereby one component monitors all states of the queue in a complete simulation model. Another option would be to chose for a de-central approach, whereby all Swimlane Entry components know which resource is available so – if a resource is available – he can be claimed. However, because of the different options of how resources in real business processes select an entity waiting to be processed (see Section 5.2), the the choice was made to let each Swimlane component have its own Resource Manager component.

Swimlane Entry simulation component

The Swimlane Entry component depicts the place where an entity enters a Swimlane and waits until a resource is available to 'pick up' the entity and perform specified activities. A Swimlane Entry is is not directly modeled in a conceptual model as a separated modeling element, but exists at every point where a Sequence Flow intersects with the boundary of and enters a Swimlane. The functionality of Swimlane Entry is shown in the state diagram in Figure 5.17.

The Swimlane Entry contains five states, to know: 'Passive', 'Add Entity to queue', 'Remove entity from queue', 'Forward entity' and 'Forward signal', and has two input ports: `in` (for receiving entities) and `inSignal` (for receiving signals). Signals are used for communication between Swimlane Entries and the Resource Manager component. Due to the specification of the DEVS formalism, information can only be exchanged between models through couplings. Because we decide to use the Resource Manager, as the active component to organize the assignment of resources to entities, the Resource Manager should be aware of the current state of all Swimlane Entries (i.e., how many entities are currently waiting in each Entry queue).

When an entity enters a Swimlane Entry and is placed in a queue, a signal is send to `outSignal` output-port. This `outSignal` port is coupled with the `in` port of the Resource Manager component. The signal contains an identifier of the Swimlane Entry (so the Resource Manager knows the origin of the signal), as well as the current state of the queue. In case a Resource is available, the Resource Manager will send back a signal containing the Swimlane Entry destination identifier, as well as the identifier depicting exactly which resource is available. This signal arrives through the coupling between the Resource Manager component and the Swimlane Entry `inSignal` port, a check is performed whether the destination of the signal corresponds with the identifier of that specific Swimlane Entry. In case the destination of the signal corresponds with the Swimlane Entry, an entity is removed from the Swimlane Entry queue and is send to the out output-port.

In case the destination of the signal does not correspond with the Swimlane Entry,

the signal is forwarded to the outForwardSignal output-port. The outForwardSignal port is connected with the next Swimlane Entry inSignal input-port. When there are more than one Swimlane Entry components, these components are mutually coupled through the outForwardSignal output-port and the inSignal input-port, creating a chain of Swimlane Entries through which the Signal can move and finally arrive at its destination.

It may occur that an entity arrives at the in input-port which does not need to wait for an available resource. This is the case for duplicate entities (which are created by the Parallel Gateway Split component). These entities leave a Swimlane and possibly re-enter it later on in the process to be synchronized with the original entity. However, the resource working on the original entity is still occupied with the original entity. In case the flow enters the swimlane and connects directly to a Parallel Gateway Join, the duplicate entity should be forwarded directly to the out port.

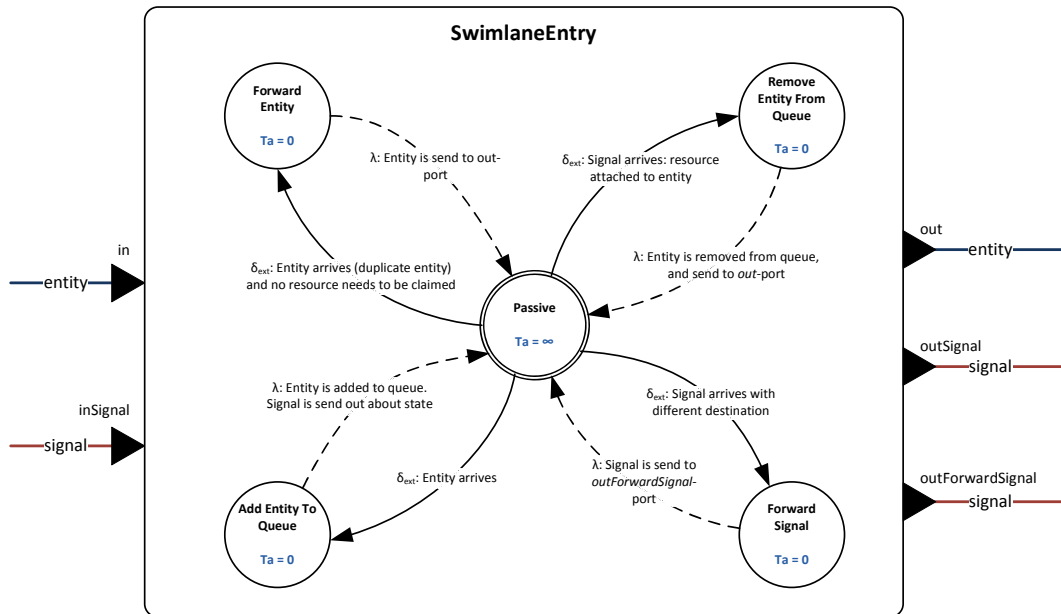


Figure 5.17: State diagram Swimlane Entry

Resource Manager simulation component

The Resource Manager is a component which cannot be modeled visually by the modeler, but is part of every Swimlane. The Resource Manager is not part of the entity flow and can only receive signals (through the in input port) and send signals (through the out output port) (as shown in Figure 5.18). After a signal is send by a Swimlane Exit because a resource came available, the Resource Manager releases the resource formally. This is done by updating a state variable which contains all information about all resources in the associated swimlane and their states (whether a resource is currently available or busy). After the variable is updated, or in case the signal's origin was a Swimlane Entry

component, the state of the component changes from 'Passive' to 'Check resources and queue'.

This state consists generally of two main steps. First, the resource state variable is checked to see whether any resource is available to pickup an entity (i.e., whether a resource is looking for work to do). If this is not the case, the state changes back to 'Passive'. In case a resource is available, the second step takes place. Another state variable is now checked to see whether any entities are waiting in any Swimlane Entry queues. In case an entity is waiting in a certain queue, the state of the resource that was found available during the first step is set to busy and a signal is sent to the output-port. This signal contains the identifier of the Swimlane Entry in which an entity is waiting, as well the resource that is attached by the Resource Manager to the waiting entity.

As was specified in Section 5.2, various ways exist how an available resource will select a queue. Due to this, Resource Manager is specified with a 'Queue Lane Mode' parameter, depicting how a resource will select a queue from several possible queues with entities waiting. This can be done based on a cyclical pattern, whereby first a waiting entity is taken from Swimlane Entry 1; then Swimlane Entry 2, ..., until the last Swimlane Entry is reached. Then the pattern starts over again at Swimlane Entry 1 and the cycle is repeated. If in a Swimlane Entry no entities are waiting, then the next Swimlane Entry is checked. The other ways as discussed in Section 5.2 under the heading "Resources and entity queues" can also be selected.

The Resource Manager is specified through the following parameters:

- **Number of resources** Number of resources capable of performing the activities within a swimlane
- **Queue Lane Mode** Specifies how an available resource selects a queue, for example Cyclical; Cyclical repeat; Longest average waiting time; Most entities waiting
- **Pattern value** Specifies – in case of Cyclical repeat mode – the number of times entities are taken from one queue, before moving on to the next queue in the cycle.

5.3.5 Conceptual model to simulation model transformation

In Section 5.3.3 we spoke briefly about Model-Driven Development and Model-to-Model (M2M) transformation. Now that we have discussed in more detail the designed DEVS components, we will suggest in this section how a conceptual model using the business process modeling elements can be transformed into a DEVS simulation model. First, each modeling element is matched with a DEVS simulation component. The Start Event, the End Event, the Task, the Parallel Gateway Split, the Parallel Gateway Join and the Exclusive Gateway are all matched with their atomic DEVS model counterparts. The Swimlane element and the Sub Process element are matched with respectively a Swimlane coupled model and Sub Process coupled model. The Resource Manager, the Swimlane Entry and the Swimlane Exit are not visually represented, but should be added manually or automatically for each Swimlane coupled model.

In Appendix F we included an elaborate set of rules that should be commenced in order to translate a conceptual business process model to a DEVS simulation model. We

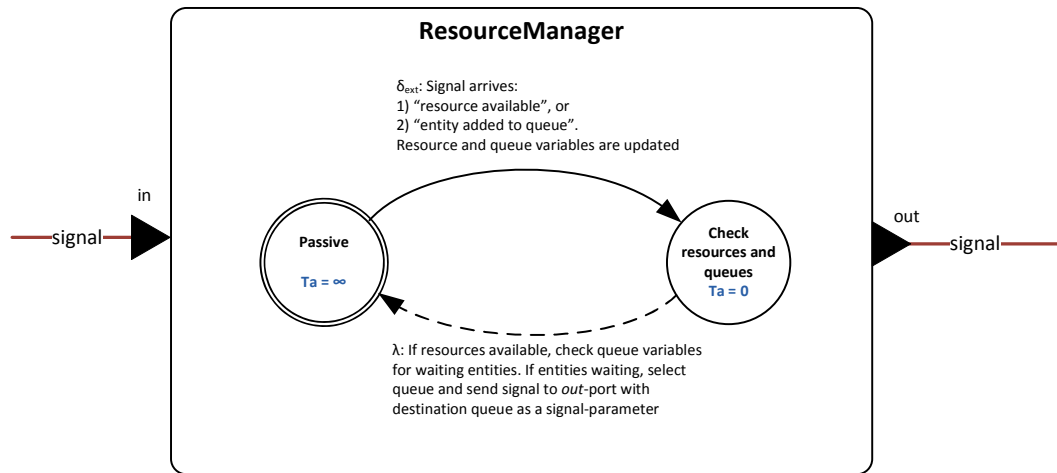


Figure 5.18: State diagram Resource Manager

will here only discuss briefly the main steps that should be undertaken to transform the conceptual model into a simulation model.

The main steps which should be undertaken, are:

1. Create for the conceptual model a corresponding coupled DEVS model (representing the Simulation model, all other components are placed within this coupled DEVS model).
2. Create for each Swimlane a coupled DEVS model and create a Resource Manager component in each created coupled DEVS model.
3. Create for each business process modeling element (all flow objects, except the Sub Process) a corresponding atomic DEVS model. Consider the placement of the component: If the flow object is placed in the root of the conceptual model, it should be placed in the highest-level coupled DEVS model. If the flow object is placed within a Swimlane, place the corresponding component in the coupled DEVS model corresponding with the Swimlane. Create for each Sub Process element a coupled DEVS model.
4. Create for each Sequence Flow entering a Swimlane a corresponding Swimlane Entry atomic DEVS model inside the Swimlane.
5. Create for each Sequence Flow leaving a Swimlane a corresponding Swimlane Exit atomic DEVS model inside the Swimlane.
6. Create couplings between all atomic and coupled DEVS models, corresponding to the Sequence Flows as drawn in the conceptual model. These can be Internal Couplings, External Input Couplings and External Output Couplings (see Figure 5.19). Make sure the couplings between the Exclusive and Parallel Gateways are connected to the appropriate Output Ports.

7. Couple all Swimlane Entry atomic DEVS model and Swimlane Exit atomic DEVS model with the Resource Manager atomic DEVS model.
8. Couple the Resource Manager atomic DEVS model with the first Swimlane Entry atomic DEVS model.
9. Couple the first Swimlane Entry atomic DEVS model with the following Swimlane Entry atomic DEVS model, until all are connected through a unidirectional link (Internal Coupling from Forward Signal Output Port to Signal Input Port of following Swimlane Entry atomic DEVS model).

In Figure 5.19 we show the various possible cases how Sequence Flows connecting two flow objects are translated into internal couplings (IC), external input couplings (EIC), or external output couplings (EOC), depending on where the source and / or target flow object is placed: a white square on the left-hand side of the figure depicts a Swimlane modeling element, a white square on the right-hand side of the figure depicts a coupled DEVS model.

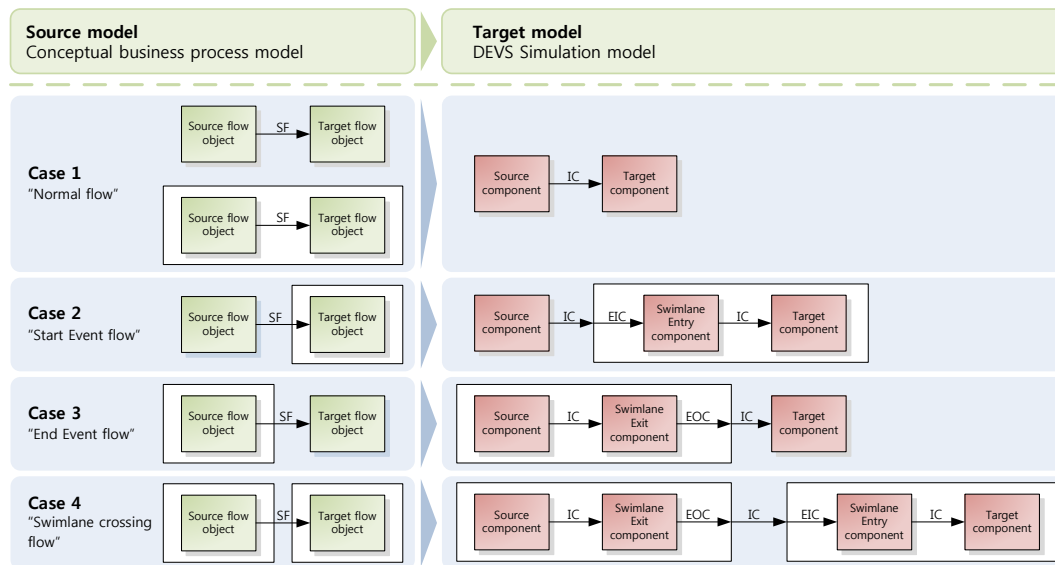


Figure 5.19: Transformation from Sequence Flows to Couplings

In the remainder of this section we will provide several examples how DEVS simulation models can be constructed based on a conceptual models.

One task example

The first example is depicted in Figure 5.20 and Figure 5.21. This represents a simple business process model, where one type of entities arrives (Start Event), a group of resources is available (Swimlane) to perform one task (Task), and finally the entity is disposed (End Event). In Figure 5.20 the conceptual model is shown of this business process. In Figure 5.21 the DEVS simulation model is shown. It includes all coupled DEVS models (one for representing the complete model, and one for Swimlane model), as

well as the atomic DEVS models representing the components and internal and external couplings.

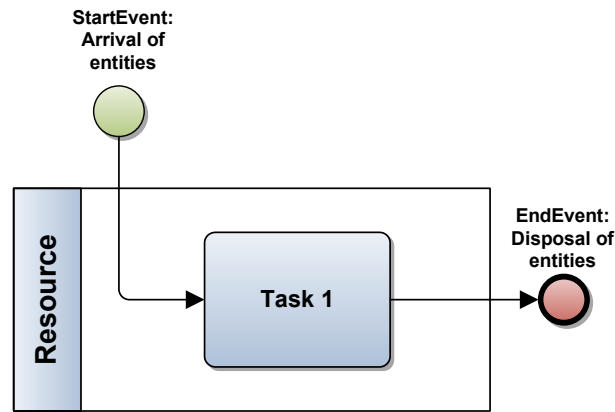


Figure 5.20: Sample Business Process Model: One Task

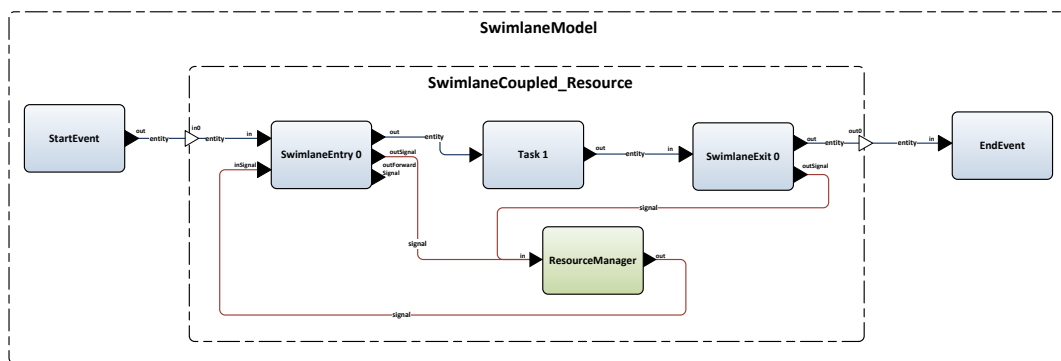


Figure 5.21: Sample DEVS model: One Task

Two Swimlanes and Exclusive Gateway example

The conceptual model shown in Figure 5.22 consists of two Swimlanes. These Swimlanes represent two groups of resources: Resource 1 and Resource2. Entities arrive in the first Swimlane and wait first until a resource is available. Then Task 1 is performed by the resource that 'picked up' the entity. After the task is finished, an Exclusive Gateway depicts a decision. Depending on the condition and the evaluation of this condition, the entity flows in one of the two directions. In one case the resource will perform Task 2 after which the entity leaves the system. In the second case, the entity flows out of the Swimlane (thus releasing the resource) and enters the second Swimlane. This depicts a resource handing over an entity to a resource with different capabilities. An available resource of the second type (Resource 2) is requested, after which the resource performs Task 3. Finally, after Task 3 is completed, the entity leaves the Swimlane, the resource is released, and the entity is disposed.

In Appendix G in Figure G.2 the DEVS model of this conceptual model is shown. Clearly visible are the 2 Swimlane Entries, for the flows entering both Swimlane coupled models, as well as the Resource Manager components.

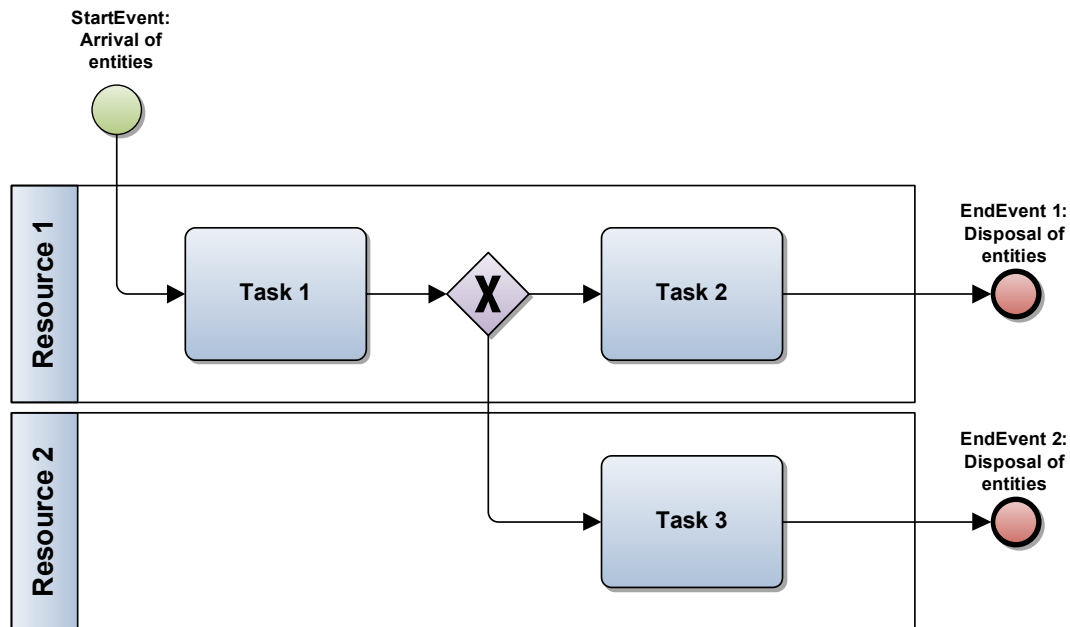


Figure 5.22: Sample Business Process Model: Two Swimlanes and Exclusive Gateway

Two Swimlanes and Parallel Activity example

The last example is a worked out example of the three cases discussed in Section 5.2 under the heading “Activities and decisions”. It includes the three ways how an activity can be performed. namely: 1) one resource starts and finishes an activity on its own; 2) a resource hands over the entity to another resource who will perform one or more activities; or 3) the amount of work is divided over two or more resources.

In this example (see Appendix G in Figure G.3), emails are received by an organization and each email contains three question. Depending on the difficulty of the email, a Front Desk Employee can answer all question by himself, or needs the help of a Back Office Employee in answering some of the questions, or all questions are answered by the Back Office Employee. Depending on the evaluation of the Exclusive Gateway conditions, the entity is duplicated in the Parallel Gateway Split. The Front Desk Employee performs the task “Answer some of the questions” and after this task is completed, the entity will be hold in the Parallel Gateway Join until the second part of the activity is also finished. The Front Desk Employee will in this case wait until the second Back Office Employee is finished with his part of the questions (if the Back Office Employee does not finish before the Front Office Employee). The Back Office Employee can be requested in two cases, namely when an email arrives that he needs to handle completely, or when he needs to answer part of the questions (the parallel activity). In the second case, after a resource is finished answering part of the questions, the duplicate entity will move back into the

Swimlane were it was originally created (surpassing the Swimlane Entry), and moves into the Parallel Gateway Join component. In case the original entity was already finished, the Front Desk Employee will combine the answers and call the customer with the final answers.

This example is useful to depict how the Exclusive and Parallel Gateways can be used to model part of a real business process (this example was modeled by consultant from Accenture). Also, by specifying the Task with the actual task description, the Business Process Model is clarified. In Appendix G in Figure G.4 the DEVS model is shown for this case.

5.4 Statistics and outcomes

Goal of business processes modeling is to get an understanding of the organization, but more importantly to enable evaluation of performance indicators. In Figure 5.1 this is depicted by the generic simulation life cycle step experimentation and results in "Solutions and understanding". Before we continue explaining how the proposed design in Section 5.3 can be actually implemented in a working tool (this will be discussed in Section 5.5), we need to look closer at what consultants are interested in knowing about business processes, and how this can be realized.

5.4.1 What to collect?

Three levels of performance can be distinguished within an organization: business performance, functional performance and process performance. In Figure 5.23 the levels of performance are made visible in a pyramid-shape. The top-layer, namely of the business performance, relates to the overall performance of a whole organization (as being part of a larger network or environment). The middle-layer, namely of the functional performance, relates to the performance of distinctive business units or business process within an organization. The bottom-layer, relates to the actual process performance, like for instance the actual activities and operational decisions. The performance of the processes contribute to the functional performance, and the functional performance on itself contributes to the highest level, the business performance.

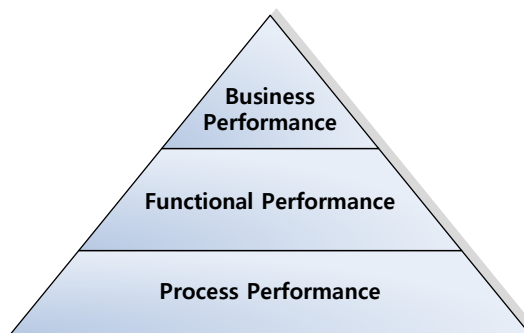


Figure 5.23: Organization Performance Pyramid

Evaluation of the performance within an organization can have one of two goals: 1) for descriptive evaluation of performance indicators; and 2) for prescriptive analysis of performance indicators.

Descriptive evaluation Descriptive analysis is performed to analyze whether the current performance of an organization meets the pre-specified performance goals. Performance goals are set for a period of time like for instance performance per month, per quarter, or per year. In case of monthly performance analysis, the period from the start of the month until the current day is called 'month-to-date' (MTD). Calculating for instance an MTD performance, enables to see whether an organization performs as planned. In other words, it compares the reality with the planned expectations. When if for instance the current performance continues for the remainder of the month, the performance goals will not be met, the business processes can be adjusted on time.

Prescriptive analysis Prescriptive analysis differs from descriptive evaluation as it is meant to forecast how an organization will perform when it continues to operate as it currently is performing, or when certain business process redesign decisions would be implemented.

Figure 5.23 is useful because it clarifies that descriptive and prescriptive performance analysis of the lowest level of business processes contributes to the analysis of higher level performance indicators of an organization. To be able to evaluate the performance on the lowest level, statistics should be collected of all elements that contribute to the performance (like for instance resources, activities and entities). The statistics generated and collected on the lowest level can be then be aggregated into key performance indicators (KPI's), or separate parts of a business process. Based on discussions with consultants and literature study (see for instance [Tum95] and [VD98]), statistics should be collected on the following business process element:

- Entities
- Resources
- Queues
- Activities

5.4.1.1 Statistics on Entities

A consultant is interested in knowing the following statistics about the different entity types that enter, leave and are handled by an organization:

- the total number of entities which arrive in a business process over a period of time,
- the total number of entities which leave a business process over a period of time,
- the total number of entities which are at certain moment within a business process,

- the total number of entities which require rework during a process over a period of time.

Besides number of entities that went in or out a business process, a consultant is also interested in knowing more about time related aspects are of entities:

- the total Time in System (TiS): how long was an entity within a business process (the time difference between the time of arrival and the time of departure)
- the value-added time: the time that an entity was actively being processed (namely when a resource performs tasks which changed the state of the entity)
- the total non-value-added time: the total waiting time of an entity (namely when an entity is waiting until it is picked up by a resource, or when it waits in some other queue).

Based on the individual entity statistics, also the minimum, maximum and average statistics can be collected *per entity type* over a period of time. This is of more interest for a consultant, as these are easier and more useful interpretable figures.

Calculation of statistics

To keep track of the number of entities that arrived and have left, the implementation of the Start Event and End Event component can be extended with a counter variable with an initial value of 0. Every time an entity is created by the Start Event, the 'entities-in'-counter is increased with 1. Every time an entity enters the End Event, the 'entities-out'-counter is increased with 1. At any moment in time, the value of the counters, depict the number of entities that are created and disposed (entered and left the business process). The difference between these counters depict the total number of entities that are currently within the system (both waiting in queues, as well as currently being processed). To collect statistics about the number of entities which required rework, the entity object can be extended with a counter variable which is increased with 1, every time rework is performed. The End Event component can collect these statistics by being extended with a separate counter variable.

To collect statistics related to time (TiS, value-added time, non-value added time), the entity object can be extended with different variables which specify the creation time, the disposal time, the time being processed, and the time waited. The average total time in system of entities can be calculated through Equation 5.1, where $t_{disposal,i}$ is the moment that entity i was disposed, $t_{arrival,i}$ is the moment that entity i was created, and n is the total number of entities that entered the End Event.

$$AverageTotalTimeInSystem = \frac{\sum_{i=1}^n (t_{disposal,i} - t_{arrival,i})}{n} \quad (5.1)$$

The average total time of entities leaving the business process, can be calculated Equation 5.2, where S_i is the total time of entity i being processed.

$$AverageTotalProcessingTime = \frac{\sum_{i=1}^n S_i}{n} \quad (5.2)$$

The average total waiting time can be calculated through Equation 5.3, whereby W_i is the total waiting time of entity i in a business process.

$$AverageTotalWaitingTime = \frac{\sum_{i=1}^n W_i}{n} \quad (5.3)$$

5.4.1.2 Statistics on Resources

Resources are graphically represented in swimlanes depicting a group of resources with similar capabilities. It is of interest to a consultant knowing what the utilization is per group of similar resources. The utilization of a resource is the total amount of time during which a resource is active performing activities, divided by the total time the resource was available. When for instance one or a group of resources have a utilization of 1.0, this demonstrates that the resource(s) were continuously active. This might occur in two cases: the number of resources that was scheduled was exactly right to perform all the work, or – and which is more likely – the number of resources was inadequate to handle all the work. By also providing statistics about queues within a business process, a consultant can analyze whether the second case is valid: if on average a number of entities is always waiting to be processed by a group of resources with a utilization close to 1.0, this means that the number of resources is inadequate to handle all the arriving work.

In Equation 5.4 the formula is shown to calculate per resource the utilization. The 'Total Time Busy' can be calculated by summing all the durations from when a resource became busy working on an entity, until the moment it was released. In case there are more resource with the same capabilities and one is interested in knowing the total utilization for this resource group, it can be calculated by summing the individual utilization parameters, and dividing it by the total number of resources.

$$ResourceUtilization = \frac{TotalTimeBusy}{SimulatorTime} \quad (5.4)$$

5.4.1.3 Statistics on Queues

As mentioned in the previous paragraph, a consultant is interested in knowing all places where entities wait (queues) before entities are handled by a resource. More specifically, he / she is interested in knowing:

- the number of entities waiting at a certain moment within a queue,
- the average number of entities which waited within a queue over a period of time,
- the smallest number of entities which waited within a queue over a period of time,
- the largest number of entities which waited within a queue over a period of time.

Furthermore, waiting times of entities within a queue are also of interest to a consultants. The waiting time depicts the time difference between the moment that an entity entered a queue (i.e., started to wait) and the moment an left the queue or the current moment. Statistics of interest are:

- the average waiting time of all entities which are at a certain moment waiting within a queue,
- the total average waiting time of all entities that entered and left a queue,
- the longest waiting time of an entity compared with all entities that entered and left a queue,
- the shortest waiting time of an entity compared with all entities that entered and left a queue,
- the waiting time of the longest waiting entity in a queue at a certain moment (compared to the other waiting entities at that moment),
- the waiting time of the shortest waiting entity in a queue at a certain moment (compared to the other waiting entities at that moment).

Calculation of statistics

To calculate statistics on the queues, the queue-object should be extended with several variables, containing for instance the total number of entities that have entered and left the queue. In [LK00] multiple equations are provided (based on a queuing system), to calculate the steady-state average delay (see Equation 5.5) and the steady-state time-average number in queue (See Equation 5.6). The statistics of interest mentioned above can be derived by adapting Equations 5.5 and 5.6.

$$d = \lim_{n \rightarrow \infty} \frac{\sum_{i=1}^n D_i}{n} \quad (5.5)$$

$$Q = \lim_{T \rightarrow \infty} \frac{\int_0^T Q(t) dt}{T} \quad (5.6)$$

5.4.1.4 Statistics on Gateways and Activities

To evaluate what effects possible redesign efforts may cause, it is useful to collect statistics on Gateways. When a process is redesigned, it may namely be of interest to a consultant to see how the flow of entities will change and how this influences the business process overall. This provides the option to experiment with alternative design choices.

Consider an example in which a consultant wants to evaluate the KPI: “percentage of entities status *good*”. This KPI can be subdivided into the following sub-KPI’s:

1. sub-KPI 1: percentage of entities directly good
2. sub-KPI 2: percentage of entities good within predefined norm time
3. sub-KPI 3: percentage of entities not good

Within a business process model, the different flows of entities regarding these sub-KPI’s may be directed by Exclusive Gateways. Collecting statistics on the Gateways

enables a consultant to evaluate immediately the number of entities that were directly good, good after some time, or not good. The number of entities that leave through each Exclusive Gateway outgoing flow is thus of interest.

For activities, the average duration (as well as minimum and maximum duration) is of interest to the consultant. Activity statistics may provide various analysis options. By grouping the statistics of various tasks together, an aggregation can be made how much time a part of process costs. Also, when comparing average duration of one task with the total average processing time of an entity type, the consultants can analyze what percentage of the processing time is created due to that activity. This can create insight in what the most time consuming or crucial activities are within a business process.

5.4.2 How to present?

An important aspect besides the collection of statistics is how the statistics can be presented to consultants. In Section 5.4.1 an overview was provided of a minimum set of statistics that a consultant is interested in knowing. When the modeled business process is relatively large (for example 10+ Swimlanes, 50+ queues, 20+ Start and End Events, ... etc), this will result in a large array of statistics (of which few of the statistics are probably only of interest to the consultant). It is important that the consultant is able to select the statistic that he / she is interested in knowing. The way the statistics are named and grouped should thus be defined in correspondence with how consultants would see this as understandable and easy to use².

Another question, besides how to present the overview of all collected statistics, is how to present the actual values per statistic. It depends on the type of statistics and parameters of interest how this can be best done. Some widely applied methods to display data are: Pie-charts, Box-plots, Histograms, Bars, Gantt-charts, Scatter-plots, Time-series, Dials, Meters, X-Y-plots, and several more [Kul96]. It is up to the person experimenting with a simulation model to decide how it is most useful to have the output presented. For many statistics, like for instance the utilization of resources, a graph with on the horizontal axes the simulation time, and on the vertical axes the utilization value (with a range from 0.0 to 1.0), would create insight in the course of the business process, regarding the resources. Due to the possible variability in statistics, it is useful to present data in such a way that the variance is also shown. This could be achieved by presenting data in a box-and-whisker plot. It creates a better understanding for the consultant to see the distribution of all values and how the outcomes relate to the calculated average value.

In general there are two ways when output can be presented, namely: 1) during the execution of a simulation model, and 2) when the simulation run is finished. Regarding the first point, a consultant may be interested in actively analyzing a simulation model, by having the possibility to continuously see how parameters change during the execution of a model. Because of certain output-values he may also want to view different parameters. Thus, it would be useful if the solution provides an interface that during the execution of a simulation model can be modified and expanded with different output-statistics. In

²Due to the limited amount of time available for this research and time available with management consultants from Accenture, we did not elaborate on how the naming and grouping can be designed best.

Figure 5.24 a screenshot is shown of the statistics-window provided by the simulation software DSOL. It allows one to display output-statistics during the execution of a simulation model and supports different ways of how the data can be presented. On the left-hand side an overview is given of all collected statistics, and these can be moved into the right-hand side space. The Arena simulation software allows to add different kind of statistics presentation methods to the simulation model worksheet. In Figure 5.25 a screenshot is shown where both counter-objects are placed (showing only a numerical value) as well as time-dependent plots next to a drawn simulation model. It is however not possible in Arena to add new outputs to the worksheet during the execution of a simulation model.

An aspect related to the visual output of a simulation model execution and an analyzing consultant, is the visual animation of an model execution. Besides having statistics presented during or after the execution of a model, it is considered useful to have for instance the flow of entities during the execution also made visible, or to make the status of a certain model entity clear (for instance whether a resource is busy or idle). In [Rob04] a list is presented which discusses the advantages of having a visual interactive display and displaying the state of a simulation model dynamically. The advantages are:

- Greater understanding of the model (users can track events as they occur).
- Easier model verification and validation (finding modeling errors and non-simulation experts can comment on the validity of a model).
- Enables interactive experimenting (new ideas can be implemented).
- Improved understanding of the results (results can be related to specific events which occur during the execution of the model).
- Improved communication of the model and its findings to all parties (non-simulation experts are more easily able to understand the model).
- Provides the potential of using simulation in group problem solving (validation and experimentation can also be carried out in a group setting, thus getting more input from different involved stakeholders).

During the design workshops with consultants, as well as discussions with experts, it became clear that a solution should provide visual feedback from a model execution. One aspect frequently mentioned was that the flow of entities should be made visible in a model. The term “marble-track” was often mentioned, as a way how the flow of entities can be represented. An entity is for instance generated by the Start Event and rolls through the business process model, until it reaches an End Event. Preferably, it should also be possible to change the appearance of an entity regarding its type and state.

Regarding the second point, statistics report may be generated automatically after a simulation model execution. Many simulation software packages allow to present the performance statistics after a simulation run in such reports, and sometimes also allow a modeler to customize what it included in it [Kul96]. However, after discussions with consultants, more emphasis is put on the generation and visual customization of statistics *during* the execution of a simulation model.

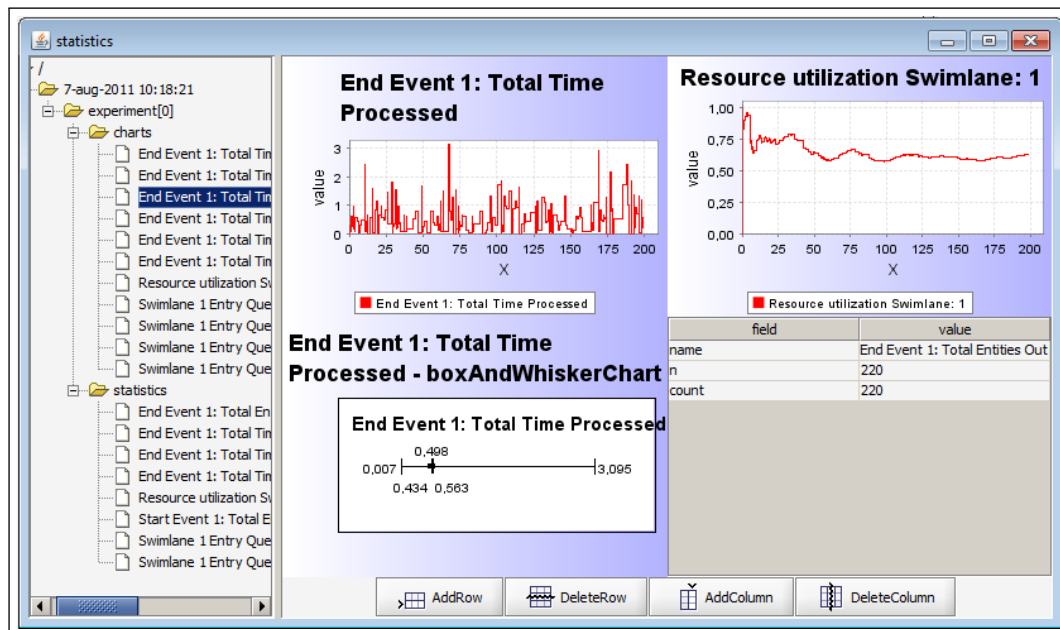


Figure 5.24: Screenshot DSOL statistics output

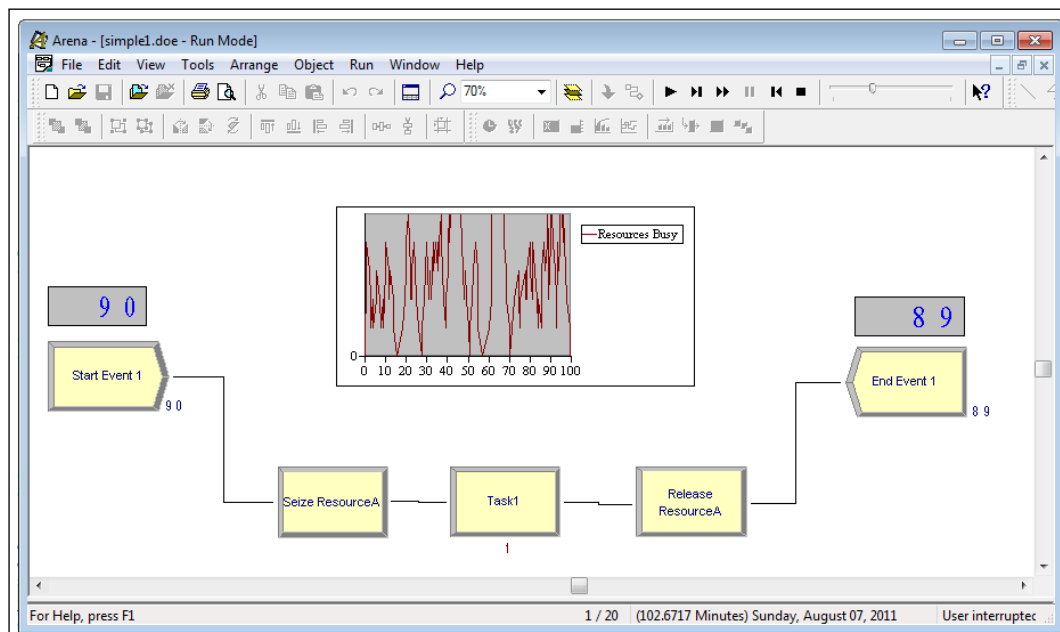


Figure 5.25: Screenshot Arena statistics output

5.5 Interaction and Process

Two aspects which are not covered yet in this chapter relate to the interaction with a solution and the visual appearance of it, as well as how the solution can be used and

integrated in the actual processes of consultants. In this section we will first elaborate on how a consultant can interact with a tool in order to develop conceptual models (Section 5.5.1). Next, in Section 5.5.2 we will elaborate on how the solution can be used in a client-setting.

5.5.1 Interaction with a tool and visual appearance

The interaction between a consultant and a solution is an important aspect which contributes greatly to the usability of a solution. A consultant main interactions with a tool are to develop or modify a business process model, to specify components, to execute the simulation engine, and to analyze the outcomes. Extended research on how the interaction could best be designed, is considered outside the scope of this research. For more information on Interaction Design, see [Coo07][Saf06].

In order to support the consultant with modeling, we will look at currently available modeling tools. Microsoft Visio is for instance a program which can be used to develop all sorts of diagrams. Microsoft Visio also supports the development of BPMN models³. Drawing a conceptual BPMN model with Microsoft Visio is supported by a template which consists of the various modeling elements as specified by the BPMN Specification [Obj10]. These modeling elements can be selected and placed on a virtual worksheet. By selecting the Sequence Flow connector the elements can be linked to each other. In Figure 5.26 a screenshot of the interface of Microsoft Visio 2007 is shown. Objects can be selected, moved around and deleted, in order to modify the layout or flow of the model.

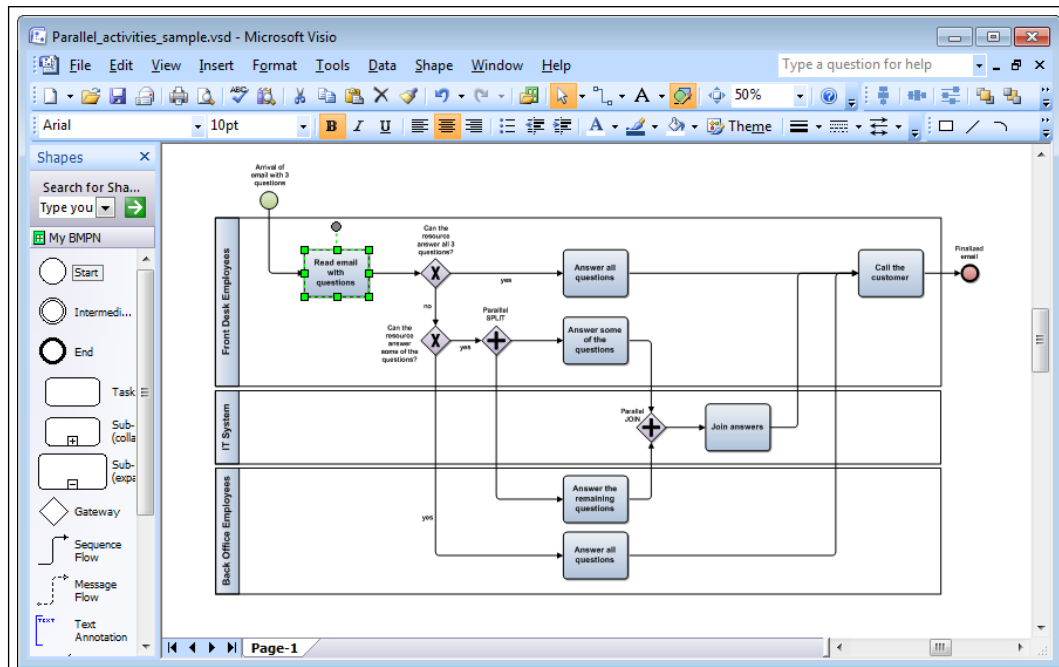


Figure 5.26: Screenshot of Microsoft Visio 2007 with BPMN template

³Microsoft Visio 2010 has a BPMN modeling template integrated. For older versions of Microsoft Visio there are various templates available.

In Figure 5.27 a screenshot of the interface of Arena is shown. Arena provides a visual modeling interface to enable simulation model development, as well as a simulation engine to execute a developed simulation model. Arena does not include a template of BPMN modeling elements, but developing a simulation model is done similarly to the development of BPMN models (or any other type of diagram) with Microsoft Visio: the development of an executable simulation model can be done solely through the visual interface of Arena, including the specification of components. Specification can be done in a direct manner (see the table in the right-bottom of Figure 5.27) as well as through popup windows (see Figure 5.28).

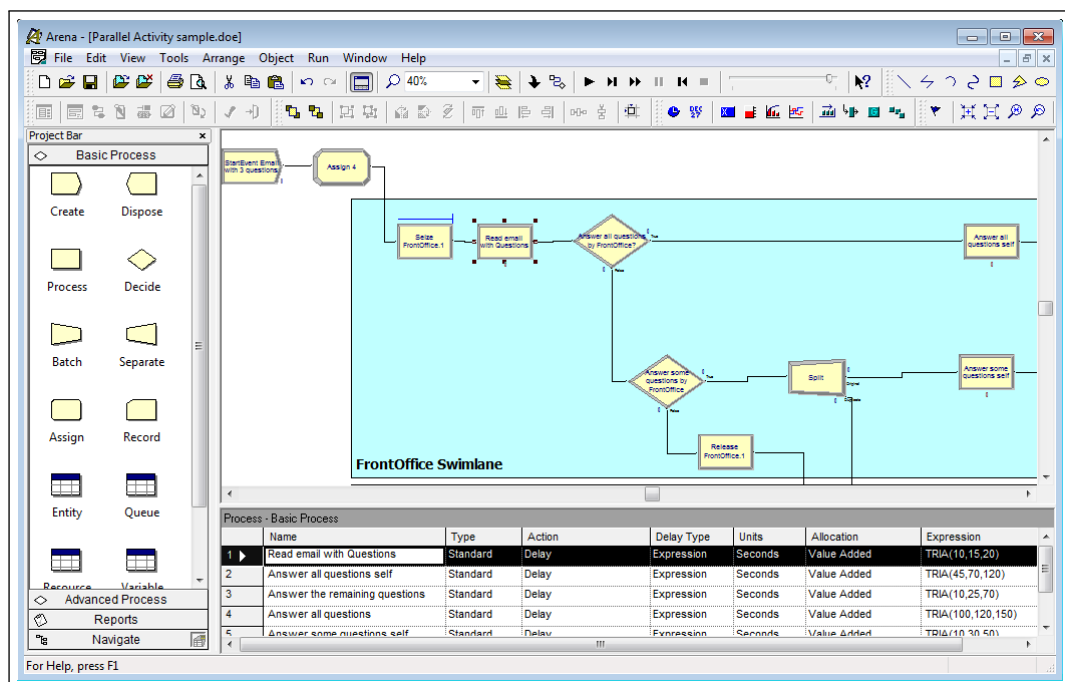


Figure 5.27: Screenshot of Arena 13

Although there are many visual modeling and simulation tools besides Microsoft Visio and Arena and they may differ significantly with regard to their appearance, the main concept how a user interacts with these tools to develop a visual model is quite similar in most. Microsoft Visio and Arena are mentioned because the first supports conceptual modeling of Business Processes, whereas the latter support visually developing simulation models.

Most management consultants have experience with working with tools like for instance Microsoft Visio. Although consultants may not be used to the modeling elements of BPMN, the interaction remains the same. Because of this reason, we suggest that a solution should allow a consultant to model business processes in a manner similar to Microsoft Visio. For the specification of modeling elements we suggest that an approach similar to the one provided by Arena would be most appropriate for consultants.

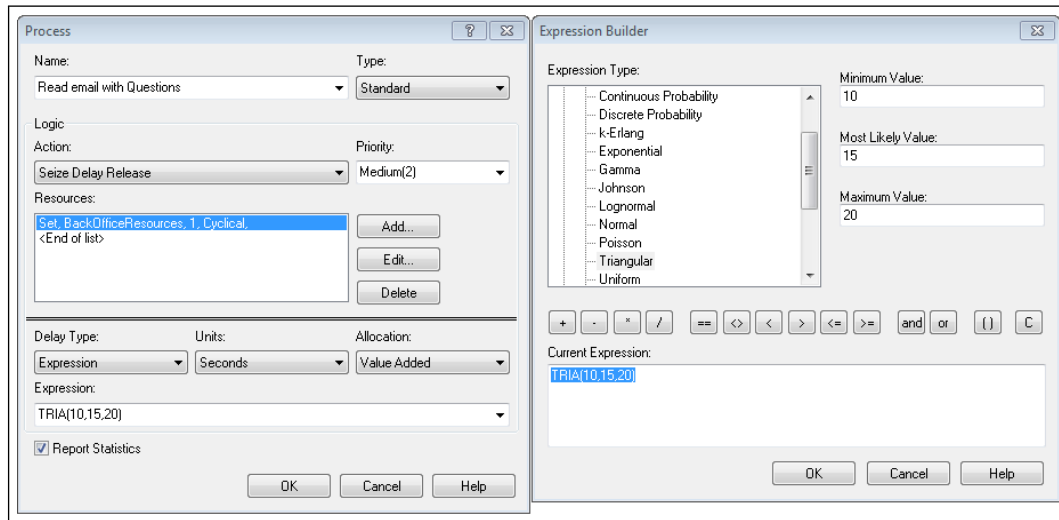


Figure 5.28: Screenshot of Arena 13 specification windows

5.5.2 Implementation in consultants' process

The last aspect to be covered relates to the integration of a modeling and simulation solution in the actual processes of consultants. Although every client's project is different, we can make some more general suggestions how a solution can be used by consultants. First, let us refer to Chapter 2 and more specifically Figure 2.4. This figure depicts the current development process of decision support tools based on simulation models. We found that there is a clear separation of activities undertaken by consultants in order to finally deliver a simulation based planning tool. Three main development roles can be identified in the process, namely the *process modeling* role; the *data collecting* role and finally the software specialist or *simulation model programming* role. Several issues are also identified with regard to this separation of activities, as well as the overall development process (see Section 2.3).

We suggest in this research that the conceptual model development and simulation model development can be more interrelated, namely by using the business process elements, the proposed DEVS-based simulation components and the stated way of transforming a conceptual model into a simulation model. This covers to some extent the issues regarding the simulation model development. However, we can also elaborate more on the issues regarding the process of gathering information about a client's firm and interactions with client-side specialists and business managers. Two issues that are mentioned in Section 2.3 are that there is *no common and agreed understanding of business processes at client-side* and *data specialist at client-side understand processes differently than business managers*. In the literature study we also came across similar issues [Ma01].

An approach which seems fruitful to support issues related to process mapping and model development, is collaborative modeling [RKD08]. Collaborative modeling is defined in [RKD08] as "*the joint creation of a shared graphical representation of a system*". What is emphasized in this definition is the *joint* creation of developing models,

Collaborative modeling is the joint creation of a shared graphical representation of a system

namely by actively involving stakeholders as well as external professionals during the whole modeling process. In the current model development process (as described by Accenture consultants) the modeler develops the model based on the descriptions provided by client-side stakeholders. However, when a more collaborative modeling approach would be used, the role of the modeler will also be to support groups and interfere with the content to make the involved stakeholders understand the system and the processes [RKD08].

Two examples can be given how models can be developed using the proposed solution. The first example relates more to the current way of model development. A consultant sits down with a client-side stakeholder and asks him questions about the business processes. Instead of first making a list of all processes, the consultant is now able to immediately start drawing business process model on his laptop or even on a tablet-computer. Together with the client-side specialist he can elaborate on it and possibly specify some of the processes. Because conceptual modeling and simulation are integrated in the solution, the consultant can also – if enough data is available to specify the model – execute the model and demonstrate the dynamics and behavior. Direct feedback can then be provided by the client-side stakeholder. If more data or process descriptions are needed, the consultant can take this model and meet with other stakeholders. By also integrating a revision control system, changes suggested by different stakeholders can also easily be tracked.

A second example of how a solution can be used, relates more to collaborative modeling. Instead of meeting with the client-side stakeholders individually, the consultant organizes a workshop with a group of stakeholders. These sessions would include different stakeholders, ranging from business unit managers to data specialists. This allows for an exchange of perspectives and assumptions among group members. The consultant projects the modeling environment on a projection sheet and asks questions to get more insight in the processes. Because more stakeholders are involved, a discussion may follow which enables the different stakeholders to discuss their views on the business processes. This will hopefully lead to a more agreed understanding of their own processes, but also an agreed representation of it. If the solution allows also visualization of the flows of entities and output of statistics, it enables the consultant to provide feedback of the model execution to all group-members and more discussion can take place.

To support the mentioned limitations of the current approach, namely regarding the separation of activities and consultant roles, it would be an option to integrate these roles more. By involving for instance both consultants with the knowledge on data, as well as consultants with more knowledge on business process modeling, (as well as the related client-side stakeholders), these limitations may more easily be overcome. To what extent however this will be fruitful, or how groups should be composed, or which roles the consultants should take and how they should interact with the group, is outside the scope of this research. Follow-up research on these topics is however strongly suggested.

5.6 Summary and conclusions

In this chapter a new modeling approach is proposed for consultants to model and simulate business processes. Following a user-centered design approach, we established a set of

modeling elements based on the BPMN specification. These modeling elements were designed interactively with consultants to make them connect as much as possible with how consultants see business processes. In order for a consultant to develop business process model, we define the following modeling elements: a Start Event; an End Event; a Task; a Sub Process; an Exclusive Gateway; two Parallel Gateway (to split and join the flow of an entity) and a Swimlane-element.

To support the transformation of a conceptual model which uses the proposed modeling elements, into an executable simulation model, we first mapped the modeling elements with simulation components. These components are specified following the DEVS formalism. Some additional components were needed to represent certain characteristics of business processes in a simulation model. For instance because a new concept was introduced to allow an arriving entity to request and release a resource. Swimlanes represent resources in a business process model. When an entity enters a Swimlane, a resource is requested by the Swimlane Entry component. The resource is released when an entity leaves a Swimlane, namely when entering the Swimlane Exit component. We finally elaborated on how the conceptual model can actually be translated into a simulation model, by specifying transformation rules. When implemented, a transformation can then be executed manually or possibly automatically.

For the generation of statistics, a generic set of parameters is defined which should be collected in an implementation of the solution. These relate for instance to the utilization of resources, the average waiting times of entities in a system, and the average number of entities in queues.

This chapter concluded with a brief discussion on how a solution can be used by consultants and how it can be integrated in actual processes. Regarding development of the business process models, we suggest a drag-and-drop approach, similar to one provided by for instance Microsoft Visio. For the actual usage of the tool in order to develop models together with client-side stakeholders, we suggest that follow-up research directs its attention to collaborative modeling as a possible fruitful approach. Many issues were identified (both in literature, as well as by Accenture consultants), than can possibly be minimized by choosing for such an approach. When however in an implementation the transformation of the conceptual model into a simulation model can be provided, we believe that this will have already significant positive effects on the development process. To find out whether the proposed solution will actually work, we will discuss in the following chapter a prototype which is developed as a proof of concept.

Proof of Concept

Christina is again together with the student. This time for an evaluation workshop. Apparently, he finalized a prototype of the solution and wanted her to evaluate it. In the email he sent her some days ago, he asked her if she could bring along one of the business process models she developed of the last project. They would try to use the prototype and rebuild the model using the designed elements, and see if the tool would allow them to simulate the processes, without having to program anything. Christina sits down behind the notebook, and she looks at the screen. She sees a program, which is basically a white sheet with on the right side a panel showing the modeling elements they defined during the design workshops. She sees a swimlane element, a task element, an exclusive gateway (for modeling decisions), and several other elements. The student asks her to start drawing the model based on the model she brought with her. Both Christina and the student are curious to see what will happen, as the model is quite complicated.

Quite quickly, she becomes accustomed how to use the tool and draws all the processes. Some things are a bit different than in her original model, but all things specified during the workshops are included. Next, she needs to specify the processes. For instance, how long a resource needs to perform a certain task. It takes her some time, but finally, the complete model is specified. The student clicks on a few buttons on the screen. And there, within a few seconds, a small screen is shown in front of her. With a 'play'-button. After she presses the button, Christina looks amazed: she sees a graph constantly being updated with the utilization of the resources she just specified. And she sees a graph which tells her the processing times of entities that arrived and were processed. And how long they had to wait.

"I don't have to write any lines of code myself?", Christina asks. "No, I did all the programming for you. However, it is still a prototype and it should be developed further," the student answers. "Sure," Christina answers, but without hearing the last sentence. She is still amazed. "I've just worked with a tool," she thinks, "that allowed me to easily model business processes, using elements that are based on how I see the processes, and an executable simulation model was almost automatically generated!"

In this chapter we will evaluate whether the designed solution is indeed a solution for the end-users, namely the management consultants. As a proof of concept, this chapter will elaborate on the development of a prototype based on the characteristics as defined in Chapter 5 and evaluate several related aspects. Firstly, we are interested in knowing whether the earlier defined simulation components can be implemented into software code and can then be coupled to form a simulation model. To improve the feasibility of developing a working prototype in the limited amount of time available, we will discuss in Section 6.1 the Model-Driven Development framework for Modeling and Simulation (MDD4MS) and the available prototype.

In Section 6.2 we will discuss the implementation of the simulation components and the development of the prototype. First, DEVSDSOL will be briefly introduced, which is followed by an elaboration on the implemented components. Next, the defined transformation rules will be discussed, which allow for (semi)-automatic transformation of a conceptual model into an executable simulation model. Section 6.3 will provide an discussion on the verification and validation of the prototype. Several business process models are developed with the prototype and the resulting simulation models are verified. Next, in Section 6.4, we will evaluate the prototype with a real business process case of a Dutch telecom operator. To evaluate also the usability of the solution and acquire feedback from the end-user, we will discuss in Section 6.5 the evaluation sessions held with management consultants from Accenture. Lastly, in Section 6.6, we will evaluate to what extent our solution and prototype meet the design requirements stated in Chapter 4.

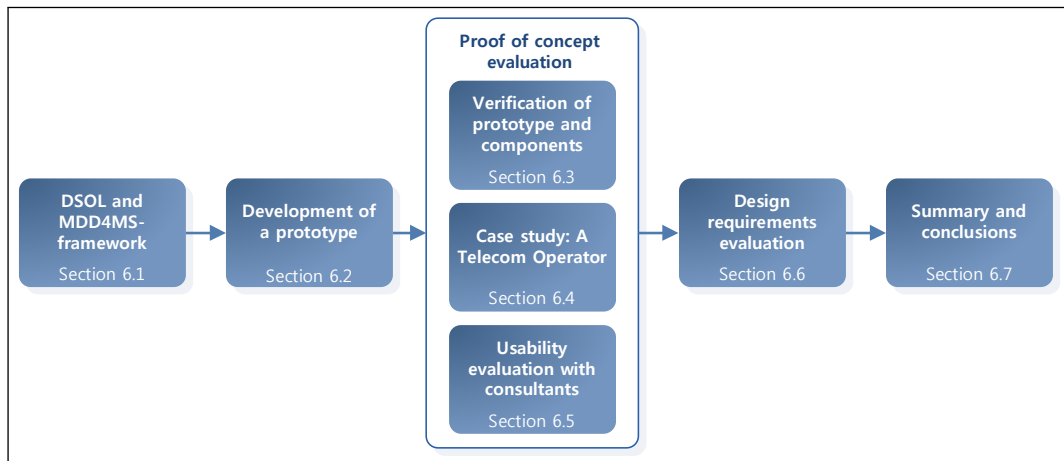


Figure 6.1: Outline Chapter 6

6.1 DSOL and MDD4MS-framework

Although the concept design as proposed in the previous chapter is based on input and feedback provided by management consultants (through the UCD approach), this does not prove whether the solution is actually usable and whether it can and will work. To

evaluate this, we suggested to develop a prototype which incorporates to some extent¹ the designed and required features and functions. There are two options how a prototype (but also a final support tool) can provide the required functionality, namely by building a software tool from scratch, or by adapting existing and validated software (or parts of existing programs and software code) and developing part of the functionality and features self. Developing a complete tool from scratch is costly, as well as time consuming. We suggest therefore that because there are several tools available which can provide parts of the required functionality and are validated, the second option is preferred for the prototype. Whether the second option is also preferred for a final tool, in other words, whether integrating existing software is adequate for a final tool, requires separate extensive research.

As mentioned in Section 5.5.1, a large number of tools exist that allow modeling of business processes based on BPMN. Also, various tools exist that support modeling and execution of simulation models. None of these tools however provide direct support for all of the preferred features and characteristics. Some of the business process modeling tools (like for instance Aris Express or Visual Paradigm) allow to simulate business processes. However, their simulation capabilities and functionality are often too limited and non-customizable. Regarding the existing dedicated simulation programs that support visual development of simulation models, these tools do not support BPMN directly. Arena for instance allows to develop components which contain some of the functionalities as specified. However, the concept of an entity entering a swimlane and thus requiring a resource can be modeled using distinctive components, but not visually by drawing the model and Arena interprets this automatically.

For a prototype to support most of the features as defined or designed in the previous chapter, we suggest to adopt Distributed Simulation Object Library (DSOL) in combination with the Eclipse platform. DSOL is an open source multi formalism simulation library developed at Delft University of Technology [Jac05]. DSOL is written in the Java programming language and has proven to be very effective for various simulation projects. DSOL also supports execution of simulation models based on the DEVS formalism through the DEVSDSOL library (which is compatible with hierarchical DEVS) [SV09]. Eclipse is a software development environment that can be used to build integrated development environments (IDEs). Various plug-ins are provided by Eclipse community to also support the development of graphical model editors.

A prototype should preferably also support the automatic transformation of a visually modeled business process model into an executable simulation model. This because firstly, the consultants wants to be able to model and experiment with a simulation model himself, without the help of a simulation programmer. And secondly, through automated transformation of a conceptual model into a simulation model (if this transformation is based on a strictly defined set of rules), the simulation model represents the conceptual model and the consultant knows that the behavior of the simulation model corresponds to how he modeled it conceptually.

At Delft University of Technology research is currently undertaken on a model-driven approach for simulation model development. The Model-Driven Development framework

¹Due to the limited available time, not all features could finally be implemented.

for Modeling and Simulation (MDD4MS) includes a modeling and simulation life cycle, metamodel definitions for conceptual modeling, specification and implementation stages, model transformations for the suggested metamodels and a tool architecture for the overall process [CVS11]. A working prototype implementation of the MDD4MS framework is also available. We consider it to be an opportunity to combine our research with the ongoing research by Cetinkaya, because from our side we are able to use the already developed MDD-based prototype to evaluate our modeling approach for consultants. We considered it namely to be impossible within the available time to develop a fully working prototype ourselves. Our research however can also – partly – contribute to Cetinkaya's research, as it may be considered as a case study.

The MDD4MS prototype supports model to model (M2M) transformation from *Simulation Conceptual Model* (CM) to *Platform Independent Simulation Model* (PISM), to finally *Platform Specific Simulation Model* (PSSM). The PISM is formally defined using a certain formalism (e.g., DEVS, Petri Nets, State Charts) and defines the system functionality. It does not however take into account on what platform the model will be implemented later on. The PSSM is defined by a higher level representation of a programming language and is an implementation of a PISM model [CVS11]. A PSSM can be transformed into an executable simulation model using a model to text (M2T) transformation.

Each of the models should be formally defined by a metamodel, in order to make transformation between the models possible. For the CM metamodel we suggest to use the metamodel as shown in Figure 5.9 of our proposed modeling language. The prototype also includes a model builder which, based on the metamodel of a modeling language, automatically generates a visual model building tool. In the MDD4MS prototype, DEVS was selected and implemented for the PISM model and its metamodel is shown in Figure 5.12. DEVSDSOL (as mentioned briefly above) was selected for the PSSM and its metamodel is shown in Figure 6.2. To support the transformation between models, several M2M transformation languages exist. These allow to transform a source model (like for instance the CM) into a target model (like the PISM), based on a set of transformation rules. The transformation rules define how exactly source model elements are matched and navigated to create and initialize the elements of the target model [Ecl11]. Examples of these languages are ATL (ATLAS Transformation Language) and QVT (Query/View/Transformation) Operational Mappings. ATL was implemented in the prototype and the transformation rules for transformation from PISM to PSSM were already defined. Also the transformation from PSSM to an executable DSOL simulation model is supported. In Figure 6.3 the transformation of metamodels and instances of the metamodels (the actual models) is depicted.

To conclude this section, we need to state what needs to be developed in order to be able to use the MDD4MS prototype. We already suggested that for the conceptual modeling metamodel (see Figure 6.3) the metamodel of the proposed business process modeling language can be used. Based on this metamodel, a visual builder is then generated. Regarding the model to text transformation, we need to implement all DEVS simulation components in Java-DEVSDSOL in order to make them executable by DSOL. This will be discussed in Section 6.2. Lastly, the transformation rules from conceptual model to PISM need to be specified and this will be discussed in 6.2.3.

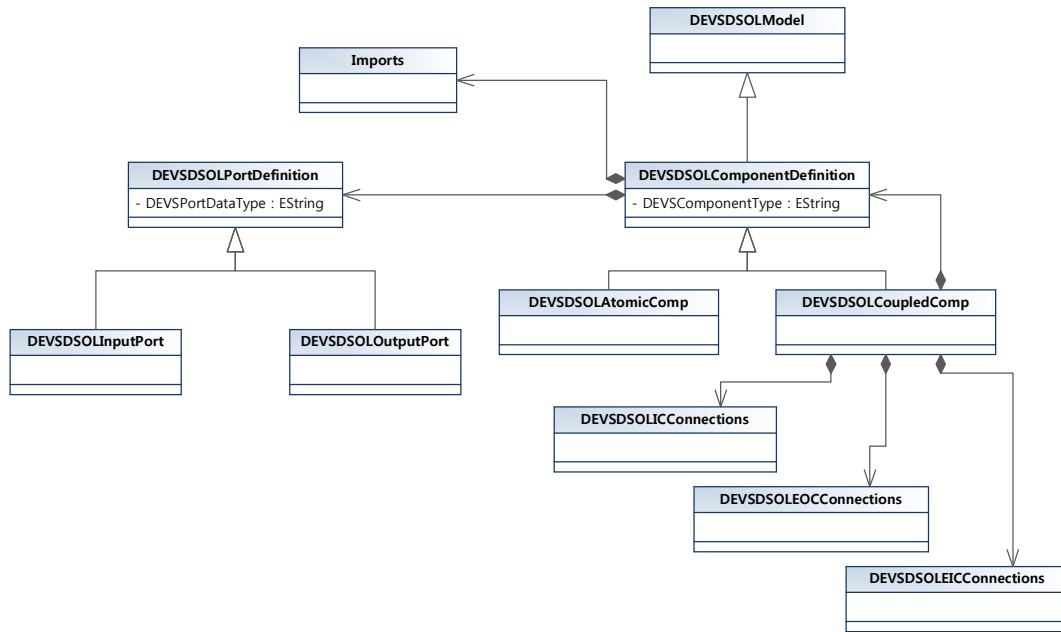


Figure 6.2: Simplified DEVSDSOL Metamodel (adopted from [CVS11])

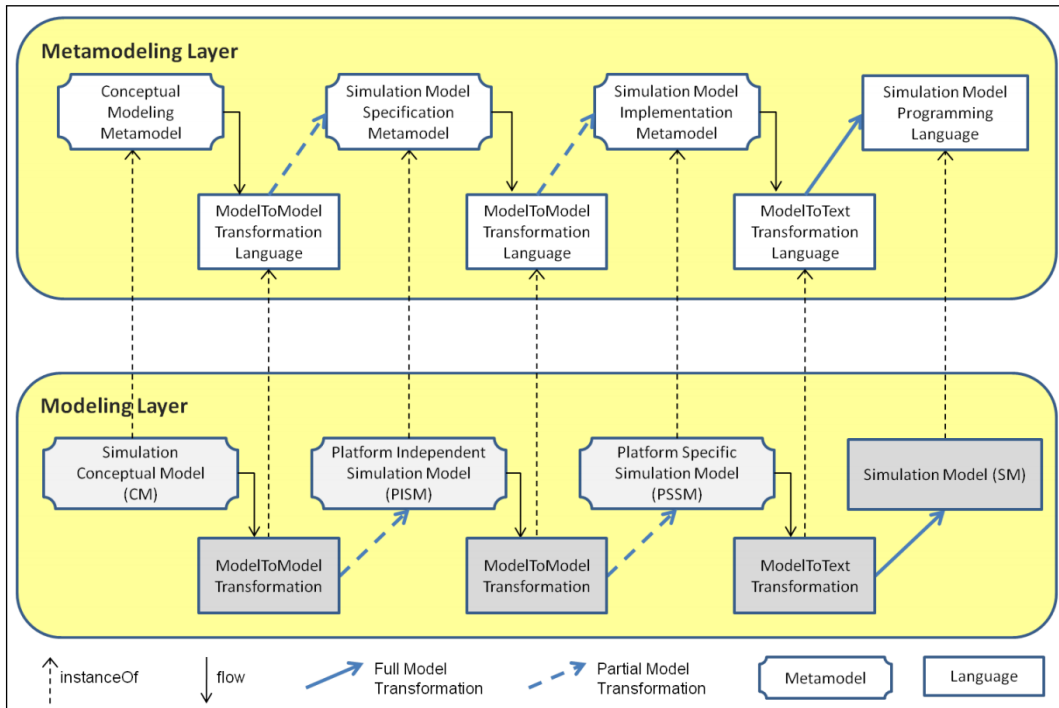


Figure 6.3: Metamodels and Model transformations in MDD4MS (adopted from [CVS11])

6.2 Development of a prototype

In this section we will discuss the development of a prototype. In Section 6.2.1 we will briefly elaborate on the DEVSDSOL library which is used for the implementation of the simulation components. In Section 6.2.2 we will discuss the actual implementation of the simulation components. Next, in Section 6.2.3 we will discuss the ATL-transformation rules which allow the prototype to transform a visually drawn conceptual model into a PISM (i.e., a DEVS-based simulation model). And finally, in Section 6.2.4, we will present the developed prototype itself.

6.2.1 Usage of DEVSDSOL

The DSOL library and the DEVSDSOL library are used to support the development of executable simulation components [Jac05][SV09]. Atomic DEVS models and coupled DEVS models are expressed as Java classes. The main methods of the atomic model class are the external and internal transition method, the output function (lambda) method and the time advance function method. The states of an atomic model can be instantiated by Phase objects and the Input and Output ports by respectively InputPort and OutputPort objects. In Listing 6.1 an example is given of how an input port and output port are instantiated, how a 'Passive' phase is instantiated, including the specification of its infinite lifetime, and the methods.

Listing 6.1: Example AtomicModel DEVSDSOL

```
public InputPort<Object> in = new InputPort<Object>(this);
public OutputPort<Object> out = new OutputPort<Object>(this);

public Phase passive = new Phase("passive");
passive.setLifeTime(Double.POSITIVE_INFINITY);

    deltaExternal(double e, Object inp) { ... }

    deltaInternal() { ... }

    lambda() { ... }

    timeAdvance() { ... }
```

In Listing 6.2 an example is given of how for a coupled model an input port and an output port are instantiated, how a new model (which can be an atomic or coupled model) can be instantiated, and how internal and external couplings are created.

Listing 6.2: Example CoupledModel DEVSDSOL

```
public InputPort<Object> in = new InputPort<Object>(this);
public OutputPort<Object> out = new OutputPort<Object>(this);

    AtomicModel var_AtomicModel = new AtomicModel(this);
    AtomicModel var_AtomicModel2 = new AtomicModel(this);

    this.addExternalInputCoupling(this.in, var_AtomicModel.in);

    this.addInternalCoupling(var_AtomicModel.out, var_AtomicModel2.in);
```

6.2.2 Implementation of simulation components in Java

Each component mentioned in Section 5.3.4 is implemented in Java code and combined in a Java library. Two separate Java libraries are constructed: the BPMNLibrary and the QueueingLibrary. The BPMNLibrary contains all the atomic simulation components as well as the Signal-object. The QueueingLibrary contain the Entity object, as well as all other objects that are not DEVS-based, but provide supporting functions to DEVS models.

BPMNLibrary

StartEvent.java	<—	Atomic Start Event component
EndEvent.java	<—	Atomic End Event component
Task.java	<—	Atomic Task component
SwimlaneEntry.java	<—	Atomic Swimlane Entry component
SwimlaneExit.java	<—	Atomic Swimlane Exit component
ResourceManager.java	<—	Atomic Resource Manager component
ExclusiveGateway.java	<—	Atomic Exclusive Gateway component
ParallelGatewaySplit.java	<—	Atomic Parallel Gateway Split component
ParallelGatewayJoin.java	<—	Atomic Parallel Gateway Join component
BPMNCoupledModel	<—	Coupled main model
Signal.java	<—	Signal object

QueueingLibrary

Entity.java	<—	Entity object
Queue.java	<—	Queue object
QueueArray.java	<—	QueueArray object
Resource.java	<—	Resource object
ResourceArray.java	<—	ResourceArray object

In Section 5.3.4 we discussed several simulation components, namely the Start Event, the End Event, the Task, the Swimlane, the Swimlane Entry and the Resource Manager. In this section we will discuss how these are implemented in Java code. For a full overview of all implementation and more details, the reader is referred to Appendix E.

Start Event implementation

The StartEvent-class is an implementation of the Start Event component and enables the instantiation of Entity-objects, which are send to the only defined output port (out). A StartEvent-instance is created and specified through its constructor (see Listing 6.3). The constructor allows to specify a unique identifier for each Start Event component; a description of the Start Event; the inter-arrival time mode (e.g., constant, uniform or exponential); several parameters to specify the statistical distribution function; the batch size (number of entities that are generated at once); and the time at which the first entity is created.

Listing 6.3: Constructor StartEvent.java

```
public StartEvent(BPMNCoupledModel parentModel, int myID, String description,
    String entityType, int mode, double mean, double min, double max, double
    stdDev, int batchsize, double firstCreationTime)
```

The Start Event-class uses the jstats distributions library (included in DSOL) for the creation of various statistical distribution from which random inter-arrival times are

acquired. Currently supported by the StartEvent-class are: Uniform, Exponential, Normal and Triangular distributions. Based on the mode that is specified in the constructor, a different distribution is used for the random creation of inter-arrival times. Each time an entity (or batch of entities is generated), the sigma of the model (time remaining before the output function is called) is reset to the next inter-arrival time.

To enable the output of statistics of a StartEvent-instance, a Counter object called counterEntitiesIn is instantiated. The counter object is also provided by DSOL. Every time an entity is send to the output port, the fireEvent-method is triggered (provided by DSOL), causing the the Counter-instance to be increased with 1. By using fire-events as provided by DSOL, we can use the statistics output interface of DSOL to display all generated statistics.

End Event implementation

The EndEvent-class is an implementation of the End Event component and enables the disposal of Entity-objects which arrive on the only defined input port (`in`). The EndEvent has two distinctive phases, passive and active with a lifetime of respectively infinite and 0 seconds. An EndEvent-object can be instantiated through its constructor (see Listing 6.4). The constructor allows to specify a unique identifier, as well as a description of the End Event.

Listing 6.4: Constructor EndEvent.java

```
public EndEvent(CoupledModel parentModel, int myID, String description)
```

An EndEvent-object generates several statistics about the entities and about the system. The statistical objects Counter (time independent) is used to count the number of entities which are disposed, and the Persistent-object (time dependent) is used to collect information about the total time of system of entities, the total time processed and the total time waited. The Persistent-object is also provided by DSOL.

Task implementation

The Task-class is an implementation of the Task component and enables an entity to delay for a certain amount of time. A Task-object is initialized through its constructor (see Listing 6.5) and the constructor allows to specify a description, unique identifier, mode for statical distribution selection and several parameters which specify the distribution function. The Task has two ports, one for receiving entities (`in`) and one for sending entities (`out`). Two phases are also defined: passive and active with a lifetime of both infinite (although the lifetime of the active phase is dynamically set depending on the arriving entity and possible other entities within the Task).

Listing 6.5: Constructor Task.java

```
public Task(CoupledModel parentModel, String description, int myID, int mode,
            double mean, double min, double max, double stdDev)
```

Although a task is modeled as a unique element, it represents a parallel series of tasks, for each resource within a swimlane. An option would have been to define the task as a

coupled model, with a atomic model as a switch-model and a series of atomic models to represent a task for each resource. This was developed and evaluated, and although it worked well for very small models, for larger models another implementation was chosen.

The Task consists of an ArrayList-object named `parallelTaskList`. Arriving entities are stored temporarily in the `parallelTaskList` following a specified order: the entity that is finished first is stored at the first location of the list, the entity that is finished after the first entity is stored at the second location, and so on. When an entity arrives, all remaining service times of the waiting entities in the `parallelTaskList` are updated by the `updateTimes()`-method. The `updateTimes()`-method decreases the remaining service time with the elapsed time since the last external or internal event (e). Next, the service time is calculated of the newly arrived entity (in a similar way as how the inter-arrival time is calculated in the Start Event) and is compared with the remaining service times. Based on these updated times, the newly arrived entity is stored in the `parallelTaskList` at the appropriate location and σ is set to the remaining service time of the entity which is first finished being serviced.

When e becomes equal to σ , and no external event occurred, the first waiting entity is removed from the `parallelTaskList`, its statistics are updated and the entity is send to the output port. Next, the `updateTimes()`-method is called again which updates the remaining times of the entities by decreasing each with the originally set σ (before the previous entity left. When no more entities are `parallelTaskList`, the phase of the Task changes back to passive.

Swimlane implementation

The Swimlane-class is the implementation of the Swimlane component. As was mentioned in Section 5.3.4 the Swimlane implementation represents a coupled DEVS model and contains the couplings between all components that are placed in a Swimlane. For each Swimlane that is modeled a new Swimlane-class is defined containing the unique couplings. Thus, in a DSOL simulation model, a Swimlane-object can only be instantiated ones. To clarify, if there are two Swimlanes in a model, named *ResourceA* and *Resource*, then `Swimlane.java` is copied and renamed to `Swimlane_ResourceA.java` and `Swimlane_ResourceB.java`. In each file, the unique couplings between the internal components are defined, as well as the instantiation of the components itself.

The Swimlane-class contains two methods which allows for the automatic generation of ID's for the Swimlane Entry and Swimlane Exit component. The ID's are for instance used to direct a signal to a specific Swimlane Entry: Its ID is used as sort of an 'address'. Also, the Resource Manager keeps track of all queues in Swimlane Entries, according to their associated ID.

Swimlane Entry implementation

The `SwimlaneEntry`-class is an implementation of the Swimlane Entry component and is initialized by the constructor as shown in Listing 6.6. The constructor allows to specify the identifier of the Swimlane that the Swimlane Entry belongs to, as well as a unique identifier. A `SwimlaneEntry`-object has two input ports (`in` and `inSignal`) and three output ports (`out`, `outSignal` and `outForwardSignal`). The five phases are

instantiated as specified in the component description section of the Swimlane Entry (see Section 5.3.4).

Listing 6.6: Constructor SwimlaneEntry.java

```
public SwimlaneEntry(CoupledModel parentModel, int swimlaneID, int myID)
```

The Swimlane Entry also instantiates a Queue-object, named `entryQueue`, in which arriving entities are stored until a signal is received which removes one entity from the `entryQueue` and sends it to the output port `out`. Each time an entity is added or removed from the `entryQueue`, a fire-event takes place to produce statistics about the average waiting time and the number of entities waiting in the queue (which are both time-dependent Persistent-objects).

Resource Manager implementation

The `ResourceManager`-class is an implementation of the Resource Manager component and is initialized by the constructor as shown in Listing 6.7. The constructor allows to specify: a description; the queue lane mode (how to select a queue from several queues with waiting entities); the identifier of the Swimlane that the `ResourceManager` belongs to; the number of resources per queue, the pattern value (used for cyclical pattern) and the number of queues in the swimlane where a resource can be directed to (excluding the Parallel Gateway Join queue). A `ParallelGatewayJoin`-object has one input port (`in`) and one output port (`out`). Two phases are instantiated, namely `passive` and `active`.

Listing 6.7: Constructor ResourceManager.java

```
public ResourceManager(CoupledModel parentModel, String description, int
    modeQueueSelection, int swimlaneID, int numberOfResources, int
    patternValue, int numberOfQueues)
```

The Resource Manager uses three objects to maintain information about the current states of all resources and queues within a Swimlane.

1. A `ResourceArray`-instance named `resourceArray` is instantiated which contains a number of `Resource`-instances (the number is equal to the value specified in the constructor). The `ResourceArray` itself maintains data about the average utilization of the Resources, and provides methods to change the state of a resource (available or busy). These resources are stored in an Array.
2. An integer Array-instance named `queueArray` contains the current number of waiting entities in all queues.
3. A double Array-instance named `longestWaitingTimesInQueue` maintains the current longest waiting time of every queue.

6.2.3 ATL-Transformation rules

The MDD4MS prototype uses ATLAS Transformation Language (ATL) for model transformation from conceptual model to DEVS-model, to finally DEVSDSOL-model. For the transformation from conceptual model (CM) to DEVS-model (PISM), three ATL-transformation files are created, namely:

- BPMN_2_DEVS_step1.atl
- BPMN_2_DEVS_step2.atl
- BPMN_2_DEVS_step3.atl

Three transformation steps are needed to transform all modeled elements to DEVS components, to generate all couplings, and to create some components (e.g., the Resource Manager) which is not modeled visually. The transformation from DEVS (PISM) to DEVSDSOL-model (PSSM) is done by two ATL-transformation files:

- DEVS_2_DEVSDSOL_step1.atl
- DEVS_2_DEVSDSOL_step2.atl

The transformation from DEVSDSOL model to Java files is done by the DEVSDSOL_2_Java application which was already included in the MDD4MS-prototype available. A Visitor-based interpreter is written with Java. This means that each model element is visited separately and the code is generated. Recursive visiting is done to generate coupled component code. This interpreter is added as an extension to the DEVSDSOL Editor plug-in which is part of the MDD4MS prototype. This allows to right-click on the DEVSDSOL-model to generate the code.

ATL-transformation files contain in general *rules* and *helper functions*. The rules specify how an instance in the source model is transformed into an instance in the target model. The transformation rules use the metamodels of the source and target models. For that reason, a BPMN_metamodel.ecore file is created, as well as DEVS_metamodel.ecore and DEVSDSOL_metamodel.ecore (the latter two were already included in the prototype). A helper function is an auxiliary function that computes a result needed in a rule. For instance, we use a helper to count the number of output-ports on an Exclusive Gateway element. The remainder of this section will elaborate only on the BPMN_2_DEVS-transformation². The DEVS_2_DEVSDSOL-transformation files were namely already included in the prototype.

BPMN_2_DEVS_step1.atl

In this first step, the first two rules transform the main model and the root object. Flow Objects drawn in the conceptual model in the root of the model are then transformed into Atomic DEVS components, which is followed by the Flow objects drawn within the Swimlanes. Next, the Resource Manager component is defined. This is followed by the transformation of the Swimlane into a coupled component. Because the Resource Manager is defined, the last rule also attaches the Resource Manager to it. For all components, ports are generated. However, because ports are not explicitly defined in the metamodel of the conceptual modeling language, these can not yet be coupled. This will be done in step 2.

Several helper functions are used in this first file. Because the modeling elements can be specified through parameter-fields in the property window of the model-builder, these values should be included in the atomic or coupled components. The helper function

²These transformation files were developed in collaboration with D. Cetinkaya.

supports this functionality, by reading the values attached to the conceptual modeling element, and copying those values to the component. Another helper function supports the specification of those components that are specified with distribution parameters. For instance, for a Start Event the modeler can specify through a drop-down menu whether the inter-arrival time is exponentially, uniformly, triangularly, normally distributed, or constant. In case the inter-arrival time is exponentially, the helper reads this and sets the value of the mode-parameter to a value of 2.

BPMN_2_DEVS_step2.atl

In the previous step, the ports were defined but could not be coupled yet. The main purpose of this step is to create couplings between the ports of the different components. In Figure 5.19 an overview was given of the different possible cases how components can be coupled. For each of these cases, a transformation rule is defined. Because several components have more than one input- or output-ports defined, for instance for sending entities and for sending signals, these rules are rather extensive. To check which case applies for a certain connection, multiple helper functions are defined. These perform checks whether for instance the parent model of a source component is different from the parent model of a target component. If this is the case, possibly a Swimlane Entry or Swimlane Exit component should be 'placed' between the two components.

BPMN_2_DEVS_step3.atl

This last step creates couplings between the Swimlane Entry components which are created in the previous step, as well as couplings between the Swimlane Entries components and the Resource Manager components. We mentioned before that Swimlane Entry components should be connected to each other through the outForwardSignal-port and the inSignal-port. These couplings are also created in this last step. Helpers are defined to enable for instance the creation of only one signal-coupling from the Resource Manager component in a Swimlane to the *first* Swimlane Entry component.

6.2.4 The final prototype

In Figure 6.4 a screenshot is shown of the developed prototype, based on the MDD4MS framework. The column on the left-hand side of the workspace shows an overview of the different ATL-transformation steps as were mentioned in Section 6.2.3, as well as output-files of a sample-model. On the workspace, a simple business process model is drawn with some of the proposed modeling elements. Each element can be specified with the Properties-window on the bottom of the screen. Depending on the selected modeling element, different input parameters are shown. In Appendix J an overview is shown how the available elements should be specified, in order to generate an executable simulation model.

In the palette on the right-hand side, the modeling elements are shown which can be used to draw a business process model. A model is drawn by selecting a component and dragging it on the workspace. Sequence Flows can be made with the Connection-tool, also shown on the palette on the right-hand side. A connection is made between two

elements by selecting the Connection-tool, then clicking on the source-element, and then on the target element. Each element should in this prototype still be specified with a parameter depicting its class. This allows the final model-to-text transformation rule to instantiate the correct objects.

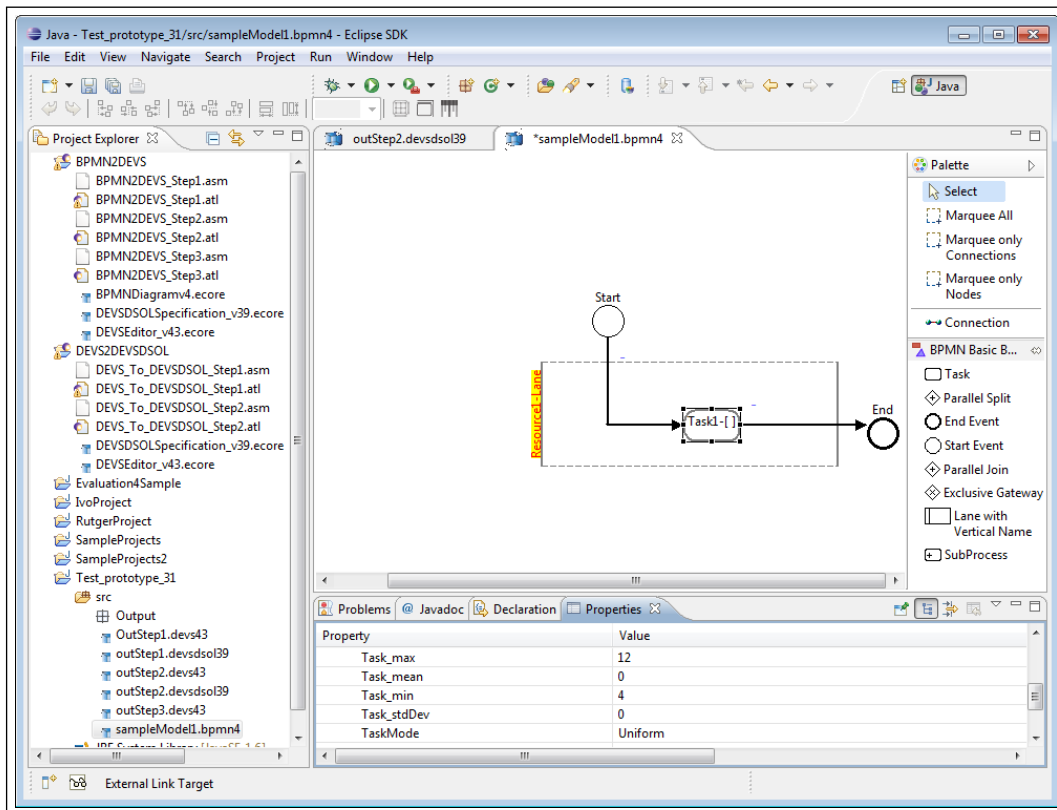


Figure 6.4: Screenshot MDD4MS model builder prototype

The execution of a generated simulation model requires some additional steps. However, when the created Java-files are copied into an Eclipse-workspace which also contains the BPMN and Queueing libraries, as well as the DSOL and DEVSDSOL libraries, the model can be executed using DSOL. A screenshot of the statistics output interface of the prototype, is shown in Section 5.4.2 in Figure 5.24. Various statistics are implemented in the prototype of which some can be displayed as time-dependent charts and box-and-whisker-plots, and of all a numerical overview containing the sample-mean, sample-variance, standard deviation, min and max can be shown. In Table 6.1 an overview is provided of the statistics that are gathered by the prototype and how they can be made visual. Each statistic is gathered per modeling element. If 2 Start Events are modeled, then these are shown separately.

Execution of the model can be controlled through the main DSOL interface and provides options to play, pause, stop and fast forward the model. Specification of the run-setup can in the prototype be done by specifying an XML-file containing the experimental frame setup.

	Time- dependent Chart?	Box- whisker- plot?	Statistics?
Entities			
Total time in system	yes	yes	yes
Total time waited	yes	yes	yes
Total time processed	yes	yes	yes
Total number of entities created	no	no	yes
Total number of entities disposed	no	no	yes
Resources			
Resource utilization per Swimlane	yes	yes	yes
Queues			
Number of entities waiting per queue	yes	yes	yes
Waiting time per queue	yes	yes	yes

Table 6.1: Prototype statistics generation

6.3 Verification of the prototype and components

In Figure 6.5 the concept of verification and validation as suggested in [Sar04] are shown in a simplified version of the simulation modeling process. In simulation studies validation is considered to be the process of determining whether 1) a conceptual model is a reasonable representation of the problem (conceptual model validity), and 2) whether the outcomes produced by a corresponding simulation model are sufficiently accurate to be applicable on the real world (operational validity). Verification relates to ensuring that the computer programming and implementation of the conceptual model is correct.

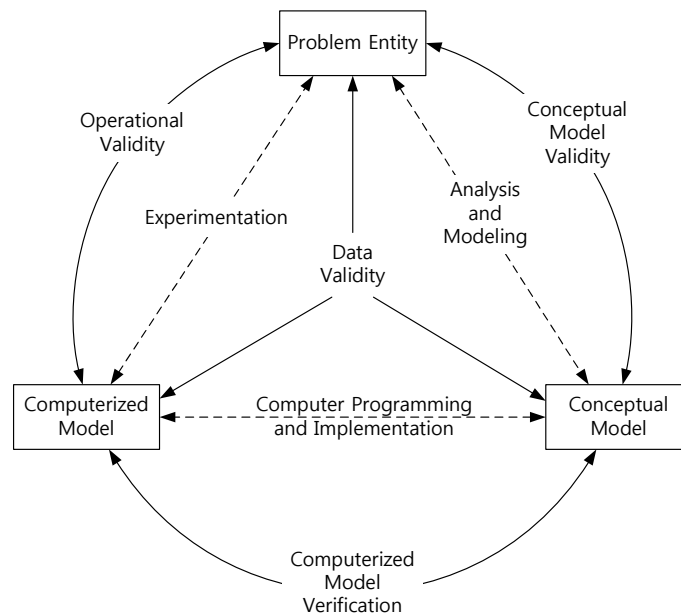


Figure 6.5: Verification and validation (Adopted from [Sar04])

Validation and verification of simulation models generally applies to specific simulation cases, like for instance the design and analysis of a cargo handling system of an airport. However, the design which is suggested and the prototype which is developed in this research (including the simulation components, graphical representation, graphical model builder and transformation rules), is not a modeling & simulation case on it self. It is intended to support consultants to model and simulate business processes. Our main question would thus be *whether the proposed solution is adequate for and usable by consultants to model real business processes*. Although it is not possible due to the limited amount of time to evaluate the prototype in a real setting (for instance a consultant uses it to model and simulate business processes of an actual client), we will evaluate the prototype with a sample case. This will be discussed in Section 6.4. We suggest however strongly to also evaluate the proposed solution by modeling a real case in a real situation, as this may depict limitations of the current implemented components and thus suggest improvements to be made to finally increase the overall credibility and quality of the solution.

Although it is difficult to evaluate the conceptual model and operational validity of a generic concept as developed in this research, we can evaluate the correctness of the prototype and the implemented simulation components. We are namely interested in knowing *whether a conceptual model is correctly transformed into an executable simulation and whether a generated simulation model behaves as how it is intended to behave (independent of whether the outcomes correspond to a real situation)?* Answering these questions is important, as this can confirm whether the implemented simulation components and transformed simulation models behave correctly.

The first question relates to whether the ATL-transformation rules as how they are specified, generate a simulation model which corresponds with the conceptual model. We are interested in knowing the following aspects:

- Are all components modeled using the model builder also instantiated correctly in the simulation model (location regarding their parent coupled model);
- are all component specifications using the model builder also interpreted correctly by the translator and specified as constructors of the components;
- are all the components coupled correctly, both the connections drawn visually in the model builder, as well as the intended connections (like for instance between all Swimlane Entry components within a Swimlane and between all Swimlane Entry components and the Resource Manager component).

We undertook several experiments in order to verify the correctness of the ATL-transformation rules. These experiments and the results are discussed in Section 6.3.1.

The second question relates to whether the simulation components are programmed correctly and both the behavior of the individual components as well as a complete simulation model can be considered as correct. Law & Kelton suggest eight techniques to verify whether a simulation computer program behaves correctly [LK00], which are:

1. Split the program up in modules or subprograms
2. More reviewers of the program

3. Run simulation under a variety of settings of the input parameters, and check to see that the output is reasonable
4. One of the most powerful techniques: Trace
5. Run model, when possible, under simplified assumptions for which its true characteristics are known or can easily be computed
6. Observe animation
7. Compute sample mean and sample variance for each simulation input probability distribution, and compare them with the desired (e.g. historical) mean and variance. This suggests that values are being correctly generated
8. use commercial simulation package to reduce the amount of programming required (but be care full for unknown subtle errors in package)

The reader is referred to [LK00] for an elaborate discussion about these techniques. The first technique, namely splitting up a simulation program is applied in the implementation by using the object-oriented programming approach supported by the Java programming language, as well as the usage of components representing the different atomic DEVS models. The second technique took partly place during the development of the components and corresponding state-diagrams. However, a suggestion for future work is to let experts verify also the implementation of the components. The third technique was done and will be (partly) covered in Section 6.3.2. Technique four, tracing the outputs of the model was done extensively during the development of the implementation, as well as afterwards. Technique five is also applied and discussed in Section 6.3.2. The implementation does not support animation of a simulation execution (yet), so technique six could not be applied. Technique seven as described in [LK00] is not directly applied, as we rely on the statistical distribution functions provided by the DSOL library. However, in Section 6.3.2 we do present results which resemble the principle of this technique. Technique eight is also party applied and discussed in Section 6.3.2.

6.3.1 Verification: model transformation

In this section we will evaluate whether a conceptual model developed using the conceptual model builder implemented in the prototype is transformed correctly into an executable simulation model. Generally, only two files are generated by the prototype, namely `SwimlaneModel.java` and the Swimlane coupled models (name is similar to `Swimlane_NameOfSwimlane.java`). We thus only provide the syntax of these files in Appendix H.

Test 1: One Swimlane and one Task

In Figure 6.6 the conceptual model is drawn of a business process with one Swimlane (containing 3 resources), one task, one Start Event and one End Event. Executing the transformation steps results in two coupled models, `SwimlaneModel.java` and `Swimlane_Resource1.java`. The syntax is included in Appendix H in Listing H.1 and Listing H.3 and Listing H.2.

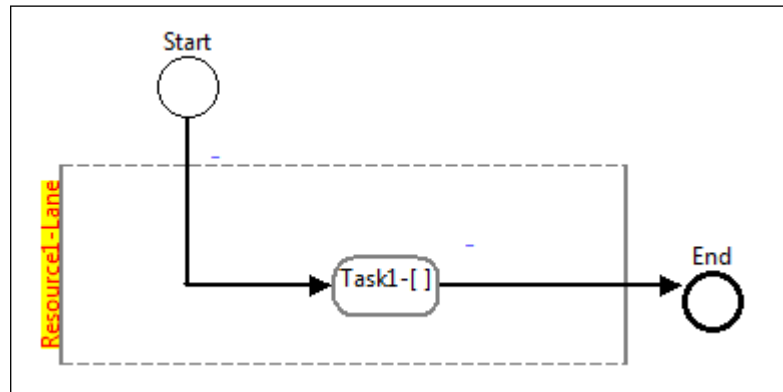


Figure 6.6: Test 1: conceptual model

We verified the generated ports, couplings and objects, as well as the specification how these were entered in the model builder, and concluded that the transformation was performed correctly and the resulting simulation model was executable.

Test 2: Two Swimlanes

In Figure 6.7 the conceptual model is drawn of a business process with two Swimlanes, of which the top Swimlane contains 10 'Resource_A' employees and the bottom Swimlane contains 10 'Resource_B' employees. Two types of entities are generated by separate Start Events, after which a series of (parallel) activities is performed by resources from both Swimlanes. Finally, the entities are disposed by the End Event.

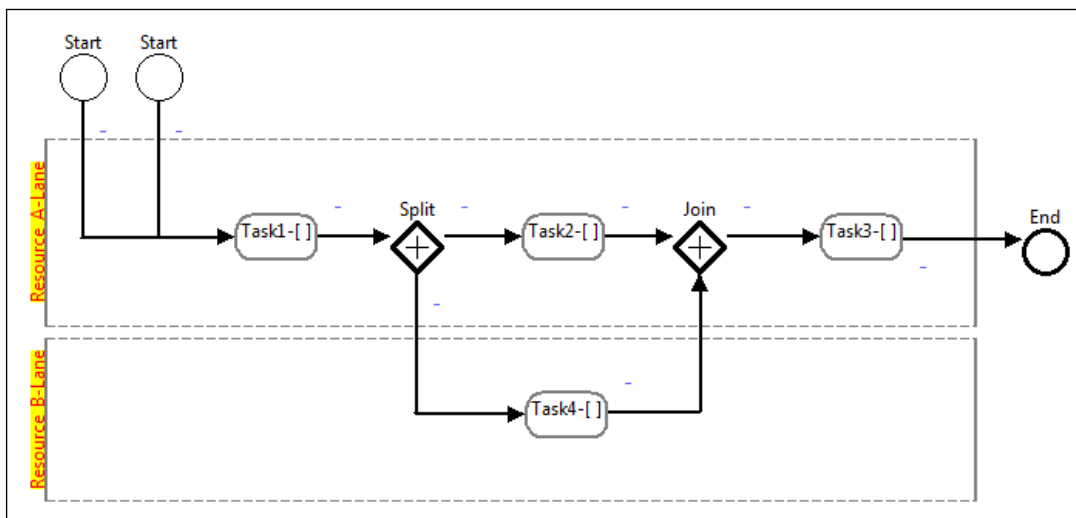


Figure 6.7: Test 2: conceptual model

The generated syntax of `SwimlaneModel.java` is included in Appendix H in Listing H.4. The syntax for the ports and couplings of `Resource_A` swimlane are listed in Listing H.5 and Listing H.6. The syntax for the ports and couplings of `Resource_B`

swimlane are listed in Listing H.7 and Listing H.8.

After verifying the correctness of the generated coupled models as well as the specification, we conclude again that the conceptual models generated through the visual builder are correctly transformed into executable simulation models.

Conclusion

During the development and specification of the transformation rules, programming errors were found several times and these were corrected. The two tests performed in this section were both successful. Several other test were performed (as part of an iterative development and improvement process), and although these were successful they were not documented. It is advised that larger, more complicated models should be modeled and the transformation should be verified. However, due to the limited amount of time available, we conclude thus far that the transformation rules are adequate to translate a visual build business process model correctly into an executable simulation model. One exception on this is the generation of coupled models representing Sub Processes. This was namely not implemented in the final prototype.

6.3.2 Verification: correctness of behavior and outcomes

To increase the credibility of the proposed simulation components and corresponding implementations, we need to ensure that the behavior and outcomes of a developed simulation model (using the implemented simulation components) are correct. To do this, we make a distinction between the correct working of the individual components and between a model of coupled components. Regarding the correct working of the individual components, we want to ensure that each component works as how it is designed. For instance, when we specify a Start Event to generate entities with an average interarrival time of 1 minute, we expect that in one hour approximately 60 entities are generated. When we specify a two-exit Exclusive Gateway based on a probability of 70% of the arriving entities will leave through the first outgoing flow, and the remaining 30% will leave through the second outgoing flow, we expect that the outcomes will also depict this distribution.

First, two tests are performed with rather simple simulation models are developed which include the proposed simulation components separately. This allow us to verify the component behavior, as well as the generation of output statistics. By using some of the techniques mentioned in the previous section, we will confirm the correctness and thus increase the overall credibility of the solution and implementations. The thirds test includes a business process model with a parallel activity. And finally, the fourth test, is a more complicated simulation model is developed which includes a combination of the various implemented simulation components. This allows us to verify the correct interaction between the component.

The simple models are based on a single-server queuing system (SSQS) (see Figure 6.8). In a SSQS there is one server which can serve customers. When a customer arrives (after an average interarrival time) and the server is idle, the customer is serviced for a specified average amount of time, after which he leaves and the server returns to an idle state. When a customer arrives and the server is currently busy, the customer will wait in a

queue until the server becomes idle again. When the server becomes idle, the first waiting customer will be serviced by the server (which becomes busy again). When there is one server and the interarrival and the service time are exponentially distributed, the SSQS is known in the queueing theory as an M/M/1-queueing system. Queueing theory provides us with formulas to calculate for instance the utilization of a server and the average delay of arriving customers, over a large number of customers. The verification of the simple models is discussed in Section 6.3.2.1.

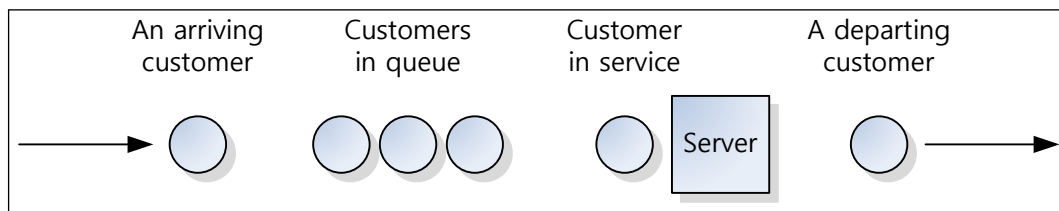


Figure 6.8: A single-server queueing system

The more complicated model is based on a description provided by consultants from Accenture how activities (both sequential as well as parallel) are undertaken in actual business processes. This is already discussed in Section 5.2 under the header “Activities and Decisions”. The business process model is included in Appendix G and shown in Figure G.4. Because the outcomes of a simulation model representing this business processes is too complicated to be compared with what the outcomes are expected to be, we use the simulation software *Arena*³ to compare the outcomes of a model developed using the prototype with the outcomes of a corresponding simulation model developed with the *Arena*.

Arena is a widely accepted simulation package (both academically as well as in the industry) and provides simulation capabilities which can be – to some extent – considered as correct. *Arena* provides numerous possibilities to develop and specify simulation models. According to us *Arena* also allows to develop and specify simulation models which should behave similarly and produce outcomes that are expected to be statistically related to simulation models as will be developed and specified by our prototype. The choice for *Arena* was made because 1) a student license was available; 2) the experience we have in developing simulation models with *Arena*; 3) simulation models developed using the proposed simulation components can also be modeled using the modules provided in the *Arena* templates; and 4) we consider *Arena* not only able to model and specify a business process as is done by our solution, but it is also able to adequately simulate the behavior of such a model and provide trustworthy outcomes. To underpin this last point, we also use *Arena* for a comparison with the simple models.

To evaluate the correspondence of the outcomes of a simulation model based on the proposed simulation components with a similar model developed in *Arena*, we will compare outcomes statistically by applying an ‘independent two-sample t-test’ using the statistical data analysis software SPSS⁴. We expect namely that the means of the outcomes of the two simulation models are nearly equal (independent of which specific

³Arena Simulation Software by Rockwell Automation (<http://www.arenasimulation.com/>)

⁴IBM SPSS software (<http://www.ibm.com/software/analytics/spss/>)

outcomes we select to compare). This test allows us to compare the means of two data sets which are expected to be equal. We have to take into account that the generation of random values based on specified statistic distribution functions is most likely different in DSOL and Arena, due to for instance the implemented (pseudo) random-number generators. However, when sets with a large sample size are compared, the means and variance of both will approach each other (law of large numbers).

6.3.2.1 Component verification

Test1: Single-server queuing system

The first system which is modeled is a single-server queuing system with fixed run length, as shown in Figure 6.8. Figure 6.9 shows the coupled DEVS model corresponding to the SSQS. In Appendix I the complete analysis is included of the SSQS modeled using the implemented simulation components, including also outcomes of statistical tests. We compared the generated DEVS model with the Arena model, based on 1) the number of entities generated, 2) the server (resource) utilization, 3) the average number in queue, 4) the average delay in queue, and 5) the number of entities disposed.

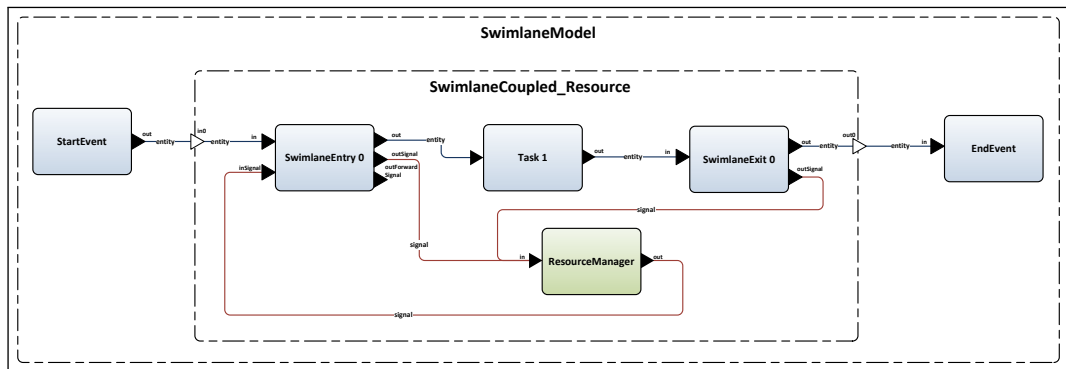


Figure 6.9: DEVSDSOL model single-server queueing system

Based on the analysis as discussed in Appendix I we concluded that the simulation components behave as expected. Although it seems that there are small differences between values generated by our prototype and a corresponding Arena-model, statistical tests proved that the means of the before mentioned statistics (of 20 executions of both models) are equal. Based on the tests, we concluded that (so far) the behavior of the implementations of the Start Event, the Task, the End Event, as well as the Swimlane Entry, the Swimlane Exit and the Resource Manager seems correct.

Test2: Single-server queuing system with Exclusive Gateway

In Figure 6.10 another business process model is shown, similar to the SSQS, but now we included an Exclusive Gateway to verify its behavior. The component was specified to send 75% of the arriving entities through the top Sequence Flow, and 25% through the bottom Sequence Flow. Again, we performed a statistical test (One-Sample T-Test), and we found that the number of entities being disposed by “End Event 1” is equal to

75% of the generated entities, based on 20 executions (see Appendix I). We can thus conclude that the behavior of the Exclusive Gateway component is also correct.

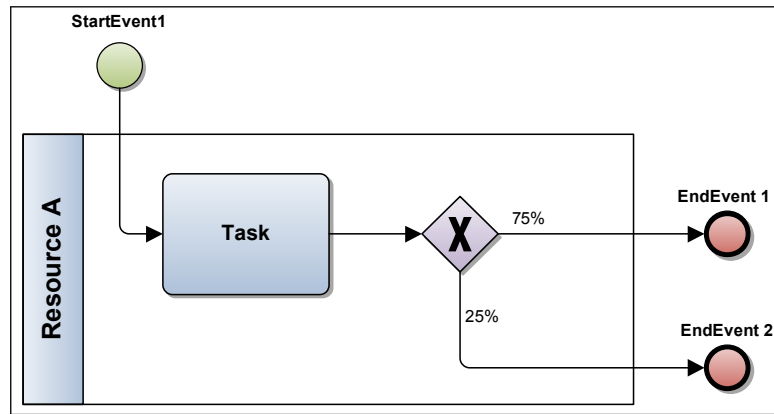


Figure 6.10: Test 2: Single-server queuing system with Exclusive Gateway

Test 3: Two Swimlanes - Parallel Activity

In Figure 6.11 the model is shown which was modeled using the prototype. Figure I.11 in Appendix I shows the corresponding Arena model. The simulation models were executed for 18 000 seconds and all component parameters were defined in seconds. By running the models for 18 000 seconds, in total approximately 4500 entities were generated and processed by the models.

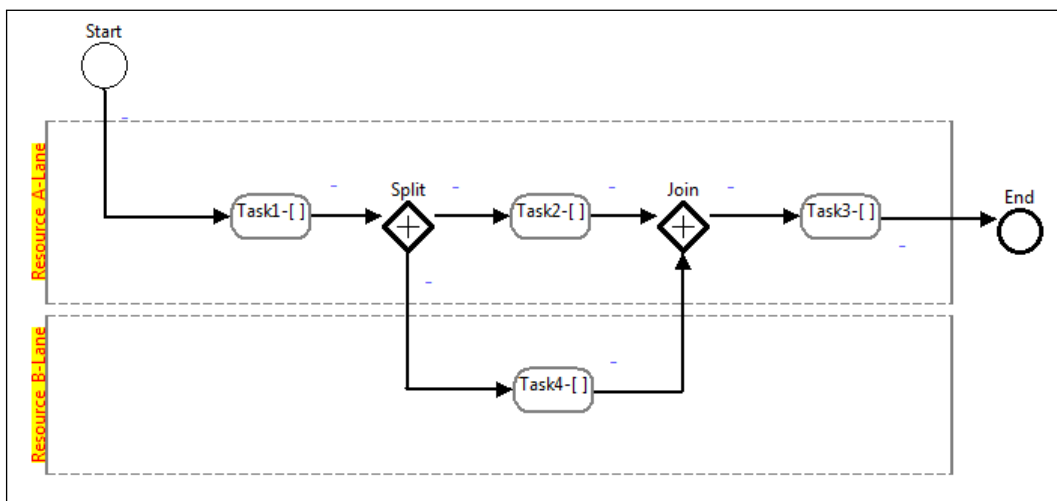


Figure 6.11: Test 3: Business process model as drawn with prototype model builder

In Table I.7 a comparison is made between the outcomes of Arena and of the DSOL simulation model, regarding the total number of entities generated and disposed during the execution, and the average resource utilization. In Table I.8 a comparison is made

between the total time in system of the generated entities. In Table I.9 the total processing times are compared, and in Table I.10 the total waiting times are compared.

Due to the limited amount of time available, we will not perform a statistical test to compare both the Arena and the prototype results. Based on the small differences between the values, but also the trace we commenced separately, we conclude that the behavior of the components is in this model also correct. The only more significant difference is between the total time in system and the total waiting time. These difference relate to how these times are calculated if parallel activities appear within a business process. If an entity is duplicated and later-on synchronized, it should be decided whether the total waiting time is the sum of both entities (the duplicate and the original) since the split, or for instance the maximum of both. The same question can be asked for the waiting time parameter. We did not have time available to get a clear confirmation how consultants see this.

Another aspect that was found and should be considered, relates to the Queue-statistics of the Parallel Gateway Join component. When an entity arrives at this component and its duplicate is already waiting, should the last entity first enter the Queue (and thus influence the statistics), or can it immediately be combined with the waiting entity? Arena includes the last arrived entity in the Queue, before it sends the combined entity forward. We however found this rather vague and suggest that more research is undertaken on this topic, to guarantee accurate Queue-statistics.

Test 4: Two Swimlanes - Parallel Activity and Exclusive Gateways

The last test which is performed as part of this component verification section, is a rather extensive model (compared with the previous models). It has already been discussed before and the business process model is shown in Appendix G and in Figure G.3. The corresponding DEVS model is shown in Figure G.4 and corresponding Arena-model is shown in Figure G.5. Similar to Test 3, we will not statistically analyze the statistics generated by the Arena and the DEVS model.

Based on the tables shown below (Table 6.2, Table 6.3, Table 6.4 and Table 6.5) we conclude again that the results match rather closely and thus we conclude that the components, even in a more complicated business process model, behave correctly. Further research can elaborate on these values and undertake more in-depth analysis of both models, in order to confirm that the values generated by both models are 'equal'.

	Arena	Prototype
Entities		
- Total In	805	810
- Total Out	798	803
Resource Utilization		
- Resource Front Desk	0.75	0.71
- Resource Back Office	0.57	0.65

Table 6.2: Test4: Resource utilization

	Arena	Prototype
min	83.67	90.11
max	294.36	279.38
mean	151.76	152.59

Table 6.3: Test4: Total Time in System

	Arena	Prototype
min	100.87	97.16
max	225.09	229.34
mean	153.62	150.00

Table 6.4: Test4: Total Process Time

	Arena	Prototype
min	0	0
max	194.94	172.51
mean	6.506	6.621

Table 6.5: Test4: Total Waiting Time

6.4 Case study: A Telecom Operator

In this section we evaluate whether the modeling elements and corresponding simulation components are adequate to model and simulate real business processes. A preferred option would be to let consultants use the prototype in a real client's environment and model real business processes. However, due to the limited available time, as well as some limitations of the prototype, we decided to model ourselves a sample case using the prototype. This would allow us also to see what functionality is missing from the components to be used in a real situation. Also, it would allow to continue the verification of our implementation and prototype as discussed in the previous section. Models and figures regarding the case models are included in Appendix K.

6.4.1 Case description

The case is based on an actual project undertaken by Accenture for a large telecom operator. A conceptual business process model was provided in the form of a BPMN-model made with the Visual Paradigm software⁵. To give an indication of the scale of the original model that was provided: approximately 100 Start Events were modeled to depict arriving entities; approximately 260 End Events were modeled to depict the leaving or disposed entities; approximately 670 separate activities are performed by 28 different resource groups and approximately 260 decisions are made by these resources. Based on these number, we concluded that modeling the full business process model would be too time-consuming. Besides the case-model, a Microsoft Excel specification-sheet was also provided with the data about the duration of the activities, as well the probabilities of the decisions made.

We selected only one department of the organization, which is responsible for repair services and installing of equipment at customer's homes. In total, 26 Activities are performed by three different groups of resources: the supply technicians, the service technicians and the back office (which was in the original case considered to be a 'black box'). Entities with differently specified entity types, like order-tickets and phone calls, arrive from both inside, as well as outside the organization. The service and supply technicians have no direct contact with each other, but they can contact the back office to ask questions when they don't know the answer on a certain customer problem. The models of this department are included in Appendix K (see Figures ?? and ??).

⁵Visual Paradigm: <http://www.visual-paradigm.com/>

6.4.2 Model development and transformation

Based on the provided model, we developed a business process model using the prototype and modeling elements. The business process model provided to us was modeled using BPMN-elements. Most of the model could be directly transformed to our modeling notation as most modeling elements are implemented in our design. However, we did not define an Intermediate Event modeling element. The Intermediate Event is used within the original model to facilitate connections between different departments or different parts of the same process without having to cross the whole business process. The idea is that an entity which arrives at an Intermediate Event, is send to the Intermediate Event with the same name. Because we did not define nor implement this functionality, we checked whether the Intermediate Events are linked within the same model, or outside the model (thus pointing to another department). In the first case, we replaced the source and target Intermediate Event with a Sequence Flow, thus creating a longer / crossing connection. In the second case, we replaced the Intermediate Events with Start and End Events and placed these outside the Swimlanes (to enable the automatic creation and coupling of Swimlane Entry and Exit components by the prototype).

Parallel Gateways drawn in the original model are used to depict the duplication of an entity which is then send to another department or outside the organization. We defined Parallel Gateways used as a pair to enable parallel activities by multiple resources. However, the Parallel Gateway Split component can also be used to clone entities and send these clones immediately outside a Swimlane to an End Event, or another Swimlane⁶. This way, additional statistics can be acquired from a simulation execution. Another aspect concept used in the original model was to direct entities in a certain direction based on probability or based on entity type. The implemented Exclusive Gateway component supports this behavior as well and was specified according to its original model counter part.

For specification of most of the model, the data provided by Accenture was used. However, for the specification of the Start Events we estimated some values which we considered likely as actual inter-arrival times of entities. In Figure ?? the model is shown as how it was build using the prototype model builder. After the developed model was validated to correspond with the original model, it was transformed using the ATL-transformation steps into a DEVSDSOL simulation model. The generated coupled DEVS models (representing the overall model as well as the Swimlanes) were verified for correctness of the model transformation. An error was found which relates to the different constructors used to specify an attribute or probability-based Exclusive Gateway. The error in the first ATL-transformation-file was corrected and finally the model was correctly transformed into an executable DEVS simulation model.

6.4.3 Conclusions

A sample screen shot of the statistics that are generated by running the simulation for 5 hours, is shown in Figure 6.12. Due to the absence of output-data about the real business

⁶The modeler should however be well aware that a cloned-entity re-entering its original Swimlane will not claim a resource (as it is expected to be synchronized in order to fulfill a parallel activity).

processes, we are unable to validate the outcomes of the generated model. This would require more information about the processes itself, as well as executing the model for a much longer period of time (to make a possible warm-up period also visible).

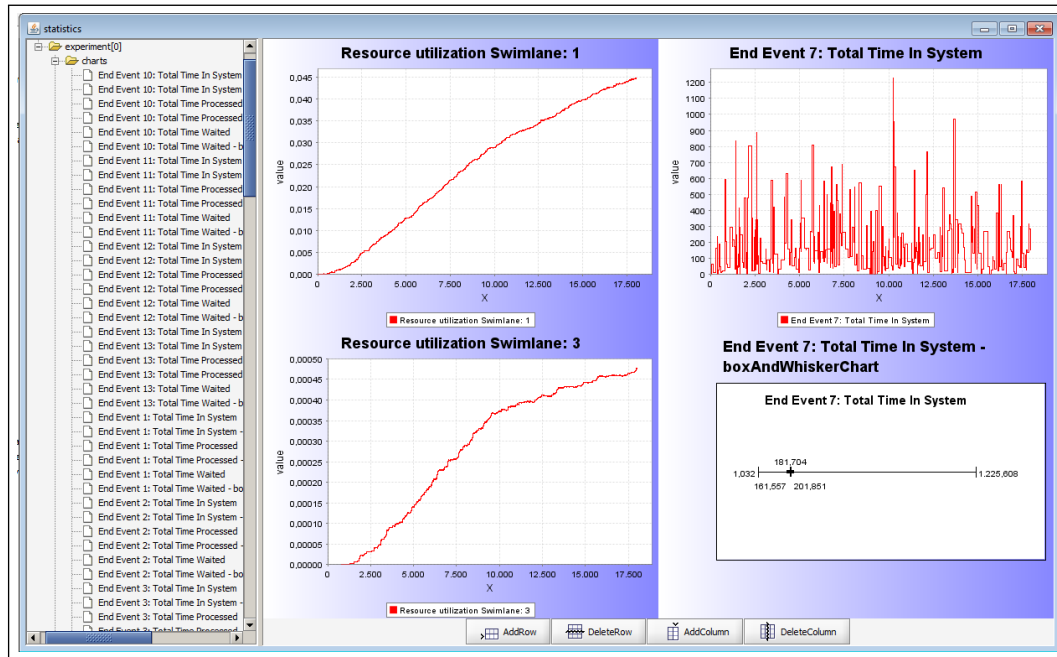


Figure 6.12: Case study: Output statistics example

With regard to the prototype, we can conclude that even more complicated business process models can be modeled using the prototype. We were namely able to model a real business process case with the implemented modeling elements and simulate this directly. However, during the construction of the case-model the prototype crashed several times, which makes it currently rather unusable in an actual consultant-client situation. We can also conclude that complicated models, including component specifications, are also correctly transformed into executable simulation models. With regard to the usage of DSOL for the prototype (and possibly later developments), we find that execution of a simulation model takes rather long. The run-setup was set to 5 hours, but this took more than 5 minutes to execute. In case the run-setup of the experiment would be set to 1 month, or even a year (which is a more likely value), it would require an extensive amount of execution time.

Besides evaluating the functioning of the prototype, we are more interested in knowing to what extent the proposed modeling elements and implemented simulation components are adequate to model real business processes. For this Telecom Operator-case, the proposed elements are adequate to represent the business processes. This depends also on the provided BPMN-model and to what extent this included all details about the real business processes. However, what is not yet implemented in the components is the specification and modification of alternative attributes of entities, besides the type of an entity. We noticed that in some process models of other departments of the organization this would be required for a correct representation (e.g., when an activity is repeated

several times before an alternative activity is performed).

Lastly, we suggest that an extensive model is developed, for which all model specification values, as well as output statistics are available. This would increase the credibility of the implementation and usefulness of the solution even more.

6.5 Usability evaluation with consultants

As part of the proof of concept, the usability needs to be evaluated. This provides feedback on the original stated design requirements (see Section 4.3), as well as recommendations on how a next prototype can be improved. To evaluate the usability of the prototype, evaluation workshops are held with management consultants and afterwards questionnaires are distributed and collected. This section elaborates on the organization of the workshop and summarizes the main findings.

6.5.1 Evaluation workshop and questionnaires

Goal of the evaluation sessions is to evaluate the usability of prototype and modeling approach with participants. Its main purpose is to get direct feedback about the usability and functionality from participants when actually working with the prototype. The workshops can be organized in a one-on-one manner, or in a group-setting with several participants involved during one evaluation session. Group evaluation enables active discussion between the participants and the facilitator, but it is more difficult to capture all the participants' responses. With one-on-one sessions, it is possible to use a *think-aloud procedure*, where the objective is to get a participant to talk continuously [Gal07]. The participant actively works with the prototype by performing a series of tasks and mentions out-loud all his findings and confusions. This may uncover details that may not otherwise be mentioned.

*Think-aloud
procedure*

An evaluation session is suggested to have a duration between 60 and 90 minutes [Gal07]. Because of this limited amount of time, only three steps are defined for the participants. These steps contain modeling and specification activities of business process, whereby for each step more modeling elements should be added. Sample models are developed on before hand, but the participant did not get to see these. An explanation of each business process was instead provided and the participant had to find out how to use the prototype to make the process model.

Step 1 Model a business process which contains three activities performed by 1 group of resources. Only one entity type arrives after a certain inter-arrival time and leaves after the three activities are performed in sequence. See Figure L.1 in Appendix L for the corresponding business process model.

Step 2 Extend the previous model with an extra resource type. Based on a certain probability, the new resource performs the second task. The first resource always performs the first and the last task. See Figure L.2 in Appendix L for the corresponding business process model.

Step 3 Implement the following process characteristics, namely that there are three options how an arrived entity is processed: 1) The first resource can perform the second tasks him self completely; 2) the second resource performs the second task completely; or 3) both resources split the task and synchronize the work later on. See Figure L.3 in Appendix L for the corresponding business process model.

The questionnaire which will be given after the workshop to the participants to fill in, includes questions related to the usability of the prototype, but is also meant for evaluation of the overall modeling approach (e.g., the selection of modeling elements provided). The questions are based on the description provided in [Nie93] what usability relates to, namely 1) Learnability; 2) Efficiency; 3) Memorability; 4) Errors (Accuracy); and 5) Subjective Satisfaction. The questions included in the questionnaire are shown in the list below⁷.

1. *Do you find it easy to learn to work with the prototype and the components?*
2. *Do you find the proposed modeling approach efficient in use?*
3. *Do you think you will easily remember how to work with the prototype and the components, when you will use it again?*
4. *Do you consider working with the prototype and the simulation components as being pleasant?*
5. *Do you consider the proposed modeling approach (simulation components and way of using the prototype) easy to use to develop a business process model?*
6. *Do you think the proposed modeling approach can work well in a real situation with involved clients (usage of the tool, understandability of a model, communication about a model)?*

6.5.2 Outcomes and findings

In total, two evaluation sessions and a separate demonstration were held with in total three management consultants from Accenture. All involved consultants received a questionnaire and two were send back. An overview of the aspects mentioned during the evaluation workshops is included in Appendix L. This overview also included suggestions for improvement provided by the participants. Although in general when during a think-aloud workshop suggestions for improvement are given by participants, the facilitator should find out what the reason is for a participant to suggest it. However, although the prototype is still in an early versions with several shortcomings, as well as the suggestions were considered as common sense, we documented everything. The received questionnaires are also included in Section L.4.

Through the evaluation workshops much useful information and suggestions were acquired about the modeling elements and usage of the prototype itself. A summarized

⁷We are aware that some of the terminology used in the questionnaire does not correspond with the terminology used in this document. However, we wanted to include the questions in this section, as how they were originally formulated in the questionnaires.

overview of suggestions and shortcomings is provided in the list below. For a full overview, see Appendix Section L.

- There is no visual difference between the Parallel Gateway Join and Parallel Gateway Split, which makes a model more difficult to interpret.
- Some confusion was also recognized, when a participant confused the X-marker of an Exclusive Gateway, and the +-marker of a Parallel Gateway
- Much more clarity would also be provided if names or descriptions of for instance the Start Event and Exclusive Gateway are shown next to a drawn element.
- Specification of the time-unit (e.g., seconds, minutes, hours) should be possible.
- Model experimentation (e.g., changing parameters of an element) during the execution of a model was mentioned as a useful feature.
- Extended user-friendliness of drawing the model would be a great improvement, e.g., connecting a drawn Sequence Flow to another object, or splitting and re-connecting a drawn Sequence Flow when an element is placed on top of the flow.
- Regarding the output interface, it would be more convenient to use the name of the first element after a Swimlane Entry (for which currently a name can not be specified) for its queue-statistics.

Many issues and suggestions were mentioned during the evaluation, but finally both participants were able use the prototype and modeling elements to model the three business processes. Although the prototype model builder itself is rather limited regarding its usability (software crashes, user-friendliness, . . . etc), this evaluation has proven us that the defined modeling elements, as well as the suggested modeling approach (drawing models by drag-and-drop of elements), *is a fruitful concept to support management consultants with modeling and simulation of business processes.*

The received questionnaires confirm this last conclusion. According to one respondent “*the prototype is very intuitive*” and “*model building is very efficient*”. The same respondent mentions that specification of the modeling element parameters could be made simpler. Unfortunately, the prototype-version that was used during the evaluation session did not include the latest updates, which allows a modeler now to specify model-parameters more easily (see screen shots in Appendix J). Lastly, the respondent mentions that the modeling elements *force* the modeler (i.e., management consultant) to think how business processes really are and work. This, in our opinion, improves the chance for increased conceptual validity of a developed model.

The second respondent mentions that this solution combines the advantages of easy-to-use business process modeling software (like Microsoft Visio), with the advantages of simulating the business processes. Possible areas of usage for the suggested tool are process improvements projects like Lean Six Sigma (which require short time-lines, client workshops and validation meetings), as well as sizing of teams (i.e., number of FTEs required) in a business process. One aspect that would improve the usability is the usage of arrival rate, as opposed to inter-arrival time as is currently implemented. Consultants, as well as their clients, are more likely to think in number of events occurring

in a certain amount of time (e.g., 70 arrivals per hour), then in the time between events. This would increase the understandability and ease of specification. Finally, the respondent mentions that “*visuals are very important for usability during workshops*”, like for instance animations of the flow of entities, as well as waiting entities in queues.

6.6 Design requirements evaluation

Before continuing on to the main conclusions of this research in the following chapter, we will first discuss the originally stated design requirements (see Chapter 4) and evaluate to what extent they are finally met.

Requirement 1. *The tool should be able to simulate the dynamic behavior of events and interactions within business processes.*

This requirement relates to the usefulness of a decision support tool [KS08]. Our prototype incorporated DSOL to support the simulation functionality of it. In this chapter we demonstrated that a DEVS simulation model based on the set of proposed conceptual modeling elements was executable and produced valid results.

Requirement 2. *The tool should accommodate an intuitive approach for consultants to develop and specify a business process model.*

In the previous section we evaluated the usability of the prototype which supports a drag-and-drop manner for the construction of business process models. It was regarded by participants of the evaluation workshop as being intuitive and efficient to use. The prototype could be further improved, by looking at tools that deploy a similar way of developing models (e.g., Microsoft Visio, Visual Paradigm, . . . etc). The final prototype supports also specification of the model-parameters through property windows (see Appendix J). Improvement can be made by allowing the modeler to also specify the time-unit.

Requirement 3. *The tool should provide a modeling language to represent business processes which connects with the mental model of a management consultant.*

The modeling language which is proposed in this research is based on the mental model and understanding of business processes of management consultants. We used a User-Centered Design approach to incorporate this into the modeling language. It was mentioned in a received questionnaire that the logic of the modeling language is similar to the real life logic in a business process. To re-confirm whether the modeling language is really adequate to represent business processes, the language should be used by management consultants in a real project setting.

Requirement 4. *The tool should provide a modeling language to represent business processes which is understandable to all stakeholders.*

As was mentioned before, the inclusion of the word *all* makes this requirement difficult to satisfy. However, we have shown that it was possible to model and execute a simulation model based on a BPMN-model developed by consultants and validated by client-side

stakeholders. Because our modeling language is very similar to the one used in this model, we tend to believe that our model is also easy to validate. However, further research is suggested to focus on increasing the understandability of the modeling elements.

Requirement 5. *The tool should present relevant statistics of a simulation model execution in an understandable manner.*

Currently, the prototype produces various statistics related to the resources, entities and queues. These statistics can be presented in time-elapsing graphs and in box-and-whisker-plots. It was mentioned that the presentation is understandable and that this selection of statistics is enough for for instance operational excellence projects or sizing of project teams. More improvements can however be made with regard to the naming and grouping of these statistics, as well as the possibility to select an entity type for which statistics are then presented. Lastly, a consultant may want to be able to select part of the business process model and have statistics presented only about this part.

Requirement 6. *The tool should allow specification of the simulation run set-up in a manner understandable to consultants.*

The prototype allows a modeler to specify the simulation run set-up through an XML-file. This is however not very user-friendly and further research should focus on this aspect. In Figure 6.13 a screen shot is shown as an example how the simulation run set-up is specified in Arena simulation software.

Requirement 7. *The tool should be able to transform a conceptual business process model into an executable simulation model.*

We specified a series of transformation rules which define how a conceptual model can be transformed into a simulation model. We implemented this in the prototype by using the MDD4MS-framework and specifying a series of ATL-transformation rules. Based on the verification outcomes of the prototype, we conclude that it is possible to automatically generate an executable simulation model based on a specified conceptual model.

Requirement 8. *The tool should be executable on a default consultant's notebook.*

Our prototype has proven that a solution which supports both model development and execution can be run on a default notebook. Both the simulation components and the complete prototype were also developed on a standard notebook. We were finally able to execute a generated simulation model. However, during our verification session we found that executing the model was relatively slow. To increase the usability in a project situation, we suggest that further research focuses on improving the speed with which the models are executed. One possibility may be to rewrite the DEVSDSOL-library to interact more directly with the DSOL-library.

Requirement 9. *The modeling language should be extendible with additional modeling elements.*

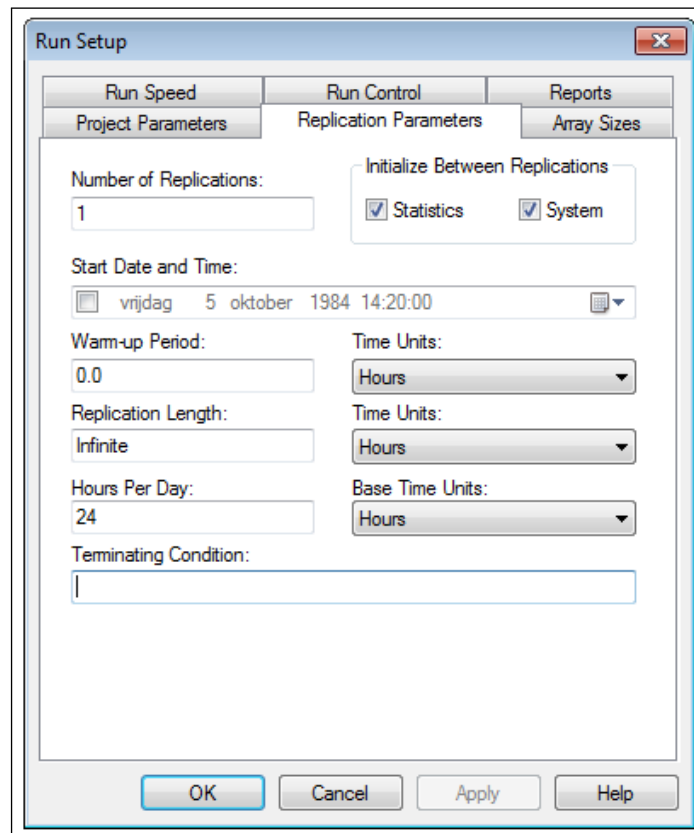


Figure 6.13: Arena example run set-up

The modeling language is defined through its metamodel which is shown in Figure 5.9. The simulation components corresponding to each modeling element have clearly defined interfaces and can be connected in numerous manners. We have not explicitly demonstrated that it is possible to extend the modeling language. However, in a few steps we believe it is possible: 1) A new modeling element is defined and added to the metamodel; 2) the implemented BPMN-metamodel and the model-builder are updated; 3) an atomic DEVS model corresponding to the modeling element is defined and implemented in DEVSDSOL; and 4) the ATL-transformation rules are updated.

Requirement 10. *The tool should enable conceptual model development and model execution in a multi-actor and multi-stakeholder context.*

In Section 5.5.2 we discussed the concept of collaborative modeling. Regarding the developed prototype, only one person can currently at a time interact with the tool. However it is possible that this tool is used in a group-setting, for instance if it is projected on a large screen. Although only one person (a facilitator) can interact with the tool, all other participants are able to contribute to the modeling and validation process.

6.7 Summary and conclusions

In this chapter we evaluated the design we proposed in Chapter 5 by developing a working prototype. We used the Model-Driven Development framework for Modeling and Simulation (MMD4MS) and the available Eclipse-based prototype to implement our solution. The included model builder enables the creation of business process models based on the earlier defined modeling elements. Each DEVS simulation component is implemented in Java to support easy construction of simulation models. The result is a library of DEVSDSOL-based simulation components. Based on the ATL-transformation language, a series of rules are written which allow for the (semi)-automatic generation of simulation models, based on a visually drawn conceptual model.

The prototype and functioning of it, as well as the implemented simulation components, are verified extensively. Several simulation models are developed and analyzed for correct behavior using DSOL. This resulted in strong confidence that the simulation components are correctly implemented and behave as expected. Also the modeling prototype itself functions adequately, able to transform a drawn business process model correctly into an executable simulation model. We confirmed this finally by modeling a real case, namely the service department's business processes of a large telecom operator. The resulting model contains 26 activities and various decisions, as well as different types of arriving and leaving entities and three types of resources. The prototype and implemented modeling elements proved to be able to represent the service department's business processes and to generate an executable DSOL simulation model.

Several evaluation sessions were held with consultants from Accenture to test the usability of the proposed solution. A think-aloud procedure was followed to discover any shortcomings, as well as possible improvements and future functionality. Although the prototype is currently not usable in a real client-situation (due to irregular software crashes), all participants were able to use the proposed modeling elements to model a sample business process. Based on returned questionnaires received from several involved consultants, we concluded that the new approach is intuitive and connects with the mental model of a consultant. Lastly, we evaluated the design requirements stated in Section 4 and confirmed that all requirements were met (some of which only partially, but this is due to the limited available time and means).

Conclusions, Reflection and Recommendations

7

With this chapter we will finalize our research by drawing conclusions whether we have achieved the original objective and more specifically how management consultants can easily model and simulate business processes. We will also place these conclusions in a larger, scientific context, namely how continuity can be achieved between conceptual model and simulation model development. In Section 7.1 we will first provide answers on the originally formulated research questions, which will lead to answering the main research question. In Section 7.2 we will reflect on our conclusions, as well as the research process itself. Finally, in Section 7.3, we will provide general recommendations for further research on the main topic of this research, as well as recommendations towards Accenture.

7.1 Conclusions

In Chapters 2 and 3 of this thesis we elaborated on the current way of how business processes are modeled as part of Accenture's consulting projects and how business process modeling and simulation are described in literature. These chapters provided us with the knowledge to answer the first research question:

1. What are the limitations of Accenture's current business process modeling and simulation approach and how does this relate to limitations mentioned in literature?

Business process simulation is within Accenture considered as a fruitful opportunity to evaluate current and new business process performance of client organizations. However, several simulation related projects were undertaken within Accenture, which resulted in the acknowledgment that current tools are not able to support management consultants in business process simulation. As the management consultants are not able to develop simulation models themselves (due to the difficulty of it), they are depending on a software programmer and his experience. The two main limitations of the current approach are 1) not having the possibility themselves to validate immediately a developed business process model through business process simulation, and 2) the dependency on one person with experience of computer programming and hopefully simulation model development. These limitations cause a project to be considered as inefficient, as well as inconvenient.

After studying various scientific publications about business process simulation, it became evident that the issue related to simulation model development as being difficult is widely recognized (besides several other related issues). It is one of the main reasons

why simulation is currently not as widely applied within the field of business process analysis, as researchers and simulation software developers might have hoped for.

2. What requirements should a new solution meet in order to support modeling and simulation of business processes by management consultants?

The main need of a management consultants is to have the possibility to 1) develop 'as-is' and 'to-be' business processes models of a client's organization easily, and 2) to execute these models using some software tool in order to simulate the business processes and analyze specified performance indicators and redesign scenarios. Three main factors are suggested in literature that influence the effectiveness of a decision support tool (which we consider a solution to these needs to be), namely *Usefulness*, *Usability* and *Usage*.

Regarding the usefulness, the main requirement is that a solution should finally be able to simulate the dynamic behavior of events and interactions within business processes. Regarding the usability, the end-user are the management consultants and thus to allow easy development by them of business process models, an intuitive approach to develop the models should be provided, as well as a modeling notation that connects with their mental model. As most management consultants have no programming experience, the main usability requirement of the solution is however to support the translation of a conceptual model into an executable simulation model. Finally, as development and validation of models takes place often at a client's location together with client-side specialist, the solution should be executable on a standard consultant's notebook and in a multi-stakeholder context.

3. What could be a new approach which enables business process modeling and simulation by management consultants?

To overcome the difficulty of simulation model development, we suggest a new approach based on Model-Driven Development. By formally specifying both a conceptual modeling language, as well as simulation model specification language, MDD allows to (semi)automatically transform a conceptual model into a simulation model. To enable this for management consultants, we defined a conceptual modeling language based on the Business Process Modeling Notation (BPMN) and we adapted it to connect with the consultant's mental model of business processes. This was accomplished through interactive design workshops with several management consultants from Accenture. Each modeling element defined by the language represents some aspect within a business process (e.g., activities and resources) and for each element a graphical representation was defined to allow visual business process model development.

The main requirement for the solution is to support the development of simulation models and indirectly to promote the continuity from conceptual to simulation model. A

component-based approach is considered to be a fruitful concept to increase the efficiency of simulation model construction. Corresponding to each defined modeling element, a DEVS-specified simulation component was designed (plus some additional required components), which can be coupled to form a simulation model. If formally specified, it also enables the application of MDD. To support the actual transformation from conceptual model to simulation model, we defined a set of transformation rules which describe how a drawn conceptual model can be transformed to construct a simulation model using the simulation components.

Management consultants are not interested in a simulation model of business processes an sich, but in the statistics generated during and after executing a simulation model. Statistics of interest relate amongst others to the utilization of resources; processing and waiting times of entities; average number of waiting entities within queues and average waiting times per queue. Depending on the statistic and goal of a simulation study, a consultant may want to have this presented in a box-and-whisker-plot, a time-dependent plot or in data-tables.

4. Does the proposed solution enable modeling and simulation of business processes by management consultants?

As a proof of concept of our designed solution for management consultants, we used a prototype based on the Model-Driven Development framework for Modeling and Simulation (MDD4MS). After implementing the metamodel of the conceptual modeling language, implementing the simulation components using the DEVSDSOL-formalism, and specifying ATL-transformation rules, we obtained a working prototype. This prototype allows us to develop business process models through a visual model builder and transform these into a simulation model which can be executed using the DSOL-simulation library.

We used the prototype to model and simulate part of a sample case, in order to evaluate: 1) the ability of the solution to allow the representation of business processes as how consultants would model these, and 2) whether an executable simulation model can be (semi)automatically generated based on this conceptual model. The business process model of the case was developed by management consultants for an actual consulting-project. Based on this evaluation, we conclude that modeling more elaborate business processes using the proposed modeling elements and applying a MDD-approach for transformation from conceptual to simulation model, seems fruitful. To evaluate the usability requirements regarding interaction and the intuitivity of the modeling approach, a usability testing workshop was held in which several management consultants used the prototype, and afterwards received a questionnaire. Based on the workshop, as well as the retrieved questionnaires, we conclude that the suggested solution is an easy and intuitive way for consultants to model business processes.

Main research question: *How can a support tool help management consultants in a new way to model and simulate business processes?*

The main objective of this research is stated in the introduction chapter of this thesis, namely to find a new way how a support tool can help management consultants to easily model and simulate business processes (see Section 1.3). We proposed a concept design of a solution which enables consultants to model and simulate business processes. By answering research question 3 we discussed the main concepts of this design. The evaluation of this concept, namely by developing a prototype and using this prototype to analyze a sample case and perform a usability test with management consultants, confirms that *the proposed design is indeed a fruitful possibility to enable business process simulation by management consultants*. A few side-notes can be placed next to this statement, but these will be discussed in Section 7.2.

The proposed design is a fruitful possibility to enable business process simulation by management consultants

By applying a user-centered design approach, the management consultant's understanding of and view on business processes is included in the proposed conceptual modeling language. This allows management consultants to develop business process models as how they 'see' the actual business processes. By applying a Model-Driven Development approach for the transformation of a conceptual model into a simulation model, we have successfully shown that based on a conceptual model a simulation models can (semi)automatically be generated. Thus, without programming knowledge nor simulation model development experience, a management consultant can be able to model and simulate business processes.

Contribution of this study to Accenture

This research and the outcomes are a contribution to Accenture in several ways. Firstly, in Chapter 2 we elaborated on the process of how simulation models are currently developed and what the limitations are of this process. Almost all knowledge describing the process as well as the limitations, was acquired through interviews with consultants who were involved in these projects, as almost no concrete written information was available. Secondly, by describing the mental model of consultants and implementing this in a modeling language, we have made the consultant's view on business processes explicit. Although Accenture provides several learning courses to support employees how to see business processes, this research may contribute to that as it is based on years of experience of the involved consultants. Thirdly, this research is a first step to finally a fully functional and usable modeling and simulation support tool for management consultants. This study provides a proof of concept that it is feasible to continue research on this topic, as well as a starting point for future studies (see Section 7.3.2 for recommendations on this). And finally, this study provides the argumentation, as well as confirmation by several experts from Accenture, for the usefulness of applying business process simulation as part of different types of projects within Accenture (e.g., operational excellence projects, team-sizing as part of organization design, . . . etc). Simulation of business processes may be used to append best-practice methods, to evaluate the influences of business process changes on the performance of organizations.

Scientific contribution

The main scientific contribution of this research is that we have shown that simulation model development based on a Model-Driven Development (MDD) approach indeed seems fruitful. By applying some concepts of the MDD4MS-framework as suggested in [CVS11] in our research, we realized a working prototype which enables conceptual modeling and simulation by management consultants from Accenture. Although this is a specific case, we assume that this solution will also be usable by management consultants from other firms as well as other business analysts, with a lack of programming and

simulation experience. Our main reasons for assuming this, is because 1) we based our solution on the mental model and experience of several management consultants with different backgrounds, knowledge and skills during the research, design and evaluation stages, 2) because we based the conceptual modeling language on BPMN, which is considered to be a standard for business process modeling throughout industries, and 3) we tend to believe that although management consulting firms may differ from each other regarding their corporate image and climate, their analysis approaches are based on similar principles and methodologies. However, we suggest further research to elaborate on this.

Another aspect which is also considered as a scientific contribution of this research, is the application of a user-centered design (UCD) approach for developing a modeling and simulation solution. Although our case focused specifically on management consultants from Accenture, we assume that applying a similar UCD-approach, in combination with prototyping and usability testing, for other industries with the desire for a usable modeling and simulation solution, will also be effective. Although there is no direct proof for this, we join the general opinion that involving end-users throughout the design process, will increase the likelihood of developing a usable solution.

7.2 Research reflection

In this section, we will first reflect in Section 7.2.1 on the chosen design science research approach, and evaluate whether this research can be considered as effective. In Section 7.2.2 we will reflect on the overall research by elaborating on the separate research and design steps made.

7.2.1 Reflection on design science research approach

Seven guidelines are formulated for effective design science research [HC10]. Each of these guidelines will now be discussed to evaluate whether this research was effective.

Design as an Artifact The main designed artifact of this research is the business process modeling and simulation solution for management consultants. We thus conclude that this guideline was fulfilled, as we developed a working prototype which proved that the solution is considered usable and useful.

Problem Relevance The solution addresses a relevant business problem, namely the lack of a tool which supports consultants in modeling and simulating business processes. From an academic point of view, we addressed a relevant problem, namely the difficulty of simulation model development and the lack of continuity between conceptual and simulation model development.

Design Evaluation The design was evaluated by developing a working prototype and performing both a case-study evaluation, as well as a usability test with it. These tests demonstrated that the requirements for utility, quality and efficacy were fulfilled.

Research Contributions A user-centered design (UCD) approach was chosen for the design process of a modeling and simulation solution for management consultants. We were unable to find academic literature referencing to the application of a UCD approach in similar studies. Also, we applied the MDD4MS-framework proposed in recent literature. The MDD4MS-framework has to our knowledge not been applied so far.

Research Rigor Several methodologies were applied during the construction and evaluation of the designed artifact, like for instance interviewing, prototyping and usability testing.

Design as a Search Process The final proposed design is the result of an active search process to find a solution for management consultants. Reading academic literature contributed to this quest by providing a better understanding on the principles of designing, as well as modeling and simulation. The design workshops with management consultants also greatly contributed to the search for and finding of a solution.

Communication of Research This research is part of obtaining the degree of M.Sc. and will be presented and publicly defended for a diverse audience.

7.2.2 Reflection on research stages

In this section we will briefly reflect on our research by analyzing the three main phases of this research: 1) Problem definition, 2) Solution design, and 3) Evaluation. . We will finalize with a personal reflection on the overall research.

7.2.2.1 Problem definition

Accenture During the problem definition phase, we interviewed management consultants from three service lines: Talent & Organization Performance (T&OP), Process & Innovation Performance (P&IP) and Supply Chain Management (SCM). This was done to understand better the current way of working within Accenture regarding business process modeling, as well as acquiring knowledge about past simulation projects. However, it would have been better to involve more consultants from different service lines to create a more complete overview. Choice for the selected consultants and service lines was mainly based on their availability and suggestions made by Accenture employees.

Literature The literature library we build up through out this research contains approximately 200 scientific articles, books and dissertations. Due to the scope of our research, a variety of different domains were researched (e.g., conceptual modeling, discrete-event simulation, design research, usability testing, prototyping, human-computer interaction, user-interface design, . . . etc). However, this is still a fraction of the research that is performed and documented during the past years.

One work not included so far in our research, but certainly of interest, relates to the development and implementation of both a DEVS-library, as well as an Arena and SIMAN library, for the Modelica simulation language [Pra10]. SIMAN is the simulation

language used in Arena simulation software. Reason for implementing the Arena libraries is to enable the combination of process-oriented simulation models with the rest of the available Modelica libraries. The research discussed in [Pra10] connects closely to our research, as we also used the DEVS-formalism to specify process-oriented simulation components, which are similar to Arena-components. However, our research differs from the Modelica-research: we defined a conceptual modeling language tailored to management consultants, whereas the Arena-components that can be used to build visually a simulation model, are too incompatible with the mental model of management consultants on business processes. However, further research may want to investigate the possibilities of the Modelica-language for a management consultant modeling and simulation solution.

7.2.2.2 Solution design

Consultants' mental model During the design phase of our research we tried to understand and represent the mental model of management consultants on business processes. However, even during the usability evaluation sessions we discovered new aspects related to the view of consultants on business processes. We concluded that it is difficult to describe something as abstract as a mental model on something like for instance business processes, in only a few sessions. More insight can be gained by continuing research on a solution and applying it also in a real project.

Modeling language and DEVS-components We only had the possibility to include two management consultants during the design workshops who already had experience with business process simulation (BPS). We think it would have been better to involve more consultants, also some with no experience of BPS. This would namely connect better with the concept of end-user involvement. However, due to the limited amount of time available, we chose deliberately for consultants with experience, as this would require less time to explain the basic principles of discrete-event simulation. Because this study can be considered more as a proof of concept, and a next step could be the involvement of more consultants.

Regarding the developed modeling language, as well as the designed (and implemented) simulation components, we tend to state that this is only a first step to allow for the representation of business processes based on the mental model of consultants. Although the modeling elements were adequate to represent the sample-case, application of the modeling elements in a real project will most likely lead to conclusions that more modeling elements are needed, or more functionality is required by the simulation components.

Furthermore, we had to make a trade-off regarding the simplicity of modeling business processes and the possible functionality of a solution. Although we specified and implemented various features mentioned during the design workshops, exceptions will almost always arise during a simulation project. However, expanding the solution with more and more features, may result in a lower usability experienced by the end-user. A consideration should be made in what context a solution will be used finally. If it will be used for modeling and simulation in a preliminary phase of a project, then simplicity of modeling with a limited set of features is probably preferred. However, if it will be

used elaborately during a project, more functionality is required (of course in combination with an appropriate level usability provided). Currently, the proposed modeling language is based on the basic set BPMN elements, which is easy to use and understandable by probably most stakeholders with some experience in business process modeling. However, by extending the proposed modeling language with more modeling elements from the BPMN specification, more complicated models can be developed. A possibility would be to make a distinction between two sets: one for quick and simple business process models, and one for complicated and more elaborate models.

Component-based solution We concluded that a component-based approach for simulation model construction was effective, especially in combination with a MDD-solution. An aspect which was largely neglected in this research, is researching the possibility of adding and modifying modeling elements, simulation components, and translation rules. The application of MDD and metamodel transformation allows this, but further research should clarify to what extent this should be supported by management consultants or software specialists.

Statistics and visuals Due to the limited amount of time available for this research, less attention was paid to the visualization and output presentation of statistics. One important aspect which relates to the statistics is: *how can management consultants cope with the large quantities of statistics generated?* As simulation model development is considered complicated, it is only the first main step in a simulation study. Experimentation with a simulation model, and the analysis and interpretation of generated statistics, is at least as important and difficult as the construction of a model (if not even more difficult). It is strongly advised that further research focuses on these aspects of BPS by management consultants.

Integration in process We suggested that collaborative modeling is an option for the usage of BPS in consulting project. This may also diminish the currently encountered issues in simulation projects, regarding the inconsistency of business process descriptions provided by different stakeholders within a client organization. However, for consultants to work with a modeling and simulation solution in a real project, requires a transformation of their current approach. Further research on this is suggested (within Accenture much knowledge is available regarding “change management”). Joining and observing management consultants (“fly-on-the-wall” principle) during a real project would have also contributed to understanding the context in which a final solution would be applied. Unfortunately, this was not possible.

7.2.2.3 Evaluation

Implementation of components We used the DSOL and the DEVSDSOL library for the implementation and simulation of the simulation components. Although we have verified the working of the components, we suggest that experts verify the implemented code as well, in order to optimize it. Also, the various states defined per component is adequate to simulate some business processes. However, this can most likely also be

furthermore improved. Also, of all the features mentioned during the design workshops, not all could have been implemented due to the limited available time. Most features are specified throughout this thesis, but the minutes in the appendix also include some functionality which was not explicitly discussed.

Final prototype We have proven that our solution can be implemented and that a prototype of this solution is functional. However, we consider the prototype not as a usable solution for management consultants yet. The execution of simulation models is namely rather slow and the model builder will crash frequently during model development. Also the interface can be greatly improved. Lastly, the Eclipse environment offered us all the functionality needed for a prototype. However, it is a rather advanced and complicated environment to work with by users with no advanced computer experience. We suggest that much research can be performed on all these aspects.

Verification of implemented components We verified the implemented modeling components by comparison with corresponding Arena simulation models. However, a question that can be asked is whether Arena is an adequate benchmark to compare our implementation with. It would be more appropriate to test the implemented components in a real business project for which enough data is available to evaluate the operational validity of the solution.

Also, the components were verified extensively throughout the implementation phase by walk-through: reading and interpreting each console output line generated by the components to confirm appropriate working of a model. An issue which was discovered while testing a model, related to a component calling the confluent-function of DEVSDSOL (when both an external-event as well as an internal- event occur at the same time instance). However, this should not have happened, as the internal-event of the component was scheduled some seconds later and only an external-event occurred at that time instance. Elaborate study of the components, as well as the DEVSDSOL-library, made us finally conclude that there was bug in the DEVSDSOL-code. However, the version of the DEVSDSOL-library we used was outdated and the bug was fixed in the most recent version.

Based on the verification of the components, we tend to believe that the implementations are correct. However, further research should elaborate on this, as well as the correctness of statistics generation.

Telecom operator case-study The case-model which was used as part of the evaluation process, was developed by management consultants of which some had simulation experience. Application of the modeling language in a real project by management consultants with no or little simulation experience, will provide more insight whether the elements and implementation are adequate.

Usability test The usability test could be considered more a test regarding the functionality of the solution, than an actual usability test. However, as a proof of concept it provided much information what can be improved and should be added in later versions.

A suggestion would be to involve an professional interface designer to test and improve more elaborately the usability.

7.2.2.4 Personal reflection on overall research

For the SEPAM M.Sc. graduation project a total amount of 6 months is prescribed. Finally, this research took almost 10 months, 4 months longer than planned. This is mainly caused by a slow start, slightly in the wrong direction. At the start of the project, too much emphasis was placed and time was spent on analyzing the process of simulation model development within Accenture. Although this was very helpful for understanding the processes and limitations of these projects, it was not the main objective of this research. With a bit of support by my supervisors, the main objective was re-stated and the last 5 months of this research were focused on the original objective.

Another remark can be placed regarding the implementation of the components. Defining the simulation components started after the first workshop held with management consultants. At the same time, I started implementing these in Java. This allowed me to validate the concept of the components and see whether some concepts were usable. Reason for immediately implementing the components directly, was also a lack of exact knowledge and experience on DEVS and states. By implementing the components, I learned to understand the concepts behind DEVS and was able to direct the design workshops better.

7.3 Recommendations

Many topics for further research have been mentioned through out this thesis, and most are included in the previous section (Section 7.2.2). In this chapter will present the main recommendations, both related to further research (Section 7.3.1), as well as for Accenture (Section 7.3.2).

7.3.1 Recommendations for further research

We applied the Model-Driven Development framework for Modeling and Simulation (MDD4MS) as suggested in [CVS11]. The tool architecture for a Modeling and Simulation Environment (MSE) involves several users, namely the *conceptual modeler*, the *simulation modeler*, the *simulation programmer*, and the *simulation expert*. The role of the conceptual modeler is mainly to prepare a simulation conceptual model (CM) and use it to model the problem owner's system. The role of the simulation expert is to analyze the results obtained after executing a simulation model. In our solution, we suggest the role of management consultants to be both the conceptual modeler, as well as the simulation experts,

By applying a user-centered design (UCD) approach through active end-user involvement (e.g., various design workshops, evaluations sessions, interviews), we were able to formalize part of the consultant's mental model on business processes in a conceptual modeling language. However, to our knowledge, no concrete frameworks or methodologies are available which guide the development of a conceptual modeling language

with the final purpose to apply metamodel-transformation to a simulation model. Our design process was partly a try-out of concepts taken from design sciences (e.g., UCD, usability testing), to realize a final solution. To increase the likelihood of applying the MDD4MS-framework effectively, we suggest further research to investigate and develop (semi)standardized ways for developing conceptual modeling languages. This can include steps to be undertaken by a researcher, and distinctions between conceptual models for different type of industries. Our research can contribute to it, as it describes the process and the outcomes how we finally realized a conceptual modeling language for management consultants.

Another recommendation for the further research relates to the implementation of a conceptual modeling language fully compatible with the BPMN 2.0 specification [Obj10]. We already reflected briefly in the previous section on the applicability of our designed modeling language. We assume that it will function adequately in the preliminary phase of simulation project, as well as to model relative simple business processes. Exceptions which can not be modeled using the proposed set will however most certainly appear, and thus we suggest that a more rigorous approach for conceptual modeling is necessary. We believe that BPMN could be that approach, as it is well-specified and is widely used to model complicated business processes. If fully compatible with the BPMN specification, it allows for specialist with BPMN knowledge to easily use it and simulate business processes. Also, importing existing BPMN-compatible models would be possible

However, such an endeavor will require much research and work. Although BPMN is well-specified, it contains many modeling elements and formal rules. Also, to allow execution of a BPMN-compatible business process model requires the development and implementation of the simulation components. Our research can function as a starting-point, but for instance the choice of placing Start Events and End Events outside swimlane is *not* compatible with BPMN (although it allows us nicely to use the boundaries of swimlanes to depict the request for and release of a resource).

Furthermore, we suggest that future research may focus on the evaluating and possibly improving the DSOL and DEVSDSOL-libraries. DSOL was originally presented in 2005 [Jac05], whereas the DEVSDSOL-library was presented in 2009 [SV09]. Our experience when executing generated simulation models was that it behaved rather slow. Although this may relate to our implementations and design choices, we believe that improvements can be made. The statistics output presentation is functional, but considered outdated and less user-friendly. Also, the DEVSDSOL uses parts of the DSOL-library, which apparently causes some lag in the execution of a simulation model.

7.3.2 Recommendations for Accenture

We propose several recommendations for Accenture future course regarding modeling and simulation for management consultants:

- Our research can be considered as a proof of concept for a modeling and simulation solution for management consultants. However, further research is suggested to finally realize a functional and usable tool. For improving the user-experience, as well as designing the user-interface, we would suggest to include industrial design experts. With regard to the software tool itself, a computer science expert can

possibly improve the implemented code of the components, as well as implement an integrated modeling and simulation environment. For the integration in the processes of consultants, we believe that “change management” as applied in the T&OP service line could contribute to this.

- Although we evaluated the prototype using a sample case, we believe that more testing is necessary to possibly identify shortcomings of the modeling language and the implemented simulation components. In a following development stage, an updated prototype should be used in a real consulting project by management consultants to evaluate to usability more thoroughly.
- Further research is also recommended on the applicability of business process simulation (BPS) within Accenture’s different service lines. As was mentioned earlier in this thesis, BPS is considered useful for determining and evaluation of team sizes in new or modified business processes. Also for operational excellence / lean six sigma projects, BPS may provide additional value. A suggestion could be to apply the solution on a past project of which both the business process descriptions are available, as well as the specification. This could allow for the evaluation whether the current approaches (e.g., best practices, spread sheet calculations) resulted in adequate solutions, or better solutions were possible.
- Finally, to improve the effectiveness of future simulation related projects, we recommend to set-up a permanent team of management consultants who are specialized in modeling and simulation (M&S). During the problem definition phase of this research, we identified several issues which relate to the composition of simulation project teams and the experience of involved management consultants regarding simulation. Currently, simulation projects are not frequently undertaken by Accenture. The members of the M&S team could remain part of their original service line or industry-group, when no simulation projects are available. However, when a simulation related project is accepted, these management consultants will form the team responsible for the project. This will provide knowledge to Accenture about effectively undertaking simulation projects in a continuous manner, as a permanent group of consultants is in general responsible for M&S projects and can be approached for their experience and knowledge.

Bibliography

- [Acc11a] Accenture[2], *Company Description*, Retrieved March 10, 2011, from <http://www.accenture.com/us-en/company/overview/description/Pages/index.aspx>, 2011.
- [Acc11b] Accenture[3], *Reference Guide: Business Process Modeling Deliverables*, Retrieved March 23, 2011, from https://mylearning-products.accenture.com/Business_Process_Design_Fundamentals/m03/0301020202.htm, 2011.
- [Acc11c] Accenture[4], *Reference Guide: Business Process Modeling Standards and Deliverables*, Retrieved March 23, 2011, from https://mylearning-products.accenture.com/Business_Process_Design_Fundamentals/m03/03010202.htm, 2011.
- [AS04] Ruth Sare Aguilar-Savén, *Business Process Modelling: Review and Framework*, International Journal of Production Economics **90** (2004), no. 2, 129–149.
- [Ban98] Jerry Banks, *Handbook of Simulation - Principles, Methodology, Advances, Applications, and Practise*, John Wiley & Sons, Inc., New York, 1998.
- [Bar02] Carol M. Barnum, *Usability Testing and Research*, Pearson Education, Inc., 2002.
- [Bot87] Pieter W.G. Bots, *An Environment to Support Problem Solving*, Ph.D. thesis, TU Delft, 1987, pp. 225–231.
- [BVCH07] Vesna Bosilj-Vuksic, Vlatko Ceric, and Vlatka Hlupic, *Criteria for the evaluation of business process simulation tools*, Interdisciplinary Journal of Information, Knowledge, and Management **2** (2007), 73–88.
- [Cet10] Deniz Cetinkaya, *PhD Research Proposal: Model Driven Development of Hierarchical Simulation Models*, Ph.D. thesis, Delft University of Technology, 2010, p. 73.
- [Coo07] Alan Cooper, *About Face 3.0: The essentials of interaction design*, 3rd ed., Wiley Publishing, Inc., 2007.
- [CVS10] Deniz Cetinkaya, Alexander Verbraeck, and Mamadou D. Seck, *Applying a Model Driven Approach to Component Based Modeling and Simulation*, Proceedings of the 2010 Winter Simulation Conference (Baltimore, MD) (B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, eds.), no. Robinson 2006, 2010, pp. 546–553.

- [CVS11] ———, *MDD4MS : A Model Driven Development Framework for Modeling and Simulation*, Proceedings of the Summer Computer Simulation Conference 2011, 2011.
- [dJ11] Maarten de Jong, *Interview with Maarten de Jong - 2011-01-18*, 2011.
- [DS90] T.H. Davenport and J.E. Short, *The new industrial engineering: information technology and business process redesign*, Sloan Management Review **31** (1990), no. 4, 11–27.
- [DVvE03] Gert-Jan De Vreede, Alexander Verbraeck, and Daniel T.T. van Eijck, *Integrating the Conceptualization and Simulation of Business Processes: A Modeling Method and Arena Template*, Simulation **79** (2003), no. 1, 43–55.
- [Ecl11] Eclipse, *ATL/User Guide - Introduction*, Retrieved June 30, 2011, from http://wiki.eclipse.org/ATL/User_Guide, 2011.
- [FMS09] Kathrin Figl, Jan Mendling, and Mark Strembeck, *Towards a Usability Assessment of Process Modeling Languages*, Proceedings of the 8th Workshop Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (EPK 2009) (Berlin, Germany), 2009, pp. 118–136.
- [FW96] Johanna Fullerton and Michael A. West, *Consultant and client - working together?*, Journal of Managerial Psychology **11** (1996), no. 6, 40–49.
- [Gal07] Wilbert O Galitz, *The Essential Guide to User Interface Design*, 3rd ed., Wiley Publishing, Inc., Indianapolis, US, 2007.
- [GT94] Bruce Galdwin and Kerim Tumay, *Modeling Business Processes with Simulation Tools*, Proceedings of the 1994 Winter Simulation Conference (S. Manivannan, ed.), vol. 1994, 1994, pp. 114–121.
- [GW04] Andrew Gemino and Yair Wand, *A framework for empirical evaluation of conceptual modeling techniques*, Requirements Engineering **9** (2004), no. 4, 248–260.
- [Har91] H. James Harrington, *Business Process Improvement: The Breakthrough Strategy for Total Quality Productivity*, McGraw Hill, New York, 1991.
- [HC93] Michael Hammer and James Champy, *Reengineering the Corporation*, Harper Collins Books, New York, 1993.
- [HC10] Alan R. Hevner and Samir Chatterjee, *Design Research in Information Systems*, Information Systems Research **22** (2010), 9–22.
- [HD05] Vlatka Hlupic and Gert-Jan De Vreede, *Business process modelling using discrete-event simulation: current opportunities and future challenges*, International Journal of Simulation and Process Modelling **1** (2005), no. 1, 72–81.

- [HF96] Charles R. Harrell and Kevin C. Field, *Integrating Process Mapping and Simulation*, Proceedings of the 1996 Winter Simulation Conference (J.M. Charnes, D.J. Morice, D.T. Brunner, and J.J. Swain, eds.), 1996, pp. 1292–1296.
- [HMPR04] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram, *Design Science in Information Systems Research*, Management Information Systems **28** (2004), no. 1, 75–105.
- [HR98] Vlatka Hlupic and Stewart Robinson, *Business process modelling and analysis using discrete-event simulation*, Insight, Proceedings of the 1998 Winter Simulation Conference, 1998, pp. 1363–1369.
- [IH01] Melody Y. Ivory and Marti A. Hearst, *The state of the art in automating usability evaluation of user interfaces*, ACM Computing Surveys **33** (2001), no. 4, 470–516.
- [Jac05] Peter H.M. Jacobs, *The DSOL simulation suite*, Ph.D. thesis, Delft University of Technology, 2005.
- [Jan11] Glenn Janssen, *Interview with Glenn Janssen - 2011-01-12*, 2011.
- [KS08] Peter G.W. Keen and Henk G. Sol, *Decision Enhancement Services - Rehearsing the Future for Decisions That Matter*, IOS Press, Amsterdam, 2008.
- [Kul96] Jasna Kuljis, *HCI and Simulation Packages*, Proceedings of the 1996 Winter Simulation Conference (Coronado, California, US) (J.M. Charnes, D.J. Morrice, D.T. Brunner, and J.J. Swain, eds.), 1996.
- [Lar08] Maria Dolores Proano Lara, *Visual layout for drawing understandable Process Models*, M.sc. thesis, Technische Universiteit Eindhoven, 2008, p. 102.
- [LG86] L. Larwood and U.E. Gattiker, *Client and consultant managerial problem-solving values*, Group and Organization Studies **11** (1986), no. 4, 374–86.
- [LK00] Averill M. Law and W. David Kelton, *Simulation Modeling and Analysis*, McGraw Hill, Singapore, 3rd ed., 2000, pp. 292–397.
- [LM08] Tomaž Lukman and Marjan Mernik, *Model-Driven Engineering and its introduction with metamodeling tools*, October (2008), no. October, 1–6.
- [Ma01] Nuno Melão, *Improving the effectiveness of business process modelling and simulation*, Dissertation, Lancaster University, UK, 2001.
- [Mag01] Martin Maguire, *Methods to support human-centred design*, International Journal of Human-Computer Studies **55** (2001), no. 4, 587–634.
- [MaP03] Nuno Melão and Michael Pidd, *Use of business process simulation: A survey of practitioners*, Journal of the Operational Research Society **54** (2003), no. 1, 2–10.

- [MR08] Michael Muehlen and Jan Recker, *How much language is enough? Theoretical and practical use of the business process modeling notation*, Advanced Information Systems Engineering, Springer, 2008, pp. 465–479.
- [Nan95] Richard E. Nance, *Simulation Programming Languages: An Abridged History*, Proceedings of the 1995 Winter Simulation Conference (Arlington: VA, USA) (C Alexopoulos, K Kang, W R Lilegdon, and D Goldsman, eds.), IEEE, 1995, pp. 1307–1313.
- [Nie93] Jakob Nielsen, *What is Usability?*, Usability Engineering, Morgan Kaufmann, San Francisco, 1st ed., 1993, pp. 23–48.
- [NJ82] Justus D. Naumann and A. Milton Jenkins, *Prototyping: The New Paradigm for Systems Development*, MIS Quarterly **6** (1982), no. 3, 29.
- [Obj08] Object Management Group Inc.[2], *BPML*, Retrieved June 12, 2011, from www.bpmi.org/, 2008.
- [Obj10] Object Management Group Inc., *BPMN v2.0 Specification Document*, 2010, p. 520.
- [Obj11] ———, *"Unified Modeling Language" Superstructure 2.4 Beta 2 - March 2011*, Tech. report, 2011.
- [Peg10] C Dennis Pegden, *Advanced Tutorial: Overview of Simulation World Views*, Proceedings of the 2010 Winter Simulation Conference (B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, eds.), IEEE Computer Society Press, 2010, pp. 210–215.
- [Pet81] James Lyle Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice Hall, New Jersey, 1981.
- [Pid02] Michael Pidd, *Simulation software and model reuse: a polemic*, Proceedings of the 2002 Winter Simulation Conference (2002), 772–775.
- [Pid04a] ———, *Computer Simulation in Management Science*, 5th ed., John Wiley & Sons, Inc., West Sussex, England, 2004.
- [Pid04b] ———, *Simulation Worldviews - So What?*, Proceedings of the 2004 Winter Simulation Conference, 2004. (2004), 280–284.
- [Pra10] Victorino Sanz Prat, *Hybrid System Modeling Using the Parallel DEVS Formalism and the Modelica Language*, Ph.d. dissertation, Universidad Nacional de Educación - Madrid, August 2010, p. 282.
- [RBKZ11] Stewart Robinson, Roger Brooks, Kathy Kotiadis, and Durk-Jouke Van Der Zee (eds.), *Conceptual modeling for discrete-event simulation*, CRC Press, 2011.

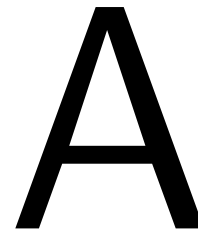
- [RHVM06] Nick Russell, Arthur H M Hofstede, Wil M P Van Der Aalst, and Nataliya Mulyar, *Workflow Control-Flow Patterns: A Revised View*, Tech. report, 2006.
- [RKD08] Michiel Renger, Gwendolyn L. Kolfschoten, and Gert-Jan De Vreede, *Challenges in collaborative modelling: a literature review and research agenda*, International Journal of Simulation and Process Modelling **4** (2008), no. 3/4, 248.
- [RL04] H Reijers and S Limanmansar, *Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics*, Omega, The International Journal of Management Science **33** (2004), no. 4, 283–306.
- [Rob04] Stewart Robinson, *Simulation: The Practice of Model Development and Use*, John Wiley & Sons, Inc., West Sussex, England, 2004.
- [Saf06] Dan Saffer, *Designing for Interaction: Creating Smart Applications and Clever Devices*, Peachpit Press, 2006.
- [Sar04] Robert G. Sargent, *Verification and Validation of Simulation Models*, Proceedings of the 2004 Winter Simulation Conference (WSC04) (New York) (R. Ingalls, M. Rossetti, J. Smith, and B. Peters, eds.), ACM, 2004, pp. 17–28.
- [Sha75] Robert E Shannon, *Systems simulation: the art and science*, Prentice-Hall, Englewood, Cliffs, New Jersey, 1975.
- [SM01] Alec Sharp and Patrick Mcdermott, *Workflow Modeling: Tools for Process Improvement and Application Development*, Artech House, Inc., Boston, 2001.
- [Sol82] Henk G. Sol, *Simulation in information systems development*, Dissertation, University of Groningen, 1982, p. 221.
- [SV09] Mamadou D. Seck and Alexander Verbraeck, *DEVS in DSOL: Adding DEVS Operational Semantics to a Generic Event-scheduling Simulation Environment*, Proceedings of the Summer Simulation Multiconference (Istanbul, Turkey), 2009.
- [Tum95] Kerim Tumay, *Business process simulation*, Proceedings of the 1995 conference on Winter simulation conference on Winter simulation, IEEE Computer Society, 1995, pp. 55–60.
- [VADH03] Wil M P Van Der Aalst, L. Aldred, M. Dumas, and Arthur H M Hofstede, *Design and implementation of the YAWL system*, Tech. report, Center for IT Innovation - QUT, Brisbane, Australia, 2003.
- [Van00] Hans L.M. Vangheluwe, *DEVS as a common denominator for multi-formalism hybrid systems modelling*, CACSD. Conference Proceedings. IEEE International Symposium on Computer-Aided Control System Design, IEEE, 2000, pp. 129–134.

- [Van02] ———, *Discrete Event Modelling and Simulation*, COMP522 Mc Gill's Computational Science and Engineering programme, vol. 15, March 2002, pp. 1–16.
- [VD98] Daniel T.T. Van Eijck and Gert-Jan De Vreede, *Simulation support for organizational coordination*, Proceedings of the 1998 Hawaiian Conference of Systems Sciences (Los Alamitos) (J. Nunamaker, ed.), vol. 1, IEEE Computer Society, 1998, pp. 633–642.
- [Ven06] John R. Venable, *The role of theory and theorising in design science research*, Proceedings of DESRIST (2006), 24–35.
- [VHAR10] Wil M P Van Der Aalst, Arthur H M Hofstede, Michael Adams, and Nick Russel, *Modern Business Automation Process - YAWL and its Support Environment*, Springer-Verlag, Dordrecht, London, New York, 2010.
- [VLM02] Hans L.M. Vangheluwe, Juan De Lara, and Pieter J. Mosterman, *An Introduction to Multi-Paradigm Modelling and Simulation*, Proceedings of the AIS'2002 Conference (AI, Simulation and Planning in High Autonomy Systems (Lisboa, Portugal) (Fernando J. Barros and N. Giambiasi, eds.), 2002, pp. 9–20.
- [VV02] E.C. Valentin and Alexander Verbraeck, *Guidelines for designing simulation building blocks*, Proceedings of the Winter Simulation Conference (2002), 563–571.
- [WDDR06] Wei Wang, Hongwei Ding, Jin Dong, and Changrui Ren, *Comparison of Business Process Modeling Methods*, IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI '06) (2006), 1136–1141.
- [Wes07] Mathias Weske, *Business Process Management: Concepts, Languages, Architectures*, Springer-Verlag, New York, 2007.
- [Whi04] Stephen A White, *Process Modeling Notations and Workflow Patterns*, BP-Trends (2004), no. March 2004, 1–24.
- [You04] Ralph R. Young, *The Requirements Engineering Handbook*, Artech House, Inc., Norwood, MA, USA, 2004.
- [ZPK00] Bernard P Zeigler, H. Praehofer, and T.G. Kim, *Theory of Modeling and Simulation*, second edi ed., Academic Press, 2000.

Part II

Appendix

Responsibility assignment matrix for BPDST projects



An overview is shown in Figure A.1 of the different Accenture-side and client-side roles during the development of a business planning decision support tool, also known as the responsibility assignment matrix. Purpose of the Responsibility Assignment Matrix is to visualize the roles and responsibilities of stakeholder in a project.

		Accenture-side				Client side				
		Project Manager	Process consultant	Data consultant	Software specialist / programmer	Business Unit Manager (BUM)	Sub-Unit Business Manager (SUBM)	Specification / data supplier	Task executor (workfloor employee)	End-user BPDST
Business process model	Mapping business processes	I	R/A	C	O	C	C	O	C	O
	Developing business process model	I	R/A	C	O	C	C	O	C	O
	Validating process model	I	R/A	C	O	C	C	O	C	O
Specified Business process model	Collecting data and assumptions	I	C	R/A	O	O	O	C	C	O
	Specifying business process model	I	C	R/A	O	O	O	C	C	O
	Validating data and process model	I	C	R/A	O	C	C	C	C	O
DSOL Simulation model	Developing DSOL Simulation model	I	O	O	R/A	O	O	O	O	O
	Verification simulation model	I	C	C	R/A	O	O	O	O	O
	Operational validation simulation model	I	C	R/A	C	C	C	C	O	O
Business Planning Decision Support Tool	Developing I/O Interfaces with users and systems, and testing	I	C	C	R/A	O	O	O	O	C
Model analysis and experimentation	Preparing simulation model for analysis	I	C	C	R/A	O	O	O	O	O

Figure A.1: Responsibility assignment matrix of

Explanation Responsibility assignment matrix R = Responsible; A = Accountable; C = Consulted; I = Informed; O = Omitted

Responsible Those who do the work to achieve the task. There is typically one role with a participation type of Responsible, although others can be delegated to assist in the work required (see also RASCI below for separately identifying those who participate in a supporting role).

Accountable The one ultimately accountable for the correct and thorough completion of the deliverable or task, and the one to whom Responsible is accountable. In other words, an Accountable must sign off (Approve) on work that Responsible provides. There must be only one Accountable specified for each task or deliverable.

Consulted Those whose opinions are sought; and with whom there is two-way communication.

Informed Those who are kept up-to-date on progress, often only on completion of the task or deliverable; and with whom there is just one-way communication.

Omitted Designating individuals or groups who are specifically not part of the task. Specifying that a resource does not participate can be as beneficial to a task's completion as specifying those who do participate.

Source: http://en.wikipedia.org/wiki/Responsibility_assignment_matrix

B

Expert-interviews

- Interview with G. Janssen (T&OP service Line consultant), 2011-01-12, 11:00 - 12:00,
- Interview with M. de Jong (P&IP service Line consultant), 2011-01-18, 10:00 - 11:00,
- Interview with M. van Loo (SCM service Line consultant), 2011-02-22, 10:00 - 11:00,

Due to confidentiality reasons, these interviews are not included in the public version of this thesis

Background on simulation

In this appendix, two simulation life cycles are show, as proposed in [Ban98] (see Figure C.1), and in [Sha75] (see Figure C.2).

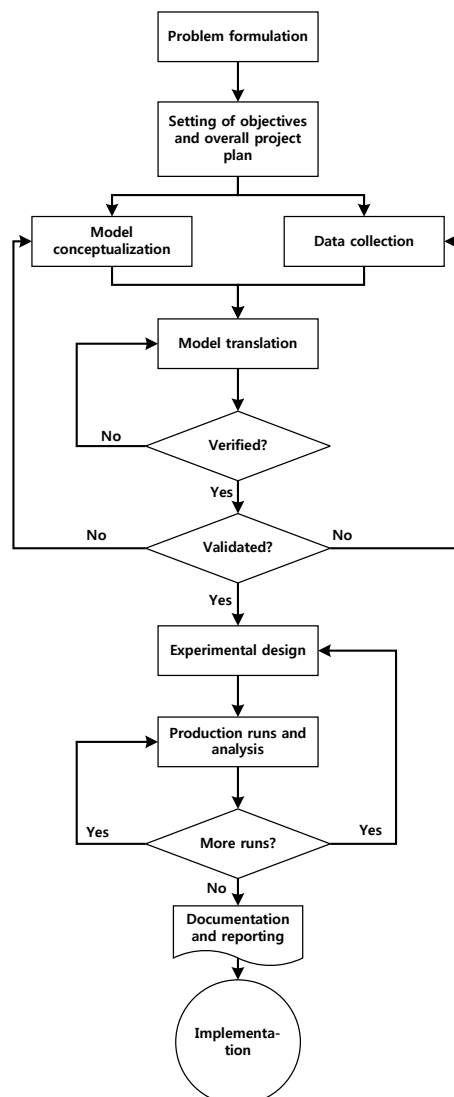


Figure C.1: Steps in a simulation study according to [Ban98]

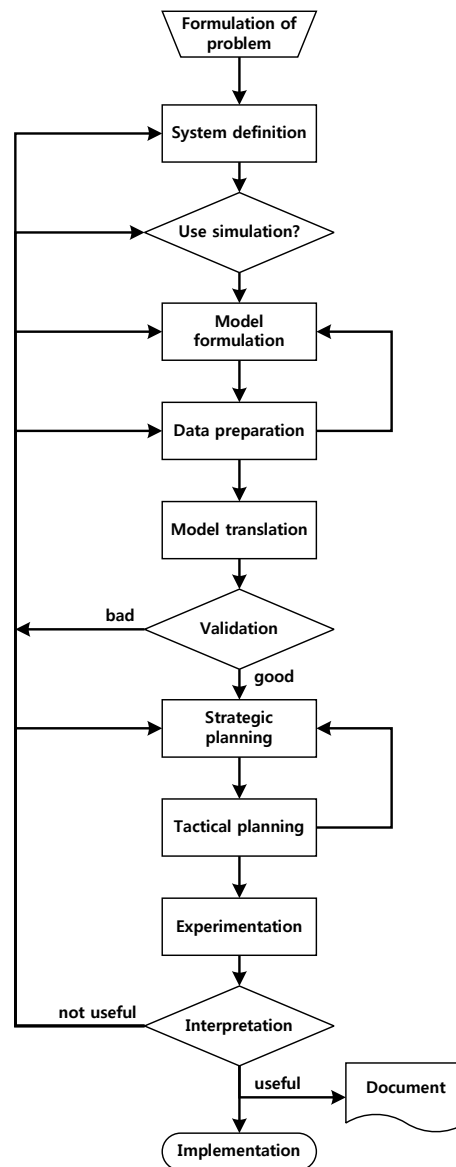
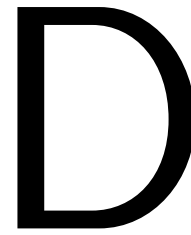


Figure C.2: Steps in a simulation study according to [Sha75] (adopted from [Cet10])



Minutes of design sessions

D.1 Design session 1: March 28, 2011

Goal and purpose of session

Two main goals for this design session were stated:

1. Gaining insight in process of mapping, modeling and simulation by consultants
2. Discussion about and selection of BPMN / simulation components

The plan of the workshop was as follows:

- Selection of components was provided, as well as a sample case (based on original BPDST case)
- Model a sample case using the components, by gathering process descriptions as how it would be done at a client's side (roles of process mapping consultant, data gathering consultant, business unit manager).
 - Client-side interaction process cannot be represented fully correctly (due to that already the process model is known on beforehand)
- Have a discussion afterwards about the components, their specification, and the modeling for simulation approach

Prepared material

1. Microsoft PowerPoint presentation
2. Sample case
3. A5-sized sheets of paper with printed modeling elements (based on BPMN and Arena blocks)

Outcomes

Evaluation session 28-03-2011 – outcomes and findings
“Modeling for simulation and modeling components”

What was the plan of the workshop?

- Selection of components was provided, as well as a sample case (based on original BPDST case)
- Model a sample case using the components, by gathering process descriptions as how it would be done at a client's side (roles of process mapping consultant, data gathering consultant, business unit manager).
 - Client-side interaction process cannot be represented fully correctly (due to that already the process model is known on beforehand)
- Have a discussion afterwards about the components, their specification, and the modeling for simulation approach

Was this plan followed?

- Partly
 - Discussion immediately started about the first component, the start event and how to specify arrival times, batch size, etc
 - Next, the discussion continued about adding a task, and how a resource is specified and associated with a task

What were the findings/ questions / etc?

- What is a global variable [during the introduction]?
 - It is confusing what a variable is, is it part of the system?
 - No, it's with regard to the simulation model / statistics
- **Start-event “Letter arrivals”**
 - Specification: commonly used is “number per time-unit”
 - Batch-size vs. 1 entity arrives
 - Clear distinction between items that arrive separately, or that arrive as batches
 - Distinction between parameters that are essential to be entered, and that are optional (basic / advanced)
 - Basic: Entity type; Inter-arrival times vs. batch arrivals per time unit; batch size (default: 1)
 - Advanced: Maximum number of batches that arrive
 - Schedule (poses a constraint on a task / resource): to define the availability of a resource, or when can a task be done?
- **Task “Read letter”**
 - General discussion “Duration of a task”, instead of “Delay duration”
 - Where to specify the duration? At the task itself? Or per resource (which is according to Ivo / Rutger more logical)
 - Defined by: constant; min-max (uniform distribution); average; other distributions (like for instance exponential, pois, weibull, normal, etc)
 - Consultant make first a choice of the type of distribution / duration specification
 - Example: what if task will be processed the next day
 - Every task is associated with a resource
 - “A task is a box where items collect which should get processed”
 - “Task is a coupling between a resource and an entity”

- Task parameters:
 - Duration defined at resource
 - Exit logic: does an processed entity leave directly, or wait until certain moment or event?
- Difference between tick of clock: continues vs. discrete
- **Resources**
 - A separate block with “Claim resource” or “Release resource” is difficult to explain to a client
 - Swimlanes concept seems better / more communicable
 -

Main findings

- Focus is on resources!
 - Task durations depends often on which employee it actually does, so there should be a separate definition of these
 - Example of idea of “Resources definition table / approach”:

<u>RESOURCE</u>	Productivity	Capacity	Priority
<u>B</u>			
Answer	4 min (expo)	1 FTE	2
Call	3 min (expo)		1
Mail	10 min (expo)		3

<u>RESOURCE</u>	Productivity	Capacity	Priority
<u>C</u>			
Answer	2 min (expo)	0.5 FTE	1
Call	6 min (expo)		3
Mail	5 min (expo)		2

Design choices

The following explicit design choices were made during this session:

- Resources are represented through swimlanes in a conceptual model
- An entity entering a swimlane through a flow (connector) depicts the claim of a needed resource. If no resource is available to handle the entity, the entity waits in a queue until a resource becomes available. The queue is placed between the swimlane boundary and the first model element.
- An entity leaving a swimlane through a flow (connector) depicts the release of a needed resource.
- A resource has the following attributes: Capability, Productivity, Capacity, Priority

D.2 Design session 2: May 10, 2011

Goal and purpose of session

Goal of this workshop was to discuss the following topics:

- Validation previous concepts
- Prioritization
 - What do you want to have included (for now)? What is most important?
 - Decide (mention) per topic the priority: Should be included! Can be include; Not necessary for now
- Goals of notation
 - To make the modeling process by consultants easy and understandable
 - To keep the model understandable (readable) by consultants and clients
 - To provide features and functionality that are common in most business processes
 - **Consideration between functionality and process (easiness)**

Outline of the workshop:

1. Design choices: Resources, Roles and Tasks
 - Priorities; Handoffs (from one Resource to another); Additional Resources; FTEs and Schedules; Resource Specification Sheet
2. Design choices: Pools, Events, Delays
3. Naming of elements
4. Routings and Patterns
5. Preview workshop 3

Prepared material

1. Microsoft PowerPoint presentation
2. Sample case in PowerPoint Slides (see figures)

Design choices: Competing Items

Priority assign options:

Option 1	Option 2	Option 3	Option 4	Option 5
Duration Task 1: 5 min Task 2: 8 min	Role Priorities Role A: low (3) Role B: high (1)	Task Priorities Task 1: low (3) Task 2: high (1)	Longest waiting item Item @ Role A: is waiting for 0 min Item @ Role B: is waiting for 23 min	In case even more items are waiting Items waiting Role A: 5 items Role B: 16 items

Multiple tasks? Sequence flows entering a swimlane half-way? Next slide.

TU Delft Workshop 2 6

Design choices: Competing Items

My suggestion:

1. First priority order: between roles
Role A: high
Role B: medium
Role C: low
2. Second priority order: waiting roles
Role A: (3) → (2) → (1)
Role B: (2) → (3)
Role C: (1)

TU Delft Workshop 2 8

Design choices: Logic in Swimlanes

Annotations:

- Item crosses boundary of Role A swimlane (crosses Task 1 swimlane, without creating a new one). Resource with Role A is still attached with this item.
- Resource with Role A finished Task 1 and needs to hand the item over to Resource with Role B.
- The Sequence flow crosses a boundary again, which means that the Resource is released, and one with role A is requested.
- The Sequence flow crosses a boundary which means that the previous Resource with Role A should be released, and one with Role B should be requested.
- An option would be to check whether the previous Resource has finished Role B also Role A, in that case the item is not released and forwarded immediately to Task 3.

TU Delft Workshop 2 12

Design choices: Logic in Swimlanes II

Annotations:

- Queue for Resource with Role A
- Release Resource
- Queue for Resource with Role B
- Release Resource
- Release Resource

TU Delft Workshop 2 13

Schedules and FTE: Employees

Annotations:

- Nothing happens in the swimlane

TU Delft Workshop 2 23

Resources Specification Sheet

Resource Specification Sheet

Step 1: Create Resource and specify type, capacity, associated role

Resources	Type	Capacity (FTE)	On schedule based	Machine?	Role A	Role B
Resource 1	Fixed capacity	2		No	Yes	Yes
Resource 2	Fixed capacity	2		No	Yes	Yes
Resource 3	Fixed capacity	2		No	Yes	Yes
Machine 1	Fixed capacity	2		Yes		

Step 2: Specify resources

Resources	Task 1	Task 2	Task 3
Resource 1	5 min	5 min	
Resource 2	5 min	5 min	
Resource 3			5 min

Step 3: (Optional): Specify machine follows

Machines	Follows
Machine 1	Specified follows
Machine 2	Specified follows

TU Delft Workshop 2 28

Outcomes

EVALUATION SESSION 2 – MAY 10, 2011 – OUTCOMES

“MODELING BUSINESS PROCESSES FOR SIMULATION”

WORKSHOP CHARACTERISTICS

Attendees: Ivo Wenzler and Rutger Deenen (Accenture)
Total duration 2,5 hours

WHAT WAS THE PLAN OF THE WORKSHOP?

- Validation previous concepts
- Assigning importance to modeling/simulation elements
- Discussion and making design choices / getting agreement about the following topics
 - Resources, Roles and Tasks (priorities, handoffs, additional resources, FTE's and Schedules)
 - Pools, events, delays
 - Naming of elements
 - Routings, Gateways and patterns

OUTCOMES

- “Keep in mind: what is the purpose of the (simulation)-models?”
 - Calculating / analyzing / assigning capacity
 - Quick modeling at client-site: for real complexity later: still programmer's support
 - Don't focus on micro-management: plans are calculated possibly for a year

Competing Items

- With regard to the different “competing item”-priority rules, when multiple entities wait in different queues
 - Margins (effect of a certain priority rule) becomes less significant (smaller) when you calculate a plan for a year
 - “Option 1: task duration” is not a priority option, neither is “Option 4: Longest waiting Item”
 - **These are the priority options:**
 1. **Random:** just pick an entity from a queue
 2. **Patterns:**
 - 1-by-1: one from first queue, one from second queue, and so on
 - batch: 5 from first queue, 5 from second queue, and so on
 - *still needs choice which is the first queue, which is the second...*
 3. **Entity type / attribute:** e.g. phone call goes before email (urgent email vs. low priority email)
 4. **Prioritization:** “waiting time”; “importance”; “number of items”
 - *This should be worked out more, and formalization should be validated again*
 - How to take an item from a queue: FIFO (First In First Out) / LIFO
 - Decision logic in role / resource
 - What is the logic?
 - Information is within the entity



...

...



- Priority suggestion: First among roles, than among queues.
 - Why between roles? Not necessary anymore
- Roles only have 1 resource type attached and a resource can only have 1 role associated to him
 - "Roles (swimlanes) are containers of resources" (e.g. Role A: 3 FTE, Role B: 6 FTE)

Competing resources and prioritization



- This appears in business processes: for instance consultant vs. analyst doing a certain task
 - But (for now) it's not significant to include
 - 1 role has namely the same type of resources



- Logic in Task or Swimlanes
 - "Elegant modeling": model it how it is in reality
- Assign duration to Tasks, and not in the resources per task (*different to workshop1*)
 - Like how it was done in KPN model

Additional resources



- Distinction between whether an Entity is needed or a resource (person, machine)
 - Increase clarity between entities, resources, and consumable resources!
- E.g. a mechanic needs a modem to be able to install a connection
 - The modem, is a needed resource, but in this case it is an entity
- Every resource type has its own swimlane
 - But what about the machines?
 - Machines don't have their own swimlanes / role
 - Machine specification like Arena (per Task)?
- In case of people:
 - Person needed for a task? Duplicate / spread over multiple swimlanes...
 - Or: New component, attached to components that need an additional resource?
- Specific questions-slide in presentation: answer these myself

...

Schedules / FTE



- 1 FTE is a full time employee, working a normal week (40 hours, or 36h)
 - The exact amount of hours is possible to be specified by the modeler (run setup)
 - In case of a 36h workweek, the Friday afternoon is free
 - This can be modeled, using schedules, but left open for later implementation
- 0,5 FTE (part-time employee), not to be included (for now)
- Machines are available during the same time as employees

Pools and Swimlanes



- According to BPMN, a swimlane should be placed insight a pool. A pool may contain several swimlanes.
- For this modeling approach, the choice is made to only use swimlanes to depict roles of resources.

Delays

- In case an Entity needs to be hold for a certain amount of time, the task is placed outside the swimlane, so it is clear that no resource is attached during the delay-time



Names

- **Agreed upon:**
 - Resource
 - Entity (instead of Item)
 - Queue
 - Gateway (with different variants)
 - Flow (instead of Sequence Flow)
- Give names according to what they “do”, to make it clear to the modeler
 - Duplicate, split, decide, etc

Routings: exclusive gateway (decide)

- Enter values through component-dialog
 - Possibly last value is automatically calculated to add up to 100%
- Values appear on exit-flows of decide

Assigning attributes, Entity types, logic

- *Consideration, between making all logic visible, or to keep it simple*
- Option1: an element with the assign functionality
- Option2: a task expanded with the functionality,
 - it should be (visually) clear, that the task has extra functionality
- A task changes the entity type: after a Task is performed on an entity, this entity will be different.

Multi-choice gateway

- Split it up as much as possible: keeping decisions / logic “pure”

D.3 Design session 3: May 24, 2011

Goal and purpose of session

- Discussion previous discussed concepts
- Entities, Sub processes, Parallel activities
- 2 modeling evaluation with paper prototype

Preparation and material

1. Microsoft PowerPoint presentation
2. Sample case
3. A5-sized sheets of paper with printed modeling elements

Outcomes



Figure D.1: Workshop participant uses proposed modeling elements to model sample process (workshop no. 3 - May 24, 2011)

EVALUATION SESSION 3 – OUTCOMES

“MODELING BUSINESS PROCESSES FOR SIMULATION”

WORKSHOP CHARACTERISTICS

Date: May 24, 2011
Attendees: Ivo Wenzler (first 30 min. through call-in) and Rutger Deenen
Total duration 2 hours

OUTLINE OF THE WORKSHOP

Discussion about:

- Sub-processes (not discussed during workshop)
- Entities
- Gateways:
 - Parallel activities
 - Exclusive gateways
- Statistics and outputs

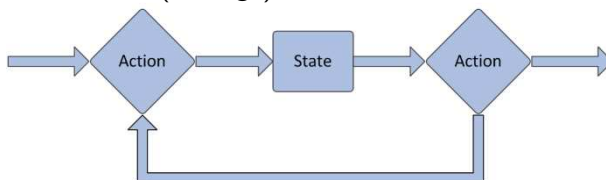
OUTCOMES

Entities: What is an Entity?

- “An entity is a work package. It is an element/construct/concept that initiates a business process by its arrival or creation and flows through the process, undergoing status changes through every activity.”
- An entity is a generic flow element (similar to a token in BPMN), typified by two attributed:
 - Type e.g. mail, call, order
 - Status e.g. received, concept, processed, finished
- The entity-type depends on the actual business process

Assigning Entity Types

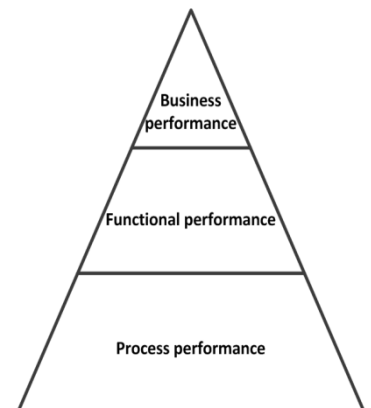
- When does the entity type change?
 - Every activity changes the status of an entity
 - An activity is followed by a decision whether an entity will “continue its flow”, or wait, or go back, or go “somewhere else”
 - You want to know when the status changes (the times), for your analysis / statistics, but also the (average) duration of the activities



- The arrival of entities with a certain type should represent how it happens in the real business process. This is done through separate start events, for every entity type.
 - An attribute can be defined from the start
 - Or during the process: in that case, the logic is already given, possibility to modify the value
 - In case the entity type is based on an uncertainty (assumption), this is modeled in the process.

Outputs and statistics: what are we interested in knowing about the processes?

- Statistics
 - Age, arrival / finished times of activities, handling times
 - Decisions: how many entities went in each direction?
 1. This is interesting for design options: "what if we change this, that influences the number that goes through this part of the business process"
- Activity: average handling time per activity / per business process / etc
- Example of KPI's divided in sub-KPI's, e.g.:
 - Main KPI: percentage of entities status "good"
 1. sub-KPI 1: percentage directly good
 2. sub-KPI 2: percentage good within predefined norm time
 3. sub-KPI 3: percentage not good
 - KPI's follow from the process: the process is leading
- Output generation
 - also specific characteristics for periods of time: year-to-date, month-to-date, etc
 - "You want to know everything!"
 - If you know everything, you can make abstractions (periods, resources, etc), and you can check on totals
 - Can be another scientific research: what / how to group / etc
 - Should think about what / how you give names to statistics
 - Which blocks do you want to know? Swimlanes? Or:
 - Tasks 1 + 2 + 3 = lead-time of activity group of interest
- Design alternatives: what has the focus?
 - product type / priority / responsibilities ("who?")
 - also: *which queue-prioritization options you select, how does it influence the output?*
- Statistics of activities
 - per activity: time ("age")
 - per process / swimlane (Accenture "default")
 - Relates to for instance: "what are the value-added processes?"
- Performance Overview / hierarchy:
 - Business performance → Whole Chain / organization
 - Strategic level / decisions?
 - Functional performance → Business processes, units
 - Tactical level / decisions?
 - Process performance → email, activities
 - Operational level / decisions?



XOR gateway

- XOR gateway example: it are the same activities! So array (like Rutgers example about priorities: 1..7) ... clarify...

Parallel activities: What is a parallel activity?

- When two (or more) employees perform a process in parallel, whereby one is initiated by the other one, and at some moment the process is (may be) synchronized
- Duplication of entities: when there are two information streams. "Where it happens in a real business process"
- Join-Gateways when all resources are finished, or based on time original resource continues.
 - Question [Ivo]: what will a resource do when he is finished with his part of the process, but the other resource not? He may wait for the other business process, or perform other tasks
- Question: how to solve the second split outgoing flow problem? (which releases the resource which shouldn't happen)

- Possibility of implementing a new flow connection: message flow. However, the modeler should take care implementing a SPLIT and Synchronize correctly.
 - Suggestion: possibility to provide a pop-up window with a warning to the modeler (carefull using the outputs of a parallel split).
 - Suggestion: automatic validation mechanism of the business process model
- Duplication (SPLIT) not necessarily for statistics: but representing what happens in the real business process

Experiment design

- Execution of experiments (X runs, statistical analysis): maybe for lean 6sigma projects
- In previous what-if cases, they fixed the simulation-seed, modified parameters, and executed the simulation model

Modeling principles

- Also for processes: as generic as possible (elegant modeling)
- Activity/task should not be dependent on for instance entity type:
 - Preferred: "handle question", instead of "handle question by call", "handle question by mail", etc
- Two activities following in each other may signal a modeling error
- Main line how to model: "How does it happen in real business processes?"
- Abstractions: clients may say that their processes can't be structured, (e.g. "there are 20+ different order types"). Well, then you know that the incoming flow is that of "an order"
 - It's almost always possible to make an abstraction of the client's case, how difficult they might describe it themselves.
- How you model, and so how you formalize the language, depends also on the client
 - It's a combination of best practices of Accenture, and what the client has / wants
 - Suggestion: Pre-specified guidelines how to model (e.g. in IDEF0 max. 5 processes per level)

Component descriptions

This Appendix provides the complete overview of all modeling elements, including their specification and implementation.

- Start Event (see section Section E.1.1)
- End Event (see section Section E.1.2)
- Task (see section Section E.1.3)
- Sub Process (see section Section E.1.4)
- Sequence Flow (see section Section E.1.5)
- Exclusive Gateway (see section Section E.1.6)
- Parallel Gateway (see section Section E.1.7)
- Swimlane (see section Section E.1.8)
- Swimlane Entry (see section Section E.1.9)
- Swimlane Exit (see section Section E.1.10)
- Resource Manager (see section Section E.1.11)
- Entity (see section Section E.1.12)
- Signal (see section Section E.1.13)
- BPMNCoupledModel (see section Section E.1.14)
- Resource (see section Section ??)
- Queue (see section Section ??)
- QueueArray (see section Section ??)

E.1 Modeling elements and components

E.1.1 Start Event

Design and specification

The Start Event is an Event and represents the arrival of entities into an organization or specific business process. To mimic this behavior in a simulation model, this event is translated into an atomic DEVS model. This component generates entities based on a certain specification. It is similar to the 'Create-block' as can be found in the simulation software Arena. The parameters describing its specification, are:

- **Inter-arrival time** Time between the arrivals of two entities or two batches of entities
- **Entity Type** Type of the entity that arrives
- **Batch Size** Number of entities that arrive at once
- **Time first arrival** Point in time at which the first entity (or entity batch) arrives

The graphical representation of the Start Event is shown in Figure E.1.

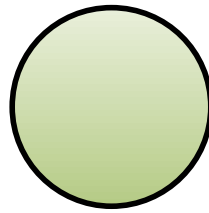


Figure E.1: Graphical representation of a Start Event

The formal description of the Start Event component is given by the state diagram as shown in Figure E.2. This component has one output-port through which a newly created entity leaves, and one state, namely the 'Passive' state. The time duration after which an output function (λ) takes place is equal to the inter-arrival time as specified by the modeler. This inter-arrival time may be constant (e.g., every 10 minutes an entity is created), or following a statistical distribution (e.g., an entity is created on average after 5 to 10 minutes, following a uniform distribution). In case batches of entities arrive, the component generates the specified number of entities, and sends them to the output-port.

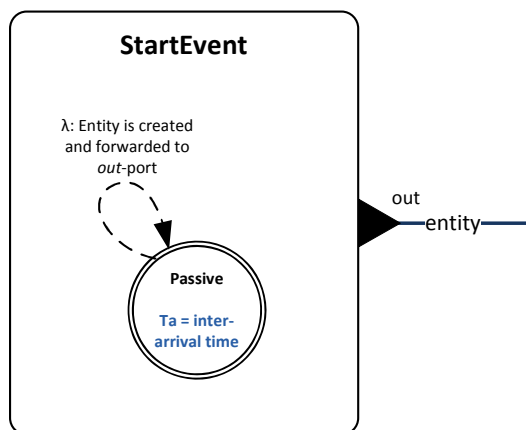


Figure E.2: State diagram Start Event

The Start Event is placed outside a Swimlane, to depict that arriving entities are coming from outside the organization or business unit and no resources are at the moment of arrival working on the entity. This conflicts with the BPMN specification regarding the placement of flow objects (e.g., activities, events, gateways) outside of pools or swimlanes. However, Accenture consultants found this to be more clear to be explained to a client.

Implementation prototype

The StartEvent-class is an implementation of the Start Event component and enables the instantiation of Entity-objects, which are send to the only defined output port (out). A StartEvent-instance is created and specified through its constructor (see Listing E.1). The constructor allows to specify a unique identifier for each Start Event component; a description of the Start Event; the inter-arrival time mode (e.g., constant, uniform or exponential); several parameters to specify the statistical distribution function; the batch size (number of entities that are generated at once); and the time at which the first entity is created.

Listing E.1: Constructor StartEvent.java

```
public StartEvent(BPMNCoupledModel parentModel, int myID, String description,
    String entityType, int mode, double mean, double min, double max, double
    stdDev, int batchsize, double firstCreationTime)
```

The Start Event class uses the jstats distributions library (included in DSOL) for the creation of various statistical distribution from which random inter-arrival times are acquired. Currently supported by the StartEvent-class are: Uniform, Exponential, Normal and Triangular distributions. Based on the mode that is specified in the constructor, a different distribution is used for the random creation of inter-arrival times. Each time an entity (or batch of entities is generated), the sigma of the model (time remaining before the output function is called) is reset to the next inter-arrival time.

To enable the output of statistics of a StartEvent-instance, a Counter object called counterEntitiesIn is instantiated. The counter object is also provided by DSOL. Every time an entity is send to the output port, the fireEvent-method is triggered (provided by DSOL), causing the the Counter-instance to be increased with 1. By using fire-events as provided by DSOL, we can use the statistics output interface of DSOL to display all generated statistics.

E.1.2 End Event

Design and specification

The End Event is an Event and represents the end of a business process, namely when an entity leaves the organization. The End Event depicts the boundary of interest of a modeled organization. The graphical notation of an End Event is shown in Figure E.3. To mimic the behavior of an entity leaving the organization, the End Event is formalized as an atomic DEVS model with two main states, namely 'Passive' and 'Active' (see Figure E.4). The component will remain in 'Passive' state, until an entity arrives at its input-port, triggering an external transition. Before the state changes back through an internal transition from 'Active' to 'Passive', the entity is disposed (i.e., destroyed).

The End Event is, similar to the Start Event, placed outside a swimlane, to depict that an entity leaves the organization and no resource is working any more on the entity. This also conflicts with the BPMN specification (as was the case with the Start Event), but it was found more clear by consultants to visually model End Events this way.

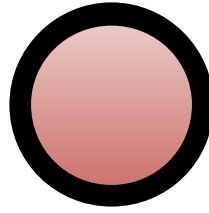


Figure E.3: Graphical representation of an End Event

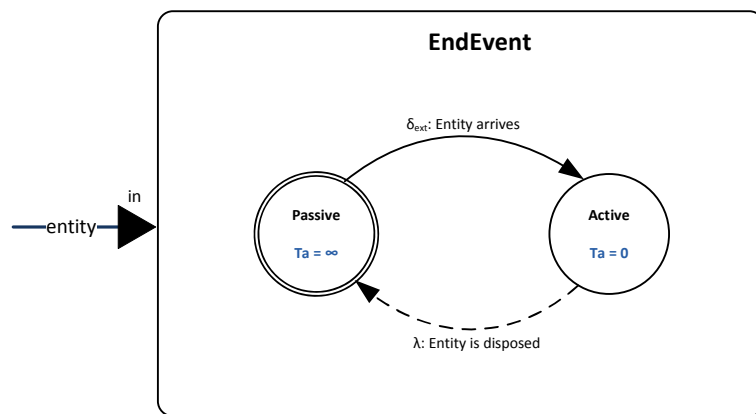


Figure E.4: State diagram End Event

Implementation prototype

The EndEvent-class is an implementation of the End Event component and enables the disposal of Entity-objects which arrive on the only defined input port (`in`). The EndEvent has two distinctive phases, passive and active with a lifetime of respectively infinite and 0 seconds. An EndEvent-object can be instantiated through its constructor(see Listing E.2). The constructor allows to specify a unique identifier, as well as a description of the End Event.

Listing E.2: Constructor EndEvent.java

```
public EndEvent(CoupledModel parentModel, int myID, String description)
```

An EndEvent-object generates several statistics about the entities and about the system. The statistical objects Counter (time independent) is used to count the number of entities which are disposed, and the Persistent-object (time dependent) is used to collect information about the total time of system of entities, the total time processed and the total time waited. The Persistent-object is also provided by DSOL.

E.1.3 Task

Design and specification

A Task is an Activity which represents work that is performed by a resource and consumes a certain amount of time. The graphical notation of a Task is shown in Figure E.5. To mimic behavior of a resource performing an activity for some time, the Task component (an atomic DEVS model) will delay an entity arriving at the input-port for a specified duration, before sending it to the output-port (see Figure E.6). Because one Task can be performed by multiple resources at the same time (work performed on different entities), the Task may contain multiple entities at a same instance.

When a Task is in 'Passive' state it means that no resource is currently performing that task. Due to an external event of an arriving entity, the state changes to 'Active'. If an entity arrives while the state is 'Active', the state remains 'Active' but the time advance is reset depending on which resource will be first finished with the task. If the duration of the newly arrived entity is shorter than the current shortest remaining duration of entities already being processed, the time advance function is set to the duration of this newly arrived entity. Else, the entity will be added to a list containing all entities that are currently processed.

When an internal transition occurs due to the elapsed duration of an entity, the state of the Task changes to 'Passive' if no more resources are currently performing that task, or remains in 'Active' state if one or more resources are performing that same task. The next internal transition event is scheduled for when the next entity with the shortest remaining duration.

A Task is specified by the following parameters:

- **Task duration** Time that it takes to perform the task by a resource on an entity.
- **Entity Type Change** In case the task changes the entity type, the new type can be specified through this parameter.
- **Entity State Change** After a task is performed, the state of an entity changes which is specified by the parameter.

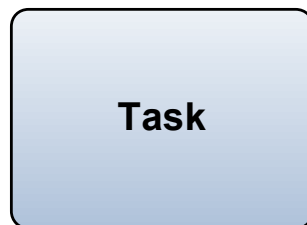


Figure E.5: Graphical representation of a Task

Tasks are placed inside a swimlane to depicts the capability of a resource performing that task, or outside the swimlane to depict a delay of an entity whereby no resource is required to wait.

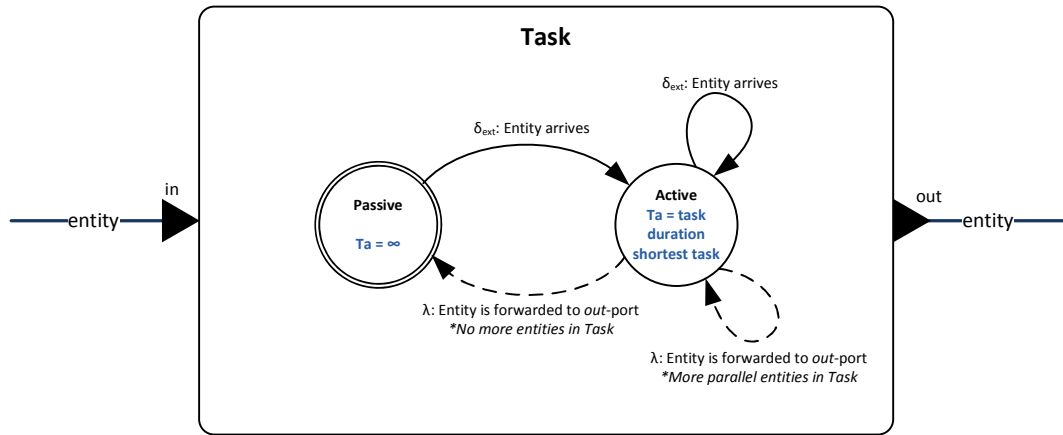


Figure E.6: State diagram Task

Implementation prototype

The Task-class is an implementation of the Task component and enables an entity to delay for a certain amount of time. A Task-object is initialized through its constructor (see Listing E.3) and the constructor allows to specify a description, unique identifier, mode for statical distribution selection and several parameters which specify the distribution function. The Task has two ports, one for receiving entities (*in*) and one for sending entities (*out*). Two phases are also defined: *passive* and *active* with a lifetime of both infinite (although the lifetime of the active phase is dynamically set depending on the arriving entity and possible other entities within the Task).

Listing E.3: Constructor Task.java

```

public Task(CoupledModel parentModel, String description, int myID, int mode,
            double mean, double min, double max, double stdDev)

```

Although a task is modeled as a unique element, it represents a parallel series of tasks, for each resource within a swimlane. An option would have been to define the task as a coupled model, with a atomic model as a switch-model and a series of atomic models to represent a task for each resource. This was developed and evaluated, and although it worked well for very small models, for larger models another implementation was chosen.

The Task consists of an ArrayList-object named *parallelTaskList*. Arriving entities are stored temporarily in the *parallelTaskList* following a specified order: the entity that is finished first is stored at the first location of the list, the entity that is finished after the first entity is stored at the second location, and so on. When an entity arrives, all remaining service times of the waiting entities in the *parallelTaskList* are updated by the *updateTimes()*-method. The *updateTimes()*-method decreases the remaining service time with the elapsed time since the last external or internal event (*e*). Next, the service time is calculated of the newly arrived entity (in a similar way as how the inter-arrival time is calculated in the Start Event) and is compared with the remaining service times. Based on these updated times, the newly arrived entity is stored in the *parallelTaskList* at the appropriate location and sigma is set to the remaining service time of the entity

which is first finished being serviced.

When e becomes equal to σ , and no external event occurred, the first waiting entity is removed from the `parallelTaskList`, its statistics are updated and the entity is send to the output port. Next, the `updateTimes()`-method is called again which updates the remaining times of the entities by decreasing each with the originally set σ (before the previous entity left). When no more entities are `parallelTaskList`, the phase of the Task changes back to `passive`.

E.1.4 Sub Process

Design and specification

The Sub Process component is an Activity that can contain other Activities as well as Gateways. It is used to visually organize the business process model by hiding details. It provides a way of hierarchical modeling, as it may contain again Sub Processes. The Sub Process has two visual states: 'folded' and 'unfolded'. When a Sub Process is folded, it hides the internal details and is represented as shown in Figure E.7. When the Sub Process is unfolded, the internal components and flows become visible.

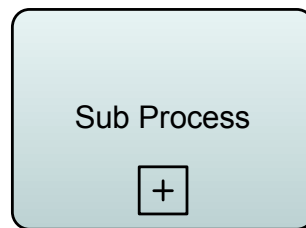


Figure E.7: Graphical representation of a Sub Process when folded

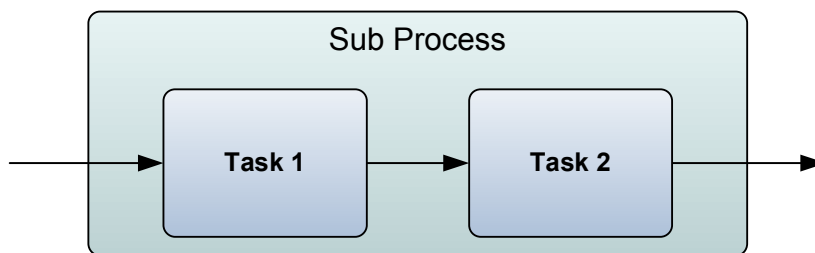


Figure E.8: Graphical representation of a Sub Process when unfolded

The Sub Process is modeled in DEVS as a coupled model, where it can a unlimited number of input ports as well as an unlimited number of output ports.

Implementation prototype

The Sub Process is not implemented in the prototype. Although the sub process element is included in the panel of the visual model builder, and can be drawn as specified, it is not correctly translated into a coupled model. Future work should focus on this.

E.1.5 Sequence Flow

Design and specification

The Sequence Flow (or Flow) is used to create a path between components through which entities can flow. The path depicts the sequential order in which activities are undertaken by a resource on an entity. A Sequence Flow can only link one output-port of a modeling element with one input-port of another modeling element. In the DEVS model the Sequence Flow is modeled through the couplings between atomic models and inside coupled models.

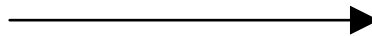


Figure E.9: Graphical representation of a Sequence Flow

Implementation prototype

The Sequence Flow is not implemented as a separate Class, but can be constructed by specifying DEVSDSOL-couplings between the components.

E.1.6 Exclusive Gateway

Design and specification

An Exclusive Gateway is a Gateway which is used to represent decisions made in a business process and to direct the flow of an Entity in a specified direction, based on the evaluation of a condition. This condition can be either the evaluation of an entity-specific attribute (e.g., entities of a specified type or state move in one specified direction, other entities move in another direction), or assumption (e.g., 70% of the arriving entities move in one direction, the other 30% move in the other direction). The graphical representation of an Exclusive Gateway is shown in Figure E.10.

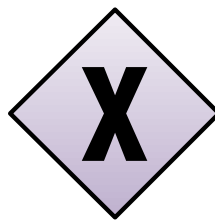


Figure E.10: Graphical representation of an Exclusive Gateway

In Figure E.10 the formalized DEVS model of an Exclusive Gateway component is given. This component has one input-port (`in`) through which entities arrive and two output-ports (`out0` and `out1`) through which entities leave. It should be noted that an arriving entity can leave through only one output-port (it cannot occur that an entity leaves through both output-ports at the same time). After an entity arrives at the

input-port of an Exclusive gateway component, the state of the component changes to 'Active' for a duration of 0 seconds. Before the internal transition takes place, the output function is called which evaluates the condition and sends the entity to one of the two output ports. The internal transition changes the state back to 'Passive'.

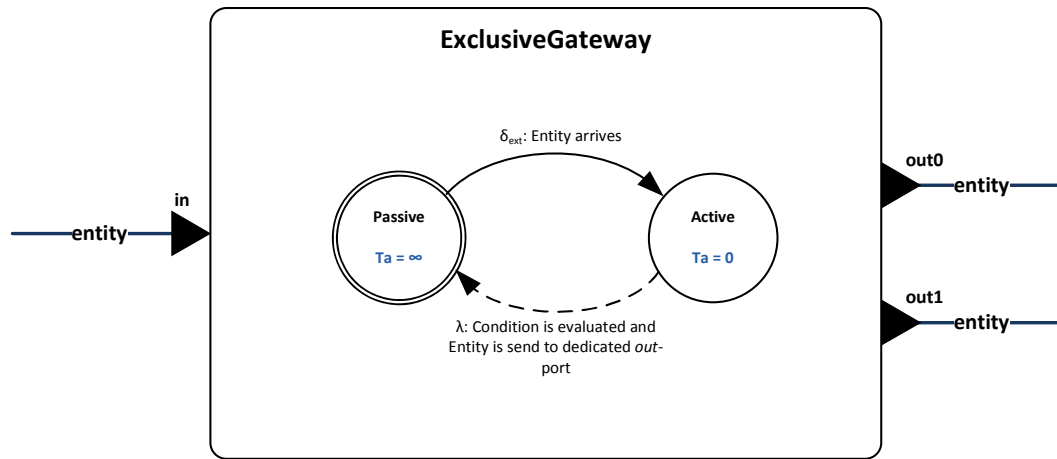


Figure E.11: State diagram Exclusive Gateway

An Exclusive Gateway is specified by the following parameters:

- **Evaluation condition** Condition that is evaluated to decide through which output-port the entity will leave the component. This can be either a percentage or an attribute-value.

Implementation prototype

The ExclusiveGateway-class is an implementation of the Exclusive Gateway component and can be initialized by one of the two constructors as shown in Listing E.4. The first constructor allows to specify an Exclusive Gateway component which evaluates an entity-attribute, the second constructor is to specify a probability based Exclusive Gateway. An ExclusiveGateway-object has one input port (in), two output ports (out and out1) and two phases are instantiated, namely passive and active.

Listing E.4: Constructors ExclusiveGateway.java

```
public ExclusiveGateway(CoupledModel parentModel, String description, int myID
    , String attributeToEvaluate, String valueOfAttribute)

public ExclusiveGateway(CoupledModel parentModel, String description, int myID
    , double probabilityExit0)
```

Currently, an attribute-based ExclusiveGateway-object can only direct an entity based on the evaluation of the entity-type attribute of an entity. To allow an Exclusive Gateway to direct an entity based on a certain probability, a uniform distribution is instantiated with a minimum value of 0 and a maximum value of 1. Based on the probability specified in the constructor, a comparison is made between this value and a random value from the uniform distribution and the entity is directed to the appropriate output port.

In the prototype, the first drawn connector leaving the Exclusive Gateway, is connected by the ATL-Transformation rules to the out0-output port. When an Exclusive Gateway is modeled with two outgoing flows, it is unclear which is connected to the out0-port. This implicated that it is unknown where an entity is send to, if the condition is evaluated 'true'. This should be improved in a future version of the prototype.

E.1.7 Parallel Gateway

Design and specification

A Parallel Gateway is a Gateway which is used to support modeling and simulation of parallel activities in a business process. To enable parallel activities, Parallel Gateways are used in pairs: one gateway is used at the start of a parallel activity. It duplicates an entity and forwards the entities to the activities that are performed by different resources in parallel. A second Parallel Gateway is used to synchronize a parallel activity again after both activities are completed. The graphical representation of a Parallel Gateway is shown in Figure E.12.

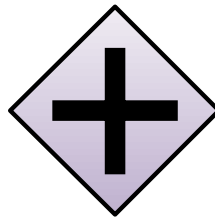


Figure E.12: Graphical representation of a Parallel Gateway

For the simulation model representation the functionality of a Parallel Gateway was divided over two distinctive components, namely a Parallel Gateway Split and a Parallel Gateway Join component. The state diagrams of these components are shown in Figure E.13 and Figure E.14. The Parallel Gateway Split changes from a 'Passive' state to 'Active' when an entity arrives on its input-port. The output function that is triggered just before the internal transition changes the state back to 'Passive', duplicates (clones) the arrived entity and sends its original to the out0 output-port. The duplicate entity is send to the out1 output-port.

As can be seen in Figure E.14, the Parallel Gateway Join has a set of three states: 'Passive', 'Active', and 'Check Entities Already Waiting'. Synchronization of parallel activities through a Parallel Gateway Join is based on the concept that an entity will wait in this gateway for an unspecified amount of time until its 'twin' entity arrives.

When an original or duplicate entity arrives (which depends on which partial activity is finished first) while the component's state is 'Passive', the entity will be placed in a waiting list and the state changes to 'Active'. When another entity arrives, the state changes from 'Active' to 'Check Entities Already Waiting'. In this state a comparison is made between the just arrived entity, and the already waiting entities, to see whether its twin is already waiting in the waiting list. If this is the case, the original entity will leave through the out output-port. If no match is found between the just arrived entity and

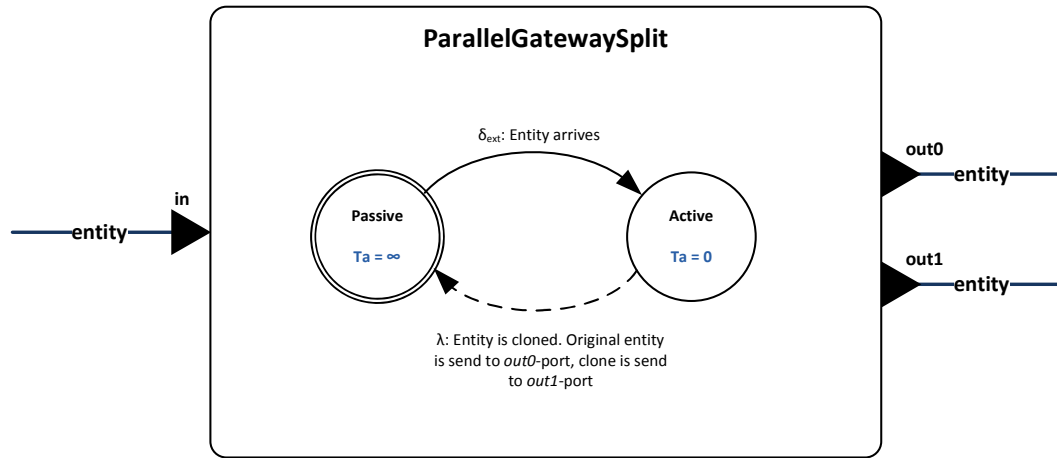


Figure E.13: State diagram Parallel Gateway Split

the current waiting entities, the newly arrived entity will also be added to the waiting list and the component state will change back to 'Active'.

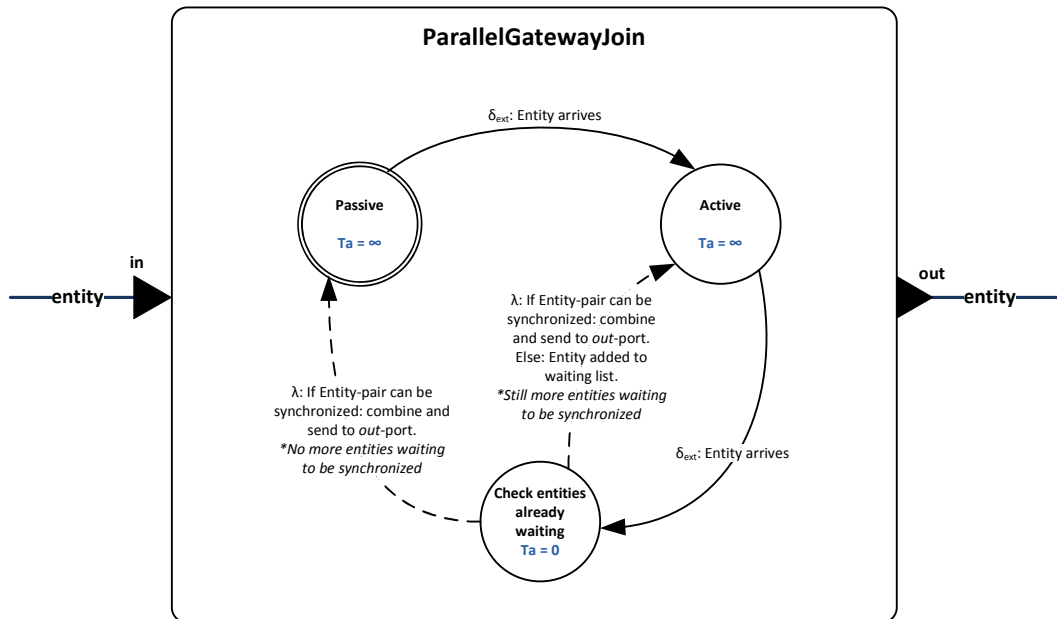


Figure E.14: State diagram Parallel Gateway Join

Implementation prototype

ParallelGatewaySplit.java

The ParallelGatewaySplit-class is an implementation of the Parallel Gateway Split component and is initialized by the constructor as shown in Listing E.5. The constructor allows

to specify a description, a unique identifier, and the identifier of the Swimlane that the `ParallelGatewaySplit` belongs to. A `ParallelGatewaySplit`-object has one input port (`in`), two output ports (`out` and `out1`) and two phases are instantiated, namely `passive` and `active`.

Listing E.5: Constructor `ParallelGatewaySplit.java`

```
public ParallelGatewaySplit(CoupledModel parentModel, String description, int
    myID, int swimlaneID)
```

When an entity arrives, a bit-wise copy is created of the entity through the implemented `clone()`-method. To enable this, the `Entity`-class implements the `Cloneable`-interface of the more generic `Object`-class. The entity that originally arrived is send to the `out` output port. The attribute depicting that the clone entity is indeed a clone, is changed to `true` and the identifier of the Swimlane it is cloned in (parent coupled model of the `ParallelGatewaySplit`-instance) is also updated in the entity attribute. For statistics purposes the 'total serviced time' and the 'total waited time' attributes of the entity are reset to 0.0. Finally, the cloned entity leaves through `out1` output port.

The implemented ATL-transformation rules automatically check whether a sequence flow leaves the swimlane (whether it is connected to an element outside the Swimlane), or whether it stays within the Swimlane. In the first case, the DSOL-coupling is created from the `out1`-port. The `out0`-port (where the original entity is send to) needs to stay within the Swimlane.

Future work should focus on how to make the difference between the two (or more) outgoing flows visible and usable. An option would be to have a second connector, for instance a Message Flow (similar to the BPMN specification), which a modeler can use to draw the connection which contains the duplicate entity.

`ParallelGatewayJoin.java`

The `ParallelGatewayJoin`-class is an implementation of the Parallel Gateway Join component and is initialized by the constructor as shown in Listing E.6. The constructor allows to specify a description, a unique identifier, and the identifier of the Swimlane that the `ParallelGatewayJoin` belongs to. A `ParallelGatewayJoin`-object has one input port (`in`) and one output port (`out`). Three phases are instantiated, namely `passive`, `active` and `checkEntities`.

Listing E.6: Constructor `ParallelGatewayJoin.java`

```
public ParallelGatewayJoin(CoupledModel parentModel, String description, int
    myID, int swimlaneID)
```

The `ParallelGatewayJoin` also uses a `Queue`-object called `waitingEntities` to store entities temporarily, until the clone (or original) arrives. When an entity arrives, the `lookUp()`-method of the `Queue`-class is called to compare the identifier (ID) of the arrived entity with the entities currently waiting. If a match is found (an entity is waiting in the `waitingEntities`-queue with the same ID as the arrived entity), then this entity is removed from the queue. The attributes of the original entity are updated by increasing the processing and waiting time with the attributes of the cloned entity since the split, and is forwarded to the `out` port. The clone-entity is then disposed.

Each time an entity is added or removed from the waitingEntities-queue, a fire-event takes place to produce statistics about the average waiting time and the number of entities waiting in the queue (which are both time-dependent Persistent-objects).

E.1.8 Swimlane

Design and specification

A Swimlane represents a resource or a group of resources that share the same capabilities of performing certain activities. The graphical representation of a swimlane is shown in Figure E.15. The swimlane can contain Activity and Gateway modeling elements, as well as Sequence Flows to connect the elements.



Figure E.15: Graphical representation of a Swimlane

Compared with the previous mentioned models, the Swimlane is not an atomic DEVS model but a coupled model. An entity arrives in a Swimlane after it is created by a Start Event which is placed outside the Swimlane. The point where the Sequence Flow crosses the boundary of a Swimlane, depicts the place where an entity will wait in a queue until a resource is available and can be assigned to work on it. After all activities in a Swimlane are performed, the entity can leave to another Swimlane, or enter an End Event. The point where the entity crosses the boundary to leave the swimlane depicts the place where the resource that was originally working on the entity becomes available to start working on another waiting entity. These points are modeled using Swimlane Entry and Swimlane Exit components.

The choice to have Swimlane Entry and Swimlane Exit components was made in a design workshop with involved consultants. Although some business process simulation tools (e.g., Rockwell Arena) use process components (similar to our Task component) which include the possibility to claim per process a resource and release him after the process is finished, it made more sense to the consultants that entities which enter a Swimlane should claim a resource from that point on. During a series of activities namely, the resource remains associated with the entity, until he hands it over to another resource, or the entity leaves the swimlane and leaves the organization (through an End Event).

To enable these characteristics (of a Swimlane Entry queue; claiming a resource; and releasing a resource) in a simulation model, choices had to be made how to organize the assignment of resources to entities that are waiting to be processed. A possibility is to chose for a central approach, whereby one component monitors all states of the queue in a complete simulation model. Another option would be to chose for a de-central approach, whereby all Swimlane Entry components know which resource is available so –

if a resource is available – he can be claimed. However, because of the different options of how resources in real business processes select an entity waiting to be processed (see Section 5.2), the the choice was made to let each Swimlane component have its own Resource Manager component.

In the following three subsections, the Swimlane Entry, the Swimlane Exit and the Resource Manager are discussed more elaborately

Implementation prototype

The Swimlane-class is the implementation of the Swimlane component. As was mentioned in Section 5.3.4 the Swimlane implementation represents a coupled DEVS model and contains the couplings between all components that are placed in a Swimlane. For each Swimlane that is modeled a new Swimlane-class is defined containing the unique couplings. Thus, in a DSOL simulation model, a Swimlane-object can only be instantiated ones. To clarify, if there are two Swimlanes in a model, named *ResourceA* and *ResourceB*, then *Swimlane.java* is copied and renamed to *Swimlane.ResourceA.java* and *Swimlane.ResourceB.java*. In each file, the unique couplings between the internal components are defined, as well as the instantiation of the components itself.

The Swimlane-class contains two methods which allows for the automatic generation of ID's for the Swimlane Entry and Swimlane Exit component. The ID's are for instance used to direct a signal to a specific Swimlane Entry: Its ID is used as sort of an 'address'. Also, the Resource Manager keeps track of all queues in Swimlane Entries, according to their associated ID.

E.1.9 Swimlane Entry

Design and specification

The Swimlane Entry component depicts the place where an entity enters a Swimlane and waits until a resource is available to 'pick up' the entity and perform specified activities. A Swimlane Entry is not directly modeled in a conceptual model as a separated modeling element, but exists at every point where a Sequence Flow intersects with the boundary of and enters a Swimlane. The functionality of Swimlane Entry is shown in the state diagram in Figure E.16.

The Swimlane Entry contains five states, to know: 'Passive', 'Add Entity to queue', 'Remove entity from queue', 'Forward entity' and 'Forward signal', and has two input ports: *in* (for receiving entities) and *inSignal* (for receiving signals). Signals are used for communication between Swimlane Entries and the Resource Manager component. Due to the specification of the DEVS formalism, information can only be exchanged between models through couplings. Because we decide to use the Resource Manager, as the active component to organize the assignment of resources to entities, the Resource Manager should be aware of the current state of all Swimlane Entries (i.e., how many entities are currently waiting in each Entry queue).

When an entity enters a Swimlane Entry and is placed in a queue, a signal is send to *outSignal* output-port. This *outSignal* port is coupled with the *in* port of the Resource Manager component. The signal contains an identifier of the Swimlane Entry

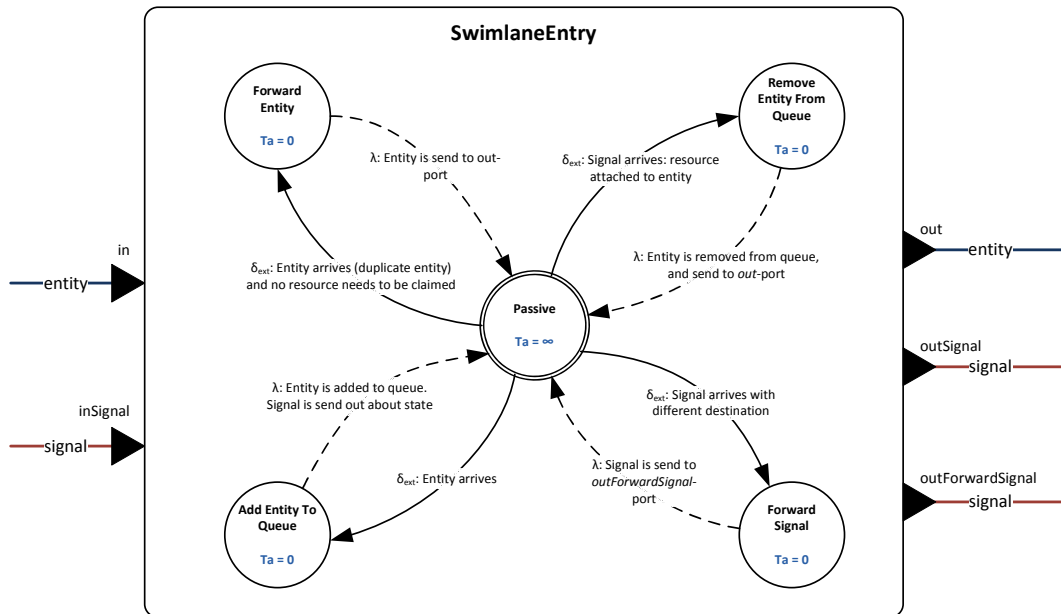


Figure E.16: State diagram Swimlane Entry

(so the Resource Manager knows the origin of the signal), as well as the current state of the queue. In case a Resource is available, the Resource Manager will send back a signal containing the Swimlane Entry destination identifier, as well as the identifier depicting exactly which resource is available. This signal arrives through the coupling between the Resource Manager component and the Swimlane Entry `inSignal` port, a check is performed whether the destination of the signal corresponds with the identifier of that specific Swimlane Entry. In case the destination of the signal corresponds with the Swimlane Entry, an entity is removed from the Swimlane Entry queue and is send to the `out` output-port.

In case the destination of the signal does not correspond with the Swimlane Entry, the signal is forwarded to the `outForwardSignal` output-port. The `outForwardSignal` port is connected with the next Swimlane Entry `inSignal` input-port. When there are more than one Swimlane Entry components, these components are mutually coupled through the `outForwardSignal` output-port and the `inSignal` input-port, creating a chain of Swimlane Entries through which the Signal can move and finally arrive at its destination.

It may occur that an entity arrives at the `in` input-port which does not need to wait for an available resource. This is the case for duplicate entities (which are created by the Parallel Gateway Split component). These entities leave a Swimlane and possibly re-enter it later on in the process to be synchronized with the original entity. However, the resource working on the original entity is still occupied with the original entity. In case the flow enters the swimlane and connects directly to a Parallel Gateway Join, the duplicate entity should be forwarded directly to the `out` port.

Implementation prototype

The SwimlaneEntry-class is an implementation of the Swimlane Entry component and is initialized by the constructor as shown in Listing E.7. The constructor allows to specify the identifier of the Swimlane that the Swimlane Entry belongs to, as well as a unique identifier. A SwimlaneEntry-object has two input ports (`in` and `inSignal`) and three output ports (`out`, `outSignal` and `outForwardSignal`). The five phases are instantiated as specified in the component description section of the Swimlane Entry (see Section 5.3.4).

Listing E.7: Constructor SwimlaneEntry.java

```
public SwimlaneEntry(CoupledModel parentModel, int swimlaneID, int myID)
```

The Swimlane Entry also instantiates a Queue-object, named `entryQueue`, in which arriving entities are stored until a signal is received which removes one entity from the `entryQueue` and sends it to the output port `out`. Each time an entity is added or removed from the `entryQueue`, a fire-event takes place to produce statistics about the average waiting time and the number of entities waiting in the queue (which are both time-dependent Persistent-objects).

E.1.10 Swimlane Exit

Design and specification

The Swimlane Exit component depicts the place where an entity leaves a Swimlane and the resource becomes available to 'pick up' a possible waiting entity. A Swimlane exit is not directly modeled in a conceptual model as a separated modeling element, but exists at every point where a Sequence Flow intersects with the boundary of and leaves a Swimlane. The functionality of Swimlane Exit is shown in the state diagram in Figure E.17.

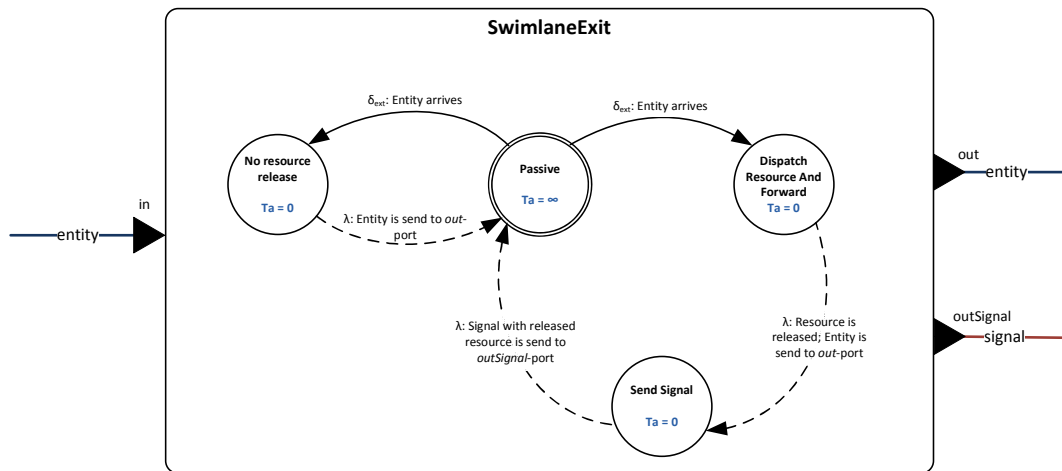


Figure E.17: State diagram Swimlane Exit

A Swimlane Exit has four states, namely 'Passive', 'Dispatch resource and forward', 'Send signal' and 'No resource release'. When an entity enters the Swimlane Exit component, the entity is first checked whether it is a duplicate entity or an original. In case the entity is a duplicate entity which is leaving the swimlane where in it was created, it should not release the resource. This because the resource is still 'attached' to the original entity and would cause problems later on in a simulation model execution. In case the entity is not a duplicate, or is a duplicate but leaves a swimlane in which it was not created, the resource attached to the resource should be released. This is done by updating an entity variable containing information about whether and which resource is working on the entity, and by sending a signal to the Resource Manager component, containing the identifier of the resource that became available. This signal is send to the outSignal output-port and the entity is send to the out output port.

Implementation prototype

The SwimlaneExit-class is an implementation of the Swimlane Exit component and is initialized by the constructor as shown in Listing E.8. The constructor allows to specify the identifier of the Swimlane that the Swimlane Exit belongs to, as well as a unique identifier. A SwimlaneExit-object has one input port (in and two output ports (out and outSignal). Four phases are instantiated, namely passive, dispatchResourcesAndForward, sendSignal and noResourceRelease.

Listing E.8: Constructor SwimlaneExit.java

```
public SwimlaneExit(CoupledModel parentModel, int swimlaneID, int myID)
```

When an entity enters the SwimlaneExit, the attribute of the entity that contains the attached resource is cleared and the entity is send the out port. Next, a new Signal-instance is created which contains the identifier of the resource that was just released and is send to the ResourceManager through the outSignal port.

E.1.11 Resource Manager

Design and specification

The Resource Manager is a component which cannot be modeled visually by the modeler, but is part of every Swimlane. The Resource Manager is not part of the entity flow and can only receive signals (through the in input port) and send signals (through the out output port) (as shown in Figure E.18). After a signal is send by a Swimlane Exit because a resource came available, the Resource Manager releases the resource formally. This is done by updating a state variable which contains all information about all resources in the associated swimlane and their states (whether a resource is currently available or busy). After the variable is updated, or in case the signal's origin was a Swimlane Entry component, the state of the component changes from 'Passive' to 'Check resources and queue'.

This state consists generally of two main steps. First, the resource state variable is checked to see whether any resource is available to pickup an entity (i.e., whether a resource is looking for work to do). If this is not the case, the state changes back to

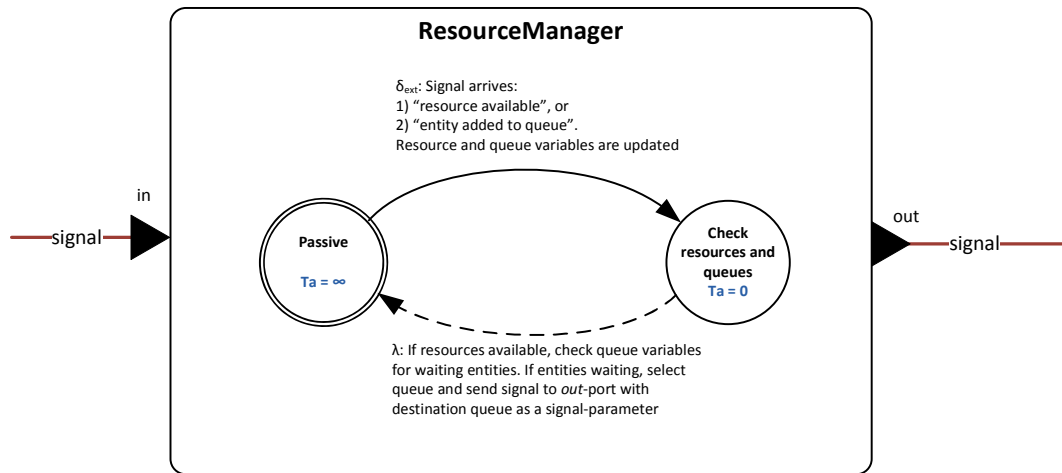


Figure E.18: State diagram Resource Manager

'Passive'. In case a resource is available, the second step takes place. Another state variable is now checked to see whether any entities are waiting in any Swimlane Entry queues. In case an entity is waiting in a certain queue, the state of the resource that was found available during the first step is set to busy and a signal is sent to the out output-port. This signal contains the identifier of the Swimlane Entry in which an entity is waiting, as well the resource that is attached by the Resource Manager to the waiting entity.

As was specified in Section 5.2, various ways exist how an available resource will select a queue. Due to this, Resource Manager is specified with a 'Queue Lane Mode' parameter, depicting how a resource will select a queue from several possible queues with entities waiting. This can be done based on a cyclical pattern, whereby first a waiting entity is taken from Swimlane Entry 1; then Swimlane Entry 2, ..., until the last Swimlane Entry is reached. Then the pattern starts over again at Swimlane Entry 1 and the cycle is repeated. If in a Swimlane Entry no entities are waiting, then the next Swimlane Entry is checked. The other ways as discussed in Section 5.2 under the heading "Resources and entity queues" can also be selected.

The Resource Manager is specified through the following parameters:

- **Number of resources** Number of resources capable of performing the activities within a swimlane
- **Queue Lane Mode** Specifies how an available resource selects a queue, for example Cyclical; Cyclical repeat; Longest average waiting time; Most entities waiting
- **Pattern value** Specifies – in case of Cyclical repeat mode – the number of times entities are taken from one queue, before moving on to the next queue in the cycle.

Implementation prototype

The ResourceManager-class is an implementation of the Resource Manager component and is initialized by the constructor as shown in Listing E.9. The constructor allows to

specify: a description; the queue lane mode (how to select a queue from several queues with waiting entities); the identifier of the Swimlane that the ResourceManager belongs to; the number of resources per queue, the pattern value (used for cyclical pattern) and the number of queues in the swimlane where a resource can be directed to (excluding the Parallel Gateway Join queue). A ParallelGatewayJoin-object has one input port (in) and one output port (out) . Two phases are instantiated, namely passive and active.

Listing E.9: Constructor ResourceManager.java

```
public ResourceManager(CoupledModel parentModel, String description, int
    modeQueueSelection, int swimlaneID, int numberOfResources, int
    patternValue, int numberOfQueues)
```

The Resource Manager uses three objects to maintain information about the current states of all resources and queues within a Swimlane.

1. A ResourceArray-instance named resourceArray is instantiated which contains a number of Resource-instances (the number is equal to the value specified in the constructor). The ResourceArray itself maintains data about the average utilization of the Resources, and provides methods to change the state of a resource (available or busy). These resources are stored in an Array.
2. An integer Array-instance named queueArray contains the current number of waiting entities in all queues.
3. A double Array-instance named longestWaitingTimesInQueue maintains the current longest waiting time of every queue.

In Listing Listing E.11 implemented code is shown which allows for the generation of cyclical sequence-lists. Such a list provides the functionality that the Resource Manager for instance first checks queue 0 for waiting entities, then queue 1, then queue 2, and so on, until all queues are checked. It will then repeat the cyclical sequence-lists. The code shown in Listing E.10 depicts how the actual cyclical sequence-lists is used in an Resource Manager object.

Listing E.10: ResourceManager.java - Pattern array generation code

```
135 if (modeQueueSelection == 3 || modeQueueSelection == 4){
136     // outcome of the following loop will be an array like [0,1,2,3,4,0,1,2,3]
137     (if there are in total 5 queues)
138     queuePattern = new int [(2 * numberOfQueues - 1)];
139     int a = 0;
140     for (int i = 0; i < queuePattern.length ; i++){
141         queuePattern[i] = a;
142         a++;
143         if (a == (numberOfQueues )){
144             a = 0;
145         }
146     }
```

Listing E.11: ResourceManager.java - Pattern Selection code

```
275 if (modeQueueSelection == 3 || modeQueueSelection == 4){
276     for (int i = nextPosition; i < (nextPosition + numberOfQueues); i++){
277         if ( queueArray[queuePattern[i]] > 0 ){
```

```

278         selectedQueue = queuePattern[i];
279         timesSameQueue ++;
280         nextPosition = i;
281         if (timesSameQueue == patternValue){
282             timesSameQueue = 0;
283             nextPosition ++;
284         }
285
286         if (nextPosition >= numberOfQueues ){
287             nextPosition = nextPosition - numberOfQueues ;
288         }
289         break;
290     }
291     timesSameQueue = 0;
292 }
293 }

```

E.1.12 Entity

Design and specification

The entity represents a object flowing through a business process model. In Section 5.2 under the heading “Entities” the concept of an entity was already discussed. The entity is, similarly to a signal, represented as a message in DEVS and is a passive object. It is passive in the sense that it is not an atomic DEVS model and does not have states. The entity can flow through the couplings between atomic models and coupled models. It contains information about which resource is currently attached to it, and whether it is a duplicate entity. An entity also contains several variables to store information for statistics generation, like for instance the time that it was created, or the total time it is being processed, but this will be discussed more in detail in Section 5.5.1.

Implementation prototype

The creation of an Entity-instance can be done through the constructor of the Entity-class and is shown in Listing E.12. It allows to specify: the entity type; a unique identifier (acquired through the getNextEntityID()-method from the BPMNCCoupledModel-class); the creation time and whether it is a duplicate entity. An entity-instance contains information about its creation time, its total serviced time, its total waited time, as well as a variable to specify its remaining time in a task. Through different methods, information from the entity can be acquired.

Listing E.12: Constructors Entity.java

```

public Entity(String entityType, int entityID, double creationTime, boolean
    isSyncEntity)

```

E.1.13 Signal

Implementation prototype

The Signal-class has two constructors (see Listing E.13) and an instance of the class represents a signal send between the SwimlaneEntry, the SwimlaneExit and the Resource-

Manager. The constructors are similar, with the exception that the first can also be instantiated with the current value of the current maximum waiting time of the entities in a queue. This constructor is used by a SwimlaneEntry-instance. The constructor also contains: the origin identifier (depicting where an entity was added to a Swimlane Entry queue); the destination identifier (depicting the destination where a signal is send to); and a resource identifier (to depict the resource that is released or attached to an entity).

Listing E.13: Constructors Signal.java

```
public Signal(int originID, int destinationID, int resourceID, double
    currentMaxWaitingTimeOfQueue)

public Signal(int originID, int destinationID, int resourceID)
```

The Signal-class contains methods to retrieve values of the variables, like `getFreedResource()`, `getDestinationID()`, `getCurrentMaxWaitingTimeOfQueue()`.

E.1.14 BPMNCoupledModel

Implementation prototype

The BPMNCoupledModel-class is used to extend Swimlane Coupled Models, as well as highest-level model containing all components. It BPMNCoupledModel contains methods to assign unique identifiers to each different object instance (except for the SwimlaneEntry and Exit). These assigned identifiers can be used for debugging purposes through the Eclipse console output. An overview of all methods provided by BPMNCoupledModel.java is shown in Listing E.14.

Listing E.14: Methods of BPMNCoupledModel.java

```
getNextEntityID()
getNextTaskID()
getNextSwimlaneID()
getNextStartEventID()
getNextEndEventID()
getNextExclusiveGatewayID()
getNextParallelGatewaySplitID()
getNextParallelGatewayJoinID()
```


Transformation rules descriptions

F

F.1 Pseudo-code description

Below we present an overview of all the Model-to-Model transformation rules. These should be applied in a top-down order to translate a conceptual model using the proposed business process modeling elements into a corresponding DEVS simulation model.

Atomic and Coupled DEVS Model creation; Input and Output DEVS Ports definition

For the main conceptual business process model	→	create a root coupled DEVS model
For each flow object (except the Sub Process) in the root model (not placed inside any Swimlane)	→	create a corresponding atomic DEVS model within the (root) coupled DEVS model
For each Swimlane and Sub Process modeling element	→	create a corresponding coupled DEVS model
For each flow object (except the Sub Process) not in the root model (thus placed inside a Swimlane)	→	create a corresponding atomic DEVS model within the corresponding coupled DEVS model
If one or more Sequence Flow(s) is connected to the input port of a flow object modeling element	→	define a DEVS Input Port for the corresponding atomic DEVS model
If one or more Sequence Flow(s) is connected to the output port of a flow object modeling element	→	define a DEVS Output Port for the corresponding atomic DEVS model
For each created coupled DEVS model which represents a Swimlane modeling element	→	create an atomic DEVS model representing the Resource Manager component
For each Sequence Flow entering a Swimlane modeling element	→	define a DEVS Input Port for the corresponding coupled DEVS model
	→	create an atomic DEVS model representing the Swimlane Entry component within the corresponding coupled DEVS model
For each Sequence Flow leaving a Swimlane modeling element	→	define a DEVS Output Port for the corresponding coupled DEVS model
	→	create an atomic DEVS model representing the Swimlane Exit component within the corresponding coupled DEVS model
For each Sequence Flow entering a Sub Process modeling element	→	define a DEVS Input Port for the corresponding coupled DEVS model

For each Sequence Flow leaving a Sub Process modeling element

→ define a DEVS Output Port for the corresponding coupled DEVS model

Couplings

For each Sequence Flow connecting a flow object in the root conceptual model and another flow object in the root conceptual model

→ create within the root coupled DEVS model an **Internal Coupling** from the DEVS Output Port of the atomic DEVS model corresponding to the source flow object to the corresponding DEVS Input Port of the atomic DEVS model corresponding to the target flow object

For each Sequence Flow connecting a flow object in a Swimlane modeling element and another flow object within the same Swimlane modeling element

→ create within the corresponding Swimlane coupled DEVS model an **Internal Coupling** from the Output Port of the atomic DEVS model corresponding to the source flow object to the corresponding DEVS Input Port of the atomic DEVS model corresponding to the target flow object

For each Sequence Flow connecting a flow object in the root conceptual model and entering a Swimlane modeling element

→ create three couplings:
 1) create within the root coupled DEVS model an **Internal Coupling** between Output Port of the atomic DEVS model corresponding to the flow object and the corresponding DEVS Input Port for the corresponding Swimlane coupled DEVS model
 → 2) create within the corresponding Swimlane coupled DEVS model an **External Input Coupling** between the corresponding Input Port of the coupled DEVS model and the DEVS Input Port for the corresponding Swimlane Entry atomic model component
 → 3) create within the corresponding Swimlane coupled DEVS model an **Internal Coupling** between the DEVS Output Port for the corresponding Swimlane Entry atomic model component and the DEVS Input Port for the corresponding atomic model component corresponding to the flow object

For each Sequence Flow leaving a Swimlane modeling element and connecting to a flow object in the root conceptual model

→ create three couplings:
 1) create within the corresponding Swimlane coupled DEVS model an **Internal Coupling** between the DEVS Output Port for the corresponding atomic model component corresponding to the source flow object to the DEVS Input Port for the corresponding Swimlane Exit atomic model component

- For each Sequence Flow connecting a flow object in a Swimlane modeling element with a flow object in another swimlane
- 2) create within the coupled DEVS model corresponding to the Swimlane coupled DEVS model an **External Output Coupling** between Output Port of the corresponding Swimlane Exit atomic model component and the corresponding DEVS Output Port for the corresponding Swimlane coupled DEVS model
 - 3) create within the root coupled DEVS model an **Internal Coupling** between the corresponding DEVS Output Port of the coupled DEVS model corresponding to the source Swimlane to the corresponding DEVS Input Port of the atomic DEVS model corresponding to the target flow object
 - create five couplings:
 - 1) create within the corresponding source Swimlane coupled DEVS model an **Internal Coupling** between the DEVS Output Port for the corresponding atomic model component corresponding to the source flow object to the DEVS Input Port for the corresponding Swimlane Exit atomic model component
 - 2) create within the coupled DEVS model corresponding to the source Swimlane coupled DEVS model an **External Output Coupling** between Output Port of the corresponding Swimlane Exit atomic model component and the corresponding DEVS Output Port for the corresponding Swimlane coupled DEVS model
 - 3) create within the root coupled DEVS model an **Internal Coupling** between the corresponding DEVS Output Port of the coupled DEVS model corresponding to the source Swimlane and the corresponding DEVS Input Port for the corresponding target Swimlane coupled DEVS model
 - 4) create within the coupled DEVS model corresponding to the target Swimlane coupled DEVS model an **External Input Coupling** between the corresponding Input Port of the coupled DEVS model and the DEVS Input Port for the corresponding Swimlane Entry atomic model component
 - 5) create within the coupled DEVS model corresponding to the target Swimlane coupled DEVS model an **Internal Coupling** between the DEVS Output Port for the corresponding Swimlane Entry atomic model component and the DEVS Input Port for the corresponding atomic model component corresponding to the target flow object

- | | |
|--|---|
| For each created Swimlane coupled DEVS model | <ul style="list-style-type: none"> → create within the coupled DEVS model corresponding to the Swimlane coupled DEVS an Internal Coupling between the Output Signal Port of each Swimlane Entry atomic DEVS model with the Input Port of the Resource Manager atomic DEVS model → create within the coupled DEVS model corresponding to the Swimlane coupled DEVS an Internal Coupling between the Output Forward Signal Port of each Swimlane Entry atomic DEVS model with the Input Signal Port of the next Swimlane Entry atomic DEVS model (until all swimlane entries are connected with each other via an unidirectional link) → create within the coupled DEVS model corresponding to the Swimlane coupled DEVS an Internal Coupling between the Output Signal Port of each Swimlane Exit atomic DEVS model with the Input Port of the Resource Manager atomic DEVS model → create within the coupled DEVS model corresponding to the Swimlane coupled DEVS an Internal Coupling between the Output Signal Port of the Resource Manager atomic DEVS model with the Input Signal Port of the first Swimlane Entry atomic DEVS model |
|--|---|

Note: Although not explicitly mentioned in table above, make sure that all Output Ports of the atomic DEVS models representing the Exclusive Gateway and Parallel Gateway Split are connected to the appropriate components (as drawn in the conceptual model).

Sample BPMM and DEVS models

G

Two Swimlanes and Exclusive Gateway example

Figure G.2 is the corresponding DEVS simulation model of the conceptual business process model as shown in Figure G.1.

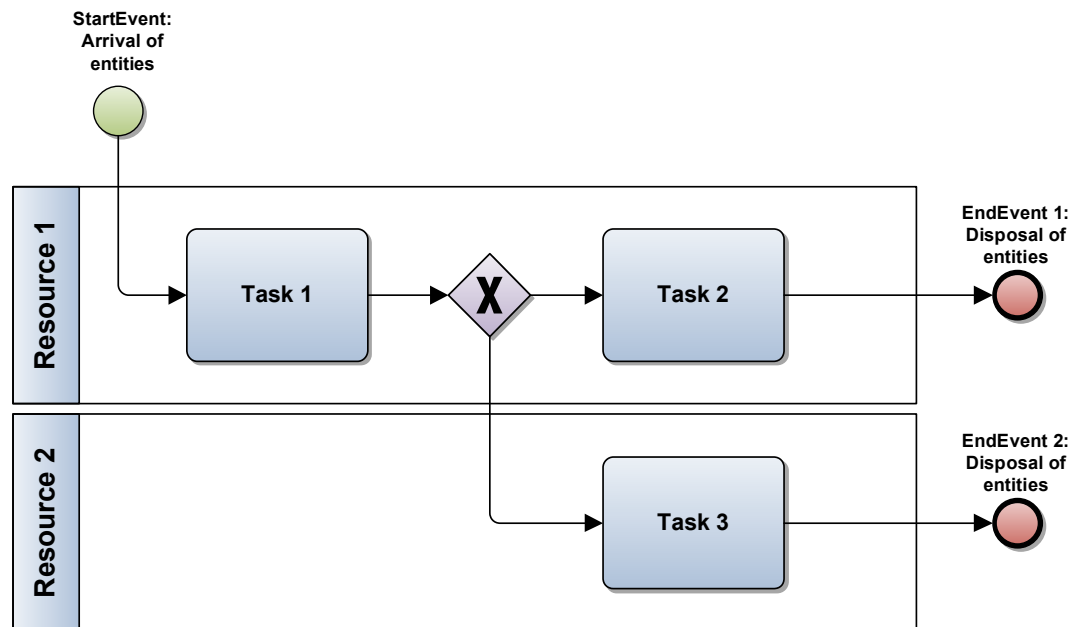


Figure G.1: Sample BPM: Two Swimlanes and Exclusive Gateway

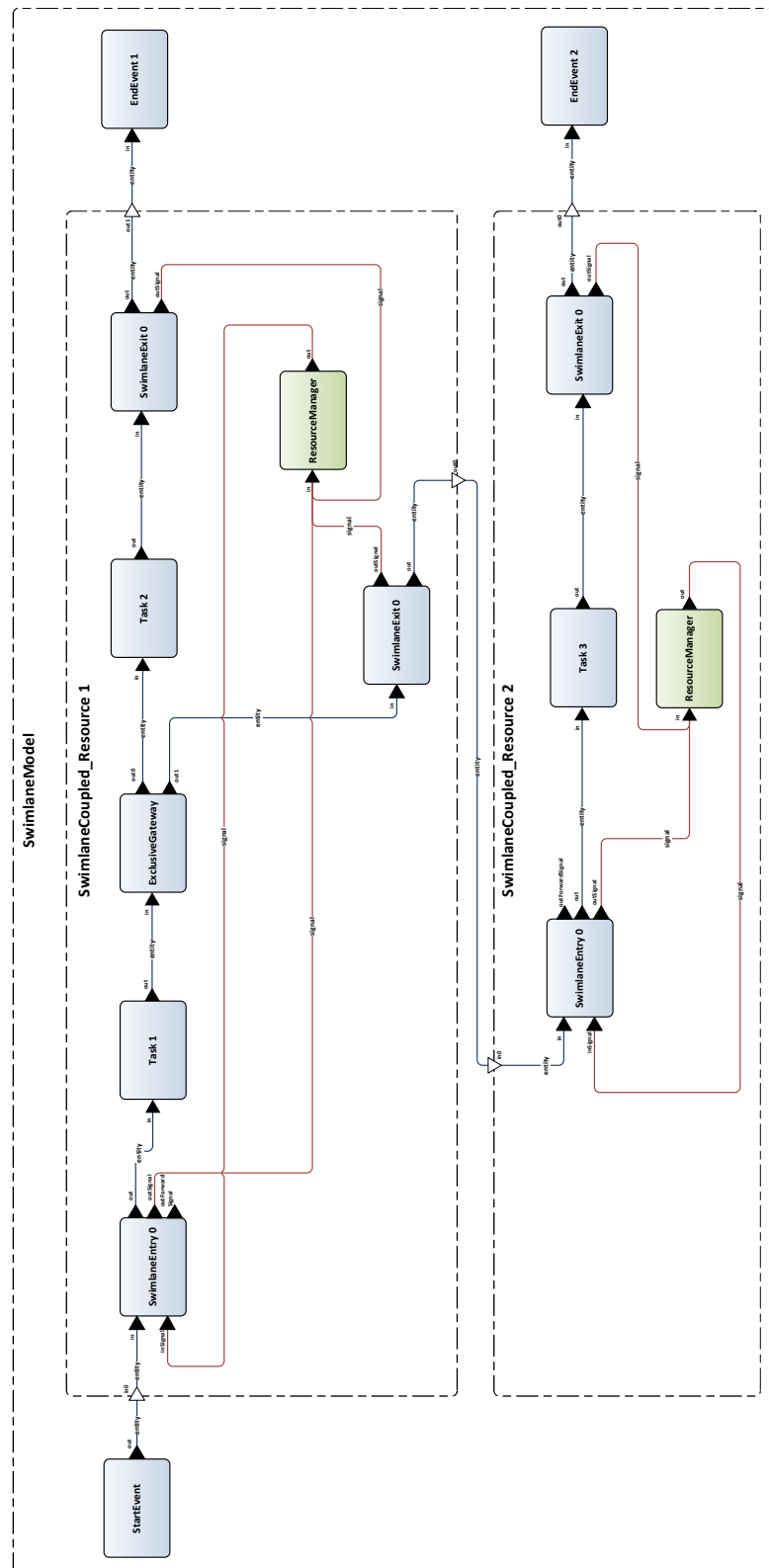


Figure G.2: Sample BPM: Two Swimlanes and Exclusive Gateway (DEVS)

Two Swimlanes and Parallel Activity example

Figure G.4 is the corresponding DEVS simulation model of the conceptual business process model as shown in Figure G.3.

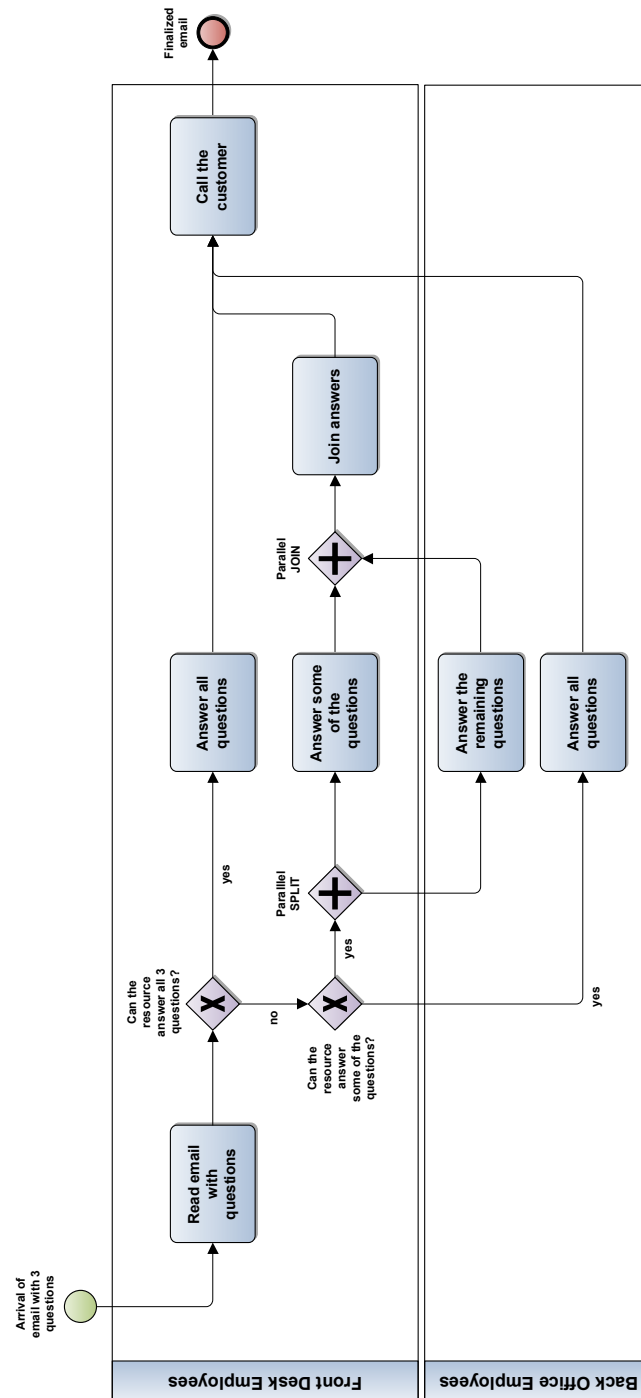


Figure G.3: Sample BPM: 2 Swimlanes and Parallel Activity

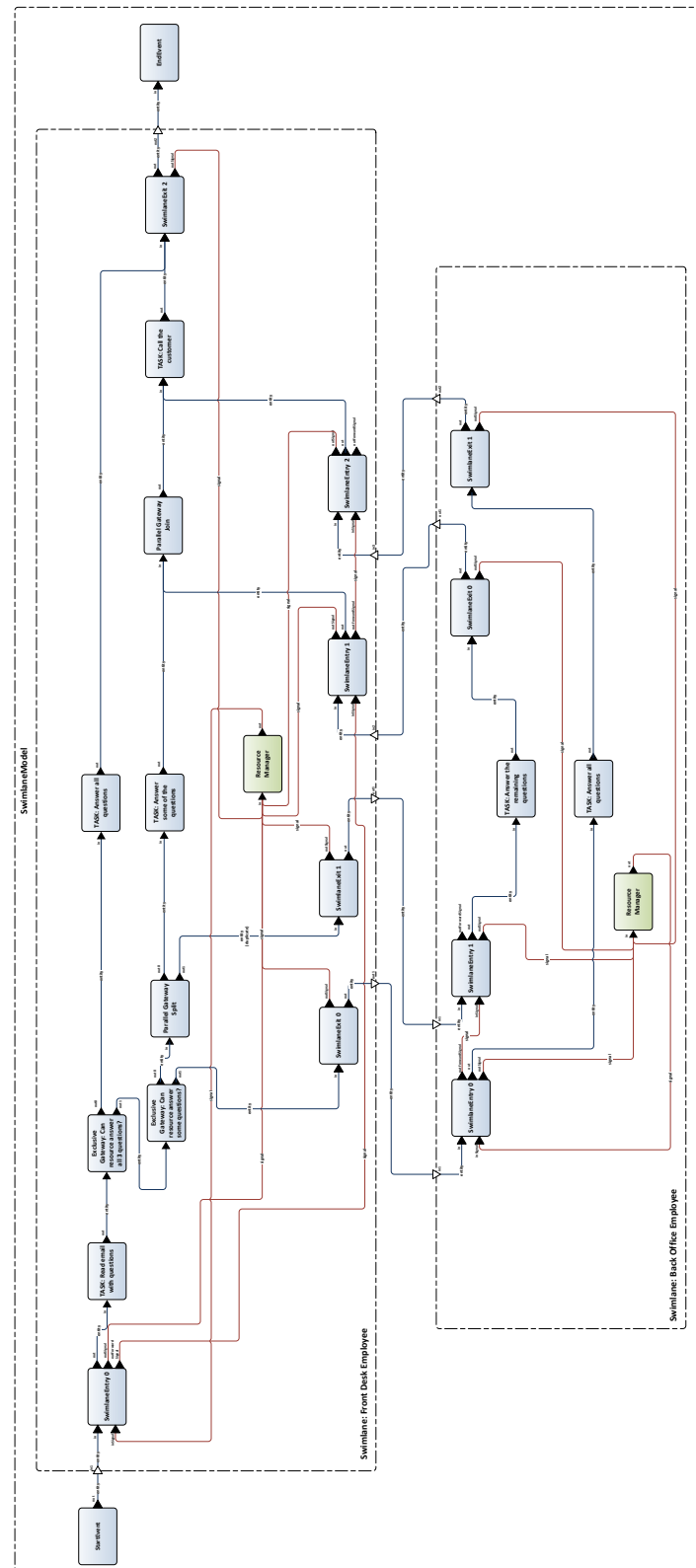


Figure G.4: Sample BPM: Two Swimlanes and Parallel Activity (DEVS)

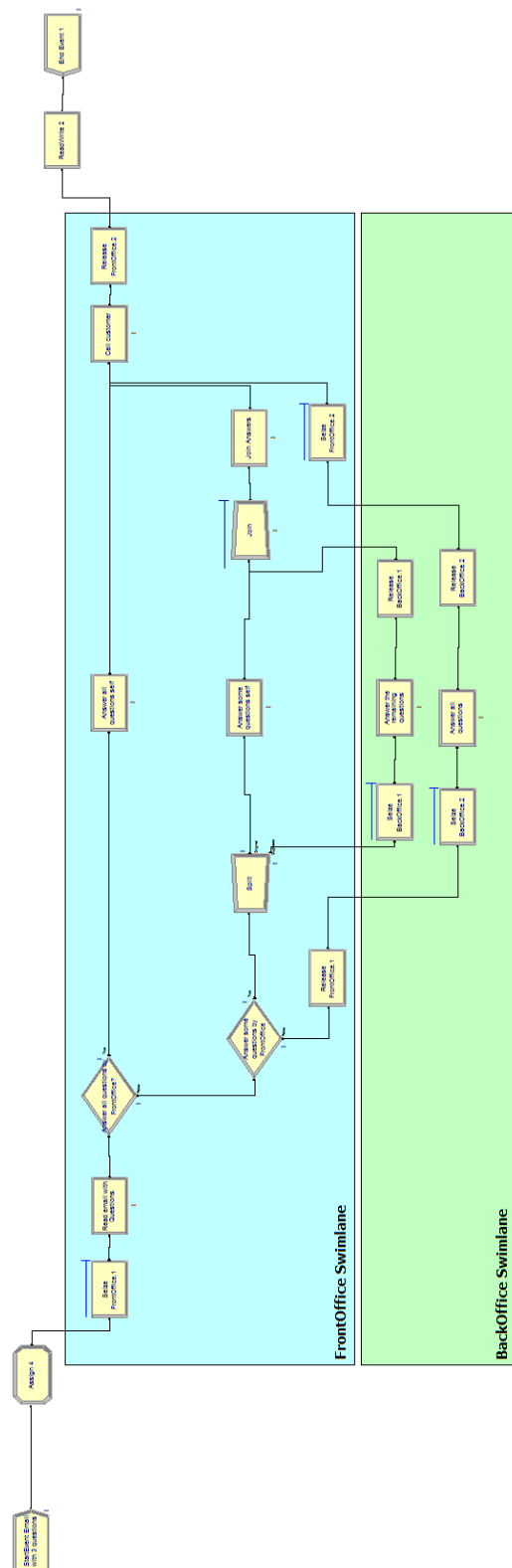
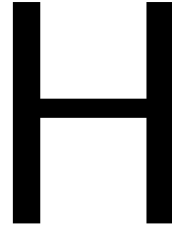


Figure G.5: Sample BPM: Two Swimlanes and Parallel Activity (Arena)

Verification of model translation



The syntax presented in this Appendix, relate to the test as described in Section 6.3.1.

Test 1: One Swimlane and one Task

Listing H.1: Test 1: SwimlaneModel.java

```
23 public SwimlaneModel(String modelName, DEVSSimulator simulator) {
24     super(modelName, simulator );
25
26
27     StartEvent var_ArrivalOfEntities = new StartEvent(this,
28         getNextStartEventID(), "ArrivalOfEntities", "Entity1", 4, 5, 3, 10, 0,
29         1, 0);
30
31     EndEvent var_DisposalOfEntities = new EndEvent(this, getNextEndEventID(),
32         "DisposalOfEntities");
33
34     Swimlane_Resource1 var_Swimlane_Resource1 = new Swimlane_Resource1(this, "
35         Swimlane", getNextSwimlaneID());
36
37     this.addInternalCoupling(var_Swimlane_Resource1.Swimlane_Resource1_out0,
38         var_DisposalOfEntities.in);
39
40     this.addInternalCoupling(var_ArrivalOfEntities.out, var_Swimlane_Resource1
41         .Swimlane_Resource1_in0);
42 }
```

Listing H.2: Test 1: Swimlane_Resource1.java - ports

```
13 public class Swimlane_Resource1 extends BPMNCoupledModel {
14
15     public InputPort<Entity> Swimlane_Resource1_in0= new InputPort<Entity>(
16         this);
17
18     public OutputPort<Entity> Swimlane_Resource1_out0= new OutputPort<Entity>(
19         this);
20 }
```

Listing H.3: Test 1: Swimlane_Resource1.java - objects and couplings

```
26 public Swimlane_Resource1(SwimlaneModel parentModel, String description, int
27     swimlaneID) {
28     super("Swimlane_Resource1", parentModel );
29
30     this.description = description;
31     this.swimlaneID = swimlaneID;
32     swimlaneModelParent = parentModel;
33
34     SwimlaneEntry var_Entry0 = new SwimlaneEntry(this, swimlaneID,
35         getNextSwimlaneEntryID());
36
37     SwimlaneExit var_Exit0 = new SwimlaneExit(this, swimlaneID,
38         getNextSwimlaneExitID());
39
40     Task var_Task1 = new Task(this, "Task1", swimlaneModelParent.getNextTaskID
41         (), 3, 0, 4, 12, 0);
42 }
```

```

36     ResourceManager var_RM_Resource1 = new ResourceManager(this, "
37         ResourceManager", 4, swimlaneID, 3, 2, 1);
38     this.addExternalInputCoupling(this.Swimlane_Resource1_in0, var_Entry0.in);
39     this.addExternalOutputCoupling(var_Exit0.out, this.Swimlane_Resource1_out0
40 );
41     this.addInternalCoupling(var_Entry0.out, var_Task1.in);
42     this.addInternalCoupling(var_Entry0.outSignal, var_RM_Resource1.in);
43     this.addInternalCoupling(var_Exit0.outSignal, var_RM_Resource1.in);
44     this.addInternalCoupling(var_Task1.out, var_Exit0.in);
45     this.addInternalCoupling(var_RM_Resource1.out, var_Entry0.inSignal);
46 }

```

Test 2: Two Swimlanes

Listing H.4: Test 2: SwimlaneModel.java

```

23 public SwimlaneModel(String modelName, DEVSSimulator simulator) {
24     super(modelName, simulator);
25
26
27     StartEvent var_ArrivalOfEntitiesType1 = new StartEvent(this,
28         getNextStartEventID(), "ArrivalOfEntitiesType1", "Entity 1", 1, 17, 0,
29         0, 0, 5, 0);
30     StartEvent var_ArrivalOfEntitiesType2 = new StartEvent(this,
31         getNextStartEventID(), "ArrivalOfEntitiesType2", "Entity 2", 3, 0, 10,
32         20, 0, 1, 100);
33     EndEvent var_DisposalOfEntities = new EndEvent(this, getNextEndEventID(),
34         "DisposalOfEntities");
35     Swimlane_Resource_A var_Swimlane_Resource_A = new Swimlane_Resource_A(this
36         , "Swimlane", getNextSwimlaneID());
37     Swimlane_Resource_B var_Swimlane_Resource_B = new Swimlane_Resource_B(this
38         , "Swimlane", getNextSwimlaneID());
39
40     this.addInternalCoupling(var_Swimlane_Resource_A.Swimlane_Resource_A_out0,
41         var_Swimlane_Resource_B.Swimlane_Resource_B_in0);
42     this.addInternalCoupling(var_Swimlane_Resource_A.Swimlane_Resource_A_out1,
43         var_DisposalOfEntities.in);
44     this.addInternalCoupling(var_Swimlane_Resource_B.Swimlane_Resource_B_out0,
45         var_Swimlane_Resource_A.Swimlane_Resource_A_in2);
46     this.addInternalCoupling(var_ArrivalOfEntitiesType1.out,
47         var_Swimlane_Resource_A.Swimlane_Resource_A_in0);
48     this.addInternalCoupling(var_ArrivalOfEntitiesType2.out,
49         var_Swimlane_Resource_A.Swimlane_Resource_A_in1);
50 }

```

Listing H.5: Test 2: Swimlane_Resource_A.java - ports

```

13 public class Swimlane_Resource_A extends BPMNCoupledModel {
14
15     public InputPort<Entity> Swimlane_Resource_A_in0= new InputPort<Entity>(<
16         this);
17     public InputPort<Entity> Swimlane_Resource_A_in1= new InputPort<Entity>(<
18         this);
19     public OutputPort<Entity> Swimlane_Resource_A_out0= new OutputPort<Entity>
20         (>(this);
21     public InputPort<Entity> Swimlane_Resource_A_in2= new InputPort<Entity>(<
22         this);

```



```

19 public OutputPort<Entity> Swimlane_Resource_A_out1= new OutputPort<Entity
    >(this);

```

Listing H.6: Test 2: Swimlane_Resource_A.java - objects and couplings

```

29 public Swimlane_Resource_A(SwimlaneModel parentModel, String description, int
    swimlaneID) {
30     super("Swimlane_Resource_A", parentModel );
31
32     this.description = description;
33     this.swimlaneID = swimlaneID;
34     swimlaneModelParent = parentModel;
35
36     SwimlaneEntry var_Entry0 = new SwimlaneEntry(this, swimlaneID,
        getNextSwimlaneEntryID());
37     SwimlaneEntry var_Entry1 = new SwimlaneEntry(this, swimlaneID,
        getNextSwimlaneEntryID());
38     SwimlaneExit var_Exit0 = new SwimlaneExit(this, swimlaneID,
        getNextSwimlaneExitID());
39     SwimlaneEntry var_Entry2 = new SwimlaneEntry(this, swimlaneID,
        getNextSwimlaneEntryID());
40     SwimlaneExit var_Exit1 = new SwimlaneExit(this, swimlaneID,
        getNextSwimlaneExitID());
41     Task var_Task1 = new Task(this, "Task1", swimlaneModelParent.getNextTaskID
        (), 4, 10, 8, 17, 0);
42     ParallelGatewaySplit var_Split = new ParallelGatewaySplit(this, "split",
        getNextParallelGatewaySplitID(), swimlaneID);
43     ParallelGatewayJoin var_Join = new ParallelGatewayJoin(this, "join",
        getNextParallelGatewayJoinID(), swimlaneID);
44     Task var_Task2 = new Task(this, "Task2", swimlaneModelParent.getNextTaskID
        (), 2, 15, 0, 0, 0);
45     Task var_Task3 = new Task(this, "Task3", swimlaneModelParent.getNextTaskID
        (), 5, 5, 0, 0, 0.5);
46     ResourceManager var_RM_Resource_A = new ResourceManager(this, "
        ResourceManager", 4, swimlaneID, 5, 3, 3);
47
48     this.addExternalInputCoupling(this.Swimlane_Resource_A_in0, var_Entry0.in)
        ;
49     this.addExternalInputCoupling(this.Swimlane_Resource_A_in1, var_Entry1.in)
        ;
50     this.addExternalInputCoupling(this.Swimlane_Resource_A_in2, var_Entry2.in)
        ;
51     this.addExternalOutputCoupling(var_Exit0.out, this.
        Swimlane_Resource_A_out0);
52     this.addExternalOutputCoupling(var_Exit1.out, this.
        Swimlane_Resource_A_out1);
53     this.addInternalCoupling(var_Entry0.out, var_Task1.in);
54     this.addInternalCoupling(var_Entry0.outSignal, var_RM_Resource_A.in);
55     this.addInternalCoupling(var_Entry0.outForwardSignal, var_Entry1.inSignal)
        ;
56     this.addInternalCoupling(var_Entry1.out, var_Task1.in);
57     this.addInternalCoupling(var_Entry1.outSignal, var_RM_Resource_A.in);
58     this.addInternalCoupling(var_Entry1.outForwardSignal, var_Entry2.inSignal)
        ;
59     this.addInternalCoupling(var_Exit0.outSignal, var_RM_Resource_A.in);
60     this.addInternalCoupling(var_Entry2.out, var_Join.in);
61     this.addInternalCoupling(var_Entry2.outSignal, var_RM_Resource_A.in);
62     this.addInternalCoupling(var_Exit1.outSignal, var_RM_Resource_A.in);
63     this.addInternalCoupling(var_Task1.out, var_Split.in);
64     this.addInternalCoupling(var_Split.out, var_Task2.in);
65     this.addInternalCoupling(var_Split.out1, var_Exit0.in);
66     this.addInternalCoupling(var_Join.out, var_Task3.in);
67     this.addInternalCoupling(var_Task2.out, var_Join.in);
68     this.addInternalCoupling(var_Task3.out, var_Exit1.in);

```

```

69     this.addInternalCoupling(var_RM_Resource_A.out, var_Entry0.inSignal);
70
71 }

```

Listing H.7: Test 2: Swimlane_Resource_B.java - ports

```

13 public class Swimlane_Resource_B extends BPMNCPModel {
14
15     public InputPort<Entity> Swimlane_Resource_B_in0= new InputPort<Entity>(<
        this);
16     public OutputPort<Entity> Swimlane_Resource_B_out0= new OutputPort<Entity>
        >(this);

```

Listing H.8: Test 2: Swimlane_Resource_B.java - objects and couplings

```

26 public Swimlane_Resource_B(SwimlaneModel parentModel, String description, int
    swimlaneID) {
27     super("Swimlane_Resource_B", parentModel );
28
29     this.description = description;
30     this.swimlaneID = swimlaneID;
31     swimlaneModelParent = parentModel;
32
33     SwimlaneEntry var_Entry0 = new SwimlaneEntry(this, swimlaneID,
        getNextSwimlaneEntryID());
34     SwimlaneExit var_Exit0 = new SwimlaneExit(this, swimlaneID,
        getNextSwimlaneExitID());
35     Task var_Task4 = new Task(this, "Task4", swimlaneModelParent.getNextTaskID
        (), 4, 18, 13, 20, 0);
36     ResourceManager var_RM_Resource_B = new ResourceManager(this, "
        ResourceManager", 3, swimlaneID, 10, 0, 1);
37
38     this.addExternalInputCoupling(this.Swimlane_Resource_B_in0, var_Entry0.in)
        ;
39     this.addExternalOutputCoupling(var_Exit0.out, this.
        Swimlane_Resource_B_out0);
40     this.addInternalCoupling(var_Entry0.out, var_Task4.in);
41     this.addInternalCoupling(var_Entry0.outSignal, var_RM_Resource_B.in);
42     this.addInternalCoupling(var_Exit0.outSignal, var_RM_Resource_B.in);
43     this.addInternalCoupling(var_Task4.out, var_Exit0.in);
44     this.addInternalCoupling(var_RM_Resource_B.out, var_Entry0.inSignal);
45
46 }

```

Verification of prototype behavior

I.1 Test 1: Single-server queuing system

To increase the credibility of the proposed modeling components and corresponding implementations, we verify whether the outcomes of various simulation models representing the same system and specification will result in similar results.

This section discusses the analysis of a single-server queuing system as proposed in [LK00] (see Figure 6.8). The single-server queuing system was modeled both using the in this research proposed and implemented simulation components (see Figure 6.6), as well as with the Arena simulation software (see Figure I.1 for the Arena model). In Figure 6.9 the DEVSDSOL model is shown of the single-server queueing system.

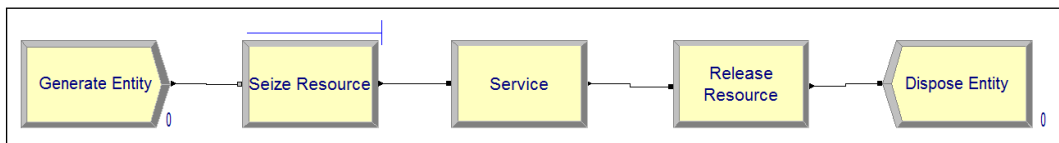


Figure I.1: Arena representation of a single-server queueing system

The specification of the models can be found in Table I.1. The interarrival time of entities is set to 1 minute (exponentially distributed) and the service time is set to 30 seconds (also exponentially distributed). The simulation was executed for 480.0 minutes. We expect that in a simulation model after executing it for 480 minutes approximately 480 entities are generated. Because the service time is only half the interarrival time, one server should be adequate to handle all incoming entities (although because of the exponentially distributed interarrival time and service time, it is expected that some entities will arrive while the server is busy). In the long term (for instance after running the model 480.0 minutes), the server utilization is expected to be approximately 50% (on average the server is busy 30 seconds, while approximately every 1 minute an entity arrives). The server is expected to handle nearly all arriving entities within a simulation run.

Outcomes

In Table I.2 the outcomes of a single-server queueing system simulation model by Law & Kelton is provided (results are of one execution) [LK00]. In Table I.3 and Table I.4 the average outcomes of the model execution by respectively the DSOL and the Arena model are presented (each model was executed 20 times). The outcomes of the DEVSDSOL and Arena models are discussed in the following paragraphs.

Parameter	Value (min)
Length of the simulation:	480.000 min
Mean interarrival time:	1.000 min
Mean service time:	0.500 min

Table I.1: Specification of Single-server queueing system model

Parameter	Value (min)
Entities In	480
Server utilization	0.49471
Average number in queue	0.50864
Average delay in queue	0.47314
Number of delays completed (Entities Out)	480

Table I.3: Average results of 20 executions (DSOL)

Parameter	Value (min)
Entities In	NA
Server utilization	0.464
Average number in queue	0.394
Average delay in queue	0.399
Number of delays completed (Entities Out)	475

Table I.2: Outcomes 1 execution as presented in [LK00]

Parameter	Value (min)
Entities In	490
Server utilization	0.509
Average number in queue	0.5005
Average delay in queue	0.49
Number of delays completed (Entities Out)	489

Table I.4: Average results of 20 executions (Arena)

Entities In After executing the DEVSDSOL and the Arena simulation models 20 times, the average total number of entities generated was calculated. The outcomes of the DEVSDSOL model corresponds with the expected number of entities generated, namely 480. The Arena model generated on average 10 entities more. Based on the outcomes of the T-Test (shown in Figure I.2 for the number of entities generated), we conclude however that the means of the Arena model and the DEVSDSOL model are not significantly different. Therefore we conclude that the Start Event generated entities as intended.

Server Utilization The server utilization is expected to be approximately 50%. The average values are shown in Table I.3 for the DEVSDSOL model and Table I.4 for the Arena model and are both close to 50%. In Figure I.3 the outcomes of the statistical test are shown and a significance value of 0.650. We therefore conclude that the resulting server utilization statistics of the DEVSDSOL-models corresponds with the general expected value.

Average number in queue The average number of entities in the queue of both the DEVSDOL model as well as the Arena model seem higher than the values as presented in [LK00]. However, the value presented in [LK00] is the result of only one model execution. Unfortunately, no average results of more executions are presented by Law & Kelton. We therefore only compare the results of our DEVSDSOL model, with an Arena model (assuming that the Arena model produces valid results). In Figure I.6 the outcomes of

the statistical test are shown, and based on the significance value of 0.789 we conclude that the DEVSDSOL model produces outcomes regarding the average number of waiting entities equal to an Arena simulation model.

Average delay in queue The average delay of entities in the queue of both the DEVSSDOL model as well as the Arena model seem also higher than the values as presented in [LK00]. For the same reasons as mentioned in the previous paragraph, we only compare our DEVSDSOL model with the Arena model. Based on the results of the statistical test (shown in Figure I.6) we conclude that the DEVSDSOL produces outcomes that can be considered equal to the outcomes of the Arena model.

Entities Out Lastly, the number of entities serviced in the DEVSDSOL simulation model has an average of 480 entities. This is equal to the number that was expected.

T-Test: Entities IN; Arena - DSOL				
Group Statistics				
Group	N	Mean	Std. Deviation	Std. Error Mean
EntitiesIN DSOL	20	480.60	23.148	5.176
Arena	20	490.30	23.598	5.277

Independent Samples Test		
Levene's Test for Equality of Variances		
	F	Sig.
EntitiesIN Equal variances assumed	.030	.863
Equal variances not assumed		

Independent Samples Test				
t-test for Equality of Means				
	t	df	Sig. (2-tailed)	Mean Difference
EntitiesIN Equal variances assumed	-1.312	38	.197	-9.700
Equal variances not assumed	-1.312	37.986	.197	-9.700

Independent Samples Test				
t-test for Equality of Means				
	Std. Error Difference	95% Confidence Interval of the Difference		
		Lower	Upper	
EntitiesIN Equal variances assumed	7.391	-24.663	5.263	
Equal variances not assumed	7.391	-24.663	5.263	

Figure I.2: Entities in comparison Arena - DEVSDSOL model

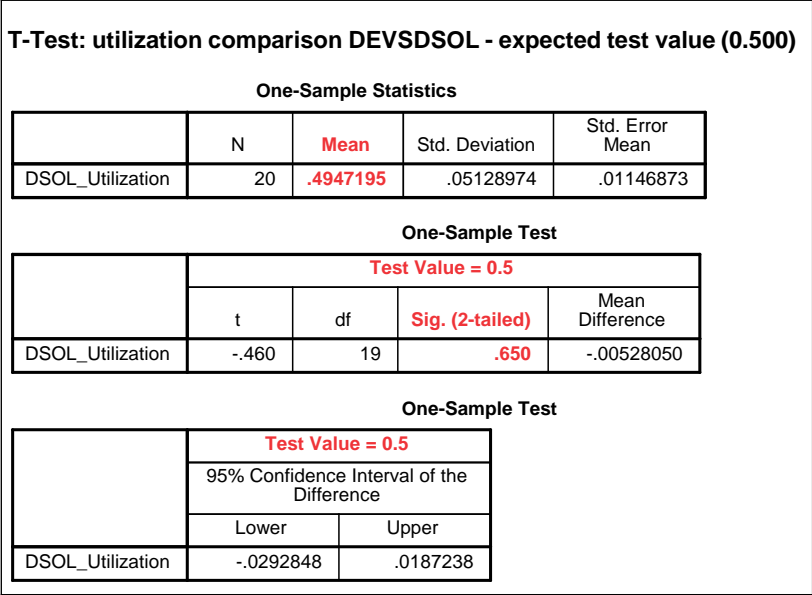


Figure I.3: Server utilization comparison DEVSDSOL model - expected value

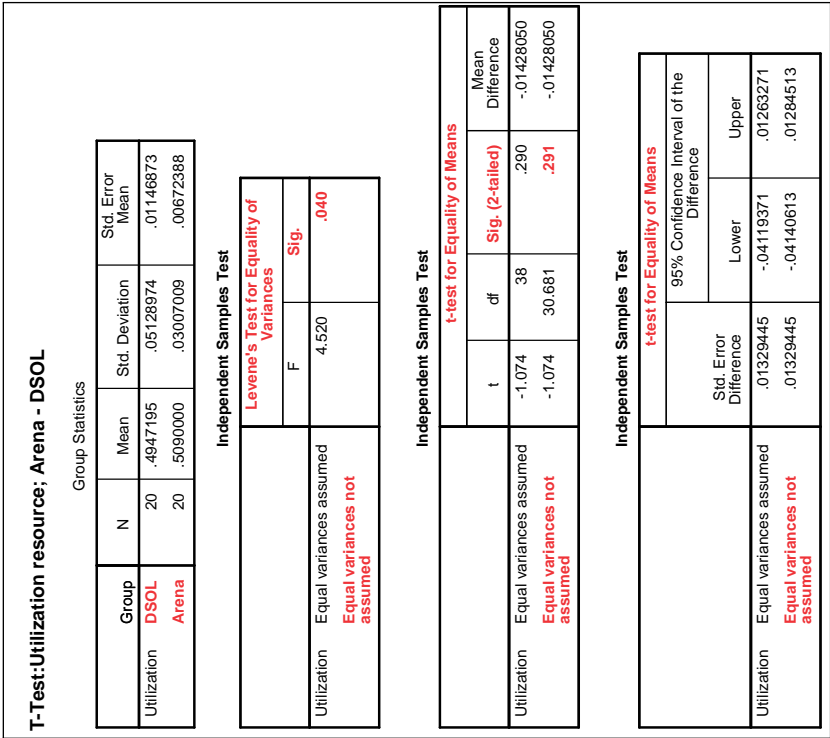


Figure I.4: Utilization comparison Arena - DEVSDSOL model

T-Test: Average number of entities in queue: Arena - DSOL					
Group Statistics					
Group	N	Mean	Std. Deviation	Std. Error Mean	
NumberInQueue	20	.5086425	.09705586	.02170235	
DSOL					
Arena	20	.5005000	.09372272	.02095704	

Independent Samples Test		
Levene's Test for Equality of Variances		
	F	Sig.
NumberInQueue	.003	.954
Equal variances assumed		
Equal variances not assumed		

Independent Samples Test				
t-test for Equality of Means				
	t	df	Sig. (2-tailed)	Mean Difference
NumberInQueue	.270	38	.789	.00814250
Equal variances assumed				
Equal variances not assumed	.270	37.954	.789	.00814250

Independent Samples Test				
t-test for Equality of Means				
	95% Confidence Interval of the Difference			
	Std. Error Difference	Lower	Upper	
NumberInQueue	.03016935	-.05293215	.06921715	
Equal variances assumed				
Equal variances not assumed	.03016935	-.05293459	.06921959	

Figure I.5: Entities in queue comparison Arena - DEVSDSOL model

T-Test: Average waiting time: Arena - DSOL					
Group Statistics					
Group	N	Mean	Std. Deviation	Std. Error Mean	
WaitingTime	19	.4731368	.14255456	.03270426	
DSOL					
Arena	20	.4900000	.09026277	.02018337	

Independent Samples Test		
Levene's Test for Equality of Variances		
	F	Sig.
WaitingTime	2.775	.104
Equal variances assumed		
Equal variances not assumed		

Independent Samples Test				
t-test for Equality of Means				
	t	df	Sig. (2-tailed)	Mean Difference
WaitingTime	-.444	37	.660	-.01686316
Equal variances assumed				
Equal variances not assumed	-.439	30.176	.664	-.01686316

Independent Samples Test				
t-test for Equality of Means				
	95% Confidence Interval of the Difference			
	Std. Error Difference	Lower	Upper	
WaitingTime	.03800044	-.09385935	.06013304	
Equal variances assumed				
Equal variances not assumed	.03843094	-.09533046	.06160415	

Figure I.6: Waiting time comparison Arena - DEVSDSOL model

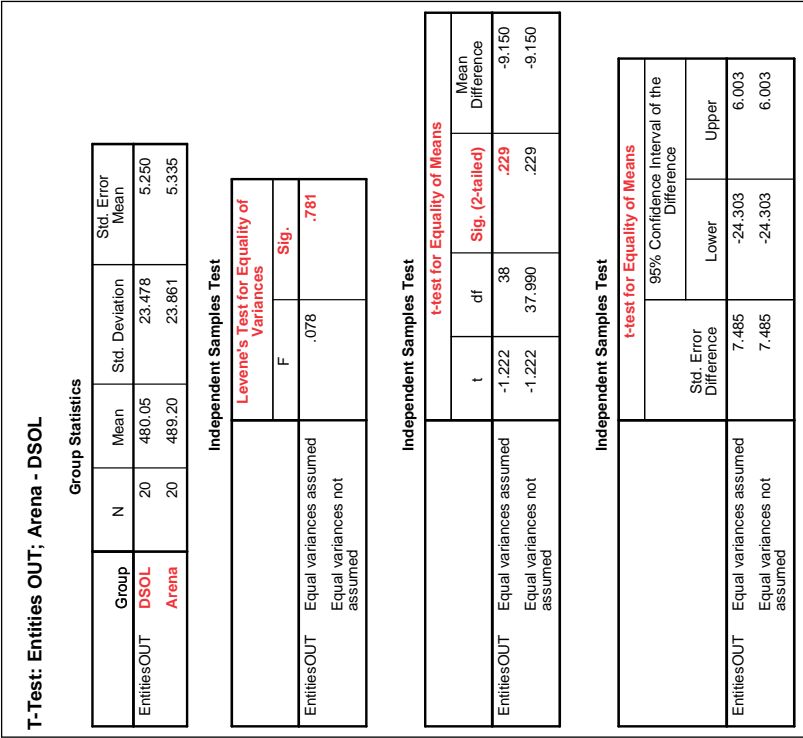


Figure I.7: Entities out comparison Arena - DEVSDSOL model

Listing I.1: Console output run

```

1 Simulator time: 0.0
2 [StartEvent 1]
3   Entity created of type [entity1] with EntityID [1] Waiting time for next
   Entity: 0.12344995003209835
4 [SwimlaneEntry 0, swimlane: 1]
5   Entity arrived [Entity ID 1]; placed in queue; Entities now waiting in
   entry-queue: 1
6 [ResourceManager (swimlane: 1)]
7   Signal arrived: [ origin: 0 ][ destination: -1 ][ resourceFreed: -1 ]
8   signal received: entity added
9   Available resource: 0
10  LMB - entities waiting in queue: 0, namely 1 entities
11 [SwimlaneEntry 0, swimlane: 1]
12  dExt - Signal arrived: Resource available: 0
13  LMB - Entity forwarded to outport. EntityID: 1 with resource: 0
14 [Task 1 by Resource: 0]
15  Entity arrived, ID [1], service time set at: 0.33904792287061014
16  dExt - entity arrived while Task is passive, entity ID: 1
17  Service started: Expected service time (sigma): 0.33904792287061014
18
19 Simulator time: 0.12344995003209835
20 [StartEvent 1]
21   Entity created of type [entity1] with EntityID [2] Waiting time for next
   Entity: 0.37198658007309027
22 [SwimlaneEntry 0, swimlane: 1]
23   Entity arrived [Entity ID 2]; placed in queue; Entities now waiting in
   entry-queue: 1
24 [ResourceManager (swimlane: 1)]
25   Signal arrived: [ origin: 0 ][ destination: -1 ][ resourceFreed: -1 ]
26   signal received: entity added
27   No resource available
28
29 Simulator time: 0.33904792287061014
30 [Task 1 by Resource: 0]
31   Service finished of Entity: 1. Scheduled Service Time: 0.33904792287061014,
   Actual serviced time: 0.33904792287061014 (sigma is :
   0.33904792287061014), with ID: 1
32   [Task 1] dINT: No more entities in the taskList
33   [Task 1] dINT: phases to passive - new sigma: Infinity
34 [SwimlaneExit 0, swimlane: 1]
35   Entity entered the swimlane exit, entityID: 1
36   Resource is made available, Resource: 0
37 [EndEvent 1 ]
38   Entity arrived
39 [ResourceManager (swimlane: 1)]
40   Signal arrived: [ origin: -1 ][ destination: -1 ][ resourceFreed: 0 ]
41   Available resource: 0
42   LMB - entities waiting in queue: 0, namely 1 entities
43 [SwimlaneEntry 0, swimlane: 1]
44   dExt - Signal arrived: Resource available: 0
45   LMB - Entity forwarded to outport. EntityID: 2 with resource: 0
46 [Task 1 by Resource: 0]
47   Entity arrived, ID [2], service time set at: 0.0369382513630963
48   dExt - entity arrived while Task is passive, entity ID: 2
49   Service started: Expected service time (sigma): 0.0369382513630963
50
51 Simulator time: 0.37598617423370645
52 [Task 1 by Resource: 0]
53   Service finished of Entity: 2. Scheduled Service Time: 0.0369382513630963,
   Actual serviced time: 0.03693825136309631 (sigma is :
   0.0369382513630963), with ID: 2
54   dINT: No more entities in the taskList
55   dINT: phases to passive - new sigma: Infinity

```

```

56 [SwimlaneExit 0, swimlane: 1]
57   Entity entered the swimlane exit, entityID: 2
58   Resource is made available, Resource: 0
59 [EndEvent 1 ]
60   Item arrived
61 [ResourceManager (swimlane: 1)]
62   Signal arrived: [ origin: -1 ][ destination: -1 ][ resourceFreed: 0 ]
63 [ResourceManager (swimlane: 1)]   Available resource: 0
64 [ResourceManager (swimlane: 1)]   no entities in queues
65
66 Simulator time: 0.4954365301051886
67 [StartEvent 1]
68   Entity created of type [entity1] with EntityID [3] Waiting time for next
   Entity: 0.24860870142693792
69 [SwimlaneEntry 0, swimlane: 1]
70   Entity arrived [Entity ID 3]; placed in queue; Entities now waiting in
   entry-queue: 1
71 [ResourceManager (swimlane: 1)]
72   Signal arrived: [ origin: 0 ][ destination: -1 ][ resourceFreed: -1 ]
73   signal received: entity added
74   Available resource: 0
75   LMB - entities waiting in queue: 0, namely 1 entities
76 [SwimlaneEntry 0, swimlane: 1]
77   dExt - Signal arrived: Resource available: 0
78   LMB - Entity forwarded to outport. EntityID: 3 with resource: 0
79 [Task 1 by Resource: 0]
80   Entity arrived, ID [3], service time set at: 0.2838441720724808
81   dExt - entity arrived while Task is passive, entity ID: 3
82   Service started: Expected service time (sigma): 0.2838441720724808

```

I.2 Test 2: Single-server queuing system with Exclusive Gateway

In this test, we modeled the business process model as shown in Figure I.8. The specification of the model is the same as in the previous example. However, now we included an Exclusive Gateway element, and specified it with a probability of 75% that entities flow through the top Sequence Flow, and 25% chance that entities flow through the bottom Sequence Flow. We modeled also 2 End Events to count the number of entities that flow through each Sequence Flow.

The model was executed 20 times (same amount of times as the previous example) and the outputs were analyzed. To calculate the percentage of entities leaving through the first exit, we divided the number of entities that were disposed through End Event 1 by the number of entities generated. The results are shown in Table I.5.

Next, the data-set was analyzed using SPSS and a one-sample T-test (using 0.75 as the expected mean-value). The results are shown in Figure I.9. Based on the results and the shown statistics, we concluded that an Exclusive Gateway behaves correctly.

Based on this conclusion we also conclude that a decision based on an attribute-value of an entity also works. It uses namely the same component but a different specification. This was also tested and confirmed, however the results of this test are not included in this document.

Table I.5: Percentage outcomes of 20 model executions

0.72591	0.771084	0.737575	0.715859	0.745387
0.752137	0.752784	0.741348	0.74359	0.755102
0.7375	0.760155	0.749478	0.707965	0.752556
0.761388	0.75442	0.747178	0.760349	0.746171

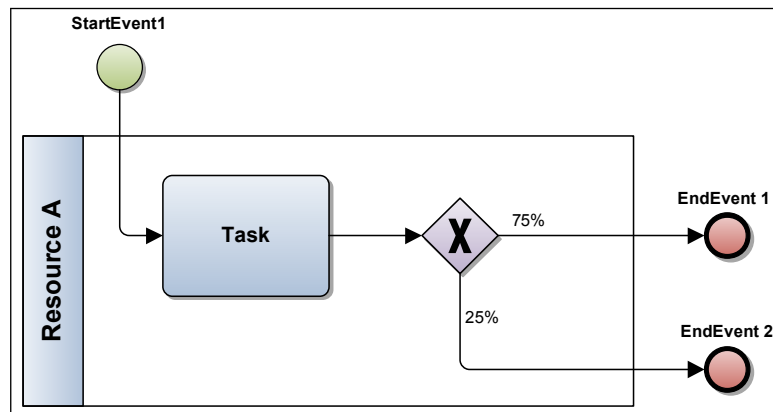


Figure I.8: Test 2: Single-server queuing system with Exclusive Gateway

T-Test: Test 2: Single-server queuing system with probability based decisions**One-Sample Statistics**

	N	Mean	Std. Deviation	Std. Error Mean
OUT1	20	.745905	.0153765	.0034383

One-Sample Test

Test Value = 0.75						
	t	df	Sig. (2-tailed)	Mean Difference	95% Confidence Interval of the Difference	
					Lower	Upper
OUT1	-1.191	19	.248	-.0040950	-.011291	.003101

Figure I.9: Entities in queue comparison Arena - DEVSDSOL model

I.3 Test 3: Parallel activities

In Figure I.10 the business process model is shown with parallel activities. The specification is provided in Table I.6. Finally, the corresponding Arena model is shown in Figure I.11. The outcomes of the comparison are discussed in Section 6.3.2.1.

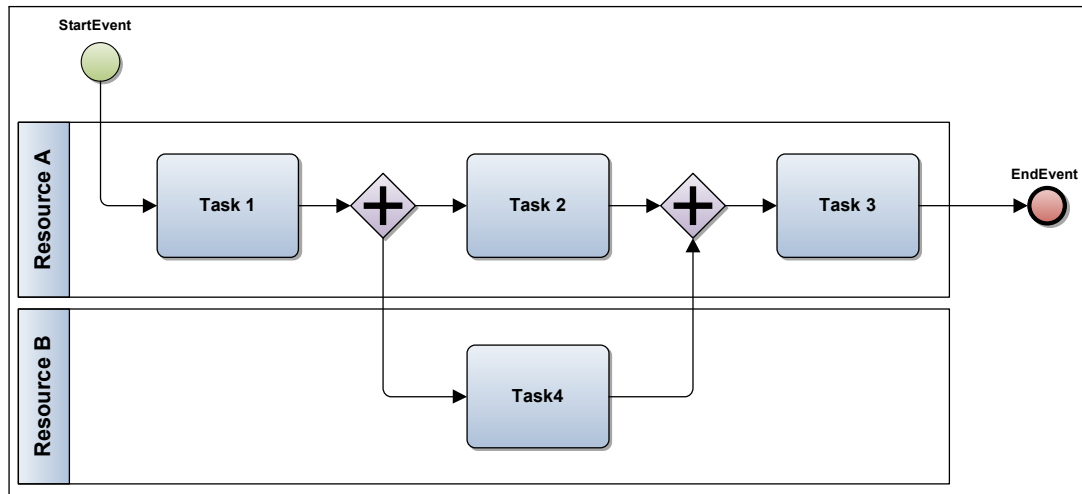


Figure I.10: Entities in queue comparison Arena - DEVSDSOL model

Table I.6: Specification of parallel activities model

Start Event	
Inter-arrival time:	Triangular(Min: 2, Mean: 4, Max: 6)
Batchsize:	1 entity
Time of first arrival:	0:00
Entity type:	Entity1
Task duration	
Task 1:	Triangular (Min: 12, Mean: 15, Max: 17)
Task 2:	Exponential (Mean: 15)
Task 3:	Normal (Mean: 5, Std.dev: 0.5)
Task 4:	Triangular (Min: 5, Mean: 12, Max: 18)
Availability Resources	
Swimlane Resource A:	10 resources
Swimlane Resource B:	4 resources

	Arena	Prototype
Entities		
- Total In	4502	4491
- Total Out	4485	4479
Resource Utilization		
- Resource_A	0.98	0.95
- Resource_B	0.73	0.72

	Arena	Prototype
min	24.339	24.244
max	149.07	159.98
mean	48.562	43.901

Table I.7: Test3: Resource utilization

Table I.8: Test3: Total Time in System

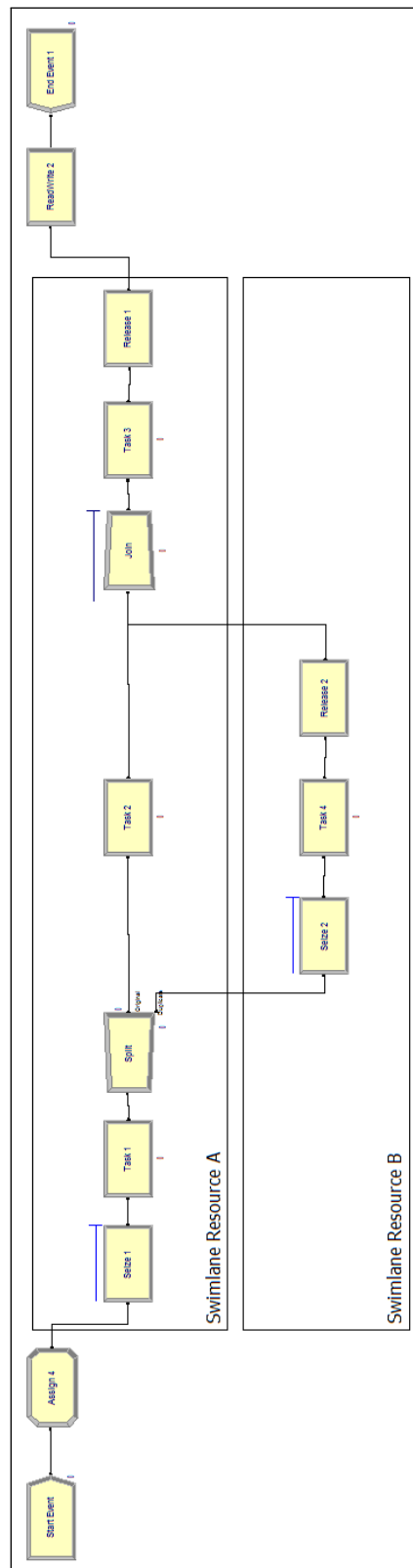


Figure I.11: Test 3: conceptual model prototype

	Arena	Prototype
min	25.285	24.803
max	151.32	155.57
mean	46.831	45.579

Table I.9: Test3: Total Process Time

	Arena	Prototype
min	0.01	0.01
max	118.2	129.8
mean	21.212	16.695

Table I.10: Test3: Total Waiting Time

I.4 Test 4: Parallel activities extended

In Table I.11 the specification is provided of the model shown in Appendix G in Figure G.3. The simulation models was executed for 18 000 seconds and all component parameters are defined in seconds.

Table I.11: Specification of extended parallel activities model

Start Event	
Inter-arrival time:	Triangular(Min: 17, Mean: 22, Max: 28)
Batchsize:	1 entity
Time of first arrival:	0:00
Entity type:	Entity1
Task duration	
Task 'Read email with questions':	Triangular (Min: 10, Mean: 15, Max: 20)
Task 'Answer all questions self':	Triangular (Min: 45, Mean: 70, Max: 120)
Task 'Answer the remain questions':	Triangular (Min: 10, Mean: 25, Max: 70)
Task 'Answer all questions':	Triangular (Min: 100, Mean: 120, Max: 150)
Task 'Answer some questions self':	Triangular (Min: 10, Mean: 30, Max: 50)
Task 'Call customer':	Uniform (Min: 30, Max: 70)
Task 'Join answers':	Triangular (Min: 5, Mean: 10, Max: 15)
Availability Resources	
Swimlane Front Desk:	7 resources
Swimlane Back Office:	3 resources

Modeling and Simulation prototype



J.1 Specification of components

The screenshots in this Appendix are taken from the visual model builder which is part of the prototype. When developing a model with the prototype, some basic parameters should be specified. These are included in the screenshots.

Property	Value
Attributes	
Annotation	
External Link Target	
Name	StartEventName
SpecificationClass	StartEvent
SpecificationLink	0
Start_batchSize	1
Start_firstCreationTime	0
Start_max	0
Start_mean	10
Start_min	0
Start_stdDev	0
StartEntityType	Entity1
StartMode	Exponential

Figure J.1: Properties Window: Start Event

Property	Value
Attributes	
Annotation	
External Link Target	
Name	EndEventName
SpecificationClass	EndEvent
SpecificationLink	0

Figure J.2: Properties Window: End Event

Property	Value
Attributes	
Annotation	
External Link Target	
LaneNumberOfResources	5
LaneQueueMode	Patternx
LaneQueuePatternValue	5
Name	ResourceType1
SpecificationClass	-
SpecificationLink	0

Figure J.3: Properties Window: Swim-lane

Property	Value
Attributes	
Annotation	
CMDiagramType	-
External Link Target	
Name	SwimlaneModel
SpecificationClass	-

Figure J.4: Properties Window: Root model

Property	Value
Attributes	
Annotation	
DecisionEntityType	-
DecisionType	Probability
DecisionValue	0.70
External Link Target	
Name	Decision1
SpecificationClass	ExclusiveGateway
SpecificationLink	0

Figure J.5: Properties Window: Exclusive Gateway - Probability

Property	Value
Attributes	
Annotation	
DecisionEntityType	entityType
DecisionType	Attribute
DecisionValue	Entity1
External Link Target	
Name	Decision2
SpecificationClass	ExclusiveGateway
SpecificationLink	0

Figure J.6: Properties Window: Exclusive Gateway - Attribute

Property	Value
Attributes	
Annotation	
External Link Target	
Name	Split1
SpecificationClass	ParallelGatewaySplit
SpecificationLink	0

Figure J.7: Properties Window: Parallel Gateway Split

Property	Value
Attributes	
Annotation	
External Link Target	
Name	Join1
SpecificationClass	ParallelGatewayJoin
SpecificationLink	0

Figure J.8: Properties Window: Parallel Gateway Join

Property	Value
Attributes	
ActivityDescription	[]
Annotation	
External Link Target	
Name	NameOfTheTask
SpecificationClass	Task
SpecificationLink	0
Task_max	10
Task_mean	7.5
Task_min	5
Task_stdDev	0
TaskMode	Triangular

Figure J.9: Properties Window: Task

Case study: Telecom Provider

K

Due to confidentially reasons, the case-study models are not included in the public version of this thesis

Usability evaluation



L.1 Material and preparations

Figure L.1 depicts the first business process that the consultant was asked to model. Next, the consultant was asked to extend the first model according to Figure L.2. Finally, also a parallel activity was included Figure L.3. The consultant was explained what was the case in an organization, but the models as shown in these figures were not shown to him directly. Goal was to get an understanding of his modeling approach: would he use the modeling elements the same way as were used to draw these business process models.

The feedback that was provided by the consultant during this modeling process, as well as the discussions that were held, are documented in Section L.2 and Section L.3

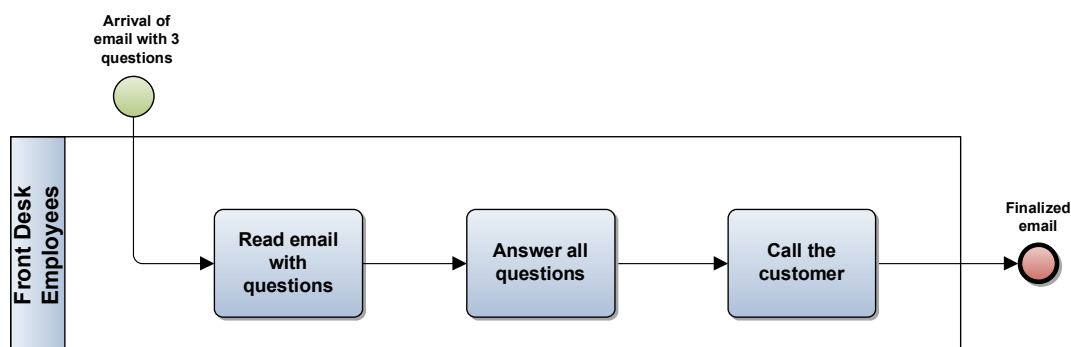


Figure L.1: Sample model usability evaluation: step 1

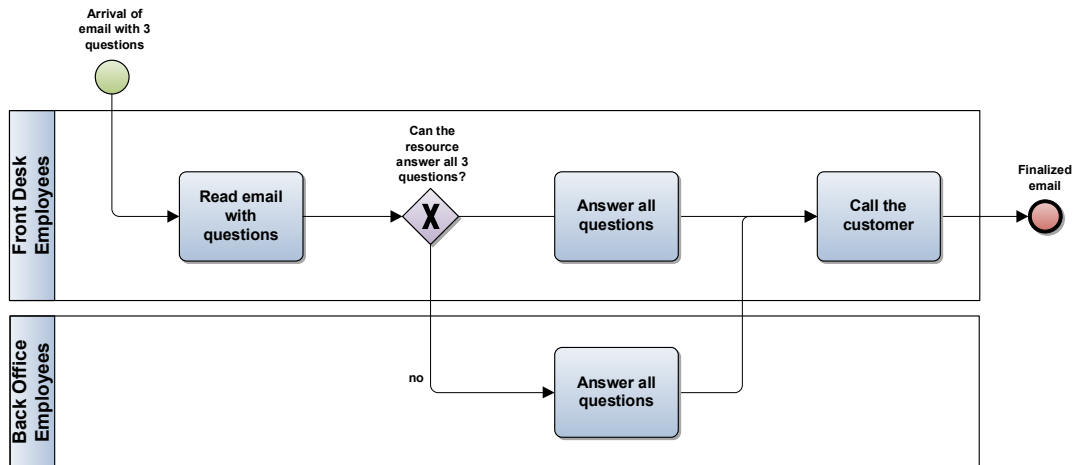


Figure L.2: Sample model usability evaluation: step 2

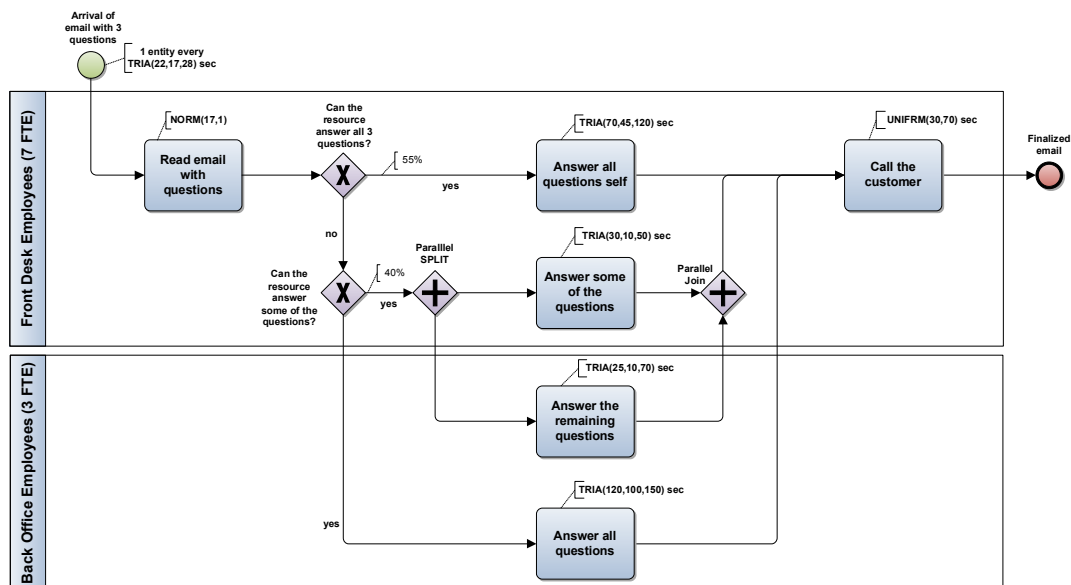


Figure L.3: Sample model usability evaluation: step 3

L.2 Minutes evaluation with I. Wenzler: June 23, 2011

Minutes evaluation session 4 with Ivo Wenzler, 23-06-2011

Outcomes

- Parallel activities can also lead to two outgoing flows within a swimlane, for instance:
 - you have a pool of 10 employees, and one resource needs another one with the same capabilities
- The outgoing flow of a parallel activity, that contains the duplicate entity could automatically change into a different form (e.g. dotted), to clarify that it is the duplicate-outgoing-flow.
 - It should be clear that it is the duplicate-entity.

Evaluation

- Possibilities for improvement / inclusion in next prototype:
 - Automatic resize of the task,
 - suggestion: fixed size, with text wraps
 - Hiding class-names of the components (visually you know which component represents what in a process)
 - There is no snap-to-grid to guide modeling: would be useful to create nice looking diagrams
 - Strange behavior of prototype (sometimes): not always possible to connect two components. Moving components outside swimlane, connecting, and moving them back into the swimlane sometimes fixes this.
 - There is no visual difference between Parallel Gateway Join and Parallel Gateway Split
 - Connecting two components is not intuitive: Ivo is used to drag a connection from one component to another
 - Where to specify the (default) time units?
- Discussion about modeling
 - Main message: don't assign meanings to components that don't have a meaning: basically, think as the resource: he only performs activities (task), and makes decisions
 - A Parallel Gateway Split can't have an activity description
- Parallel activities discussion with an example
 - Three possibilities:
 1. You perform the complete activity ← covered in prototype
 2. Another resource performs the whole activity ← covered in prototype (handover)
 3. You do part of the activity, as well as another resource ← **discussion**
 - How can we model case 3?
 - This case is more difficult / complicated: will a resource wait until the other resource is finished? Why should he/she? Does that actually occurs in business processes (in production processes maybe, but not in business processes?)
 - In case none of the resources will wait for the other: place the parallel join outside both swimlanes: in "hyperspace", or: in another swimlane with the IT system (try to figure out what really happens: where does an entity "wait" and how does it re-enter ones swimlane?)
 - If one of the resources will wait: place the parallel join in one of the two swimlanes
 - Alternative option: add extra attribute and functionality to ParallelGatewayJoin (namely whether a resource will wait, or be released)

- Implication of implementing this, is that ResourceManager should be aware of this queue
- Statistics collection
 - You want to gather statistics about the resources (utilization), entities (time in system, etc), Queues (average waiting time (age), average number waiting)
 - It would be good to be able to execute a model, and change it during execution
 - This way you can see immediately what a change causes

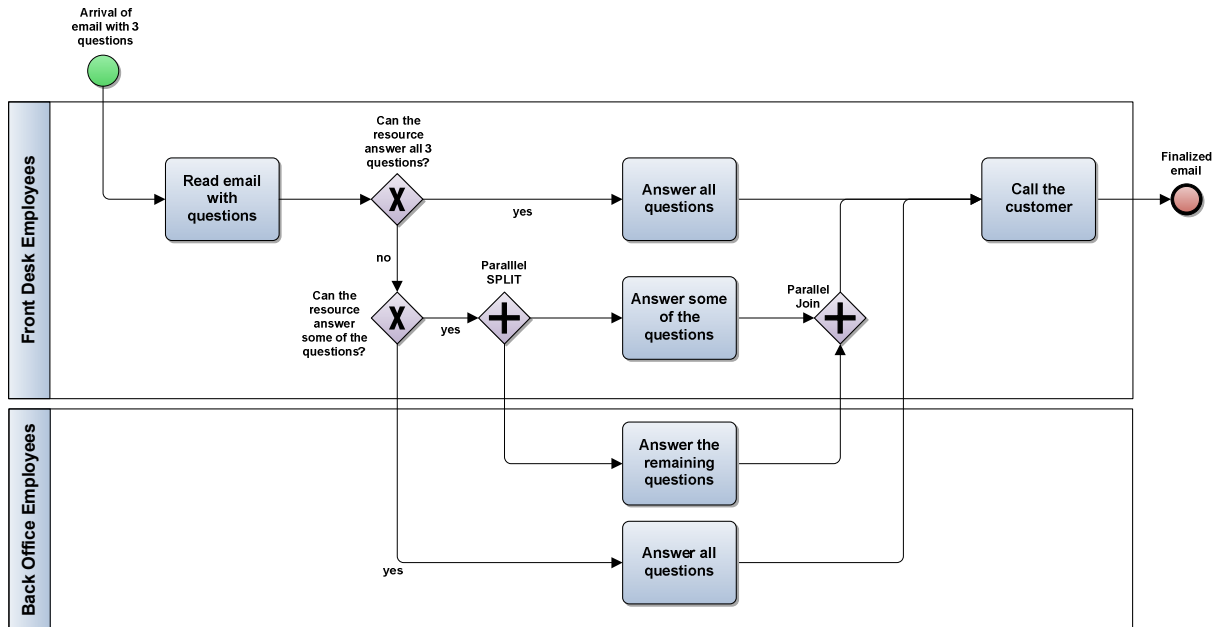


Figure 1 Front Desk resource will wait for Back Office Resource to be ready?

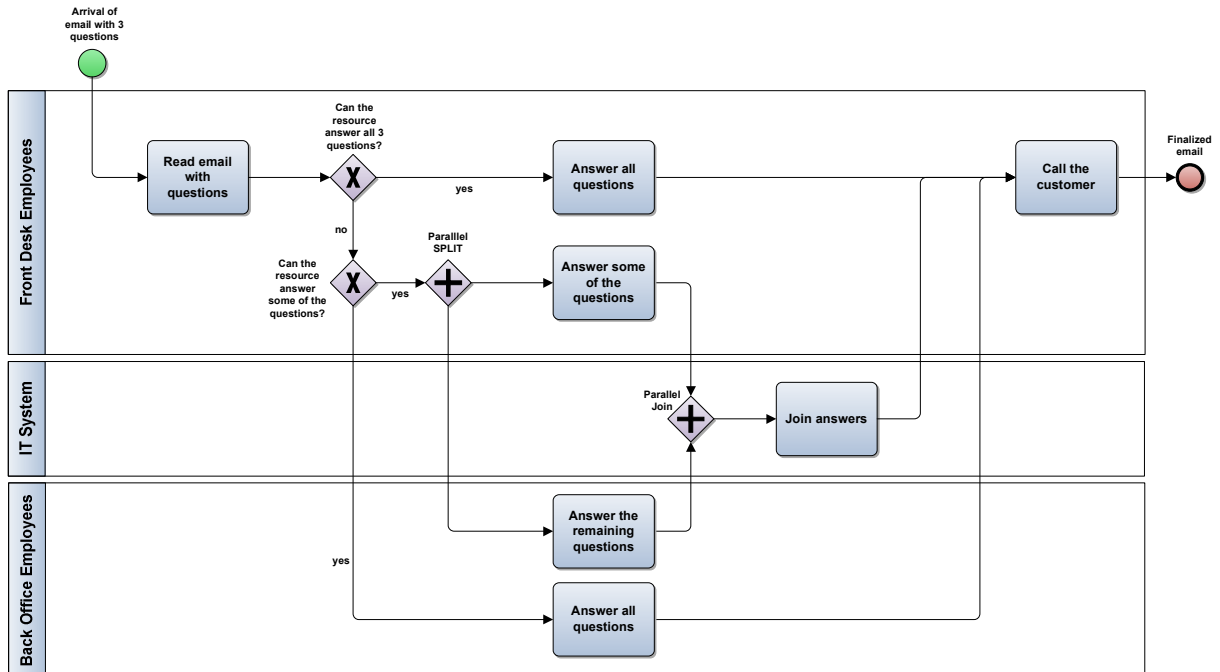


Figure 2 Both resources are released after they finished their activity

L.3 Minutes evaluation with R. Deenen: June 24, 2011

Minutes evaluation session 4 with Rutger Deenen, 24-06-2011

Usability evaluation prototype

- Difficulty with adding new business process elements to an existing business process flow
 - Prototype sometimes shows strange behavior (crash, connectors don't connect)
 - After connector is drawn, you can't connect it to a different component. You need to delete the connector and draw it again
 - A suggestion is that if a component is placed on-top of a sequence flow, the connector is split in two separate flows, leading to, and from the newly added component
 - Moving a component can only be done through mouse click and movement. Suggestion is to also add arrow-key support
- Difference between Exclusive Gateway and Parallel Gateway may be unclear
 - An X-sign marks an Exclusive Gateway, and a plus-sign marks a parallel gateway. Rutger drew (on a whiteboard as a sketch) parallel-gateways when he intended to represent a business process decision (exclusive gateway)

Outcomes / discussion

- Naming of queues (for statistics output)
 - the entry queue can have the name of the first activity that the flow connects with, after crossing the swimlane boundary
 - In case there are multiple incoming flows (and queues) connecting to one activity, than they can be numbered (e.g. Activity: Read e-mail → queue name: "read-email queue 1", "read-email queue 2", etc)
- Decisions can best be represented independently (in case two decisions are made in a real business process), so don't combine two decisions into one gateway-component

Parallel activity queues and behavior

- Rutger also mentions that there are two option in case of a parallel activity: the resource will wait until the second part of an activity is finished, but more likely (occurs more frequently in business processes), is that a resource will continue doing other work, until the other part is finished.
- Rutger recognizes that there are three queues in a parallel join gateway (see Figure 2): 2 queues (q1 and q2) for the separate entity flows (for the original and duplicate entity), and one queue (Q1) for the finished combined entity (after both activities are fulfilled), until a resource is available and can continue the work
- Possibly this queue has a different (higher) priority to claim a resource (compared with the other swimlane entry queues).
- To keep this consistent with the design principle (swimlane crossing = queue), a parallel join can be placed outside the swimlane in "hyperspace" (on top of the boundary), or in a third swimlane. Then, Q3 (see Figure 2) will be equal to the swimlane-entry queue
- If it's placed within a swimlane, the resource will wait until the second activity is fulfilled.
- Exception: if a resource finishes and the second activity was already finished, than logically this resource will immediately continue this work (instead of being released, and possibly picking up work at another queue). Question is, does it matter if another resource picks up work, instead of the "original" one? All resources within a swimlane are 'identical' and for simulation, it doesn't matter (in general cases)

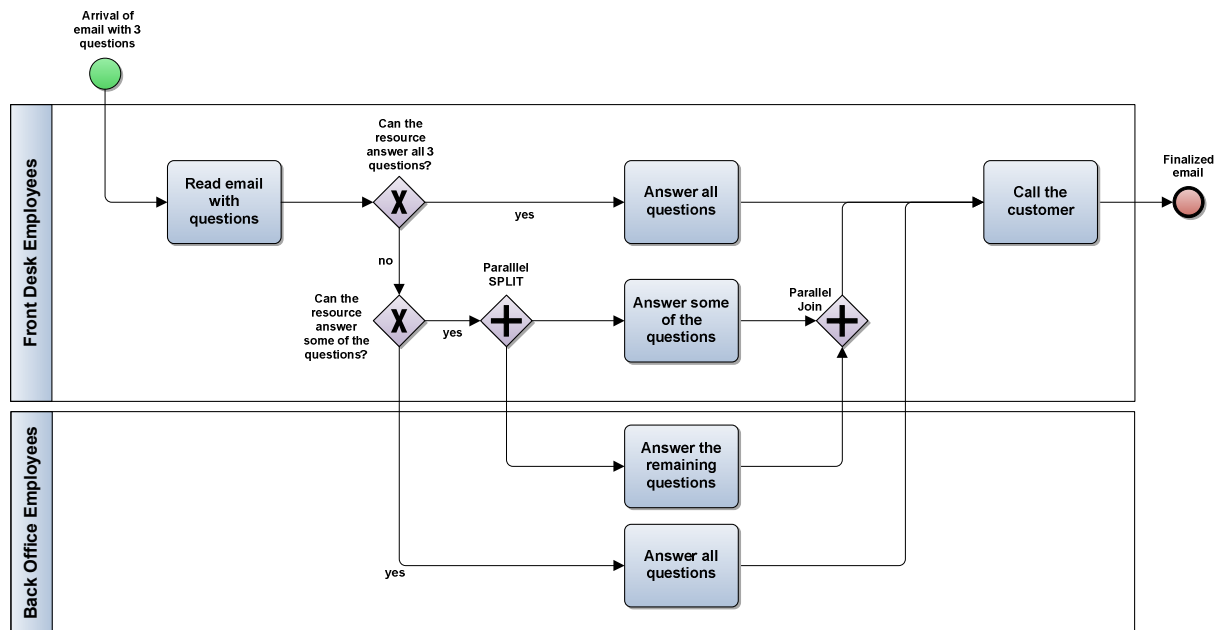


Figure 1 Front Desk resource will wait for Back Office Resource to be ready?

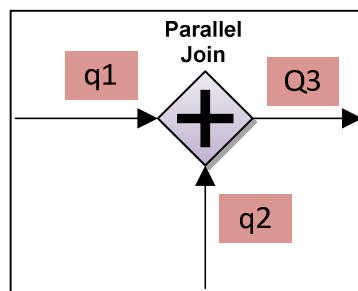


Figure 2 Number of queues for a Parallel Gateway Join: 3

L.4 Received questionnaires

The received questionnaires are included at the following pages.

Questionnaire

Usability of the suggested modeling approach

Please answer the following questions about the proposed modeling approach, while considering both the interaction with the prototype as well as the proposed modeling language and components

Please elaborate on your answer

Do you find it easy to learn to work with the prototype and the components?

Yes, the prototype is very intuitive and works in a similar way as business process modeling software like Visio and Visual Paradigm.

Do you find the proposed modeling approach efficient in use?

The model building is very efficient, only the model specification in this version is still inefficient (f.i. you need to know the exact distribution formulas to specify a task)

Do you think you will easily remember how to work with the prototype and the components, when you will use it again?

Yes, at least for the model building I will, the specification activities can be harder.

Do you consider working with the prototype and the modeling components as being pleasant?

Yes, apart from the bug in reusing a model (swimlane / task combination). I would prefer the situation in which I can reuse an existing model and expand it. Now I needed to build a new model from scratch.

Do you consider the proposed modeling approach (modeling components and way of using the prototype) easy to use to develop a business process model?

Yes, the logic is similar to the real life logic in the business process. It forces you to really think how the reality works.

Do you think the proposed modeling approach can work well in a real situation with involved clients (usage of the tool, understandability of a model, communication about a model)?

If it all works, it would work well in a real situation.

Bugs and errors are killing in a situation where you want the client to understand the model and communicate about the model in a client setting.

Questionnaire

Usability of the suggested modeling approach

Please answer the following questions about the proposed modeling approach, while considering both the interaction with the prototype as well as the proposed modeling language and components

Please elaborate on your answer

Do you find it easy to learn to work with the prototype and the components?

Please consider the differences (pro's / cons) between alternatives: Visio (does this also have simulation functionality?), DSOL and Arena:

- Visio is flexible but difficult to use consistently and maintenance: specification of attributes, can be imported in Access / Excel
- Arena is even more flexible but you need a higher level of expertise to work with the application
- DSOL has the opportunity to fill in the gap to have an easy-to-use simulation tool, but needs to be further developed for practical usage at consulting projects.

Do you find the proposed modeling approach efficient in use?

Yes, DSOL can be a suitable approach to process modeling depending on the situation. Please consider some scenarios to further develop the tool, in consulting context:

- sizing (FTE) modeling and related variables like regions, jobs, etc.;
- traditional process modeling in companies which has more focus on the whole lifecycle cq. Maintenance costs and requirements;
- process improvements projects (like Lean Six Sigma) that require short timelines, client workshops and validation meetings.

Do you think you will easily remember how to work with the prototype and the components, when you will use it again?

N.A.

Simulation was held; I have not worked with the system myself.

Do you consider working with the prototype and the modeling components as being pleasant?

N.A.

Simulation was held; I have not worked with the system myself.

Do you consider the proposed modeling approach (modeling components and way of using the prototype) easy to use to develop a business process model?

Yes, to highlight the ease-of-use, a limited set of modeling components is used to model business processes, some remarks:

- use of inter-arrival time vs. arrival rate
 - o it is more clear to use the arrival rates to expressing parameters, as this related more to clients and is thus more understandable. Often analysis is done for a period of time (what is the X per year / per month, etc);
 - o in Lean Six Sigma projects the arrival rate (e.g. X per hour) is used; a methodology which is more and more used to model processes in operational excellence initiatives.
- competing queues (resource choice for specific queue)
 - o Work in progress vs. directly added value
 - o If entities are almost finished, it is "generally" preferred to finish these first, before start working on newly arrived entities (lower inventory in progress levels and thus less "money in the system")

Do you think the proposed modeling approach can work well in a real situation with involved clients (usage of the tool, understandability of a model, communication about a model)?

The current focus of DSOL is easiness to use the application, with the pitfall that flexibility is less considered. From my experience, every project has unique requirements in modeling the situation so it is important to be able to adapt the elements for additional components:

- Visuals are very important for usability during workshops (e.g. Lean Six Sigma value stream mapping sessions, queuing simulations at supply chain departments);
- Use of modeling components: example is usage of queues to see where inventory entities are hold as this creates a direct understanding to the client / analyst of process bottlenecks;
- Possibility for animations of the flow of entities: used for verification during workshops, investigation of bottlenecks, etc. Without animations, preparation time in between modeling, verification and client discussions are more needed;
- Etc.