

Round-trip
Business Process Driven SOA
modelling between ARIS and Cordys

Master's thesis, 13 January 2010

Melissa Cheung BSc.

Round-trip Business Process Driven SOA modelling between ARIS and Cordys

THESIS

submitted in partial fulfilment of
the requirements for the degree of

MASTER OF SCIENCE
in
COMPUTER SCIENCE
TRACK INFORMATION ARCHITECTURE

by

Melissa Yuen Shan Cheung
born in Rotterdam, The Netherlands



Web Information Systems Group
Department of Software Technology
Faculty EEMCS, Delft University of Technology
Delft, The Netherlands
<http://eemcs.tudelft.nl>



IDS Scheer Nederland BV
Loire 162, gebouw B
2491 AL Den Haag
<http://www.ids-scheer.nl>

Round-trip Business Process Driven SOA modelling between ARIS and Cordys

Author: Melissa Yuen Shan Cheung
Student id: 1228161
Email: myscheung@gmail.com

Abstract

Business process modelling is the core activity in Business Process Driven SOA. ARIS is a Business Process Analysis (BPA) tool adequate for analyzing and designing business processes, while the execution and monitoring of these processes is empowered by Cordys as Business Process Management Suite (BPMS). The challenge is to enable iterative round-trip modelling across these two tools. Event-driven Process Chains (EPC) and Business Process Modeling Notation (BPMN) have established themselves as the most used modelling languages in the industry. Coupling ARIS and Cordys involves 1) transforming the high-level business process models in EPC developed in ARIS into executable process models in Cordys BPMN, and 2) ensuring round-trip development by means of interoperability.

A conceptual framework is proposed to couple a BPA and BPMS tool for round-trip business process modelling. The framework utilizes concepts from the Model Driven Architecture for structural addressing interoperability and model transformations. Business process models are developed and assessed from high-level business, operational and technical viewpoints. Depending on the level of interoperability wished for, different types of model transformations within and between these viewpoints apply. A combination of methods from state-of-the-art literature will present how to achieve model transformations between business process models. By ensuring interoperability with traceability enables managing the perception on the real world with different viewpoints.

The framework is used for the ARIS and Cordys case. Analyzing and comparing the tools have given insight into types of models, the information for exchange, and where possible coupling points are. Model transformations are defined for EPC and Cordys BPMN models. The results of the framework provide a theoretical base for achieving interoperability between ARIS and Cordys.

Keywords: business processes, BPM, SOA, model transformations, MDA, workflow patterns, ontology

Graduation Committee:

Prof. dr. ir. G.J. Houben, Faculty EEMCS, TU Delft

Dr.ir. A.J.H. Hidders, Faculty EEMCS, TU Delft

Ir. G.H.J. Van Gent, Manager, IDS Scheer

Dr.ir. J. van den Berg, Faculty TPM, TU Delft

Acknowledgements

During my Master Computer Science Information Architecture I was introduced to the paradigm of enterprise agility. This master is a collaboration between the faculty EEMCS and TPM at Delft University of Technology. I have gained a lot of knowledge by learning from two different faculties with different expertises.

My thesis is executed at IDS Scheer on this topic from April 2009 - January 2010. I am very thankful that I had the opportunity to gain some practical experience, at a company with much expert knowledge in the area of supporting enterprises in achieving agility.

There are many people who I would like to thank for educating and supporting me throughout this thesis.

First, I would like to thank the members of the University. I thank my daily supervisor, Jan Hidders. He provided feedback and guided me on scientific grounds during my thesis. His support and valuable advice were of great help during the execution of the thesis. Also thanks to Geert-Jan Houben, for his feedback, advice and encouragements. His evaluation of my preliminary research has provided me with direction and foundation for this thesis. And thanks to Jan van den Berg, for the advice he has given during the last project in my masters (the preparation project for the thesis), and for providing feedback on my thesis project.

I would like to thank IDS Scheer, the company who has presented me with the opportunity to do my thesis. Although he is not employed at IDS Scheer anymore, I would like to thank Marcel Heijnen, who was my supervisor and had guided me during the first part of my thesis project. I am very grateful to his successor, Gerard van Gent, for guiding and supporting me during the latter half of my thesis project. I appreciate the time and effort he has invested in evaluating my thesis report, the discussions we've held, and the advice he has given. I would also like to express my gratitude to Eric Roovers and Jeroen Havenaar for sharing their expert knowledge and experience on this topic and their feedback. Eric had helped me by critically evaluating my thesis report, and supported and educated me in meetings with regard to content. Jeroen had explained service oriented process modelling in detail to me, and commented on example transformations. Many thanks also to all the other employees at IDS Scheer who have shared their knowledge and provided tool trainings.

The thesis was executed at IDS Scheer, but would not be possible without cooperation of Cordys. I would like to thank Henk Rietveld and Erwin Nooteboom for their willingness to participate in this project and their investments. I am particularly thankful for Erwin's support and feedback. Thanks for evaluating the thesis report, evaluating model transformations, facilitating interviews with other Cordys employees and support during training of the Cordys tool. Many thanks also to the other Cordys employees for sharing their knowledge.

I am grateful that both IDS Scheer and Cordys facilitated interviews with clients who have given me a very enlightening impression of how the tools are deployed in the company. And of course, thanks to these clients for sharing their experiences.

Last but not least, I thank my parents, brother, and friends for encouraging me during this thesis.

Melissa Cheung,
13 January 2010, Delft

Contents

Acknowledgements	iii
Contents	v
List of Figures	vii
List of Tables	ix
Chapter 1 Introduction	2
1.1 Problem description	2
1.2 Research strategy	4
1.3 Deliverables	4
1.4 Outline	5
Part I. Introduction to Business Process Driven SOA	6
Chapter 2 Business Process Driven SOA	7
2.1 A theoretical BPD SOA life cycle	7
2.2 Perspectives and roles	8
2.3 Strategies for BPD SOA adoption	9
2.4 Tooling for BPD SOA	11
Chapter 3 Process modelling for BPD SOA	13
3.1 Event-driven Process Chain	13
3.2 Business Process Modeling Notation	14
3.3 Views on Business Process Modelling	16
Part II. Conceptual Framework for BPD SOA round-trip modelling.....	18
Chapter 4 Conceptual Framework for BPD SOA round-trip modelling across two tools	19
4.1 Preliminary: process governance and qualitative modelling	19
4.2 Context analysis	21
4.3 Dimensions	22
4.4 Model transformation	24
4.5 Maintaining interoperability: traceability	27
4.6 Overview of the framework	29
Chapter 5 Approaches for model transformations	31
5.1 Ontologies and meta-models	31
5.2 Patterns in business processes	33
5.3 Related work	35
5.4 Horizontal transformation	37
5.5 Vertical transformation	38
Part III. Using the framework for ARIS and Cordys	42
Chapter 6 BPD SOA modelling coupling for ARIS and Cordys	43
6.1 Context analysis	43
6.2 Dimensions	46
Chapter 7 Model transformations and Traceability for ARIS and Cordys	48
7.1 Horizontal transformation	48
7.2 Exchange formats	58
7.3 Vertical transformation	58
7.4 Traceability for ARIS and Cordys	62
7.5 Validation	65

Chapter 8	Interoperability between ARIS and Cordys.....	66
8.1	<i>Current tool support in model exchange</i>	66
8.2	<i>Design-time versus run-time</i>	66
8.3	<i>Step to step ARIS EPC to Cordys BPMN</i>	67
8.4	<i>Design considerations for IDS Scheer and Cordys</i>	69
Part IV. Conclusions & Recommendations	72
Chapter 9	Conclusions	73
9.1	<i>Design-time in ARIS and Run-time in Cordys</i>	73
9.2	<i>Round-trip between ARIS EPC and Cordys BPMN process models</i>	74
9.3	<i>Ensuring iterative modelling between ARIS and Cordys</i>	74
9.4	<i>Concluding remarks</i>	75
9.5	<i>Contributions</i>	75
Chapter 10	Recommendations.....	76
10.1	<i>Limitations & Future Work</i>	76
Bibliography	78
List of Abbreviations	86
Reflection	88
Appendices	90
Appendix A.	ARIS Business-Driven SOA by IDS Scheer	91
Appendix B.	Cordys Business Process Management Suite	98
Appendix C.	Roles in BPD SOA	103
Appendix D.	Meta-model analysis EPC & BPMN.....	105
Appendix E.	Evaluation of BPMN and EPC as modelling languages	110
Appendix F.	Assessment of non-main EPC and C-BPMN objects	113
Appendix G.	Structural Design Patterns for EPC and BPMN	115
Appendix H.	ARIS EPC to BPMN transformation support	127
Appendix I.	Parameterized patterns Domain Example	129

List of Figures

Figure 1 Coupling ARIS and Cordys	3
Figure 2 Business Process Driven SOA life cycle	8
Figure 3 EPC example of invoice process and EPC constructs	14
Figure 4 BPMN example and constructs legend	15
Figure 5 Stakeholders in process modelling	16
Figure 6 Process Modelling with views, roles, phases and modelling languages	17
Figure 7 Levels of Conceptual Interoperability Model (LCIM) [Turnitsa 2005]	23
Figure 8 Interoperability [Cimander and Kubicek 2009].....	24
Figure 9 MOF meta-model structure.....	25
Figure 10 Directionality in transformation (uni and bidirectional).....	25
Figure 11 Scenarios in transformation (Integration, Translation and Synchronization).....	25
Figure 12 Decomposed transformation between two tools	26
Figure 13 Model transformation overview.....	27
Figure 14 Traceability concepts	28
Figure 15 Position of BPA BPMS modelling coupling.....	30
Figure 16 Metamodel and Ontology hierarchy	31
Figure 17 Patterns in business processes overview.....	35
Figure 18 Horizontal mapping approach	37
Figure 19 DSML and Pattern based Vertical transformation (based on [Brahe and Bordbar 2007]).	39
Figure 20 Relation Meta-model, DSML and Ontologies	40
Figure 21 BPD SOA Life cycle comparison	43
Figure 22 High level mapping ARIS and Cordys models (and properties) based on description	44
Figure 23 ARIS-Cordys coupling points against theoretical life cycle	45
Figure 24 Point Process Modelling related to ARIS and Cordys process modelling	46
Figure 25 Conceptual transformations in ARIS and Cordys	47
Figure 26 Part of translation of EPC to C-BPMN (Appendix G.2).....	57
Figure 27 Position of DSML approach in POINT.....	59
Figure 28 Example approval pattern based transformation.....	62
Figure 29 Meta-model Traceability for ARIS and Cordys (concept).....	63
Figure 30 Simple traceability example between EPC to BPMN, to C-BPMN	64
Figure 31 ARIS design-time and Cordys run-time	67
 <i>Appendices</i>	
Figure 32 IDS Scheer BPM life cycle.....	91
Figure 33 MDA and ARIS services	94
Figure 34 ARIS service oriented EPC constructs	95
Figure 35 ARIS models relations	96
Figure 36 Service allocation diagram.....	96
Figure 37 Process Modelling activities Service Oriented EPC and associated diagrams	97
Figure 38 Cordys BPM and roles.....	99
Figure 39 Cordys Business Process Model Constructs.....	100
Figure 40 Cordys BPM properties	100
Figure 41 Cordys Process Modelling activities.....	102
Figure 42 EPC meta-model	105

Figure 43 BPMN metmodel core	106
Figure 44 BPMN meta-model foundation	107
Figure 45 Cordys BPMN meta-model	109
Figure 46 Basic patterns: start event, sequence path and end event	115
Figure 47 Decision Parallel Patterns	117
Figure 48 Decision Alternative Patterns	118
Figure 49 Decisions Multiple Alternative Patterns	119
Figure 50 Combination of Decision Patterns	121
Figure 51 Example Operator-Operator in Decision pattern [J. Mendling 2009].....	122
Figure 52 BPMN translated example Figure 51	123
Figure 53 BPMN workflow patterns alternatives [Wohed, et al. 2006]	124
Figure 54 EPC example [IDS_Scheer_Academy 2009]	125
Figure 55 C-BPMN translated from EPC	126
Figure 56 EPC "Request Bank Account"	129
Figure 57 C-BPMN "Request Bank Account"	130
Figure 58 Technical C-BPMN "Request Bank Account"	131
Figure 59 Approval Pattern Transformations domain specific	136
Figure 60 Informative Service Pattern Transformations domain specific	136
Figure 61 Performative Service pattern transformation domain specific	136

List of Tables

Table 1 Practitioner roles in BPD SOA life cycle.....	9
Table 2 Perspectives, phases and roles in BPD SOA	21
Table 3 Conceptual Framework for coupling BPA and BPMS tool for modelling - overview	29
Table 4 Business Domain Ontologies: description and focus (extended and edited [Filipowska, et al. 2009])	33
Table 5 Activity Patterns in Business Processes [Thom, et al. 2008]	34
Table 6 Related work Approaches for business to IT transformation	36
Table 7 Example of parameterized patterns for vertical tranformation	41
Table 8 SO EPC and C-BPMN meta-level mapping	51
Table 9 Workflow pattern operators comparison EPC – BPMN	52
Table 10 Structural Design Patterns EPC – C-BPMN	53
Table 11 Structural Design Patterns C-BPMN to EPC	54
Table 12 Linking the process tasks for “Request Bank Account”	60
Table 13 Derived DSML Task Types for “Request Bank Account”	60
Table 14 Identified domain specific parameterized patterns based on “Request Bank Account”	61

Appendices

Table 15 Summary ARIS Business Driven SOA	92
Table 16 Summary AVE and roles	93
Table 17 Cordys BPM: phase, steps, deliverables, and models and modelling activities	98
Table 18 Cordys BPMN 2.0 compliance [Gakkhar 2009]	101
Table 19 Roles comparison ARIS and Cordys in BPD SOA.....	104
Table 20 Bunge-Wand-Weber EPC-BPMN comparison	110
Table 21 Workflow Patterns EPC-BPMN comparison [van der Aalst, et al. 2003].....	112
Table 22 Assessment Web service mapping.....	113
Table 23 Assessment Data flow mapping.....	113
Table 24 Assessment UI mapping.....	113
Table 25 Assessment Role mapping	114
Table 26 Assessment KPI mapping	114
Table 27 Assessment Risk & Control mapping	114
Table 28 ARIS EPC2BPMN evaluation	128
Table 29 Approval User Parameterized pattern EPC Business to Functional	132
Table 30 Informative Service Parameterized pattern EPC Business to Functional.....	132
Table 31 Performative Service Parameterized pattern EPC Business to Functional	132
Table 32 Approval Parameterized Pattern C-BPMN Business to Functional	133
Table 33 Performative Service Parameterized Pattern C-BPMN Business to Functional.....	133
Table 34 UserTask Parameterized patterns C-BPMN Functional to Executable.....	134
Table 35 Informative ServiceTask Parameterized patterns C-BPMN Functional to Executable.....	134
Table 36 Performative Service Task Parameterized patterns C-BPMN Functional to Executable.....	135

Chapter 1

Introduction

This master thesis researches how ARIS and Cordys can be coupled for Business Process Driven Service Oriented Architecture (BPD SOA) modelling. ARIS Software by IDS Scheer, global leader in independent business process and performance management, provides support for business-driven SOA design. The SOA Solution offered by IDS Scheer supports application scenarios for service architecture, service orchestration and process automation and service and application development. IDS Scheer is named a Leader in Enterprise Architecture, Business Process Analysis (BPA) and IT Planning Tools by Independent Research Firm Forrester [IDScheer 2009]. Cordys is a global provider of software for business process innovation and Enterprise Cloud Orchestration. “The Cordys platform and its cutting-edge Cloud technology empower customers to dramatically improve the speed of change, fundamentally altering the way they innovate their Business Operations to achieve a true customer-centric philosophy [Cordys 2009].” The platform includes an open, integrated set of tools and technologies including Composite Application Framework (CAF), Master Data Management (MDM) and a SOA Grid. Cordys is classified as a Business Process Management Suite (BPMS). BPMS tools offer an executable business process management environment.

1.1 Problem description

ARIS by IDS Scheer is acknowledged as a BPA tool, adequate for analyzing, optimizing and documenting business processes for business process management (BPM) purposes. Cordys is a BPMS, enabling the management and execution of business processes. Although both tools are designed for BPM and SOA, there is a clear difference in scope. ARIS is suitable for managing organizations with business processes from an analytical perspective. Cordys has a strong technological base for operational BPD SOA. Considering BPD SOA development on high-level, there are two exchange points for ARIS and Cordys: in the analysis/design phase and monitoring/optimization phase. Coupling ARIS and Cordys is to improve interoperability between both tools. In other words, coupling ARIS and Cordys is enabling the exchange of business process models on syntactic, semantic, and technical level. Several challenges will arise as I deal with tools that differ in scope and support for modelling standards.

The industry has claimed that BPM and SOA can be used in synergistic sense to create great business value [Catts and St. Clair 2009] [Cheung 2009]. However, there is not one tool for BPD SOA development and execution. Current problem is *lack of a coherent and collaborative tool environment*. From a business point of view, improving the interoperability of both tools will reduce time-to-market from modelling to execution. Hereby, at one hand, it reduces the time from business requirements to IT specifications. On the other hand, it minimizes deployment activities and manually maintaining multiple versions of one truth.

When focusing on the gaps from a modelling perspective, two major issues arise in attempt to couple ARIS and Cordys for Business Process Driven SOA development. The first major issue is a *functional modelling gap*. There is a gap between the business model designed in ARIS and the model for execution in Cordys. Both ARIS and Cordys can process the same modelling notation, Business Process Modelling Notation (BPMN), but there is still a functional modelling gap between the models representing business needs and what is necessary for IT delivery. In addition, considering BPD SOA development in practice, Event-Driven Process Chain (EPC) has established itself as popular modelling language for business due to its ease-of-use and the success of ARIS. Therefore, achieving syntactical and semantic interoperability between EPC and BPMN is part of addressing the functional gap.

The second major issue is to *ensure iterative round-trip Business Process Driven SOA modelling* when coupling ARIS and Cordys. Iterative modelling is process model development with continuous improvement. Round-trip indicates

that process models can be exchanged between the two tools at any time. Round-trip is required as design changes may be processed in one tool, and in the end it needs to be synchronized with the other tool, as the process model is supposed to represent one single truth. After design in ARIS (design-time) and execution in Cordys (run-time), the services are monitored for redesign and optimization. This information may lead to redesign and optimization (change-time) in ARIS. The redesigns can in turn be executed in Cordys. There is no need to redo all activities necessary for execution and start the whole process again.

Figure 1 illustrates the situation in coupling ARIS and Cordys. In the BPA tool the major efforts in design-time take place, determining and linking strategic objectives with functional process models. In ARIS the design-time modelling languages are EPC and BPMN. The functional models need to be exchanged to Cordys for run-time. Cordys requires BPMN based process models. These are in turn transformed into executable BPMN for execution. During execution process instances are monitored. These indicate the performance of processes and whether they align with the strategic objectives. However, coupling ARIS and Cordys for monitoring is out of scope. The focus is on exchanging and transforming process models, as indicated by the dark red arrows.

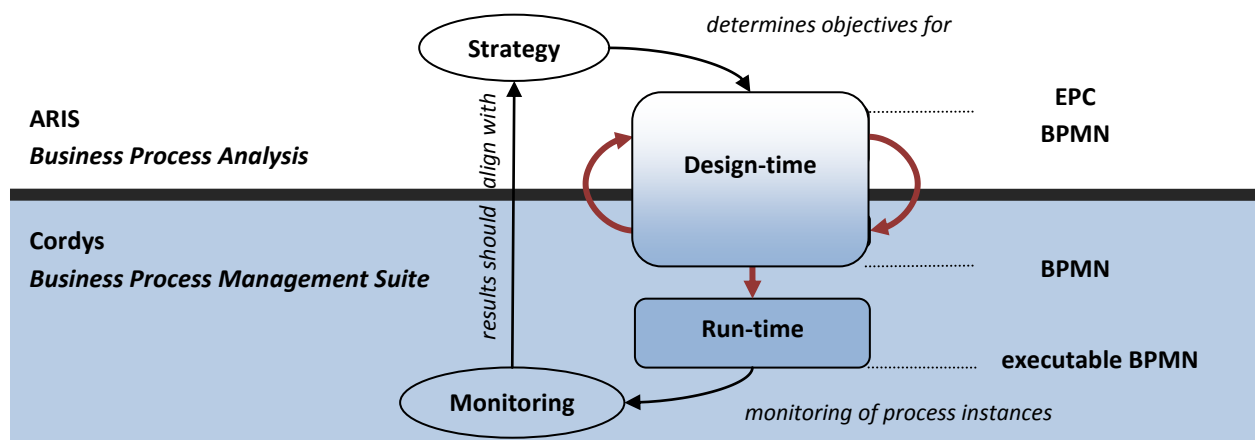


Figure 1 Coupling ARIS and Cordys

These two major issues address a number of challenges. A systematic approach is required to identify all perspectives and dimensions in order to reduce the complexity in coupling ARIS and Cordys and bridge the functional modelling gap. Therefore, this thesis will determine:

To what extent is iterative round-trip Business Process Driven SOA modelling possible by coupling ARIS, a business process analysis tool, and Cordys, a business process management based execution tool?

The research question is decomposed in three sub-questions:

- What is Business Process Driven SOA management with tool support?
- What conventions can be determined to ensure a round-trip between the EPC models designed in ARIS from business perspective, and the BPMN based models which can be executed in Cordys?
- Which functional requirements can be determined for ARIS and Cordys to ensure iterative modelling in Business Process Driven SOA context?

This thesis will elaborate on *how* to address the functional modelling gap and round-trip modelling within a conceptual framework. The framework is used to propose a solution to bridge the gap between ARIS and Cordys business process modelling.

1.2 Research strategy

Several interesting research topics are present within this thesis. The domain of BPD SOA is widely growing in the industry to manage organizations and help them survive in a globalizing and dynamic competitive environment. Understanding the concept of BPD SOA and how tools such as ARIS and Cordys can contribute to this is essential. From user perspective it is not always clear what the distinction is between ARIS and Cordys. Identifying all phases, activities, and roles provides a theoretical basis for comparison. The *first phase* in this research is analyzing ARIS and Cordys as tools in order to determine overlap and differences. Analysis results will show what information is relevant in BPD SOA and where possible interchange moments exist. This contributes in determining the different couple points between the tools. This is achieved by studying available documentation materials and special ARIS (version 7.1) and Cordys (version BOP-4) workshops. Both literature and vendor perspectives on BPD SOA will contribute to context awareness. The *second phase* is to research the relevant modelling notations in BPD SOA. This involves researching EPC and BPMN on syntactic and semantic level. Furthermore, qualitative modelling will also be of importance since business processes need to be executed. Research will also be conducted on the related work in model transformations. With the preliminary research a conceptual framework will be developed, which guides BPD SOA development with BPA and BPMS platform from modelling perspective in the *third phase*. The framework is applied for ARIS and Cordys and modelling transformations are executed for EPC and BPMN. Examples will illustrate the concepts. Interviewing a client with both ARIS and Cordys tools will also provide necessary insights. Experts of ARIS and Cordys will validate the theory. In the *fourth phase* recommendations will be made for ARIS and Cordys on how they can achieve interoperability, and determine what needs are where.

1.3 Deliverables

Initially this research was requested from a client of IDS Scheer and Cordys. A pilot project is the initiative for this thesis. In this pilot project IDS Scheer and Cordys have tested whether process models could be exchanged between the tools. This thesis will elaborate on conceptually defining how to map the process models and round-trip development in Business Process Driven SOA development. The problem of business to IT transformation is well-researched for the cases of EPC or BPMN to BPEL. This case is different as is depicted that both high and low level models are described in one modelling language, BPMN. By analyzing the ARIS and Cordys case, a general concept can be derived for coupling BPA and BPMS tools. The relevant dimensions, such as different perspectives, models, data sets and levels of interoperability, will be depicted. A conceptual framework will be developed to guide the creation of a coherent collaborative development environment by coupling a BPA and BPMS tool for BPD SOA.

Deliverables (and section):

- Context analysis of Business Process Driven SOA with BPA and BPMS tools (Part I & III).
- Conceptual Framework guiding BPD SOA modelling between BPA and BPMS (Part II).
- State of the art methods for “Business to IT transformation” (Part II).
- Model transformation ARIS and Cordys specific models (Part III).
- Best practices for model transformations ARIS and Cordys (Part III).
- Recommendations for most feasible technical realization (Part III & IV).

1.4 Outline

The thesis is structured into four parts:

Part I introduces the concept of Business Process Driven SOA. It provides insight on what BPD SOA is in Chapter 2, including life cycle and roles, adoption scenarios, and tool support. Within the scope of the thesis the modelling phase of BPD SOA is researched in Chapter 3. This includes the modelling languages EPC and BPMN and the viewpoints in modelling.

Part II presents a conceptual framework for BPD SOA modelling across two tools. The framework provides a structure to identify all relevant dimensions in attempt to couple a BPA and BPMS tool in Chapter 4. Model transformations are a core element of the framework. The approaches for model transformation are described in Chapter 5.

In *Part III* the framework is used on the ARIS and Cordys case. Chapter 6 depicts what scenarios for model transformation are relevant for ARIS and Cordys. The model transformation approaches are applied for ARIS EPC and Cordys BPMN in Chapter 6. In Chapter 8 achieving interoperability for ARIS and Cordys as tools is addressed.

Part IV presents the conclusions in 0. Recommendations and future work are found in Chapter 10.

Part I.

Introduction to Business Process Driven SOA

Chapter 2

Business Process Driven SOA

Business Process Driven (BPD) SOA relates to the world of Business Process Management (BPM) and Service Oriented Architecture (SOA).

Business Process Driven SOA is to improve business processes within organizations for business agility enabled by a loosely coupled service infrastructure.

BPD SOA revolves around business processes. A *business process* is a coherent sequence of activities that produces a product or service, taking one or more inputs in order to create output of value for the customer [Hammer and Champy 1994]. BPM is a discipline which focuses on improving business processes within organizations in order to function more effectively, efficiently and adaptable to the dynamic globalizing environment. BPM addresses process improvement by quality measurement, awareness for processes and customer value, organization performance and alignment with strategy defined by management [Smith and Fingar 2002]. SOA is a form of distributed systems architecture with services as agents. It consists of a set of components which can be invoked, and whose interface descriptions can be published and discovered [W3C, Booth and Haas, Web Services Architecture 2004].

In the business domain, BPM and SOA are considered as concepts which can be used complementary for creating great business value. BPM can be rapidly implemented by SOA, and BPM provides SOA with a strong business-case and better alignment with the business. BPM-SOA combination can be used as enterprise-wide approach for business performance improvement. Both encourage reuse and adaption to the dynamic competitive environment [Cheung 2009].

Introducing BPD SOA is done by presenting the life cycle (2.1), the perspectives and roles (2.2), adoption strategies (2.3), and tool support in BPD SOA (2.4).

2.1 A theoretical BPD SOA life cycle

A theoretical BPD SOA life cycle is based on the workflow process life cycle [M. Z. Muehlen 2004]. This life cycle is decomposed to the right abstraction level representing the process aspects within the life cycles of BPM and SOA. Figure 2 depicts the life cycle.

The *strategy* phase is concerned with achieving consensus on what objectives are supposed to be met with business processes, and the development of a strategic plan accomplishing these objectives. Within this phase the definition and analysis of the business goals, environment and organization need to be formalized. *Process design* consists of *analysis* and *modelling*. Process analysis is to analyze specific as-is processes and its context. Design and modelling of to-be process models is subsequent to analysis to depict desired situations. This phase specifies the process structure including resources and responsibilities. To-be processes can be validated by *Process simulation* on their functional behaviour. Simulation is considered as a separate phase, because it initially may be used to validate designs, but also used to validate for performance optimization after evaluation. *Process implementation* involves the design of the infrastructure to support the business processes, and the deployment of the business process. *Process Enactment* involves the coordination of the individual process instances derived from the process models and executing the process. An executable business process consists of human and system tasks orchestrated over a software platform. *Process Monitoring* measures performance and time of execution. These measurements are subject for evaluation. *Process Evaluation* regards analysis of process instances. The results can be used for planning and redesign, and adjustments within processes can be validated with *simulation*. The results from evaluation are reported to management and business for redesign, and the following iteration.

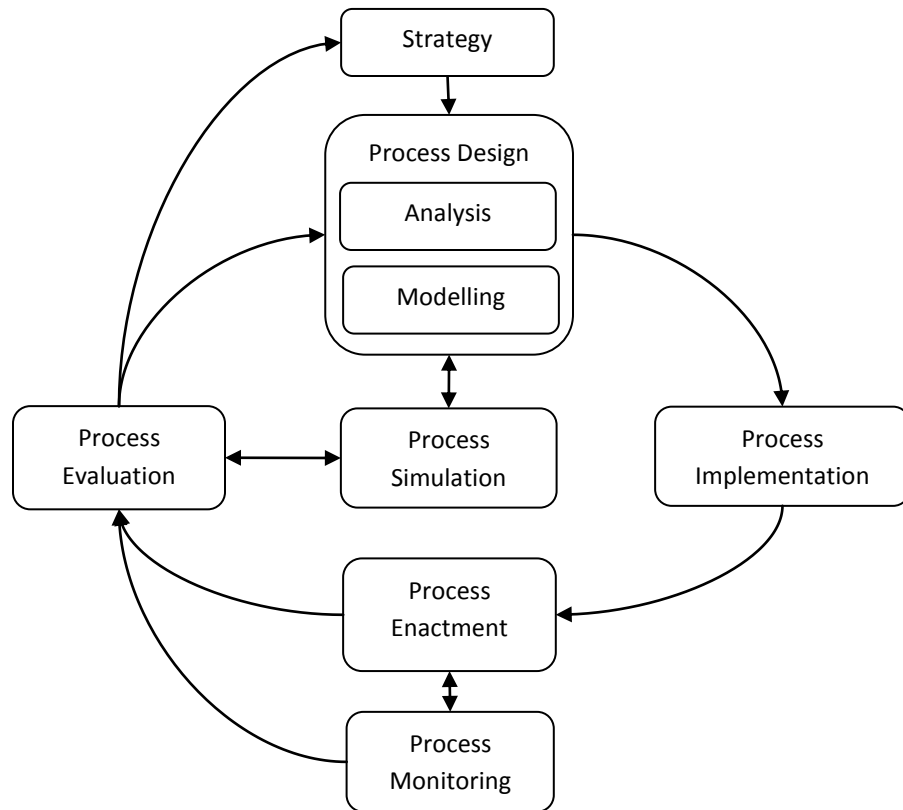


Figure 2 Business Process Driven SOA life cycle

2.2 Perspectives and roles

Business processes can be looked upon from perspectives of different stakeholders. Insight on the different perspectives can provide awareness of the different stakeholders and their needs. Stakeholders are *researchers* developing methods addressing Business Process Modelling (BPMo) issues, tool *vendors* offering BPMo software and consultancy, and the *practitioners* who design, and use process models. Each stakeholder has its own perception on how business processes can be used for business agility. A Delphi Study by [M. Indulska, et al. 2009] showed the perceptions on the current issues and future challenges (within five years) of BPMo for these stakeholders are not completely in line. The main issues identified by the practitioners are not top priority challenges for the academics. The researchers are visionary, as the challenges they identify are not considered by the practitioners. Vendors have the same critical issues and challenges as practitioners. However, across the stakeholders there is a similarity between the top three critical issues and challenges, to wit: standardization of process notation, methods and tooling, established value of BPMo, and support for model-driven process execution based on process models.

The Delphi study recommends adjusting industry needs and research topics for business process modelling. By adjusting the business process modelling maturity within organizations can be improved and the creation of research topics will be stimulated. Vendors can provide the proper tool support and guidance in industry adoption. Acceptance and adoption of the theoretical developments by the industry is essential. The methods designed in theory for BPMo issues lack acceptance of the practitioners because the benefits are usually not tangible presented. Another aspect is when management or tool support from organizations is not adequate for fully deploying BPMo solutions. The absence of a holistic approach of BPMo for business process analysis, execution, mining and optimization is identified as the main hurdle for acceptance of theoretical developments in practice [Vergidis, Turner and Tiwari 2008].

2.2.1 The practitioners

Many discussions have been held on what one's responsibility is in BPD SOA, as converging disciplines require collaboration and coordination between two domains. Convergence requires a mind-set change within certain roles influencing the method of working. An example: a business analyst has to identify the opportunities SOA offers, which requires an understanding of SOA from business, architecture and technical perspective. [Seeley 2008] [Craig 2006] However, this set of capabilities need to be developed in collaboration with IT analysts/architects due to the different perspectives. Some IT analysts/architects claim that the business analyst focus on users needs clashes with the service boundaries [Morganthal 2009]. Another phenomenon is the rise of a role with cross-domain expertise, the solution architect, converting architecture design into IT solutions [Bogue 2005]. The function is to be the bridge between business and IT, understanding the complexity of the business and the ability of solving business problems with service oriented IT solutions. The roles present in BPD SOA are depicted in Table 1. They are mapped to the life cycle phases. Detail and definitions of each role and their responsibilities is given in Appendix C.

Strategy	Process analysis	Process design	Process implementation	Process enactment	Process monitoring	Process evaluation	Process simulation
Business Professional	Business Analyst	Process Architect	<ul style="list-style-type: none"> · IT Analyst · IT Architect · IT Developer 	IT Developer	Administrator	<ul style="list-style-type: none"> · Business Analyst · Process Analyst 	<ul style="list-style-type: none"> · Business Analyst · Process Analyst

Table 1 Practitioner roles in BPD SOA life cycle

2.3 Strategies for BPD SOA adoption

There are several adoption strategies for BPD SOA. The strategies for process modelling and service discovery are determined by the scope of a BPD SOA project. Strategies include top-down, bottom-up and middle-out adoption [Meijler, Kruithof and van Beest 2006].

The top-down strategy is adopting BPD SOA project from business perspective. Business users design the processes and the high level services which should be implemented by IT. Business processes are analyzed, designed and modelled to gain insight in the organizational structure and workflow. High level services are designed from business perspective. From here IT can derive its requirements for implementation. In other words, it is *forward engineering* in a Greenfield site. In software development Greenfield is to not have constraints imposed on the project by previous IT initiatives. The business objectives are aligned with the IT services, however, a challenge remains at governing the efficient service implementation. A scenario is when BPMo is adopted in order to gain insight in the organization and improve the alignment with strategic objectives. IT execution is then initiated from business perspective.

Bottom-up scenario is that the IT implementations form a foundation for the business processes to be developed, also considered as *reverse engineering*. IT drives the creation of reusable services on top of legacy systems. These services are the basis for the high level services and business processes. This way the legacy is closely aligned with the high level services. Processes are heavily influenced by legacy systems, as they may depict structures unfit for purpose due to the legacy systems. An example is when implemented workflow projects have to leverage their executable workflow processes to business processes for alignment with business objectives.

Middle-out is a combination of top-down and bottom-up strategy; coupling business aligned process realization from top-down strategy with services and legacy systems derived from bottom-up strategy. Top-down and Bottom-up strategies are most suitable in projects, while middle-out strategy is suitable for enterprise wide adoption. The scope of the project and the context determine which strategy is most suitable. An illustrative case suitable for middle-out strategy is displayed in Example 1.

State of BPD SOA at Dutch Insurance Company

A Dutch Insurance Company has both ARIS and Cordys. Executive management has imposed that ARIS should be the new platform for business process modelling from business perspective. They have introduced ARIS to manage their organization and involved information. Essential is improving the collaboration between different departments from organizational perspective. ARIS has a model objects library and reporting capabilities required for business analysis, which in turn will improve and change the organization. Enterprise Architecture is adopted to coordinate the development of these business processes. Business conventions are developed to have standardized terminology and language. Furthermore, organizational conventions have been developed to coordinate the interaction in process modelling and maintain coherency and consistency. A concurrent development is that Cordys has proven to be strong platform for business process execution, and is chosen as execution tool at this company. Operational processes can be developed and refined to technical processes which in turn can be executed. From business perspective the processes, which are yet to be developed, eventually will have to be executed by the execution platform at Dutch Insurance Company. Concurrently, several business process modelling activities have been started on the Cordys platform. For certain departments the operational processes are already documented and made executable. As the architecture activities are just initiated, communications between both developing teams at Dutch Insurance Company have not been initiated. Two scenarios can be distinguished here. At one hand, the business will develop processes in ARIS which eventually need to be executed at Cordys. On the other hand, the business processes already documented in Cordys will have to be aligned with the strategic and business objectives. These implemented processes have instances for monitoring at run-time which should match the process indicators and key performance indicators. This requires synchronizing model transformations between ARIS and Cordys. It is evident that every change and optimization in the models should be synchronized between the platforms. The Dutch Insurance Company already underwent many transformations while merging with other companies and departments. Therefore, both the business and IT team have their own reference models regarding service design, discovery and orchestration. Due to the transformations and mergers, reorganization from business perspective is necessary to improve efficiency and business agility, and legacy systems need to be incorporated in their processes.

Example 1 A glimpse of the state of BPD SOA (Interview 20 October 2009, Dutch Insurance Company)

The case of the Dutch Insurance Company illustrates the complexity within an organization adopting both BPM and SOA. A few observations by the author based on the example above:

- The maturity of Enterprise Architecture (EA)/BPM/SOA development determines the need for coupling, and the degree of interoperability. The concepts and relations of EA, BPM and SOA are elaborated in the preliminary research assignment [Cheung 2009].
- Communication and collaboration is essential to determine what is needed and (really) happening. Projects often start in parallel while the objectives exist to reduce redundancy and increase consistency, but due to the lack of communication this problem is not yet solved. Currently, the business conventions for modelling in ARIS do not align with modelling in Cordys.
- Budget and executive management determine the course of action.
- Legacy systems have major influence on the processes, as the interviewed have indicated that sometimes processes seem to have structures unfit for purpose due to the legacy systems.

As illustrated in the Dutch Insurance company example adopting BPD SOA is not straightforward. BPM and SOA are originally not designed as complementary concepts. BPM originates from business management theory, while SOA is an IT concept. There are several challenges in the adoption of middle-out strategy [Brahe 2007] [Kamoun 2007]:

- When both business and IT see the benefit in collaboration, a *mindset change* is required for successful adoption. Architects may have problems with fully employing BPM solutions, while process owners do not completely comprehend SOA and its interaction with BPM.
- A proper level of *service granularity* is another point of attention in BPD SOA development. From BPM point of view, fine grained services are difficult to use and manage in meaningful applications, but coarse grained services can be managed and located through associated meta-data. However, coarse granularity reduces the degree of flexibility in applications. It is necessary to specify the right level of granularity matching the business components.
- When using BPM and SOA as unified approach, another challenge rises in *terminology*. Standardization of technical terms is necessary, as the same terminology can mean different things within SOA and BPM. Example terms are business process, business service, business practice, and business component.
- Another challenge is to *design for scalability*, regarding the continuous improvement by iterations with both BPM and SOA. For enterprise-wide solutions it is important that versioning is considered in order to manage the growth of business processes and services.

A BPD SOA solution needs the right executive support, governance policies and clear accountabilities. The commitment from all involved organizations in realizing a BPD SOA solution is most important. For this it is necessary to understand the benefits of a BPD SOA approach. The governance policies should address the people, authorizations, risks and controls. Accountability is an important point for discussion when involving two communities. Separation of concerns will provide a structured approach for coupling. The Dutch Insurance Company case showed consensus on terminology and BPD SOA purpose is another necessity.

2.4 Tooling for BPD SOA

BPD SOA consists of BPM and SOA, each proposing different requirements on the modelling environment. Much analysis has been done on tools for BPD SOA by analyst researchers from Gartner and Forrester. They have introduced the terms BPA and BPMS which are now widely known in the industry.

A Business Process Management Suite (BPMS) has evolved from workflow centric technologies [Cantara 2008]. Their original roots determine the set of components they support. They originate from the following applications: Human Workflow, BPM Pure-Play, Enterprise Application Integration (EAI)/Middleware, Document Management, Business Process Outsourcers. Most recognizable components of BPMS are according to the analysts:

- A process and state execution engine
- Business Activity Monitoring
- System management and Administration
- Document & Content Management
- User & Group collaboration
- Simulation facilities
- Business Rule Management

BPA tool have the following characteristics according to the analysts [Blechar 2008]:

- Support for documenting, analyzing and organizing complex processes, in order to better understand their processes at a more abstracted level of detail.

- Developed for business analysts and business process architects defining business processes on conceptual level, but they offer support for users in enterprise with business and technical modelling expertise.
- The BPA tool is positioned between an EA and BPMS tool. Support both architectural facilities as more workflow oriented. It complements a BPMS tool by enabling conceptual, logical and physical modelling in greater detail and relationship mapping.
- Links with to the workflow assembly and orchestration engines and business activity monitoring (BAM) tools

A BPA tool supports provides more analytic and strategic facilities than a BPMS. A BPMS is more technology oriented, as it provides an execution platform for processes. But there is also an overlap of functionality, to wit process modelling. Analysis of ARIS and Cordys can be found in Appendix A, respectively Appendix B. ARIS and Cordys are researched their point of view regarding BPD SOA, including the life cycle, roles, and involved process models. A comparison is found in section 6.1.

A tool must offer certain capabilities and usage patterns to support one discipline. But when the discipline is not supported by one vendor, different tools must provide a coherent collaborative environment for integration and transition. There are several issues which can arise at model-based tool integrations [Kapsammer, et al. 2006] :

- The difference in scope of the models. Different domains can be supported by a tool.
- The difference in representation. There are many modelling languages and ways to represent a concept by means of a model. Meta-model mapping between models is required for integration, and the differences in syntax and semantics need to be addressed.
- The difference in exchange format. Tools offer support for exchange of models between different tools. The standardization of exchange formats and the support of both tools for these standards is a challenge. Thus compatible standards are required between both tools.

Chapter 3

Process modelling for BPD SOA

Process modelling is the core activity in BPD SOA. The models traverse from the business domain, business process models, to the IT domain, executable process models. This addresses different level of granularity, different terminology, and different modelling approaches. Business process modelling is possible in different modelling languages and from different point of views (3.3). The open standard modelling language is Business Process Modelling Notation (BPMN), developed by the Business Process Management Initiative (BPMI) and maintained by the Object Management Group (OMG). BPMN is the open standard for modelling business processes, but in the industry the Event-driven Process Chain (EPC) is also often used thanks to the wide customer-base of ARIS and ease of understanding. Executable models are usually in Business Process Execution Language (BPEL) developed by OASIS. BPEL is an executable language specifying interactions with Web services. The Cordys executable models are BPMN based. This thesis focuses on bridging a functional modelling gap between business models in ARIS and executable models in Cordys. Therefore, EPC (3.1) and BPMN (3.2) are analyzed in depth.

3.1 Event-driven Process Chain

Event-driven Process Chains (EPC) capture business logic with simplicity to increase the ease-of-understanding for business people. EPC was developed in 1992 as business process modelling language for reference modelling in SAP R/3 ERP systems by Keller, Nüttgens and Scheer for SAP AG [van der Aalst, Desel and Kindler 2002]. Reference models document generic business operations of a specific domain. The developed language was extensively used for defining reference models in the SAP Reference Model. This has amongst others led to the popularity of EPC for business process modelling. Deduced from the number of publications in the scientific world, the SAP Reference model and EPC had made an impact. Scientists saw potential of the use of EPC in other domains. Another reason for the high adoption of EPC in practice is because of its simplicity and it is supported by an extensive number of tools, including ARIS (IDS Scheer) and SAP with a wide-spread customer base.

EPC was developed in the Architecture of Integrated Information Systems (ARIS) framework to reduce the complexity of business processes [Scheer and Schneider 1998]. The ARIS methodology distinguishes four perspectives which can be represented by and come together within an EPC. Within ARIS, main and extended elements of EPC are defined. The main elements of EPC are events, functions, connectors and arcs [Green and Rosemann 1999]. A sequential flow of events and functions represent the logical dependencies of activities in business processes. Functions represent the activities, and events are pre- and postconditions of these functions. An event is a state which should occur before the function may be executed. A function, a decision, in turn can trigger an event. Connectors depict logical join or split of a flow, allowing more complex control flows within the model. Connectors types are AND, OR and XOR. The arcs are the links between the functions, events and connectors. Furthermore, process interfaces depicts a link between consecutive EPCs. Figure 3 provides an example and legend of the EPC. Informal semantics according to the ARIS specification are the following:

- A process starts and ends with an event or process interface.
- Functions and Events alternate each other directly or indirectly via connectors.
- Functions and Events, excluding start and end events, have one incoming arc and outgoing arc.
- AND-split activates all subsequent branches concurrently, AND-join waits for all incoming branches to complete.
- XOR-split activates one branch of the possible choices, OR-join synchronizes all active incoming branches.
- OR-split allows the choice for activation of one or more branches, XOR-join merges alternative branches.
- Events cannot be followed by OR and XOR splits, as no decision is made during an event.

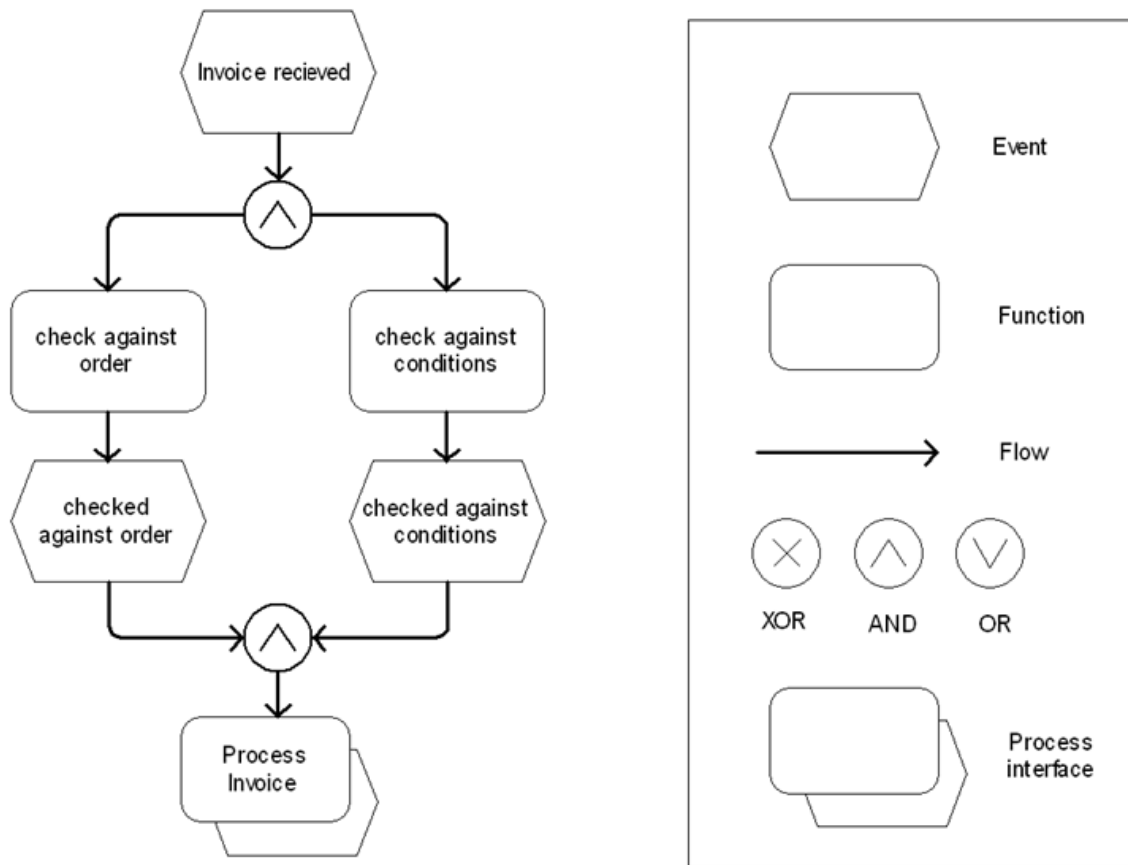


Figure 3 EPC example of invoice process and EPC constructs

Several different attempts have been made to formalize EPC as it was described in an informal way. Formalization of EPC reduces the ambiguity, and enables model consistency and completeness checks. This is required as these models are the basis for execution of processes. Furthermore, formal semantics enable the exchange of models between tools of different vendors [W. van der Aalst 1999]. [J. Mendling 2009] has collected the formalizations of EPC over time and in conclusion presented syntax and semantic formalizations. Syntax formalization makes a distinction between flat and hierarchical EPCs. A flat EPC consists of the basic EPC elements as described. A hierarchical EPC has sub processes, which means that a certain function in the EPC is decomposed in other activities. The hierarchy EPC depicts different levels of granularity of complex processes. Semantic formalization regards the state changes within EPC and is realized via mapping to Petri nets. Semantic rules formalize the transition relations between states and the reachability of the states. States are tokens assigned to arcs, and depict whether a certain node can be enabled or not, depending on the tokens on the incoming arcs. If a node can be enabled it fires. Sequences of fired nodes depict the reachability. A discussion point is semantics for the OR-join [W. van der Aalst 1999]. Semantic check for EPC is supported by tools like ProM and ARIS.

3.2 Business Process Modeling Notation

BPMN is developed by BPMI and proposed as standardized notation for business process modelling. The purpose of BPMN is that it is understandable by all business users, ranging from business analysts to technical developers. BPMN is developed to bridge the gap between business design and execution. BPMN 1.X has been released in 2004, and currently the release of BPMN 2.0 is waiting for approval. Major technical changes in BPMN 2.0 are the definition

of a formal meta-model by class diagrams, and an interchange format for semantic model and diagram interchange (DI) in XMI [OMG 2009]. These changes increase the interoperability between BPM tools. Furthermore, several notational changes are made, including amongst others the introduction of two new diagrams - choreography and conversation.

BPMN has defined a set of main elements. These include activities, events, gateways, sequence flows, message flows, associations, pools, lanes, data objects, groups and data associations. Activities are associated with work performed within a business process. Activities types are tasks, sub processes, and call activities. Most common are tasks, atomic activities. Event is something that happens, which has a cause or impact on a flow. Start, intermediate, and end events are distinguished. Gateways depict the converging or diverging sequence flows. Sequence flows depict the order of flow elements, while message flows depict the flow of messages between participants. Data objects depict the data. Pools and lanes represent participants, people and systems. Associations, groups and text annotations are artefacts. Associations depict the relation of artefacts to the flow elements. Group is to group a set of flow elements informally. Text annotations are used to depict additional information. Figure 4 is an example process depicting an order process, and the BPMN constructs.

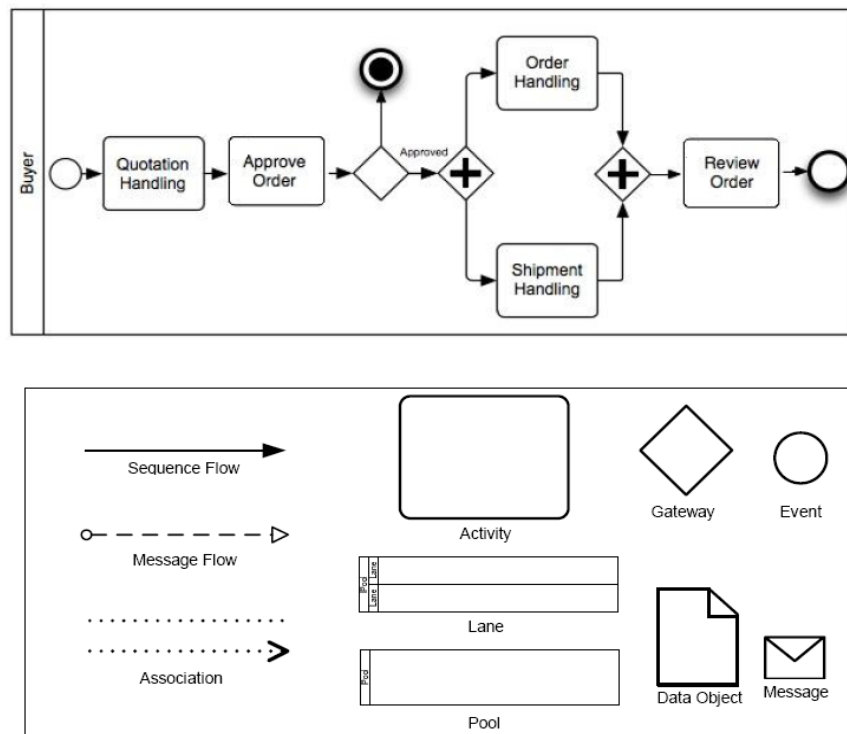


Figure 4 BPMN example and constructs legend

Several attempts have formalized semantics of BPMN. BPMN mapping against Communicating Sequential Processes have shown how to verify consistency and compatibility between business processes specified in BPMN [Wong and Gibbons 2008]. The semantic model can also be used to formally analyze and compare BPMN models. Furthermore, BPMN 1.x has been mapped to Petri nets to provide a semantic validation [Dijkman, Dumas and Ouyang 2007]. In their conclusions they identified that BPMN 1.x has a very detailed syntactic description, but lack in semantic descriptions. The validation of BPMN against Petri nets has been implemented in the ProM tool. The execution semantics are specified in BPMN 2.0. The mapping to Petri net has two limitations, exception handling for sub processes that execute concurrently multiple times and OR-join gateways. These are not in scope of Petri Net but addressed in the extension of YAWL. A challenge remains as YAWL is computationally more complex. In [Ye, et al. 2008] an attempt is made to verify BPMN to YAWL with focus on activities and message flows. But the limitations as indicated by [Dijkman, Dumas and Ouyang 2008] are not addressed.

BPMN is evaluated to the semiotic quality framework as language for business modelling [Wahl and Sindre 2005]. The semiotic quality evaluation framework creates an understanding and evaluates the quality of conceptual models. This framework has been extended to support the evaluation of modelling languages. The framework distinguishes goals and means. It is based on linguistic and semiotic concepts allowing evaluation of quality on different levels. The results have shown that BPMN is a functional oriented language for modelling processes suited for the business domain. It has familiar and easy graphical notations, but BPMN becomes more complex when the advanced features are required. The categorization of graphical elements and aggregation of activities in the meta-model increases the comprehensiveness. Considering extending the knowledge to non specialists it was concluded that “business users” include a wide variety of actors and it is hard to evaluate in a general way. Modelling in the business domain and vertical domains are supported. Regarding technical actor appropriateness BPMN alignment with BPEL is supported.

3.3 Views on Business Process Modelling

Views on process modelling categorize process models in a specific abstraction level. A process model represents a specific view on the organization. A distinction can be made between intention and perception. A stakeholder intention is captured in a model from his point of view and its design is to be interpreted by others. A clear description of each perspective will eliminate a gap between intention and perception.

Three perspectives can be distinguished in modelling considering BPD SOA:

- **Business:** The business perspective is concerned with identifying strategic objectives and areas of concerns, and relating these to the high level business processes. The organization is perceived without any operational or implementation detail. Management is the high-level business view on an organization.
- **Functional:** The functional perspective is the operational point of view on processes. The operation of the organization is represented by the workflow within a specific department. The flow of activities is identified with the supporting resources, information, and organization. From this perspective the IT requirements can be derived.
- **IT:** The IT perspective is concerned with processes containing IT implementation details, and is adjusted to the IT infrastructure.

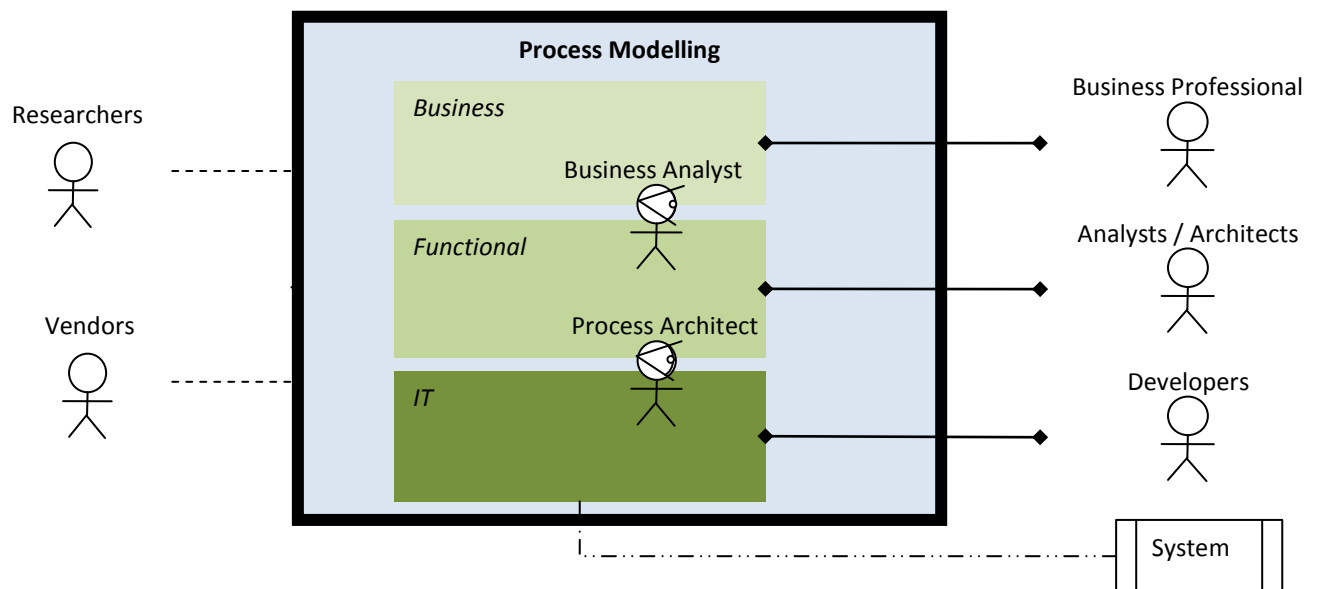


Figure 5 Stakeholders in process modelling

Figure 5 depicts the views and stakeholders in process modelling. The system demarcation is process modelling. The business analyst and process architect are the stakeholders within the system as they capture the views of the stakeholders in models. The business professionals' view is captured in a business view. The business and IT analysts and architects contribute to the functional view, capturing the operational details in the processes. The perspective of the IT developer is represented by the IT view. A system will execute the models defined in IT perspective. The external stakeholders, researchers and vendors, have influence on process modelling, as they develop new techniques and/or tools (mentioned in section 2.2).

3.3.1 Relating to Model Driven Architecture

The views determined have similarities with the viewpoints in Model-Driven Architecture (MDA) [OMG 2003]. In MDA a distinction is made between Computation Independent Models (CIM), Platform Independent Models (PIM) and Platform Specific Models (PSM). CIM models depict the context and the purpose of a model without adding any computational complexities. PIM models describe the behaviour and structure of an application regardless of the implementation platform. PSM models are ready for execution on a specific platform. MDA prescribes a model driven development approach for information systems, and is initially designed for Unified Modelling Language (UML). The MDA principles have been recognized to be applicable in the BPD SOA domain [Perez, Ruiz and Piattini 2008]. MDA development has been promoted in ARIS, however, adoption in the industry is unknown. The views on process modelling are from a stakeholder point of view, while the MDA views are from a system point of view. The views correspond on characteristics of models. The CIM corresponds with Business, PIM with Functional, and PSM with IT. Figure 6 provides a complete overview on with the views, roles, languages and phases of process modelling. Process modelling is an activity during the first five phases of BPD SOA.

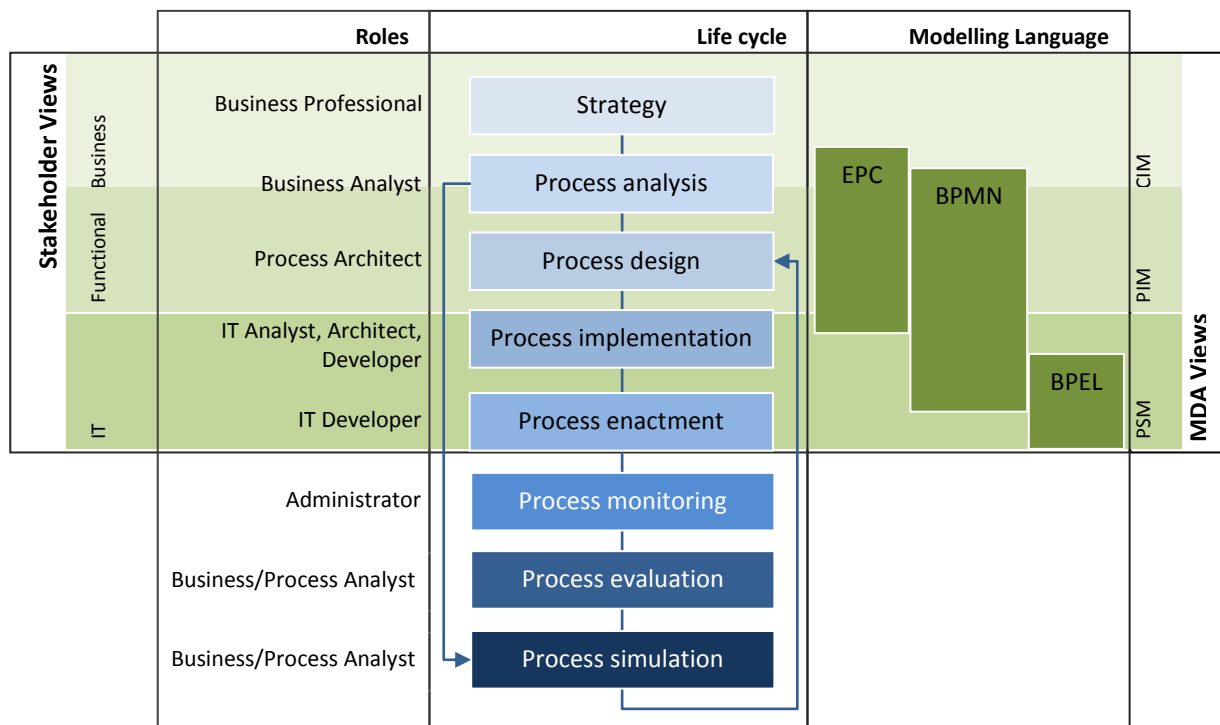


Figure 6 Process Modelling with views, roles, phases and modelling languages

Part II.

Conceptual Framework for BPD SOA round-trip modelling

Chapter 4

Conceptual Framework for BPD SOA round-trip modelling across two tools

A conceptual framework outlines possible courses of action or presents a preferred approach to an idea. The conceptual framework for iterative round-trip Business Process Driven SOA development *is developed as a structural approach to guide business to IT transformation from modelling perspective between a BPA and BPMS tool*. Focus is to depict all relevant dimensions in the development of business processes from high-level business to execution with two tools. The problem is also perceived as lack of alignment between the workflow centric processes and the models created by enterprise ontology. There exists a functional bottleneck between business perspective on operations and the actual execution of operations, where the business process space inside the organization is not coherent on semantic level [Hepp and Roman 2007].

How to achieve interoperability between a Business Process Analysis and Business Process Management Suite tool from modelling perspective and bridge the business to execution modelling gap?

The “five w’s and one h” concept is used to deliver a structured and complete approach. This has been seen in Zachman framework [Zachman 2008]. This framework will address why, who, what, when, where and how in the context of coupling a BPA and BPMS tool for Business Process Driven SOA development. It is assumed that preliminaries regarding process governance and modelling transformations are defined before using this framework (4.1). The *why*, *who* and *where* are related to the context of the problem (4.2). The *why* addresses the motives for coupling two tools. The *who* is related to the roles within BPD SOA, and the perspectives on which models can be considered. *Where* refers to the BPA and BPMS tools. *What*, *when*, and *how* are addressed in dimensions within the framework (4.3). *What* regards the set of information which should be interchanged. *When* is related to the intention of coupling, the scope of the project in which both the BPA and BPMS tools can be used (elaborated in 4.4). *How* is the degree of interoperability between BPA and BPMS tools. There are different levels of interoperability. Furthermore, traceability is a concept which is used for maintaining interoperability (4.5). A summary and overview of the framework is provided in 4.6.

4.1 Preliminary: process governance and qualitative modelling

There are preliminary requirements for using this framework. It is assumed that procedures and controls regarding process governance are determined within the organization, and qualitative and correct modelling is an objective. Quality contributes to the understanding of process models and the execution of processes. Therefore the metrics are described and guidelines for models are presented.

4.1.1 Process governance

Process governance is to govern the defined processes regarding information by defining procedures and roles. The information is usually stored in a central repository. Clear agreement should be made on organizational level on how to control the processes. When a process is transformed to another platform and changes are made, it is necessary to control these changes. It may happen that at one time in both tools the process is edited, which would require merging and version control of processes. Defining governance guidelines will help organizing and maintain these processes. There are many frameworks developed for process governance, such as ITIL or ASL. The following is important in Process Governance [Jeston and Nelis 2008] [Catts and St. Clair 2009]:

- Enabling assessment and management of risks, complaint with regulatory requirements.
- Implement business strategy, while achieve the desired value.

- Support performance measurements to monitor organizational performance, process vitality, strategic alignment, and value realization.
- Defining rights and roles for accountability for decision-making.
- Establish various guidelines and frameworks which will ensure consistent, repeatable and sustainable delivery of successful projects.

Process governance is a preliminary requirement. Working with two tools it is important to define guidelines regarding communication, approval and permissions for all stakeholders in process modelling. Depending on the adoption scenarios, as described in 2.3, different guidelines can be proposed. Guideline example is “only the process owner or its delegate can add information to a process or change its control flow”.

4.1.2 Quality metrics and modelling guidelines

Quality metrics contribute to the management of the quality of a process model [Vanderfeesten, et al. 2007]. A high quality model is supposed to be more efficient, manageable, maintainable, easy to understand and less error-prone. The quality metrics are derived from the quality metrics used in the software engineering field [Laue and Gruhn 2007] [Gruhn and Laue 2007] [Aguilar, et al. 2006] . The business process models have many similarities with software processes. Both provide a structure of a concept achieving a specific goal. Business process models have a composition of activities, and software process models compose modules or functions. Furthermore, the purpose of modelling have similarities as both fields consider management and control of the processes, create understanding and smoothen communication, process improvement, and automation of processes as major objectives. The quality metrics are implemented in the field of process mining. Process mining is analyzing and composing business processes on event logs. The quality metrics for business process models are:

- Coupling: the interconnections between modules of a model
- Cohesion: the coherence within parts of the model
- Complexity: the complexity and comprehensiveness of design
- Modularity: the number of modules of a design
- Size: the size of a model

There is a clear correlation between execution errors and quality metrics [Mendling, Neumann and van der Aalst 2007]. Evaluating processes on the quality metrics can reduce errors and expedite proper execution.

The quality metrics evaluate process models on graphical notation and structure; however, they do not address layout of graphical model and comprehensiveness of texts in the model. These are very important in understanding models. Research on the annotations on the labels of activity constructs concluded that verb-object style of labelling is preferred and most understandable [Mendling, Reijers and Recker 2009]. Other influential factors on understanding business processes are the knowledge of the user/designer, separability and text length of activity labels [Mendling and Strembeck 2008]. From organizational point of view, culture and language interpretation are very influential on understanding business processes [Whitman and Panetto 2006].

The research on quality metrics and empirical research on creating understandable process models for business users has resulted in process modelling guidelines named 7PMG [Mendling, Reijers and van der Aalst 2009]. These guidelines can be considered as general modelling rules, language independent. These guidelines help avoid errors and increase understanding of business process models. Guidelines in modelling for BPD SOA should guide the modelling process. 7PMG [Mendling, Reijers and van der Aalst 2009]:

- Limit the elements used in the model. Size of a model affects understandability and likelihood of errors.
- Minimize routing paths. The degree of an element (number of input and output arcs) is correlated with number of modelling errors.
- Use one start and one end event. Multiple start and end events confuse and increase error probability.

- Model as structured as possible. Balanced use of split and joins are recommended, to reduce errors and to increase understandability.
- Avoid OR-routing elements. AND and XOR connectors have well defined semantics, unlike OR connectors which cause implementation problems.
- Use verb-object activity labels. This is tested as most understandable.
- Decompose models with more than 50 elements, due to the positive correlation of size and errors.

4.2 Context analysis

Context analysis of the framework regards demarcation of the system of BPD SOA. A clear view on the organization regarding the context helps identifying in what degree the dimensions should be addressed. BPD SOA has a life cycle with several phases. For each phase the activities, roles and models should be clear. The next step is to identify what can be realized within which tool, that contributes in identifying the possible coupling points for model transformations.

BPA and BPMS tools (where)

Identification of the two tools which represent the BPA and BPMS platform for the organization, and what their purposes are. Analyzing the role of the BPA and BPMS tool within an organization and depicting their overlap and differences provides insight into what degree coupling might be relevant.

Motive (why)

The motive for coupling should be evident. A consensus on what is expected of the coupling is important. This step concerns the formulation of requirements for the coupling, both on organizational as well as on technical level.

Viewpoints and roles (who)

Related to the views in business process modelling, three clear distinctions can be made. Business, Functional and IT perspective which can be related to the MDA abstraction layers (this is highlighted in the views on process modelling in 3.3.1). Business process models can be classified to a perspective. Each perspective is coupled to a certain roles, which depict the responsibilities which belong to a specific perspective. The roles depict the intention of the perspective. Furthermore, the phases depict the activities, and what type of information can be specified at one perspective. This is summarized in Table 2.

Perspective in modelling	Business Process Driven SOA Phases	Roles
Business	Strategy	Business Professional
Functional	Process analysis	Business Analyst
	Process design	Process Architect
	Process implementation	IT Analyst IT Architect
IT		IT Developer
	Process enactment	IT Developer
	Process monitoring	Administrator
Functional	Process evaluation	Business Analyst Process Analyst
	Process simulation	Business Analyst Process Analyst

Table 2 Perspectives, phases and roles in BPD SOA

4.3 Dimensions

Dimensions capture all relevant aspects when coupling the BPA and BPMS tool. Each dimension depicts several levels. In attempt for coupling for each dimension it needs to be determined which levels are relevant. For this framework three dimensions are depicted: information, scenarios and interoperability.

Information for interchange (what)

Information is captured models and one model can depict different type of information. For each viewpoint the set of relevant information needs to be identified. Mapping the type of constructs against the perspective provides a clear overview of the information for interchange.

Scenarios for coupling (when)

Another dimension is related to the different intentions of business process driven SOA modelling. This dimension identifies the possible scenarios of exchange between the platforms. Within this dimension different abstraction layers are identified, which deal with the perspectives from which a model can be considered. Thus, is related to viewpoints in modelling, depicting which concepts and constraints are relevant.

In order to get from a high business model to an execution model requires different type of model transformations are required. These are named the *scenarios*. Model transformations can be looked upon from different angles allowing different transitions. Horizontal transformations include model integration, translation or synchronization, bidirectional or unidirectional. Meta-level mapping and structural patterns can take care of horizontal transformation. Vertical transformation is either a refinement, or abstraction step. This is elaborated in section 4.4.

Level of interoperability (how)

The IEEE defines interoperability as “the ability of two or more systems or components to exchange information and to use the information that has been exchanged”. BPA and BPMS tools can be considered as the systems. Interoperability can be achieved with integration or compatibility, but is not interchangeable with these terms. The Interoperability dimension is to identify what degree of exchange can be expected when coupling a BPA and BPMS tool. There are many interoperability standards for enterprises and modelling [Guédria, Naudet and Chen 2009], including amongst others IDBAC Enterprise Interoperability Framework (EIF), and Levels of Conceptual Interoperability Model (LCIM). The levels of interoperability within the scope of this framework are relevant in the process layer of the enterprise interoperability framework, which defines all domains in enterprise interoperability [Chen, Doumeingts and Vernadat 2008].

Interoperability for BPD SOA context can be measured by the layers of LCIM [Turnitsa 2005]. The LCIM is defined for modelling and simulation. Modelling and simulations is used for research in decision making in complex and dynamic environments. Modelling is an abstraction of reality captured in a process, and simulation is its execution on a computer. The types of interoperability:

1. Technical interoperability is that the source and target systems have the technologies to support exchange, i.e. message exchange.
2. Syntactic interoperability is that two systems are capable of exchanging data. Exchange formats are compatible for both source and target.
3. Semantic interoperability is the ability of two systems to comprehend the data that is being exchanged.
4. Pragmatic interoperability is that the source and target system have the same intention, expectation and understanding of the exchange.

5. Dynamic interoperability is achieved when the source and target systems can comprehend state changes.
6. Conceptual interoperability is when the conceptual model is formal enough to be implemented by systems.

The layers defined in LCIM can in a sense be applicable to the BPD SOA context. The first four levels of interoperability are relevant. However, level 5 and 6 cannot be realized, as they are also challenging in the modelling and simulation domain [Tolk, Diallo and Turnitsa 2007] [Wang, Tolk and Wang 2009]. Interoperability regarding process models can be achieved by 1) a standard for representing process models 2) unification based on a common meta-level proofing semantic equivalence of the process models 3) ontological federation based on semantic equivalence [Cimander and Kubicek 2009].

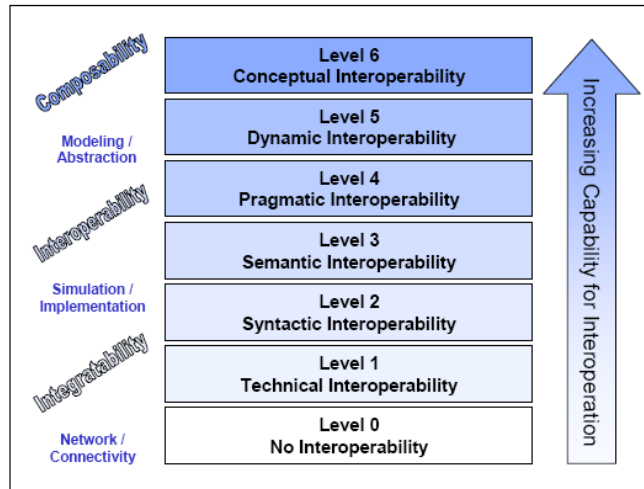


Figure 7 Levels of Conceptual Interoperability Model (LCIM) [Turnitsa 2005]

From enterprise point of view, another type is defined relevant for process modelling: organizational interoperability [Cimander and Kubicek 2009]. Organizational interoperability is concerned with integration of business processes and the exchange of inter- and intra-organizational information. Aim is to achieve coherence on organizational level between the parties which wish to exchange information.

From the people perspective, culture has impact on the method of working of the business [Whitman and Panetto 2006]. Cultures have different design philosophies and objectives, which influence how they operate. Language has influence on the semantic interpretation. Cultural interoperability is to achieve interoperability between the different cultures in the globalized business environment. On smaller scale, it may also refer to company culture.

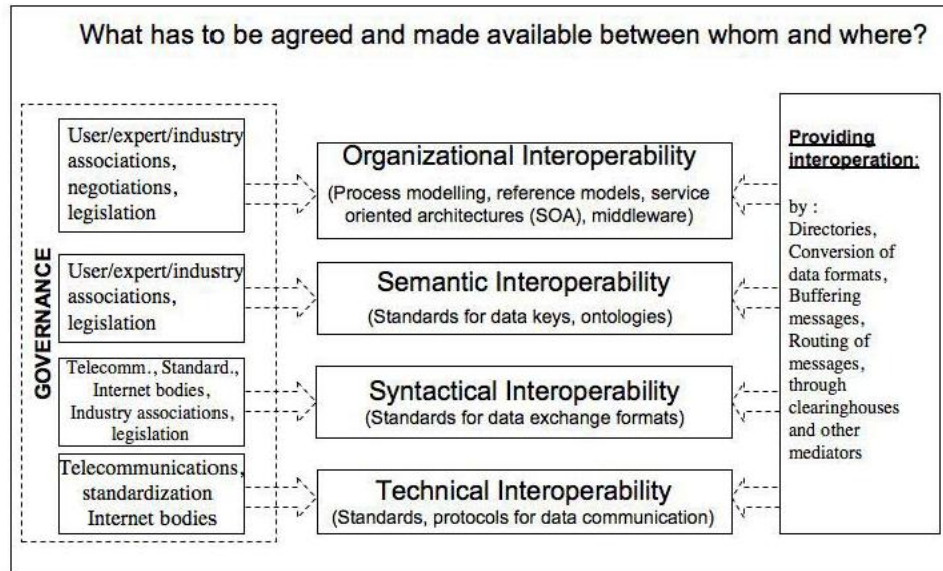


Figure 8 Interoperability [Cimander and Kubicek 2009]

Within the scope of this framework the interoperability levels can be used to evaluate to what extent the coupling is interoperable. Figure 8 provides a good illustration of the type of interoperability and the governance required to achieve such interoperability. In a sense pragmatic interoperability of systems is included in organizational interoperability in this figure. LCIM describes interoperability levels for process models, whereas organizational and cultural interoperability are more subject to enterprise governance.

4.4 Model transformation

Model transformation concerns interoperability of modelling languages. Two models languages are interoperable when for every source model there is a transformation into the target model. There are different levels and forms in transformations. Challenges include preserving the semantics, avoiding loss of information, provide user interaction and semantically enrichment [Kappel, et al. 2006].

4.4.1 MOF Meta-model structure

Figure 9 presents the Meta-Object Facility (MOF) structure [Dreiling, et al. 2008] [Perez, Ruiz and Piattini 2008]. The MOF is a convention for understanding relationships between different kinds of data and metadata. Every model embodies a real-world concept on instance level (M0 level), and can be expressed in a specific modelling language (on M1 level). The modelling language has a meta-model (on M2 level), which is an abstraction of the M1 and can be expressed in a certain modelling language. The meta-meta-model (M3 level) is the representation ontology of the meta-model, the “abstract language” defining different kinds of meta-data. The MOF defines the MOF model, using the Object Constraint Language (OCL), as meta-meta-model.

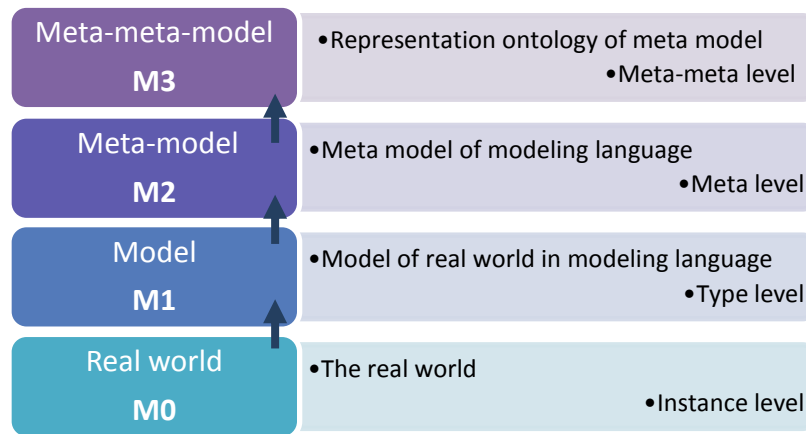


Figure 9 MOF meta-model structure

The structure gives insight on what different abstraction levels a model can be considered. The MOF structure is the basis for transformations in MDA. There are different methods for model transformations in MDA. In literature the model-to-model transformation languages of MDE have been evaluated [Jouault and Kurtev 2007], as well as model transformation approaches [Czarnecki and Helsen 2003] [Prakash, Srivastava and Sabharwal 2006]. A known model transformation language is the Query/View/Transformation (QVT) developed by OMG. QVT is a standard defining meta-models for defining model transformation languages. A transformation language can transform source to target languages. There are many different approaches for transformation including transformation rules, cardinality, directionality and traceability. *Cardinality* specifies how many instances of an entity relate to one instance of another entity. *Directionality* (Figure 10) regards whether interoperability is uni- or bidirectional between source and target languages. In round-trip development directionality is addressed. Unidirectional interoperability is when the mapping can be achieved in one direction. Bidirectional interoperability concerns mapping in both directions. *Traceability* (section 4.5) is to trace the transformations, link source and target elements. These concepts are applicable for modelling transformation in BPD SOA.



Figure 10 Directionality in transformation (uni and bidirectional)

Model transformations have three transformation scenarios (Figure 11): integration, translation and synchronization [Murzek (2) and Kramler 2007]. Model integration is the integration of two models in different languages into a model of one language. Model translation is the mapping of a model in one language into a model of another language. The model synchronization regards keeping pairs of models after changes up to date.

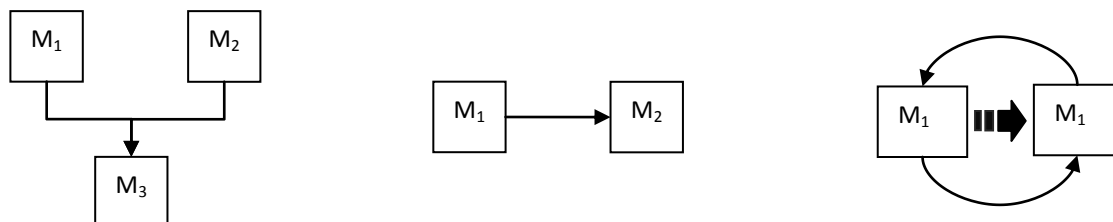


Figure 11 Scenarios in transformation (Integration, Translation and Synchronization)

4.4.2 Transformation classifications: horizontal and vertical

In BPD SOA there are multiple views on a model. Transformations can be considered as transforming models from across and within a view: vertical and horizontal transformation. In other words, vertical transformation is concerned with the transformation of models between different levels of abstraction. Horizontal transformation is the transformation of models on the same level abstraction. Usually it concerns the transformation of a model in language L_1 into a model of language L_2 .

When exchanging a model from one tool to another tool, these transformation classifications are useful to identify which transformations are present. The distinctions decompose the transformation into manageable steps. When a model is designed in a BPA tool and should be made executable in a BPMS tool. Either way, there is always a vertical and horizontal transformation involved.

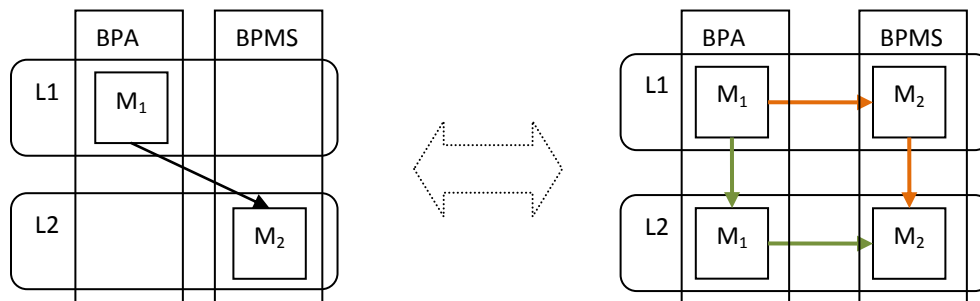


Figure 12 Decomposed transformation between two tools

For the transformations it is necessary to determine what set of information is relevant for mapping (as proposed in the dimension ‘information for interchange’). A subset of one modelling language is determined for each abstraction layer. On one hand vendors may support a limited amount of constructs. On the other hand one can restrict the number of constructs to what is actually commonly used. Implementing transformations for a complete syntax and semantics of modelling languages is redundant as not all is used. For BPMN, research has been conducted on which subset of constructs is most used [Muehlen and Recker 2008]. They have shown that a clear distinction in complexity of BPMN can be made of what in practice is used, compared to what the theory defines. Furthermore, the construct correlations are also interesting for identifying patterns.

4.4.3 Transformations overview

Combining the transformation approaches and classifications the complete model transformation overview for BPD SOA with a BPA and BPMS tool is created in Figure 13. As shown many scenarios are possible in general. Three abstraction layers correspond to the stakeholders’ views on modelling: business, functional and IT. In general, transformations can be within, horizontal, or across, vertical, these abstraction layers. Horizontal transformations are either uni- or bi-directional, and either concern translation or integration or synchronization. Translation and integration should be executed in one tool, while synchronization is realized by exchange formats. Vertical transformations are refinements, from business to executable, or abstractions, from executable to business, of process models. The source and target language are the same in vertical transformations.

The transformations are subject to technical, syntactic, semantic and pragmatic interoperability. Two Guidelines are defined in order to support maintainability and control of the transformations [Stein, Kühne and Ivanov 2009]:

- Level of detail on each abstraction layer should be compliant to the purpose of the layer
- Restrict expressiveness of source modelling language to subset

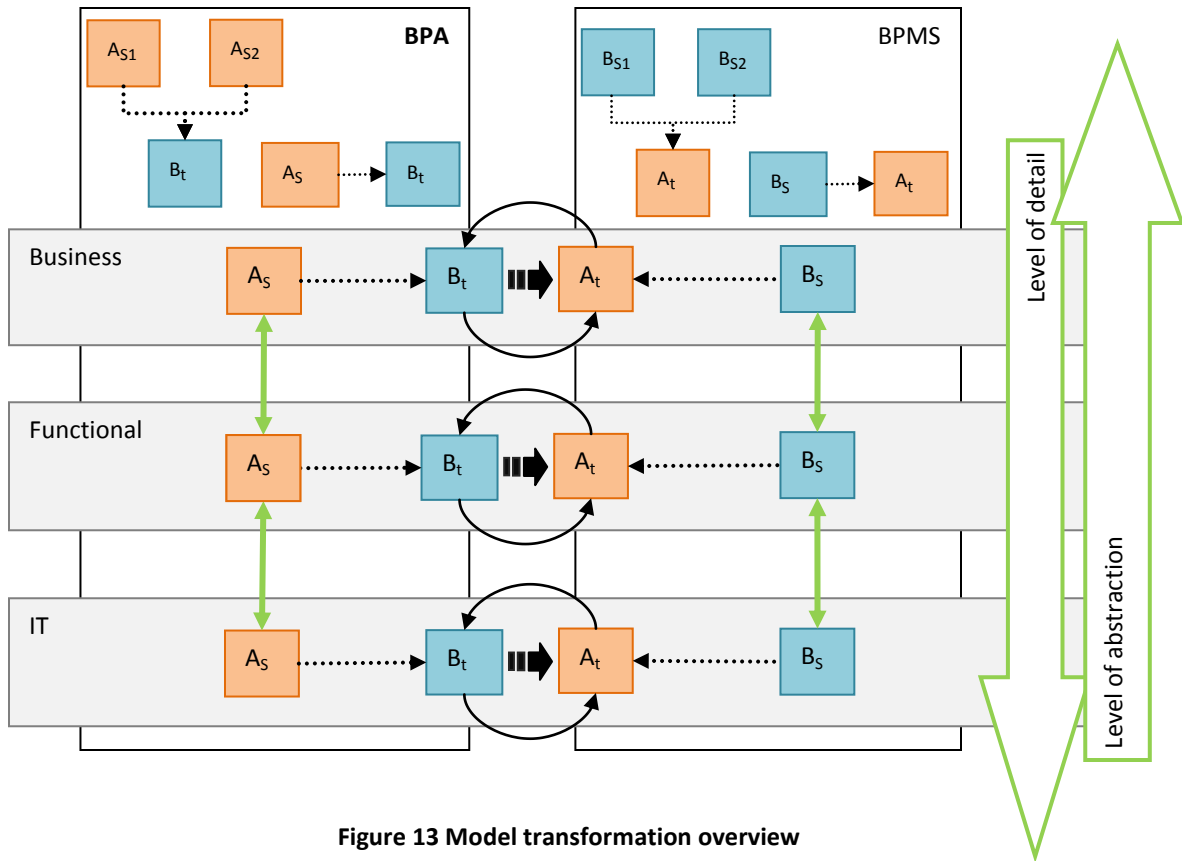


Figure 13 Model transformation overview

The rectangles in Figure 13 depict process models. Orange models have BPA conventions (A_s = A source model, A_t = A target model), and blue have BPMS conventions (B_s = B source model, B_t = B target model). The model scenarios are as in Figure 11: integration, translation and synchronization. The green arrows depict vertical transformation. The dotted lines depict that the outcome of horizontal transformation is compliant for synchronization the other tool. For example, on business level in the BPA tool the integration of models of A_{s1} and A_{s2} results in B_t , which is subsequently synchronized with the BPMS. In the BPMS this model B_s can be subject to vertical transformation or further design. When the changes need to be synchronized with BPMS, B_s should be compliant for synchronization with BPA tool.

4.5 Maintaining interoperability: traceability

As mentioned before, model transformations for BPM development are inspired from MDA concepts. The question of how to ensure interoperability between models when they are transformed between different platforms is in different domains addressed by traceability. Traceability is to trace the path of information. Enabling traceability allows making changes, decisions and easy maintenance. Traceability is used in various domains. In MDA traceability is used to trace the model elements from design to implementation [Bondé, Boulet and Dekeyser 2006]. In Requirements Engineering traceability is used to analyze the life of requirements, determine the impact of a requirement and the elements that depend on it throughout its life cycle [Champeau and Rochefort 2003]. In software architecture modelling traceability of concerns in architectural views, deals with traceability within and across different views. Concern Traceability Meta-model (CTM) is developed to enable this [Tekinerdogan, Hofmann and Aksit 2007]. CTM can be used for amongst others impact analysis of evolution of concerns, by depicting the dependency links among the architectural concerns in the architectural views.

The concept of traceability, as in MDA, can be used to keep track of the transformation changes and synchronizations of process models in BPD SOA with two platforms. It can also be applied for requirements management throughout

the BPD SOA life cycle, but keeping track of the changes has higher priority. Traceability management in software development recognizes the following steps [Ramesh and Jarke 2001] [Boronat, Carsí and Ramos 2005]:

1. Trace definition: Define the type of information to be traced.
2. Trace production: Indicate which elements generate traces for further use.
3. Trace extraction: Indicate how the traces from production can be queried to achieve certain goals.
4. Trace verification: Verify the traces

In BPD SOA *design* context the first two steps are relevant. Trace extraction is to analyze the traces, which might be relevant in the future. Trace verification regards the consistency of the traceability models. This is a challenge for testing the implementation of traceability functionality within tools.

The trace information needs to be specified, including what to trace, and where to trace. In addition the traceability model, defined by the trace storage method, needs to be determined. Figure 14 provides an overview of the traceability concepts: the TraceModel and information to trace. In the subsequent sections these concepts are presented.

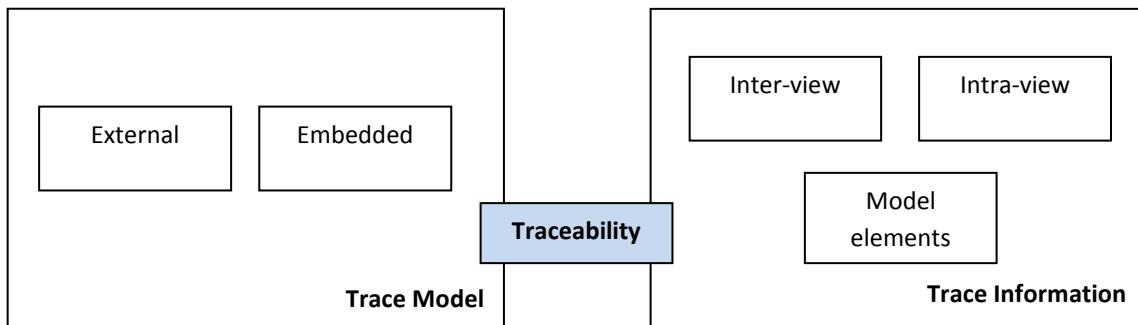


Figure 14 Traceability concepts

4.5.1 Traceable information

It is necessary to 1) define the information to be traced 2) define the information of a transformation which should be kept, in other words trace definition and production. A trace is a record of a change of and between elements. Traceability can have different focuses depending on its purpose.

It is possible to trace the *model elements* subject to transformation [Champeau and Rochefort 2003]. Model element based traceability has to depict the linkage between model elements which have transformed. There are different types of transformations, as defined in horizontal and vertical transformation. Except for showing the linkage between model elements, it is also necessary to determine which *type of linkage* is present. Model elements can be translated, or integrated on horizontal level, or refined, refactored, or merged on vertical level. For traceability the type of transformation in MDA are recognized as generic operators which depict traceability navigation [Boronat, Carsí and Ramos 2005].

Tracing model elements shows the evolution of models. Traceability can be configured to display different kind of traces. Depending on the type of traces the trace information varies.

4.5.2 Storing traces: Trace Model

There are different ways for *storing the traces*. The traceability model can either be within one model or the traces form a separate model. Embedded traceability is embedding the traces in the changed models by means of annotations or associations. With this method issues regarding pollution and uniformity may rise. If there are many transformations which are traced, the high numbers of traces pollute the model. Uniformity should be maintained, as in one model one should be able to distinguish the traces from original model elements. External traceability is to create a separate model storing all traces. The model requires unique identifiers to maintain the links to the model,

and meta-models need to be defined for each type of traceability model. A generic meta-model is developed in MDE field, Unified Traceability Schemes [Limón and Garbajosa 2005], which have extensibility features to support any type of traceability. Compared to the embedded traces, external traceability models do not pollute the original models. However by using identifiers a trade-off exists between the manageability for automation and user-friendliness. Alternative is to combine both methods for storing, by merging elements and matching specific annotations to display transformations [Kolovos, Paige and Polack 2006].

4.6 Overview of the framework

The overview of the framework is displayed in Table 3. The framework proposes *a structural approach in identifying how interoperability for modelling in BPD SOA should be achieved with a BPA and BPMS tool*. It is divided in two major parts. One is specifying the context and stakeholders in modelling, and the other classifying the transformations and level of interoperability. First the context should be clarified, subsequently the dimensions.

Framework for BPD SOA modelling with BPA and BPMS			
	Layers	Description	Deliverables
Context Analysis (section 4.2)	Where- BPA and BPMS tool	Analyze the BPA and BPMS tool and their differences and similarities in modelling activities	<ul style="list-style-type: none"> – Clear overlap and differences in tools – Possible exchange points of models
	Why- Motivation	Clarify what is expected of coupling a BPA and BPMS tool regarding modelling	<ul style="list-style-type: none"> – Requirements for the coupling
	Who- Viewpoints	The viewpoints in modelling should be determined and associated roles identified for recognisability	Mapping of viewpoints and roles <ul style="list-style-type: none"> – Business – Functional – IT
Dimensions (section 4.3)	What- Information for interchange	Define the information which needs to be exchanged. Relate the information to each viewpoint determined	Subsets of modelling languages mapped against the viewpoints. Model languages are most commonly EPC, BPMN and BPEL
	When- Scenarios for coupling	Define which scenarios in vertical and horizontal transformations apply after having identified what information needs to be exchanged (detail in section 4.4)	Identified transformations and their scenarios: <ul style="list-style-type: none"> – Horizontal: translation, integration, synchronization – Vertical: refine or abstract
	How- Level of interoperability	Awareness of the different types of interoperability and define how is expected to achieve the levels of interoperability	Determine how to achieve interoperability for each level; <ul style="list-style-type: none"> – Technical – Syntactical – Semantic – Pragmatic – Organizational – Cultural

Table 3 Conceptual Framework for coupling BPA and BPMS tool for modelling - overview

A graphical sketch of the position of the framework is provided in Figure 15. The two transition points are illustrated. The framework addresses the first gap, model transformations in analysis and design. The models are the foundation for implementation, execution and monitoring. The second gap regards evaluation of monitoring data. Different options exist for exchanging monitoring data. Either the monitoring is done and analyzed in the BPMS, or the BPA

tool is linked to the execution engine and retrieves the data real-time. Research on the second gap is not within scope of this thesis.

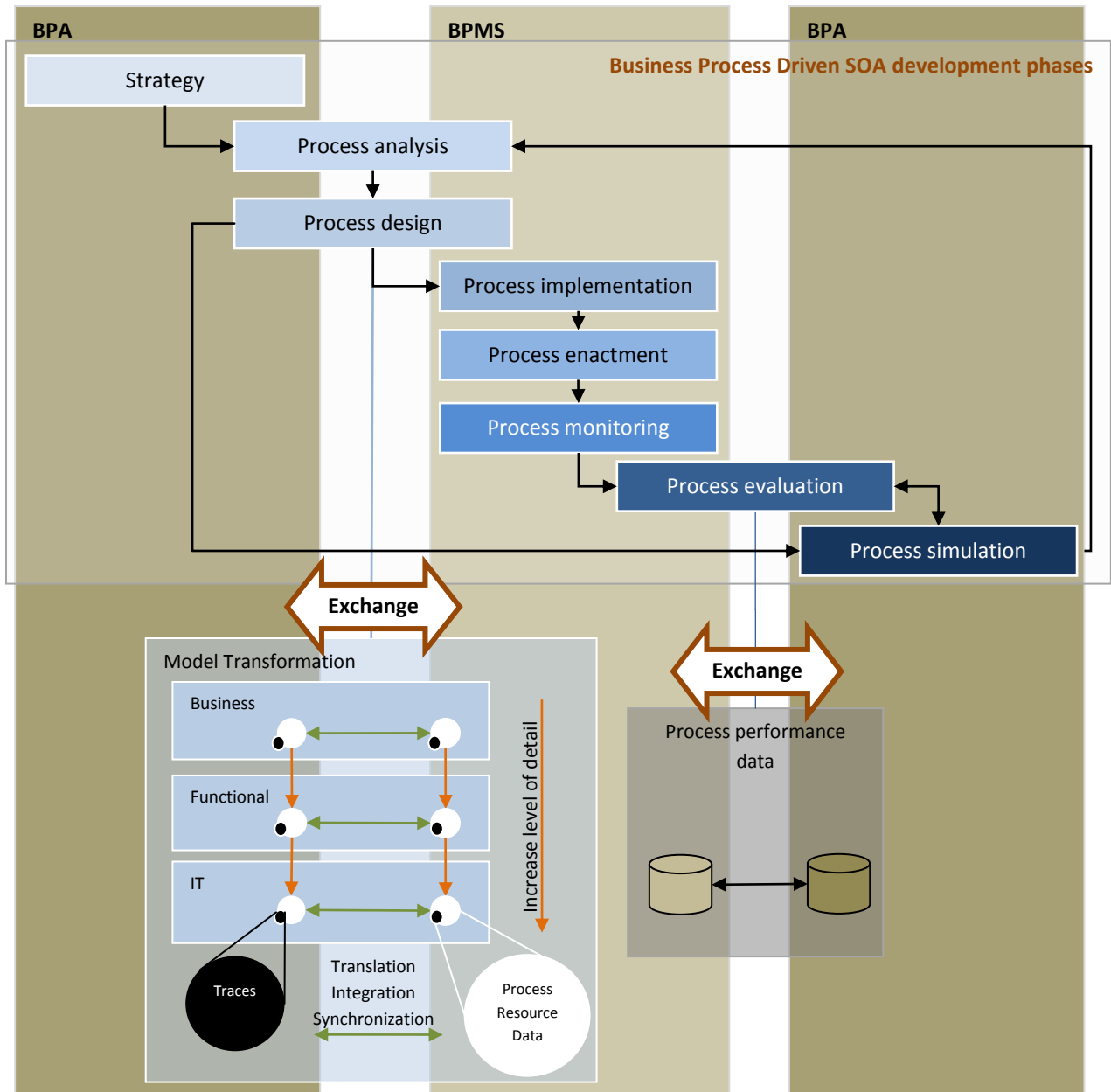


Figure 15 Position of BPA BPMS modelling coupling

Chapter 5

Approaches for model transformations

In this chapter approaches for the model transformations (as in 4.4) are described. In the past years research has been done on model transformations. For the model transformations approaches, an understanding of ontologies (5.1) and patterns (5.2) are useful as those are recurrent concepts in the transformations. Related work is shown in section 5.3. This work depicts the state of the art in business to IT transformation. Based on literature, approaches have been selected and combined for the model transformations as defined in the conceptual framework in Chapter 4. Horizontal transformation approach is presented in 5.4. Vertical transformation approach is described in section 5.5.

5.1 Ontologies and meta-models

The hierarchy of meta-models, Figure 9 in 4.4, depicts the instantiation relations of models. A model (M1) is an abstraction of an instance in real life (M0). The meta-model (M2) is an abstraction model depicting the grammar (syntax and semantics) of a model. The model hierarchy embeds syntactical structure, however, does not depict anything on the semantics of domain specific properties of an instance. The domain specific properties are captured by ontology. In computer and information science, *ontology* is a formal explicit specification of a shared conceptualization [Gruber 1995] [Borst 1997]. An explicit specification implies that concepts, interrelations and relationships within a domain are explicitly defined. Formal insinuates that the explicit specifications must be machine-readable and unambiguous. A shared conceptualization regards to a consensus on vocabulary and terminology between different views (i.e. stakeholders) within a domain.

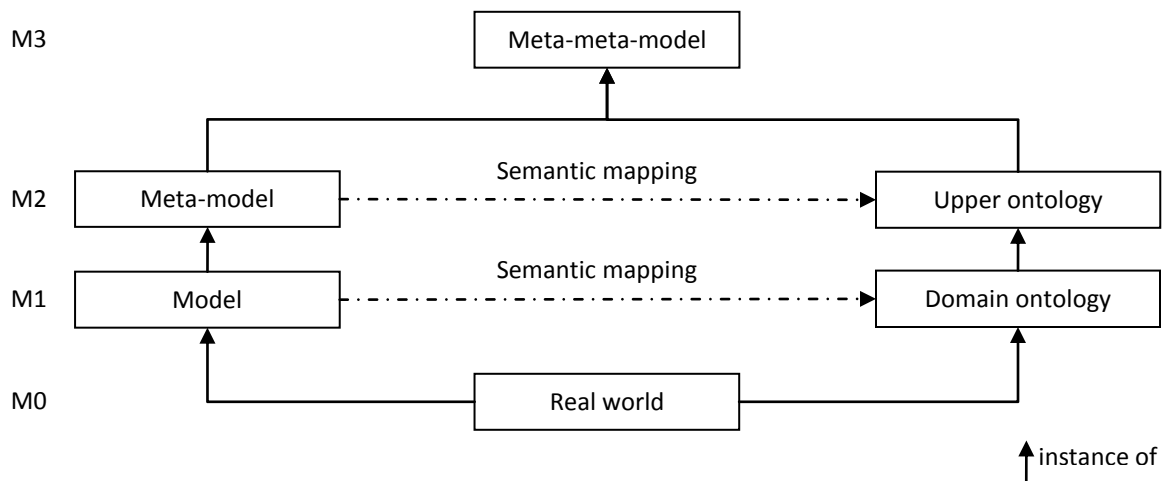


Figure 16 Metamodel and Ontology hierarchy

An ontology hierarchy, corresponding to the meta-model hierarchy, provides insight on the meta-data relations between domain specific properties [Aßmann, Zschaler and Wagner 2006]. Figure 16 depicts these hierarchies. On M0 level, a real world situation is the foundation for the instance to be modelled and the domain ontology. On M1 level, a model captures the real world instance. From ontological perspective, *domain ontology* captures the concepts and relations of a certain domain of the real world. The domain ontology can also be considered as constructed from multiple viewpoint ontologies [Borst 1997]. A *viewpoint ontology* is a formalized conceptual category of concepts within a domain. Note that the viewpoint distinction does not have an “instance of” relation, thus absent in the figure. The domain ontology is instantiated by an upper ontology. The *upper ontology* describes

concepts and relations on meta-level covering a broad range of domain areas. Although meta-models and ontologies have different purposes, a relation can be made as they both are instances of the real world. The relation in M1 and M2 level is semantic mapping of the model/meta-model to domain/upper ontology. The link is to derive ontological representations from the concepts used for modelling in models. Several approaches have been proposed to semantic map the meta-models to ontologies. In [Saeki and Kaiya 2006] the node-arc of a model is mapped to a concept-instance of relation of the ontology. In [Kappel, et al. 2006] meta-models are lifted to pseudo-ontologies, which in turn will be refactored by patterns to proper ontologies. On M3, the meta-meta-model is an abstraction language describing different types of meta-data, instantiating meta-data: meta-model and upper ontology.

Ontology is the consensus on all concepts and relations within a certain domain, with vocabulary and terminology specific to that domain. The difference between an ontology and meta-model is that an ontology specifies the semantics of both the modelling constructs and the model instances, while a meta-model only regards the modelling constructs. A valid meta-model is an ontology, but ontology might not be explicitly modelled as meta-models.

5.1.1 Domain ontologies

For the business (modelling) domain different ontologies have been developed. Domain ontologies for the business domain include amongst others enterprise ontology [Dietz and Hoogervorst 2008], E3 value [Gordijn and Akkermans 2001], TOVE Project [Grüniger, Atefi and Fox 2000], Resources Event Actor (REA) Enterprise [Geerts and McCarthy 2000], SUPER Set Ontologies [Hepp and Roman 2007] and Business Model Ontology (BMO) [Osterwalder 2004]. Table 4 provides an overview on their main categorizations for ontologies and focus. Several ontologies were developed for certain domains, such as REA for accounting, and have been lifted to enterprises in general. The focus depicts what aspect in the business domain in general is addressed. For each ontology, it is depicted if it specifies a business process ontology. Business process ontology defines the concepts of a business process and the relationships among them. The E3-value approach clearly distinguishes business modelling from process modelling [Gordijn, Akkermans and van Vliet 2000], and E3-value does not specify a business process ontology. Furthermore, BMO and RAE Enterprise ontology do not address business process ontology. From the domain ontologies viewpoint ontologies can be determined (note: domain ontologies, i.e. SUPER, refer to their viewpoint ontologies as domain ontologies). Viewpoints are either following the logical decomposition of an enterprise or a specific phase in BPD SOA. Examples are the core ontology for Business Process Analysis [Pedrinaci and Domingue 2008] and the organisational ontology framework for Semantic BPM [Filipowska, et al. 2009]. The domain ontologies can be used in order to develop a shared vocabulary and terminology within the business domain. Often they prescribe requirements, in form of competency questions, which are the characteristics of information for the ontology.

5.1.2 Upper ontologies

Upper ontologies include amongst others Suggested Upper Merged Ontology (SUMO) [Niles and Pease 2001] and Bunge-Wand-Weber (BWW) [Wand and Weber 1990]. These upper ontologies depict the semantic relations of meta-models. SUMO is a standard ontology that will promote data interoperability, information search and retrieval, automated inferencing, and natural language processing. Bunge Wand Weber is recognized as ontology for information systems, and has been lifted as ontology for conceptual modelling [Green and Rosemann 1999].

Domain Ontology	Description / categorizations	Main Focus	Business Process Ontology
Enterprise Ontology	Organisational structure, activities and management. Four subject areas are distinguished: activities and planning, organisation, strategy, marketing	Enterprise and environment	+ [Dietz and Hoogervorst 2008]
E3-value	Identifying exchange of value objects between the business actors. The basic concepts in e3-value are actors, value objects, value ports, value interfaces, value activities and value exchanges.	E-business, enterprise and environment	- [Gordijn, Akkermans and van Vliet 2000]
TOVE	Foundational ontologies for activities, and resources. Furthermore, business ontologies include organization, Product and Requirements, ISO9000 Quality and Activity-Based Costing.	Public and commercial enterprise modelling	+ [Grüniger, Atefi and Fox 2000]
RAE Enterprise	Creation of transfer of economic value. The core concepts are: Resource, Event, and Actor.	Transaction modelling	-
SUPER	Organisation context with main focus on functions, goals, organization structure, roles, resources.	Semantic BPM	+ [Hepp and Roman 2007]
BMO	BMO main pillars are Product, Customer Interface, Infrastructure Management, and Financial Aspects.	Enterprise and environment	-

Table 4 Business Domain Ontologies: description and focus (extended and edited [Filipowska, et al. 2009])

5.2 Patterns in business processes

There are different types of patterns in business processes. Best-known are workflow patterns. *Workflow patterns* are recurring and generic concepts within process aware information systems [van der Aalst, et al. 2003]. Workflow patterns are developed to provide a comprehensive description of concepts relevant to the modelling and execution of workflow systems both now and on an ongoing basis. The patterns are expressed in an imperative workflow style expression independent from a specific language. It can be used as method to evaluate the expressive power of business process modelling languages. Four perspectives exist in workflow patterns: control-flow, data, resource and exception handling. Control-flow perspective regards execution flow of activities through different constructs. The data perspective views the business and processing data related to the control-flows. The resource perspective regards the organizational structure in form of human and system roles for executing activities. The exception handling perspective describes various causes of exceptions and the various actions that need to be taken as a result of exceptions occurring. The control-flow perspective is essential for depicting the workflows effectiveness, whereas data, resource and exception handling are complementary to the control-flow. The patterns are useful for examining the business process modelling languages on their capabilities regarding expressing different business concepts.

5.2.1 Structural Design Patterns

Structural design patterns capture recurrent business control-flow concepts of business processes closer to the nature of business processes on high-level [Murzek, Kramler and Michlmayr 2006]. They are constructed from combined workflow control-flow patterns. When both design patterns exist in a certain modelling language, a mapping can be realized without loss of semantic meaning in the context. The design patterns capture the graphical variations of certain concept in a specific modelling language. These need to be mapped, as one-to-one construct mapping will omit these contextual translations. There are several basic design patterns, which are always present in end-to-end processes; the start event pattern, the end event pattern and the sequence path pattern. Combinations

of the split/ join or branch/merge paths are defined as decision patterns. Three types of decision patterns exist: parallel, alternative, and multiple alternatives. The branching pattern captures the behaviour of split and joins according to the semantics of one language. It is an advantage to tackle the split and joins as one pattern to see the relations. For example, a join in BPMN may happen implicit, and one would have to trace the split to determine the nature of the join. However, more complex situations arise when decision patterns are nested and overlap have no common end merger, or predecessor of a split. A loop pattern can usually be depicted in variations.

5.2.2 Activity Patterns

For operational business processes, empirical research has been conducted on which business functions occur most often [Thom, et al. 2008]. Recurring business functions are considered as *activity patterns*. These patterns can help during process model design, quality and comparison. The activity patterns can be considered as building blocks for business process modelling. Activity patterns can be composed of workflow (WP) and service interaction patterns (SIP) [Barros, Dumas and ter Hofstede 2005]. Recurrent business functions are amongst others notification, decision and approval. Table 5 displays the activity patterns, description and implementation patterns. Research was conducted on 241 processes of 13 different organizations in different domains, with varied organization sizes and kinds of decision making. The research evaluated the frequency of the activity patterns in business processes.

Activity Pattern	Description	Implementation pattern
Approval	An object has to be approved by one or more organisational role(s).	<ul style="list-style-type: none"> · Send/receive (SIP) · Multi instance with a priori Runtime knowledge (WP)
Question-Response	Question during process enactment which requires response from an organisational role(s).	<ul style="list-style-type: none"> · Send/receive(SIP) · Multi instance with a priori Runtime knowledge (WP)
Unidirectional Performative	Sender request execution of particular task from another process participant. Process execution continues after request.	<ul style="list-style-type: none"> · Send (SIP)
Bi-directional Performative	Sender request execution of particular task from another process participant. Process execution continues after notification of performance.	<ul style="list-style-type: none"> · Send/receive(SIP) · Multi instance with a priori Runtime knowledge(WP)
Notification	Status or result of an activity execution in communicated to one or more process participants.	<ul style="list-style-type: none"> · One way send (SIP)
Informative	An actor request information of process participant. Process execution continues after receiving the info.	<ul style="list-style-type: none"> · One way send (SIP) · One-to-Many Send/Receive (SIP)
Decision	Activity followed by an decision (multiple branches)	<ul style="list-style-type: none"> · Activity+Decision operator · Deferred Choice(WP)

Table 5 Activity Patterns in Business Processes [Thom, et al. 2008]

It showed that unidirectional and bidirectional performative, notification informative, and decision are most frequent and domain independent. The approval pattern occurs most often, but is domain specific. Furthermore, the activity patterns are evaluated on occurrence in system or human type processes, where system types regard processes with minimal or no human intervention during execution, while human types require a lot of human interaction, intuition and judgment for decision making. Results showed that unidirectional and bidirectional performative, notification, and decision patterns are present in both types of processes, while approval and

informative are only frequent in human centric processes. Research is currently done on co-occurrence of the activity patterns [Thom, et al. 2008].

The patterns in business processes are summarized in Figure 17. Dotted lines are 'part of'. Arrows is 'basis for'.

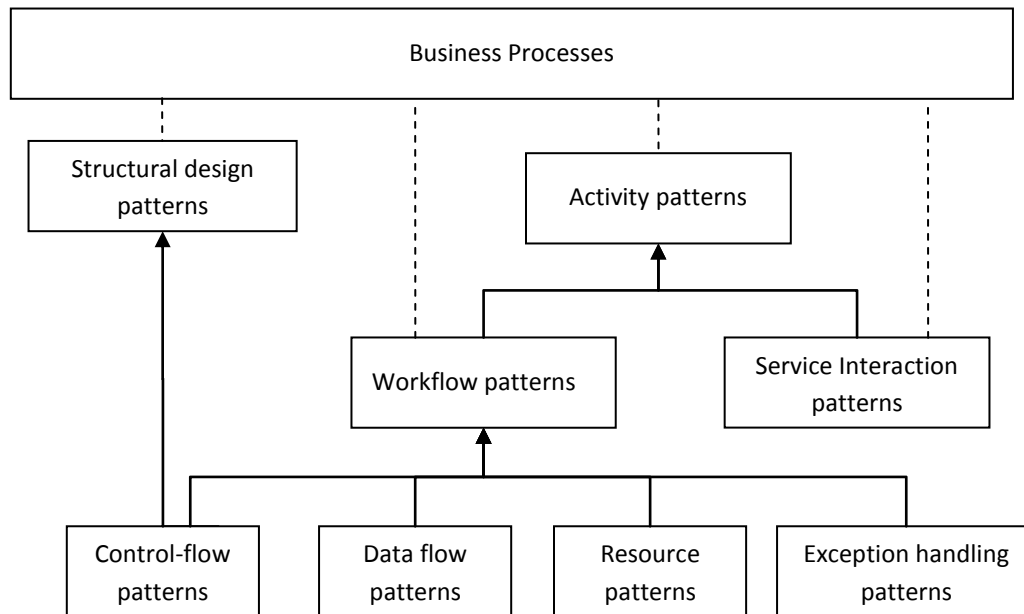


Figure 17 Patterns in business processes overview

5.3 Related work

Although MDA model transformations are very well explored, business process model transformations are still subject to research and not yet deployed or standardized throughout the industry. [Stein, Kühne and Ivanov 2009] have evaluated literature on business to IT transformations. They have also evaluated vertical and horizontal transformation approaches in the period of 2004-2008. The horizontal transformations are usually from EPC or BPMN to BPEL. The vertical transformations usually map between UML profiles, annotated EPC, Domain Specific Languages (DSL) to BPEL. From their analysis they have derived certain axioms for their framework for business to IT transformation which are taken into consideration in the conceptual framework.

Scientific literature has reviewed for approaches for is vertical transformations of business process models in general, and transformation between EPC and BPMN. Approaches found in literature are summarized in Table 6. For each source the following is depicted:

- Description: short description of the approach.
- Type of transformation: Horizontal transformation support is noted by "H". Vertical transformation is depicted by "V".
- Domain: depicts the domain the approach is designed for.
- Model mapping: the example model mappings in the papers.
- Patterns: type of patterns used for mapping data.

Parts of several approaches have been selected and combined for the model transformations as defined in the conceptual framework. In the subsequent sections, arguments for the decisions which approaches were selected are given.

Authors	Description	Type of transformation	Domain	Model mapping	Ontology	Patterns
[Dreiling, et al. 2008]	Meta-level Process configurations for vertical transformation	H V	Enterprise systems	CBS to EPC to YAWL	Metadata mapping according to MOF principles	Control-flow
[Weber, Markovic and Drumm 2007]	A framework based on components and semantic web services for business process model composition, Integrated Project SUPER.	V	Enterprise Applications & SOA	-	Web service ontology with WSMO	-
[Koehler, et al. 2008]	IBM proposes a methodology for vertical transformation and describes each step in their methodology in detail	H V	Business-driven development	UML activity to BPEL	State machine mapping and Turing equivalence	-
[Brahe and Bordbar 2007]	DSML and parameterized patterns.	V	Service Oriented Enterprise	UML activity	DSML	Parameterized patterns
[Theling, et al. 2005] [Vanderhaeghen, et al. 2005]	Cross-enterprise approach with shared repository and xml-transformation.	H	Collaborative Business Process Management	EPC to BPMN	BPMN repository	-
[Hoyer, Bucherer and Schnabel 2008]	From private EPC to public BPMN transformation by means of an intermediate EPC, determined certain abstraction rules for transformation.	H	e-business	Private EPC to public BPMN	-	Rules for transforming certain patterns in EPC to BPMN
[Murzek, Kramler and Michlmayr 2006]	An abstraction of the workflow control-flow patterns results in structural patterns.	H	Business Process Modelling	ADONIS to EPC	-	Structural patterns based on Control-flow WP
[Roser and Bauer 2005]	MDA meta-level and Ontology mapping based on meta-models. An approach for automation. Result is onMT-Tool.	H	Enterprise systems integration	EPC to "Service Model" via OWL	Mapping between meta-model and Ontology technology space	-

Table 6 Related work Approaches for business to IT transformation

5.4 Horizontal transformation

Horizontal transformation requires mapping of meta-level descriptions and ontological evaluation of the modelling languages [Höfferer 2007]. Analyzing the meta-models will provide insight on which constructs of the modelling language can be mapped. Several meta-model frameworks for evaluating process modelling languages have been developed, including [Söderström, et al. 2002] [List and Korherr 2006] and [Murzek (1) and Kramler 2007]. The ontological evaluation is achieved by an upper ontology, like BWW, which is used as reference framework to evaluate modelling languages in depicting real life concepts. Constructs belonging to the same category can be considered as semantically equivalent. On one hand, the languages are compared on their grammar, on the other hand, the languages are compared on how they can be mapped ontologically. An automation of this method of mapping is offered by the tool ontMT, which integrates ontologies for horizontal transformations by lifting each meta-model to an ontology for mapping [Roser and Bauer 2006]. BWW can be used as reference model for conceptual modelling languages and provides a stronger basis for evaluation. In [P. Green 2000] [M. Indulska, et al. 2007] detailed BWW model is explained.

Meta-level mapping provides a one-to-one construct mapping of concepts of two modelling languages, which is not sufficient to achieve semantic interoperability. One-to-one mapping of constructs might create a mapping between source and target where the result is not semantically correct. Patterns can eliminate this problem. Patterns capture the structures which express recurring business concepts. Patterns for business processes can be considered from different abstraction levels: workflow patterns and structural design patterns. Workflow patterns are the building blocks for structural design patterns.

The approach for horizontal mapping is summarized in Figure 18. In order to realize horizontal transformation of business processes conceptually, the source and target modelling language need to be **mapped on Meta-level** and for **structural design patterns**. Structural design patterns capturing generic control flow concepts within process languages. Workflow patterns are the foundation for structural patterns. Structural patterns capture concepts within processes which are omitted by one-to-one mapping. Meta-level mapping provides the basis for mapping the constructs. Using a **reference ontology** a semantic foundation for the mapping can be established.

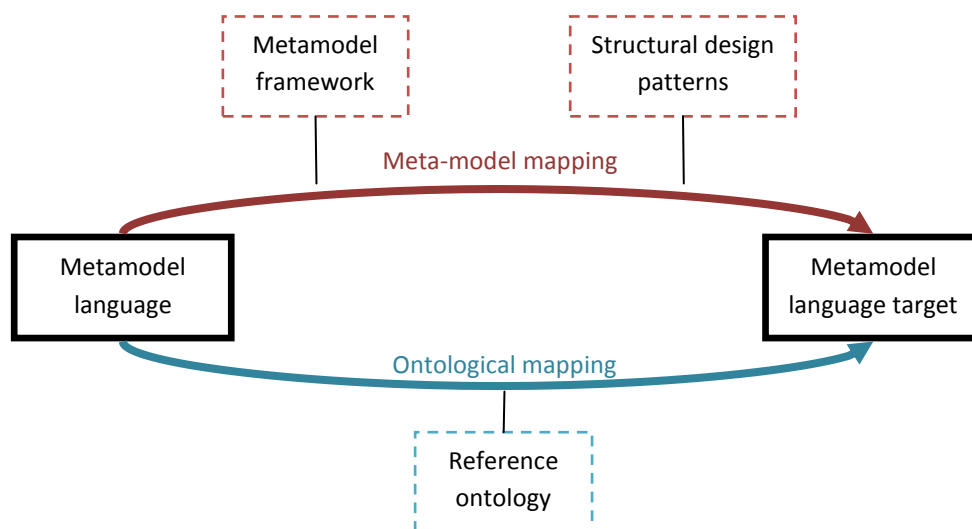


Figure 18 Horizontal mapping approach

For (run-time) workflow process models several principles have been derived guiding transformation without losing structural equivalence [Sadiq and Orłowska 2000]. They have developed structural patterns based on a reference language according to the Workflow Management Coalition (WfMC). When the source and target language are mapped onto the reference language, the exchangeable structures represent a horizontal mapping. They have identified three classes for transformation: equivalence, imply and subsume. These principles are not applied in this thesis, as it does not address exchange of run-time models.

5.5 Vertical transformation

Vertical transformation is transforming models from one perspective to another perspective. This transformation regards models within one specific modelling language, thus one meta-model. Process models are either refined or enriched when transforming between perspectives. When a high-level concept needs to be implemented, the abstraction level diminishes. The other way around, a group of activities in an executable process represents one high-level concept. Currently, vertical transformation is done manually, requiring collaboration of both a Business and IT expert. Providing (semi-)automation for vertical transformation will eliminate redundant modelling and ease the transition from business to IT models.

(Semi-)Automation of vertical transformation has some challenges. First, each process is organization specific in a certain business domain. The processes are modelled from a specific perspective in their *own vocabulary and terminology*, which impacts the understanding of processes [Mendling, Reijers and Recker 2009]. A distinction can be made between the vocabulary and terminology from management, business and IT perspective. Second, another challenging aspect is the *adoption scenarios* of BPD SOA. When starting in green field, solutions can be easily adopted, as there are probably no legacies which have to be dealt with. This has impact on the information set for transformation. However, when legacy systems exist certain IT processes are basis for concepts on higher level. In reality the middle-out scenario is most common, and both top-down and bottom-up cases are applicable. Transformations between abstraction layers have to deal with legacies. And third, for automation *formal and explicit languages are required*. Business processes can be modelled in semi-formal languages.

There are two types of options for vertical transformation: pattern based or semantic BPM. Pattern based solutions use domain specific modelling languages (DSML) to capture specific concepts in one perspective and transform to a matching concept from another perspective based on patterns. A domain specific modelling language is developed to capture domain specific concepts and information. This is in contrast with the EPC and BPMN, which are general purpose modelling languages, and not specifically designed for supporting modelling in a domain's own vocabulary and terminology [Brahe and Bordbar 2007]. A DSML can be created by extending a general purpose language with specific domain terminology. DSML can also be created from scratch for specific domains. An attempt has been made for the public administrations domain [Becker, Pfeiffer and Räckers 2007]. They have developed a domain specific modelling method PICTURE meeting the reorganization conditions in the domain of public administrations, capturing concepts that EPC and BPMN cannot. Within the BPM field, DSML have been developed to capture monitoring, measurement and control concerns, complementing business processes [Gonzalez, Casallas and Deridder 2009].

A major research initiative for using Semantic Web Service technologies in Business Process Management is addressed by the SUPER project [Information Society Technologies 2006]. Semantic BPM addresses ontology languages and semantic web frameworks to represent spheres in an enterprise to enable automation of transformations in BPM [Roser and Bauer 2005] [Bhat, et al. 2007] [Hepp and Roman 2007]. Semantic BPM enables automatic detection of process fragments and composing processes by using semantic web service technologies is central in this research. The main idea is to capture executable artefacts such as semantic web services and compose this to processes. The algorithm in this solution relies on AI planning for service composition [Weber, Markovic and

Drumm 2007]. Both solutions appoint that having an understanding and consensus on vocabulary terminology of the business domain is the enabler for vertical transformation. This understanding can be provided by domain ontology.

Dealing with two existing tools there is a constraint on the extent technologies can be implemented by the vendors. The semantic BPM approach requires extensive technological support, including lifting to semantic processes [Thomas and Fellmann 2007] [De Nicola, et al. 2008], semantic patterns and natural language processing for annotation [Bögl, et al. 2009], and AI techniques for service composition. The technological complexity does not only rise at implementation of these techniques, but also during the exchange between tools as additional semantic structures and information are added. Currently, no vendor provides support for semantic BPM, but they do notice it as a future promise. In addition to the proper tool support, BPM maturity in an organization need to at high-level [Scheer and Klueckmann, BPM 3.0 2009]. The pattern based approach has a lower implementation effort, as it preserves the current situation for modelling. Therefore, in this framework the pattern based approach is recommended for vertical transformation, and researched in depth.

For BPD SOA the approach for vertical transformation is pattern based [Brahe and Bordbar 2007]. A **Domain Specific Modelling Language (DSML)** is developed for each perspective. **The transformation pattern** transforms each task (activity) from one perspective to another perspective. The approach is displayed in Figure 19 with the three perspectives as defined in the framework. A transformation pattern, also known as parameterized pattern, exists for each specific object defined in the domain specific language. This pattern consists of:

- The pattern template: depicting the structure the entity will result in within the next perspective.
- Any additional parameters: any additional information which should be added.
- Transformation rules: the coordination of the transformation.

A model on business level M_B is transformed via the parameterized pattern to a model M_O on operational (functional) level, which in turn can be converted to a model M_E on executable (IT) level.

Specific business domain patterns can capture business-modelling problems and their solutions. The method consists of an identification phase, in which a domain analysis is performed and creation of the DSML. The second phase consists of process orientation and pattern discovery. During business process modelling thinking of patterns is required.

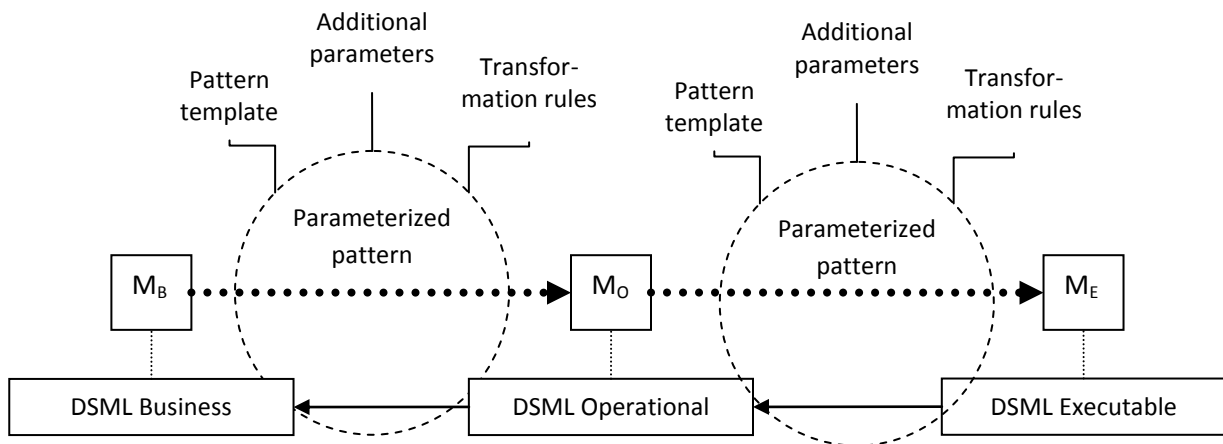


Figure 19 DSML and Pattern based Vertical transformation (based on [Brahe and Bordbar 2007]).

The first phase in vertical transformation is domain analysis and development of DSMLs. This requires the development of common vocabulary and terminology of the business domain. A domain ontology is the ontological foundation for the DSMLs [Tairas, Mernik and Gray 2009].

There are several steps identified for developing DSMLs [Oberortner, Zdun and Dustdar 2008]:

- (1) Identifying domain concepts and relations, and abstraction levels: Understand domain concepts and the relation with implementation. Determine whether top-down, bottom-up or parallel development is required.
- (2) Specifying the language models: Specify the meta-model for the DSML. Aligning this with the stakeholder’s modelling conventions improves recognisability.
- (3) Creating notations for the language based on the language models: visual representing the DSML.
- (4) Generate code of valid domain models: to generate code from the validated domain specific models.

A domain is subject to different viewpoints. For instance, in the banking industry the activity 'check account' can refer from technological point of view to several activities which represent the concept of process Order, i.e. 'getAccount' and 'validateAccount'. Developing and maintaining domain ontology is to create consensus on the reality from multiple perspectives. Ontology frameworks are described in section 5.1. The ontology can be used as basis to determine which concepts on business level match the concepts on a more technical level. Within this framework each perspective requires its own DSML to capture its domain concepts and knowledge. Each stakeholder’s perspective is represented and is readable and understandable within its domain. Important is that relations should exist between the DSMLs of each perspective for one domain, where the low-level DSML extends the high-level DSML. Otherwise the transition between domains is hard to realize, when no relation exist between the domains. These relations are depicted in Figure 20. Three abstraction levels are determined for the DSML, from executable to operational to business. The DSML is thus related to the domain ontology (step 1) , and derived from the meta-model of the process models (step 2). The domain ontology and its relation to meta-models are explained in section 5.1. Two guidelines are proposed to ensure the maintainability of the DSMLs [Oberortner, Zdun and Dustdar 2008]:

- Relation between modelling language and DSML should be evident.
- Complexity should be maintained by paying attention to redundancy, inconsistencies and overlap of DSMLs.

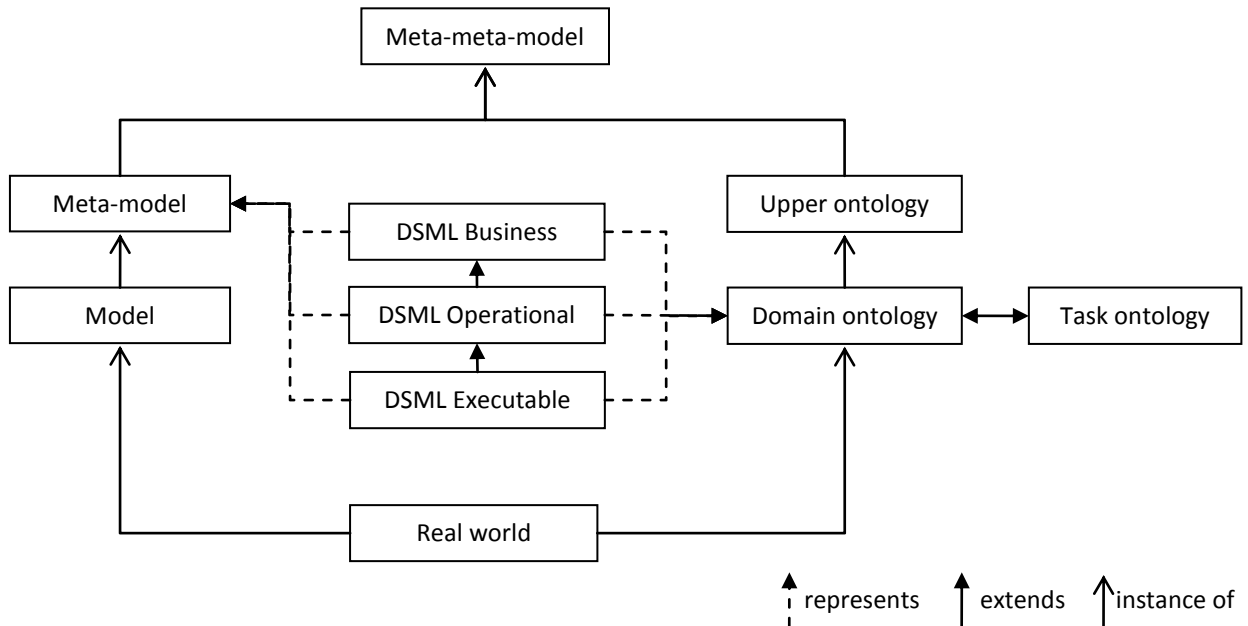


Figure 20 Relation Meta-model, DSML and Ontologies

In this approach the transformation is between task types. Identification and recognition of these task types are challenging. Domain ontology can contribute in identifying domain vocabulary and terminology, and develop a complete DSML. Task ontology can help in identifying different task types. Task ontology is either [Ikeda, et al. 1998]:

- a) task decomposition in subtasks and categorization
- b) specification of concepts and relations of a task of interest

For this approach, a) is required to determine the task types categorizations. The integration domain and task ontology result in application ontologies [Gomez-Perez, Corcho and Fernandez-Lopez 2004].

The second phase is process orientation and patterns discovery. Patterns capture recurrent business concepts and enable reuse. The use of patterns within a business domain for capturing business knowledge is recognized by [Seruca and Loucopoulos 2003]. A systematic approach including domain analysis and process orientation is the foundation for capturing patterns. Patterns need to be identified on operational and executable level, as the concept on business level is the highest abstraction level. Activity patterns can be considered as patterns on operational level. The recurrent business functions, thus activities, represent the tasks types which can be transformed from DSML Business. The activity patterns can be used as reference in defining the parameterized patterns. Each identified type of activity pattern can be considered as concept in the domain ontology. The recurrent business function corresponds with the high-level business task types. The implementation of these activity patterns is specified on executable level workflow patterns and/or service integration patterns. From these latter patterns task types can be derived for a DSML on operational and executable level. Example task types for an operational DSML are BPMN task types, and for an executable DSML, BPEL may provide task types. An example of parameterized patterns is illustrated in Table 7, the vertical transformation of an approval task type from high-level business to IT level. The perspectives correspond with the perspectives defined in the framework (3.3, 4.2).

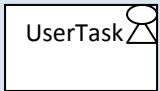
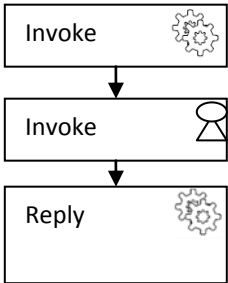
Task type				
Approval		Approve Request		
Parameterized pattern				
Domain	Description	Pattern	Additional parameters	Transformation rules
Business to Operational	Approval		Number of Roles Assign Manual or System (service)	Change the approval to a manual or user task, the number corresponding with number of approvers.
Operational to Executable	UserTask		Single, Concurrent or Iterative Services Descriptions System Events (Time out, Send/Receive messages)	Create the single, loop or concurrent structure Attach services Determine timeouts when necessary

Table 7 Example of parameterized patterns for vertical transformation

Part III.

Using the framework for ARIS and Cordys

Chapter 6

BPD SOA modelling coupling for ARIS and Cordys

To enable round-trip development between ARIS and Cordys, from a high-level business process model to a technical executable model, decomposition of the problem according to the framework in Chapter 4 will guide the transformation. The context part of the framework is presented in 6.1. Comparison of the life cycles and models of ARIS and Cordys is presented in section 6.1.1, respectively 6.1.2. This is based on analysis of ARIS, Appendix A, and Cordys, Appendix B. The coupling points are presented in 6.1.3. The dimension part is described in 6.2. Model transformations and a traceability model are presented in Chapter 6.

6.1 Context analysis

The context analysis addresses ARIS and Cordys, determining the points and requirements for coupling. To determine which coupling scenarios are applicable, the following requirements are taken into consideration:

1. Exchange EPC and Cordys BPMN between ARIS and Cordys on different granularity levels.
2. Synchronization of process models and related services between the two tools to keep consistency and coherence of the models, including traceability of exchanged and transformed models/ constructs in ARIS and Cordys.

In addition to these requirements the following wishes have been determined:

3. Synchronization of organizational models/ information.
4. Synchronization of KPI models/information.
5. Synchronization of Risks and Control models/information.
6. Synchronization of UIs.

6.1.1 The life cycle comparison

Figure 21 depicts the theoretical life cycle of BPD SOA (Figure 2) against the life cycle of Cordys (in blue) and IDS Scheer (in magenta). The strategy phase of IDS Scheer addresses strategic to operational objectives for one organization at enterprise level, while the BPM cycle of Cordys determines the objectives on functional level. *Process analysis* resembles *qualify and analysis* of Cordys, and the as-is phase in *process design* of IDS Scheer. *Process design* is for to-be processes for BPD SOA, and in both ARIS and Cordys operational processes need to be designed and modelled for execution. The *process implementation* phase is in essence the deployment of executable processes. Both IDS Scheer and Cordys distinguish this as a phase. In Cordys implementation is actually facilitated, while ARIS considers this to be realized in an external execution platform. The last four phases in the cycle, *enactment*, *monitoring*, *evaluation*, and *simulation* are summarized in *Run & Monitor* for Cordys and *Process Control* for IDS Scheer. Process monitoring (real-time), process performance, analysis for redesign, and optional simulation for validation are activities within these phases.

Strategy	Process analysis	Process design	Process implementation	Process enactment	Process monitoring	Process evaluation	Process simulation
	Qualify & Analysis	Design & Model	Design & Deploy	Run & monitor			
Process strategy	Process design		Process Implementation	Process control			

Figure 21 BPD SOA Life cycle comparison

Analysis of the life cycle of ARIS (in Appendix A1) and Cordys (in Appendix B.1) have shown that the first difference is the focus of life cycles. Cordys has a more project oriented focus. The activities and products are focused on delivering an operational solution. ARIS, on the other hand, specifies more content on strategic level and elaborates on the business context of SOA, including alignment of technical services with business objectives via business services. This difference also becomes evident when analyzing their methods offered for implementing BPM using ARIS or Cordys. Furthermore, the roles are compared to provide detail on what is expected of a certain role in ARIS and Cordys in Appendix C.3. The roles distinguished from theory provide the basis for a comparison. The essence of this comparison is to show what is meant by a role and their responsibilities as depicted by the vendors. The discussion remains - what's in a name?

On high level the difference is clear, however, in depth analysis and execution of these activities show similarities and differences between ARIS and Cordys. From modelling perspective, detailed analysis of the *process analysis* and *process design* is relevant for determination of the possible coupling points.

6.1.2 ARIS and Cordys models comparison

Models are context specific and developed from a certain perspective. Although EPC and BPMN can both be used for business modelling, there is a difference in what they can depict. EPC's flow chart influences and the limited amount of constructs and rules allow easy understanding. BPMN on the other hand is more complex. Modelling skills are required in order to comprehend the many different types of constructs and rules. One has to be capable to handle the design freedom. Figure 22 depicts the ARIS and Cordys models mapping based on description. Details on the model definitions and constructs can be found in Appendix A.2.1 for ARIS models, and Appendix B.2 for Cordys BPMN. Cordys BPMN process models are referred to as C-BPMN. The types of models are mapped against the three perspectives in modelling (business, functional and IT). This matches the viewpoints identified in 3.3.

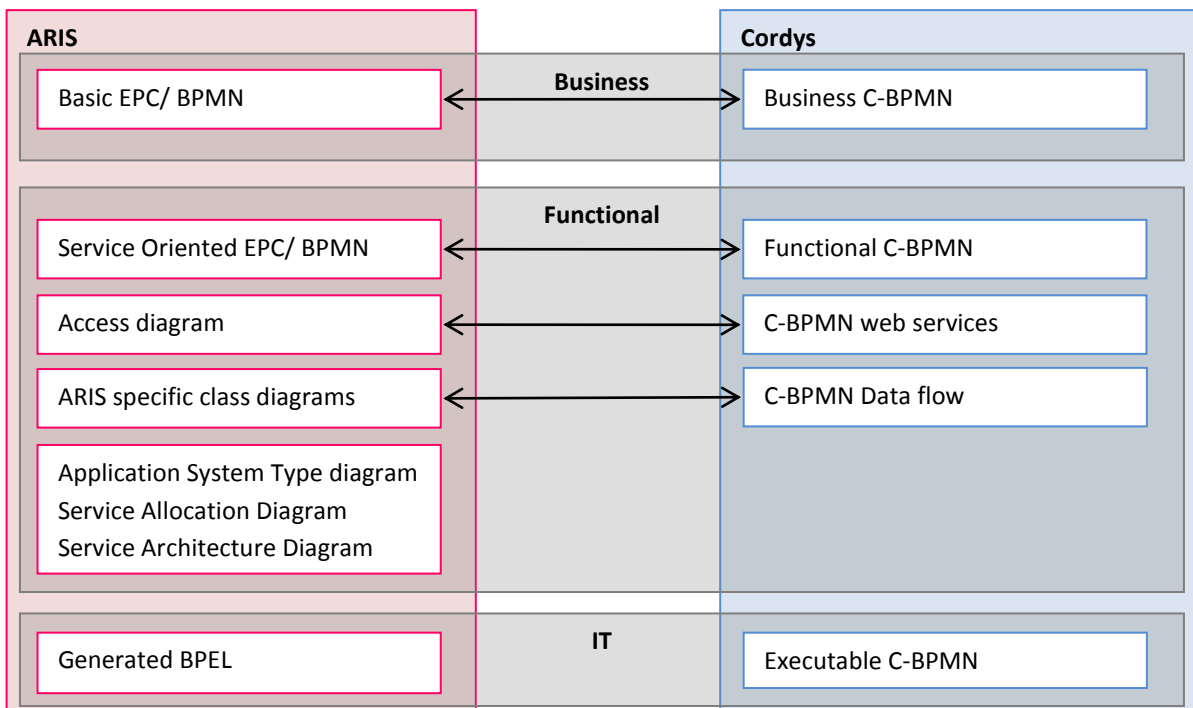


Figure 22 High level mapping ARIS and Cordys models (and properties) based on description

After this assessment, EPC offers more explicit modelling compared to C-BPMN. In BPMN, constructs have more variations, referred to as properties in Cordys. These C-BPMN properties include web services and data-flows. These can be conceptually mapped to what an access diagram and ARIS specific class diagram represent. Other service oriented ARIS models cannot directly be mapped to a Cordys concept in process modelling. On IT level, BPEL could be converted to executable C-BPMN. But this is not in scope of this thesis, as interoperability of business process models is researched.

Analysis of the activities for modelling processes (depicted in Appendix A.2.2 for ARIS, Appendix B.2.1 for Cordys) with regard to BPD SOA, presents that ARIS and Cordys both support the modelling of *service-oriented business processes, service discovery and design*. Also maintaining the *functional and data glossary* is supported by both tools. Differences include *business services* in ARIS, *user interfaces* in Cordys, *workflow assignments* in Cordys, and *level of abstraction in data flows*. In conclusion, both ARIS and Cordys support modelling from high-level business perspective to a functional perspective, however, the assignments of services, executing roles, data flow, and UIs is done on a different abstraction level.

6.1.3 Coupling points ARIS and Cordys

Two coupling points are identified in the BPD SOA life cycle (Figure 23). One possibility would be to export a simple EPC or BPMN process from ARIS to Cordys for further development. Another possibility is to create a complete to-be process design and model service orchestration in ARIS, subsequently importing these into Cordys for execution. The couple points for monitoring of processes are not considered within the scope of this research, as that can be considered as another challenging research.

Option	Strategy	Process analysis	Process design	Process implementation	Process enactment	Process monitoring	Process evaluation	Process simulation
Distribution ARIS – Cordys								
1	Strategy	Process analysis	Process design	Process implementation	Process enactment	Process monitoring	Process evaluation	Process simulation
2	Strategy	Process analysis	Process design	Process implementation	Process enactment	Process monitoring	Process evaluation	Process simulation
	ARIS			Cordys		Cordys	ARIS	

Figure 23 ARIS-Cordys coupling points against theoretical life cycle

An adoption scenario (2.3) determines which couple point is most suitable. The most influential aspect is whether service discovery is done on business service or web service level. Cordys does not support business service modelling. If service orchestration is managed from business perspective, option 2 is most suitable for exchange. ARIS provides more facilities for aligning business objectives and functional design. However, option 1 is suitable for projects with strong operational purpose, such as workflow projects. In this case, designing business services does not have a high priority.

6.1.4 In perspective with POINT

In ARIS process modelling can be done according to the POINT method (based on DEMO methodology [Dietz and Hoogervorst 2008]) customized for IDS Scheer. Figure 24 illustrates the POINT method for modelling in relation with the ARIS and Cordys process models. Each activity in POINT is linked to ARIS and Cordys models, and grouped to the perspectives as defined in 3.3. *Context analysis* determines the areas of interests, expressed in external stakeholders/actors who have an association with the enterprise. For each area of interest trigger analysis is

performed. *Trigger analysis* defines the triggers of a process in the enterprise. These are the end-to-end processes. *End-to-end processes* are complete high-level processes, which are not composed with other processes to form a larger process within an area of interest. Usually they consist of a number of sub processes. *Decomposing* end-to-end processes results in several granularity levels, depending on the structure of the organizations. End-to-end processes are decomposed to processes across departments (*sub process level*), which in turn can be decomposed to across roles processes (*sub sub processes level*), and on lowest level from one role point of view (*work-level*). The lowest level should not contain any sub processes. *Clustering* is to cluster the end-to-end processes in a logical sense for the business perspective. [Maas 2008]

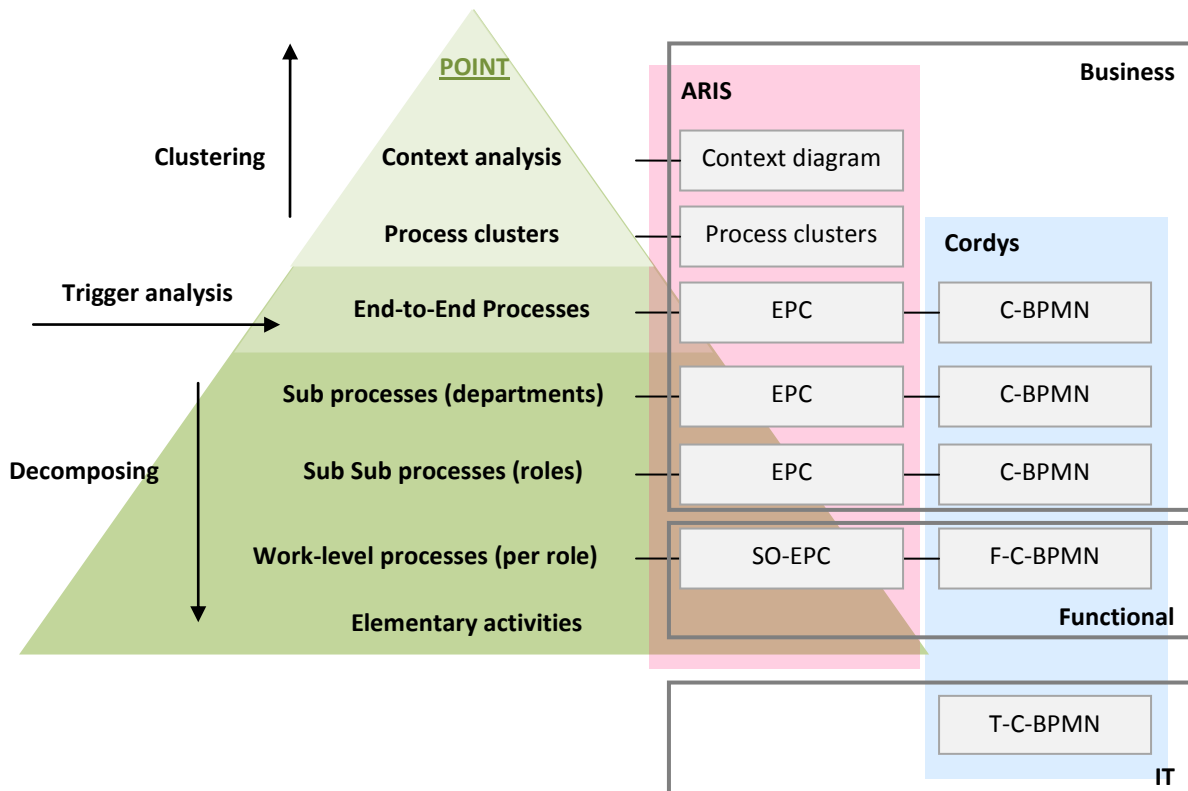


Figure 24 Point Process Modelling related to ARIS and Cordys process modelling

6.2 Dimensions

With the context clarified, the coupling regarding model transformations is addressed. All sorts of forms for model transformation are applicable for the ARIS and Cordys case. Figure 25 depicts all possible process model transformations between ARIS and Cordys which can be addressed conceptually. Between each layer vertical transformation needs to be realized, and at either the business or functional abstraction layer a horizontal transformation needs to be realized. In the tool analysis the models relevant in BPD SOA context are identified. The models and relevant constructs are categorized onto the abstraction layers to determine which information needs to be exchanged or enriched. At business level, a simple EPC can be translated to a simple C-BPMN, and for redesign synchronization is required. Note this simple EPC is expected to be on across-roles process granularity level (in Figure 24). Furthermore, a Function Allocation Diagram (FAD) (Appendix A.2.1) depicts relevant KPIs and objectives. The EPC and FAD need to be integrated, as KPI and objectives are relevant information during monitoring. At functional level Service Oriented (SO-) EPC models need to be translated to functional C-BPMN. Also within this layer synchronization is required. Additionally the ARIS allocation and architecture diagrams depict relevant information

which is required in functional C-BPMN. This requires integration of Service Oriented EPC models and linked models resulting in a Functional C-BPMN. The IT level contains technical, executable processes, which are only relevant in Cordys.

Considering the requirements and the case of the Dutch Insurance Company, first priority is to facilitate synchronization and traceability between the two vendor tools. But as this organization matures on BPD SOA, with more service oriented development, it is progressive to take a look at (semi-)automation of vertical transformation. It is desired to have syntactical, semantic and pragmatic interoperability between ARIS and Cordys. Syntactical and semantic are subject to model transformations, where pragmatic interoperability is achieved by traceability.

Horizontal transformation is done on functional level, because it represents the most complete picture regarding model and data exchange between ARIS and Cordys. The business level mapping is covered within the functional level, as the constructs on business level are also present on functional level. The business level mapping is therefore implicitly researched (depicted by the dotted arrows). The model transformations for ARIS and Cordys are described in Chapter 10.

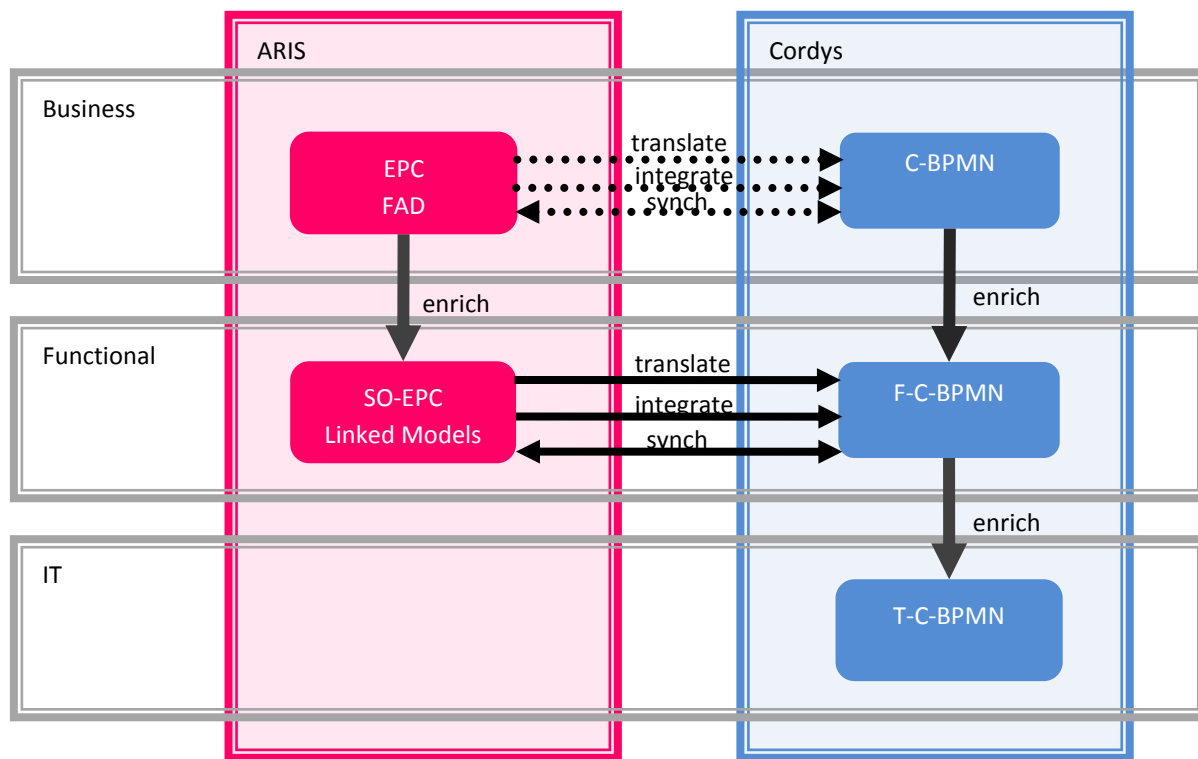


Figure 25 Conceptual transformations in ARIS and Cordys

Chapter 7

Model transformations and Traceability for ARIS and Cordys

This chapter describes the model transformations for ARIS EPC and Cordys BPMN according to Figure 25 and the approaches described in Chapter 5. Horizontal transformation between EPC and Cordys BPMN is described in section 7.1. Vertical transformation is presented in section 7.3. A traceability concept is proposed in 7.4. In Cordys terminology, the Cordys subset of BPMN is referred to as BPM, Business Process Model. In this report the term C-BPMN, Cordys BPMN, is used to refer to the Cordys BPM.

7.1 Horizontal transformation

Model translations are required between EPC and C-BPMN models. In addition, integration is required as ARIS uses other types of models to depict architecture and allocation diagrams for BPD SOA. Analyzing the meta-modelling language of these models, they are either EPC or UML based. In this context only the UML component diagram is UML based. The UML component diagram depicts the specific WSDL details in model structure. This diagram can be exported as WSDL and imported in Cordys. There is no need to integrate the UML with EPC. The other EPC based diagrams (access diagrams, service allocation diagrams, service architecture diagram, and application type architecture diagram) depict information related to the EPC. These models can be merged with the EPC in ARIS to retrieve a complete model. The constructs used in these models are also considered be in the subset for mapping of EPC. When the EPC is transformed to BPMN, it needs to be synchronized to the Cordys platform. The horizontal mapping addresses model integration, translation and synchronization. Integration and translation are addressed conceptually. Synchronization is achieved by exchange formats.

First step is analyzing the meta-models and using reference ontology to map constructs. This is the basis for translating one language to another. Second is to use structural design patterns to capture structural patterns which are not addressed by one-to-one construct mapping. Third, an assessment is made on how to translate non-main elements and models into Cordys concepts. Non-main elements do not belong to the main set of the modelling language. In summary horizontal mapping EPC and BPMN is:

- Integration of the EPC and EPC based models by model merging functionality in ARIS.
- Translation of the elements of service oriented EPC (Appendix A.2.1) and C-BPMN subset (Appendix B.2).
- Mapping of structural patterns.
- Assessment of translating non-main elements.

7.1.1 Mapping of meta-level

A meta-level mapping of the subset of BPMN for Cordys and EPC is made. The meta-model analysis of EPC and C-BPMN is found in Appendix D and provides insight on the constructs that can be mapped. The meta-model of BPMN makes a distinction between flow elements, artefacts and interaction elements, which forms a contextual relationship. Dependent on the elements usage, it is grouped to a certain context. No formal meta-model has been defined for EPC. The approach chosen to create the EPC meta-model is based on the views of the ARIS methodology, which means grouping on similar semantic relationships. A meta-level mapping is also specified in [List and Korherr 2006] and [Murzek (1) and Kramler 2007].

In addition to meta-model comparison, the mapping to a reference ontology provides a more semantic foundation for the mapping. A proposed meta-level mapping (5.4) is to evaluate both meta-models to the BWW ontology [Wand and Weber 1990] [Dreiling, et al. 2008]. Constructs with the same ontological category could be mapped to each other. The ontological categories depict the real-life concept to which a construct can be mapped.

The evaluation of BPMN and EPC on BWW has shown that both languages have some deficiencies. The mapping of EPC and BPMN to BWW is shown in Appendix E.1. EPC evaluation showed to what extent they are representative for perceptions of business analysts [Green and Rosemann 1999]. The major conclusions were that EPC is not sufficient to show all state and transformation laws. Furthermore, an EPC lacks representation of the scope and boundaries of a system. Evaluation of BPMN to BWW has shown to some extent, the same conclusions [Recker, Wohed and Rosemann 2006]. BPMN also lacks state representation, and does not support system demarcation. At one hand, BPMN has a lot of constructs with no real-life meaning. On the other hand, construct overload to one particular ontological category causes confusion. This is the case for lane and pool. In addition, there are constructs which belong to several ontological categorizations.

Meta-model analysis and BWW evaluation and has led to the following:

- A mapping between activities and functions can be made. BPMN defines more types of activities than functions are available in EPC. A bidirectional mapping exists for simple atomic tasks and functions, and aggregated functions (simple sub processes). Functions and Activities have the same ontological meaning according to BWW. They depict a transformation from one state to another. On meta-level they are both described as transforming input to output, as an executable element. The different types of tasks and the loop characteristics of BPMN cannot be mapped directly to EPC constructs. The process interface is a special case. The process interface either is a start or end of process, referring to another process on the same granularity level. In current EPC to BPMN transformation in ARIS Governance Engine Module, the process interface is translated to an intermediate catch or throw link event. Cordys does not support the link event.
- Events in BPMN and EPC do not have the same definition, which is also depicted in the BWW model. EPC Event describes an *occurrence of a state* triggering or resulting from a function, while a BPMN Event is *something what happens* with cause and impact on process flow (catch and throw events). In the BWW model the EPC events is classified as state of a thing, while BPMN events are events occurring in a thing. Exceptions are for start and end events. The intermediate events and different types of events cannot be mapped to EPC from BPMN.
- Gateways and operators do not have a real-life meaning according to the BWW model. Analyzing the meta-models depicts that they can be mapped. However, BPMN has more types of gateways than EPC has operators. Workflow patterns will provide more insight in operator mapping.
- Flows represent lawful transformations. There a different types of flows in BPMN. The uncontrolled sequence flow can be bidirectional mapped with a normal flow in EPC. The association flows in BPMN can be mapped with information flows in EPC. Conditional flows are supported by both languages by their attributes, however in EPC no distinct graphical notation is provided.
- A unidirectional mapping from EPC organizational units, and positions to BPMN lanes can be made according to the BWW things. The other way around is also possible, but a lane can represent two different constructs in EPC. A choice needs to be made. Another discussion point arises when userTasks in BPMN are considered. According to the definition, this can be represented by an organizational unit or position which (always) links to a function in EPC. There are several options for mapping organizational aspects in both languages. According to BWW the organizational constructs can be considered as things. Cordys defines organizational units and roles. These can be attached to a task in order to create a UserTask. ARIS EPC also defines Persons, another organizational element.

There are several other constructs, which are not evaluated by BWW. These concern either interaction or data elements important in the BPD SOA context. Difference on abstraction level exists for mapping data flows and services. On conceptual level no representation for the concepts business services and capabilities is possible in BPMN. However, the software service type in EPC, which is related to business services/capabilities, can be considered as a service task in BPMN. A service task is actually a task with an associated web service in C-BPMN. Data flows are represented by data clusters in EPC. This cannot be specified in BPMN. However, C-BPMN uses message maps to specify data in and out flows for tasks. This is on a different abstraction level than in EPC. The actual input and output expected during execution needs to be specified. Thus a technical data input output flow. Assessment of how these non-main elements can be mapped is found in 7.1.4.

BPMN lacks representation for the concepts of KPIs, objectives and application system types. EPC lacks representation of several type of tasks, type of events, type of gateways and associations. BPMN is much richer in depicting variety of types of constructs, however, lacks representation for business context related constructs such as higher level services, objectives and systems. A service oriented EPC can depict much context, however, lacks representation for different types of the constructs.

The mapping on meta-level for the service oriented EPC and C-BPMN are summarized in Table 8. An arrow depicts a directional mapping. A cross means no mapping. NOTE: the mapping of non-main elements is not possible on construct level, alternatives are considered in 7.1.4.

Service Oriented EPC		C-BPMN	
Function		Activity	
Basic Function	←	→	Task
-	X	X	Task loop
-	X	X	Task Multiple Instance
Aggregated Function	←	→	Subprocess
Process Interface	X	X	-
Event		Event	
Event	X	X	Event
Start Event	←	→	Start Event
End Event	←	→	End Event
Start Event	←	X	Message Start
-	X	X	Message Intermediate Catch
-	X	X	Message Intermediate Throwing
End Event	←	X	Message End
Start Event	←	X	Timer Start
-	X	X	Timer Intermediate Catch
-	X	X	Error Intermediate Catch
End Event	←	X	Error End
-	X	X	Cancel Intermediate Catch
End Event	←	X	Cancel End
-	X	X	Compensation Intermediate Catch
End Event	←	X	Compensation End
Start Event	←	X	Conditional Start
Start Event	←	X	Multiple Start
-	X	X	Multiple Intermediate Catch
-	X	X	Multiple Intermediate Throw
Operators		Gateways	
AND	←	→	Parallel

Service Oriented EPC			C-BPMN
OR	←	→	Inclusive
XOR	←	→	Exclusive
-	X	X	Event-Based
Flows			Flows
Sequence flow	←	→	Uncontrolled Flow
Flow + attributes	←	→	Conditional Flow
-	X	X	Default Flow
Information flow	←	→	Association
Organization			Participants
Organization Unit	←	→	Lane / UserTask (Task + Team)
Position	←	→	Lane / UserTask (Task + Role)
Person	←	→	Lane / UserTask (Task + Role)
Others			
Free-form text	←	→	Text Annotation
-	X	X	Association
Non-main			
Objective	X	X	-
KPI	X	X	-
Cluster	X	X	-
Business Service	X	X	-
Capability	X	X	-
Application System Type	X	X	-
Software Service Type	←	→	ServiceTask (Task + Web service)

Table 8 SO EPC and C-BPMN meta-level mapping

7.1.2 Workflow patterns

It is required to map certain workflow patterns to capture context and control-flow aspects which are not addressed by the one-to-one construct mapping on meta-level and mapped by the BWW methodology. One-to-one mapping of constructs might create a mapping between source and target where the target language is not semantically correct. For example when implicit joining of tasks is allowed in BPMN, EPC restricts multiple incoming flows in functions directly, but this can be solved with an AND-join operator.

Within this research only the workflow Control-flow patterns are considered. Two reasons: 1) Control-flow patterns address the core of processes, whereas the remaining are supporting patterns. 2) Data, resource and exception patterns are well researched for BPMN [Wohed, et al. 2006], but none have been derived for EPC. BPMN 1.0 is analyzed on workflow patterns and compared to UML 2.0 AD and BPEL. Evaluations are made in terms of Control flow, Data and Resource patterns. Most coverage is found for control flow patterns, nearly half coverage for data flow and very limited support for resource patterns. Research of all patterns for EPC would require extensive effort and time, and are therefore out of scope. The support of workflow patterns for EPC and BPMN are compared to see which concepts can be transformed and which not. This comparison is found in Appendix E.2.

The control-flow patterns depict the mapping of operators in EPC and BPMN. The operators of EPC can all be mapped to gateways in BPMN, but not the other way around. Table 9 summarizes the possible EPC – BPMN operator mappings. BPMN has gateways handling specific conditions, which are not possible in EPC. The multi merge pattern is for instance not possible in EPC. An extension of EPC with an empty operator can realize this pattern [Mendling, Neumann and Nüttgens 2005]. Other unsupported patterns by EPC related to decision structures are discriminator

(Complex), deferred choice (Event-based), blocking discriminator (Complex), Cancelling discriminator (Inclusive), Structured N-out-of-M Join (Complex), Blocking N-out-of-M Join (Complex), Cancelling N-out-of-M Join (Complex).

Pattern	BPMN	EPC
<u>Parallel Split</u>	Parallel Gateway split	AND-split
<u>Synchronization</u>	Parallel Gateway join	AND-join
<u>Generalised AND-Join</u>	Parallel Gateway join	AND-join
<u>Exclusive Choice</u>	Exclusive Gateway split	XOR-split
<u>Simple Merge</u>	Exclusive Gateway join	XOR-join
<u>Multi-Choice</u>	Inclusive Gateway split	OR-split
<u>Structured Synchronizing Merge</u>	Inclusive Gateway join	OR-join
<u>Multi-Merge</u>	Exclusive Gateway join	Empty operator

Table 9 Workflow pattern operators comparison EPC – BPMN

Implicit termination patterns depict a mapping between the end events of EPC and BPMN, which did not match in the BWW comparison. Patterns related to multiple instances and interleaved routing are related to the types of tasks in BPMN, which is not present in EPC functions. Furthermore, EPC does not support threads, whereas BPMN does with attributes on activities. The end event of BPMN captures more than the EPC end event. For example explicit termination, cancel activity and cancel case are possible in BPMN.

7.1.3 Structural design patterns

Structural design patterns are useful when a transformation is required between a source language with larger design freedom than the target language. The structural design patterns depict a recurring business process concept, on higher abstraction than the workflow patterns. These concepts are represented in a source modelling language, which might be semantically impossible in the target language. An alternative representation is required. Thus mapping of patterns depicting the same concept is required to provide complete transformation. The design patterns are based on workflow patterns. There are three basic patterns always present in end-to-end processes. These are the *start event* pattern, *end event* pattern and *sequence path* pattern. A sequence path is a path in the process with successive activities. Furthermore, decision patterns also occur very often. Combinations of the split/ join paths are defined as decision patterns. Three types of decision patterns exist: *parallel*, *alternative*, and *multiple alternatives*. Parallel pattern is a combination of the parallel split and synchronize workflow pattern, depicting a parallel split and join within a process. An alternative pattern is the combination of exclusive choice and simple merge workflow pattern, depicting the split and join with an exclusive alternative. Multiple alternatives is the combination of multi-choice and synchronizing merge workflow pattern, depicting splitting and joining multiple alternative branches. Considering forward engineering, transformation patterns will be extensively explored from EPC to BPMN. EPC is a language with less design freedom compared to BPMN. From there the situation for BPMN to EPC transformation will be considered, as round-trip is a desire with ARIS and Cordys coupling. Appendix G describes the detailed patterns, and the patterns are summarized in Table 10 (EPC to C-BPMN) and Table 11(C-BPMN to EPC).

Pattern	EPC	C-BPMN
Start event	Start events	Start events
End event	End events	End events
Sequence path	Function – event -Function - event	Task - task
Parallel	AND-split/AND-join	Parallel Gateway split/Parallel Gateway join
Alternative	XOR-split/ events /XOR-join	Exclusive Gateway split/ conditions/ Exclusive Gateway join
	XOR-split/ branches with no activities /XOR-join	No mapping – EPC Events may be considered as conditions following an exclusive gateway, if and only if the alternative branch contains an activity (a function).
	XOR-split/ XOR/OR/AND-split and others /XOR-join	The EPC events following the second operator should be combined by the second operator, thus event A “Second Operator” event B, to serve as condition on the branch after the first operator. Option: If the branch is the only alternative without condition, the condition could be default.
	XOR-split/ XOR/OR/AND-join and others /XOR-join	The EPC event following the join should be considered as condition
Multiple alternative	OR-split/events / OR-join	Inclusive Gateway split/conditions/ Inclusive Gateway join
	OR-split/ branches with no activities /OR-join	No mapping – EPC Events may be considered as conditions following an inclusive gateway, if and only if the alternative branch contains an activity (a function).
	OR-split/ OR/XOR/AND-split and others /OR-join	The EPC events following the second operator should be combined by the second operator, thus event A “Second Operator” event B, to serve as condition on the branch after the first operator. Option: If the branch is the only alternative without condition, the condition could be default.
	OR-split/ XOR/OR/AND-join and others / OR-join	The event following the join should be considered as condition

Table 10 Structural Design Patterns EPC – C-BPMN

Pattern	C-BPMN	EPC
Start event	start events	Start events
End event	end events	End events
Sequence path	Task	Event -Function
Parallel	Parallel Gateway split/Parallel Gateway join	AND-split/AND-join Each task after the parallel split in the parallel decision should be transformed to an event – function, if and only if it the AND-split is preceded by a function.
Alternative	Exclusive Gateway split/ conditions/ Exclusive Gateway join	XOR-split/ events /XOR-join Conditions of the exclusive split are trigger events, and replace the event of the event-function translation after an exclusive split.
Multiple alternative	Inclusive Gateway split/conditions/ Inclusive Gateway join	OR-split/events / OR-join Conditions of the inclusive split are trigger events, and replace the event of the event-function translation after an inclusive split.

Table 11 Structural Design Patterns C-BPMN to EPC

7.1.4 Assessing the mapping and exchange of non-main EPC and C-BPMN elements

The main difference between ARIS in Cordys on functional level is the granularity of the data. In this thesis transformation for main (service oriented) EPC and C-BPMN is considered. There are more contextual elements to be considered from business point of view. In ARIS development of risks and control diagrams, organizational models, and KPI models give more insight into business analysis. Several elements of these models can be linked to the EPC models. These non-main elements are analyzed, and assessed how and whether they should be exchanged to Cordys. Detail on the elements can be found in Appendix F.

Business services and capabilities

Cordys does not have equivalent concepts for business services and capabilities. When these are linked to software service types, a mapping can be made on the right granularity level. The business services and capabilities can be transformed to annotations to depict the functionality expected of a task.

Organizational elements

On high-level, organizational units and positions can be mapped. However, an ARIS role does not have any runtime associations for positions, unlike Cordys roles. A mapping can be made between an authorizing organizational unit or position in ARIS to Cordys team or role. A solution would be to map these as UserTasks in Cordys. But currently Cordys BOP-4 restricts attaching roles and teams to tasks, if no user interface is attached. Organizational units should be transformed to lanes + team. And positions/persons to lanes + roles. Both ARIS and Cordys support modelling organizational models. Exchanging these would require a separate exchange format.

KPI

KPI models can be made in both ARIS and Cordys. Exchanging then would require a separate format. KPIs can also be linked to process models in ARIS. KPIs are useful for defining process instances during monitoring. KPI are linked to measure points in ARIS KPI allocation diagram. These KPI measure points are input for Cordys Business Measures.

Business measures are logical expressions of the measurements. KPI (and measures) should be transformed to annotations.

Risks & Control

BPMN does not have construct representation for control or risks. The control is a type of function in EPC. A control is a set of activities which need to be done when a risk fires. Considering the definition this can be best done as *escalation event sub process*. However, C-BPMN does not support the escalation event. It does facilitate work escalation via properties in Cordys. Controls and Risks can best be converted to annotations. From there the correct properties can be set. Risk & Control diagrams of ARIS cannot be exchanged, as no equivalent diagrams exist in Cordys.

User Interfaces

Transforming UIs designed in ARIS to Cordys and vice versa requires a specific implementation. The method of developing of UIs is quite different. Cordys XForms is part of Cordys Studio. XForms can generate UI's based on Web services with their WSDL and XML Schema Definitions (XSD). Transforming UIs is not recommendable, due to the complexity of synchronizing.

Web services

ARIS software service components depict a service operation. This is on the same granularity level as web services in Cordys. ARIS relates the web service to capabilities defined on business level. This is not supported by Cordys. Cordys allows for the development of web services. These services are attached to functions/tasks. Web services can be exchanged between ARIS and Cordys via WSDL import/export.

Data flow

The format of the input and output is specified on different abstraction levels. ARIS allows specification of logical data clusters. These are linked to technical equivalents via other models. The format however does not match, as Cordys requires Xpath [W3C 1993] expressions. Xpath is the XML Path from which the data to be appearing in the form control can be selected. The ARIS technical equivalents of data clusters can indicate the input and output needed on the right abstraction level. Conversion of the technical data into the correct Xpath expression is required for execution in Cordys.

7.1.5 Design considerations and limitations

Assessing the non-element and analyzing several example transformations have led to design considerations and transformation imitations.

Design considerations for non-main elements:

1. Organizational unit or position in EPC can be lanes+roles/teams. The lanes do not have a meaning during execution. Attaching them to functions in C-BPMN is preferred, but currently restricted in Cordys, when no UI is attached to the function.
2. Link only one organization unit/position to a function in EPC. Repetitive modelling of the function in case of multiple attached roles. C-BPMN cannot attach multiple roles to one task.
3. Objectives and KPI are important indicators during monitoring of processes. If they are defined in EPC they should be mapped onto a C-BPMN model. However, no notation exists in C-BPMN for KPIs.
4. Clusters represent data input and output in EPC. In C-BPMN input and output is specified on a different abstraction level, C-BPMN specifies this as message maps.

5. Business services and capabilities are not considered in C-BPMN. They are conceptual services and usually represented by software service types, which are on the same abstraction level as service tasks. When no web services are linked to functions, business services and capabilities can be considered to depict the expected functionality.
6. Loop task in C-BPMN can be mapped to a specific EPC structure with operators.
7. Timers are not present in EPC. However, the ARIS Process Automation module supports timer based Events in EPC. Consideration of introducing this might be relevant, as the timer is an important workflow event in C-BPMN.
8. Process interfaces map to link events in BPMN. This is not supported in C-BPMN. Either avoid using process interfaces or add link events to C-BPMN subset.
9. Risk and Controls in EPC can be linked to functions. Risk and controls are important, as when fired, they impact the process. Risks and controls are converted to annotations. As these risks and controls depict work escalations, these could be translated to escalation event in BPMN. The notation is unsupported in Cordys. Escalation is handled via properties in Cordys.
10. Translate start with input cluster of EPC to message start in C-BPMN, and end event with output cluster to message end event. A start event is triggered by an input, and the end event might specify an output message. This can be represented in C-BPMN.

Transformation limitations:

1. Event-based gateways cannot be mapped to any EPC structure. The events in C-BPMN are happening, while EPC events are states. Exception for start and end events.
2. Process interfaces, Resources, technical terms, application system types are not to be mapped to a specific C-BPMN concept.
3. Multiple instance tasks cannot be mapped from C-BPMN to EPC.
4. Throwing and Catching Events of C-BPMN cannot be directly mapped.

7.1.6 Mapping EPC to C-BPMN

The mapping between EPC and C-BPMN can be achieved using meta-level and structural design patterns mappings. The meta-level mapping describes uni- and bidirectional mapping in Table 8. The structural design patterns mapping is summarized in Table 10 and Table 11. The structural design pattern mapping is specified from EPC point of view, as EPC has a smaller set of workflow patterns support. Thus if the design pattern is supported by C-BPMN, and not by EPC, no mapping exist. Furthermore, C-BPMN has a large design freedom, allowing explicit and implicit representations for one concept. The design considerations are taken into account and resulted in non-main mapping rules. An example process of service oriented EPC to C- BPMN mapping is found in Appendix G.2, and a part is depicted in Figure 26.

EPC to C-BPMN

Non-main ECP to C-BPMN mapping rules:

1. A software service type belonging to a function in EPC can be represented by a Service Task.
2. For XOR or OR split followed by events, the events are translated to conditions for the C-BPMN gateway branch, see patterns in Appendix G.1.1.
3. Organizational unit/ position will be mapped as lane + team/role.
4. Business services, capabilities can text annotations, if and only if software service types are not connected to the function. Annotations are 'bs:' business service, 'cap:' capability. The annotations are followed by the description in the relevant construct.

5. Clusters, KPI, risks, controls and objectives can be text annotations. Clusters are annotated with 'in:' and 'out:' for input and output. KPI with 'kpi:'. Risk with 'ris:'. Control with 'con:'. And objectives with 'obj:'. The annotations are followed by the description in the relevant construct.

Each model should be decomposed into components as specified by design patterns.

1. Each start and end event in EPC is mapped to start message and end message event in C-BPMN, specified with input and output in property.
2. Each sequence pattern is preserved.
3. Each (combined) decision pattern structure is preserved.
 - a. For each nested structure:
 - i. Map according to decision patterns
 - ii. Subsequently, mapping of every construct according to Meta-level mapping till join.
4. Mapping of constructs and applying exceptional transformation rules.

C-BPMN to EPC

Non-main C-BPMN to EPC mapping rules:

1. A task will be mapped to event-function for semantic correctness, except for:
 - a. the function succeeding a start event
 - b. the function succeeding a parallel, exclusive or inclusive gateway split, where the conditions will be turned into event triggers
2. Service Task will be mapped to software service types related to function.
3. Team will be organizational unit, Role will be position.

Each model should be decomposed into components as specified by design patterns.

1. Start and End event transformations.
2. Each sequence pattern is preserved.
 - a. A task will be translated into an event-function, where the event is the trigger for the function, if and only if a task does not subsume a start event.
3. Each (combination) decision pattern structure is preserved.
 - a. Validation of type of gateways (as not all supported in EPC)
 - b. Map according to decision pattern
4. Mapping of constructs and applying exceptional transformation rules.

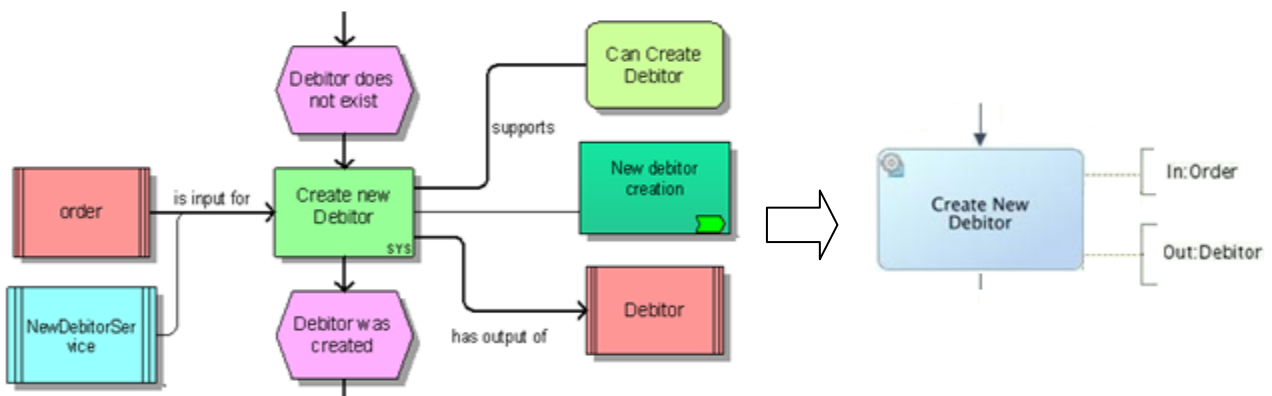


Figure 26 Part of translation of EPC to C-BPMN (Appendix G.2)

7.2 Exchange formats

Business process models have to be exchanged between ARIS and Cordys. This is enabled by exchange formats. Exchange formats are XML based formats. Several standards have been developed to transform, validate and interchange XML based formats. XSTL (EXTensible Stylesheet Language Transformations) are used for xml based transformations. XSD (XML Schema Definitions) are used to validate the modelling languages. XMI (XML Metadata Interchange) is the metadata exchange format specified by the OMG.

EPC Markup Language (EPML) is the exchange language for EPC. The EPML can be translated via XSTL into other XML-based formats. This is amongst others used in the approach to translate EPC to BPMN via EPML [Vanderhaeghen, et al. 2005]. Exchange via EPML is not necessary, as Cordys does not support EPC. The transformations are however applicable for transformation EPC to BPMN, which should be facilitated within ARIS.

There are various (expert) discussions on the exchange formats for BPMN [Silver 2009] [Kraft 2009]. Currently, BPMN 2.0 is not approved yet. In the new version an interchange format is proposed. XPDL is de-facto standard for BPMN 1.x model exchange, due the ease of implementation. Criticism include its redundant specification of each construct, and the lack of merging facilities. A discussion point is that it can be extended for each tool specific. This improves flexibility, but requires conformance of the formats supported by each vendor. XPDL is exchange on model level, where XMI allows for meta-level exchange. XPDL is customized for exchange of BPMN models, and supports diagram exchange across platforms. XPDL also facilitates exchange of executable properties used at runtime. Furthermore, each vendor can develop extensions in order to support their implementation specific details. This information is preserved in order to support round-trip development.

The OMG has proposed the DI as generic diagram interchange model for BPMN [OMG 2009]. The DI model separates the interchange model, domain model and diagram definition. The DI interchange model is a generic model referring to the domain model, and is validated against the diagram definition. This allows for separation of model and view. Both BPMN and DI are specified according to the MOF, and can be exchanged via XMI on meta-level. Expectations are that the vendors will adopt the standard XMI proposed by BPMN 2.0 when it is officially released. BPMN XSD is transformed to XMI via XSTL. No implementations exist, since the format has not officially been approved yet. Criticism for XMI is that it is developed for a more complex problem than exchange of BPMN. Furthermore, the diagram interchange format is not specific to BPMN. Relating the DI to BPMN and validation by XSD is not straightforward.

In previous attempt to provide unidirectional interoperability between ARIS and Cordys, a customized XPDL import/export was facilitated. Graphical conformance after exchanging the process models was identified as one of the major issues. In this pilot project no research was done regarding round-trip modelling.

7.3 Vertical transformation

Vertical transformation can be achieved by pattern based or by semantic BPM. Both approaches require domain ontology. Using semantic BPM is challenging and requires change in development. Considering the ARIS and Cordys case, the pattern based approached is more suitable. It concerns a lower implementation effort and does not require adding semantic structures and information to process models. The pattern based approach requires developing mappings between domain specific concepts in each abstraction layer.

7.3.1 Positioning the pattern based DSML approach

In this approach the task types for the DSML and the patterns for mapping need to be determined. Development of ontologies can contribute to deriving the task types. Additional is identifying patterns which can be mapped. Although matching generic patterns can be captured, knowledge and time are required to create a repository with

these patterns which allows automation of vertical transformation. Organizations already have either business conventions and processes, and legacy systems. Deriving a domain specific language for a specific domain can be achieved in several ways. Top-down by capturing domain concepts derived from ontologies and meta-models of models. In the approaches for modelling transformations a set of business domain task types are proposed. These can be mapped to the domain specific concepts, which should be present in the domain ontology. Bottom-up is achieved by deriving patterns from executable processes. It is argued that for DSML it is advisable to adopt a bottom-up development approach, as it is more aligned with the execution [Gitzel and Merz 2004]. A DSML is either derived from modelling language, or custom made. In [Brahe and Bordbar, A Pattern-Based Approach to Business Process Modeling and Implementation in Web Services 2007] UML profiles are recommended as meta-model. In this case there are two different meta-models. In this approach the meta-models of EPC and BPMN are considered as meta-models for the DSML. Considering the ARIS and Cordys case there are three scenarios, transformations between DSML Business to DSML Functional in ARIS, between DSML Business to DSML Functional in Cordys, and between DSML Functional to DSML Implementation in Cordys.

The comparison of POINT granularity levels and the perspectives defined in the framework (6.1.4) shows that the business perspective consists of multiple granularity levels. The End-to-End processes are decomposed in three lower levels: across departments, across roles, and from one role point of view. The work-level processes are considered on the right granularity level as the Cordys functional models. The pattern based DSML approach is used for transforming between the three perspectives as defined in the framework. It is expected to have cross-role operational processes from business perspective for the DSML business, as depicted in Figure 27. For patterns in decomposing end-to-end processes to cross-roles processes a systematic approach is proposed by [Seruca and Loucopoulos 2003]. Decomposing end-to-end processes is another challenge, and is not addressed in researching how to couple ARIS and Cordys conceptually for round-trip modelling.

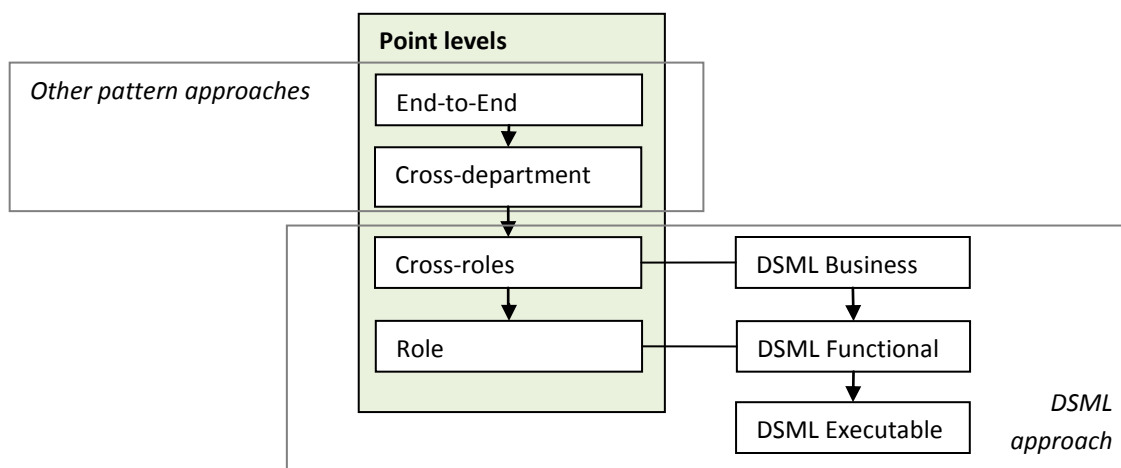


Figure 27 Position of DSML approach in POINT

7.3.2 Domain specific example

The approach for vertical transformation is used with the “Request Bank Account” process, which was developed in ARIS and executed in Cordys. The processes can be found in Appendix I.1. The process labels have been slightly altered to omit client-specific data. The EPC process is on cross-role business level, as it merely contains any functional details. A translation of the EPC model is made to C-BPMN, and the executable process is C-BPMN. The process models are analyzed for parameterized patterns. The transformation approach with DSML is focused on the task types.

As argued before, ontology can contribute in deriving task type. However, with merely one example and minimum background information constructing domain ontology for identifying task types seems excessive. Therefore, the generic task types as introduced in section 5.5 are used to derive task types from the example processes. The processes are analyzed and links are made between the high level functions and executable functions. For the four tasks in the EPC process links have been made with the processes on executable level. Each row in Table 12 presents the tasks on business level, and related tasks on executable level. These tasks are evaluated and task types have been identified, which is presented in Table 13. Note that the transition from business to functional in this case is adding operational information. The tasks remain the same.

No.	Business Perspective Tasks	Executable Perspective Tasks
1	Request bank account recommendation	<ul style="list-style-type: none"> · Update Bank account details · Update Status
2	Fill in transaction details bank account	<ul style="list-style-type: none"> · Update Transaction Details · Update Status
3	Send Bank Account Request	<ul style="list-style-type: none"> · Send Bank Request
4	Sign Bank Account Request (2x)	<ul style="list-style-type: none"> · Approve Request by Approver I · Approve Request by Approver II · Update Request with Results

Table 12 Linking the process tasks for “Request Bank Account”

DSML Type Business	Domain Task
Informative Service	Fill in transaction details bank account, request bank account recommendation
Performative Service	Send Bank Account Request
Approval	Sign Bank Account Request
DSML Type Functional	Domain Task
Informative ServiceTask	Fill in transaction details bank account, request bank account recommendation
Performative ServiceTask	Send Bank Account Request
UserTask	Sign Bank Account Request
DSML Type Executable	Domain Task
Ullnvoke	Approve Request by Approver I, Approve Request by Approver II
ServicePerform	Update Bank account details, Update Transaction Details, Update Request with Results, Send Bank Request, Update Status

Table 13 Derived DSML Task Types for “Request Bank Account”

Analyzing the task types and processes from business to IT perspective, and vice versa, resulted in three patterns: approval, informative service and per formative service. Table 14 presents a summary of the parameterized patterns. These patterns are evaluated for:

1. Transformation between DSML Business to DSML Functional in EPC in Appendix I.2.
2. Transformation between DSML Business to DSML Functional in C-BPMN Appendix I.3.
3. Transformation between DSML Functional to DSML Implementation in C-BPMN Appendix I.4.

Parameterized pattern types	Description	Business To Functional transformation rules	Functional To Operational transformation rules
Approval (Appendix I.5)	Approval by authorized person in Cordys system	Assign authorized role for Approval activity	Assign the corresponding UI to activity
Informative Service (Appendix I.6)	Service activity requiring input from stakeholder	Assign the web service, input and output data, and the role to the informative activity	Assign timeouts, iterations and update status activity
Performative Service (Appendix I.7)	Service activity execution	Assign web service, input and output data to the per formative activity	Assign iterations, and exception handling

Table 14 Identified domain specific parameterized patterns based on “Request Bank Account”

The patterns depict the target structure of a task type and what information is additional when transforming between two different perspectives. The activity “Sign bank account request” is an Approval task type. Transforming the approval function from business to functional is to add an authorization role for approving the bank account. The task type is a UserTask. In EPC the system also needs to be defined. In the functional perspective an approval activity is related to the authorization role. The transformation to IT is to attach the developed User Interface which specifies the approval form. Furthermore, in this pattern the result are updated via a web service. The transformation pattern can be found in Appendix I.5. An example of the approval pattern of the domain specific example is depicted in Figure 28.

An Informative Service pattern is when activity requires input from the user. For example, “requesting bank account recommendation” requires the user to provide details from which the system can determine a recommendation. On functional level the service and user need to be specified. On executable level the service is attached to timeouts, and exception handling details. The transformation pattern can be found in Appendix I.6.

A Performative Service pattern is the execution of a specific task by a service. The processing of the bank account request “Send Bank Account Request” is a service activity. On functional level the web service, including the in and output needs to be specified. On executable level exception handling can be added. In this domain example this is not the case. The transformation pattern can be found in Appendix I.7.

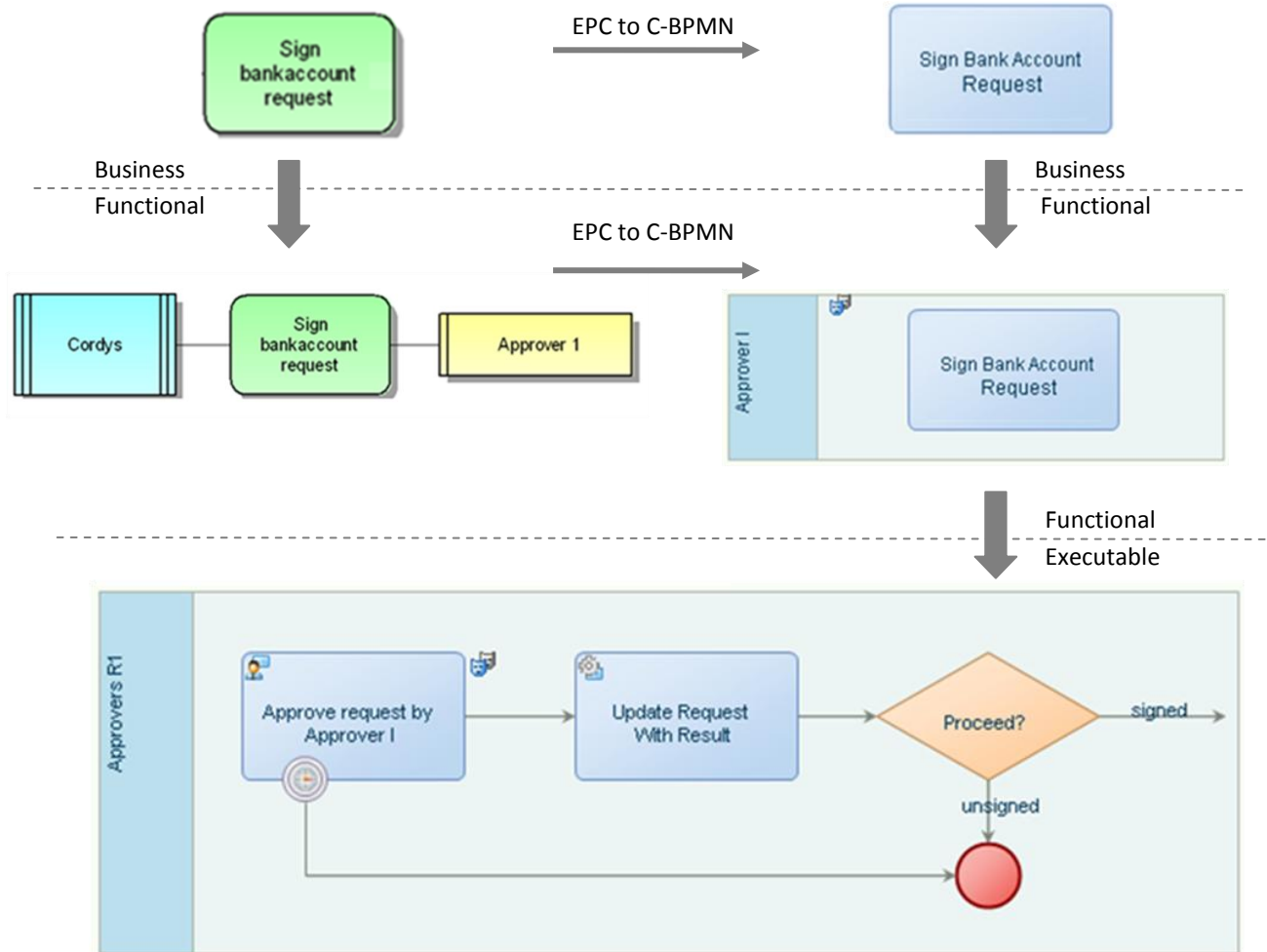


Figure 28 Example approval pattern based transformation

Defining the patterns for task types is a start. But a DSML also specifies operators and other elements, which are currently not included. Data elements are included as they relate to functions. The patterns provide a basis for transforming the functions. Manual effort is still required to finalize the model for execution. The parameterized patterns should be further researched in detail regarding additional information such as KPIs, risks and controls. This method would be best demonstrated with a practical case study. Important to note is that this approach requires great organizational effort, including creating domain ontologies and creation of DSML. In practice this is evaluated as not feasible yet. However, this approach can be used as guidance through the process of design to executable.

7.4 Traceability for ARIS and Cordys

Traceability for BPD SOA process models is defined as tracing the model elements within and across different perspectives (described in 4.5). In BPD SOA context with two tools, an agreement needs to be made on the traceability meta-model. Otherwise pragmatic interoperability will not be achieved. A few design choices have to be considered regarding the information for tracing, views which tracing should depict, and the method of storage.

7.4.1 Traceability Meta-model

The traceability meta-model is a model where both vendors need to agree on. In Figure 29 a meta-model for traces is proposed in UML class diagram. Each trace has a transformation type, and traceable element. The latter refers to the source and target elements of the transformed object. In this case it is a model element (i.e. function). The

transformation type corresponds to the type of transformation that is applied between the source and target. It can be either on one abstraction level (InterView), or between abstraction levels (IntraView). The types of transformations are presented in the scenario dimension of the framework. Based on the meta-model, a trace specific for ARIS and Cordys coupling can be defined. The trace should contain the identifier of the model elements or concerns of ARIS and Cordys. Both ARIS and Cordys object elements have a unique identifier, the GUID. The format for the transformation types can be formalized when the transformation types are actually implemented.

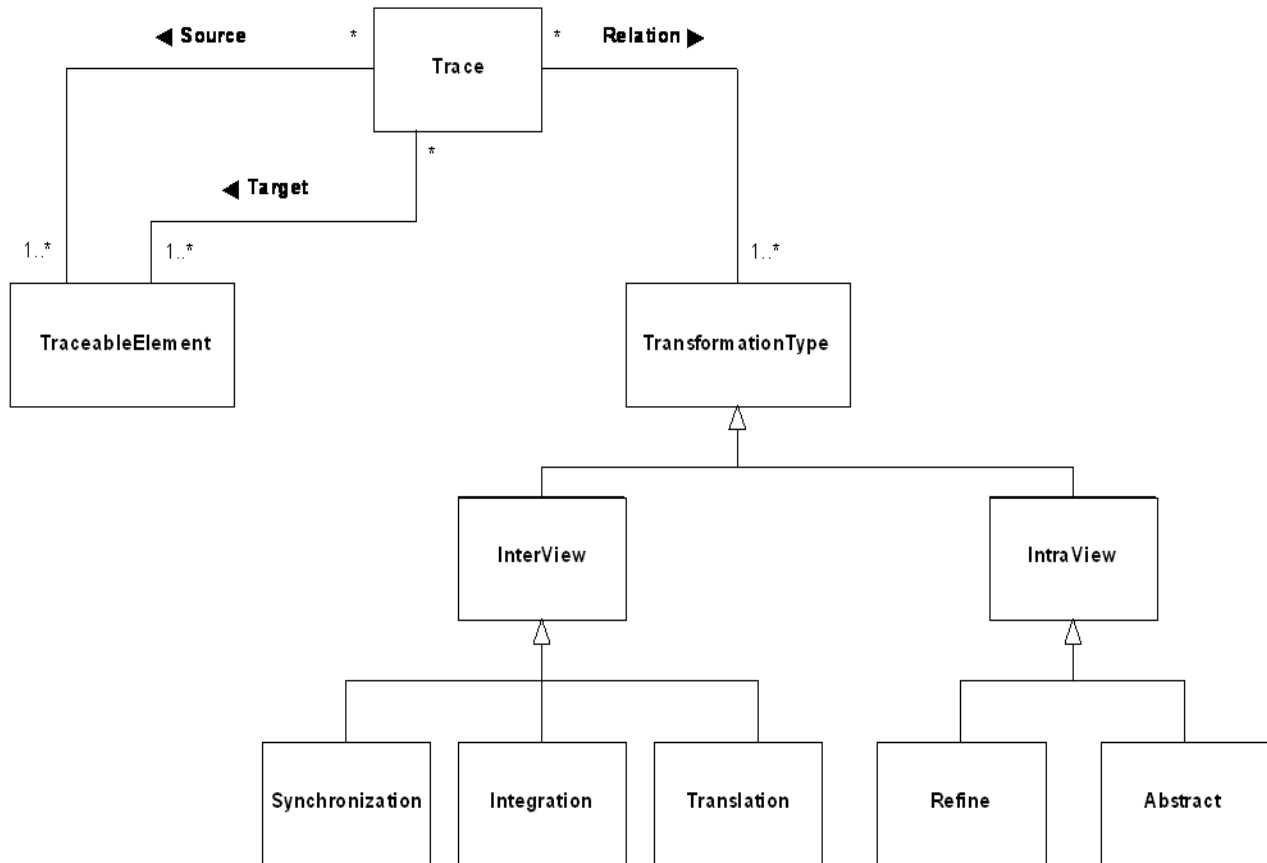


Figure 29 Meta-model Traceability for ARIS and Cordys (concept)

7.4.2 Traceability example

The source and target GUID of model element subject to transformation are traced. The type of transformation depends on what types of transformations are identified. In ARIS and Cordys context, these are horizontal translation, synchronization or integration. For vertical transformation this can be either refinement or abstraction. In round-trip development the traces need to be exchanged across platforms. The trace information needs to be transformed as additional annotation. Traces allow keeping track of the information transformed, and creates an understanding on the information transformed from high-level business and IT. Figure 30 illustrates transformation between an EPC to BPMN in ARIS, and synchronization with Cordys to C-BPMN. For each transformation the trace is illustrated in an annotation form. The labels of the objects represent example unique identifiers (example GUIDS).

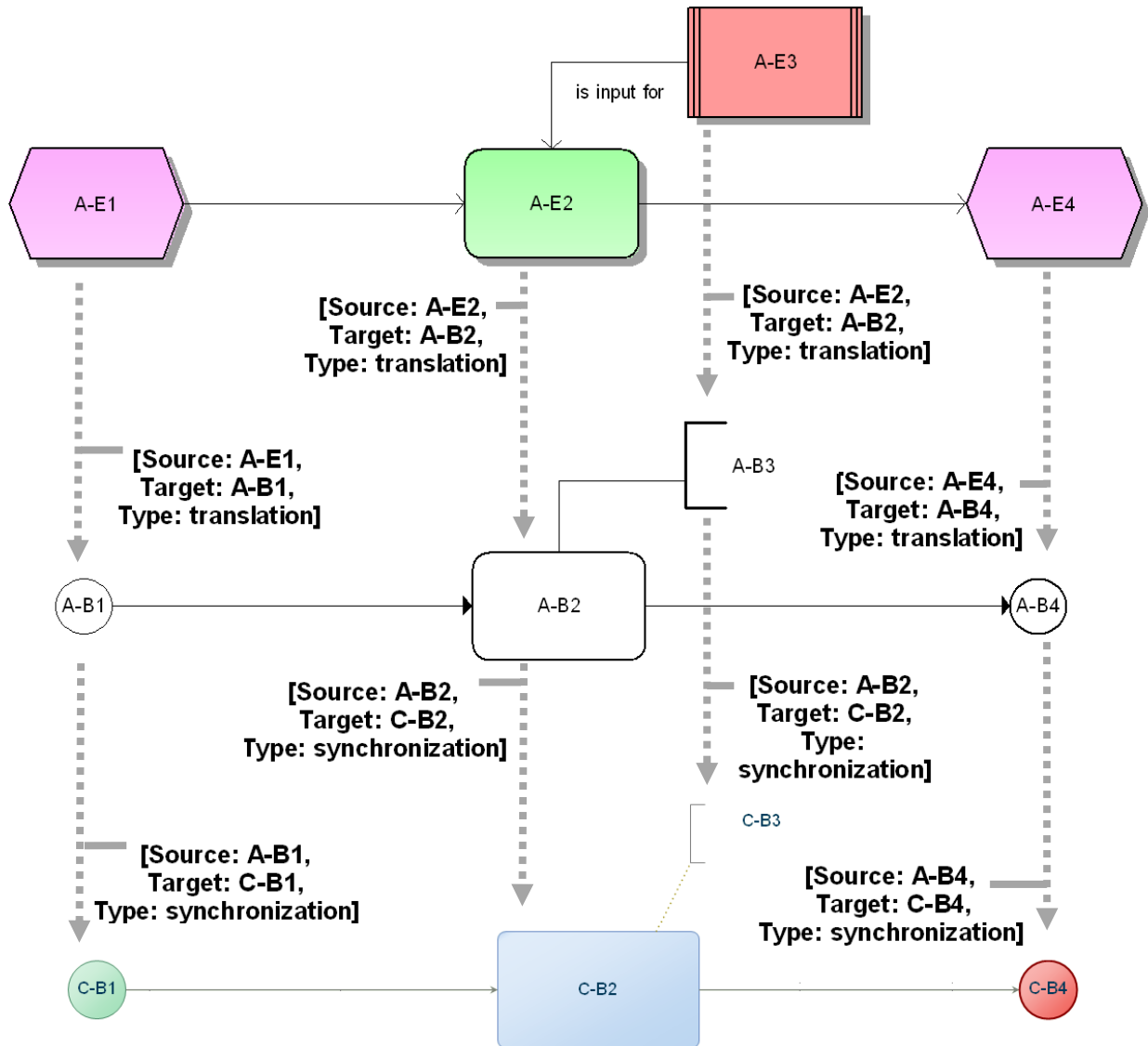


Figure 30 Simple traceability example between EPC to BPMN, to C-BPMN

Based on traces the evolution of one object can be derived. If the external storage method is implemented, different type of models can be created based on the trace information. A useful application would be creating an ontology model based on the traces. For instance, a refinement in vertical transformation can transform a business function into multiple IT functions by parameterized patterns. This depicts the relation between the business and IT functions.

The trace model will be determined by how and which transformations are eventually implemented. It requires both vendors to make an agreement on a traceability model. Traceability allows for pragmatic interoperability, as the trace information can provide the necessary insights on whether they both have the same expectations and understanding of models.

7.5 Validation

The goal is to couple ARIS and Cordys for interoperability. In this thesis a mapping between models is made on conceptual level. A validation of the theory developed during this thesis is not possible in practice due to lack of opportunity for a proof of concept. Currently, no client with both Cordys and ARIS is at the stage of integration. Two alternative ways for validation are presented: via literature and experts.

7.5.1 Validation by literature

The mapping for horizontal transformation is based on meta-level mapping, ontological mapping, and structural design patterns. The mapping of EPC and BPMN main elements can be derived from literature. Most methods have evaluated the languages in empirical research. Ontological mapping is based on evaluation of EPC by [Green and Rosemann 1999], and BPMN by [Recker, Wohed and Rosemann 2006]. Evaluation by [List and Korherr 2006] and [Murzek (1) and Kramler 2007] of meta-model mappings have also contributed to the mapping. Furthermore, the mappings are based on the well known work-flow patterns by [van der Aalst, et al. 2003]. The workflow patterns have been extensively researched and referred to in scientific publications.

The approach for vertical transformation is based on domain specific modelling languages and parameterized patterns. Methods with DSML and patterns are often proposed for vertical transformation in literature as depicted by [Stein, Kühne and Ivanov 2009]. The specific approach described in 5.5 was demonstrated in a case study in the financial industry [Brahe 2007]. Conclusions were that the approach did eliminate repetitive manual implementation activities, but adequate tool support was lacking at the moment. The vertical transformation approach is conceptually applied for ARIS EPC and Cordys BPMN.

7.5.2 Validation by experts

The example mappings, structures, and example cases are checked and validated by experts. Experts of IDS Scheer and Cordys have given their feedback and used their expert knowledge to validate the mappings and examples. The example processes are found in Appendix G and Appendix I.

In future work the mappings need to be validated with practical case study, preferably in an adequate quantity to find exceptional cases and validate the transformations.

Chapter 8

Interoperability between ARIS and Cordys

In the previous chapters it was conceptually defined what types of coupling exist for ARIS and Cordys. The model transformations for EPC and C-BPMN depict how and what information can be synchronized, and the traceability concept enables round-trip development. In this chapter recommendations are given on how to achieve iterative round-trip development between ARIS and Cordys.

Section 8.1 depicts what currently the ARIS and Cordys tool support regarding model exchange and transformations. Section 8.2 provides insight in which modelling activities should be done in which tool. Best practices can be defined for transforming EPC to Cordys BPMN in 8.3. Best practices are *“Methods and techniques that have consistently shown results superior than those achieved with other means, and which are used as benchmarks to strive for. There is, however, no practice that is best for everyone or in every situation, and no best practice remains best for very long as people keep on finding better ways of doing things [BusinessDictionary.com 2009]”*. Section 8.3.1 presents conventions for functional modelling in ARIS. Conventions prescribe the granularity level of data elements which are important for Cordys. In 8.4 the design considerations for ARIS and Cordys are summarized. There are several design issues which need to be resolved in order to achieve interoperability. The technical implementation challenges are summarized in 8.4.1.

8.1 Current tool support in model exchange

ARIS supports collaboration with several BPMS tools. This includes collaborations with Oracle, SAP, and Microsoft Biztalk. Different methods for integration have already been implemented. Best practice integration is currently with Oracle. Oracle has purchased and integrated ARIS within their own tool, the Oracle BPA Suite. Shared repository is the best alternative. With the other tools exchange of BPEL processes is supported, but there is no support for pragmatic interoperability. Cordys currently has no collaboration with a tool such as ARIS.

The technical challenge is how to ensure the interoperability both on syntactic and semantic level for BPMN. Vendors have started to adopt XPD as interchange format for BPMN models. Currently the development of XPD has matured that it both can capture BPMN syntax and semantics. Challenges remain for the vendors both supporting the same version of XPD, implementing semantic exchange and interpretation. In addition to interoperability there are synchronization and traceability challenges for ensuring round-trip development. Currently, ARIS and Cordys do not support the same (versions of) exchange formats.

Currently, the Process Automation module within the ARIS Design Platform supports an EPC to BPMN transformation. The transformation functionality is only supported in the Process Governance tool. Not all standard EPC constructs are supported. Furthermore, in version 7.1 no semantic and generation patterns are supported. Thus a subset of EPC can be syntactically transformed to a BPMN process model. Appendix H summarizes the horizontal transformation patterns supported and evaluates whether it is suitable for ARIS EPC to Cordys BPMN transformation.

8.2 Design-time versus run-time

ARIS is best for design-time activities. It is wishful to create a functional design in ARIS. In the BPD SOA context the development of services, from business services with capabilities to technical services, creates better alignment of IT services with business requirements. From there information is exchanged based on what can be exchanged, and what is required on a technical level. Recommendations are given in 8.3.1. Cordys is a strong execution platform with several design-time facilities. It allows for revisioning of models in development. ARIS is more accessible for

from business perspective and has functionalities which ease the management of many different models for analytical purposes. Cordys lacks in providing functionality for managing the analytical part of BPM. Three arguments for managing design-time in ARIS are 1) the wide variety of business models for business analysis, 2) enabling linking of models on different granularity levels enhances management and development from different perspectives, and 3) versioning of models is a development and management necessity from business level. Cordys currently does not support linking models, and versioning. Figure 31 illustrates the ARIS and Cordys modelling situation.

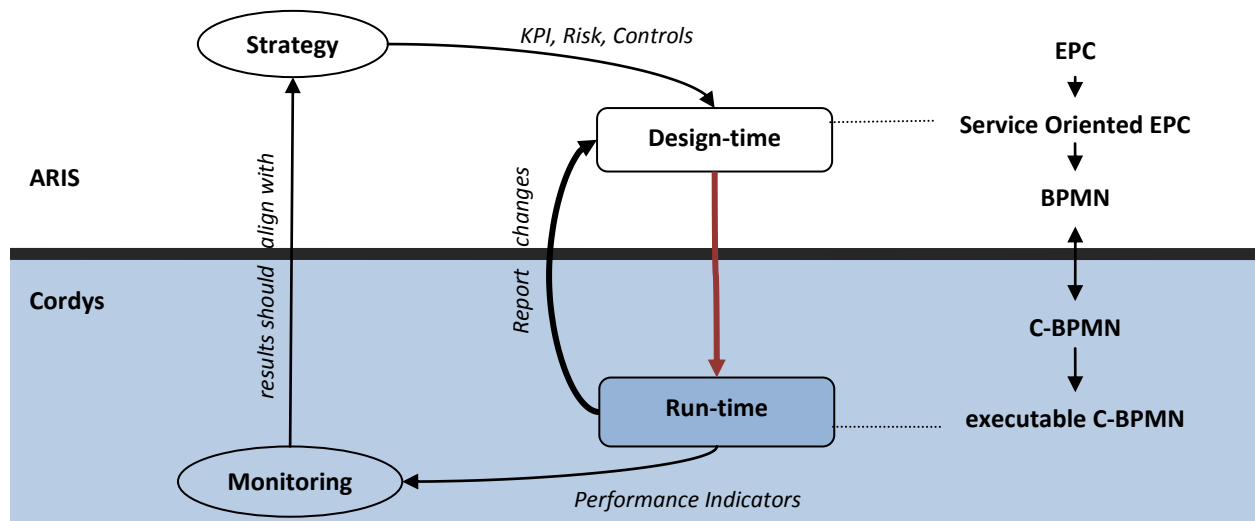


Figure 31 ARIS design-time and Cordys run-time

8.3 Step to step ARIS EPC to Cordys BPMN

The framework has shown what is required to facilitate round-trip development in ARIS and Cordys for BPD SOA. In summary best practices to transform high levels EPC to Cordys BPMN are:

1. Identify the modelling conventions and determine the set of information (set of constructs) which are relevant.
2. Develop domain ontology, task ontology to create a consensus on the vocabulary and terminology used.
3. Develop a set of patterns: structural design patterns and parameterized patterns.
4. Define traceable information, and decide on storage method for the traces. Traceability of the transformation allows for round-trip development.
5. For horizontal transformation: apply meta-level mapping, structural design pattern mapping, and ontological mapping. For ARIS EPC and Cordys BPMN this is described.
6. For vertical transformation: Use domain and task ontologies to develop DSMLs for each viewpoint in modelling. Use parameterized patterns as guidelines for transformation.

8.3.1 Guidelines SO-EPC to C-BPMN

Taken the mapping as defined in 7.1.6 into account, the following guidelines for Service Oriented EPC can be used to ease the transformation to Executable Cordys BPMN (C-BPMN):

- Define the process steps of the EPC on operational level – as *work-level processes*. Work-level processes are usually the lowest level processes considered from a role point of view (i.e. work instructions).
 - ▶ Avoid very long activity chains
 - ▶ Reusable process chains

- ▶ Avoid dependencies on the status of other processes
 - ▶ No process interface to other processes (link events are not supported in C-BPMN)
 - ▶ Reusable business processes and business functions
 - ▶ Provide input and output for start and end events
 - ▶ Ensure execution is business-driven (e.g., exception handling)
 - ▶ Avoid multiple subsequent operators to depict complex structures
- *Define a start event which have incoming data as triggers linked with (input) cluster, and end event with output data trigger linked with (output) cluster.* Start and end events with clusters are converted to start and end message events in C-BPMN. By definition the events are triggers in EPC. The (empty) none start event in BPMN does not have a trigger defined. Therefore, if there is an explicit message triggering a process, or triggered when the process ends, defining them with input and output clusters should make the trigger information available in a C-BPMN process.
 - *For conditional branching, keep in mind that events following operators specify conditions for execution.* Cordys requires XPath type conditions. Specifying events names as logical expressions, such as “x-y < 0”, contributes specifying conditions for execution. Cordys XPath Editor allows for composing the conditions.
 - *Link software service systems types to functions.* These correspond with the web services defined in service Tasks in C-BPMN.
 - ▶ When no services are defined, capabilities and business services depict what kind of functionality is expected of a process function. Capabilities/ Business services can be converted as annotations. (Note requirements engineering is linked to capabilities)
 - *Define and link the authorization roles as positions to a function with relation “decides on”, “Must inform about result of”, “Must be informed about”, or “Must be informed on cancellation”.* In Cordys, organizational users and roles are means to regulate authorization and eventually provide access by UI elements (menu's and toolbars). Authorization roles attached to functions can be translated to a user task in C-BPMN. User tasks are workflow assignments in Cordys, and require a UI and role/organizational unit.
 - ▶ In EPC multiple roles can be attached to one specific task. In C-BPMN multiple roles cannot be assigned to one task. Duplicating the function with another role is easier to transform.
 - ▶ Other relations between organizational elements and functions are manual and non Cordys executable specific, including “Carries out”, “Is technically responsible”, “Is IT responsible”, “Contributes to”, “Has consulting role in”, and “Accepts”. These can be represented by lanes in C-BPMN.
 - *Develop functional and data glossary.* Data flow is represented by clusters in a service oriented EPC. Clusters represent input and output on a business object level. ARIS SOA allows for development of a functional and data glossary. The clusters on business level can be linked to the technical data types, by a ‘depicts’ relation, in a class diagram. The technical data types are on the right granularity level for C-BPMN.
 - *Define KPIs and the measurements and link these to functions.* Although C-BPMN cannot represent KPI on process level, these should be converted into annotations because of their relevancy in monitoring. The business measures for monitoring are specified by XPath expressions in C-BPMN .

- *Define risks and controls and link these to functions.* Although C-BPMN cannot represent Risks and Controls on process level, these might be relevant during execution of processes. A risk and control should be converted to annotation. In Cordys escalation properties for a function can be set.

8.4 Design considerations for IDS Scheer and Cordys

The state and development of BPD SOA project within organizations influence the type of interoperability and transformations which are required. At first horizontal transformation should be enabled, with focus on synchronization of syntax and semantics of models between two platforms. Another aspect which will enable round-trip development is traceability. Tracing all transformations allows maintainability of the transformations. Enabling vertical transformation requires collaboration and expert knowledge. Currently, processes within organizations are dynamic and lack of standardized conventions in the company make automating the vertical transformation a challenge. It requires collaborations between Business and IT experts, as well as collaboration of all involved departments. In a previous attempt to couple ARIS and Cordys it was decided that ARIS is leading, which means the process steps determined in ARIS cannot be edited in Cordys. Furthermore, challenges remained around version control at Cordys. Enabling round-trip development both vendors need to agree on:

- A traceability model
- Conforming exchange format

8.4.1 Functional tool considerations

A few tool-based solutions need to be developed in order to enhance round-trip development between ARIS and Cordys.

Implement traceability

Determine what information should be traced based on the traceability model. Traceability is required for round-trip development, to keep track of information and its original source. Traceability information can also contribute to impact analysis, and creating domain specific information.

EPC to BPMN transformation in ARIS (and vice versa)

ARIS should facilitate EPC to BPMN transformation. Currently, it is only available in the Process Governance module. BPMN is receiving more support in practice, and is supported by most BPMS tools. For round-trip, a transformation from BPMN to EPC needs to be considered.

Support for developing ontologies

Several models in ARIS are designed to depict ontological relations (i.e. data class diagram). However, developing ontologies and maintaining them requires time and effort of the stakeholders. But adequate tool support may increase the ease of maintaining ontologies, and therefore lower the threshold to start development of domain ontologies.

Implementing pattern based transformations for vertical transformations

Development of a pattern repository for capturing recurring business functions and implementations will enhance vertical transformations. Research has been done on how to create such a pattern repository by [Seruca and Loucopoulos 2003]. This is not further researched in this thesis. For both ARIS and Cordys enablement of vertical transformation via pattern based solutions can be useful if domain knowledge is captured.

XPDL and/or XMI exchange

Currently, no implementations of the proposed standard exchange format exist for BPMN. XPDL has been de-facto standard for BPMN (1.x) exchange. The OMG has proposed the DI as generic diagram interchange model for BPMN. Both BPMN and DI are specified according to the MOF, and can be exchanged via XMI on meta-level. Expectations are that the vendors will adopt the standard XMI proposed by BPMN 2.0 when it is officially released. Industry experts [Silver 2009] [Kraft 2009] evaluated the proposed exchange formats defined in BPMN 2.0, but are not yet convinced of its functionality (see 7.2). Time will show whether the proposed exchange format is sufficient to deal with BPMN exchange, including the vendor specific alternations.

Shared Repository

Best practice of ARIS product with another BPMS is to integrate, “OEM”, the ARIS platform. However, this approach requires much investment and development. The options for technically realizing coupling ARIS and Cordys are considered as future work.

Part IV.

Conclusions & Recommendations

Chapter 9

Conclusions

Organizations deploy business process driven (BPD) SOA in order to manage their processes and improve agility. The core of BPD SOA is business processes. They are developed from different viewpoints within an organization capturing how the organization should function. Analyzing, designing and executing business processes are supported by tools. These are a Business Process Analysis (BPA) tool for analyzing and documenting processes, and a Business Process Management Suite (BPMS) for executing and monitoring processes. Although some activities in tools overlap, there is a clear difference in the granularity of how data and processes are defined. Functional design in BPA tool is different from that in BPMS. Currently, no tool environment exists which combines the strengths of BPA and BPMS. Coupling these two tools should provide a collaborative and coherent tool environment for BPD SOA modelling and reduce redundant modelling activities. Therefore, the research goal is to **determine to what extent round-trip modelling is possible by coupling a BPA and BPMS tool, respectively ARIS and Cordys**. This thesis has addressed the transition from high-level business processes to executable processes, and how to facilitate iterative round-trip modelling with these two tools. Sub-questions are what BPD SOA with tool support is in 9.1, how to facilitate round-trip between ARIS and Cordys process models in 9.2, and what is expected of ARIS and Cordys as tools when ensuring round-trip development in 9.3.

9.1 Design-time in ARIS and Run-time in Cordys

ARIS is more appropriate for design-time of modelling, by offering documenting and analyzing facilities. Designing operational models in ARIS provides better alignment with business. Functional modelling in ARIS can be done according to guidelines for functional modelling in Cordys, defined by mapping a service oriented EPC to a Cordys BPMN, to ease transition. These models can be exchanged to Cordys, and the run-time models can be implemented. There are three arguments to manage design-time in ARIS and run-time in Cordys:

- There is a wide variety of models supporting business analysis in ARIS as BPA tool. Different models provide more insight in strategic objectives and their relation with business processes. Developing functional designs in ARIS will create tighter alignment with strategic objectives. Cordys does provide support for developing Key Performance Indicator (KPI) models, and value chain modelling for business analysis, but their support is limited compared to the BPA functionalities in ARIS.
- ARIS facilitates modelling on different granularity levels and from different perspectives, and links these different levels and views via “assignments”. An object in a process can refer to a more detailed model depicting relations, depicting different views on one particular activity. In Cordys linking models is not supported. From business perspective, ARIS is more adequate for managing the business processes.
- ARIS allows for versioning of models, whereas in Cordys this is not supported. Versioning from a business perspective is an essential must-have. Cordys however, does support revisioning, which is more a workflow management functionality.

In conclusion ARIS is more accessible for business people and has functionalities which ease the management of many different models for analytical purposes. Cordys lacks in providing functionality for managing the analytical part of BPM. Cordys provides functionality on an operational level with a development and deployment environment. Technical skill is required to use Cordys optimally.

9.1.1 Exchange points

The exchange points of processes depend on the purpose of a project. From business perspective, it is an advantage to create functional processes in ARIS, as they can be aligned better with strategic objectives. But when there is no need for of maintaining the alignment of business and functional design, in projects with evident operational purposes, exchange of process models without any operational detail is advised.

9.2 Round-trip between ARIS EPC and Cordys BPMN process models

A conceptual framework for iterative round-trip Business Process Driven SOA modelling has been designed to guide business to IT transition between a BPA and BPMS tool. The framework and approaches have been used to conceptually couple ARIS and Cordys. It focuses on different levels of interoperability, defines different types of model transformations, and the set of information for transformation in round-trip modelling. Analyzing ARIS and Cordys depicted possible coupling points and which modelling transformations were applicable.

9.2.1 Horizontal transformation: EPC to BPMN

ARIS provides more analytical facilities for business processes than Cordys. Creating a service oriented EPC in ARIS is considered as beneficial as alignment with business requirements is facilitated. If this is not desired, exchanging simple processes is sufficient. Horizontal transformations between service oriented EPC and Cordys BPMN has led to the following conclusions:

- BPMN lacks representation for the concepts of KPIs, objectives and application system types.
- EPC lacks representation of several types of tasks, type of events, type of gateways and associations.
- BPMN is much richer in depicting variety of types of constructs, however, lacks representation for business context related constructs such as higher level services, and objectives.
- A service oriented EPC depicts more business context than BPMN. However, it lacks representation for different variations of the constructs.

The transformations from functional EPC to Cordys BPMN are conceptually defined. All relevant constructs and patterns are defined, and design considerations summarized. In general these references can be used to transform EPC to BPMN manually. ARIS requires an EPC to BPMN transformation. There are however some mapping considerations which should be taken into account.

9.2.2 Vertical transformation: from high level to executable processes

The pattern based domain specific modelling approach for vertical transformation provides a basis for transforming the tasks from business to executable processes. A challenge is finding generic patterns, as most organizations require domain specific implementations due to their legacy systems. Transformation is enabled by defining related Domain Specific Modelling Languages (DSMLs) for each perspective and parameterized patterns. This approach currently only facilitates transformation of task types of DSML. For other elements it is not yet researched how they transform. For example, how operators and conditions can be transformed into specific business rules. Thus, manual effort is still required to finalize the model for execution. However, a pattern based transformation would reduce redundant work as recurrent transformations of functions are proposed. This pattern based approach should be further investigated in a proof of concept. In the future semantic Business Process Management, using Semantic Web techniques, will provide better automated transformation between business and IT processes, however this requires a strong ontological basis in organizations.

9.3 Ensuring iterative modelling between ARIS and Cordys

A conceptual framework is used to structurally define what degrees of interoperability can be achieved when coupling two tools. Technical interoperability is not a challenge, because both ARIS and Cordys have the technologies to support transformation. Syntactical and semantic interoperability of models are subjects for model

transformations. Pragmatic interoperability can be achieved by traceability. Traceability enables round-trip development by keeping trace of model transformations. Tracing will allow round-trip development as it keeps track of how the information evolves, and provides the trackback in change-time. Furthermore, versioning and merging of process models and its elements are necessary to maintain the interoperability. Organizational and cultural interoperability needs to be created in projects adopting BPD SOA with ARIS and Cordys.

As indicated by various experts, challenges in BPD SOA are a mindset change, service granularity, terminology and design for scalability. The mindset change is an organizational issue. The importance is recognized in this thesis, as analyzing of the tools provides insight on the clear similarities and differences between a tool, BPA, for the business domain, and a tool, BPMS, for the IT domain. This can be used to show the experts of the domain how they can complement each other. Domain ontologies contribute in creating a shared conceptualization. A consensus on vocabulary and terminology with different stakeholders is what enables the transformations. The framework addresses the levels of granularities by defining these throughout the transition from high-level business to execution. And last, without design for scalability, interoperability cannot be achieved. Design for scalability is design for iterative development. Tracing information, versioning and merging are therefore essential.

9.4 Concluding remarks

Iterative round-trip BPD SOA modelling with a BPA and BPMS tool is a combination of tool-based solutions and process governance. The vendors need close collaboration to facilitate the exchange and trace the exchanged information for round-trip modelling. Currently, tools do not offer conforming exchange formats and pragmatic interoperability. Round-trip modelling also requires organizational effort. An organization which adopts BPD SOA requires process governance to define guidelines for round-trip modelling. Guidelines regarding the communication, approval and permissions for stakeholders in process modelling need be developed, and should endorse the development of domain ontologies.

9.5 Contributions

This research will serve as a theoretical base for future work in realizing the coupling of ARIS and Cordys, or any BPA and BPMS tool. The main contributions of this thesis are:

- Analysis of ARIS and Cordys as tools and comparison of the methods of process modelling. This has provided insight into the capabilities of the tools.
- A conceptual framework providing a structural approach to guide business to IT transformation from a modelling perspective between a BPA tool and BPMS.
- Combination of methods from state-of-art literature on how horizontal and vertical transformations of business process models can be achieved.
- Mapping of ARIS service oriented EPC to Cordys BPMN on meta-level, ontologically and for structural design patterns.
- Illustrated a domain specific pattern based approach for horizontal transformation for ARIS-Cordys specific process models.
- Traceability as concept for round-trip modelling between ARIS and Cordys.
- Assessment on how to map certain contextual elements/models (UIs, web services, data flow, KPIs, and Risks and Controls) between ARIS and Cordys.
- Recommendations on how to facilitate round-trip modelling between ARIS and Cordys functionally. The findings have shown what challenges remain for ARIS and Cordys.

Chapter 10

Recommendations

Recommendations for ARIS (version 7.1) and Cordys (version BOP-4) for achieving iterative round-trip development:

1. Creation and management of design-time process models should reside in ARIS, and run-time process modelling and development in Cordys. ARIS is adequate for design-time due to the analytical and management functionalities. Cordys is suitable for workflow projects and execution.
2. Functional modelling in ARIS should be done according to guidelines for functional modelling in Cordys to ease transition from operational to executable processes.
3. ARIS should enable transformations between EPC and BPMN models. BPMN is gaining popularity in the industry as visual modelling language. Enabling transformation between EPC and BPMN would increase their interoperability with other platforms from modelling perspective.
4. Round-trip development requires pragmatic interoperability, achieved by traceability across two tools. Agreement should be made on a traceability model for round-trip development. Developing external traceability models are advised as they can depict domain specific information and relations, which contribute to creating domain ontology.
5. Cordys should facilitate merging and versioning of process models. Merging and versioning are essential in iterative round-trip development. ARIS already supports merging and versioning.
6. Conformance should be made on exchange formats between ARIS and Cordys. Different opinions on the usability of the exchange formats for BPMN exist. An exchange format should enable merging, and deal with tool specific modelling variations.
7. Exchange of web services, KPIs, data flows, roles, and Risk & Controls in addition to process logic should be enabled. However, the reasons for exchange of User Interfaces (from ARIS) after first assessment are not evident.
8. Development of pattern templates and recognition for pattern based approaches for both horizontal and vertical transformations.
9. Best practice with ARIS and other BPMS is to integrate the ARIS platform, as done with Oracle. There are however different options regarding alignment of repositories. However, this was not researched in detail. Depending on the investments vendors are willing to make, best practice for round-trip development is integration of the ARIS platform.

10.1 Limitations & Future Work

This thesis has researched how to conceptually couple two tools for iterative round-trip Business Process Driven SOA modelling. All important aspects and dimensions are noted, and a framework is proposed to structurally tackle the coupling. The theory was illustrated by example cases. Ideally, the theory was validated in a proof of concept at a company with both tools. During the 8 months of this thesis project the opportunity for a proof of concept did not occur. Therefore, the validation of theory and examples cases relies on literature and expert knowledge.

This work has provided a theoretical base for coupling. The findings provide the following insights for future work:

- Organizational guidelines or best practices in relation to process governance with a BPA and BPMS tool. Currently, there has been one pilot project where ARIS models have been transformed to Cordys executable models. Best practices should be based on successful practice case, or quantitative evaluation of example cases in practice.

- Expand the mapping of ARIS and Cordys models with contextual factors. The scope of this thesis has taken a set of process related information within BPD SOA context for transformation. This set of information is currently based on the wishes of experts which have experience in developing process models on high level and low level. If the theory is tested in a proof of concept it may show that mapping of additional elements might be relevant.
- Conformance of exchange formats between ARIS and Cordys. Currently, there is discussion on a proper exchange format for BPMN, either the proposed (open standard) interchange format by OMG or XPD. Creating a coherent and consistent exchange format for ARIS and Cordys process models was not within scope of this thesis. There is also a need for exchange of non-standardized transformations. Exchanging non-process model related information, such as UIs, risks and business requirements, would require customized exchange formats.
- Creating a technical design related to repositories for ARIS and Cordys. There are several options, as different implementations have already been made with other vendors. Best practice is the Oracle BPA Suite. Oracle has integrated the ARIS Design Platform. Another option is to share a service repository. Exchange of process models with traceability is easier when dealing with one repository, only in practice investment and customer demand determine to what extent developing a shared repository is possible. The options are best illustrated in an architecture.
- Research on how to align the monitoring with optimization and redesign in ARIS. Options are to exchange monitoring data after run-time, or connect ARIS process performance monitoring to the run-time engine.

Bibliography

- [Aguilar et al., 2006]Aguilar, E. R., F. Ruiz, F. Garc'ia, and M. Piattini. 2006. Applying software metrics to evaluate business process models. *CLEI Electron. J.* 9(1).
- [Aßmann, Zschaler and Wagner, 2006]Aßmann, U., S. Zschaler, and G. Wagner. 2006. Ontologies, meta-models, and the model-driven paradigm. *Ontologies for Software Engineering and Software Technology.* 249–273.
- [Barros, Dumas and ter Hofstede, 2005]Barros, A., M. Dumas, and A. H. M. ter Hofstede. 2005. Service interaction patterns. *Business Process Management.* 302–318.
- [Becker, Pfeiffer and Räckers, 2007]Becker, J., D. Pfeiffer, and M. Räckers. 2007. Domain specific process modelling in public administrations: the picture-approach. In *Electronic Government*, 68-79. Springer Berlin / Heidelberg.
- [Bhat et al., 2007]Bhat, J., K. Pooloth, M. Moorthy, R. Sindhgatta, and S. Thonse. 2007. Use of ontology for automating knowledge intensive business processes. In *Ontologies*, 435-459. Springerlink.
- [Bieberstein et al., 2005]Bieberstein, N., S. Bose, M. Fiammante, K. Jones, and R. Shah. 2005. *Service-oriented architecture (soa) compass: Business value, planning, and enterprise roadmap.* IBM Press.
- [Blechar, 2008] Blechar, M. 2008. Magic Quadrant for Business Process Analysis Tools. Gartner Research.
- [Bögl et al., 2009]Bögl, A., M. Schrefl, G. Pomberger, and N. Weber. 2009. Semantic annotation of epc models in engineering domains to facilitate an automated identification of common modelling practices. ." *Enterprise Information Systems.* 155–171.
- [Bogue, 2005]Bogue, R. 2005. Anatomy of software development role: Solution architect. 12 May 2005. <http://www.developer.com/mgmt/article.php/3504496/Anatomy-of-a-Software-Development-Role-Solution-Architect.htm> [accessed October 28, 2009].
- [Bondé, Boulet and Dekeyser, 2006]Bondé, L., P. Boulet, and J.-L. Dekeyser. 2006. Traceability and interoperability at different levels of abstraction in model-driven engineering. *Applications of Specification and Design Languages for SoCs.* 263-276.
- [Boronat, Carsí and Ramos, 2005]Boronat, A., J. Carsí, and I. Ramos. 2005. Automatic support for traceability in a generic model management framework. In *Model Driven Architecture- Foundations and Applications*, 316-330. Springer Berlin / Heidelberg.
- [Borst, 1997]Borst, W. N. 1997. *Construction of engineering ontologies for knowledge sharing and reuse.* Doctoral Thesis of the University of Twente: Enschede, The Netherlands.
- [Brahe, 2007]Brahe, S. 2007. BPM on Top of SOA: Experiences from the Financial Industry. *Business Process Management, 5th International Conference, BPM 2007.* Brisbane, Australia: Springer, 2007. 96-111.
- [Brahe and Bordbar, 2007]Brahe, S., and B. Bordbar. 2007. A pattern-based approach to business process modeling and implementation in web services. *Service Oriented Computing ICSOC 2006.* 166-177.
- [BusinessDictionary.com, 2009]BusinessDictionary.com. *BusinessDictionary.com.* 2009. <http://www.businessdictionary.com/definition/best-practice.html> [accessed December 1, 2009].
- [Cantara, 2008] Cantara, M. 2008. Four Paths Characterize BPMS Market Evolution . Gartner Research.
- [Carter, 2007]Carter, S. 2007. *The New Language of Business: SOA & Web 2.0*, 1 ed. IBM Press.
- [Catts and St. Clair, 2009] Catts, A, and J. St. Clair. 2009. *Business Process Management Enabled by SOA.* IBM Redbooks.
- [Champeau and Rochefort, 2003]Champeau, J., and E. Rochefort. 2003. Model engineering and traceability. *UML 2003 SIVOES-MDA Workshop.*
- [Chen, Doumeingts and Vernadat, 2008]Chen, D., G. Doumeingts, and F. Vernadat. 2008. Architectures for

- enterprise integration and interoperability: Past, present and future. *Comput. Ind.* 59(7):647–659.
- [Cheung 2009] Cheung, M. 2009. Business and IT alignment- Introducing Enterprise Architecture, Business Process Management, and Service Oriented Architecture. Research Assignment TU Delft.
- [Cimander and Kubicek, 2009]Cimander, R., and H. Kubicek. 2009. Organizational interoperability and organizing for interoperability in e-government. In *Second European Summit on Interoperability in the iGovernment held in Rome*, 109–122.
- [Cordys, 2009] Cordys. 2009. The Intellegent Cloud Platform. <http://www.cordys.com/> [accessed June, 2009]
- [Craig, 2006]Craig, J. 2006. The new business analyst talks soa. (October). <http://www.networkworld.com/newsletters/nsm/2006/1023nsm2.html> [accessed October 28, 2009].
- [Czarnecki and Helsen, 2003]Czarnecki, K., and S. Helsen. 2003. Classification of model transformation approaches. In *Proceedings of the 2nd OOPSLA Workshop on Generative Techniques in the Context of MDA held in Anaheim*.
- [De Nicola et al., 2008]De Nicola, A., T. Di Mascio, M. Lezoche, and F. Tagliano. 2008. Semantic lifting of business process models. In *EDOCW '08: Proceedings of the 2008 12th Enterprise Distributed Object Computing Conference Workshops*, 120–126. IEEE Computer Society.
- [Dietz and Hoogervorst, 2008]Dietz, J. L. G., and J. A. P. Hoogervorst. 2008. Enterprise ontology in enterprise engineering. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing held in Fortaleza, Ceara, Brazil*, 572–579. ACM.
- [Dijkman, Dumas and Ouyang, 2008]Dijkman, R., M. Dumas, and C. Ouyang. 2008. Semantics and analysis of business process models in bpmn. *Information and Software Technology* 50(12):1281–1294. Elsevier Science Publishers B. V.
- [Dreiling et al., 2008]Dreiling, A., M. Rosemann, W. M. P. van der Aalst, and W. Sadiq. 2008. From conceptual process models to running systems: A holistic approach for the configuration of enterprise system processes. *Decis. Support Syst.* 45(2):189–207. Elsevier Science Publishers B. V.
- [Filipowska et al., 2009]Filipowska, A., M. Hepp, M. Kaczmarek, and I. Markovic. 2009. Organisational ontology framework for semantic business process management. *Business Information Systems*. 1–12.
- [Gakkhar 2009]Gakkhar, Kopal. 2009. *Cordys BOP-4 Compliance with BPMN 2.0*. Internal document Cordys.
- [Geerts and McCarthy, 2000]Geerts, G. L., and W. E. McCarthy. 2000. The ontological foundation of rea enterprise information systems. *The American Accounting Association Conference*.
- [Gitzel and Merz 2004] Gitzel, R, and M. Merz. 2004. “How a Relaxation of the Strictness Definition Can Benefit MDD Approaches With Meta Model Hierarchies.” *8th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI2004)*. SCI, Orlando, USA.
- [Gomez-Perez, Corcho and Fernandez-Lopez, 2004]Gomez-Perez, A., O. Corcho, and M. Fernandez-Lopez. 2004. *Ontological engineering : with examples from the areas of knowledge management, e-commerce and the semantic web. first edition (advanced information and knowledge processing)*. Springer.
- [González, Casallas and Deridder, 2009]González, O., R. Casallas, and D. Deridder. 2009. Mmc-bpm: A domain-specific language for business processes analysis. In *Business Information Systems*, 157–168. Springer.
- [Gordijn, Akkermans and van Vliet, 2000]Gordijn, J., H. Akkermans, and H. van Vliet. 2000. Business modelling is not process modelling. In *ER '00: Proceedings of the Workshops on Conceptual Modeling Approaches for E-Business and The World Wide Web and Conceptual Modeling*, 40–51. Springer-Verlag.
- [Gordijn, and Akkermans 2000]Gordijn, J, and H. Akkermans. 2000. “E3-value: Design and Evaluation of e-

- Business Models." *IEEE Intelligent Systems* 16, no. 4. 11-17.
- [Green and Rosemann, 1999]Green, P., and M. Rosemann. 1999. An ontological analysis of integrated process modelling. *Advanced Information Systems Engineering*:225–240.
- [Green, 2000]Green, P. 2000. Integrated process modeling: An ontological evaluation. *Information Systems* 25(2):73–87. Elsevier Science Ltd.
- [Gruber 1995] Gruber, T. 1995. "Toward Principles for the Design of Ontologies Used for Knowledge Sharing." *International Journal Human-Computer Studies* 43, no. 5-6, 907-928.
- [Gruhn and Laue, 2007]Gruhn, V., and R. Laue. 2007. Approaches for business process model complexity metrics. In *Technologies for Business Information Systems*, 13–24. Springer The Netherlands.
- [Grüninger, Atefi and Fox, 2000]Grüninger, M., K. Atefi, and M. S. Fox. 2000. Ontologies to support process integration in enterprise engineering. *Computational & Mathematical Organization Theory* 6(4):381–394.
- [Guédria, Naudet and Chen, 2009]Guédria, W., Y. Naudet, and D. Chen. 2009. Interoperability maturity models: a survey and comparison". 273–282.
- [Hammer and Champy, 1994]Hammer, M., and J. Champy. 1994. *Reengineering the corporation: A manifesto for business revolution*. HarperBusiness.
- [Hepp and Roman, 2007]Hepp, M., and D. Roman. 2007. An ontology framework for semantic business process management. In *Wirtschaftsinformatik (1)*, edited by A. Oberweis, C. Weinhardt, H. Gimpel, A. Koschmider, V. Pankratius, B. Schnizler, A. Oberweis, C. Weinhardt, H. Gimpel, A. Koschmider, V. Pankratius, and B. Schnizler, 423–440. Universitaetsverlag Karlsruhe.
- [Höfferer, 2007] Höfferer, P. 2007. Achieving Business Process Model Interoperability Using Metamodels and Ontologies. Edited by Schelp, J., Winter, R. eds. H. *Proceedings of the 15th European Conference on Information Systems (ECIS2007)*. Österle, 2007. 1620-1631.
- [Hoyer, Bucherer and Schnabel, 2008]Hoyer, V., E. Bucherer, and F. Schnabel. 2008. Collaborative e-business process modelling: Transforming private epc to public bpmn business process models. 185–196.
- [IDS_Scheer_Academy 2009]IDS_Scheer_Academy. 2009. Service-enabling Processes with ARIS SOA Architect. IDS Scheer SOA training.
- [IDSScheer 2009]IDSScheer. 2009. *IDS Scheer Business Process Excellence*. <http://www.ids-scheer.com/international/en> [accessed June 2009].
- [Ikeda et al., 1998]Ikeda, M., K. Seta, O. Kakusho, and R. Mizoguchi. 1998. Task ontology: ontology for building conceptual problem solving models. 126–133.
- [Indulska et al., 2007]Indulska, M., J. Recker, P. Green, and M. Rosemann. 2007. Are we there yet? seamless mapping of bpmn to bpel4ws.
- [Indulska et al., 2009]Indulska, M., J. Recker, M. Rosemann, and P. Green. 2009. Business process modeling: Current issues and future challenges. 501–514.
- [Information Society Technologies 2006] Information Society Technologies, European Union 6th Framework. 2006. *Semantics Utilised for Process Management within and between Enterprises (SUPER Intergated Project)*. <http://www.ip-super.org/> [accessed November 12, 2009].
- [Jeston and Nelis, 2008]Jeston, J., and J. Nelis. 2008. *Management by process: A practical road-map to sustainable business process management*. Butterworth-Heinemann.
- [Jouault and Kurtev, 2007]Jouault, F., and I. Kurtev. 2007. On the interoperability of model-to-model transformation languages. *Science of Computer Programming* 68(3):114–137.
- [Kajko-Mattson, Lewis and Smith, 2008]Kajko-Mattsson, M., G. Lewis, and D. Smith. 2008. Evolution and maintenance of soa-based systems at sas. *Hawaii International Conference on System Sciences*.

- [Kamoun, 2007]Kamoun, F. A roadmap towards the convergence of business process management and service oriented architecture. *Ubiquity* 2007(April):1.
- [Kappel et al., 2006]Kappel, G., E. Kapsammer, H. Kargl, G. Kramler, T. Reiter, W. Retschitzegger, W. Schwinger, and M. Wimmer. 2006. Lifting metamodels to ontologies: A step to the semantic integration of modeling languages. 528–542.
- [Kapsammer et al., 2006]Kapsammer, E., H. Kargl, G. Kramler, T. Reiter, W. Retschitzegger, and M. Wimmer. 2006. On models and ontologies - a layered approach for model-based tool integration. In *in Proceedings of the Modellierung 2006 (MOD2006)*, 11–27.
- [Koehler et al., 2008]Koehler, J., R. Hauser, J. Kuster, K. Ryndina, J. Vanhatalo, and M. Wahler. 2008. The role of visual modeling and model transformations in business-driven development. *Electronic Notes in Theoretical Computer Science* 211(April):5–15.
- [Kolovos, Paige and Polack, 2006]Kolovos, D. S., R. F. Paige, and F. A. C. Polack. 2006. On-demand merging of traceability links with models. *Proceedings of the 2nd EC-MDA Workshop on Traceability*.
- [Kraft, 2009]Kraft, F.M. 2009. BPMN Diagram Exchange Reflections. <http://www.bpmnforum.net/blog27/> [accessed December 1, 2009].
- [Laue and Gruhn, 2007]Gruhn, V., and R. Laue. 2007b. What business process modelers can learn from programmers. *Sci. Comput. Program.* 65(1):4–13. Elsevier North-Holland, Inc.
- [Limón and Garbajosa, 2005]Limón, A. E., and J. Garbajosa. 2005. The need for a unifying traceability scheme. In *2nd ECMDA-Traceability Workshop*.
- [List and Korherr, 2006]List, B., and B. Korherr. 2006. An evaluation of conceptual business process modelling languages. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing held in Dijon, France*, 1532–1539. ACM.
- [Maas, 2008]Maas, H. 2008. Business Process Architecture & Modeling Guidelines. Internal documentation, IDS Scheer NL
- [Meijler, Kruithof and van Beest, 2006]Meijler, T., G. Kruithof, and N. van Beest. 2006. Top Down Versus Bottom Up in Service-Oriented Integration: An MDA-Based Solution for Minimizing Technology Coupling. *Service-Oriented Computing: ICSOC 2006*, 484–489.
- [Mendling and Strembeck, 2008]Mendling, J., and M. Strembeck. 2008. Influence factors of understanding business process models. 142–153.
- [Mendling, 2009]Mendling, J. 2009. Event-driven process chains (epc). 17–57.
- [Mendling, Neumann and Nüttgens, 2005]Mendling, J., G. Neumann, and M. Nüttgens. 2005. Towards workflow pattern support of event-driven process chains (EPC). In *Proc. of the 2nd Workshop XML4BPM*, 23–38.
- [Mendling, Neumann and van der Aalst, 2007]Mendling, J., G. Neumann, and W. van der Aalst. 2007. On the correlation between process model metrics and errors. In *ER '07: Tutorials, posters, panels and industrial contributions at the 26th international conference on Conceptual modeling held in Auckland, New Zealand*, 173–178. Australian Computer Society, Inc.
- [Mendling, Reijers and Recker, 2009]Mendling, J., H. A. Reijers, and J. Recker. 2009. Activity labeling in process modeling: Empirical insights and recommendations. *Information Systems* (April), 233–241.
- [Mendling, Reijers and van der Aalst, 2009]Mendling, J., H. A. Reijers, and W. M. P. van der Aalst. 2009. Seven process modeling guidelines (7pmg). *Information and Software Technology* (August).
- [Morganthal, 2009]Morganthal, J. P. 2009. The role of the business analyst in an soa world. blog. <http://www.jpmorganthal.com/morganthal/?p=114> [accessed October 28, 2009].
- [Muehlen and Recker, 2008]Muehlen, M., and J. Recker. 2008. How much language is enough? theoretical and practical use of the business process modeling notation. 465–479.

- [Muehlen, 2004]Muehlen, M. Z. 2004. Workflow-based process controlling. foundation, design, and application of workflow-driven process information systems. Logos.
- [Murzek (1) and Kramler, 2007]Murzek, M., and G. Kramler. 2007. The model morphing approach - horizontal transformations between business process models. In *Proceedings of the 6th International Conference on Perspectives in Business Information Research - BIR 2007*, edited by J. Nummenmaa and E. Söderström, 88–103. Department of Computer Sciences, University of Tampere.
- [Murzek (2) and Kramler, 2007]Murzek, M., and G. Kramler. 2007. Business process model transformation issues - the top 7 adversaries encountered at defining model transformations. *ICEIS* 3:144–151.
- [Murzek, Kramler and Michlmayr, 2006]Murzek, M., G. Kramler, and E. Michlmayr. 2006. Structural patterns for the transformation of business process models. In *EDOCW '06: Proceedings of the 10th IEEE on International Enterprise Distributed Object Computing Conference Workshops*, 18+. IEEE Computer Society.
- [Niles and Pease, 2001]Niles, I., and A. Pease. 2001. Towards a standard upper ontology. In *FOIS '01: Proceedings of the international conference on Formal Ontology in Information Systems held in Ogunquit, Maine, USA*, 2–9. ACM.
- [Oberortner, Zdun and Dustdar, 2008]Oberortner, E., U. Zdun, and S. Dustdar. 2008. Domain-specific languages for service-oriented architectures: An explorative study. In *Towards a Service-Based Internet*, Lecture Notes in Computer Science, Chapter 14, 159–170.
- [OMG, 2003]OMG. 2003. MDA Guide V1.0.1. Object Management Group
- [OMG, 2009]OMG. 2009. Business Process Model and Notation (BPMN) Specification 2.0. V0.9.14
- [Osterwalder, 2004]Osterwalder, A. 2004. *The business model ontology - a proposition in a design science approach*. University of Lausanne, Lausanne, Switzerland
- [Pedrinaci, Domingue and Alves, 2008]Pedrinaci, C., J. Domingue, and Alves. 2008. A core ontology for business process analysis. *The Semantic Web: Research and Applications*, 49–64. Springer Berlin/Heidelberg
- [Perez, Ruiz and Piattini, 2008]Perez, J. M., F. Ruiz, and M. Piattini. 2008. Mde for bpm: A systematic review. *Software and Data Technologies*, 127–135.
- [Prakash, Srivastava and Sabharwal, 2006]Prakash, N., S. Srivastava, and S. Sabharwal. 2006. The classification framework for model transformation. *Journal of Computer Science*, 1532–1539.
- [Ramesh and Jarke, 2001]Ramesh, B., and M. Jarke. 2001. Toward reference models for requirements traceability. *IEEE Trans. Softw. Eng.* 27(1):58–93.
- [Recker, Wohed and Rosemann, 2006]Recker, J., P. Wohed, and M. Rosemann. 2006. Representation theory versus workflow patterns: The Case of BPMN. *Conceptual Modeling - ER 2006*, 68–83.
- [Roser and Bauer, 2005]Roser, S., and B. Bauer. 2005. A categorization of collaborative business process modeling techniques. *E-Commerce Technology Workshops, Seventh IEEE International Conference on*, 43–54.
- [Roser and Bauer, 2006]Roser, S., and B. Bauer. 2006. An approach to automatically generated model transformations using ontology engineering space. *Proceedings of Workshop on Semantic Web Enabled Software Engineering (SWESE)*, 332-342.
- [Sadiq and Orłowska, 2000]Sadiq, W., and M. Orłowska. 2000. On business process model transformations. *Conceptual Modeling*, 47-104. Springer Berlin/ Heidelberg, 2000.
- [Saeki and Kaiya, 2006]Saeki, M., and H. Kaiya. 2006. On relationships among models, meta models and ontologies. *Proceedings of the 6th OOPSLA Workshop on Domain-Specific Modeling*.
- [Scheer and Klueckmann, 2009]Scheer, A.-W., and J. Klueckmann. 2009. Bpm 3.0. *Business Process Management*, 15-27.

- [Scheer and Schneider, 1998]Scheer, A.-W., and K. Schneider. 1998. ARIS: a architecture of integrated information systems. *Handbook on Architectures of Information Systems*, 605-623. Springer Berlin Heidelberg, 1998.
- [Seeley, 2008]Seeley, R. 2008. Business analyst role key in soa software development projects. 14 August 2008. http://searchsoa.techtarget.com/news/article/0,289142,sid26_gci1325388,00.html [accessed October 28, 2009].
- [Seruca and Loucopoulos, 2003]Seruca, I., and P. Loucopoulos. 2003. Towards a systematic approach to the capture of patterns within a business domain. *Journal of Systems and Software* 67(1):1-18.
- [Silver, 2009]Silver, B., 2009. XPD-L-BPMN Mapping, including DI. 29 July 2009. <http://www.bpmnstyle.com/> [accessed December 1, 2009].
- [Smith and Fingar, 2002]Smith, H., and P. Fingar. 2002. *Business process management (BPM): The third wave*. Meghan-Kiffer Press.
- [Söderström et al., 2002]Söderström, E., B. Andersson, P. Johannesson, E. Perjons, and B. Wangler. 2002. Towards a framework for comparing process modelling languages. *Advanced Information Systems Engineering*, 600-611. Springer Berlin/ Heidelberg
- [Stein, Kühne and Ivanov, 2009]Stein, S., S. Kühne, and K. Ivanov. 2009. Business to it transformations revisited. *Business Process Management Workshops*, 176-187.
- [Stein, Lauer and Ivanov, 2008] Stein, S., J. Lauer, and K. Ivanov. 2008. Aris method extension for business-driven soa. *WIRTSCHAFTSINFORMATIK* 50, no. 6, 436-444.
- [Tairas, Mernik and Gray, 2009]Tairas, R., M. Mernik, and J. Gray. 2009. Using ontologies in the domain analysis of domain-specific languages. *Models in Software Engineering*, 332-342. Springer-Verlag
- [Tekinerdogan, Hofmann and Aksit, 2007]Tekinerdogan, B., C. Hofmann, and M. Aksit. 2007. Modeling traceability of concerns in architectural views. In *AOM '07: Proceedings of the 10th international workshop on Aspect-oriented modeling held in Vancouver, Canada*, 49-56. ACM.
- [Theling et al., 2005]Theling, T., J. Zwicker, P. Loos, and D. Vanderhaeghen. 2005. An architecture for collaborative scenarios applying a common BPMN-repository. *Distributed applications and Interoperable Systems*, 169-180. Springer.
- [Thom et al., 2008]Thom, L. H., M. Reichert, C. M. Chiao, C. Iochpe, and G. N. Hess. 2008. Inventing less, reusing more, and adding intelligence to business process modeling. In *DEXA '08: Proceedings of the 19th international conference on Database and Expert Systems Applications held in Turin, Italy*, 837-850. Springer-Verlag.
- [Thomas and Fellmann, 2007]Thomas, O., and M. Fellmann. 2007. Semantic epc: Enhancing process modeling using ontology languages. In *SBPM*, edited by M. Hepp, K. Hinkelmann, D. Karagiannis, R. Klein, N. Stojanovic, M. Hepp, K. Hinkelmann, D. Karagiannis, R. Klein, and N. Stojanovic, Volume 251 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- [Tolk, Diallo and Turnitsa, 2007]Tolk, A., S. Diallo, and C. Turnitsa. 2007. Applying the levels of conceptual interoperability model in support of integratability, interoperability, and composability for system-of-systems engineering. *Systemics, Cybernetics and Informatics* 5(5):65-74.
- [Turnitsa, 2005]Turnitsa, C. 2005. Extending the levels of conceptual interoperability model. IEEE CS Press.
- [van der Aalst, 1999]van der Aalst, W. 1999. Formalization and verification of event-driven process chains. *Information and Software Technology* 41(10):639-650.
- [van der Aalst et al., 2003]van der Aalst, W. M. P., Arthur, B. Kiepuszewski, and A. P. Barros. 2003. Workflow patterns. *Distributed and Parallel Databases* 14(1):5-51.
- [van der Aalst, Desel and Kindler, 2002]van der Aalst, W. M. P., J. Desel, and E. Kindler. 2002. On the semantics of epcs: A vicious circle. In *EPK*, edited by M. Nüttgens, F. J. Rump, M. Nüttgens, and F. J. Rump, 71-79. GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten.

- [Vanderfeesten et al., 2007]Vanderfeesten, I., J. Cardoso, H. A. Reijers, and W. Van Der Aalst. 2007. Quality metrics for business process models." *Future Strategies*.179-190.
- [Vanderhaeghen et al., 2007]Vanderhaeghen, D., S. Zang, A. Hofer, and O. Adam. 2005. "XML-based Transformation of Business Process Models: Enabler for Collaborative Business Process Management." *XML4BPM*. 273-291.
- [Vergidis, Turner and Tiwari, 2008]Vergidis, K., C. Turner, and A. Tiwari. 2008. Business process perspectives: Theoretical developments vs. real-world practice. *International Journal of Production Economics* 114(1):91–104.
- [W3C, Booth and Haas, 2004]Booth, D., and H. Haas. 2004. *Web services architecture*. W3CRecommendation. Technical report.
- [W3C, 1993] W3C. 1993. *XML Path Language (XPath) Version 1.0*. W3C Recommendation, W3C.
- [Wahl and Sindre, 2005]Wahl, T., and G. Sindre. 2005. An analytical evaluation of bpmn using a semiotic quality framework. In *In CAiSE'05 Workshops. Volume*, 533–544.
- [Wand and Weber, 1990]Wand, Y., and R. Weber. 1990. An ontological model of an information system. *IEEE Trans. Softw. Eng.* 16(11):1282–1292.
- [Wang, Tolk and Wang, 2009]Wang, W., A. Tolk, and W. W. 0002. 2009. The levels of conceptual interoperability model: applying systems engineering principles to m&s. In *SpringSim*, edited by G. A. Wainer, C. A. Shaffer, R. M. McGraw, M. J. Chinni, G. A. Wainer, C. A. Shaffer, R. M. McGraw, and M. J. Chinni. SCS/ACM.
- [Weber, Markovic and Drumm, 2007]Weber, I., I. Markovic, and C. Drumm. 2007. A conceptual framework for composition in business process management. In *Business Information Systems, Lecture Notes in Computer Science*, Chapter 5, 54–66.
- [Whitman and Panetto, 2006]Whitman, L., and H. Panetto. 2006. The missing link: Culture and language barriers to interoperability. *Annual Reviews in Control* 30(2):233–241.
- [Wohed et al., 2006]Wohed, P., W. van der Aalst, M. Dumas, A. Ter Hofstede, and N. Russell. 2006. On the suitability of bpmn for business process modelling. *Business Process Management*, 161–176. Springer Berlin/Heidelberg
- [Wong and Gibbons, 2008]Wong, P. Y., and J. Gibbons. 2008. A process semantics for bpmn. In *ICFEM '08: Proceedings of the 10th International Conference on Formal Methods and Software Engineering held in Kitakyushu-City, Japan*, 355–374. Springer-Verlag.
- [Ye et al., 2008]Ye, J., S. Sun, W. Song, and L. Wen. 2008. Formal semantics of bpmn process models using YAWL. *Intelligent Information Technology Applications, 2007 Workshop on* 2:70–74. IEEE Computer Society.
- [Zachman 2008] Zachman, J. 2008. *John Zachman's Concise Definition of the The Zachman Framework™*. <http://www.zachmaninternational.com> [accessed November 2009].

List of Abbreviations

ARIS	ARchitecture of Information Systems
BPA	Business Process Analysis
BPD SOA	Business Process Driven Service Oriented Architecture
BPM	Business Process Management
BPMN	Business Process Modeling Notation
BPMo	Business Process Modelling
BPMS	Business Process Management Suite
BPEL	Business Process Execution Language
BWW	Bunge Wand Weber Information – ontology for information systems
C-BPMN	Cordys BPMN
DI	Diagram Interchange
DSML	Domain Specific Modelling Language
EPC	Event-driven Process Chain
KPI	Key Performance Indicator
MDA	Model Driven Architecture
MDE	Model Driven Engineering
MOF	Meta-Object Facility
OEM	Original Equipment Manufacturer
OMG	Object Management Group
SIP	Service Interaction Patterns
SOA	Service Oriented Architecture
WfMC	Workflow Management Coalition
WP	Workflow Patterns
WSDL	Web Service Description Language
XMI	XML Metadata Interchange
XPDL	XML Process Definition Language
XSTL	EXtensible Stylesheet Language

Reflection

In the master Information Architecture I was introduced to the methodologies of professor Dietz, to wit Enterprise Ontology and Enterprise Architecture, and DEMO. I have gained insights on systems thinking and policy analysis. I consider this set of skills as the foundation in the world of business processes. Context analysis and a systematic approach are necessary to deal with the complexity of business processes. Also the importance of enterprise ontology has become clear. There are many different methodologies, standards, notations and perspectives in the business domain. Starting this project, all the business terms, and different methodologies were somewhat overwhelming. But in time the purpose and essence of the methodologies, and their relations became clear.

During this project I have gained practical experience of all I had learnt during the master. Experiencing what literature describes puts things in perspective. On one hand, theories and claims are confirmed. On the other hand, showing the gap between what practitioners use and what researchers develop. A good experience was an interview with a business member using ARIS and IT member using Cordys, to see how they actually use the tools, and awareness was created on how they could benefit from a coupling of ARIS and Cordys.

During my time at IDS Scheer, Software AG has taken over IDS Scheer. Software AG has a product, webMethods, which is a direct competitor of Cordys. Luckily, it did not impact my project regarding content. Both IDS Scheer and Cordys are still enthusiastic about the future cooperation. I hope that some of the findings can contribute to the merger with Software AG's webMethods. From organizational perspective, it was very exciting to see how a merger influences company culture, and its employees. My initial supervisor had taken precautionary measures, and had accepted another job offer.

I have experienced this thesis project as very challenging, and learned a lot about different (broad) topics. I had to deal with many different stakeholders. It is very interesting and challenging to deal with so many perspectives, but it also means having to compensate (due to time constraints). In the end I can say that I have learnt a great deal about the world of business processes, and am ready to be challenged in practice!

Melissa Cheung,
13 January 2010, Delft

Appendices

Appendix A. ARIS Business-Driven SOA by IDS Scheer

ARIS is a tool provided by IDS Scheer, based on the methodology of Professor A.W. Scheer. ARIS is recognized as market leader by Forrester and Gartner as Business Process Analysis and Enterprise Architecture tool. ARIS attempts to capture the whole organization in its design and operation. It is mainly a design and planning tool. Execution of processes is supported by integration of ARIS with other execution platforms.

1 The IDS Scheer BPM life cycle

The BPM life cycle is the essence for every solution offered by IDS Scheer. The BPM life cycle will be described within the BPD SOA context. Furthermore, more detailed analysis is provided concepts, activities and roles of BPD SOA according to IDS Scheer.

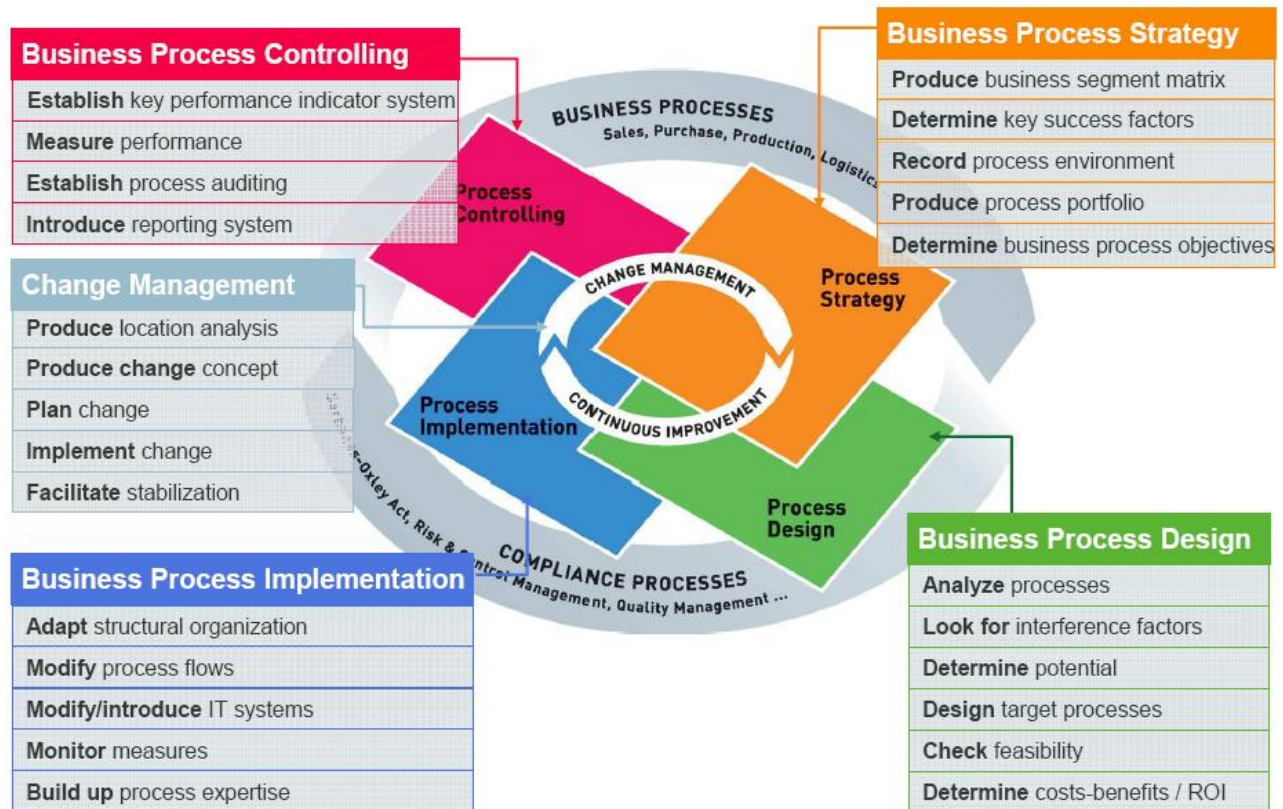


Figure 32 IDS Scheer BPM life cycle

1.1 The Business Driven SOA life cycle

The phases in Business Driven SOA have the same phases as the BPM life cycle (Stein, Lauer and Ivanov, 2008). Within the *strategy phase* cooperate objectives are determined. These cooperate objectives are without IT detail. However, they are the foundation for every activity in the consecutive phases. These objectives are translated into requirements and measured by Key Performance Indicators (KPIs). Understanding the business context and developing a strategic plan are central in this phase. The objectives, KPIs and requirements are captured within ARIS specific models. The *design phase* comprises of modelling of the business processes, as-is and to-be, and their supporting IT architecture. For SOA the functional processes can be considered as service requesters/consumers and the supporting IT as service providers. This phase also entails service architecture planning in order to reduce redundancy. Considering services during process design enlightens the relation of service with business needs. Modelling of processes is central in this phase, and several related architecture and allocation diagrams are developed to depict relations in the infrastructure. The *implementation phase* addresses the modelling of

orchestration of technical services upon the functional business processes. Important in this phase a service repository relating technical services with functional processes. This phase may address existing services and non-existent services. In the latter case, Model Driven Architecture can be used to derive services. Another important aspect is business rules which describe the process branching. These can also be orchestrated as technical services. ARIS does not support the actual execution of these processes. Implementation regards to the deployment of executable processes, which in this case is support for the conversion to BPEL and export facilities. The *controlling* phase regards the performance of the SOA, and is measured in an execution system. Process monitoring and Business Activity Monitoring can be used to measure, for example, duration of service execution and intercommunication between services. Controlling and monitoring activities give insight on to what extent the objectives are accomplished. The modelling phases, steps, products and specific models and activities are summarized in the table below.

Phase	Steps	Deliverables	Specific Models & modelling activities
Process strategy	<ul style="list-style-type: none"> Understand business environment Record enterprise map Define business service maps Determine end-to-end scenarios Define project scope and plan 	<ul style="list-style-type: none"> Business segment matrix Enterprise process map Business service maps End-to-end process scenarios Project and organization plan 	<ul style="list-style-type: none"> Objectives diagram KPI allocation diagram Requirements allocation diagram
Process design	<ul style="list-style-type: none"> Tailor service architecture framework Develop to-be concepts and processes Define business services Discover software services Orchestrate service 	<ul style="list-style-type: none"> Service architecture framework To-be concept & processes Business service model Software service model SOA process models 	<ul style="list-style-type: none"> EPC (service oriented, added clusters, capabilities and business services) Business Service Diagram Import SOA profile and data structure Access diagram Application system type diagram Service allocation diagram Service architecture diagram
Process Implementation	<ul style="list-style-type: none"> Orchestrate services (technical) Platform independent implementation Test service implementation 	<ul style="list-style-type: none"> Technical process models Implemented processes and services Service architecture repository 	<ul style="list-style-type: none"> EPC2BPEL Generate WSDL Export BPEL
Process control	<ul style="list-style-type: none"> Control running services 	<ul style="list-style-type: none"> Process performance Instance monitoring 	

Table 15 Summary ARIS Business Driven SOA

1.2 Roles

The roles and responsibilities depict the different people involved and their perspectives in BPD SOA. The following roles have been identified in business driven SOA development according to IDS Scheer:

1. The *Enterprise architect* is responsible for developing architectural strategy to enable enterprise wide interoperability. This is achieved by means of the development of enterprise standards and policy and mapping of the business and IT architecture.

2. The *Business analyst* is responsible for requirements management, modelling the as-is and to-be business processes and authorizing the process changes.
3. The *Process engineer* transforms the business process models into technical processes. He is also responsible for ongoing synchronization.
4. The *IT architect* is responsible for managing the IT infrastructure, including all systems and services. Furthermore, defines architecture standards and plans IT development.
5. The *Software engineer* develops new services or composes existing services.
6. The *Integration engineer* develops the integration logic for legacy and new applications.
7. The *Project manager* is responsible for defining the project scope, planning and managing of the project execution.

IDS Scheer has developed ARIS Value Engineering as their roadmap to offer solutions, such as business driven SOA. They have identified several steps within each phase for implementing business driven SOA. These activities should be executed by the roles identified above. The table summarizes the AVE method and maps the roles relevant in each phase.

Phase	Steps	Roles
Process strategy	<ol style="list-style-type: none"> 1. Understand business environment 2. Record enterprise map 3. Define business service maps 4. Determine end-to-end scenarios 5. Define project scope and plan 	<ul style="list-style-type: none"> · Enterprise architect · Business analyst · Project manager
Process design	<ol style="list-style-type: none"> 6. Tailor service architecture framework 7. Develop to-be concepts and processes 8. Define business services 9. Discover software services 10. Orchestrate service 	<ul style="list-style-type: none"> · Business analyst · IT architect · Process engineer
Process Implementation	<ol style="list-style-type: none"> 11. Orchestrate services (technical) 12. Platform independent implementation 13. Test service implementation 	<ul style="list-style-type: none"> · Process engineer · Integration engineer · Software engineer
Process control	<ol style="list-style-type: none"> 14. Control running services 	<ul style="list-style-type: none"> · Project manager · Administrator

Table 16 Summary AVE and roles

2 ARIS concepts for modelling SOA

There are two perspectives in business driven SOA life cycle: process automation and service architecture. The strategy phase revolves about enterprise architecture. Creating an enterprise map includes identification of enterprise processes and areas from process perspective, and the business services and the areas from service perspective. SOA development in ARIS is based on the model driven architecture [Stein, Lauer and Ivanov 2008]. The design phase is concerned with modelling on computation independent level, where the main business processes are developed, and business services and capabilities are allocated. The business services and capabilities contribute to service discovery, orchestration and design. A business service is an application programming interface that can be accessed over a network to perform a service. Capabilities can be used to describe functional and non-functional properties, although service level agreements are recommended as they are more expressive. A capability can be reused to describe different services. Therefore, the context in which each capability can be used must be clearly defined. The implementation phase regards the development of service oriented business processes and allocation

of software services. Software services are software service types. Software service types are platform independent software based services used in workflow or integration processes, which are related to the business processes. A service type is a mechanism enabling access to set capabilities, and can be specified by different service providers. A service provider must have all capabilities specified by the service type. Service types can be linked to a business function in order to specify the capabilities necessary to performing the business function. The implementation phase also addresses platform specific modelling with development of BPEL processes and WSDL implementations. The service perspective according to MDA levels is depicted in the figure. The relation between business services, capabilities, software services and implementation is visualized.

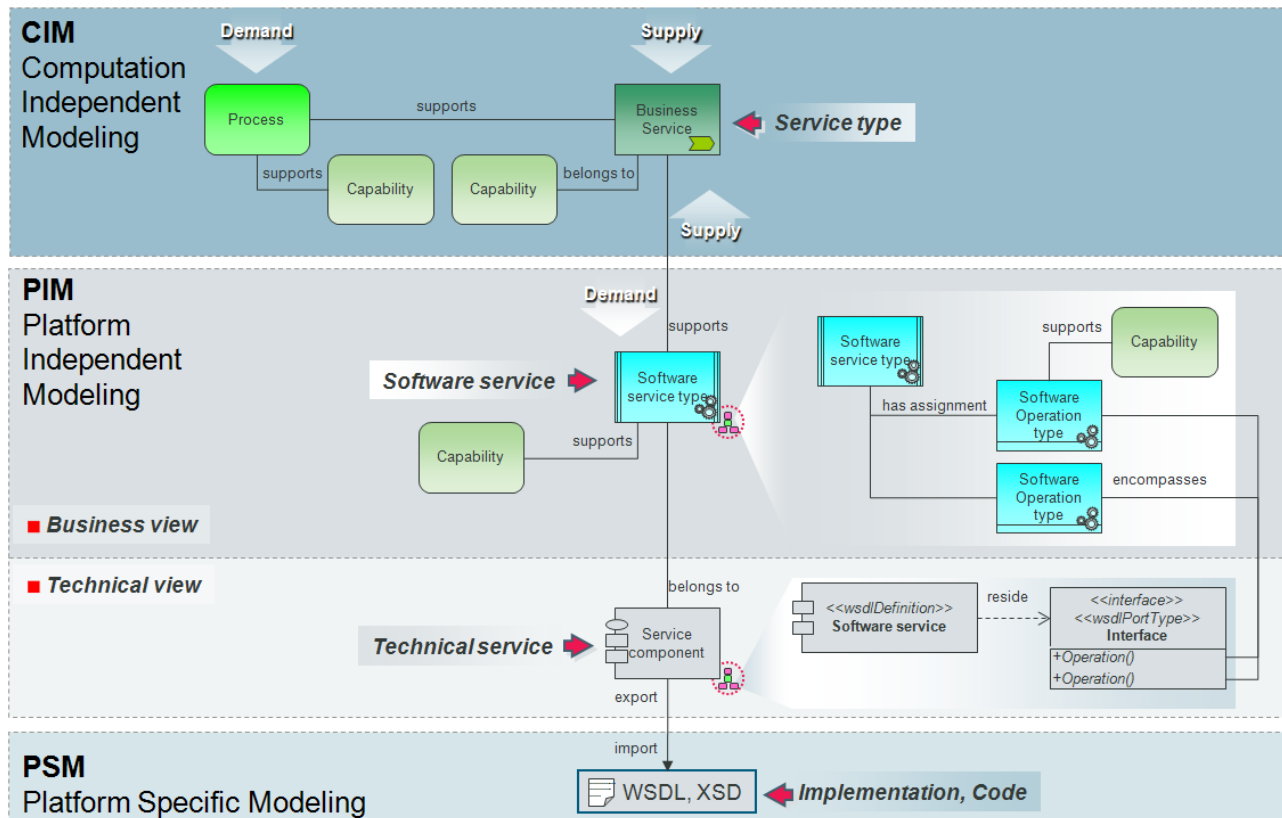


Figure 33 MDA and ARIS services

2.1 ARIS SOA specific models

There are several models which should be developed for BPD SOA in ARIS, version 7.1. As this thesis focus on process modelling the models developed to gain insight on the business context are left out of consideration. However there are two models which depict information that is very relevant: the requirements allocation diagram and the organizational chart. The requirements allocation diagram depicts the context of the requirements, relating it to objectives, Key Performance Indicators and Risks. Requirements are implemented by the capabilities and business services. The organizational chart depicts an organization in organizational units and roles. The constructs are Organizational unit type, System organizational unit type, Location type, Position and Persons.

The process models and relevant linked models are depicted below.

- EPC: process model with functions, events, arcs, and connectors.
- Service oriented EPC: an EPC with service oriented implementation details. Specific constructs are:
 - Application System Type: business oriented representation of a WSDL web service.
 - Capability: describes the ability of an entity. (function view in ARIS)

- Business service (Service Type): represents a service on CIM level, providing access to capabilities. (Function view in ARIS). Business service is an application programming interface that can be accessed over a network to perform a service.
- Software service type: represents a service on PIM level for the business oriented representation of a web service.
- Cluster: data object
- KPI: Key Performance Indicator
- Objective: Business objective

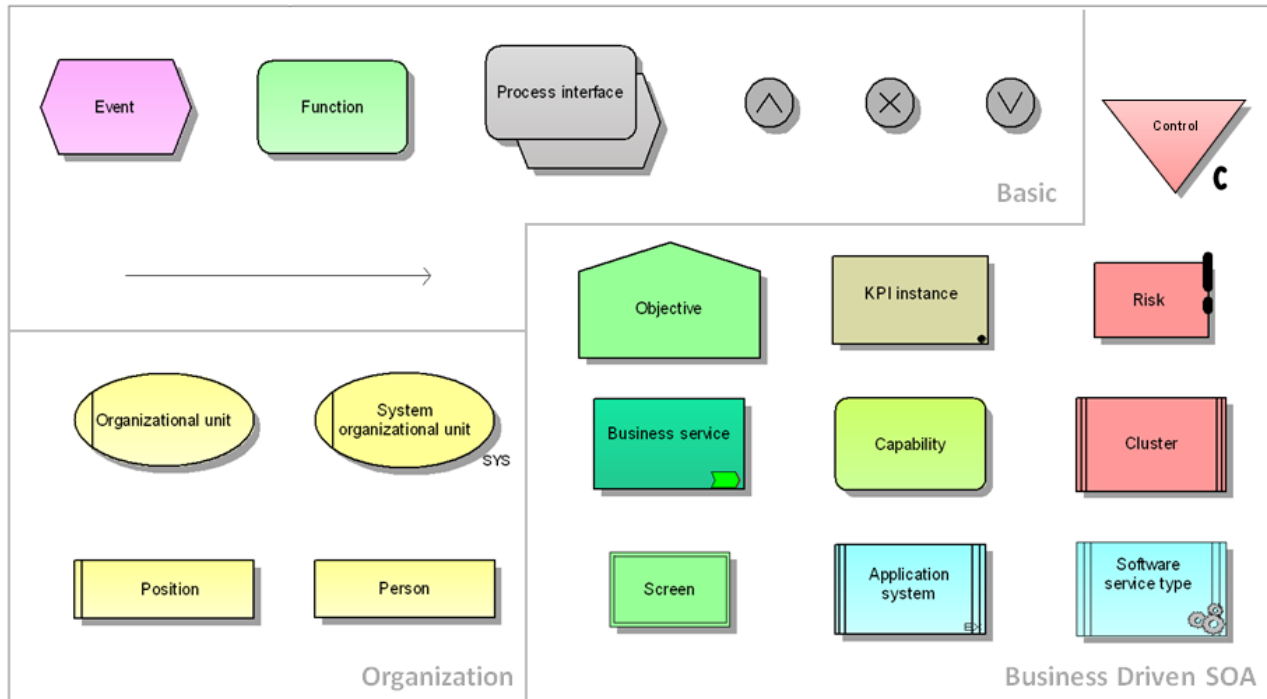


Figure 34 ARIS service oriented EPC constructs

- FAD: Function Allocation Diagram depicts functions in relation with all other sorts of objects. This can be coupled to functions in an EPC, in order to depict a 'lean' EPC.
- BPMN: process model in Business Process Modelling Notation.
- UML Component diagram: technical WSDL details represented by a specific UML profile. Uses UML components and classes.
- Access diagram: The capabilities of a WSDL web service are described by relating the information system functions to the application system types. Instances of a web service are represented by an application system object. The Application System Type is related to capabilities and component.
- Application System Type diagram: the hierarchy of software system types.

The figure depicts the three layers of development of processes and services. In the context it is depicted how the diagrams described above are related to each other.

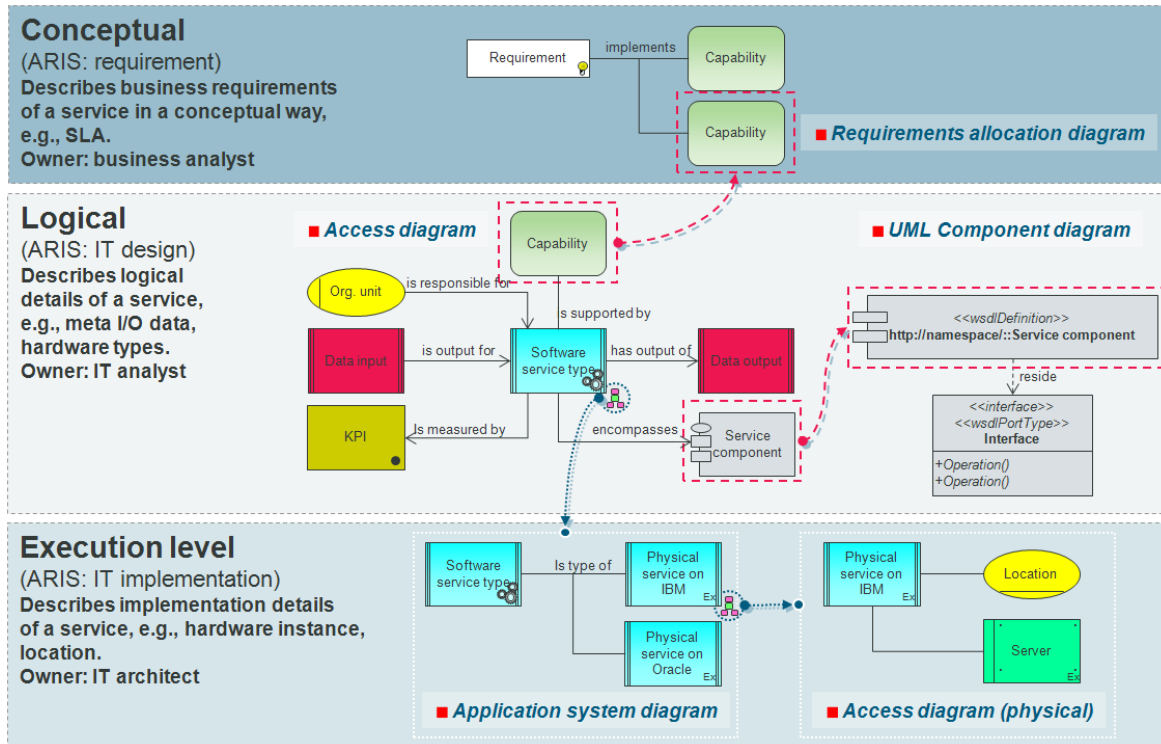


Figure 35 ARIS models relations

- Service allocation diagram: describes the service type by relating it to capabilities, data clusters, service level agreements, company goals, projects, service owner, business objects and existing service providers. (analysis) In this context the capabilities and data clusters are most relevant. Figure is an example of a service allocation diagram.
- Service architecture diagram: model the service architecture by decomposing service type(s) (business services and capabilities) into other service types. Capability architecture is required for managing unused capabilities, integration of new capabilities and the vocabulary for defining the capabilities.

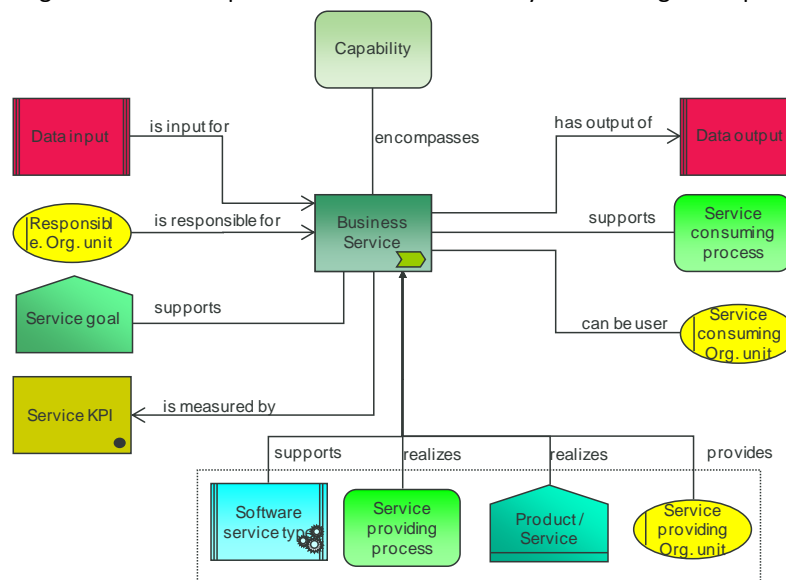


Figure 36 Service allocation diagram

There are two other relevant models, but less implemented:

- Service support matrix: model the spatial and temporal availability of service providers for a given service type.
- Service collaboration diagram: models the dynamic aspects between service types.

2.2 ARIS SOA Process Modelling Activities

Figure 37 depicts the activities in modelling a service oriented EPC. On the right the associated diagrams are depicted.

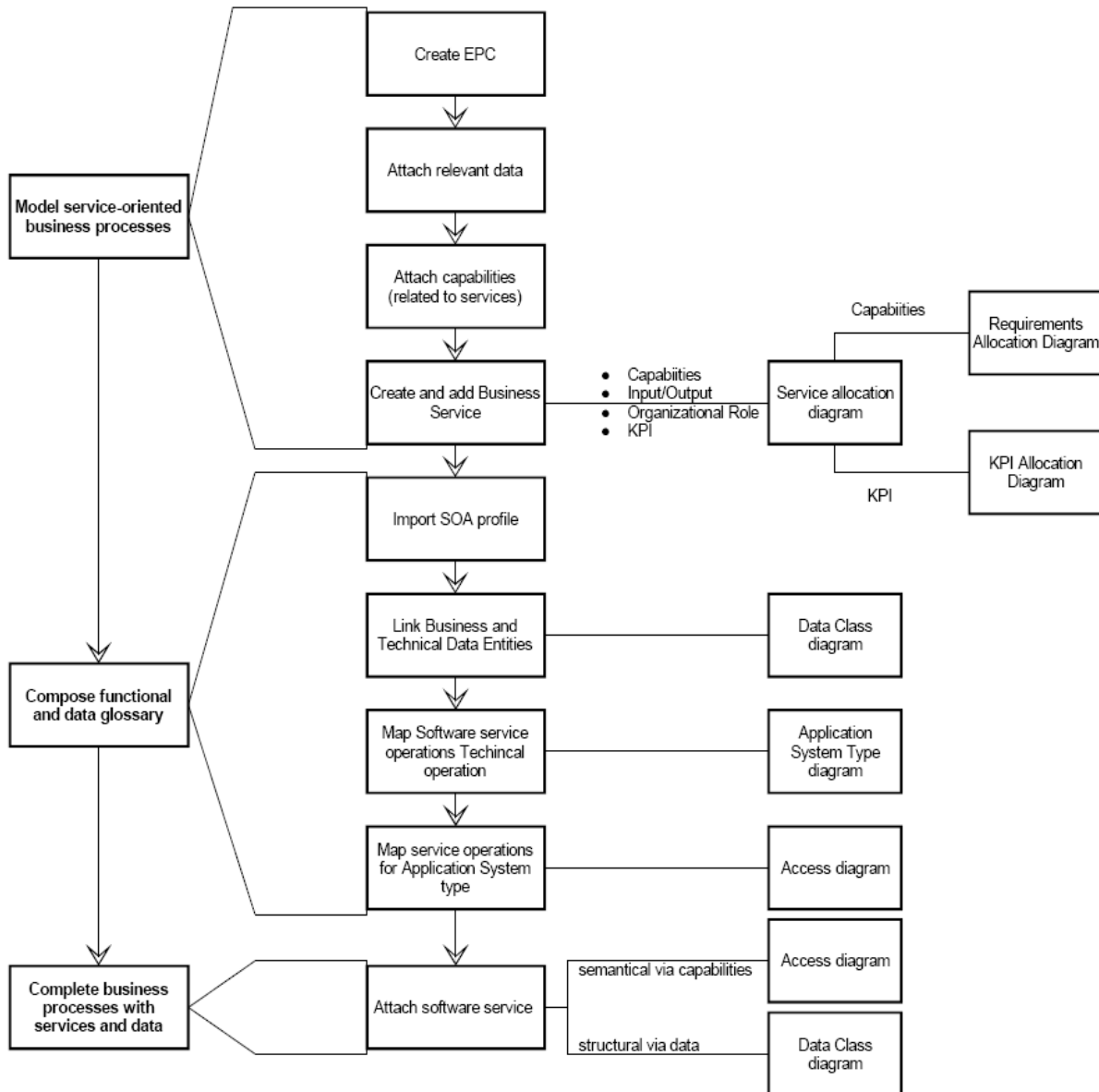


Figure 37 Process Modelling activities Service Oriented EPC and associated diagrams

Appendix B. Cordys Business Process Management Suite

Cordys is a Business Process Management Suite. It has a strong technological platform for SOA development and execution, and on top it provides business process management facilities for service orchestration. Furthermore, it provides an interface for creating mash ups and cloud orchestration.

1 The BPM closed-loop life cycle

Cordys defines four phases in their BPM closed loop cycle. Qualify & Analyze phase qualifies the organization, select a process, determine goals and scope, and analyze the as-is process and the gaps. Requirements gathering, functional scoping, quality aspects assessment, composing a high-level architecture and developing work breakdown structure are activities within this phase. Design & Model is to develop to-be process, sub processes, case activities and rules. Development of solution, software and technical architecture will be the basis for functional and technical design. Develop & Deploy is to implement the processes in existing infrastructure, including application services, integration, transformation and user interactions. Realization and deployment of the designs and documentation are central within this phase. Run & Monitor is to collect process performance information, such efficiency and effectiveness for analysis and optimization. Table 17 summarizes the steps, deliverables and models in each phase.

Phase	Steps	Deliverables	Models and modelling activities
Qualify & Analyze	<ul style="list-style-type: none"> · Determine business drivers and objectives · Define organizational structure · Analyze the as-is process · Perform gap-analysis · Define to-be process blueprint 	<ul style="list-style-type: none"> · Project qualification · Business case 	<ul style="list-style-type: none"> · (Re-)Analyze operational case
Develop & Model	<ul style="list-style-type: none"> · Model or change process · Perform business service discovery & design · Compose user interaction design · Construct system integration design 	<ul style="list-style-type: none"> · High-level to-be Process · Scoping Workshop · Workshop Report · High-Level Scoping 	<ul style="list-style-type: none"> · Create folder, deployment structure · Develop BPM · Add BPM activities
Develop & Deploy	<ul style="list-style-type: none"> · Implementation of web services · Bind services to process model activities · Implementation of data flow · Develop UI components for end-user interaction 	<ul style="list-style-type: none"> · Deployment Plan · Stage Plan · Functional Design · Technical Design · Product Hand Over 	<ul style="list-style-type: none"> · Implement BPM activities · Message Map · Workflow assignment · Validate · Publish
Run & Monitor	<ul style="list-style-type: none"> · Monitor Process instances · Real-time process performance data · Manage service level agreements · Perform trend analysis & bottleneck detection · Evaluate and prepare next iteration 		<ul style="list-style-type: none"> · Run Process Instances · Deploy

Table 17 Cordys BPM: phase, steps, deliverables, and models and modelling activities

1.1 Roles

Cordys have depicted the roles expected in the BPM closed-loop cycle. Cordys depicts best practices for each role in Cordys context can be found. They identify the following:

1. The *business user* is using Cordys. The user needs should be translated into requirements and influence the first two phases. In the end the user will have to work with Cordys.
2. The *business analyst* is amongst others responsible for qualification of the customer and depicting the organizational context. A major task is to analyze and define processes, and derive the conditions for designing processes. The business analyst relates the business objectives to processes and models these.
3. The *architect* is responsible for high-level scoping of the solution, functional design and developing architectures. There are three types of architects: solution architects, software architect and technical architects. Each is responsible for its own architecture.
4. The *developer* is responsible for the development of web components, like web services, business processes, web forms, interfacing to other systems. Developers are software engineers and solution developers creating technical designs, implement designs, software testing and verification, and user documentation.
5. The administrator is responsible for administration of the customer environment. There are different types of administrators including system administrator, technical consultant and technical architect. Responsibilities are maintaining the solution architecture, application landscape and organizational policies. Maintain availability, capacity, flexibility and security of Cordys.
6. The project manger is responsible for project planning, project control, project evaluation and project reporting.

Figure depicts the phases and the roles.

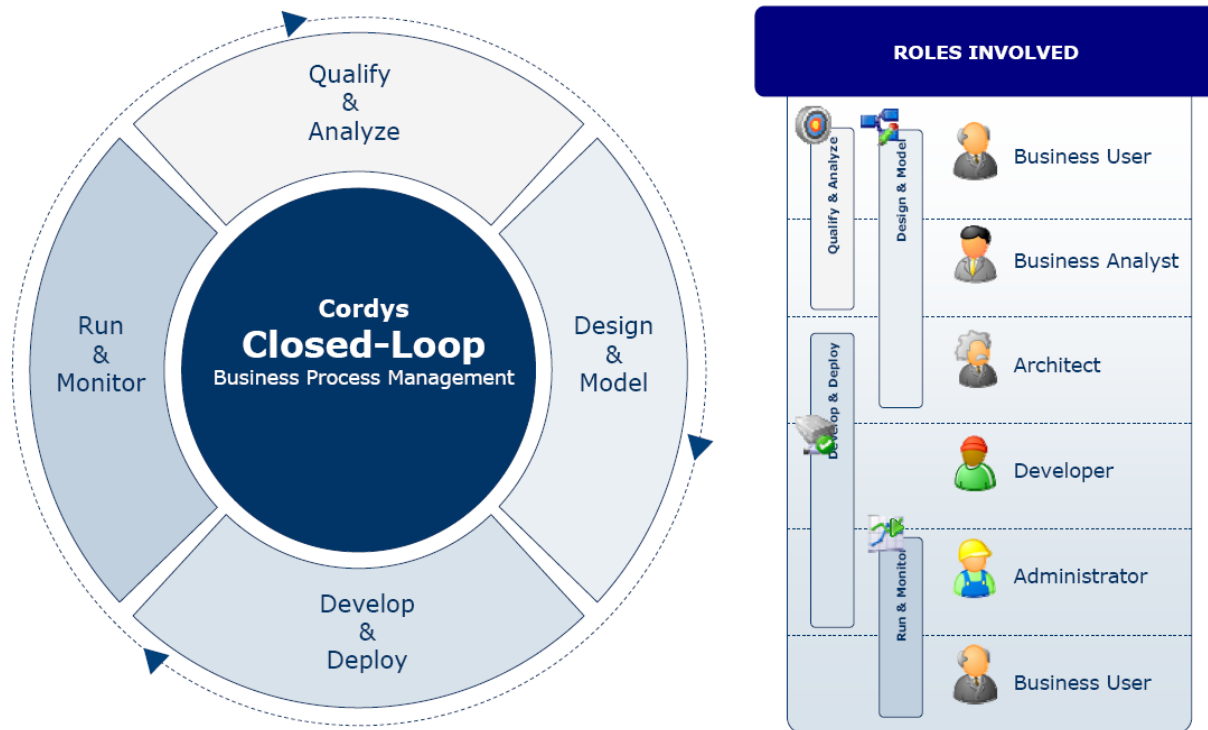


Figure 38 Cordys BPM and roles

2 Cordys Business Process Model (BPM)

Cordys, version BOP-4, provides several types of models including business context model, value chain model, case management model, organization model, and business process model. Within the scope of BPD SOA, only two types are relevant. Regarding the business context the organization model provides insight on the organizational structure of the enterprise. The organizational models are used to determine the assignment of activities to users and teams, and also for escalation management. These models give us a very powerful abstraction to build collaboration features into composite applications. Within the organization model organizational units and roles can be depicted. The constructs used are: Organization unit, Assistant unit, Independent unit, Sub diagram and Role association.

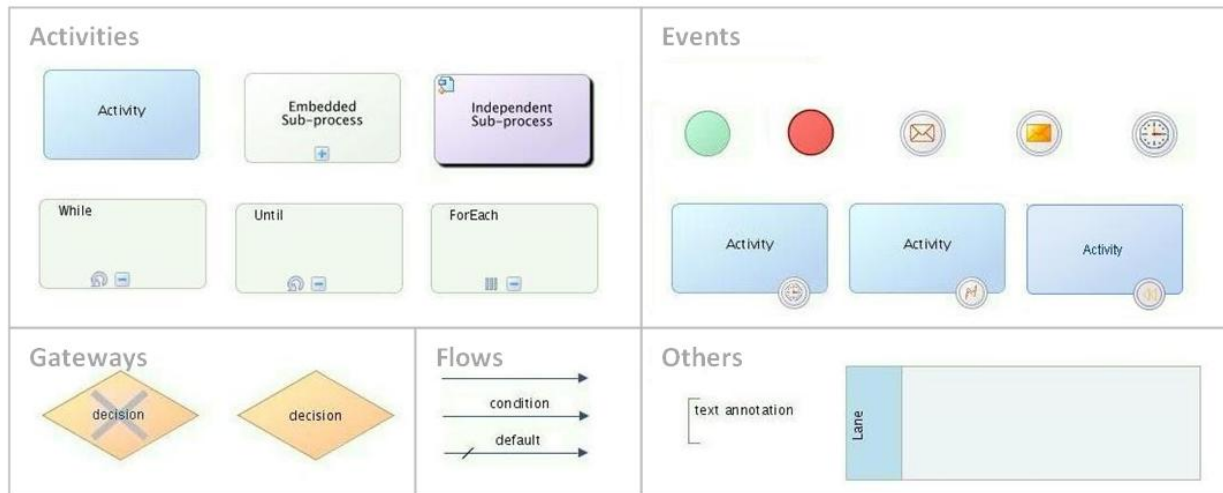


Figure 39 Cordys Business Process Model Constructs

Business process modelling in Cordys is achieved by creating Business Process Models (BPM). BPM is a BPMN based model. The current platform BOP4 supports BPMN2.0. Table depicts a summary of Cordys BPMN 2.0 compliance. Constructs are supported directly or in an alternative way. Alternative ways are implicit or via properties. Furthermore, data flow and workflow assignments need to be set via properties and are not represented by separate constructs. Data flows are set via message maps. Source and target data needs to be specified in Xpath. Workflow assignment is to assign roles to specific tasks. Roles and teams (organizational units) may be assigned to Tasks and Lanes. However, a Task can only be assigned a role or team when the Task is associated with a User Interface.

For a model or task an *activity* can be set. A BPM activity is an interaction component assigned to a task. Web service: the web services can be developed and imported via WSDL URL. User Interfaces are Xforms, and can be developed and associated with tasks.



Figure 40 Cordys BPM properties

Activities

	<i>normal</i>	<i>loop</i>	<i>multiple instance</i>	<i>compensation</i>	<i>ad-hoc</i>	<i>Compensation + ad-hoc</i>
Task (Atomic)	✓	↻	✓	✗		
Collapsed sub process	✓	↻	✓	✗	✗	✗
Expanded sub process	✓	↻	✓	✗	✗	✗
Call Activity	✗					

Events

	<i>Start</i>	<i>Intermediate Catching</i>	<i>Intermediate Throwing</i>	<i>End</i>
None	✓	✗		✓
Message	✓	✓	✓	✓
Timer	✓	✓		
Error	✗	✓		✓
Escalation	✗	✗	✗	✗
Cancel		✗		✓
Compensation	✗	✓	✗	✓
Conditional	↻	✗		
Signal	✗	✗	✗	✗
Multiple	↻	↻	↻	✗
Link		✗	✗	
Parallel multiple	✗	✗		
Terminate				✗

Gateways

Data-based	OR	✓
	OR	✓
Event-based	XOR	↻
Inclusive	XOR	↻
Complex	Conditions	✗
Parallel	AND	↻

Flows

Normal Flow	✓
Uncontrolled Flow	✓
Conditional Flow	↻
Default flow	✓
Exception Flow	✓
Message Flow	✗
Compensation Association	✓

Artefacts & Others

Association	✓
Text Annotation	✓
Group	✗
Pool	↻
Lane	✓
Data object	✗

Table 18 Cordys BPMN 2.0 compliance [Gakkhar 2009]

2.1 Activities of Process Modelling in Cordys

Figure 41 depicts the activities in modelling a Cordys BPMN. Cordys BOP-4 activities are depicted. A difference is made between run-time and design-time in Cordys. Run-time references are executable elements. (The arrows do not depict an order).

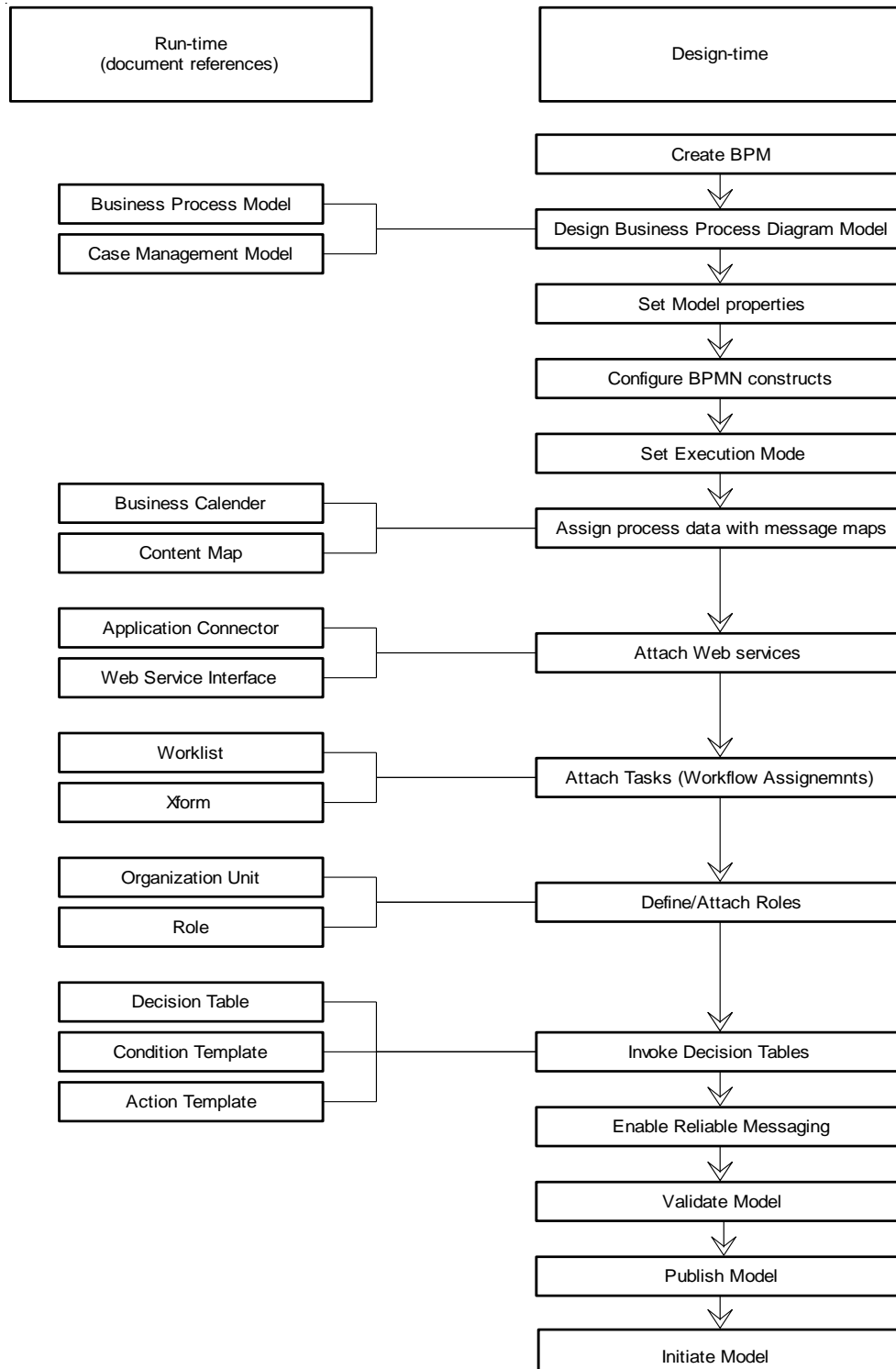


Figure 41 Cordys Process Modelling activities

Appendix C. Roles in BPD SOA

Focusing on business process modelling, the practitioners are critical stakeholders as they have to adopt, design, develop and use the process models to increase efficiency within their organization. Practitioners can be subdivided in different type of stakeholders; the roles within BPD SOA development.

1 Terms

First certain definitions of roles are given, to create a common understanding in terminology.

- **Manager:** Responsible for planning, organizing, directing and controlling an organization.
- **Architect:** visualizes a design by developing the logical and physical layout (structure) of the overall solution and its components, and guides the realization of a design in on technical and administration grounds.
- **Analyst:** skilled at analyzing, detailed deep examining, summarizing of data in specific limited area and propose recommendations.
- **Developer:** develops, works out, certain concept
- **Engineer:** Engineers are concerned with developing economical and safe solutions to practical problems, by applying mathematics and scientific knowledge while considering technical constraints.

2 Roles in BPD SOA according to theory

The roles and responsibilities depict the different people involved and their perspectives in BPD SOA. Two types of roles can be distinguished: Business roles and IT roles [Carter 2007]. The business and IT domain are both addressed in BPD SOA. The business domain identifies roles and responsibilities of the BPM and SOA relates to the IT domain. Identifying specific roles and responsibilities with separations of concerns is required, preventing to force the people making decisions in areas without the proper knowledge or point of view. Below functions within BPD SOA are mapped onto the phases. Functions may consist of one or more roles. The functions and roles are derived from case studies by [Carter 2007], [Bieberstein, et al. 2005] and [Kajko-Mattsson, Lewis and Smith 2008]. For each function several roles and responsibilities are depicted.

- **Business Professional:** The responsibility of a business professional regards business performance, compliance and governance. He is expected to manage business performance and execute strategic and tactical decisions for specific areas of responsibility.
- **Business Analyst:** The business analyst should have insight and understand the business perspectives defined by strategy. The analyzed business is translated into process models for further development.
- **Process Architect:** The process architect is a specialized business analyst, who focuses on the analysis and development of processes in their business environment and interactions. The planning of services which serve certain business capabilities in order to create service oriented business processes. Close interaction with business analyst is necessary. Other common terms are process analyst or process engineer.
- **IT Analyst:** The IT analyst interprets the business analyst requirements for IT, and develops an IT solution. Possible roles are Legacy advisor, Sourcing and using manager, Service-level manager and Security specialist.
- **IT architect:** The IT architect functions as service architect, and as service infrastructure architect. The service architect is responsible for the orchestration of software services, understanding the business requirement for the services. Identifying and structuring of interfaces, software units and service buses and communicating the best practices and trends related to SOA. The service infrastructure architect plans the service infrastructure. A worth mentioning role is integration architect.

- IT developer: The IT developer implements the services and infrastructures according to the IT architectural principles to create "building blocks" for the construction of applications. Roles include Service deployer/developer, Software developer and Interoperability tester.

There are several roles which are implicit in the BPD SOA development process, which means that these roles are present during development, but not explicitly influencing the content. The roles of administrator, customer, user, provider and supplier are implicit within the development cycle. For both Business and IT domain leaders or managers are responsible for delivering the solutions. Furthermore a specific BPM-SOA project is managed by a project manager. The administrator is responsible for the management of the business operations, which in BPD SOA is management and governor of service execution manager, and administrator of the system and databases.

3 The roles comparison ARIS - Cordys BPD SOA

The roles are compared to provide detail on what is expected of a certain role in ARIS and Cordys. The roles distinguished from theory provide the basis for comparison. The essence of this overview is to show what is meant by a role dependent of a platform. The discussion remains - what's in a name?

Perspective	Phase	Role	ARIS Role	Cordys Role
Business	Strategy	Business Professional	-	-
			Enterprise architect	
	Process Design Process Evaluation Process Simulation	Business Analyst	Business analyst	Business Analyst
Process Analyst		Process engineer		
IT	Process Implementation Process Enactment	IT Analyst	IT architect	Architect
		IT Architect		
		IT Developer	Software engineer Integration engineer	Developer
Implicit	Process monitoring	Administrator	-	Administrator
		Project manager	Project manager	Project Manager
		User	-	Business User

Table 19 Roles comparison ARIS and Cordys in BPD SOA

Appendix D. Meta-model analysis EPC & BPMN

1 EPC meta-model

No meta-model is formalized for EPC. Considering that the BPMN meta-model is represented by class diagrams and the models are to be developed in ARIS, an EPC meta-model is created according to the views in the ARIS methodology. Therefore, the constructs relevant for BPD SOA are considered in attempt to create a meta-model for EPC. The constructs are grouped according to the five perspectives in ARIS methodology [Scheer and Schneider 1998].

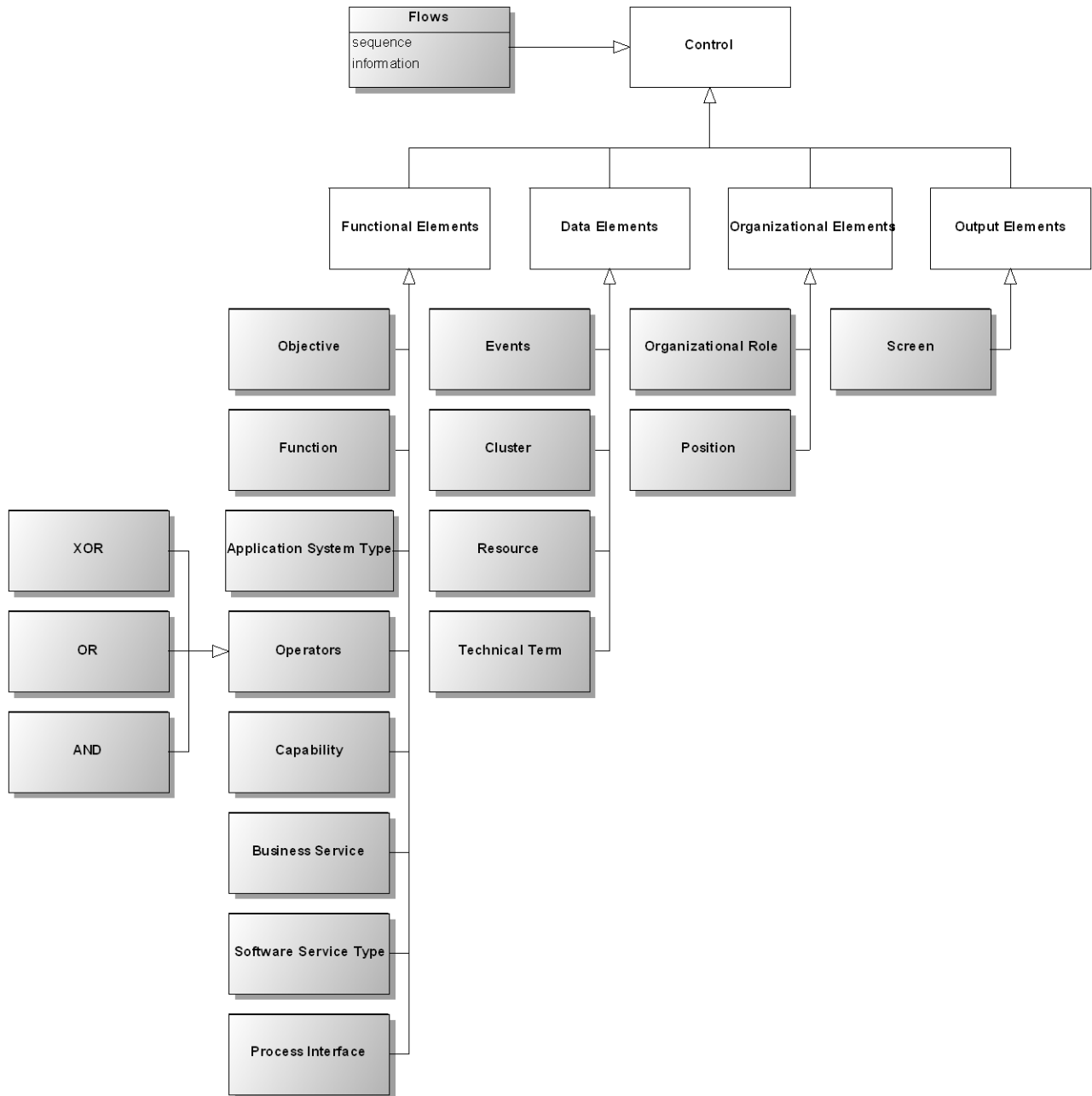


Figure 42 EPC meta-model

2 BPMN meta-model

As Cordys BPM is BPMN based, the meta-model of BPMN is used as basis to derive the meta-model of BPM. BPMN 2.0 formalizes a meta-model in form of class diagrams.

The BPMN meta-model consists of a BPMN core with extensible layers. The core is distinguished in foundation, service and common. Foundation and Service depict the fundamental constructs for semantics and interfacing respectively. Common depicts the classes which are common for the layers. This core is the basis for the layers, which are the BPMN diagrams: Process, Collaboration, Choreography and Conversations. The Process diagram describes a sequence or flow of activities in an organization with an objective. The Choreography diagram depicts the message interactions between participants without process detail. The Collaboration diagram depicts interaction between participants, including the orchestration processes or choreographies. The Conversation diagram depicts a simple communication between two parties, which are not allowed to be pools.

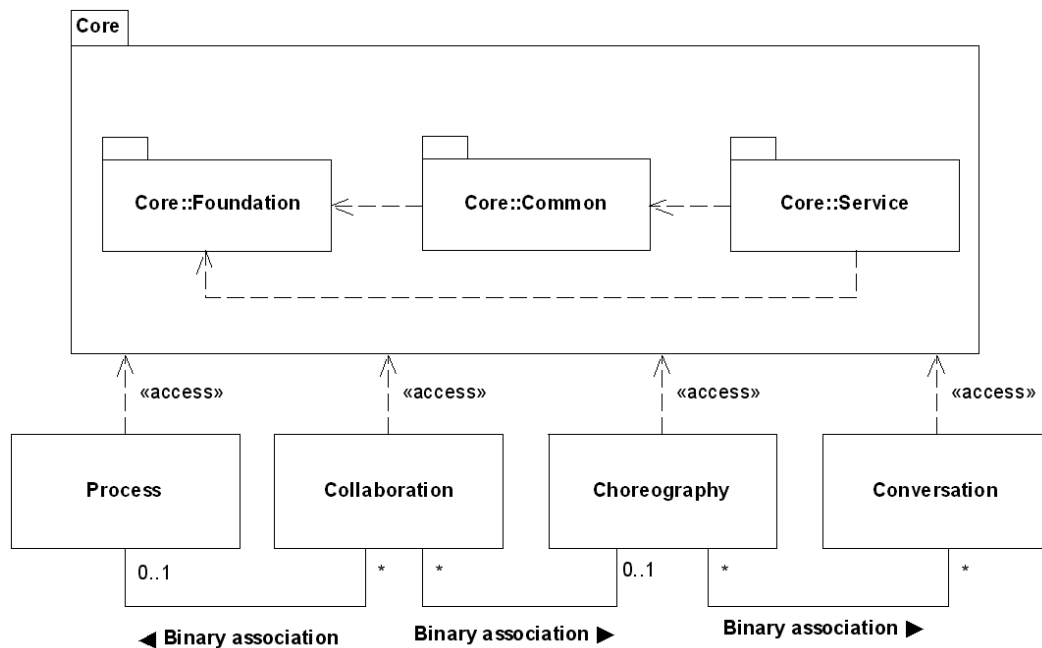


Figure 43 BPMN metmodel core

Within this thesis the meta-model of the Process diagram of BPMN is relevant, as this is the foundation for the Business Process Models of Cordys. The objects used in the Process Diagram are categorized in certain meta-levels, which in UML terms is the inheritance by super classes. In the Foundation the *BaseElement* is the abstract super class for most BPMN elements, defining the id and documentation for each element. The next level depicts the common elements, which are objects present in most diagrams. Figure 43 depicts the elements relevant for Process Diagrams. *FlowElement* is the abstract super class for elements which can appear in the Process flow, which inherits *FlowNode* and *SequenceFlows*. *FlowNode* relation to *SequenceFlow* depicts that only some nodes can use sequence flows. Subclasses of the *flowNode* are *data objects*, *activity*, *events* and *gateway* objects. A process diagram has as super class *FlowElementContainer*: a container of all elements which can be used in processes. Next to the *flowElement*, it has an association with artifacts: *groups*, *associations* and *text annotations*. Artifacts are elements which are not directly related with process, providing additional information. Indirectly via associations the process diagram is related with *lanes*, *participants* and *messageFlows*. These are elements more related to the other

diagrams, specified in the *interactionSpecification*, which mainly depict the interactions between participants. Figure 44 summarizes the structure of the BPMN meta-model.

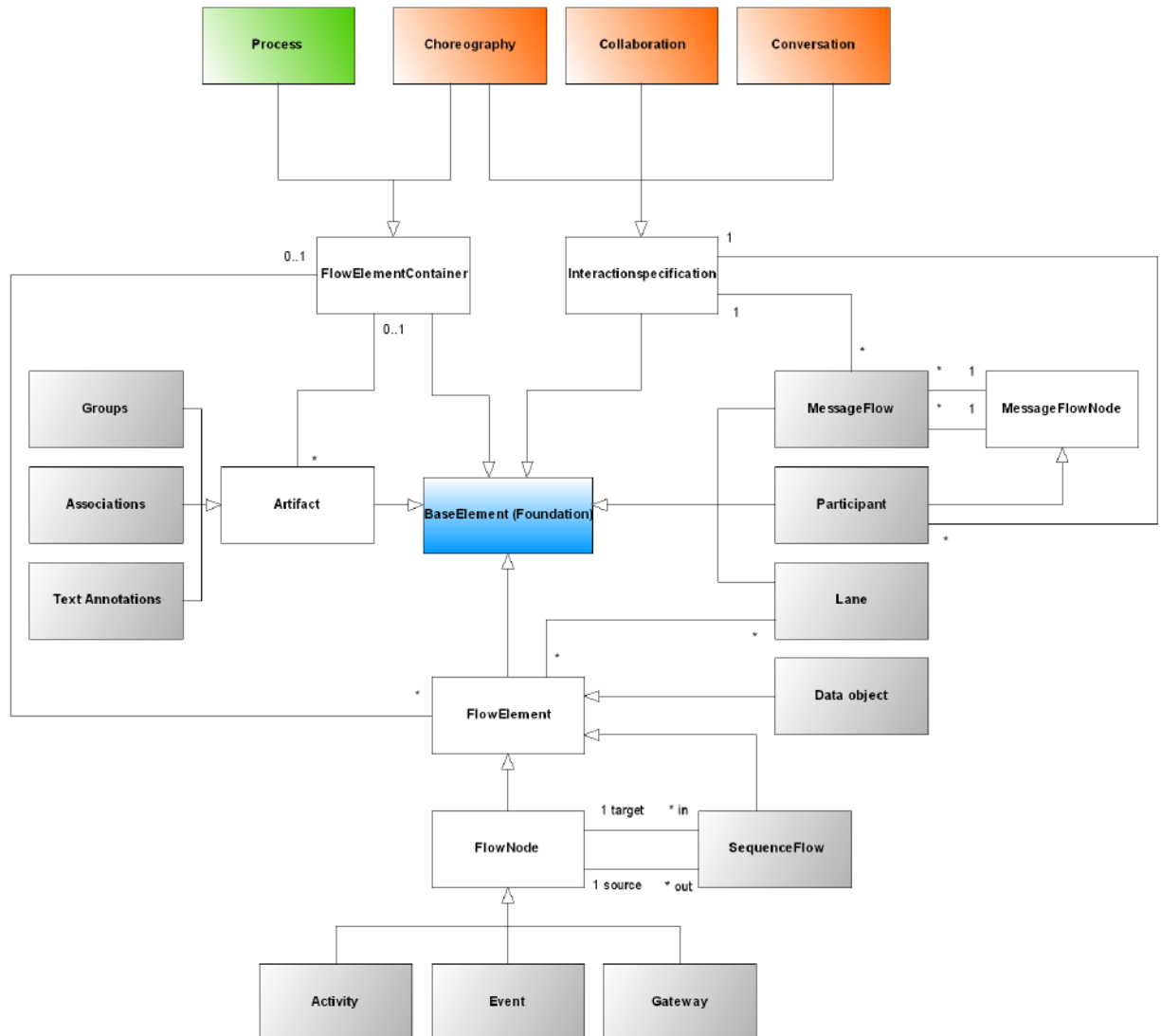


Figure 44 BPMN meta-model foundation

In Figure 44 all relevant *Process* (green) elements are displayed shaded in grey. For *SequenceFlow*, *Gateway*, *Activity* and *Event* subclasses exist. Note that an alternative approach is chosen here to display certain properties of elements. In BPMN meta-model these are defined via other abstract classes. However, in order to maintain the visibility of all relevant aspects, I will use attribute properties of classes to depict certain associations or relations.

Sequence flows can be associated with *Expressions*, which depict certain conditions for flow. The *SequenceFlow* class' attributes specify different types of flows. Uncontrolled flows do not have to satisfy any conditions. As antonym the conditional flows exist. For this flow conditions must be true before a token is passed through. Furthermore, a default flow is one which is taken when other alternative flows from activities or gateways are invalid.

Gateway is a super class for several types of gateways, which each has his own behaviour. The Exclusive Gateway depicts the OR operator, the Inclusive gateway depicts the XOR, and the Parallel gateway depicts the AND. Complex Gateway specifies complex synchronization with the use of conditions. Eventbased gateways use events as

conditions instead of expressions. The flow chosen is based on FIFO. Furthermore, Eventbased can be exclusive or parallel.

Activity has three types of classes: sub process, tasks, and call activity. Attributes depict the markers of the activities, which are actually loop characteristics which are associated with activities. There are three type of sub processes. And seven types of tasks.

Events can be either catch or throw events. The markers on the events are defined in EventDefintions for Catch (Start and Intermediate) and Throw (End and Intermediate). The EventDefinitions specify the type of trigger for a certain event. The EventDefinitions of BPMN are left out of consideration for clarity and displayed as attributes of classes. Furthermore, the distinction between interrupting and non interrupting events is not displayed.

The other elements are groups, associations, text annotations, message flows, messages, pools, lanes and data objects.

3 Cordys BPM meta-model

From the BPMN meta-model the BPM meta-model can be derived. Table 18 specifies the subset of constructs which is supported by Cordys BPM. This is used to derive the meta-model for Cordys BPM, and the basis for meta-level mapping. The red coloured are not available in Cordys.

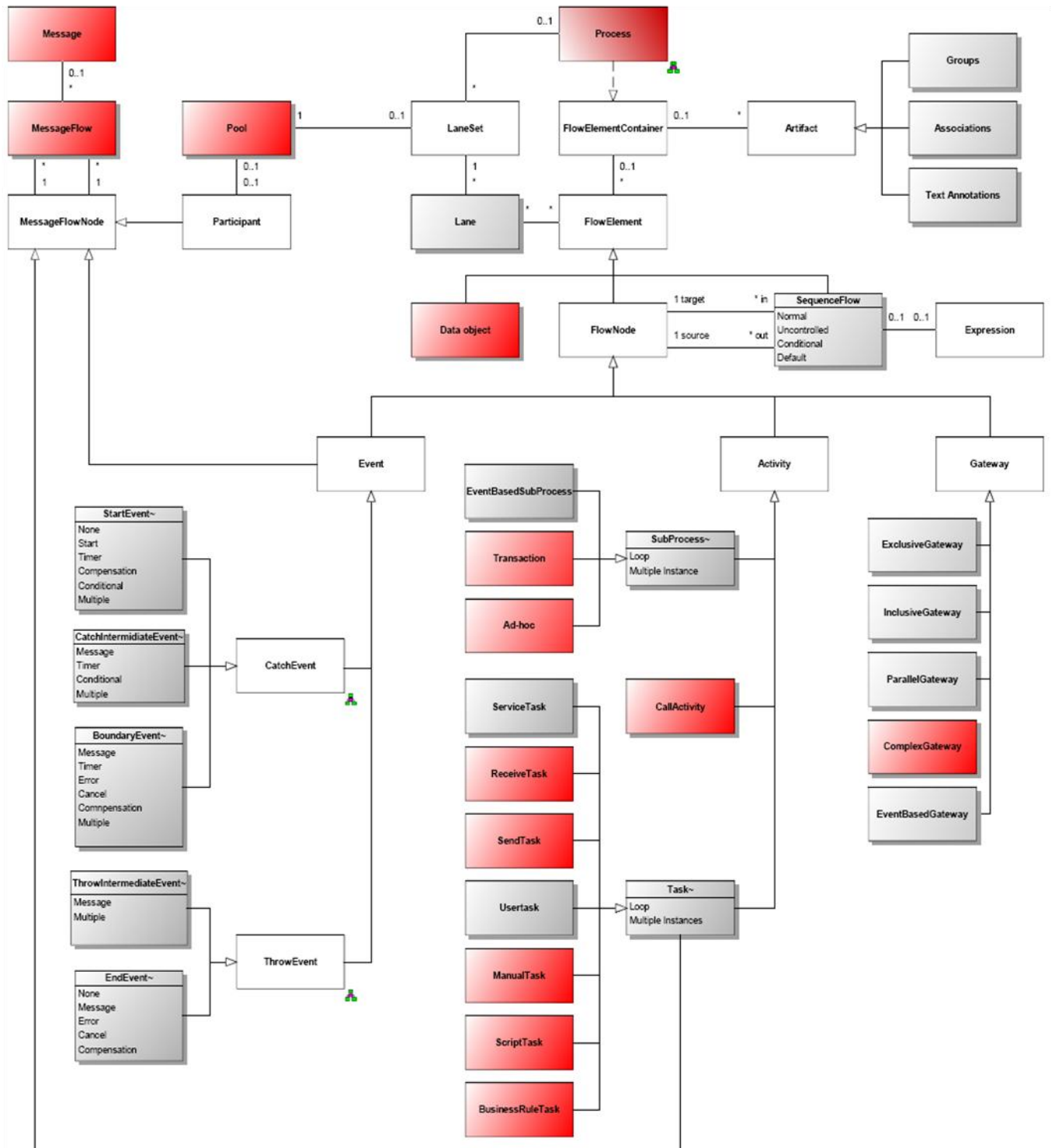


Figure 45 Cordys BPMN meta-model

Appendix E. Evaluation of BPMN and EPC as modelling languages

1 Evaluation BPMN and EPC on BWW

	BWW	BPMN	EPC
Things including properties and types of thing	Thing	Pool, Lane	Position, Organizational unit
	Property	~	~
	Class	Lane, Data Object	
	Kind	Lane	
States assumed by things	State	-	Trigger Events
	Conceivable state space	-	-
	Lawful state space	-	-
	State law	-	E-F-E
	Stable state	-	End event
	Unstable state	-	
	History	-	-
Events and transformations occurring on things	Event	{all events}	E-F-E
	Conceivable event space	-	
	Lawful event space	-	
	External event	Event{start, intermediate, end, message, timer, error, cancel, compensation}	Start Event
	Internal event	Event {start, Intermediate, end, message, error, cancel, compensation, terminate}	E-F-E
	Well-defined event	End, compensation event	E-F-E
	Poorly defined event	Event {start, intermediate, message, timer, error, cancel, terminate}	
	Transformation	{all activities}	Function
	Lawful transformation Stability/correctiveness	{All sequence flows}, Rule, Conditional flow/ Exception Task, Compensation activity	E-F-E
	Acts on	Message Flow	
	Coupling	Message Flow	Relation
Systems structural around things	System	Pool, Lane	
	System composition	Pool, Lane	
	System environment	Pool, Lane	
	System structure	-	
	Subsystem	Pool, Lane	
	System decomposition	Pool, Lane	
	Level structure	Pool, Lane	

Table 20 Bunge-Wand-Weber EPC-BPMN comparison

Table 20 is a combination results from [Green and Rosemann 1999] [Recker, Wohed and Rosemann 2006]

2 Workflow control patterns EPC and BPMN:

Pattern	BPMN	EPC
<u>Sequence</u>	+	+
<u>Parallel Split</u>	+	+
<u>Synchronization</u>	+	+
<u>Exclusive Choice</u>	+	+
<u>Simple Merge</u>	+	+
<u>Multi-Choice</u>	+	+
<u>Structured Synchronizing Merge</u>	+	+
<u>Multi-Merge</u>	+	-
<u>Structured Discriminator</u>	+/-	-
<u>Arbitrary Cycles</u>	+	+
<u>Implicit Termination</u>	+	+
<u>Multiple Instances without Synchronization</u>	+	-
<u>Multiple Instances with a Priori Design-Time Knowledge</u>	+	-
<u>Multiple Instances with a Priori Run-Time Knowledge</u>	+	-
<u>Multiple Instances without a Priori Run-Time Knowledge</u>	-	-
<u>Deferred Choice</u>	+	-
<u>Interleaved Parallel Routing</u>	-	-
<u>Milestone</u>	-	-
<u>Cancel Activity</u>	+	-
<u>Cancel Case</u>	+	-
<u>Structured Loop</u>	+	-
<u>Recursion</u>	-	-
<u>Transient Trigger</u>	-	-
<u>Persistent Trigger</u>	+	+/-
<u>Cancel Region</u>	+/-	-
<u>Cancel Multiple Instance Activity</u>	+	-
<u>Complete Multiple Instance Activity</u>	-	-
<u>Blocking Discriminator</u>	+/-	-
<u>Cancelling Discriminator</u>	+	-
<u>Structured N-out-of-M Join</u>	+/-	-
<u>Blocking N-out-of-M Join</u>	+/-	-
<u>Cancelling N-out-of-M Join</u>	+/-	-
<u>Generalised AND-Join</u>	+	+/-
<u>Static Partial Join for Multiple Instances</u>	+/-	-
<u>Cancelling Partial Join for Multiple Instances</u>	+/-	-
<u>Dynamic Partial Join for Multiple Instances</u>	-	-
<u>Acyclic Synchronizing Merge</u>	-	+

Pattern	BPMN	EPC
<u>General Synchronizing Merge</u>	-	-
<u>Critical Section</u>	-	-
<u>Interleaved Routing</u>	+/-	-
<u>Thread Merge</u>	+	-
<u>Thread Split</u>	+	-
<u>Explicit Termination</u>	+	-

Table 21 Workflow Patterns EPC-BPMN comparison [van der Aalst, et al. 2003]

Appendix F. Assessment of non-main EPC and C-BPMN objects

The information for the assessments is retrieved from vendor specific documentation.

Web services	ARIS concept	Cordys concept
Constructs	Software service component	Web service
Importing WSDL	<ol style="list-style-type: none"> 1. Select folder to import services description 2. Select WSDL file 3. Create name for Business view on service 4. Create name for IT view 5. Create additional information for imported service 6. Assign capabilities to the imported service 7. An access diagram is created with the assigned capabilities 8. An UML component diagram is created with the WSDL structure 	<ol style="list-style-type: none"> 1. Setup UDDI Registry Connection 2. Provide Inquiry URL 3. Using WSDL URL to generate Web Service
Assessment	ARIS software service components depict a service operation. This is on the same granularity level as web services in Cordys. ARIS relates the web service to capabilities defined on business level. This is not supported by Cordys. Cordys allows for development of web services. The services are attached to functions/tasks.	

Table 22 Assessment Web service mapping

Data flow	ARIS concept	Cordys concept
Constructs	Data Clusters	Xpath
Description	Data clusters are logical elements which should be linked with their technical equivalents on class level.	Xpath is the XML Path from which the data to be appearing in the form control can be selected. Xpath is set automatically for all transactional forms. For non-transactional forms, the programmers have to set the path manually at times.
Assessment	The format of the input and output is specified on different abstraction levels. ARIS allows specification of logical data clusters. These are linked to technical equivalents via other models. The format however does not match either, as Cordys requires Xpath expressions. The ARIS technical equivalents of data clusters can indicate the input and output needed on the right abstraction level. Conversion of the technical data into the correct Xpath expression is required for execution in Cordys.	

Table 23 Assessment Data flow mapping

UI	ARIS concept	Cordys concept
Constructs	UI Types	Xforms
Description	ARIS facilitates model based UI design with screens. UI data is related UML classes. Types are screens, text boxes, and text panels. Other attributes, such category, default values, etc. can be set.	Cordys XForms is part of Cordys Studio. It is a rapid application development environment for building UI's. XForms can generate UI's based on Web services with their WSDL and XSD.
Assessment	Transforming UIs designed in ARIS to Cordys and vice versa requires a specific implementation. The method of developing of UIs is quite different.	

Table 24 Assessment UI mapping

Role	ARIS concept	Cordys concept
Constructs	Organizational Unit – Position	1. Organization Unit (team) - Role
Description	Organizational charts depict the structure of an organization. All organizational elements are used in process models.	Organizational models depict the organization structure. The roles and teams can be linked to tasks.
Assessment	On high-level organizational units and roles can be mapped. However, an ARIS role does not have any runtime associations, unlike on Cordys executable level. A mapping can be made between an authorizing organizational unit or position in ARIS, to Cordys team or role respectively. Both ARIS and Cordys support modelling organizational models. Exchanging these would require a separate exchange format.	

Table 25 Assessment Role mapping

KPI	ARIS concept	Cordys concept
Constructs	KPI	KPI (not in process model)
Description	KPI tree: A tree of KPIs organized in individual, technically related KPI groups (e.g. Quality, Cost, Time, and also process-oriented groupings). It should be noted that there may be more than one key performance indicator tree. It is also possible for KPIs to be assigned to several KPI trees according to the target group. KPI allocation diagram: defines the measurement points and the data at the measurement points that is to be included for the purposes of the KPI. Linking multiple KPIs to a new KPI is also described in the KPI allocation diagram.	KPIs can be created using KPI Modeller. The KPI Modeller is a platform that enables you to define a KPI and the various associated measures that will enable monitoring and capturing information about different aspects of the process or external data source as against business objectives, based on which it triggers various business actions if the specified conditions are met. You can create dashboards to view monitored results for the KPI that will be available as trends. KPI trend is shown in dashboard based on each schedule run of the KPI and monitored results captured based on it. A Business Measure is the logical expression defining what to measure.
Assessment	KPI models can be made in both ARIS and Cordys. Exchanging would require a separate format. KPIs can also be linked to process models in ARIS. KPIs are useful for defining process instances during monitoring. The KPI measure points can be considered as input for Cordys Business Measures. Business measures are logical expressions of the measurements. KPI (and measures) should be transformed to annotations, as no C-BPMN construct exists.	

Table 26 Assessment KPI mapping

Risk & Control	ARIS concept	Cordys concept
Constructs	Risks / Control(function type)	-
Description	This will appear in a function allocation diagram. Risks, controls and control test definitions are related and depicted in a business control diagram. Risk and Controls can be modelled and linked to EPC functions. Controls are a type of function in EPC. A control is a group of activities which is executed as part of or intervening a function in a process.	No explicit support for managing governance, risk and compliance in process models. GRC is considered as a management approach which is facilitated through the development process.
Assessment	BPMN does not have construct representation for control or risks. The control is type of function in EPC. A control is a set of activities which need to be done. This could be mapped to a sub process in C-BPMN. Risks can be converted to annotations. Risk & Control diagrams of ARIS cannot be exchanged, as no equivalent diagrams exist in Cordys.	

Table 27 Assessment Risk & Control mapping

Appendix G. Structural Design Patterns for EPC and BPMN

Horizontal transformation is between Event-Driven Process Chain and Cordys BPMN, the design patterns are presented in this Appendix.

1 Design patterns

Design patterns capture concepts represented in a source modelling language, which might be semantically impossible in the target language. An alternative representation is often possible. Thus mapping of patterns depicting the same concept is required to provide complete transformation. The design patterns are based on workflow patterns. There are three basic patterns always present in end-to-end processes. These are the *start event* pattern, *end event* pattern and *sequence path* pattern. A sequence path is a path in the process with successive activities. Figure 46 depicts these three patterns. Furthermore, decision patterns also occur very often. Combinations of the split/ join paths are defined as decision patterns. Three types of decision patterns exist: *parallel*, *alternative*, and *multiple alternatives*. Parallel pattern is a combination of the parallel split and synchronize workflow pattern, depicting a parallel split and join within a process. Alternative pattern is the combination of exclusive choice and simple merge workflow pattern, depicting the split and join with an exclusive alternative. Multiple alternatives is the combination of multi-choice and synchronizing merge workflow pattern, depicting splitting and joining multiple alternative branches.

Considering forward engineering, transformation patterns will be extensively explored from EPC to BPMN. EPC is a language with less design freedom compared to BPMN. From there the situation for BPMN to EPC transformation will be considered, as round-trip is a desire with ARIS-Cordys integration.

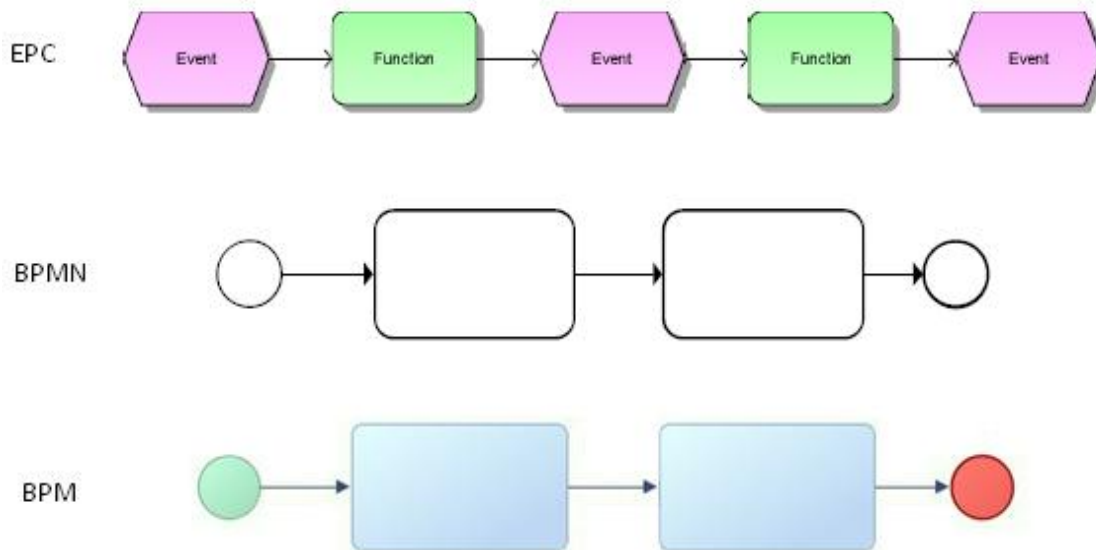


Figure 46 Basic patterns: start event, sequence path and end event

Translation from EPC to BPMN can be achieved according meta-level mapping (functions are translated, intermediate events are not to be mapped). Considering mapping from BPMN to EPC, only translating tasks into functions will not be sufficient, as semantically this will be incorrect in EPC. Therefore a rule for mapping BPMN to EPC in activities in paths is as follows:

- A task will be translated into an event-function, where the event is the trigger for the function, if and only if a task does not subsume a start event.

Considering EPC split operators there are only a few cases which are semantically correct. Functions may be succeeded by all operators, and Events can only be succeeded by AND-operators, as events cannot make decisions. A few scenarios can occur based on these rules.

Parallel split and join in EPC is depicted with the AND operator. A bidirectional mapping exists of the AND operator to the Parallel gateway. In EPC events and functions always alternate each other, disregarding the operators. Figure 47 depicts three translation scenarios of EPC to explicit BPMN with gateways to the BPM with their BPMN compliance in Cordys. Scenario a) depicts an AND-split after a function, and AND-join. The branches contain an event followed by a function. The events after an AND-split do not depict a state which determines the flow, and can be considered as intermediate events, thus not to be translated in BPMN. Scenario b) and c) depict an AND-split after an event. The same rules apply as in scenario a):

- In a parallel decision pattern from EPC to BPMN meta-level and basic pattern mapping applies.

Note that split and joins in parallel are modelled implicitly in BPMN.

Transformation from BPMN to EPC regarding the parallel pattern first requires explicit representation of in BPMN, before the meta-level and basic pattern mapping rules can apply.

- In a parallel decision pattern from BPMN to EPC, meta-level and basic pattern mapping applies, with one exception:
 - a. Each task after the parallel split in the parallel decision should be transformed to an event – function, if and only if the AND-split is preceded by a function.

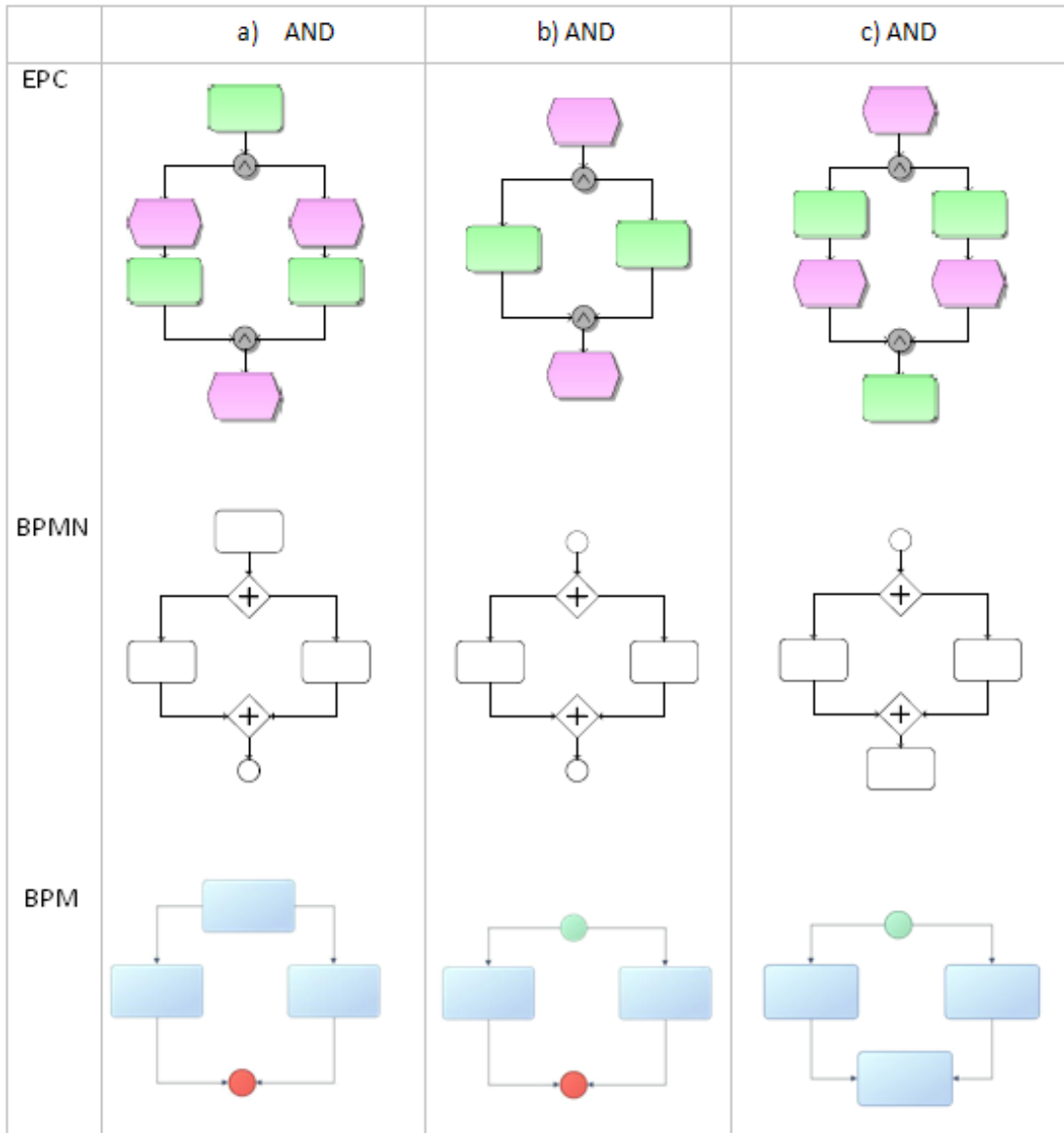


Figure 47 Decision Parallel Patterns

Alternative patterns are depicted in Figure 48 d), e) and f). The alternative pattern is a XOR branch and merge, where only one branch can be chosen. After a XOR-split the event on a branch represents a state which triggers the following function, thus depicting the condition determining the course of the flow. The XOR operators are mapped to Exclusive gateways. Scenarios e) and f) show that an event following the XOR operator can be considered as conditions on the BPMN exclusive gateway. An exclusive gateway expects conditions on which it should act. The other intermediate events have no influence and do not appear in BPMN models. Scenario e) has a branch with only an event and no function. However there is an alternative which has an active handling influencing the process. Thus, the event can be considered as a condition. Scenario d) is an exception and depicts a split with branches where no activities occur. This branching/merge only depicts two possible states following the function; however, they do not influence the process flow. As BPMN does not represent any states, this branch/merge is to be ignored. Note that merging is done implicit in all cases in BPM.

- In an exclusive decision pattern from EPC to BPMN, meta-level mapping and basic pattern applies, and:
 - a. Events may be considered as conditions following an exclusive gateway, if and only if the alternative branch contains an activity (a function).

For the BPMN to EPC transformation, before the meta-level and basic pattern mapping rules can apply after explicitly modelling the BPMN exclusive pattern.

- In a exclusive decision pattern from BPMN to EPC, meta-level and basic pattern mapping applies, with and:
 - a. Conditions of the exclusive split are trigger events, and replace the event of the event-function translation after an exclusive split.

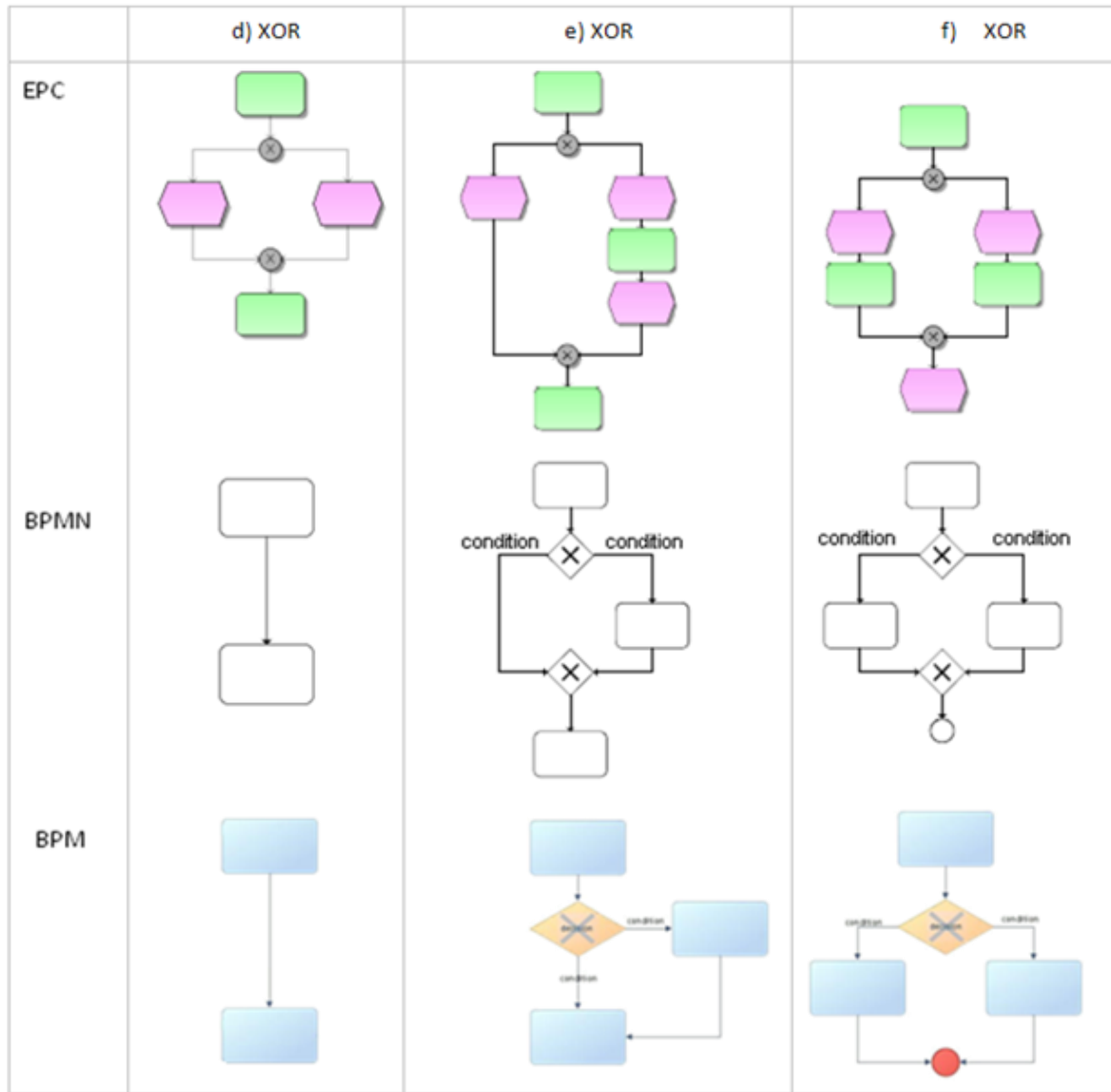


Figure 48 Decision Alternative Patterns

The rule regarding events following an XOR operator also applies for the OR operator as shown in Figure 49 g), h) and i).

- In a inclusive decision pattern from EPC to BPMN, meta-level mapping applies, and:
 - a. Events may be considered as conditions following an inclusive gateway, if and only if the alternative branch contains an activity (a function).

Note that merging is done implicit in all cases in C-BPMN. The branch/merge may be ignored in g) as both alternative branches do not contain any activities.

For BPMN to EPC translation, the same applies as at an exclusive decision.

- In a inclusive decision pattern from BPMN to EPC, meta-level and basic pattern mapping applies, with and:
 - a. Conditions of the inclusive split are trigger events, and replace the event of the event-function translation after an inclusive split.

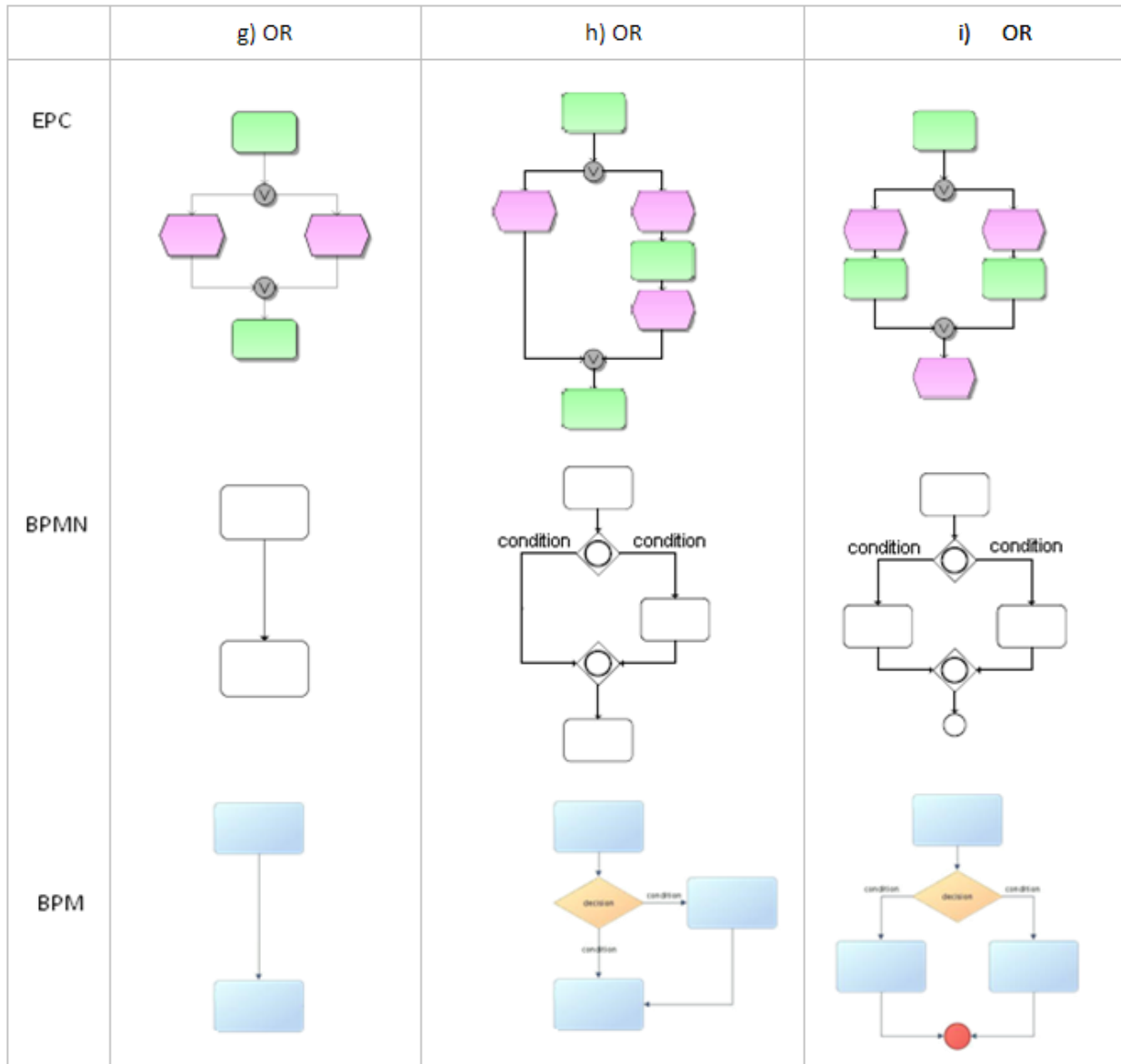


Figure 49 Decisions Multiple Alternative Patterns

Figure 50 j), k) and l) depict combination of some of the design patterns described above. Scenario j) depicts a nested decision pattern in a decision pattern. In such cases the flow should be decomposed in components, which are patterns. Scenario j) depicts start event, end event, sequence path, and a nested multiple alternative pattern in the parallel pattern. These components are easy to distinguish and can be mapped and put together. Scenario k) depicts a situation where a nested decision patterns overlap. One branch of the multiple alternative pattern joins the branch of the parallel pattern. And this joined branch is merging with the other multiple alternative branch. The patterns as above cannot be applied easily. The same holds for scenario l), depicting a nested decision of which one branch could end, loop back, or eventually join. The loop back must be captured by an operator in EPC, as functions

are only allowed on incoming and outgoing flow. In BPMN the loop back can either be captured by the gateway or implicitly joined at the task. An alternation of the definitions for the decision patterns above can capture these structures [Murzek, Kramler and Michlmayr 2006].

- Each decision pattern (parallel, alternative or multiple alternatives) results in a termination, common merge or predecessor.

The implicit joins with overlapping decision patterns cause loss of structure. Scenario i) depicts this. One cannot derive the specific AND-join. Alternatives can be suggested to still capture these structures, for instance empty activities in Cordys.

For BPMN to EPC transformation of the BPM scenarios in h), i) and j) most important is that the models are transformed to explicit BPMN models. An explicit model can be translated according to the meta-level, basic and decision pattern rules.

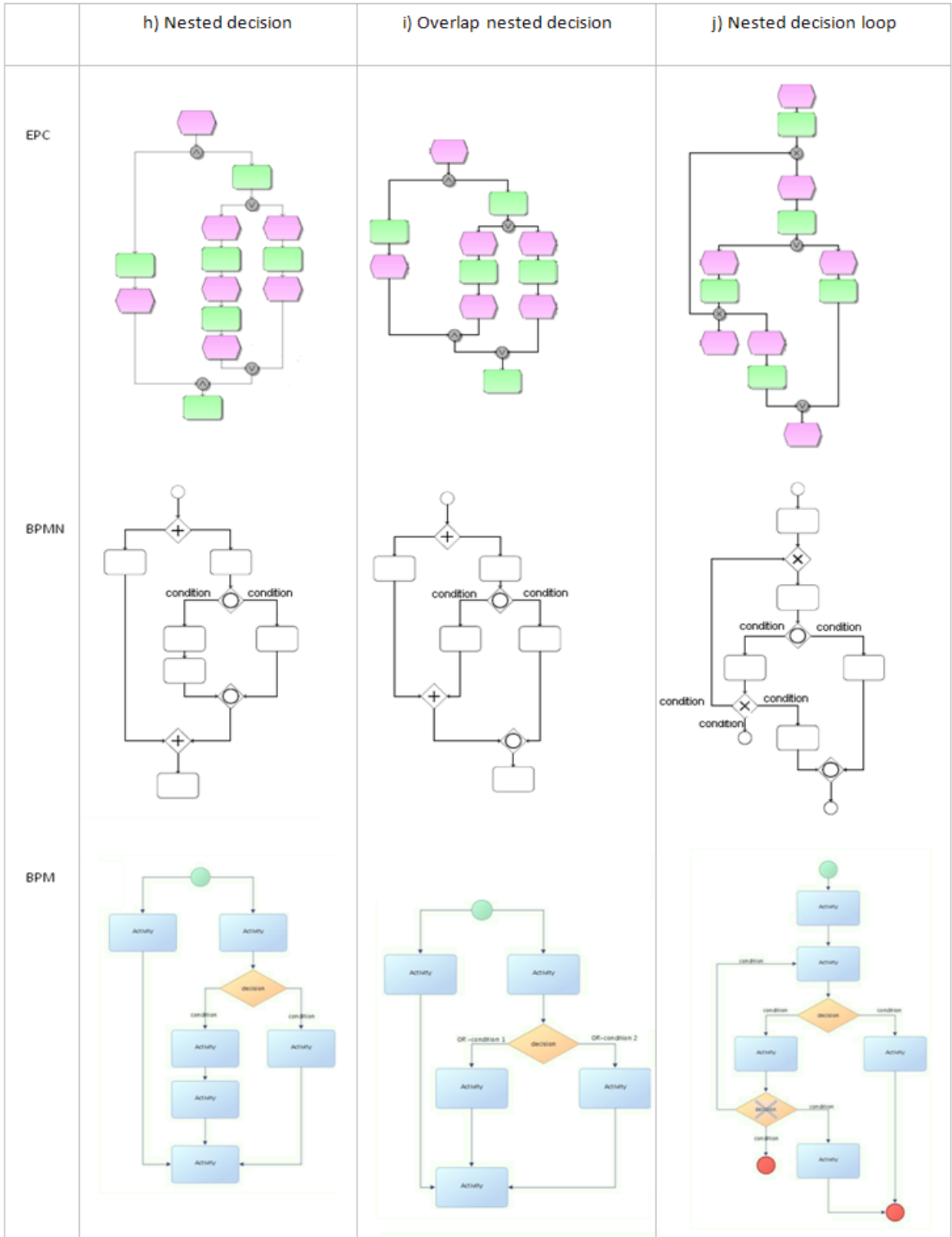


Figure 50 Combination of Decision Patterns

1.1 Multiple operators

In EPC another unique situation which might occur is a decision with XOR/OR succeeded by another decision XOR/OR without intermediate events. The rule specified above about events are mapped to conditions would not apply as no event is specified. Figure 51 captures this situation. Note that with an AND-operator this is irrelevant, as no conditions have to be derived. Here direct translation of operators can be applied. One consideration can be made: when AND-split follows AND-split, to merge the operators, and inherit the branches.

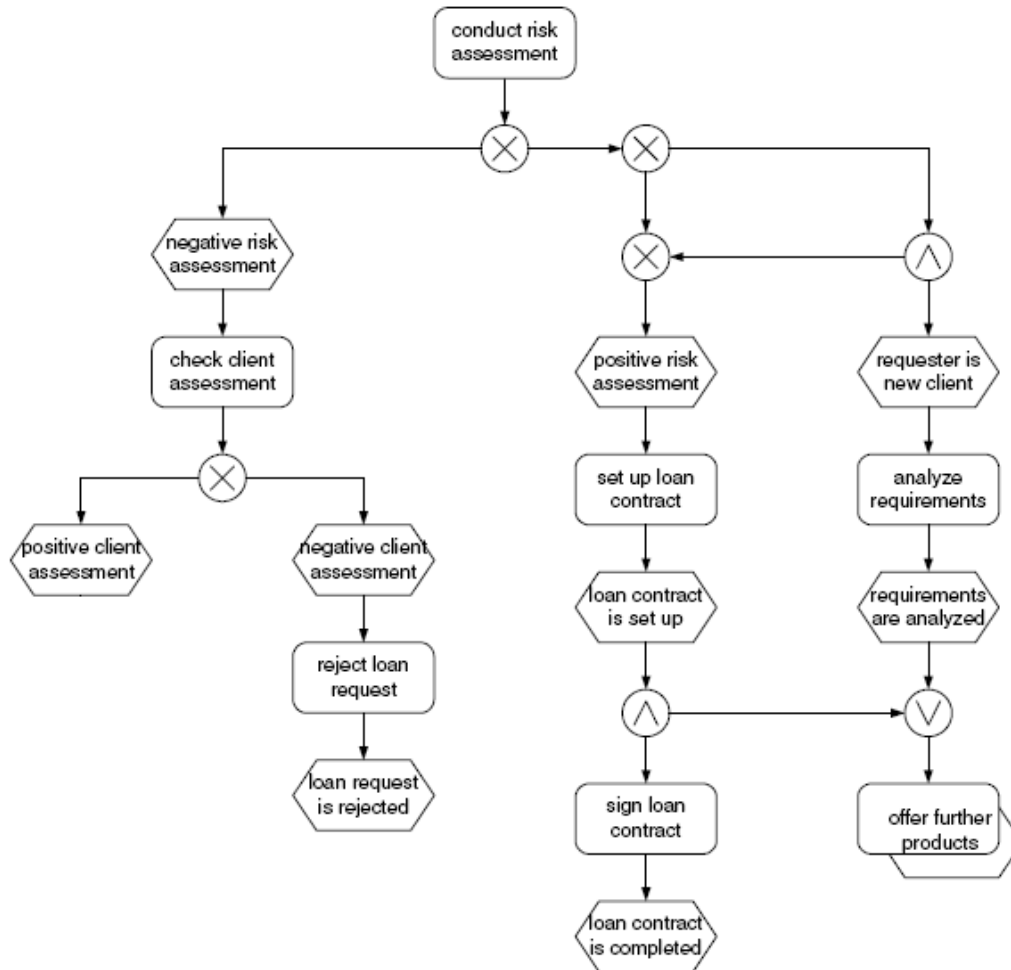


Figure 51 Example Operator-Operator in Decision pattern [J. Mendling 2009]

An XOR/OR operator is preceded by a function, as a decision has to be made. The following event would specify the condition which triggers a specific branch. When however the following event is another operator the condition for the branch is hard to determine. Several scenarios are possible for the second operator with two successive operators:

- Operator is a split:
 - The events succeeding the second split operator should be combined with the second operator as operator, to serve as condition on the branch after the first operator. I.e. a XOR split is followed by an OR split branching dependent on trigger event A and event B, the condition for the branch with XOR operator will become event A OR event B.
 - Option: If the first operator is an XOR and the branch is the only alternative without condition, the condition could be default.
- Operator is a join: the event following the join should be considered as condition.

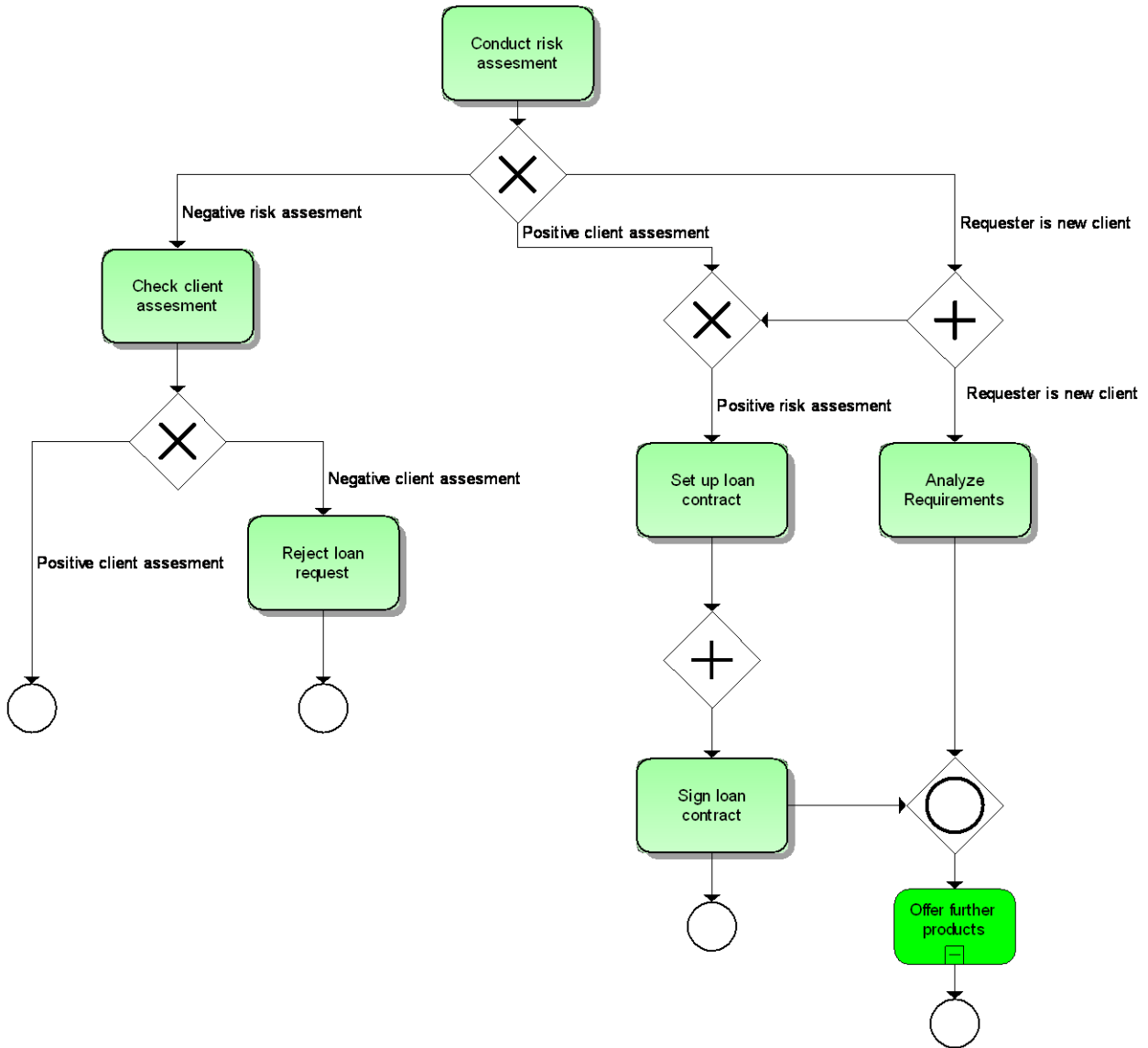


Figure 52 BPMN translated example Figure 51

For a BPMN to EPC transformation each condition after an exclusive or inclusive gateway should be translated to an event. But with two successive gateway the following holds:

- The conditions for each branch is added up to the next branch, which is should be considered as event trigger of that branch, or replace the event in the event-function following split.

1.2 BPMN workflow variants

As mentioned above, decision patterns in BPMN need to be modelled explicitly, before mapping to EPC. Research on workflow patterns support of EPC and BPMN is done extensively. Comparing the patterns shows which concepts can be mapped. BPMN allows for graphical variations depicting certain workflow patterns. Considering the basic control-flow patterns related to decisions, there are explicit and implicit alternatives representing one workflow pattern. For example, a parallel split can be represented by an AND-gateway and implicitly without a gateway. An alternative is representing it with a sub task nesting the two tasks which should execute in parallel. Figure 53 depicts several variations of representing the workflow patterns for parallel split, synchronization, exclusive choice, merge and multiple choice. Detail can be found in [Wohed, et al. 2006]. The implicit variations may not be translated correctly

with only meta-level mapping. The pattern for parallel split can be mapped to EPC. But only the explicit form with gateway will result correctly considering the constructs. By turning the implicit or alternative forms explicit, mapping to EPC is much more evident. In a pattern base approach the mapping between such patterns are noted. Variations of one concept must be appointed, and concepts can be mapped. The combination of the workflow patterns are the basis of the design pattern, thus design patterns can be constructed.

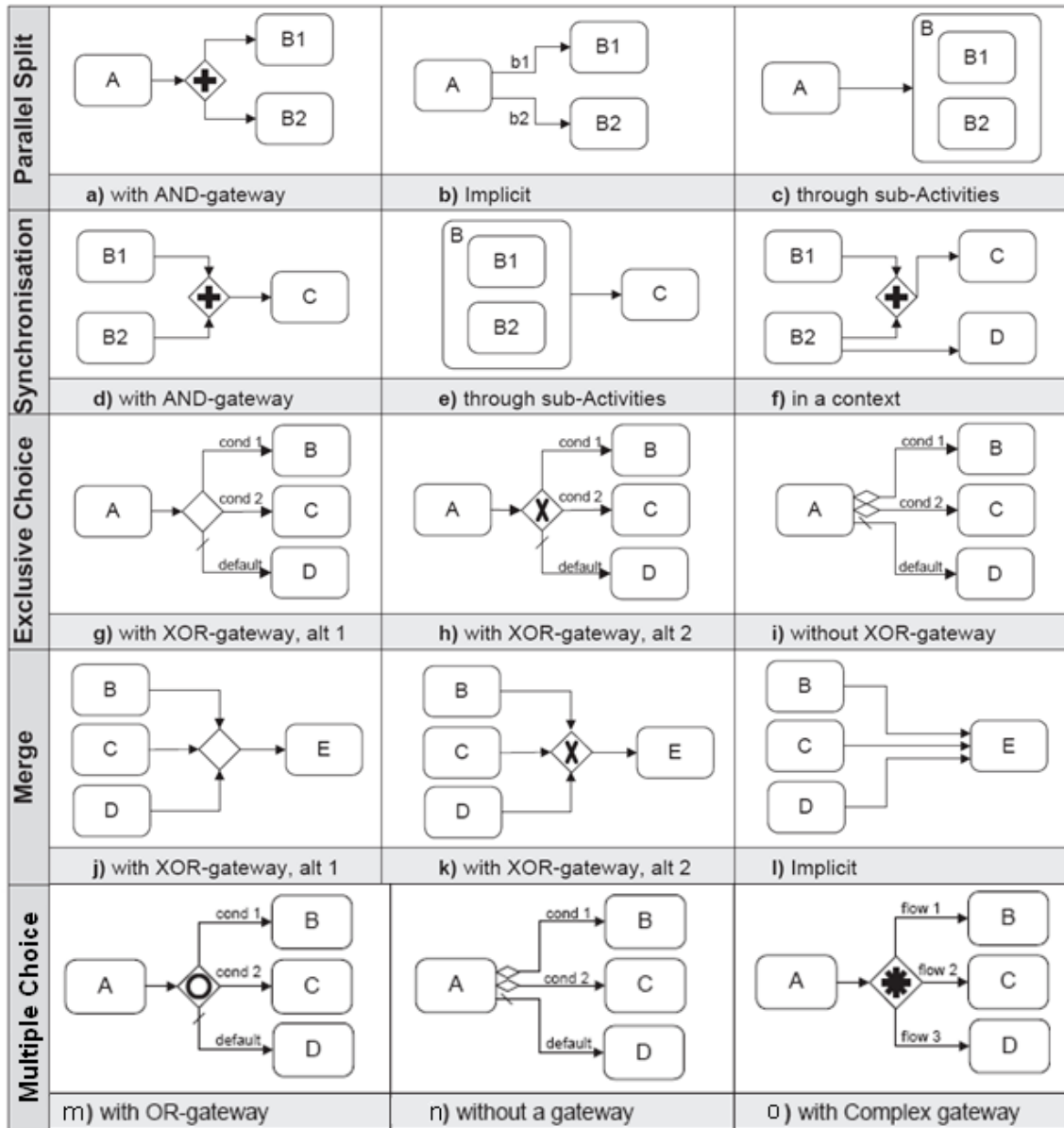


Figure 53 BPMN workflow patterns alternatives [Wohed, et al. 2006]

2

Example mapping process “execute order”

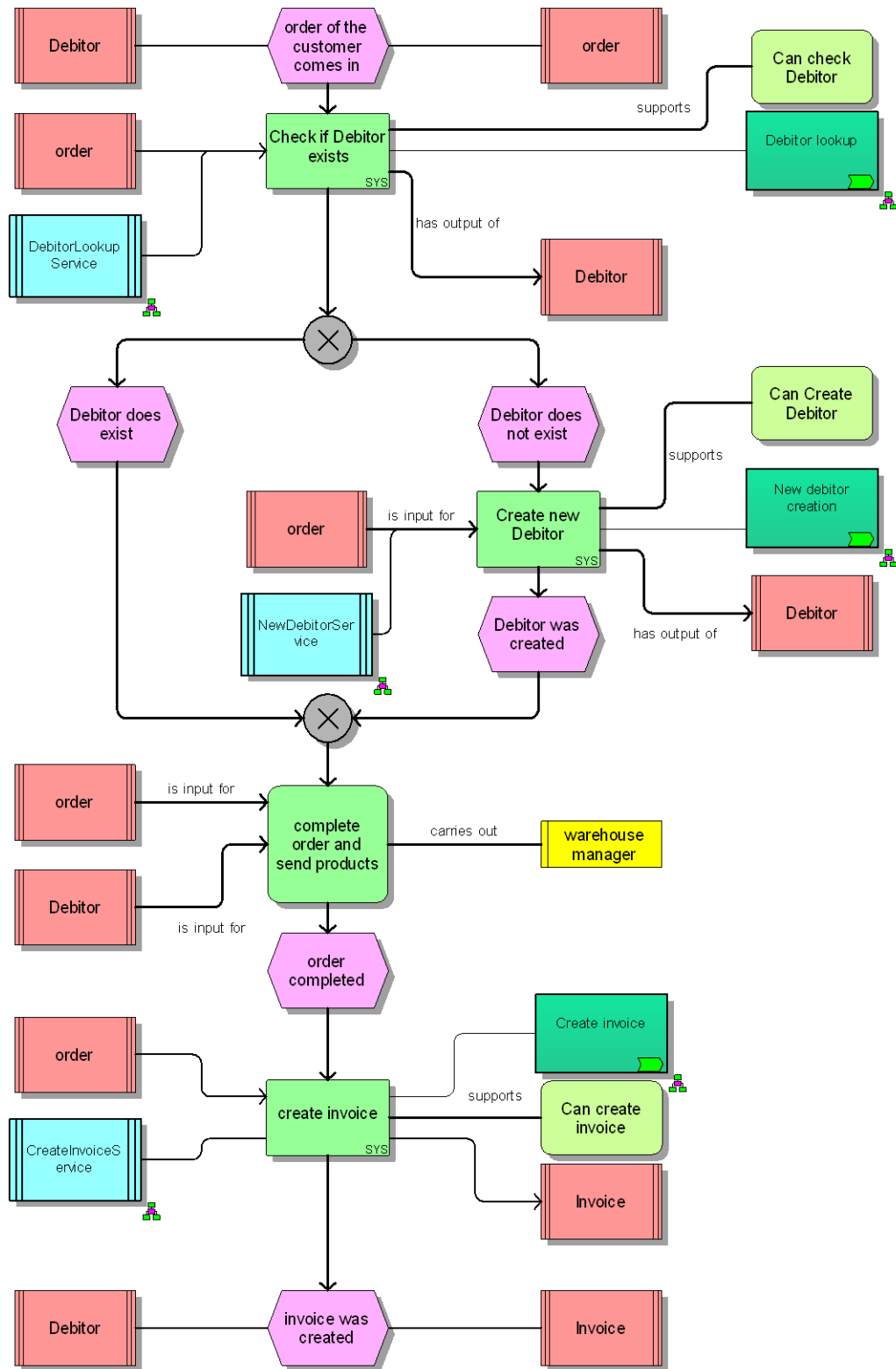


Figure 54 EPC example [IDS_Scheer_Academy 2009]

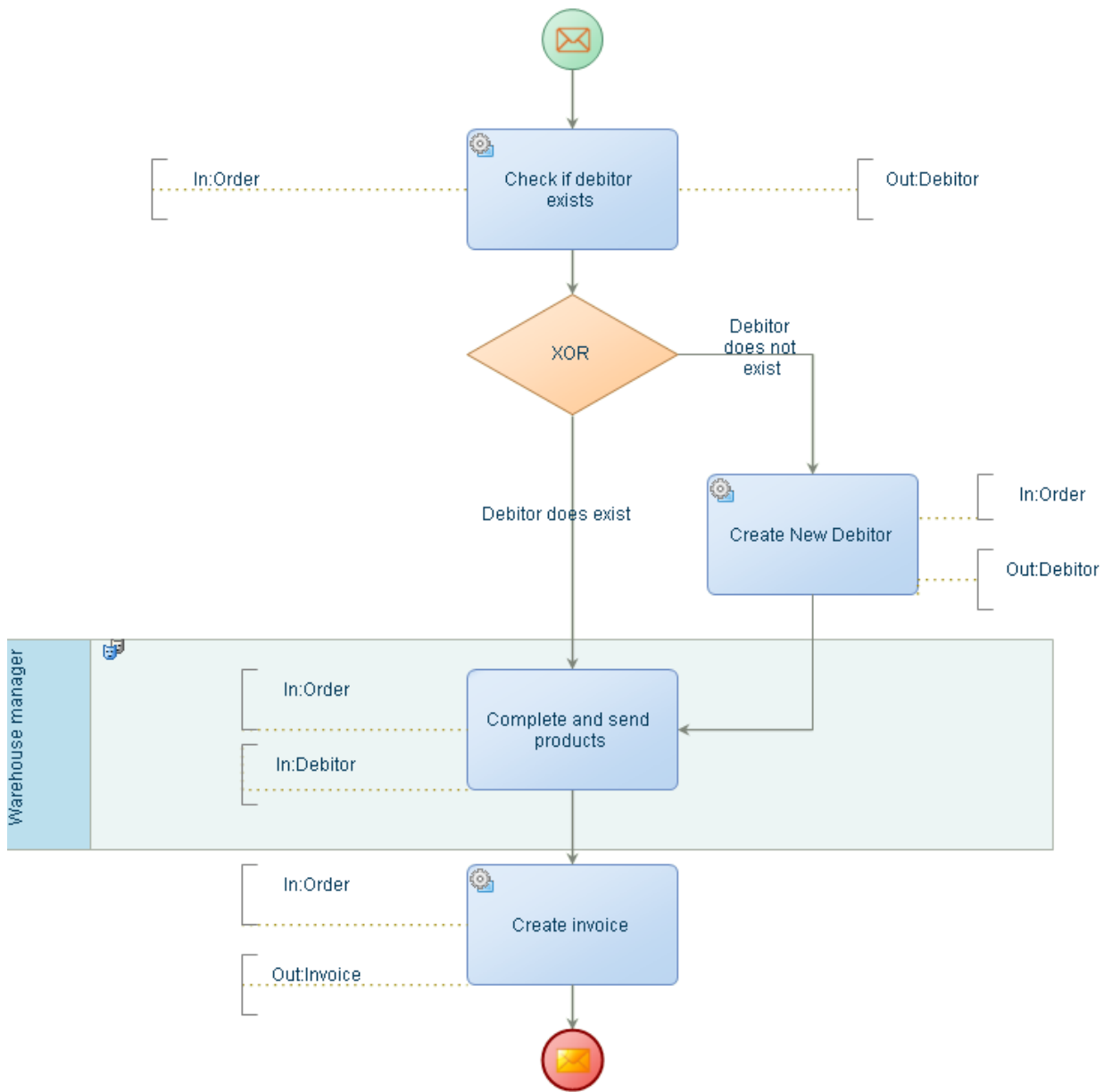


Figure 55 C-BPMN translated from EPC

Appendix H. ARIS EPC to BPMN transformation support

The table depicts which patterns EPC to BPMN transformation in ARIS is supported in the Process Automation Architect 7.1. Each pattern is described including its characteristics, mapping and whether they are relevant for the ARIS and Cordys mapping.

EPC Patterns	No.	EPC Characteristic	BPMN	Relevancy ARIS-Cordys
Start Event	1	No incoming connection, one start event	Start event	Yes
	2	Multiple start events followed by rule connector	Single start event	Yes
End Event	1	No outgoing connections	End event	Yes
	2	“Event (process instance terminated)” with placeholder in the end	End event	No, not in SO-EPC
	3	“Event (process instance terminated)” without placeholder in the end	End event	No, not in SO-EPC
	4	Rule connector followed by multiple end events	Single end event	Yes
Process Interface (PI)	1	Start	Link throw event	No, link not in BPM
	2	Start/optional rule connector/ event	Link catch event	No, link not in BPM
	3	Start/optional rule connector/ event/ rule	Link catch event + gateway	No, link not in BPM
	4	End/ optional rule/ event	Link throw event	No, link not in BPM
	5	End/ Cancelling discriminator/ Event	Cancelling discriminator (true) with link throw event	No, link not in BPM
Trigger Event	1	Event (process instance triggered)	Start Event	YES
	2	Starting placeholder/ Trigger Event	Start Event	No, not in SO-EPC
	3	Scheduling timer (+ attribute Scheduling period)	Timer start event	No, not in SO-EPC
	4	Scheduling timer(only event) (+ attribute Scheduling period)	Timer start event	No, not in SO-EPC
Timer Event	1	Timer Event (+DFD assignment considerations)	Timer event	No, not in SO-EPC
Intermediate Event	1	Intermediate Events (on in/outgoing connections > 0)	-	YES, in sequence path EPC
Rule	1	XOR, OR, AND (+ ‘must consider symbol’=false)	Gateway	YES
	2	Rule block only with intermediate events	empty	YES
	3	(X)OR/Event/(X)OR	XOR/OR gateway + gateway	YES
	4	Cancelling discriminator	Gateway (+ Boolean “cancelling denominator)	No, not in SO-EPC
Conditional	1	XOR condition + Process interface	XOR + link	No, link not in BPM
	2	XOR condition + intermediate event and placeholder	XOR/OR gateway + placeholder	No, not in SO-EPC

EPC Patterns	No.	EPC Characteristic	BPMN	Relevancy ARIS-Cordys
	3	XOR/OR + an end event	XOR/OR gateway + end event	YES
	4	XOR/OR+ intermediate event + another rule	XOR/OR gateway + gateway	YES
Abstract process patterns	1	Abstract Function	Subprocess (collapsed)	YES
	2	Abstract process (trigger)	Subprocess (collapsed) with Event (process instance triggered)	YES
	3	Abstract process VCD	Subprocess (collapsed)	No, not in SO-EPC
	4	Abstract process VCD closed	Subprocess (collapsed)	No, not in SO-EPC
	5	Abstract process (trigger) VCD	Subprocess (collapsed)	No, not in SO-EPC
	6	Abstract process (trigger) VCD closed	Subprocess (collapsed)	No, not in SO-EPC
Activities patterns	1	Notification function	Notification activity	No, not in SO-EPC
	2	Live message function	Live message activity	No, not in SO-EPC
	3	Automated task function	Automated task activity	No, not in SO-EPC
	4	Human task (4x)	Human task + additional process flow branches parallel to the main process flow.	No, not in SO-EPC
	5	Detailed task (2x)	embedded sub processes.	No, not in SO-EPC
	6	Bi-directional data flow split (4x)	Function + write variable	No, not in SO-EPC
	7	Manual functions (NOPs) (2x)	Function "NOP"	No, not in SO-EPC

Table 28 ARIS EPC2BPMN evaluation

Appendix I. Parameterized patterns Domain Example

1 Business cross role EPC to C-BPMN execution

An cross role process without service detail on EPC level is translated to C-BPMN. From there the proposed pattern transformations are applied as specified above to make the process executable. (Processes have been slightly altered. The example processes provided by IDS Scheer and Cordys of the pilot project contained private company information)

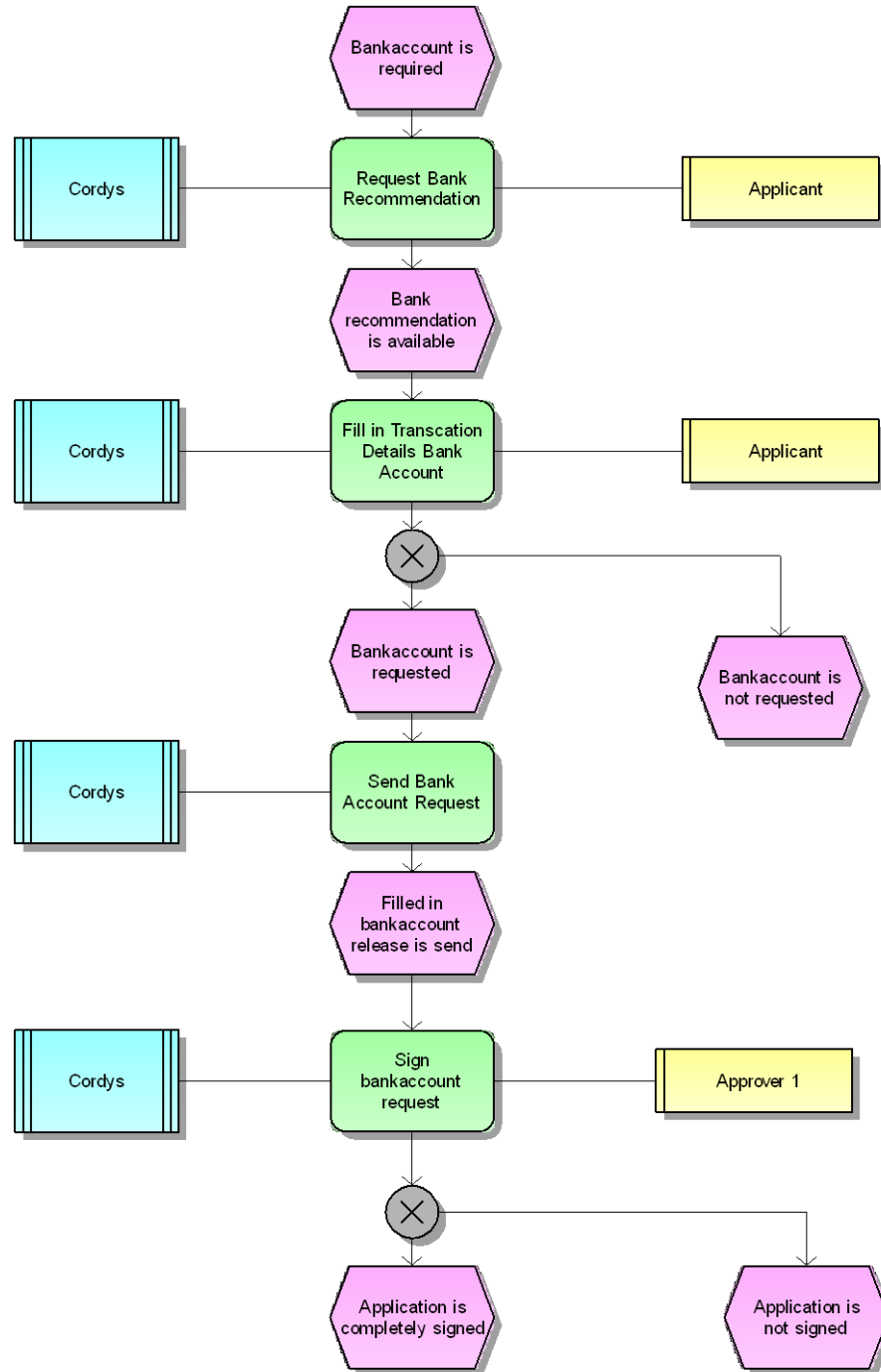


Figure 56 EPC "Request Bank Account"

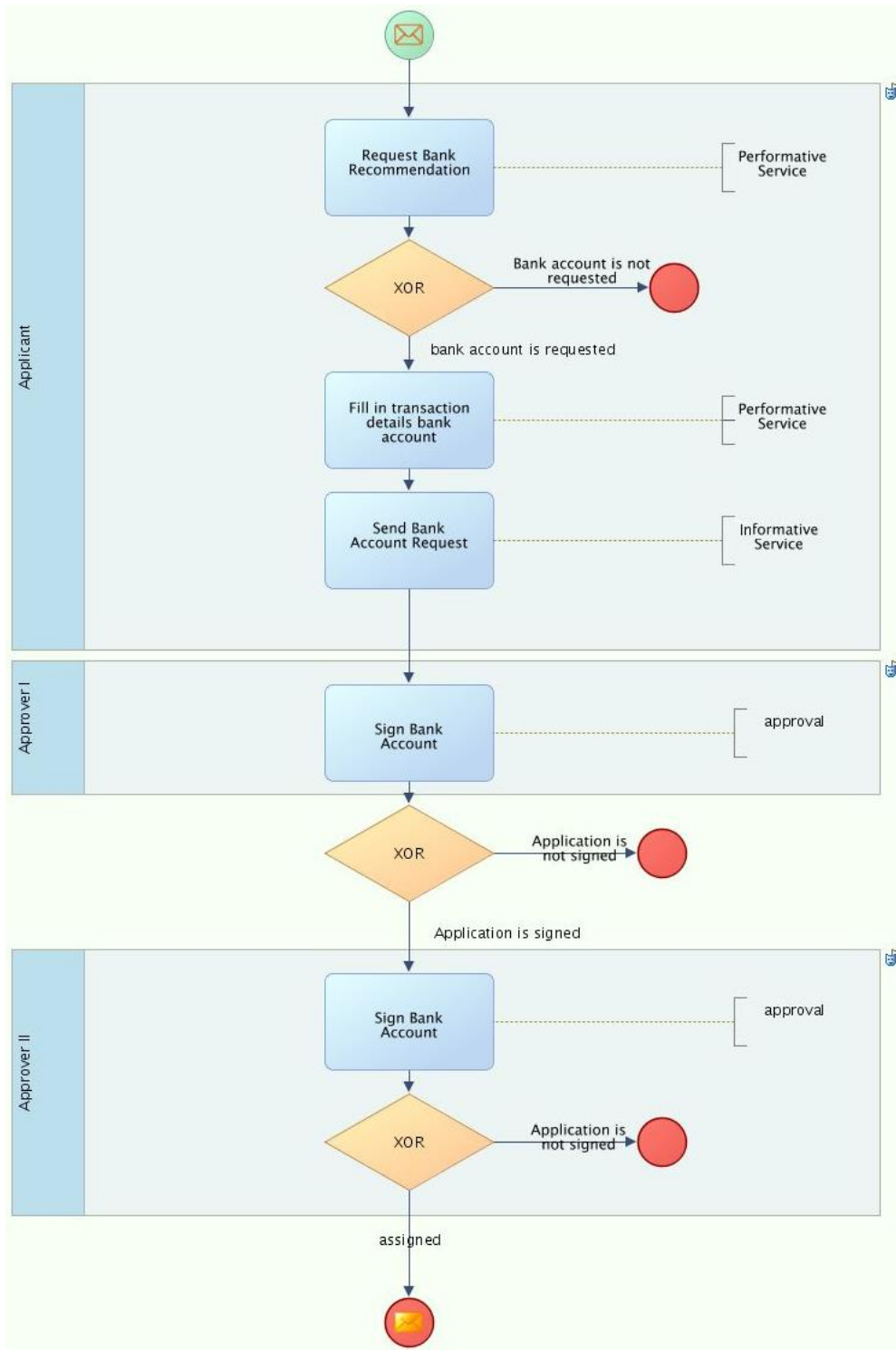


Figure 57 C-BPMN “Request Bank Account”

The text annotations depict the type of business activity. Note on this level the services has not been assigned for per formative service and notification, as these were not specified by the EPC. The services are required for execution and part of thus an extra transformation rule for operation to business.

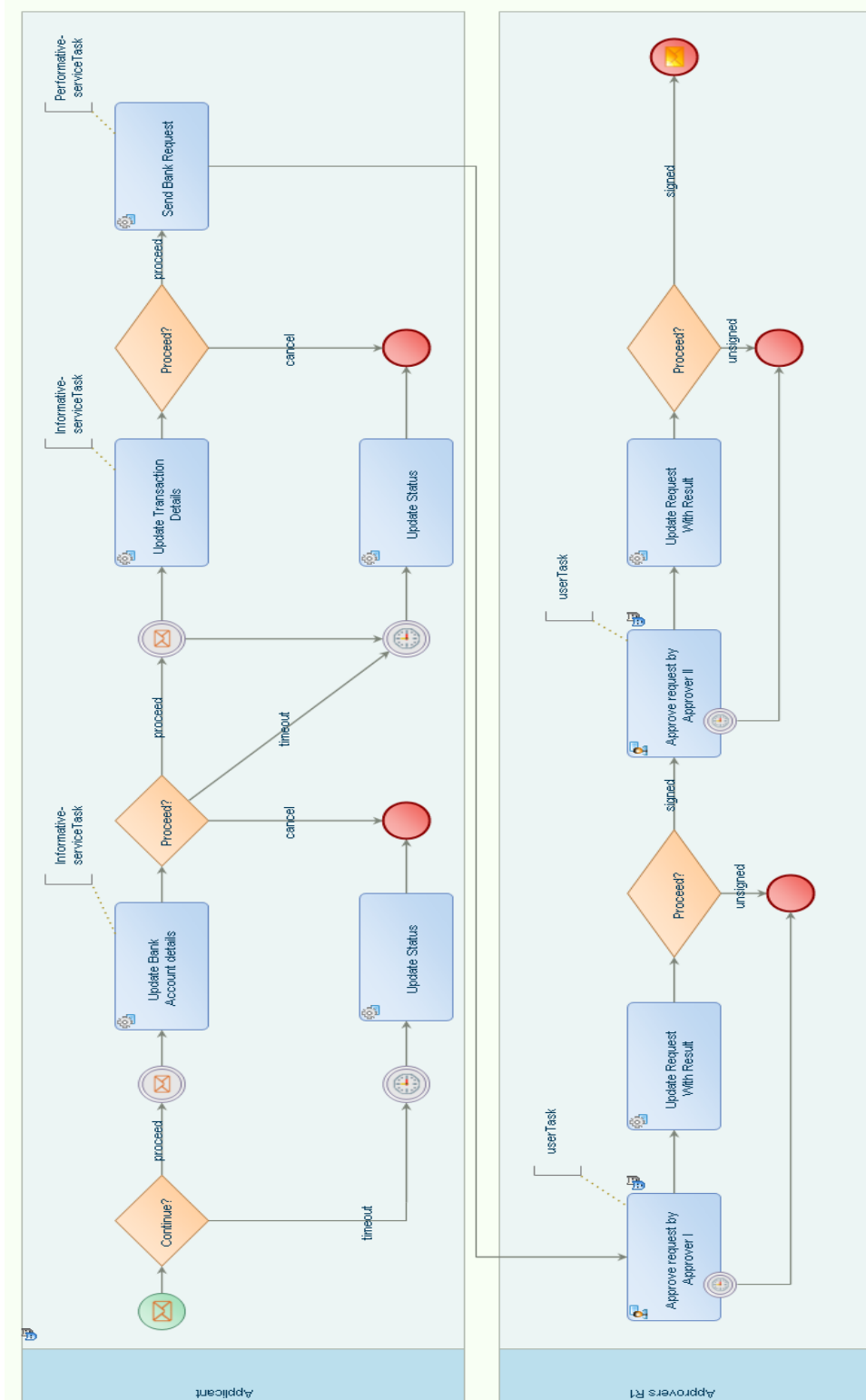


Figure 58 Technical C-BPMN “Request Bank Account”
 The text annotations depict the type of functional activity.

2 EPC Parameterized patterns Business to Functional



Approval User	
Approval	Approval by user in system
Source	Pattern
	
Additional parameters	Transformation rules
<ul style="list-style-type: none"> – Roles – Assign Manual or System 	<ul style="list-style-type: none"> – Change the approval to a manual or user task, the number corresponding with number of approvers.

Table 29 Approval User Parameterized pattern EPC Business to Functional


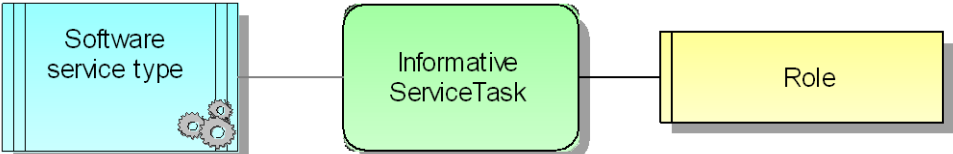
Informative Service	
Informative	Informative is a service requesting data input from user
Source	Pattern
	
Additional parameters	Transformation rules
<ul style="list-style-type: none"> – Role – WSDL 	<ul style="list-style-type: none"> – Import WSDL Link software service type – Attach role

Table 30 Informative Service Parameterized pattern EPC Business to Functional

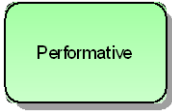
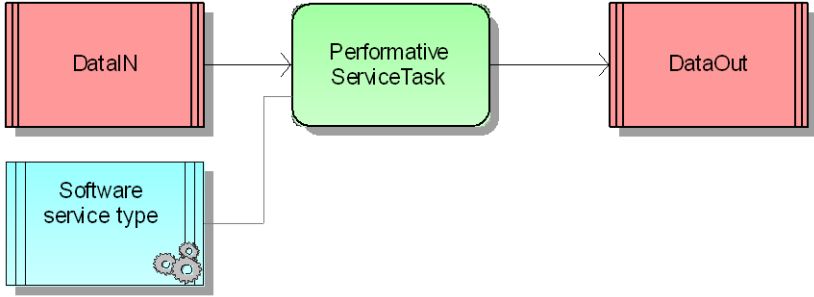
Performative Service	
Performative	Performative by service is a service with an executing activity
Description	Pattern
	
Additional parameters	Transformation rules
<ul style="list-style-type: none"> – Web Service – Data in and out 	<ul style="list-style-type: none"> – Define web service and data in and out

Table 31 Performative Service Parameterized pattern EPC Business to Functional

3 C-BPMN Parameterized patterns Business to Functional


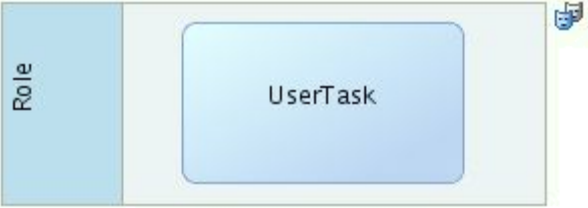
Approval	
Approval	Approval by user by interaction with system
Description	Pattern
	
Additional parameters	Transformation rules
<ul style="list-style-type: none"> – Role 	<ul style="list-style-type: none"> – Attach role to lane

Table 32 Approval Parameterized Pattern C-BPMN Business to Functional

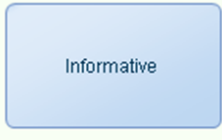
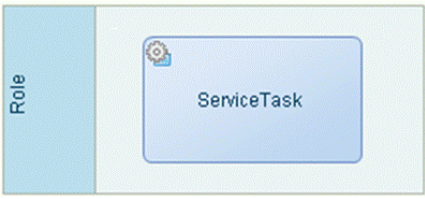
Informative Service	
Informative	An informative service task which requires input of the participant for service
Description	Pattern
	
Additional parameters	Transformation rules
<ul style="list-style-type: none"> – WSDL of service task – Message map: data in – out 	<ul style="list-style-type: none"> – Attach service – Specify data in and out

Table Informative Service Parameterized Pattern C-BPMN Business to Functional



Performative Service	
Performative	A performative service task which executes a service
Description	Pattern
	
Additional parameters	Transformation rules
<ul style="list-style-type: none"> – WSDL of service takes – Message map: data in - out 	<ul style="list-style-type: none"> – Attach service – Specify data in and out

Table 33 Performative Service Parameterized Pattern C-BPMN Business to Functional

4 C-BPMN Parameterized patterns Functional to Executable

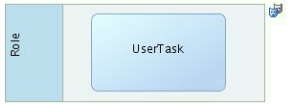
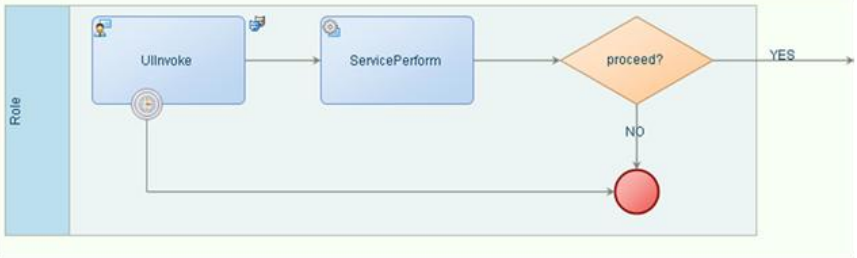
UserTask	
User Task	A user task executed with interaction with a system
Description	Pattern
	
Additional parameters	Transformation rules
<ul style="list-style-type: none"> – UI (Xform) for authorization – time out – WSDL for result update 	<ul style="list-style-type: none"> – Attach UI and role to task – Set time out – Attach update via service

Table 34 UserTask Parameterized patterns C-BPMN Functional to Executable

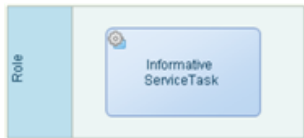
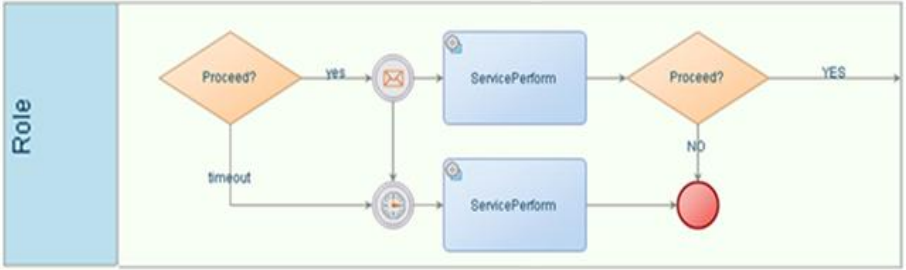
Informative ServiceTask	
Informative ServiceTask	A service task requiring input from participant
Description	Pattern
	
Additional parameters	Transformation rules
<ul style="list-style-type: none"> – WSDL for status update – Single, Concurrent or Iterative Optional: <ul style="list-style-type: none"> – Time out – Exception/Error handling 	<ul style="list-style-type: none"> – Attach service with status update – Set number of iterations (as attribute) Optional : <ul style="list-style-type: none"> – Set time out time – Set exception handling

Table 35 Informative ServiceTask Parameterized patterns C-BPMN Functional to Executable



Performative Service Task	
Service Task Executing	
Description	Pattern
	
Additional parameters	Transformation rules
<ul style="list-style-type: none"> – Single, Concurrent or Iterative Optional: <ul style="list-style-type: none"> – Time out – Exception/Error handling 	<ul style="list-style-type: none"> – Set number of iterations (as attribute) Optional: <ul style="list-style-type: none"> – Set time out time – Set exception handling

Table 36 Performative Service Task Parameterized patterns C-BPMN Functional to Executable

5 Approval activity pattern

Approval is an activity which requires a user to approve an object via a system. Examples: “Approve Bank Request”, “Sign Bank Request”.

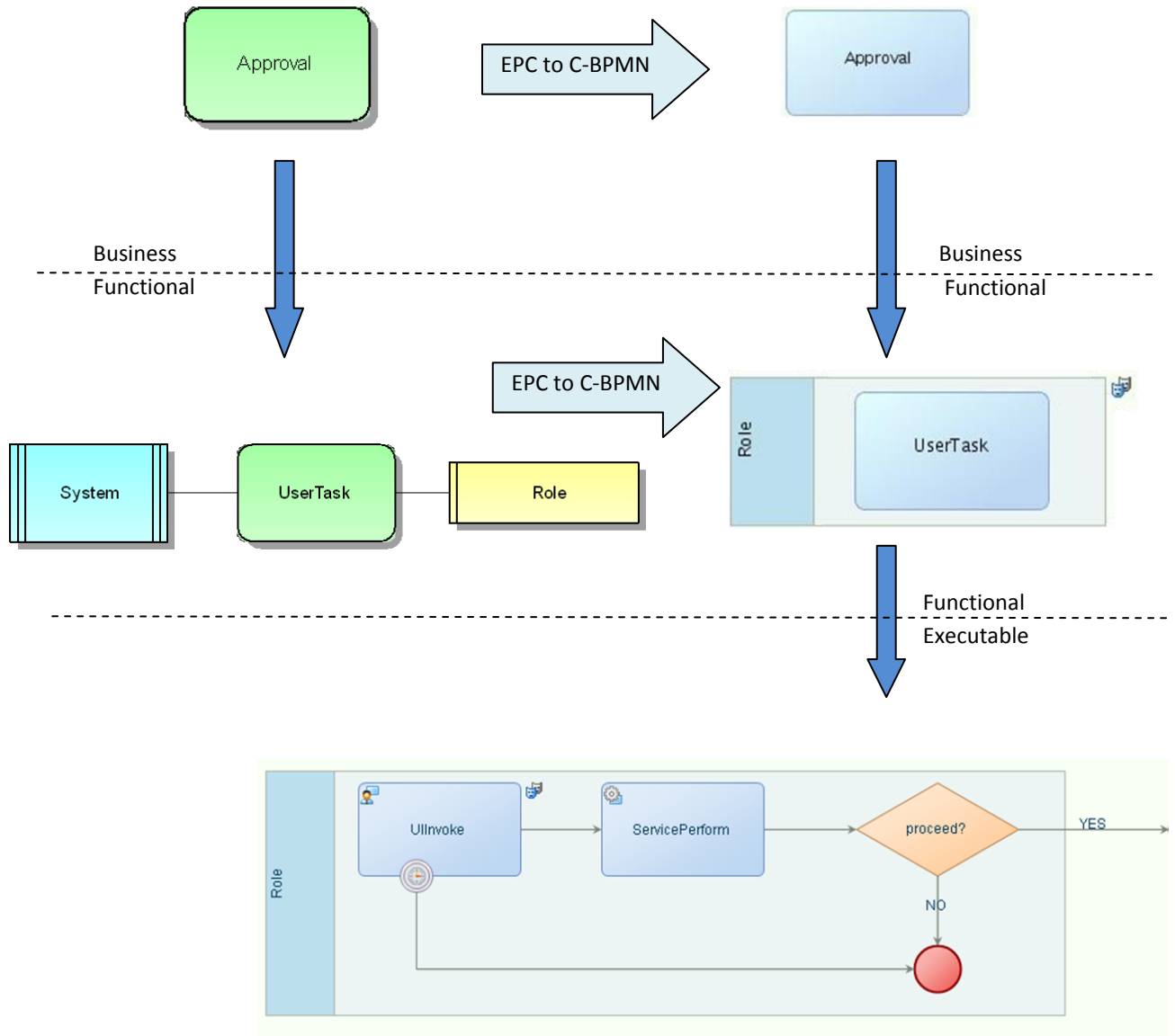


Figure 59 Approval Pattern Transformations domain specific

6 Informative Service activity pattern

Performative Service is an activity which invokes a service to perform a certain task. Example: “Create Account”, “Get Price”, “Request Bank account”.

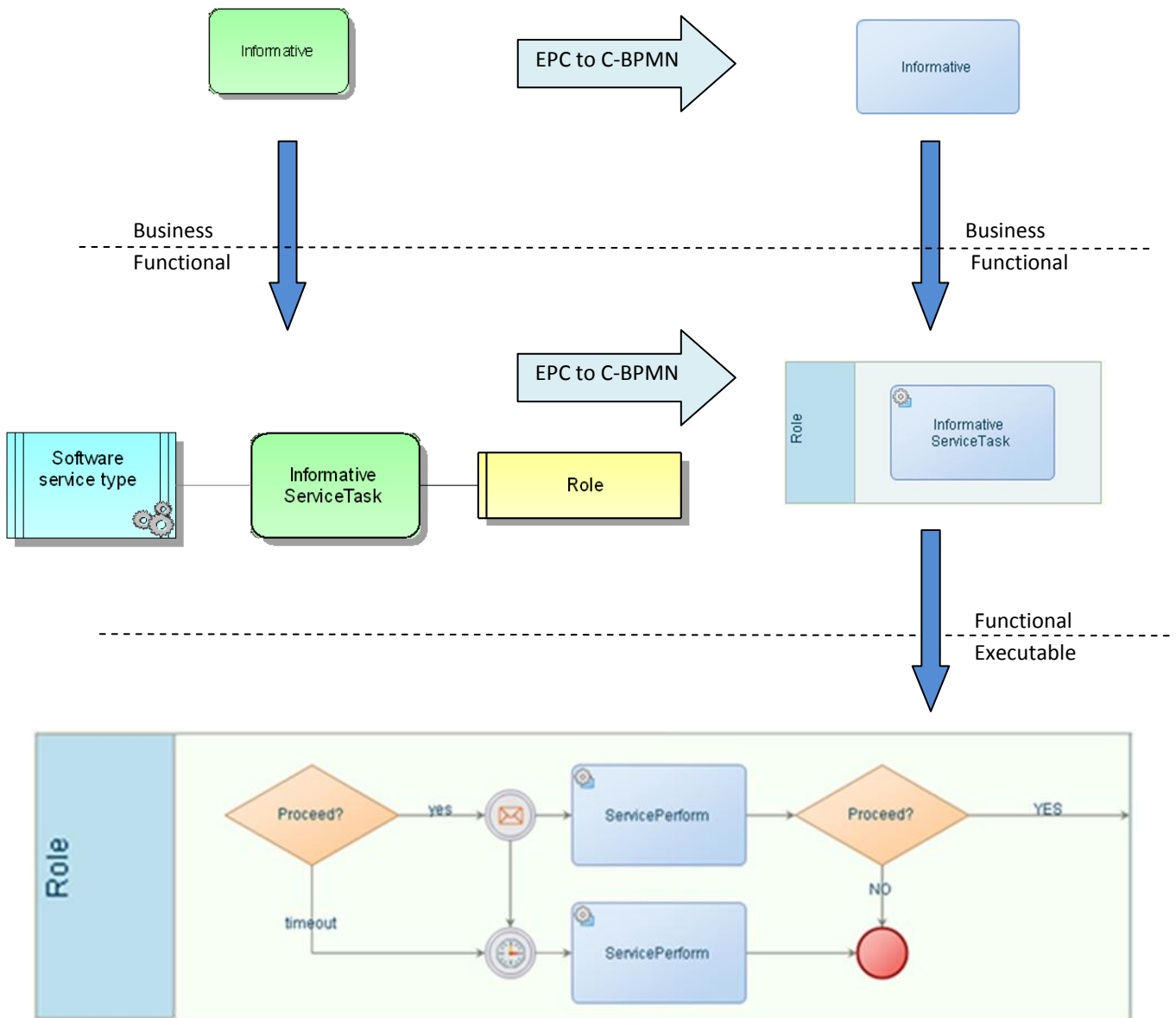


Figure 60 Informative Service Pattern Transformations domain specific

7

Performative Service activity pattern

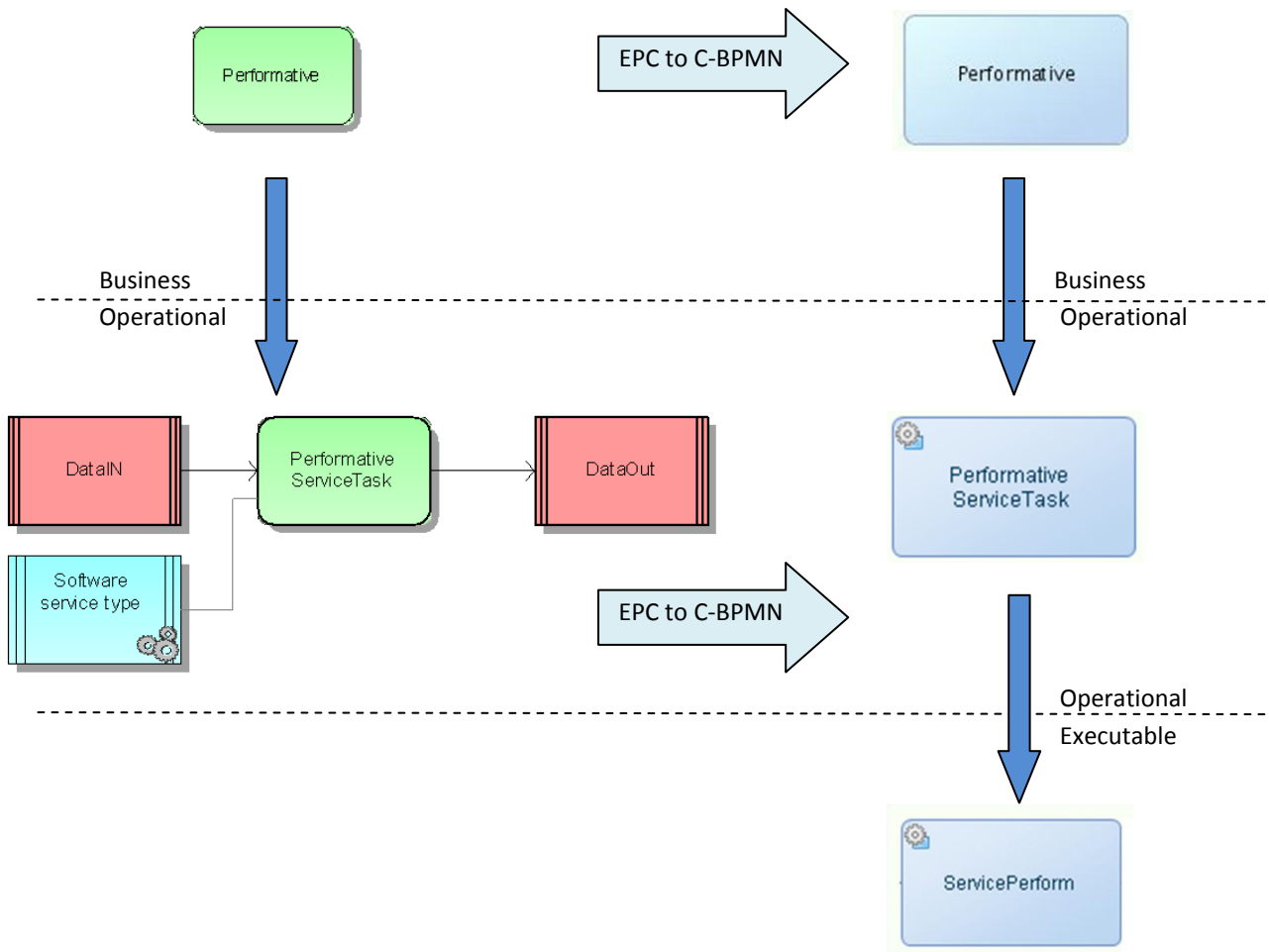


Figure 61 Performative Service pattern transformation domain specific