

MASTER

Configurable process models a municipality case study

Wagemakers, T.A.C.

Award date:
2009

[Link to publication](#)

Disclaimer

This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

**Configurable Process Models:
A municipality case study**

Teun Wagemakers, July 2009

Preface

The thesis is a result of the case study of creating configurable process models of municipality processes. The case study is representing the graduation phase of my study at the Eindhoven University of Technology. During the case study I was supported by my supervisors; Florian Gottschalk and Monique Jansen-Vullers. I want to thank Florian for his support during the case study and for always being prepared to answer my questions and wish him good luck with the defense of his promotion thesis. I want to thank Monique for her critical view and guidance to keep me on schedule as good as possible. Most of all I want to thank them for making the graduation project a nice period. I really enjoyed working with them.

Besides the support of my supervisors I also got support from family and friends. I want to thank my friends for distracting me and adding some years to my study and I want to thank my parents for always supporting me and encouraging me during all the years I studied. I want to specially thank Femke for supporting me even do I was not always in good mood especially during the period that I had to finish my thesis, work 40 hours, and doing constructing jobs in our new house.

Now a really nice period in my life is ended and a new, exciting and a challenging working career is started.

Tilburg, July 2009

\

Teun Wagemakers

Abstract

Configurable process models combine several variants of a business process into one model. The configurable process model has, with reference to other process models, the possibility to be configured. By configuring the users do not have to start from scratch but can choose from several options to configure a configurable process model till it satisfies their needs.

The approach of creating configurable process models is already researched several times but it was not tested in a real situation. In this thesis we describe the process of creating configurable process models and our findings by means of a case study. The business processes which were used are four common used processes from four Dutch municipalities. The configurable process models we created during the case study can be executed in the YAWL environment. The configuration of the models is done during this research by means of a questionnaire approach to avoid that the users are confronted with a complex process model. For each of the selected processes we created a configurable process model and evaluated them with the stakeholders.

Table of Contents

1	Introduction	1
1.1	Objective	3
1.2	Approach.....	6
1.3	Outline.....	10
2	Preliminaries	11
2.1	Protos.....	11
2.1.1	Protos behavior.....	12
2.2	YAWL.....	12
2.2.1	YAWL behavior.....	13
2.2.2	Data perspective	15
2.2.3	YAWL-engine.....	16
2.3	Configurable workflow models.....	17
2.3.1	The configurable approach	17
2.3.2	Configurable workflow model behavior	18
2.3.3	Configurable YAWL	22
2.4	Questionnaire	23
2.4.1	Questionnaire behavior	24
2.5	Mapping.....	26
2.6	Related work.....	27
3	Configurable Process models: a Case study.....	31
3.1	Collecting processes.....	31
3.1.1	Selecting the processes for the case study.....	32
3.1.2	Process descriptions.....	34
3.1.2.1	Acknowledgement unborn child	35
3.1.2.2	Birth	35
3.1.2.3	Marriage	36
3.1.2.4	Decease	37
3.1.3	Selecting the municipalities	37
3.1.4	Approach of collecting.....	39
3.1.5	Data collection: Result	40
3.2	Identify variations	44
3.2.1	Similarities.....	45
3.2.2	Differences.....	46
3.3	How to combine.....	50
3.3.1	Combining behavior.....	51
3.3.2	Modeling different order	52
3.3.3	Creating extra behavior	55
3.3.4	Checking the model	59
3.3.5	Combining models: Result	59
3.4	Transform the models in YAWL	61
3.4.1	Similarities amongst YAWL and Protos.....	62
3.4.2	YAWL combined model.....	63
3.4.3	YAWL data.....	63
3.4.4	Control flow	64
3.5	Create questionnaire	66

3.5.1	Identifying the tasks that are configurable	67
3.5.1.1	Configurable clusters.....	67
3.5.2	Specifying questions for the questionnaire	70
3.5.2.1	Directly configuration by means of a questions.....	70
3.5.2.2	Indirectly configuration by means of a questions	71
3.5.3	Design dependencies between questions	72
3.5.4	Creating constraints	73
3.6	Mapping	74
3.6.1	Excel compilation sheet	75
3.6.2	XML file	76
4	Results.....	78
4.1	Generalization	78
4.2	Observations	81
4.3	Evaluation with the stakeholders	84
4.3.1	Pallas Athena.....	85
4.3.2	Software provider	86
4.3.3	Consultancy firm	87
4.3.4	NVVB	88
4.3.5	Municipality	89
5	Discussion.....	92
5.1	Discussion.....	92
5.2	Reflection	93
5.3	Future Work	94
	References:.....	96
	Glossary.....	101

Appendices

Appendix 1 Questionnaire Interface	106
Appendix 2 Using the Configurator and the Individualizer	108
Appendix 3 individual <i>birth</i> business process model A	110
Appendix 4 individual <i>birth</i> business process model B.....	111
Appendix 5 individual <i>birth</i> business process model C.....	112
Appendix 6 individual <i>birth</i> business process model D	113
Appendix 7 individual <i>birth</i> business process reference model	114
Appendix 8 individual <i>acknowledgement of an unborn child</i> business process model A	115
Appendix 9 <i>individual acknowledgement of an unborn child</i> business process model B	116
Appendix 10 <i>individual acknowledgement of an unborn child</i> business process model C.....	117
Appendix 11 <i>individual acknowledgement of an unborn child</i> business process model D	118
Appendix 12 <i>individual acknowledgement of an unborn child</i> business process reference model.....	119
Appendix 13 individual <i>deceased</i> business process model A.....	120
Appendix 14 individual <i>deceased</i> business process model B	121
Appendix 15 individual <i>deceased</i> business process model C	122
Appendix 16 individual <i>deceased</i> business process model D.....	123
Appendix 17 individual <i>deceased</i> business process reference model	124
Appendix 18 individual <i>marriage</i> business process model A.....	125
Appendix 19 individual <i>marriage</i> business process model B	126
Appendix 20 individual <i>marriage</i> business process model C	127
Appendix 21 individual <i>marriage</i> business process model D.....	128
Appendix 22 individual <i>marriage</i> business process reference model.....	129
Appendix 23 Combining of a subprocess.....	130
Appendix 24 trace check of the birth model of appendix 4	131
Appendix 25 configurable <i>birth</i> business process model.....	132
Appendix 26 configurable <i>acknowledgement of an unborn child</i> business process model	133
Appendix 27 configurable <i>deceased</i> business process model.....	134
Appendix 28 configurable <i>marriage</i> business process model	135
Appendix 29 translation from Protos to YAWL	136
Appendix 30 Configurable <i>Birth</i> Workflow Model in YAWL.....	137
Appendix 31 Configurable <i>Acknowledgement of an unborn child</i> Workflow Model in YAWL	138
Appendix 32 Configurable <i>Deceased</i> Workflow Model in YAWL.....	139
Appendix 33 Configurable <i>Marriage</i> Workflow Model in YAWL.....	140
Appendix 34 Adding data to the YAWL model	141
Appendix 35 The variations in the Configurable <i>Birth</i> Workflow model	143
Appendix 36 Excel compilation sheet.....	144
Appendix 37 XML file <i>Birth</i> mapping.....	145
Appendix 38 Configured <i>birth</i> Workflow Model in YAWL	148
Appendix 39 Configured <i>Acknowledgement of an unborn child</i> Workflow Model in YAWL...	149
Appendix 40 Configured <i>Deceased</i> Workflow Model in YAWL	150
Appendix 41 Configured <i>Marriage</i> Workflow Model in YAWL.....	152

Table of figures

<i>Fig. 1-1 Phases to create configurable workflow models</i>	9
<i>Fig. 2-1 Symbols used in Protos</i>	11
<i>Fig. 2-2 Example of a Protos business process model</i>	11
<i>Fig. 2-3 Symbols used in YAWL [11]</i>	13
<i>Fig. 2-4 XOR-, AND-, OR- behavior</i>	14
<i>Fig. 2-5 The representation of a possible form for a task</i>	17
<i>Fig. 2-6 The number of ports of an action depends on its joining and splitting behavior [6]</i>	19
<i>Fig. 2-7 Ports can be activated or blocked, and in case of an inflow also be hidden [6]</i>	19
<i>Fig. 2-8 Three examples of configuring a configurable process model</i>	21
<i>Fig. 2-9 When a model is completely enabled it is the base model</i>	22
<i>Fig. 2-10 Transforming the output port into YAWL model [6]</i>	23
<i>Fig. 2-11 Transforming of an OR-split into an XOR-split [6]</i>	23
<i>Fig. 2-12 Transformation of an OR-split into an AND-split [6]</i>	23
<i>Fig. 2-13 Example: structure of question-facts. Legend [17]</i>	25
<i>Fig. 2-14 Fact f3 and f4 depend on the fact f2 to be true</i>	26
<i>Fig. 3-1 Phases to create configurable workflow models used as guidance</i>	31
<i>Fig. 3-2 First phase: Collecting process models</i>	32
<i>Fig. 3-3 The birth business process model of one of the municipalities</i>	43
<i>Fig. 3-4 Phase 2: identifying differences and similarities</i>	44
<i>Fig. 3-5 Similarities between two birth business process models</i>	46
<i>Fig. 3-6 Differences between two birth business process models</i>	49
<i>Fig. 3-7 Phase 3: Combining the models</i>	50
<i>Fig. 3-8 Three different ways of modeling different orders of execution are depicted by models I, II, III</i>	53
<i>Fig. 3-9 Combining model A and model B resulting in model C. Extra behavior is added to model C</i>	56
<i>Fig. 3-10 Combining model E and model F resulting in model G</i>	56
<i>Fig. 3-11 Outgoing port of action X is configured to eliminated extra behavior</i>	57
<i>Fig. 3-12 Combined model of two birth business process models</i>	58
<i>Fig. 3-13 As starting point two business process models will be merged, which results in the combined model. When more than two models must be combined every iteration of one model can be added to the combined model.</i>	59
<i>Fig. 3-14 The combined business process model of the Birth process containing five birth business process models</i>	60
<i>Fig. 3-15 Phase 4: Transform Protos to YAWL</i>	61
<i>Fig. 3-16 Configurable birth workflow model</i>	64
<i>Fig. 3-17 The flow detail for atomic task "Receive notification birth"</i>	65
<i>Fig. 3-18 The flow predicate check term of notification</i>	65
<i>Fig. 3-19 Phase 5: Create a questionnaire</i>	66
<i>Fig. 3-20 Workflow model divided in clusters of tasks which are always executed in the same formation</i>	69
<i>Fig. 3-21 Question-fact structure of the birth questionnaire</i>	73
<i>Fig. 3-22 Phase 6: Create a mapping</i>	74

Fig. 4-1 Depiction of the generic guidance to create configurable YAWL workflow models..... 80
Fig. 4-2 Configuring the model by blocking the output port from A to B instead of hiding the input port of B, to create the situation in which A must be followed by C. 83
Table. 4-1 The main comments of the interviewed stakeholders [46] 91

1 Introduction

Processes performed within and among municipalities are very similar. As the government has regulations that apply for inhabitants of the Netherlands, they apply for inhabitants of all municipalities. For example, those regulations specify that inhabitants need to go to the municipality to register a child, to get a passport, and to inform a municipality when someone has died. To make sure the quality of the municipalities in the Netherlands is sufficient, the *Department of the Interior* performs audits [1]. During the audits at the municipalities the business processes are checked. A business process is a set of logically related tasks performed to achieve a defined business outcome [2]. The defined outcome could be a passport for an inhabitant in the case of a municipality. When a set of logically related tasks is depicted by symbols it becomes a business process model. These business process models are used by the municipalities among other things to prepare for the audit of the *Department of the Interior* [3] and to inform their employees about the processes.

Because municipalities differ in size and capacity, the business processes and the related models are slightly different amongst municipalities. To prevent that every municipality has to create the business process models from scratch the Nederlandse Vereniging voor Burgerzaken (NVVB, Dutch Society for Civil Affairs) has created several reference models. Reference models are based on best practices and claim to improve the reusability [4]. In this way the municipality can adapt the reference model to their situation.

The executions of the business processes at the municipalities are in some cases supported by workflow systems. A *workflow system* deals with the automation of procedures where documents, information, or tasks are passed between participants according to a defined set of rules to achieve, or contribute to, an overall business goal [5]. When a business process model is used to enable the automated execution in a workflow system it is called a *workflow model*. If such a model exists, the particular business process is also called a workflow [6].

Adapting the business process models and/or workflow models to the situation of a specific municipality, requires extensive modeling skills and knowhow of the process. In the remainder of the thesis we will use the term *models* when we are talking about both business process models and workflow models. Normal models do

not have the possibility to be configured. During the case study we noticed that they are often a derivative of a reference model. These models depict only one variant of the process and provide no possibility for alternative configurations. All changes to the model require manual modeling efforts from the model user. *Configurable process models* deal with this situation and aim at improving the adaptability of models [6]. They incorporate several variations of how the process can be executed into one model. From this model different process variants can be derived by means of configuration. The configuration can be done by enabling or disabling tasks. For example if a municipality does not want to have a task in their model to identify a citizen, they could disable this task. This way the task won't occur in their model. By configuring the whole configurable process model the user can derive a process variant without explicit process modeling efforts and have a model as result that suits their situation. The disabling of tasks is done by blocking or hiding.

- Blocking a task results in disabling a task and all successors of the blocked task.
- Hiding a task disables the task that is hidden but all successors still are enabled.

We distinguish the three phases: (1) *build time* of the model, i.e. the time while configurable model incorporating all variants of the process was built, (2) *configuration time*, i.e. the time when a particular workflow variant is selected, and (3) *run time*, i.e. the time when process instances are executed using the configured model [6]. It is possible to transform the configuration time decisions into run-time decisions but this has some drawbacks. The configuration decisions look like run-time decisions while in advance already is decided how to use the different configuration options. Therefore unwanted combinations of enabled or disabled actions can occur and result in a deadlock or in unwanted process flows when configuration decisions become runtime decisions. A deadlock is a state in which the flow cannot continue towards the next state or task and is not the final state.

Another disadvantage of transforming configuration decisions into run-time decisions are that the size and the complexity of the models is increasing because all actions are enabled and therefore visible. The model size is of dominant importance on model understandability [7]. By having a configuration phase between the build time and the run-time, the size and the complexity of the run-time model can be

decreased. During the configuration phase all the actions will be enabled or disabled according to the desired configuration. By the enabling or disabling of actions, certain parts of the model are not reachable anymore and will therefore be eliminated from the model. Eliminating the parts that are not reachable anymore will improve usability and will decrease the size of the model. Compared to the original modeling languages, the configurable process language is more complex. But only the designers of the configurable models will be in contact with the configurable process language and that is a limited number in comparison with the number of users of the model.

To make the configurable model also accessible for other users besides the designers, a questionnaire can be used. The questionnaire is used in the configuration phase to guide the user through the configuration without exposing them to the complex configurable model. The answers of questions are related to certain tasks in the configurable model. Depending on the answer on a certain question a task will be enabled or disabled. This will result in a model that is deducted from the configurable model, which satisfies the users' preferences. In this way the range of people that can configure a configurable process model will be increased, resulting in a model that will suit the specific situation without explicit modeling skills.

Until now we only introduced the municipalities and the NVVB as organizations that could benefit from the configurable approach. Beside these organizations other organizations could also benefit from the configurable approach like, software developers and consultancy organizations. The software developers create the software that supports the business processes in organizations and the consultancy organization advises their clients about how they should improve and manage their processes. These organizations will be referred to as the stakeholders.

The configurable approach is investigated by means of a case study in the remainder of the thesis.

1.1 Objective

Until now the theory behind configurable process models has not yet been tested in practice. Therefore the approach of the configurable process models is investigated by means of a case study. A set of reference models for municipalities developed by the NVVB is used as a starting point. By collecting several business process models

from different municipalities, the models can be mutually compared and the different variants of how these processes can be executed can be detected. The comparison between the reference models and the models from the municipalities will lead to variations. A variation is the difference between two process models that describe the same process in different organizations. It can be the case that a process in one organization has extra, different, or less actions than another organization. These variations will become configuration options in the configurable process model. Besides the variations also other actions in a process model can be configured but the configuration should always be deadlock free and have a logical process model as result. A logical process model ensures a set of logically related tasks performed to achieve a defined business outcome. During the case study several process models of municipalities will be combined and made into configurable process models. This allows the feasibility of creating and using configurable process models in practical application scenarios to be evaluated. Therefore the main objective can be formulated as follows.

Objective:

Develop a set of configurable process models for civil affairs processes in municipalities to evaluate the feasibility of configurable process models.

In order to achieve the main objective the following research question is formulated.

Research question:

Is it feasible to create a configurable workflow model out of several process models of municipalities, and what is the added-value for the stakeholders?

To answer the research question we have to identify if we are capable of transforming individual business process models into a configurable workflow model. By investigating if we can create a configurable workflow model based on the processes of the municipalities, we need to identify if the variations amongst the processes of the municipalities are sufficient. If the variation is not sufficient we will not be able to create a configurable workflow model. When it is sufficient we need to check if it is possible to combine the individual business process models into one combined business process model. Then the combined business process model must be checked to know if the individual business processes of the municipalities can be

derived by means of the configurable method. At that moment we will have a combined business process model but we want to create a configurable workflow model. To achieve this, a workflow language is needed which is capable of dealing with the configurable approach. Then this workflow language than can be used to create the configurable workflow model. At the end of the creation we can state what the difficulties are in the process of creating a configurable workflow model.

To identify if the stakeholders are satisfied with the result, we need to evaluate our approach with them. This will give an indication if the configurable approach will be accepted by the industry and if the end user has a need for it. The stakeholders we selected to evaluate the approach are the municipalities that we visited, Software developer, Consultancy firm, and Pallas Athena. The involvement of the selected organizations with our approach is:

- The Municipality we chose one municipality out of the four municipalities we visited, to represent the end-users during the evaluation.
- Software developer is selected because they are supplying over two hundred municipalities with software that supports the business processes of the municipalities.
- Consultancy firm is a large consultancy organization that advice their clients about business processes.
- Pallas Athena is an organization that consults their clients about business processes and has software that supports business processes. Over two hundred municipalities use their software (Protos) to model business processes.

These organizations are chosen to evaluate our approach based on their experiences in the market.

During the case study the configuration of roles and resources will be outside the scope because of lack of time. The focus during the case study will be on the possibility of creating configurable workflow models from several business process models. During the process of creating configurable workflow models from business processes models the focus is on the actions in the model and not the roles and resources.

1.2 Approach

The approach that we used to answer the research question is a case study. By design, case studies usually take as their principal subject selected examples of a social entity within its normal context. At the simplest level, the case study provides descriptive accounts of one or more cases, but can also be used in an intellectually rigorous manner to achieve experimental isolation of one or more selected social factors within a real-life context [8]. Another definition for case study is given by [9].

A case study is an empirical inquiry:

- investigates a contemporary phenomenon within its real-life context; when
- the boundaries between phenomenon and context are not clearly evident; and
- in which
- multiple sources of evidence are used [9]

We investigated if configurable process models (phenomenon) can be created for a specific process of municipalities (real-life context). We don't know what the influence of municipalities is on the creation of a configurable process model. We do know that the departments of civil affairs of the municipalities use widely standardized processes. This is an important factor for creating configurable process models because if there is no overlap amongst the processes the configurable process models are not useful. Configurable process models use the overlap to find the variations that then can be used to configure a process. The resources that we used to collect evidence are archival records, interviews, and direct observations. The archival records we used to map the processes are business process models some municipalities distributed to us. We also went to each municipality to interview the process owner and observe the activities they executed during the process.

There are four types of case study designs specified by [9]: (1) single-case (holistic) designs, (2) single-case (embedded) designs, (3) multiple-case (holistic) designs, and (4) multiple-case (embedded) designs. The difference between holistic and embedded case studies is the number of units of analysis. We will execute a holistic case study because we want to test if it is possible to create configurable process models for civil affair processes and analyze each case individually. Because we use different cases i.e. four different civil affair processes, we are executing a multiple-case study. The decision is based on the idea that a multiple-case study is more compelling and therefore regarded as more robust [9]. We assume that if we

can create a configurable process model for four different processes the chance it will also hold for other processes is larger than when we only looked at one process.

We will answer the research question by executing six phases. The phases are depicted in Figure 1.1. Every phase represents a part of the process of creating configurable process models. During the thesis the phases will be used as a guidance to indicate which part of the process is discussed.

Phase 1

The start of creating configurable process models begins with collecting process models from several sources. The sources that we used were several municipalities and the NVVB. The municipalities gave us the opportunity to model the processes that they execute and the NVVB provided us with reference models.

Phase 2

The models that were collected in *Phase 1* need to be checked on differences and similarities. The similarities between models will be merged and the differences will be added to the model as a variation. In order to this we identify the similarities and the differences.

Phase 3

The models that were collected in *Phase 1* need to be merged into one model with several variations. By using the results of *Phase 2* we identify if we need to merge two activities or to add an activity to the model. The result is a model that combines all the activities of the collected models of *Phase 1*.

Phase 4

In order to create a configurable model a configurable modeling language is needed. The modeling language we use is YAWL (Yet Another Workflow Language). YAWL is open-source and therefore we can translate a configuration in a manner it is understandable for YAWL. Models that are notated in a different modeling language need to be translated into YAWL. When the collected models from *Phase 1* are already notated in YAWL the translation to YAWL will be redundant.

Phase 5

The configuring of a model is done with a configuration. The configuration is created depending on the preferences of the user. In order to collect these preferences a questionnaire is used. The questionnaire that is used to collect the preferences isn't an ordinary questionnaire but creates immediately after answering of the questions a file with the configuration that suits the answering of the user. The questionnaire is created in this phase.

Phase 6

When *Phase 4* is completed we have a combined YAWL model that can be configured with the configuration of *Phase 5*, except there is a link missing. The missing link is the mapping. The mapping represents the translation of answers of the questionnaires into blocking, hiding or enabling activities in the model. The mapping consists of several constraints in order to eliminate unwanted behavior.

When all phases are executed the result is a configurable process model. Before we describe the phases in more detail (chapter 3) we discuss the preliminaries in chapter 2.

To improve the readability of the remainder of the thesis we will briefly describe the frequent occurring terms in the *Glossary* that is located behind *references* section.

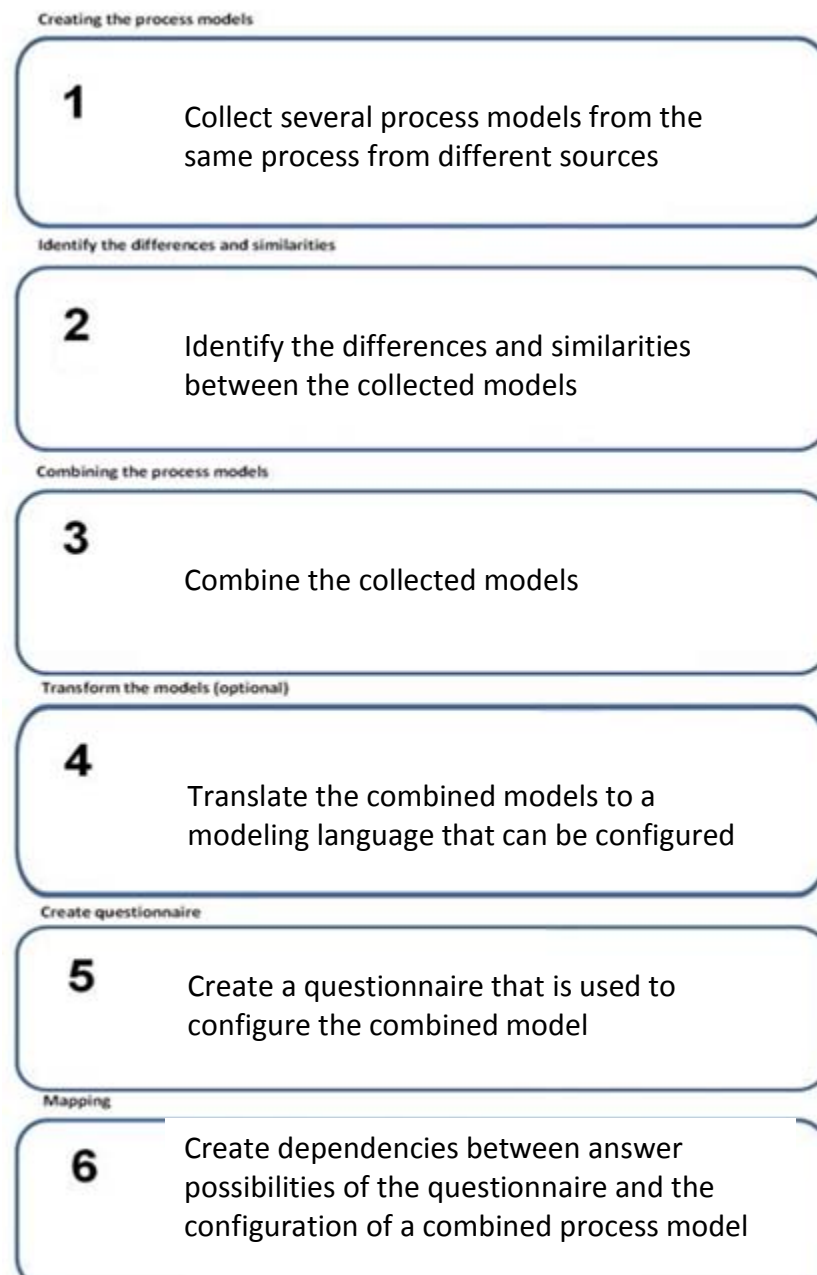


Fig. 1-1 Phases to create configurable workflow models

1.3 Outline

The remainder of the thesis is structured as follows.

Chapter 2 – *Preliminaries* – a description of the theories that are extensively used during our case study. Consisting of the business process language, the workflow language, the configurable approach, and the questionnaire approach.

Chapter 3 – *Configurable workflow models: a Case study* – in this chapter the complete process of creating a configurable workflow model from scratch is described. From the starting point of collecting the data, to combining the business process models, transforming the models to configurable YAWL model, creating the questionnaire, and the mapping.

Chapter 4 – *Results* – evaluates the observations we did during the case study and the results of the evaluation with the stakeholders.

Chapter 5 – *Discussion* – discusses the results of the observations and the evaluation with the stakeholders and relates them to the contribution of this thesis.

2 Preliminaries

This chapter gives an overview of the background of this research. In the introduction of the thesis several subjects are described and the core subjects of the thesis are extensively described in this chapter. The first section of this chapter describes the business process modeling tool that is used to create the business models, followed by the second section which the workflow language that is used during the case study is described. The third part of this chapter discusses the configurable workflow language and is followed by the theory related to the questionnaire. The closing section of this chapter describes the context of our research.

2.1 Protos

Protos is a business process modeling tool that is developed by Pallas Athena [10]. We choose to use this business process modeling tool because it is commonly used in Dutch municipalities. The tool is used to depict business process models in an orderly fashion. The symbols that are used to depict the business model are presented in Figure 2-1. Business process models describe work processes by depicting *states* and

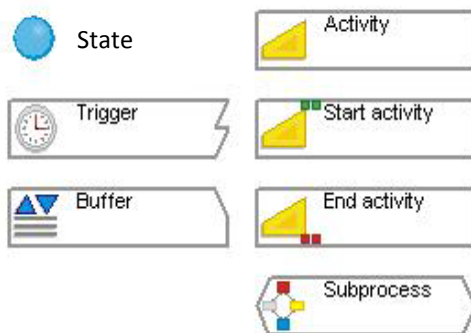


Fig. 2-1 Symbols used in Protos

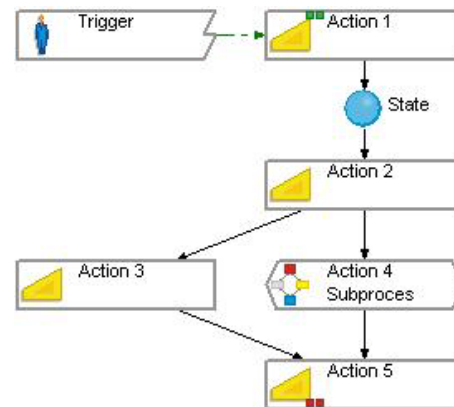


Fig. 2-2 Example of a Protos business process model

actions. The actions are displayed in Protos by rectangles and the states by a circle. The actions are indicating what activity should be executed at that moment in the process. So whenever an activity must be executed during a process, it will be displayed as a rectangle in the workflow model. Conversely the states are indicating a neutral state between activities in a workflow model and are giving an indication in

which state the process is. When the process is in a state it is waiting for the next action to be executed. By combining and connecting these two types of expressions a workflow model can be created. The connections between actions and/ or states are done by arcs. Besides the rectangles and circles Protos uses hexagons to depict sub-processes, a rectangle with two points to depict triggers, and a rectangle with a missing corner to depict a buffer. This way a business process model can be enhanced. These types of symbols are the ones we used in our business process models for the municipalities. An example of a business process model in Protos is depicted in Figure 2-2.

2.1.1 Protos behavior

The business process model starts with *Action 1* (Figure 2-2) that will start when it is initiated by the *Trigger*, in this example the trigger is a person. When *Action 1* is completed the process will continue to *state* and wait until *Action 2* needs to be executed. After executing *Action 2* it is not clearly depicted how the process should continue. It could be that both *Action 3* and *Action 4 Subprocess* need to be executed or only one of the two. By looking at the presentation of the model it does not show which option applies for this model. If an action has two or more outgoing arcs/arrows and if those arcs are then reunited in the following state, it implies it is a choice. This is not the case in Figure 2-2 but this does not imply both must be executed in parallel. The only way to determine what the flow possibilities are after *Action 2* is by consulting the properties of the action in the Protos tool. In Protos it can be specified what the flow possibilities are but this is not graphically depicted in the model.

2.2 YAWL

The workflow language YAWL (Yet Another Workflow Language) [11] is used during the case study to create the configurable workflow models. YAWL was selected because it is distributed using an open-source license. Because it is open-source we have access to the source-code and have the possibility to create applications that can cooperate with YAWL. We will use this in a later stadium to automatically configure YAWL models. Therefore YAWL was preferable to other modeling languages. YAWL consists of an editor and an engine. The YAWL-editor is used to create business process models. The business process models that are created in the

YAWL-editor can be used as a workflow model, if all the needed data is specified. When the business process models in the YAWL-editor are adjusted and all needed data is present, they can be loaded in the YAWL-engine. It is also possible to transform Protos models into workflow models but because we do not apply this in our case study we won't discuss this subject. In the next section the behavior possibilities of the business process models in the YAWL-editor will be described. The transformation from a business process model to a workflow model will also be described, followed by a description of the YAWL-engine.

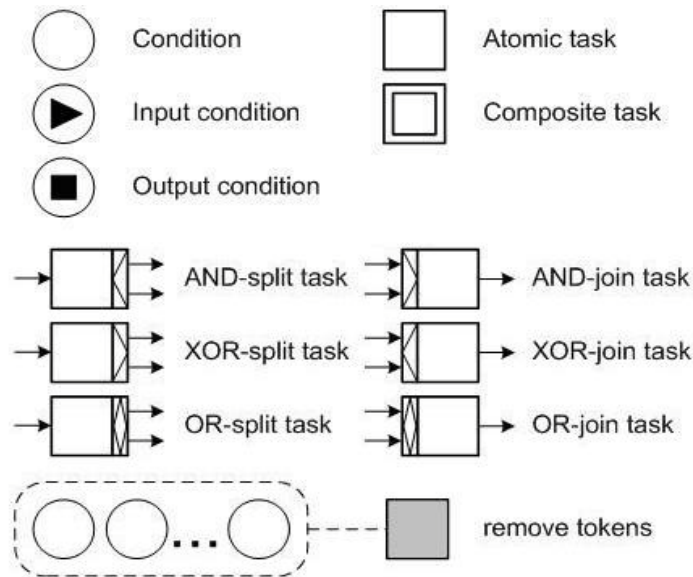


Fig. 2-3 Symbols used in YAWL [11]

2.2.1 YAWL behavior

YAWL has, like Protos, symbols for an action and for a state only it's called differently namely a *task* and a *condition*. In Figure 2-3 symbols that can be used in YAWL are displayed. By combining these symbols a business process can be modeled like in Protos. Unlike Protos the behavior of a task is graphically depicted. The symbols AND-, OR-, and XOR-joins and -splits determine the behavior of the process. Figure 2-4 shows an overview of the three types of splitting and joining tasks. At first glance the three models look the same only there is a fundamental difference in their behavior. The top model displays XOR-splitting and -joining behavior. This causes that there can only be three outcomes namely, (A,B,E) , (A,C,E) , or (A,D,E) . The XOR-split task *A* indicates that an explicit choice needs to be made after task *A* telling which of the three following tasks should follow. Because there are three tasks

there are also three possible outcomes for this particular model. Depending on the choice that is made after task A the outcome is defined. So if task D is chosen to succeed task A the outcome will be (A,D,E). No other options possibilities in this model are allowed. Like the XOR-split task A that can trigger only one of the succeeding tasks, the XOR-join task E can only be triggered by one of the preceding tasks. The middle model shows the AND-splitting and -joining behavior. Instead of three possible outcomes, it has only one outcome because all succeeding tasks of task A must be executed when task A is finished. Tasks B, C, and D will be executed in parallel. The AND-join task E will synchronize the three tasks. So task E can only execute when all preceding tasks are finished. There is still some flexibility in the order of executing tasks B,C,D. It does not matter in which order they are executed as long as they are executed. In Figure 2-4 the B, C, and D are between brackets because the order of execution does not matter. For example the outcome can be A, C, B, D, E or A, D, C, B, E as long as, for this particular example, the first task is A and the end task is E. The bottom model represents the OR-splitting and -joining behavior. The OR is the least restricted behavior because the only restriction is that

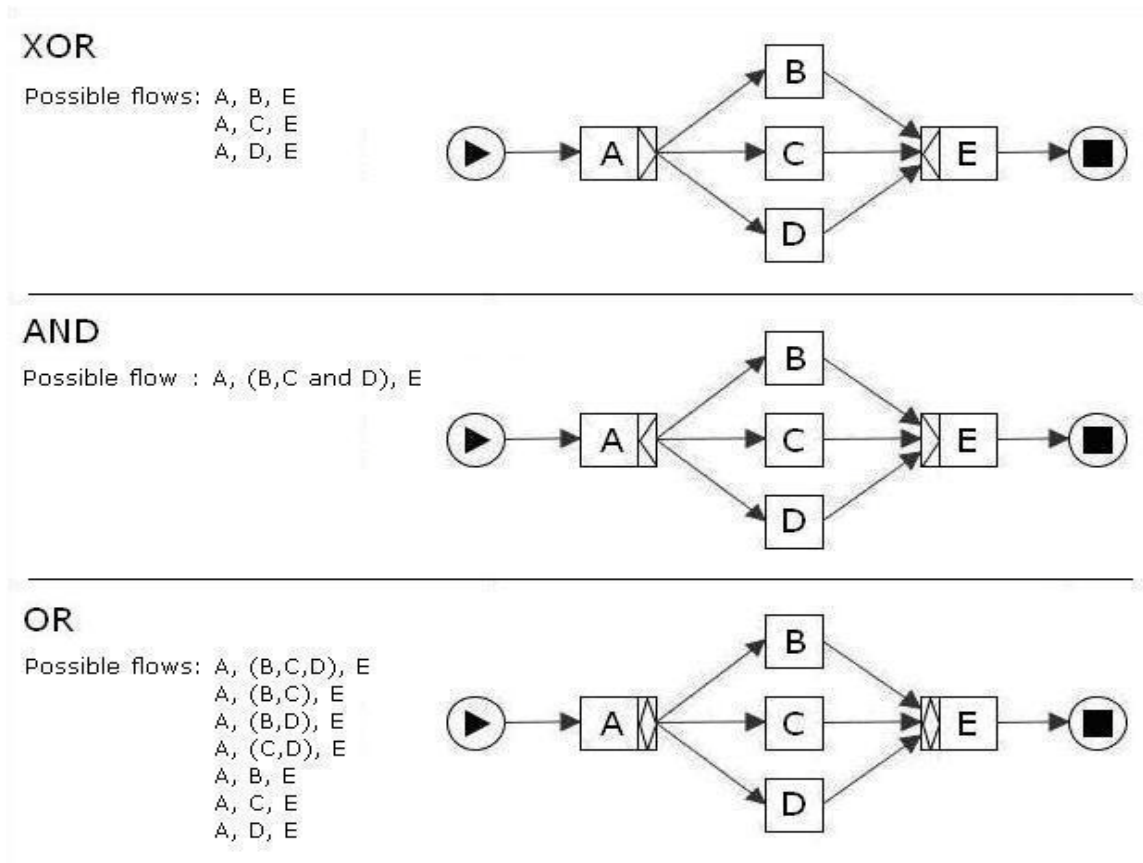


Fig. 2-4 XOR-, AND-, OR- behavior

at least one in- or output port should be triggered. This means the OR can have the behavior of an AND or a XOR. So the OR-split of task A has several possibilities to trigger. It can trigger all succeeding task or only one, but it can also trigger two succeeding tasks as long as it triggers at least one task. In total there are in total seven possibilities to trigger for task A. When the tasks are triggered by task A everything is synchronized by OR-join task E. In YAWL the OR-join checks any possibility that can enter the OR-join and when all possibilities have arrived the process will continue [12]. This is also referred to as the 'bus-driver semantics' [13], who is, the OR-join is like a bus driver that has to make a decision each time a passenger enters the bus. The bus-driver will start to drive to the next stop when he expects that all passengers entered the bus. The bus driver is never sure all potential passengers entered the bus. The OR-join works in a similar way.

So far the behavioral possibilities of YAWL are discussed which are also possible in Protos. The next pattern that will be discussed is not possible in Protos. The cancellation of several tasks can be done in YAWL with a cancellation task that is connected to a cancellation region. The symbol that is used to depict this behavior is the remove tokens task. When the remove tokens task is executed all tokens that are in the connected cancellation region will be removed. A token is an indication at which point the execution of a process is. The behavior possibilities of YAWL which are discussed are just a subset of all the possible behavior in YAWL but are a good base to read the thesis.

2.2.2 Data perspective

When a business process model is created in the YAWL-editor, it cannot be transformed into a workflow model by clicking on a button. Instead the data that is needed to automate the workflow model must be specified. First the *Net Variables* must be specified to create the basis of the workflow model. The net variables are needed to store all information relating to the workflow model that a task within the workflow model may need to read or update the information [14]. Some of the options that can be chosen for a net variable are options as *input*, *output*, *local*, or as *input & output*. These options indicate how the net variable is used. For example when the option *input* is selected then the net variable can only be used as input for a task. The *output* option is the contrary of the *input* option. If the usage of the net variable is set to *local* the information is stored locally and can be used in every situation. Besides the usages of the net variable the *type* of net variable must also

be specified. If the net variable should be some kind of text the *string* type should be selected and if it should be a yes or no answer the *Boolean* type should be selected. In this way it is specified how the net variable should be used by tasks within the workflow model.

Besides the net variable, the *task variables* must also be specified. In a way the same thing occurs as for the net variables, only now for a specific task i.e. net variables can be used by several tasks within the workflow while task variables only can be used for one specific task. Task variables have several uses. One use is transferring information between workflow users and the workflow engine. A second use is passing data between web services and/or external code and/or applications that the running workflow engine invokes and the workflow the task resides in [14].

To link the net variables to the task variables, task parameters are used. There are two kinds of task parameters. The first is the *input parameter* to specify which parameters should be invoked as input for a task. The second is the *output parameter* that logically specifies which parameters should be invoked as output for a task.

The data flow of a task should also be specified in case the task has a XOR-split or an OR-split. The workflow engine should know what flow should be activated. By updating the flow detail, the default path can be specified and also establish predicates for each flow direction [14]. When the predicate of a flow is evaluated to true in the workflow engine, its flow will be executed.

2.2.3 YAWL-engine

When all data that are needed for the workflow engine is specified the model can be loaded into the *YAWL engine*. The YAWL engine is the workflow-engine of YAWL. It gives a different representation of the workflow than the YAWL editor. The YAWL editor gives a static overview while the YAWL engine is dynamic. The user can interact by means of an interface that is depicted by the YAWL engine. The YAWL engine can also interact between tasks and web services. In this way the YAWL engine manages the data flow for the whole process and simultaneously guides the user through the process. In Figure 2-5 a representation of a YAWL worklist is depicted.



Fig. 2-5 The representation of a possible form for a task

2.3 Configurable workflow models

Configurable workflow models are an approach to adapt workflow models to a particular situation. In this paragraph the approach of Gottschalk et al. [6] will be described. The paper describes the configurable approach that we are evaluating in this thesis, by means of a case study.

2.3.1 The configurable approach

The configurable workflow language is used to derive an individual workflow model from a more general model. The general model includes several options which are not mandatory and options to execute different paths. To specify which parts of the general workflow model are needed for the individual workflow language, Gottschalk et al. identified two generally applicable methodologies to configure a workflow model. These methodologies are blocking and hiding. They are both of a restrictive nature. Besides this similarity there is also an important difference between the two. If an action is blocked it cannot be executed and therefore it will not be possible to execute any of the subsequent actions. This is unwanted behavior and therefore all subsequent actions that are not reachable anymore by any other path in the

workflow model must also be blocked. If an action is hidden, it is not executable but the subsequent actions still are. This means that the hidden action is skipped, but the process flow will continue normally afterwards. The time and resources the hidden action normally consumes will not be executed and therefore the hidden action is not noticeable. So when a workflow model will be configured it should be taken into account if an action is mandatory for the subsequent actions. When an action is mandatory it cannot be blocked or hidden because it would then result in unwanted behavior in the process. When configuring they should also anticipate if the following actions are needed. If the subsequent actions are not needed, the first action in the order of unwanted actions should be blocked. Conversely if only one action is unwanted but the subsequent actions are wanted, the unwanted action should be hidden. Besides the restrictive possibilities blocking and hiding there is also a possibility that allows activities to execute. If an action is executable it is called enabled, this indicates that the action will function like a normal action.

2.3.2 Configurable workflow model behavior

An action needs to be triggered to execute. Triggering an action is different from enabling an action. The enabling of an action is specified during the configuration of a workflow model. If an action is enabled it is possible that it will occur at run time of the workflow model. Contrarily when an action is disabled during the configuration it cannot occur during run time. The triggering of an action occurs during the run time when the workflow changes from state i.e. when for example the process goes from action A to action B. In most workflow languages this trigger activity is represented by an arc pointing into the action. There is a difference between the number of arcs needed to trigger an action. It can be that several arcs are needed, only one or a number. To identify which of these cases is occurring there are three different main trigger possibilities; AND, XOR, and OR. The AND implies that all arcs should be triggered to execute the action. Another possibility is to only allow to trigger the action by just one of the arcs, this is called an XOR. The XOR means exclusive OR that implies only one of the options can occur. The third trigger option is the OR, and implies at least one of the arcs should trigger action. This trigger option combines the possibilities of the XOR and the AND.

Because an action has an inflow and outflow the possibility to trigger can be on two sides, either the inflow and/or the outflow of the action. If one of the trigger possibilities is located at the inflow side of the action it is called a join and if it is located on the outflow it is called a split. To configure an action the incoming and outgoing flows should be set to blocked, hidden or enabled. In order to do this ports are used. The actions in Figure 2-6 are showing the number of ports depending on the number of in and outflow and the trigger possibility. For AND-joins only one port is used because the action is only triggered when all arcs have reached the inflow port. By blocking or hiding the incoming port of AND-join all incoming arcs are blocked or forwarded depending on the configuration. The same applies for the AND-split, at least for the blocking behavior not for the hiding behavior. The outgoing port cannot be hidden because it triggers paths instead of actions and if the following action should be hidden it should be configured at the inflow port of that action.

For the XOR trigger option several ports are needed because the action can be triggered by only one of the ports. To know which ports need to be enabled each individual port can be configured. In comparison with the AND trigger option, the inflow port for the XOR is called a XOR-join and for the outflow port a XOR-split. In Figure 2-6 the two actions depict the difference between the XOR and the AND ports. Here the difference between the XOR-ports and AND-ports is clearly to be seen. The ports are the actual configuration points of the actions. As depicted with the two actions in the Figure 2-7 the ports are configured. The right top action has the AND-

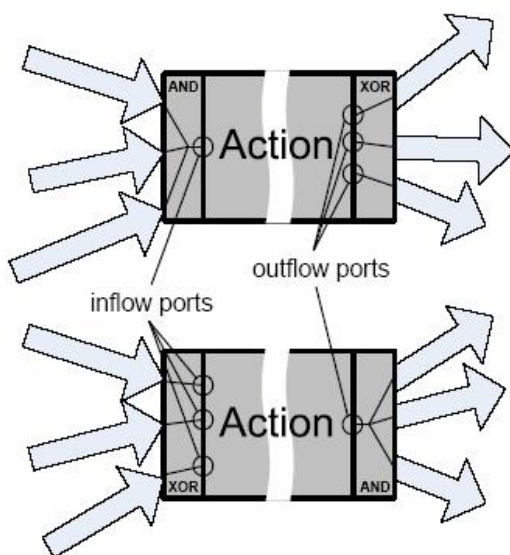


Fig. 2-6 The number of ports of an action depends on its joining and splitting behavior [6]

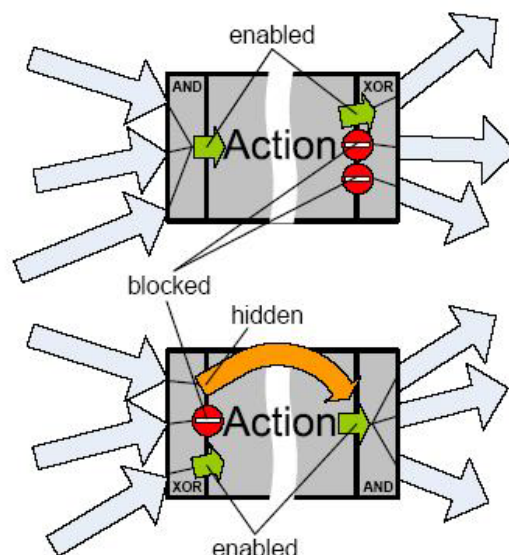


Fig. 2-7 Ports can be activated or blocked, and in case of an inflow also be hidden [6]

join enabled and the two bottom XOR-split ports are also blocked. Because the AND-join is enabled and only the top outflow port is enabled the action is triggered when all incoming arcs are completed and after executing the action it will trigger the top outflow port. The bottom right action shows, with the curved arrow, the hidden trigger, how it skips the action and directly flows to the outflow port. If the blocked arrow is triggered the action won't execute, only when the bottom enabled arrow is triggered, the action will execute. In this is example there are three possibilities for this action namely; the inflow port of the action is hidden and the action is not executed but will continue to its output flow, the inflow port is blocked and the action is not executed and will not continue the outflow, or the inflow port is enabled and the action is executed and continues the flow. It is mandatory to have at least one outflow port enabled when one of the inflow ports is enabled or hidden. Because in those two cases the inflow ports accost the outflow ports and when the outflow ports are all blocked, it will give unwanted behavior. Through making the inflow port enabled or hidden and the outflow port blocked it will give a deadlock because no transition can be done to a following action or state. Therefore all inflow ports should be blocked when all outflow ports are blocked.

In Figure 2-8 there are three different configurations of the same configurable process model. The process without configuration is straightforward, *A* is followed by *B* and the outflow ports of *B* is a XOR-split and the inflow ports of *E* are a XOR-join. The processes that can occur in the first example are (B,C,E) and (B,D,E) . The inflow port of *A* is hidden and therefore *A* will not be executed. Instead of *A* as starting point *B* will be the starting point and will flow into *C* or *D* depending on the run-time decision that is made at *B*. After the execution of *C* or *D* the process will continue to *E* and will end. In the second example also one port is configured only this time it's not the inflow port of *A* but the outflow port of *B* to *D*. The outflow port of *B* to *D* is in this case blocked. Therefore *D* is not reachable anymore and cannot be part of the process flow. In spite of the fact that all ports of *D* are enabled it cannot be executed because the inflow port will never be triggered by the outflow port of *B*. The only possibility for this process is the flow (A,B,C,E) . In the third example the configuration caused undesired behavior because the process cannot finish. This undesired behavior is caused by blocking the outflow port of *A* that results in a flow stop. So the inflow port of *B* cannot be triggered and also all the successors of *B*. Therefore it does not matter that *B,C,D* and *E* are configured because they will never

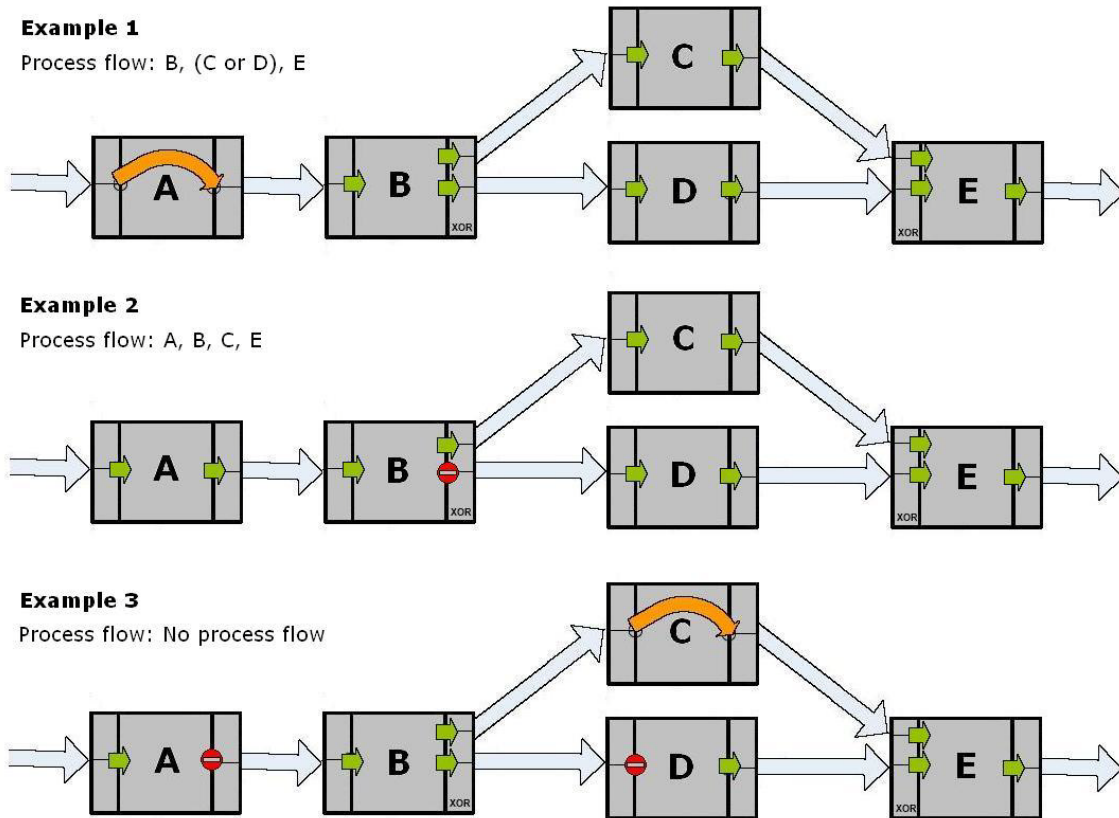


Fig. 2-8 Three examples of configuring a configurable process model

be reached. By looking at the three examples it shows that by configuring the configurable model in different ways different process flows are created. The third example shows that when the configuration is applied in the wrong way it can result in a model that is not executable anymore. Because configuration is enforcing restrictions onto the model it implies that the model without restrictions is the base model (at the beginning of this section the base model is called the general model). The base model is the least restricted configuration where everything is allowed (fig. 2-9.). By configuring the model choices are made between which actions are desired and which are not. It can be that the model has unwanted behavior when it is not configured. For example it is mandatory to choose between two actions because if both actions are enabled the process will have unwanted behavior. The opposite can also give undesired behavior when too many actions are disabled. If mandatory actions are blocked or hidden when they should be enabled the process won't reach the desired final result (Fig. 2-8. Example 3). To stop this from occurring there can be some constraints to keep this from happening. The constraints make sure the configurable model can be configured without unwanted behavior.

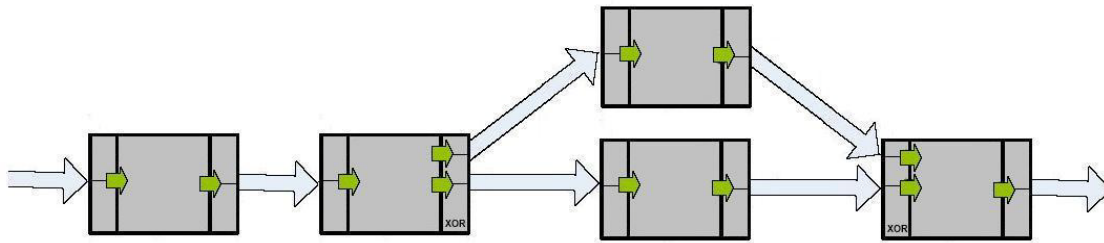


Fig. 2-9 When a model is completely enabled it is the base model

2.3.3 Configurable YAWL

We described the configurable approach and the behavior of configurable workflow models in the previous section. By applying the configurable approach to YAWL Configurable YAWL (C-YAWL) is generated. By adding in- and output ports to YAWL we can configure YAWL by enabling, hiding, or blocking the ports. In [6] C-YAWL is used to make YAWL configurable. In Figure 2-10 an OR-split is configured by blocking several output ports and only enabling the output ports b and b,c . This results in the right *Task B* in Figure 2-10 where only output port b and c are reachable by *Task B*. In this situation the OR-split will still be an OR-split when it is configured. In Figure 2-11 the configuration causes that the OR-split of *Task C* is transformed into a XOR-split. Here the only two enabled output ports are b and d . Because only one of the two can be chosen the OR-split is transformed into a XOR-split. In the same way we can configure the OR-split to transform into an AND-split. An AND-split has the property of triggering all outgoing arcs. In Figure 2-12 the only output port which is enabled is b,c it causes that both these ports must be triggered, all other output ports are blocked. The result is depicted in the right *task D* of Figure 2-12. In the same way we configured the output ports of the YAWL tasks we can configure the input ports. The only difference is that the input ports also can be hidden and the output ports cannot. During the case study we use YAWL with the addition of the C-YAWL functionality.

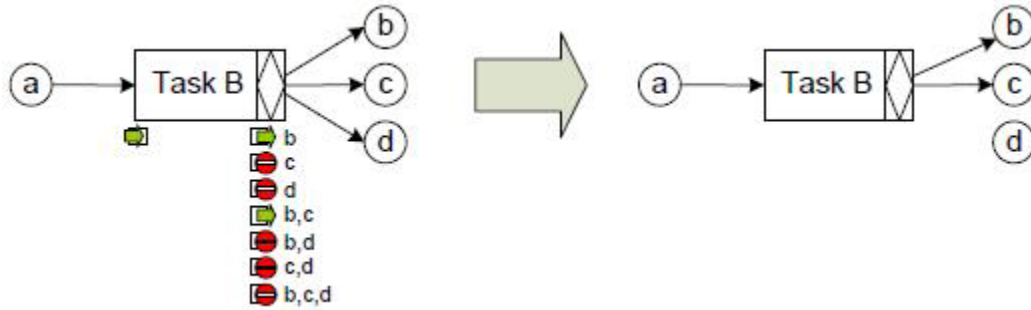


Fig. 2-10 Transforming the output port into YAWL model [6]

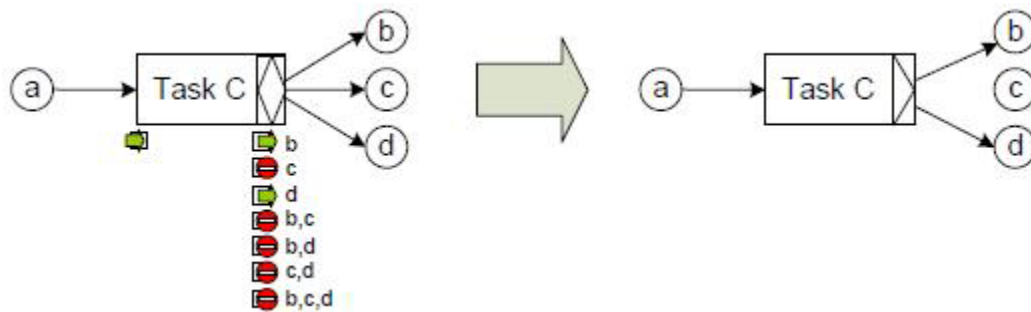


Fig. 2-11 Transforming of an OR-split into an XOR-split [6]

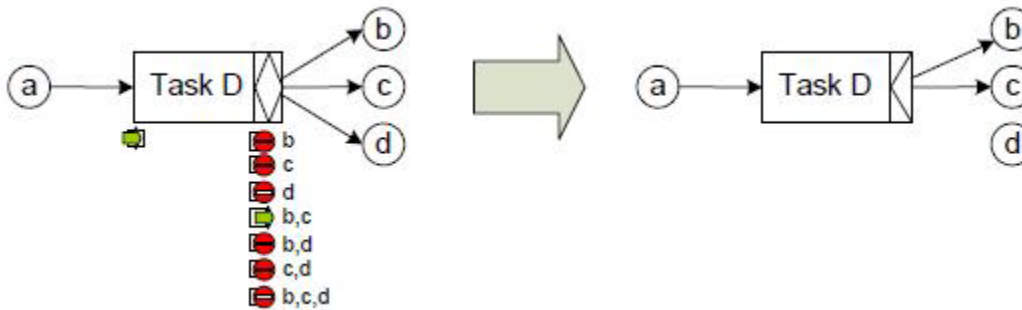


Fig. 2-12 Transformation of an OR-split into an AND-split [6]

2.4 Questionnaire

Variability of an information or software system may be captured as a collection of parameters [15] as a collection of features [16], or more generally, as a collection of choices [17]. These choices determine which actions are enabled, which are disabled, and which are hidden. To make sure that the choices that are made do not cause invalid configurations and are made on the right moment the user should be guided

[17, 18]. La Rosa et al. created an approach to guide the user through the choices. This approach will be described in this paragraph.

2.4.1 Questionnaire behavior

La Rosa et al. [17] proposed a technique to generate interactive questionnaires from question-facts structures like Figure 2-13. These questionnaires guide the configuration process by posing relevant questions in an order consistent with the dependencies between questions and facts, and also in a way that prevents the violation of the domain constraints.

The questionnaire consists of question and answer possibilities that are in line with the question-facts structure. A configurable model can be configured with blocking and hiding of tasks. To make sure that the result is a valid configuration, only tasks may be configured in a way they don't cause invalid configuration. Therefore the question should relate to the tasks that does not endanger a valid configuration. In Figure 2-13 an example of a question-facts structure is depicted. Every question (rectangles with a q) has a number of answer possibilities called facts (rectangles with a f). To know which fact belongs to which question a line is drawn between them. For example question $q1$, where is asked if one wants to check if the informer and the mother both are not married has two answer options namely $f1$ and $f2$. If the user decides to choose fact $f2$ instead of $f1$, $f2$ will be set to *true* and $f1$ will be set to *false*. These values are needed to map the answers on the configurable model. In the case $f2$ is true the specified incoming ports of a task in the workflow model should be blocked. Conversely if $f1$ is true the incoming ports of the task in the workflow model should be enabled. In this way the users configure the workflow model by answering questions without knowing the direct consequence for the model. To make sure the questions will be asked in a logical order, there are *order dependencies*. In the case of Figure 2-13 $q1$ and $q3$ can only be answered after $q2$ is answered. "We express such dependencies by associating a set of alternative preconditions with a fact x , where a precondition is a group of facts that all need to be set before x . Only one precondition needs to be satisfied for a dependency to be fulfilled. Therefore, fact x can be set only if at least all the facts in one of its preconditions have already been set. We say a fact partially depends on another fact if the latter belongs to at least one of its preconditions. On the other hand, a fact fully depends on another one if the latter belongs to all its preconditions. A full

dependency subsumes a partial dependency” [17]. In the example there occur no

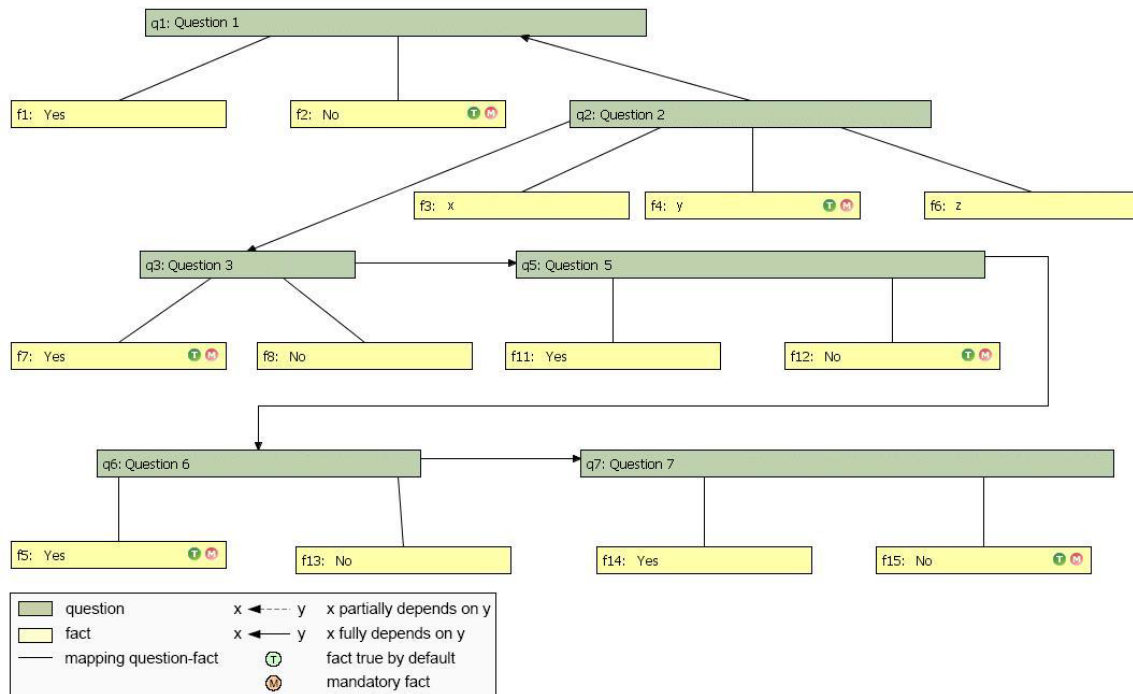


Fig. 2-13 Example: structure of question-facts. Legend [17]

partially dependencies. If a partial dependency would occur it will be depicted by a dashed arrow. The full dependency is depicted with a solid arrow.

In Figure 2-13 the dependencies are all drawn, by means of arcs, between questions but it can also be the case that there are dependencies between facts. Dependencies between facts are needed when an answer to a question has effect on succeeding questions. For example in Figure 2-14 facts $f3$ and $f4$ are depending on fact $f2$. In the case $f2$ is *true* question $q2$ will be enabled but conversely if $f2$ is *false* question $q2$ will be disabled.

To make sure the questionnaire has the same behavior as modeled in the question-facts structure constraints are needed. In the case of the situation depicted in Figure 2-14 a constraint is needed that expresses that $f3$ and $f4$ may only occur if $f2$ is true. The constraint will be modeled as a propositional logic expression. The following constraint expresses the wanted behavior for Figure 2-14: $(f2 \Rightarrow (xor(f3, f4)))$. Another example is a constraint for $f1$ and $f2$ in Figure 2-13. The fact $f1$ represents the answer *yes* and fact $f2$ represents the answer *no*. To make sure only $f1$ or $f2$ can be true and not both, a constraint is needed. In this case the constraint needs to express that the answer possibility is of the kind *XOR*. The constraint should

be $((f1 \square \square f2) \square (\square f1 \square f2))$ for this particular question. In this case the user does not have the possibility to choose both facts. When the question does not have a constraint that specifies it should be a XOR-question it automatically will be an OR-question.

Some questions need to be mandatory because of the important choice they represent. The facts of these questions will be labeled with a \textcircled{M} symbol, so they have to be explicitly set to *true* or *false* when answering the question. Every question also has a default answer that represent the most common or logical decision. The default answer are labeled with \textcircled{T} that indicates the facts' default=*true*, while no symbol means the default=*false*. Until now the technique was discussed how to create the questionnaire by means of questions and facts but the user will have a different interface when he is answering the questions. In Appendix 1 a depiction and a description of the interface of the questionnaire located.

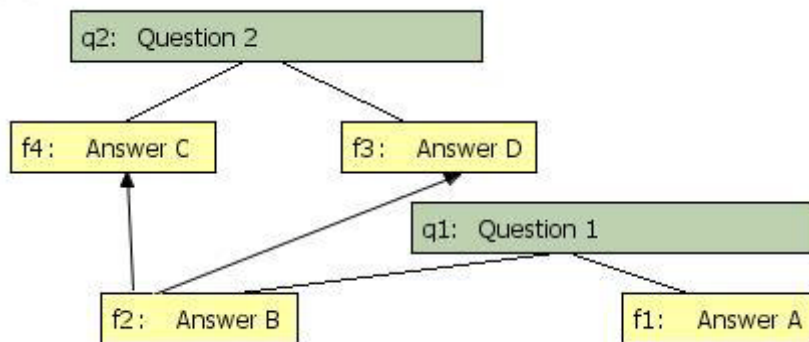


Fig. 2-14 Fact f3 and f4 depend on the fact f2 to be true

2.5 Mapping

The link between the questionnaire of Section 2.4 and the configurable workflow models from 2.3 needs to be created. The translation from the answer of the questionnaire to a configured model is done with two applications. The first application is the *Configurator* [19]. This application configures the configurable workflow model according to the results of the questionnaire. The second application is the *Individualizer* [19]. This application cleans the configurable workflow model that is configured with the *Configurator*. By cleaning we mean that all arcs, conditions, and tasks that are not needed, after the configuration will be removed from the model. If an output-port of a task is blocked, the arc towards the

succeeding task is removed. When this arc is removed it can be the case the succeeding task is not reachable anymore. The unreachable tasks must be removed from the configurable workflow model. The removing of all redundant tasks, conditions, and arcs is done by the *Individualizer*. In Appendix 2 depiction of both tools and description are presented.

2.6 Related work

Until now we described the literature that is intensively used during the case study. In this section we discuss how our work is related to other literature to place the research in context. Our research is related to several areas like identifying configuring models, differences and similarities, case studies, and reference models.

Case study

A case study that is similar in a way to our case study because it is comparing business processes of different municipalities is Algemissen et al. [20]. Algemissen et al. performed a case study to present experiences in applying reference modeling within public administrations [20]. Like [20] we perform a case study with municipalities, only we want to collect differences to create a configurable workflow model and [20] is focusing on creating one reference model for each process. In a way they are doing the same as the NVVB. The NVVB is an umbrella organization of municipalities that creates and maintains reference models for municipalities in consultation with experts from municipalities [3]. Because municipalities are customers of the NVVB they have access to knowledge of several processes at municipalities that they use to create reference models. The reference models of the NVVB are used as a starting point for the configurable model. They also provide a picture of how processes are executed at the municipalities and we used that as a preparation of the data collection at the municipalities. Like the NVVB reference models, the template repositories of SAP could also be labeled as best-practice reference models. Best practice reference models are also the goal of several scientific case studies. For example, Thomas et al. developed a reference model for event management [21], Prikladnicki et al. developed a reference model for global software development [22], and Scheer designed a reference model for industrial enterprises [23]

Another case study that is related to our case study is the case study performed by Seidel et al. about developing a business process reference model for the screen business [24]. This case study resulted in a running example for the configuration of reference process models. The configuration was done by a questionnaire-driven approach. We use the same kind of questionnaire-driven approach to configure only we configure YAWL instead of Event-driven Process Chains.

Configurable models

Some earlier work on process configuration is used during this case study [6, 25, 26, 27, 17]. These papers look into several different ways to configuring. Like [6] discusses the configurable approach of blocking and hiding and applied it to YAWL models. In [25] the configurable approach is tested to EPC and in [27] it is applied to SAP's workflow templates. In these papers configurable approach was tested in theory, the case study we executed tests the configurable approach in practice. We created the configurable workflow models from scratch by interviewing the process owners, [17] discussing the possibility of collecting business process models and creating configurable reference models by means of process and data mining. We won't use the possibility of process mining during our case study because it is outside our scope, but it is interesting for future research.

Becker et al. also suggested configurative modeling by means of design parameters [28, 29]. Depending on the use of the model it can be adapted to a specific user group. For example it can be adapted to the perspective of organizational design or to the perspective of application design. Depending on the perspective the model will be configured.

Differences and similarities

To configure a configurable process model, configuration points are needed. A configuration point is a part of a configurable process model that can be activated or deactivated. To determine equivalence between activities of different models, [30] checks if the unit of work that they represent is equivalent for the purpose of the integration based on two criteria:

- The effect that the units of work will have in the integrated process must be the same.
- The way in which the effect is achieved must be the same.

Dijkman [30] determines the equivalence of activities amongst other activities on the basis of the roles that execute the activities. The roles are outside the scope of our research and therefore we will not look at the roles to determine equivalence. Our focus will also be on the first criteria and not on the second. These criteria are created by [30] to merge departments after a merger of two companies and in that case it is important to know if the tasks are executed with support of a specific information system. Our focus is to create an approach for configurable workflow models so that a software developer can offer his customers a customized solution. So the information system that is used in the business process models should be the software developers' one.

The configuration points are created by looking at the differences between process models that were combined to one model. Dijkman provided a theory to identify the differences and similarities between process models [30]. The theory was intended to merge process models from companies that were merged but could also be applied to create configurable process models. The work from [31, 32, 33, 34] is focused on merging business processes in spite of the differences between the processes.

Unlike [30] there is also an approach of Ehrig et al. [35] to measure the similarity between business process models.

The authors proposed an approach for (semi-) automatic detection of synonyms and homonyms of process element names in order to measure the similarity between business process models. Van Dongen et al. also proposed an approach to calculate the similarities between two business process models [36]. Both approaches from [35] and [36] do not give an exact similarity but a degree of similarity. To create a configurable process model the tasks that are merged to one task should be the same. A degree of similarity gives an indication but no certainty. Both theories of [35] and [36] are based on parameters that are interpreted by (semi-) automatically leaving room for misinterpretation that can cause unwanted behavior in a configurable process model when tasks are merged that shouldn't be merged. These theories can be useful for automatic creation of combined process models that could serve as basis for configurable process models if the uncertainty is eliminated. Because both approaches from [35] and [36] do not give an exact similarity but a degree of similarity we choose to use the approach from [30].

[30] makes a distinction between three types of equivalence types. The first type of equivalence is when both criteria mentioned above are met. The second type holds

when the first criterion is met and the third type indicates equivalence between collections. In our case the first two types will be seen as the same because the second criterion is based on the roles and execution and that is outside the scope.

Inheritance

When combining models behavior and relations are inherited. The work of [37],[38], and [39] distinguish four inheritance rules:

- Protocol inheritance:
- Projection inheritance
- Protocol/ projection inheritance
- Life-cycle inheritance

The paper of [38] studies inheritance of behavior both in a simple process-algebraic setting and in a Petri-net framework. The theories of [37], [38] and [39] gave the insight that blocking and hiding are the two basic configuration operations.

[40] looked into configurable reference models and specified configuration rules for connectors. These theories are used to create configurable process models from several existing process models.

Questionnaire

The questionnaire-driven configuration of [41] is used to configure process models allowing to configure the configurable process model without confronting the person that answers the questions with the model. By answering the questions the configurable process model will adapt accordingly.

The use of questionnaires to guide the configuration is used in a similar setting for software applications. For example, the CML2 language is designed to support Linux kernel configuration by turning the user's answers to configuration questions into a file of `#define` constructs used to condition features in or out of the C code [42]. More generally, variability of 13 large software systems has been studied in the field of Software Product Line Engineering (SPLE) [43].

The theories that we discussed in chapter 2 are used during the case study. In chapter three we describe how we executed the case study. During description we will refer to literature of chapter 2 that we used.

3 Configurable Process models: a Case study

The case study is executed in several phases that will be described in this chapter and can be used as a guidance to create configurable process models. The phases are briefly discussed in chapter 1 and are described in more detail in the current chapter. Figure 3.1 is used as guidance through this chapter. When a phase is discussed the related phase is highlighted in the figure. The outline of this chapter is structured like the phases in Figure 3.1 and is accordingly discussed.

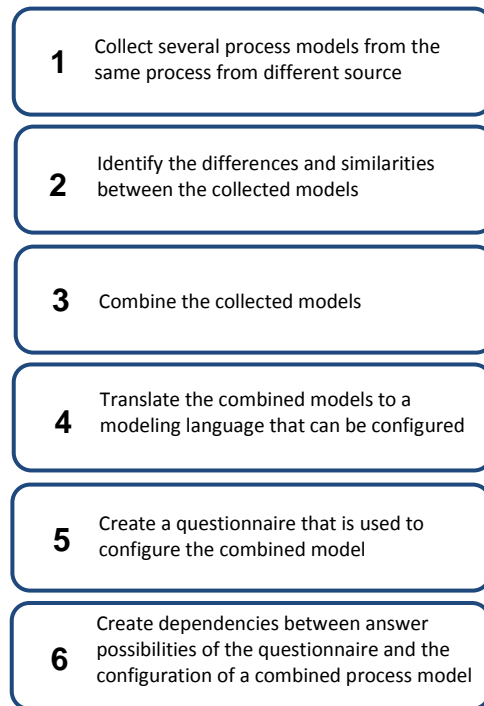


Fig. 3-1 Phases to create configurable workflow models used as guidance

3.1 Collecting processes

In order to investigate if it is feasible to create configurable workflow models, models are needed. In this case the models consist of business process models of municipalities. To determine which processes are most suitable for the investigation some considerations were made. The next subparagraphs will describe how the processes are chosen and how the processes are executed (*Phase 1*, Figure 3.2).

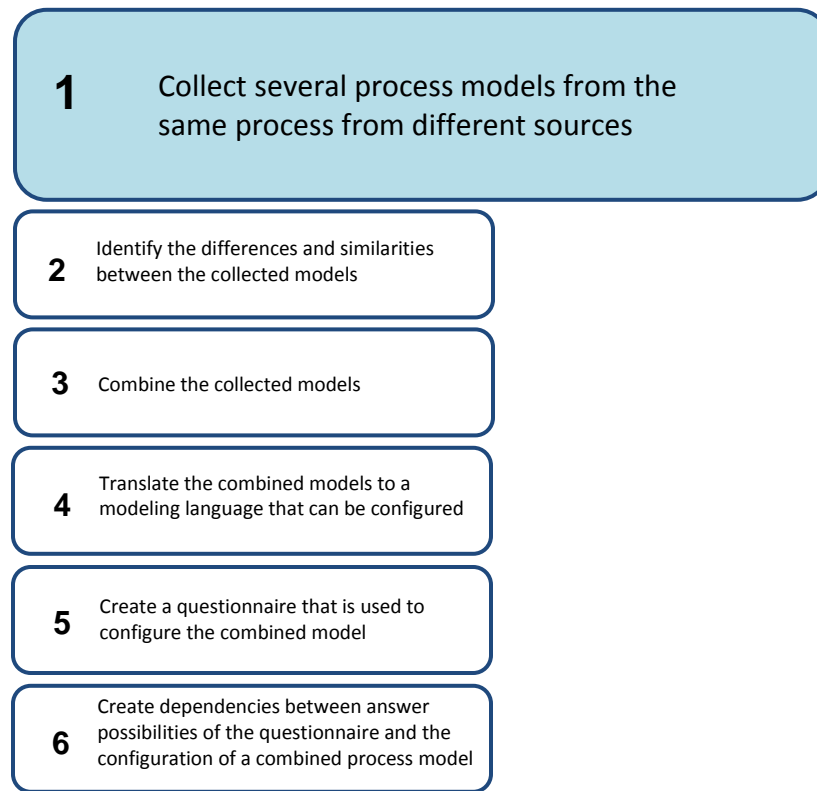


Fig. 3-2 First phase: Collecting process models

3.1.1 Selecting the processes for the case study

During the data collection phase the decision was made which processes at the municipality will be investigated. The factors which were taken into account by this decision are:

- have similarities amongst all municipalities.
- the processes should frequently occur in municipalities
- the number of processes

Similarity

It is not useful to compare processes that are that different that only the start and end state are the same. For example a request for social security differs very much per municipality. In this situation the reusability and the maintainability of the configurable workflow do not have much advantage compared to normal workflow languages. If there are no similarities between models, it will be the same like

giving someone the possibility to choose between two different models without the option of configuring them. Therefore it is better to keep the processes separated instead of making the effort to combine them.

A department at the municipality where most of the processes are alike across several municipalities is the civil affairs department. The processes at the civil affairs department are depending on the law and therefore have mandatory actions. In spite of the mandatory actions the municipalities have still the opportunity to adapt the civil affairs processes in their own way. Therefore the civil affairs processes will have variations amongst municipalities i.e. configuration is possible on the models at the variation points.

Frequency

To make sure the municipalities were familiar enough with the processes we wanted to model, we need to identify frequent occurring processes.

Based on the data of the software developer that we selected as stakeholder, which supplies over two hundred municipalities with software, we could determine the most occurring processes. The processes that appeared most are *marriage*, *divorce*, *birth*, and *decease*. Because the *divorce* process is only a data change in the database of the municipalities, this process is replaced by the fourth most occurring process and that is the *acknowledgement of the unborn child*. For these four processes we research if it is possible to create configurable models.

Number of processes

The number of processes is determined in consult with software developers. We need to find a balance between the available time and the number of processes needed to cover as many configurable situations as possible. Based on the timeframe we decided to choose for four processes. With this number of processes we still have enough bases to state our findings have the possibility to compare the processes mutually, and finish the research within the timeframe. In Table 3.1 a summary is given of the processes and the result on the criteria.

Marriage	<ul style="list-style-type: none"> - Occurring frequently - Similarities amongst municipalities - Process 1 of 4
Birth	<ul style="list-style-type: none"> - Occurring frequently - Similarities amongst municipalities - Process 2 of 4
Decease	<ul style="list-style-type: none"> - Occurring frequently - Similarities amongst municipalities - Process 3 of 4
Acknowledgement of the unborn child	<ul style="list-style-type: none"> - Occurring frequently - Similarities amongst municipalities - Process 4 of 4

Table 3.1 The selected processes

3.1.2 Process descriptions

The processes that were used during the case study will generally be described in the next section to give a better understanding of the process models that later on will be discussed. Before we describe the selected processes we explain the Gemeentelijke Basis Administratie (GBA, municipal basis administration). Changing the GBA is part of the selected processes but is not a direct task of the civil affairs department. Nevertheless the GBA is still part of the processes and will therefore be explained. In the GBA the information about all inhabitants of a municipality are stored like, name, parents, birth location, etc. In case of the processes, *birth*, *marriage*, and *decease* the situation of a person changes. Accordingly the data in the GBA should change. So if a new born child is registered at the municipality a new record must be created. The records of the parents should also be updated with the information of their child. In their own way the selected processes affect directly the GBA except the *acknowledgement of an unborn child*. *Acknowledgement of an unborn child* does not directly affect the GBA because the data of this process will only be processed at the moment the *birth* process is executed. In the next section we will discuss more extensively the selected processes.

3.1.2.1 Acknowledgement unborn child

The procedure concerns the acknowledgement of an unborn child. When a couple is not married but is expecting a child in the near future, the father needs to register as father of the child. This allows the father to have authority by law over the child. The procedure is only needed if the expected child is the first child out of a relationship. There are some restrictions to execute the procedure. First of all the procedure can only be executed for males and not for females. Secondly both parents may not be married. If they are married to each other the father is automatically acknowledged as the father of the child. But if they are not married to each other but to someone else the father may not acknowledge the child. Also the father needs to have permission from the mother to acknowledge the child to prevent that men are acknowledging without the knowledge of the mother.

Acknowledgement of an unborn child can be registered in the Netherlands by every civil servant or notary. During this procedure an acknowledgement document is created. Also an acknowledgement, that took place abroad, can be accepted to a certain extent in the Netherlands. When the child is born abroad the child can be registered at s'-Gravenhage with a foreign birth certificate and acknowledgement. When the whole *acknowledgement of an unborn child* process is completed, it will be processed when the child is born and its birth certificate is drawn up.

3.1.2.2 Birth

In the Netherlands the *birth* procedure must be executed within three days after the birth and needs to be processed by a civil servant of the civil affairs department of the municipality the child is born in. In case a child is born in hospital the child must be registered in the municipality where the hospital is located. It is not possible to register the child in your own municipality when the child is born in another municipality. The birth location determines the registration municipality. Not everyone can go to the civil affairs to register a child. The persons that are allowed to do the registration are the father or the mother, every one that was present at the birth, the owner of the building in which the child is born, the head of the institution where the child is born, or when all people that are allowed to register are not able to do the registration the mayor of the birth municipality may register the child.

During the registration the civil servant discusses the birthday and time, gender, birth location, first name, relevant parent data, and if needed the

acknowledgement and the name choice. When the child is acknowledged (*acknowledgement of an unborn child* process) the corresponding data will be processed during the *birth* process.

Name Choice

The name choice can be done during the *acknowledgement of an unborn child* process but also during the *birth* process. During the *name choice* process the last name of the child will be determined. Only when both parents are present the choice may be made. This decision has consequences for all children that will be born out of the relationship because they will get the same last name. Therefore the *name choice* process can only be done for the first child out of the relation. When there is no decision made about the last name of the child by the parents, it will be determined in another way. This determination is made based on the relational situation of the mother of the child. When the mother is single the child will get the last name of the mother but if she is married the child will get the last name of her husband. This description applies for Dutch children. If the child has a different nationality the name choice procedure of the origin country of the child must be executed.

3.1.2.3 Marriage

When a couple has decided to get married they need to notify the municipality. The notification must be done in the municipalities where one or both are registered. This can be done in writing or in person. In the Netherlands it is most common to do this in person. During the notice the needed documentation is checked among which, the birth certificate, a valid identification document, a withdraw document of a previous marriage, and the data of the best man. The notification is valid for one year starting from the date the notification is done at the municipality. So the couple must get married within one year of the notification otherwise they should notify again. Unlike the notice, the actual marriage may be done at a municipality that is not one of the municipalities where they are registered. When a couple wants to marry in a different municipality than the one of the notice, than the municipality that registered the notice should send all relevant information to the municipality where the couple will marry. The couple should already indicate at the notice which municipality will be the

marriage location. If this is not done the couple must marry in the municipality of the notice. The actual marriage ceremony at the municipality consists of a speech of the civil servant and signing the marriage certificate by the bride, groom, witnesses, and the civil servant. When the marriage ceremony is completed the couple will be registered as husband and wife in the GBA.

3.1.2.4 Decease

When someone passes away in the Netherlands a death certificate must be created. The registration of the deceased must be done in the municipality the person died. Everyone who knows that a person died may notify the municipality. In the Netherlands most of the time the undertaker is the one who notifies the municipality. The two main activities during the deceased process is too register the fact that someone died and to give permission to burry or cremate the person. The person who notified the municipality has to hand over two documents, the declaration of death and the declaration of natural cause. If the deceased did not die by natural cause a declaration should be handed over that states it is alright to bury or cremate the deceased. It is also possible that a person is put at the disposal of science i.e. this means the body may not be buried or cremated. Another exception, with regard to permission to bury or burn a body, is when the time that a person is buried or burned differs from the time that is determined by law. The time that is determined by law is within five days after death. It can be the case that a relative is on vacation and cannot to be back within this term and therefore asks for an extension of the period. Another possibility is that the deceased must be buried within 24 hours because of religious grounds. This must also be requested to make sure the person did not die as a result of a criminal act.

3.1.3 Selecting the municipalities

The municipalities were selected in consultation with the software developer that we selected as stakeholder. We selected the municipalities based on the following criteria:

- Size/ number of inhabitants
- Frequency of executing a process
- Type of software used

Size/ number of inhabitants

We looked at municipalities that vary in number of inhabitants to increase the possibility of variation in the execution of the processes. By looking at the number of inhabitants we made a selection. We want to have a small, medium and large municipality to look at their processes. The criteria we used to determine what a small, medium or large municipality is, are made in consult with the selected software developer. The criteria are depicted in Table 3.2.

Small size municipality	30.000 inhabitants or less
Medium size municipality	Between 30.000 and 80.000 inhabitants
Large size municipality	80.000 inhabitants or more

Table 3.2 The selected processes

Frequency

A large municipality has much more deceased than a small municipality and therefore the process at the large municipality is probably more efficient. A main factor for the number of deceased in a municipality, beside the number of inhabitants, is that it is common for large municipalities to have a hospital. When a hospital is located in a municipality this will logically imply that more children are born and more people die in the municipality. Based on the size we selected three municipalities that vary in size from 26.000 till 117.000 inhabitants (municipality A, B and C). The largest of these municipalities has a hospital and has therefore more people who die in the municipality and more children born. This causes that they executed the processes *birth* and *deceased* more often than municipalities without hospital and this could result in more efficient processes than municipalities without a hospital.

Software

Differences in software can also cause variation in the process execution. The three municipalities that are selected all have the same software supplier that supports their processes. In this way it can be that the execution of the processes are very alike for these municipalities because they are directed by the software. To get as many variations for the selected processes, a fourth municipality (municipality D) was selected that uses different software to support their processes. The fourth municipality has approximately 200.000 inhabitants and is the largest of the four and

completes the group of selected municipalities. The municipalities are labeled with letters. The Smallest municipality is labeled as municipality A and the largest municipality as D as depicted in Table 3.3.

Municipality A	<ul style="list-style-type: none"> - ± 26.000 inhabitants - Small size municipality - No hospital - Software developer A
Municipality B	<ul style="list-style-type: none"> - ± 42.000 inhabitants - Medium size municipality - No hospital - Software developer A
Municipality C	<ul style="list-style-type: none"> - ± 117.000 inhabitants - Large size municipality - Has a hospital - Software developer A
Municipality D	<ul style="list-style-type: none"> - ± 200.000 inhabitants - Large size municipality - Has a hospital - Software developer B

Table 3.3 The selected municipalities

3.1.4 Approach of collecting

The data of the processes *acknowledgement of the unborn child*, *birth*, *marriage*, and *decease* are collected by means of interviewing the process owners at the municipality. By asking the process owners how the processes are executed at their municipality, we were able to model the processes. We did not observe the processes during their execution because it was too time consuming. The different frequencies the processes occurred at municipalities did not allow observing the processes because it was not familiar when an inhabitant would enter one of the processes. In the smallest municipality the frequency of the processes is that low that it is not feasible in time to wait until all selected processes occurred. By interviewing the process owners they explained the processes to us. In some cases the process owners provided us with process models that were created for documentation. We

used these models and if needed modified it to the description they gave us. When there was no model present at the municipality, we created the model from scratch. Because all municipalities that had already created some process models used Protos as workflow language, we decided to use this modeling application. By using Protos the understandability of the process models, for the municipalities, is larger than when using other workflow languages because they are already familiar with Protos. To make sure the process models were correct we asked the process owners to check them. It is still possible that some errors occurred in the process models that are not noticed by the process owners. Because it is not an objective to find the best practice of the processes this will not be an influence on the case study. The possibility that the process is executed in practice in a different way is small because the processes are guided by law and therefore has mandatory steps.

Reference process models for municipalities are available from the *Nederlandse Vereniging voor Burgerzaken* (NVVB, Dutch Society for Civil Affairs). These reference models depict the “best practice” that they have distilled from experiences of municipalities. For every civil affairs process they have a single reference model thus also for the four processes we selected. We did not look into these reference models until all interviews were completed. In this way we are not influenced by these models during the creation of the models. In spite of this precaution the resulting models were influenced by the *NVVB* reference models. Almost every municipality in the Netherlands owns the *NVVB* reference models and adapts parts of these models. Regardless of the influence of the *NVVB* reference models there was still variation between the ways of execution of the processes amongst the municipalities.

3.1.5 Data collection: Result

After visiting the four municipalities we have a total of twenty individual business process models. The collection consists of four processes that we collected at the four municipalities. Accordingly we have a collection of sixteen business process models. The remaining four business process models are the *NVVB* reference models that complete the collection. This collection of business process models is used as the basis of creating the configurable workflow models.

During the remainder of the thesis the business process *Birth* is used to explain the approach we used to get from individual business process models to a configurable workflow model. Because the approach for all four processes is similar,

the other three business processes are moved to the appendix along with all corresponding models (appendix 8 till 22).

The individual business process model *Birth* of one of the four municipalities is depicted in Figure 3-3. The complete set of *Birth* business process models of the municipalities are depicted in appendix 3-7. A description of what happens during this specific process will be given. The process starts when a citizen¹ triggers the first activity. When the citizen arrives at the civil affairs department the civil servant must check if the citizen is allowed to register the child². When the civil servant notices that the citizen is not allowed to register the child, the received documents will be returned to the citizen³. If everything appeared to be correct the identity of the citizen will be checked⁴. When the citizen cannot be identified the registration of the child will be canceled⁵. Otherwise the GBA data needs to be checked⁶. During the check the information that is given by the citizen is checked on consistency by consulting the GBA. If the citizen is not registered at the birth municipality of the child the municipality has to contact the municipality where the citizen is registered. This allows the municipality to verify the information that is given by the citizen. There will be a search to check if the GBA data is correct when the information of the citizen appears to be different from the GBA data⁷. The following activity is to determine the descent of the child⁸. In this particular business process model the action *Determine descent* is a sub process. This sub process contains the actions that are related to determine the descent of the child. The descent is determined based on the information if the mother is married or not and the nationality of the parents. When the descent of the child is determined, the parents need to choose what the last name of the child will be⁹ (See for a more detailed description; sub part *Name choice* of paragraph 3.1.2.2). When all needed information is collected the birth certificate will be produced¹⁰. The birth certificate needs to be checked by the parent(s) and if the certificate is correct they must sign it to definite it¹¹. The civil servant will return the documents to the citizen and the process will be completed for the citizen¹². The following actions are all related to the GBA. First there is checked if the mother of the child is living in the birth municipality of the child¹³ (birth municipality of the child must be the same municipality as the municipality where the citizen registered the child). If the mother of the child is living in another municipality then the resident municipality will be notified by *TB01 message*¹⁷ (TB01 message is the name of a message that is send to another municipality). It can be the case that both parents are not registered in the GBA and in that case the

response should be determined¹⁶. When the mother is registered in the birth municipality the GBA should be updated with all the data related to the child¹⁴. It can be that the mother of the child does not have authority of the child and in that case the court should be notified¹⁵. For example the mother does not have authority of the child if she for instance is a minor. When all these actions are completed there is a check to see if the GBA is updated correctly¹⁸. If this is not the case the errors will be corrected¹⁹. This will be followed by sending a copy of the personal data of the child to its parents²⁰. All organizations that need to know that the child is born are also notified²¹. For instance the maternal health institution and the tax office should be notified. Finally the birth certificate is archived²² and if the child is living in the birth municipality the birth certificate is digitalized and stored²³ to complete the birth process.

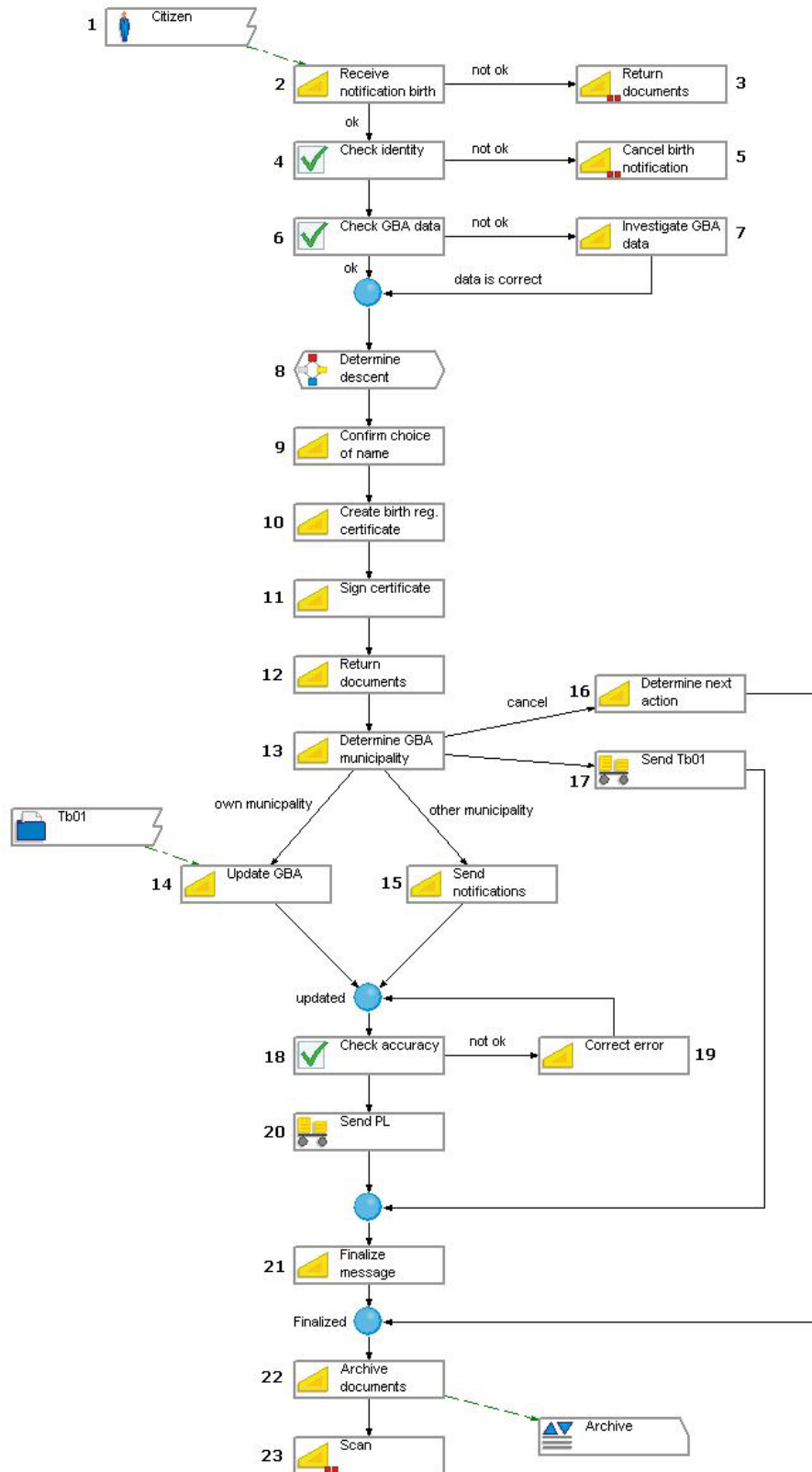


Fig. 3-3 The birth business process model of one of the municipalities

3.2 Identify variations

When all the business process models of the four municipalities are collected we need to combine them to create a business process model that contains all variations. To know if the model contains all the variation they first need to be identified. How to identify the variations is explained in this section and is related to *Phase 2* in Figure 3.4.

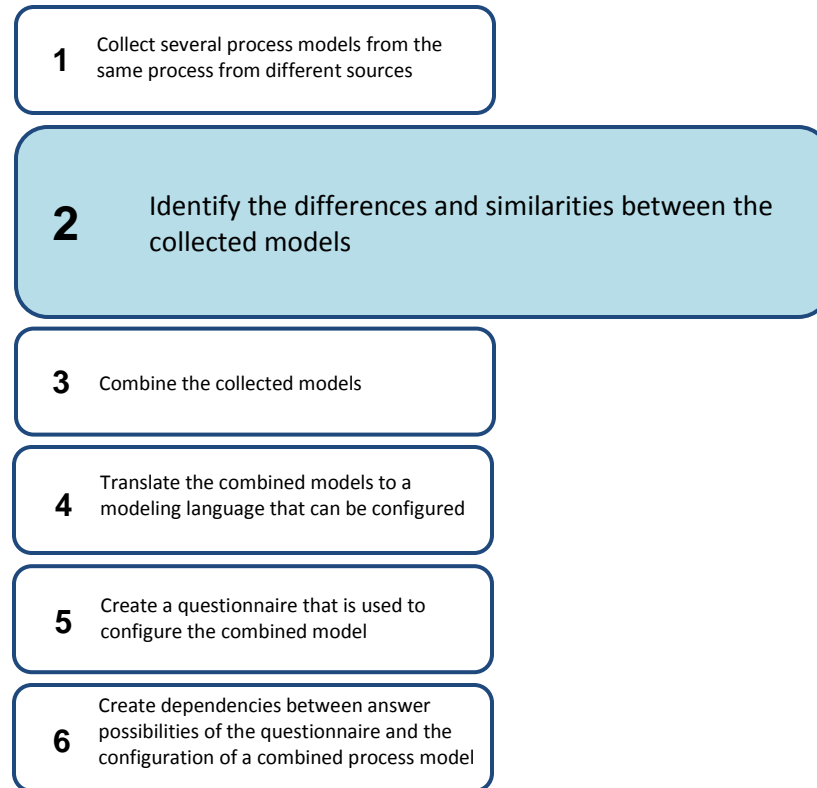


Fig. 3-4 Phase 2: identifying differences and similarities

We defined variation, in the introduction of this thesis, as the difference between two process models that describe the same process in different organizations. The variations allow making a decision between options. Unlike decisions on an instance level, *i.e.* at runtime, decisions on a type level, *i.e.* at build time have an impact on the model and its actual structure [40]. Such configuration decisions have to be clearly differentiated from runtime decisions and can be highlighted as *variation points* in a model [44]. A variation point captures a decision point together with the related possible choices. Choices are always done between several different options and therefore we have to identify what the options of choice are.

3.2.1 Similarities

Before we can determine the differences between business processes, we must identify some similarities [30]. To determine equivalence between activities of different models, [30] checks if the unit of work that they represent is equivalent for the purpose of the integration based on two criteria:

- The effect that the units of work will have in the integrated process must be the same.
- The way in which the effect is achieved must be the same.

In Figure 3-5 we applied the two types of equivalences on two business process models (process B in Fig. 3-5. is the business process model depicted in Fig. 3-3.) of the process *Birth* of different municipalities. We first look at the similarities based on the effect that the units of work will have in the integrated process. There are several actions that are similar on name and effect namely; similarities 1, 2, 3, and 4. The names of the actions of similarity 5 and 6 are different but the content of the actions is the same and are therefore similar. Secondly we look at the similarities of collections of actions. In Figure 3-5 there are three collections of similar actions namely similarities 2, 3, and 4. These collections consist of successive similar actions. For example similarity 3 consists of *Check GBA data* and *Investigate GBA data* that are part of a cluster of the same actions.

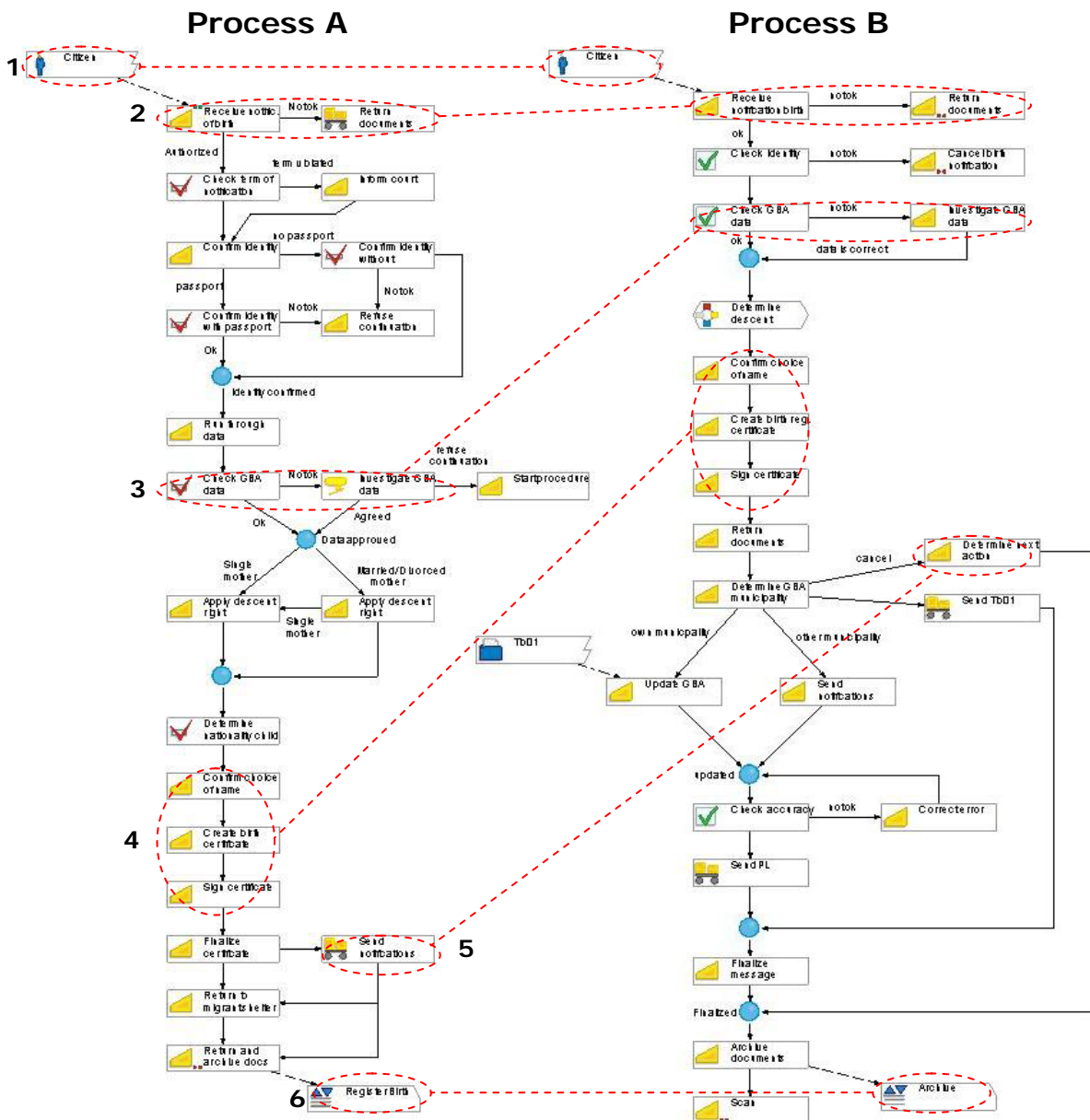


Fig. 3-5 Similarities between two birth business process models

Now we have defined the similarities of the two models we know which actions can be merged when the two models are combined into one model.

3.2.2 Differences

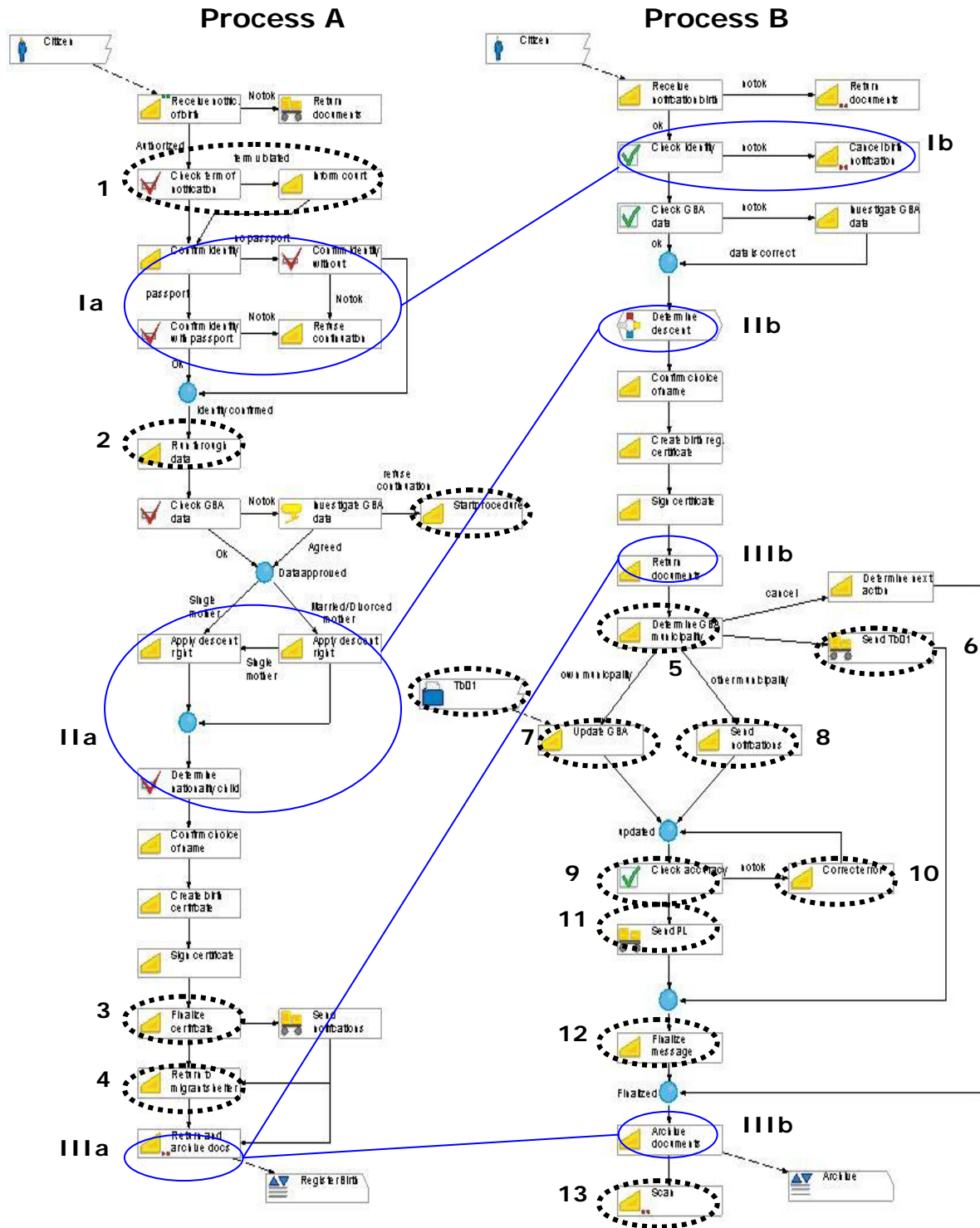
The detection of differences is done by means of looking at the actions that are not similar. We make a distinction between differences of actions in the same way [30] has done.

- 1 The first distinction is when an action that exists in one business process model but no action representing an equivalent unit of work exists in the other process. [30] called this type of difference a *Skipped activity*.
- 2 Second type of difference is the *Interchanged actions*. The result of actions is the same but the way of achieving that result is different.
- 3 If an action in one business process model is represented by several actions in another business process model it is called a *Refined activity*.
- 4 Two actions have *corresponding collections of activities*, if a collection of activities in one process is equivalent to a collection of activities in the other process, while no subset of activities from the other collection.
- 5 Two (collections of) activities are partly equivalent if they partly represent the same unit of work and partly represent different units of work and there is no possibility to re-arrange the activities into equivalent collections.

If one of the above mentioned distinctions of differences hold for activities in two separate business process models we will use them as variation points for the configurable workflow model.

We will look at the differences of the two birth business process models and categorize the differences. In Figure 3-6 the differences are depicted. As shown there are only two kinds of differences appearing between the two birth business process models. The first type of difference we found is the *skipped activity* (dashed black circle). The *Check term of notification*¹ and *inform court*¹ action are not present in process B and is not represented by a similar action with a different name and is therefore identified as a *skipped activity*. The same holds for all other skipped actions²⁻¹³. A second type of difference that was identified is the *refined activity* (blue circle). The first collections of *refined activities* are the actions related to the identification of the citizen. Those actions are *confirm identity*^{1a}, *confirm identity without passport*^{1a}, *confirm identity with passport*^{1a}, and *refuse continuation*^{1a} of process A and those are represented by *check identity*^{1b} and *cancel birth notification*^{1b} in process B. In the spite of the different number of actions between the collections, that represent the identification of a citizen, the content of the collection is the same. It is expected that if the content is similar the collection of actions should be identified during the check for similarities, but if the number of actions that is needed to execute a task is different between the business process models the manner of execution differs. We want to keep these different ways of executing a task preserved for the configurable process model. The different ways of executing will

become *variation points* i.e. so the customer can choose which way he prefers. Another *refined activity* is the collection of actions represented by II. This refined activity is a special case. The three actions^{IIa} in Process A are similar to the sub process, *determine descent*^{IIb} in process B. Because the actions of the sub process are on a different presentation layer they are not represented in the model. Still the collections of actions can be compared like the collection of refined activities to check the identity of a citizen only the sub process must not be identified as one action but as several actions (the number of actions that the sub process contains). This allows identifying the differences between the collection of actions^{IIa} and the action in the sub process of process B will become variation points. The last refined activity in these two models is represented by III. In process A the *Return and archive docs*^{IIIa} are represented by one action while the same task is represented by two actions return documents and archive documents in process B. The difference with the other two refined activities is that the actions in process B are not directly linked to each other. In this way actions^{IIIb} in process B are executed at two different moments, in the middle and at the end, of the process while action^{IIIa} in process A is only executed at the end. The different actions we defined will not be merged into one action but will be variation points in the configurable model.



- Skipped activity
- Corresponding collections of activities
- Interchanged actions
- Partly corresponding (collections of) activities
- Refined activity

Fig. 3-6 Differences between two birth business process models

3.3 How to combine

The identified variations need to be combined in one model to create a configurable process model. The model that combines all the variations is named the *combined model*. We use the term *combined model* during section 3.3 and its subsections, to indicate that business process models were merged into one model. The combined model is actually a configurable process model without the configuration constraints. The creation of a combined model is represented in Figure 3.7 by *Phase 3*.

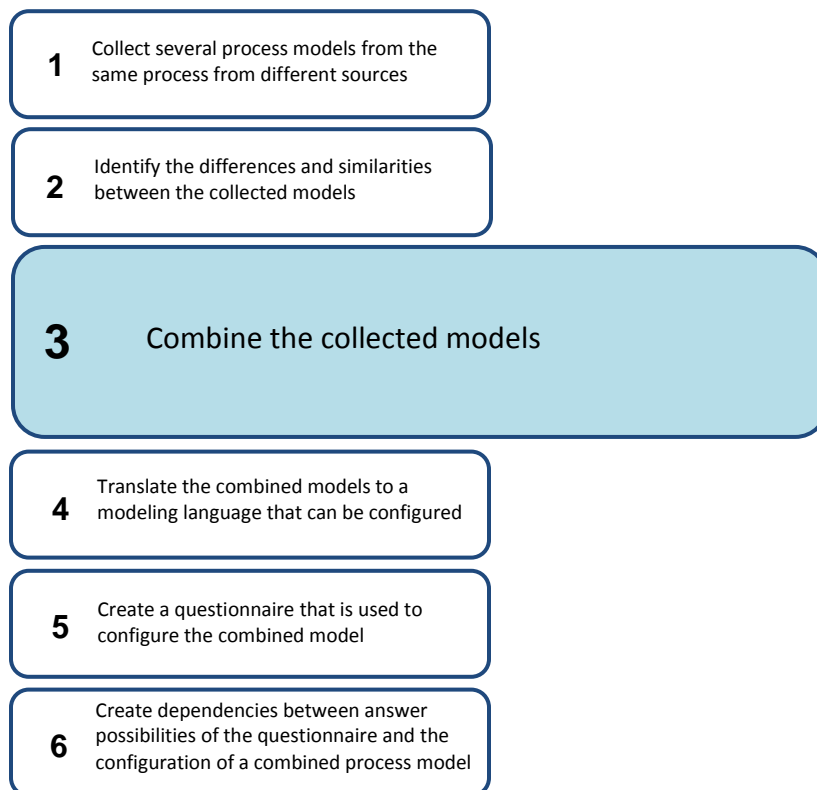


Fig. 3-7 Phase 3: Combining the models

The combining of two models is done by taking one of the two models as starting point. We will take the left business process model (process A) of Figure 3-6 as a starting point. Then we start at the beginning of the right business process model until we reach the first difference. In this case the first difference is the *refined activity Ib*. By looking which actions are the predecessor and the successor of *refined activity Ib* the location can be determined. The predecessor of the *refined activity Ib* is the action *Receive notification birth*. This is not the case if the *refined activity Ib* is located in process A. Therefore a new arc should be created from action *Receive*

notification birth to refined activity Ib. The same should be done for the successor, *check GBA*, of *refined activity Ib*. Accordingly the same path can be taken in the new combined model as in process B (*Figure 3-11 depicts the new situation*).

The same method will be applied on the next difference we reach in process B i.e. the *refined activity IIb*. The *refined activity IIb* is a sub process and therefore the containing actions are not visible. Despite the actions are not visible at this level,

they are visible on a more refined level. Hence the same method can be applied as for *refined activity Ib*. We will identify the predecessor and the successor of the *refined activity IIb* and locate *refined activity IIb* between those two (In appendix 23 the result of the combining of the sub process is depicted). We will repeat this method until all differences from process B are located in process A.

3.3.1 Combining behavior

Besides the similarities and differences of the content of the actions, we also have to take into account the behavior of the actions. The behavior possibilities are already discussed in Section 2.1.1 for Protos, 2.2.1 for YAWL, and 2.3.2 for Configurable workflow models. The possible behavior that an action can have are AND-, OR-, and XOR-joins and –splits. To decide what behavior an action must have when they are merged, some constraints are needed. [40] formulates the constraints for merging behavior of actions and are depicted in table 3.4. When we look at the behavior of the actions, we focus on the similar actions (from Section 3.2.1) that need to be merged.

Combining behavior	Result behavior
XOR + XOR	XOR
XOR + OR	OR
XOR + AND	OR
AND + OR	OR
OR + OR	OR
AND + AND	OR

Table 3.4 Combining behavior constraints

The behaviors of the actions in Protos are not visible in the depiction of the process models. They are visible when the properties of an action are consulted in the Protos tool. For example, the *Receive notification birth* in Process A has a XOR-split and the *Receive notification birth* of process B is also a XOR-split (Figure 3-5). If we use the constraints of Table 3.4 we see that combining two XORs results in a XOR. The only difference, in comparison with the old *Receive notification birth* action, is that the new *Receive notification birth* action will have three outgoing arcs instead of two (Figure 3-11). This is caused by the different successors of *Receive notification* in process A and process B. The same principal holds for the similarities in Figure 3-5 except similarity 4 because these actions don't have XOR, OR, and AND behavior they will be exactly the same as before combining the two business process models. In Figure 3-11 the result of combining the two models in Protos is depicted. If we look at Figure 2-4 it shows that the possibilities of the XOR-split and the AND-split are a subset of the OR-split. The XOR-split is not a subset of the AND-split and vice versa. If two behavior possibilities are combined the behavior of both should be preserved in the new behavior. Because the OR-behavior has the behavior of both AND-behavior and XOR-behavior, the combining of two different behaviors will always result in OR-behavior. For example if we combine an AND-split and XOR-split, the result will be an OR-split. We can then configure the OR-split in to one of the two previous behaviors. It is important that the configuration of the split is done otherwise we can end up with unwanted behavior.

3.3.2 Modeling different order

Besides combining of actions and their behavior, we also looked at the combining of sequences that have a mutual different order of execution. In the *birth* business process models that we combined this situation did not occur. It can be that a sequence of actions occurs in one business process model and also in the other business process model, with as only difference between them the order. In this case the combined model must have both orders of execution they must both be present in the model. In Figure 3-8 we use *action A* and *action B* as example to simulate sequence of actions that have different orders in separate business process models. First model *I* in Figure 3-8 has the possibility to execute first *action A* followed by *action B*. At run-time the decision can be made to do *action A* followed by *action B* or the other way around. Also model *I* can be configured by blocking the input port

from *action A* or *action B*. By hiding the input ports of the left *action A* and the right *action B* it becomes possible to execute only *action A* or *action B*. The behavior of *model I* can also be modeled as *model II*. This is done by connecting *action A* and *action B* with several arcs and by giving them XOR-split and -join behavior. As a consequence it is possible to execute *action A* followed by *action B* and vice versa. A problem of *model II* is that it has a loop that can cause unwanted behavior i.e. *action A* can be executed followed by *action B* several times in a row. Therefore during configuration time a decision must be made what kind of behavior is needed. For example, if only *B* is needed the input ports of *action A* must be blocked. Another configuration can be blocking the output port from *action B* to the end state, the input port from *action A* that is triggered by the start state, and the output port from *action A* to *action B*, that will result in *action B* must be followed by *action A*. These are two of several possibilities to configure *model II*. The third model (*III*) has additional possibilities, on *model II*, which makes it possible to execute *action A* and *action B* in parallel. The parallel execution of *action A* and *action B* can be reached when OR-split is configured as AND-split. Like *model I* and *II*, *model III* also needs to be configured to eliminate unwanted behavior. These three patterns that are depicted by *model I*, *II*, and *III* in Figure 3-8 are possibilities to create several possible orders in a configurable process model. *Model I* is in behavior possibilities equal to *model II*, and the behavior possibilities of *model I* and *II* are a subset of *model III*. Depending on the behavior that is occurring in the models that will be combined, a pattern must be chosen that matches these behaviors. In this way different orders of execution can be configured.

The way of modeling an order depends on the constraints. When the possibilities of order are restricted to *A* followed by *B* or to *B* followed by *A*, than *model I* is preferred. *Model I* or *model II* could also be used to model *A* followed by *B*

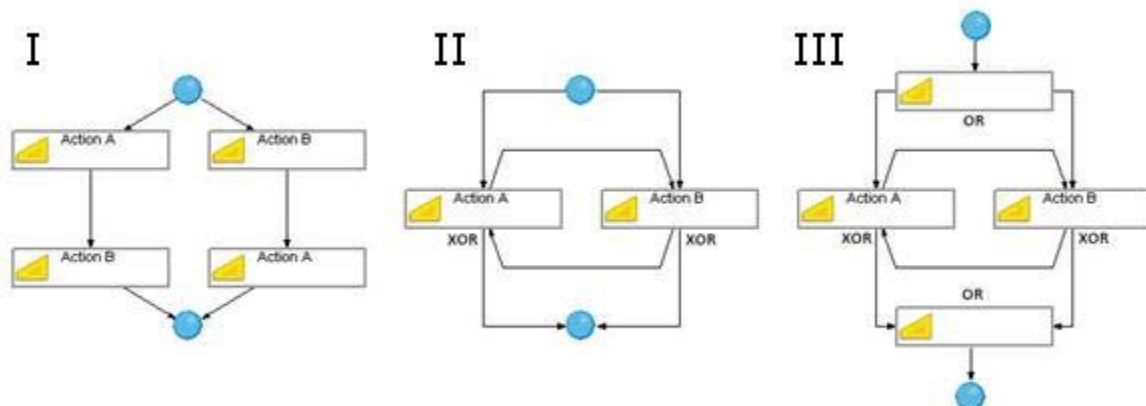


Fig. 3-8 Three different ways of modeling different orders of execution are depicted by models I, II, III

or *B* followed by *A* but these model have additional possibilities and could therefore generate extra behavior when a model is not configured right. The chance of generating extra behavior with model *I* is less than with model *II* or *III* because model *I* has less possibilities. In Table 3.5 the possibilities for each model or notated. Combinations of these possibilities can be executed in combination. For example model *II* can have a combination of ABABA, AAAA, BA, or ABAB depending on the configuration. The only model that can both execute sequential and in parallel, is model *III*. The criteria to determine which model is preferred in a specific situation are:

```

IF (Constraints model X ⊆ van Constraints model I)
THEN model I
ELSE IF Constraints model X ⊆ van Constraints model II)
    THEN model II
    ELSE IF Constraints model X ⊆ van Constraints model III)
        THEN model III
        ELSE NIL
    END
END
END

```

We used these guidance's to select which model we used to model different orders.

	Constraints	Examples of possible results
Model <i>I</i>	$(A \Rightarrow B)$ $(B \Rightarrow A)$	<i>Sequential</i> AB BA
Model <i>II</i>	$(A \Rightarrow B) = C$ $(B \Rightarrow A) = D$ $(A)^n$ $(B)^n$ $(C)^n$ $(D)^n$ $((C)^n \wedge A)$ $((D)^n \wedge B)$ $((C) \wedge ((B)^n))$ $((D) \wedge ((A)^n))$	<i>Sequential</i> A B BBB AA ABABABA BABABABABA

Model III	$(A \Rightarrow B) = C$ $(B \Rightarrow A) = D$ $(A)^n$ $(B)^n$ $(C)^n$ $(D)^n$ $((C)^n) \wedge A$ $((D)^n) \wedge B$ $((C) \wedge ((B)^n))$ $((D) \wedge ((A)^n))$ $(A \wedge B)$	<i>Sequential</i> A B AA BBB ABAB BABABA <i>Parallel</i> A → in parallel B ↗
-----------	--	---

Table 3.5 Possible outcomes for modeling orders with model I, II, and III

3.3.3 Creating extra behavior

Combining the *birth* business process models is done by looking at the behavior of every individual business process model. We required that every individual *birth* business process model could be derived from the combined model and still has its own behavior. This requirement should hold, but it can be the case that it is reasonable to add some new behavior to the combined model. For example if a new municipality wants to execute both *Confirm identity* and *Check term of notification* (from Figure 3-16) in parallel, it is not possible in the combined model because we combined the *Receive notification birth* task from every individual *birth* business process model and all of them had the XOR-split behavior i.e. combining XOR's will result in a XOR and therefore the possibility for parallel execution is eliminated because then an OR or an AND is needed. If we combine five XOR-splits we still have a XOR-split (Table 3.4). The solution to add the possibility to execute *Confirm*

$A \cup B = C$ $A \subseteq C$ $B \subseteq C$	No extra behavior
$A \cup B \neq C$ $A \subseteq C$ $B \subseteq C$ $D \subseteq C$	Extra behavior

Table 3.6 Criteria for extra behavior. A, B, C, are behaviors

identity and *Check term of notification* in parallel is to change the XOR-split into an OR-split.

Extra behavior is present when the result of combining models has possible behaviors that were not present in the individual models that were combined. In Table 3.6 we notated when extra behavior occurs and when not.

The example in Figure 3-9 depicts the combining of model A and model B. The combining of these two models results in model C. The combining of two XOR's results in a XOR but when it is preferred that both actions Y and Z could be executed in parallel extra behavior is needed. To keep the behavior of models A and B possible and have the additional possibility to executed action Y and action Z, the XOR-split of action X should be transformed to an OR-split, resulting in model D in figure 3-9.

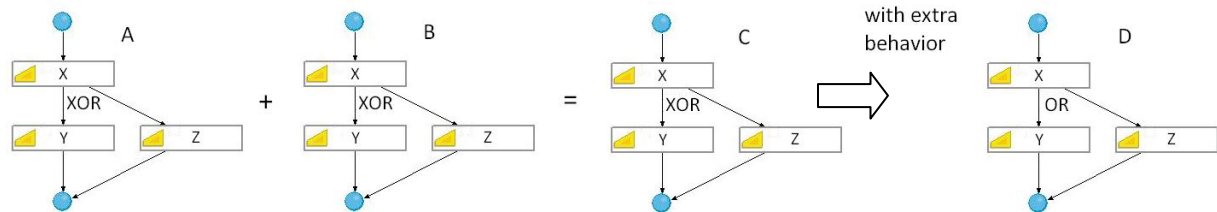


Fig. 3-9 Combining model A and model B resulting in model C. Extra behavior is added to model C

Another possibility to create extra behavior is depicted in Figure 3-10. The combination of two OR-splits results in an OR-split. In spite the split behavior stays the same there is extra behavior possible in the result model G in Figure 3-10. Model G has the possibility to execute actions W, Y, and Z in parallel while this was not possible before the merge of model E and model F. The extra behavior that is created in Figure 3-10 can be wanted but also unwanted behavior depending on how the model is applied.

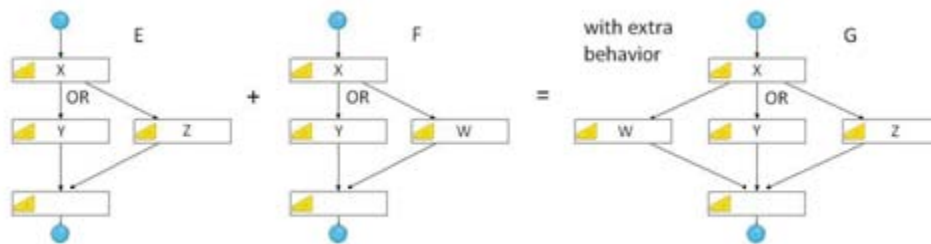


Fig. 3-10 Combining model E and model F resulting in model G.

Unwanted extra behavior can be eliminated by blocking or hiding ports. For example model *G* can be configured without behavior that was not present before the merge. In Figure 3-11 the outgoing port of action *X* is configured without behavior that was not present before the merge.

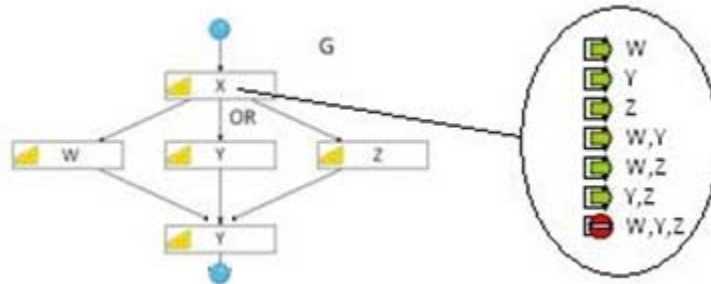


Fig. 3-11 Outgoing port of action *X* is configured to eliminated extra behavior

The dilemma of accepting extra behavior or not is depending on the result. When the result is wanted behavior it can be preserved and if it the extra behavior becomes unwanted behavior it must be corrected with configuration constraints.

Process A + B

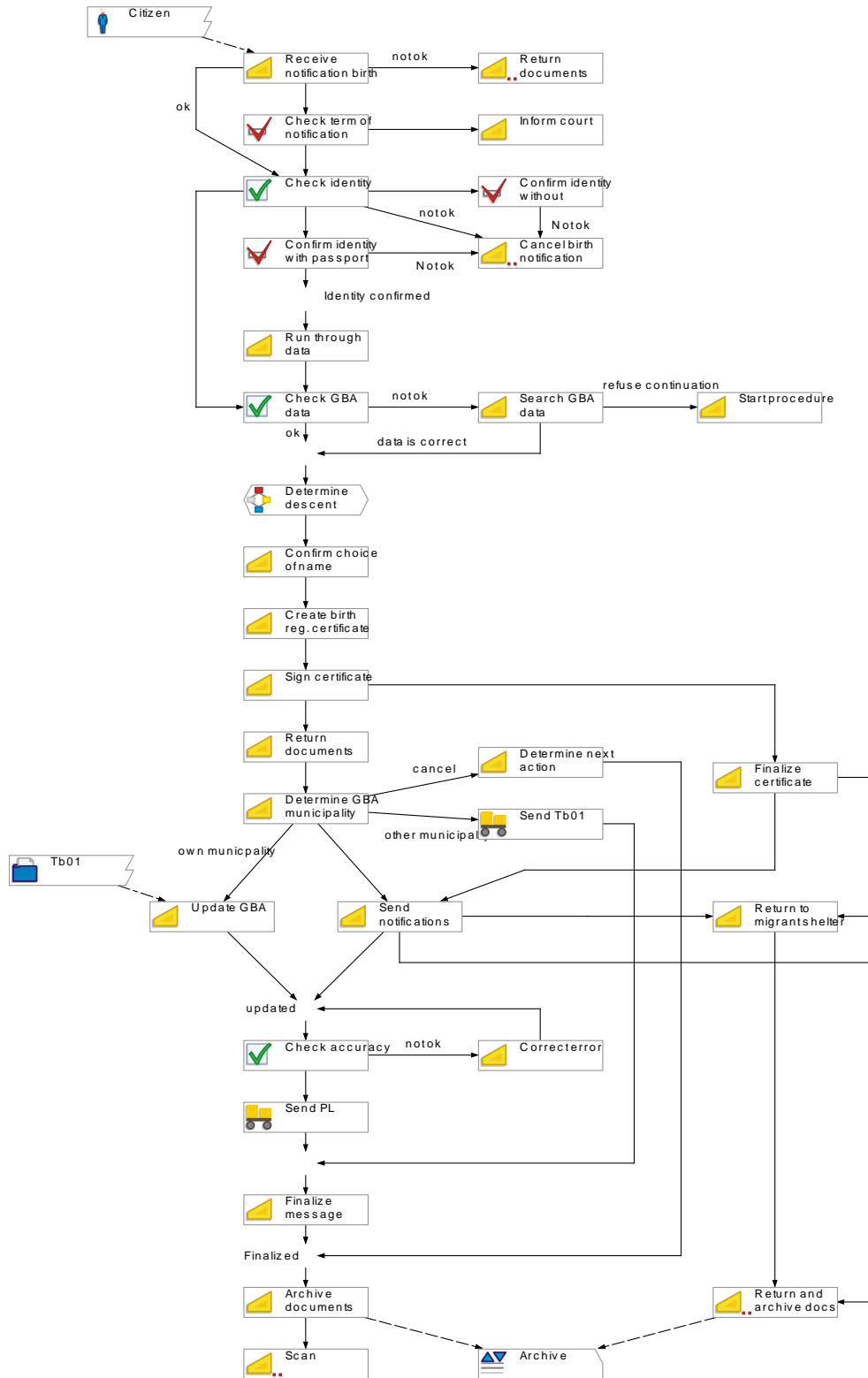


Fig. 3-12 Combined model of two birth business process models

3.3.4 Checking the model

To make sure that all the behavior and actions of process A and process B are represented in the combined model *process A + B* in Figure 3-12, a check needs to be performed. The check is straight forward and is done by taking one of the models that is combined, for example process B, and look if all flows that are possible in process B are also possible in the combined model *process A + B*. In appendix 24 the possible traces of process A are depicted in the combined model.

3.3.5 Combining models: Result

Until now we discussed how we can combine two models by identifying the similarities, differences, and the behavior of the two models, but we collected during the data collection (section 3.1) five birth business process models. To combine the five models we will perform, for every business process model we want to add to the combined model, the same steps as for combining the two models. Instead of combining the five models at once, we combined them in four iterations. In Figure 3-13 the process of combining models is graphically depicted. The A, in Figure 3-13, is the first business process model that will be combined with business process model B, resulting in a combined model. Subsequently the next business process model n will be added to the combined model until all business process models n are combined. The resulting model of combining the five *birth* business process models we collected during interviewing is depicted in Figure 3-14.

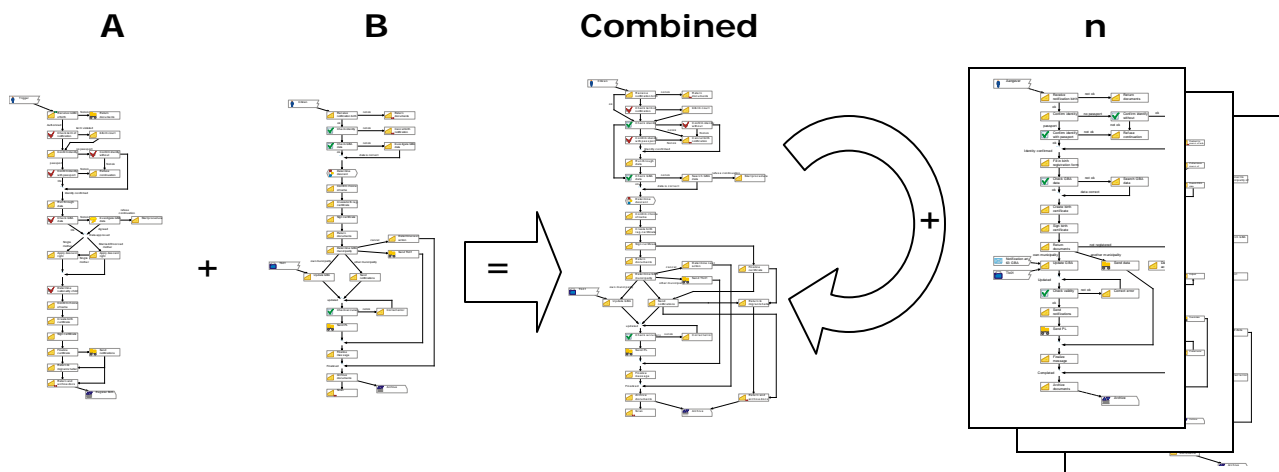


Fig. 3-13 As starting point two business process models will be merged, which results in the combined model. When more than two models must be combined every iteration of one model can be added to the combined model.

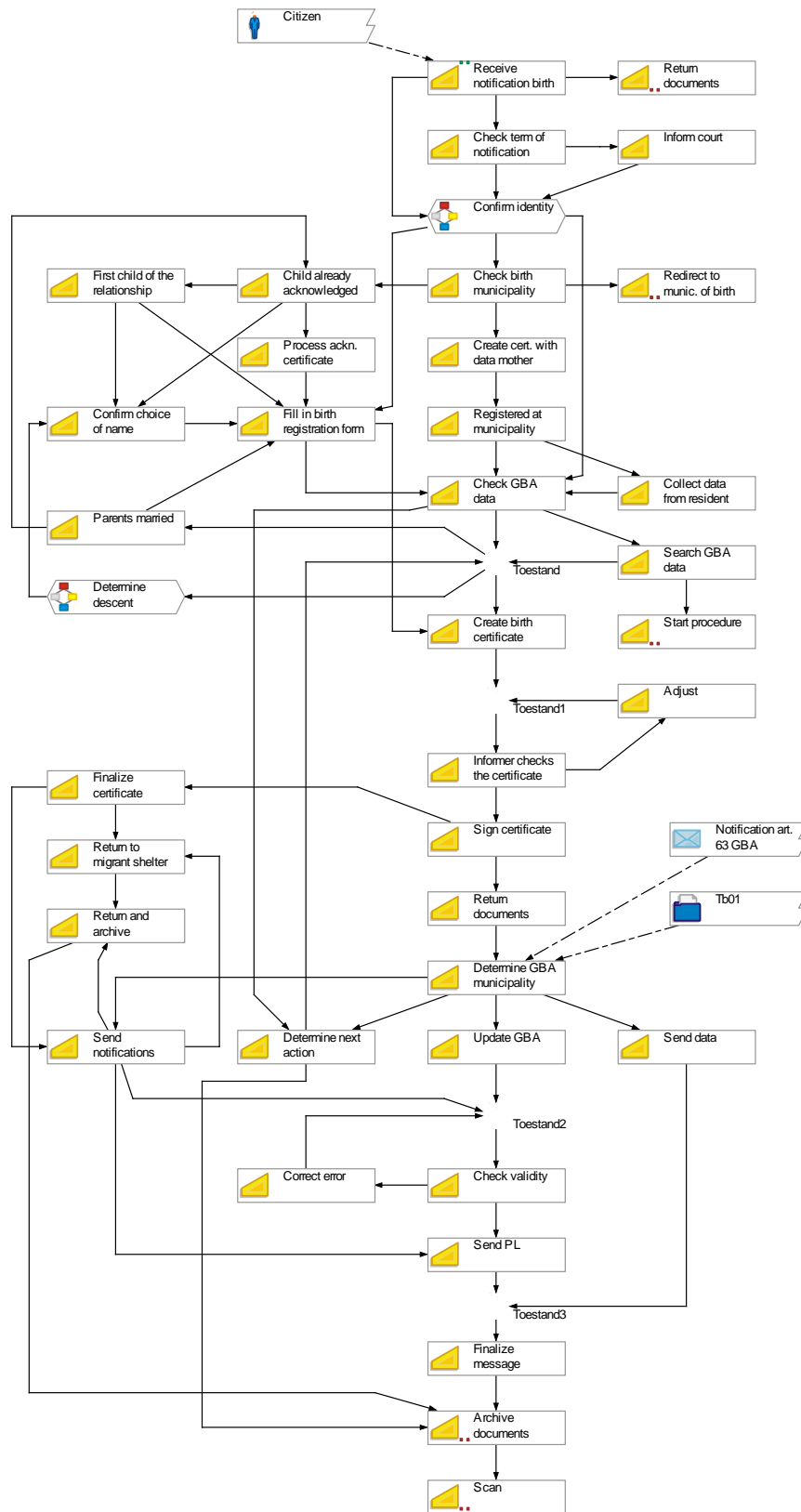


Fig. 3-14 The combined business process model of the Birth process containing five birth business process models

3.4 Transform the models to YAWL

The business process models in Protos, we created until now, are not executable. To make the business process models executable, so they become workflow models, we need to transform the Protos models into YAWL. The translation from Protos to YAWL is activity that belongs to *Phase 4* (Figure 3-15). During this section we will explain how we transformed the combined Protos models, which are described in the previous section, into YAWL. First we have a look which Protos symbols are equal to the YAWL symbols. Subsequently we translated the Protos model into YAWL followed by specifying the data that is needed to make the model executable in the YAWL engine. In section 3.4 and its subsections we will use the term *task* instead of *action*, and *condition* instead of *state* when we are talking about the combined YAWL workflow model. If we use the term *action* and *state* this implies that we are talking about the Protos business process models.

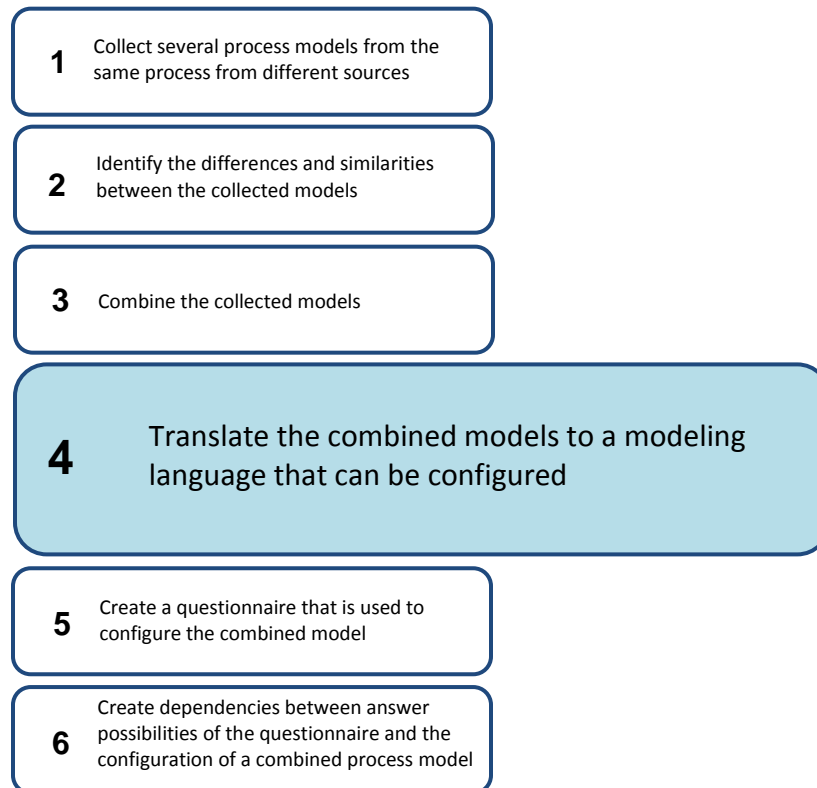


Fig. 3-15 Phase 4: Transform Protos to YAWL

3.4.1 Similarities amongst YAWL and Protos

To transform the combined business process model from Protos to YAWL we need to identify which type of symbols are equal. In the combined business process model that is created in Protos, mainly action symbols that are connected by arcs are used. Besides the action symbol, the symbol for states, sub processes, triggers, start, end, and buffers are occurring. To identify which symbol is representing a Protos symbol in YAWL we consulted [45] and [10]. [45] gives a description of the YAWL symbols













Protos symbol	Description	YAWL symbol
 State	A state a process is in at a specific time	 Condition
 Activity	A description of a unit of work that may need to be performed as part of a process.	 Atomic task
 Start activity	The first action of a process	 Input condition
 End activity	The last action of a process	 Output condition
 Sub process	Part of the main process that form an entity are grouped	 Composite task
 Trigger	An event that lead to execution of an action	Data event
 Buffer	Storing	Data event

Table 3.7. Equivalent Protos and YAWL symbols

and [10] of the Protos symbols. In Table 3.7 we depict the similar symbols and a description. The first five Protos symbols depicted in Table 3.7 have a YAWL equivalent, but the last two don't. The Protos symbols *Trigger* and *Buffer* are not depicted in YAWL by a symbol, but the same situation can be reached by data events in YAWL. For example for the buffer symbol, YAWL can store the output of one task and then later on the process this can be used as input for another task. Besides that Protos has symbols to depict *Trigger* and *Buffer* activities and YAWL does not, there is another difference that occurs between the start and end of a process. The business process models in Protos can have several start and end activities while YAWL models only can have one input condition and one output condition. To solve this problem we have a straight forward solution. We keep all the end points that are in the Protos business process models in YAWL and we create a connection by means

of an arc towards one output condition. In this situation it is still possible to end a process in several manners.

3.4.2 YAWL combined model

Now we know which Protos symbols are equal to YAWL symbols we can start transforming. By changing one by one every Protos symbol into an equal YAWL symbol we have a combined YAWL model as end result. In Figure 3-16 the YAWL model is depicted (in appendix 29 the translation is depicted and in appendices 28-31 the other combined process models are depicted).

3.4.3 YAWL data

Up to now we discussed which Protos symbols are equal to YAWL symbols and we applied this knowhow to translate the combined Protos model into a combined YAWL model. The result we want to achieve is a configurable workflow model. A workflow model is executable and at this moment the YAWL model is not executable yet. Therefore we need to specify the data for every task in the YAWL model (section 2.2.2 data perspective). [14] gives extensive explanation about how to specify the data for YAWL. We select one task out of the combined birth YAWL model to explain how the data for that task is specified. The selected task, *Receive notification birth*, is the first task in the model. To know which data is needed, we have to check what is done during this task. In the task *Receive notification birth* it is checked whether or not the person is allowed to register the child. There are four categories of people

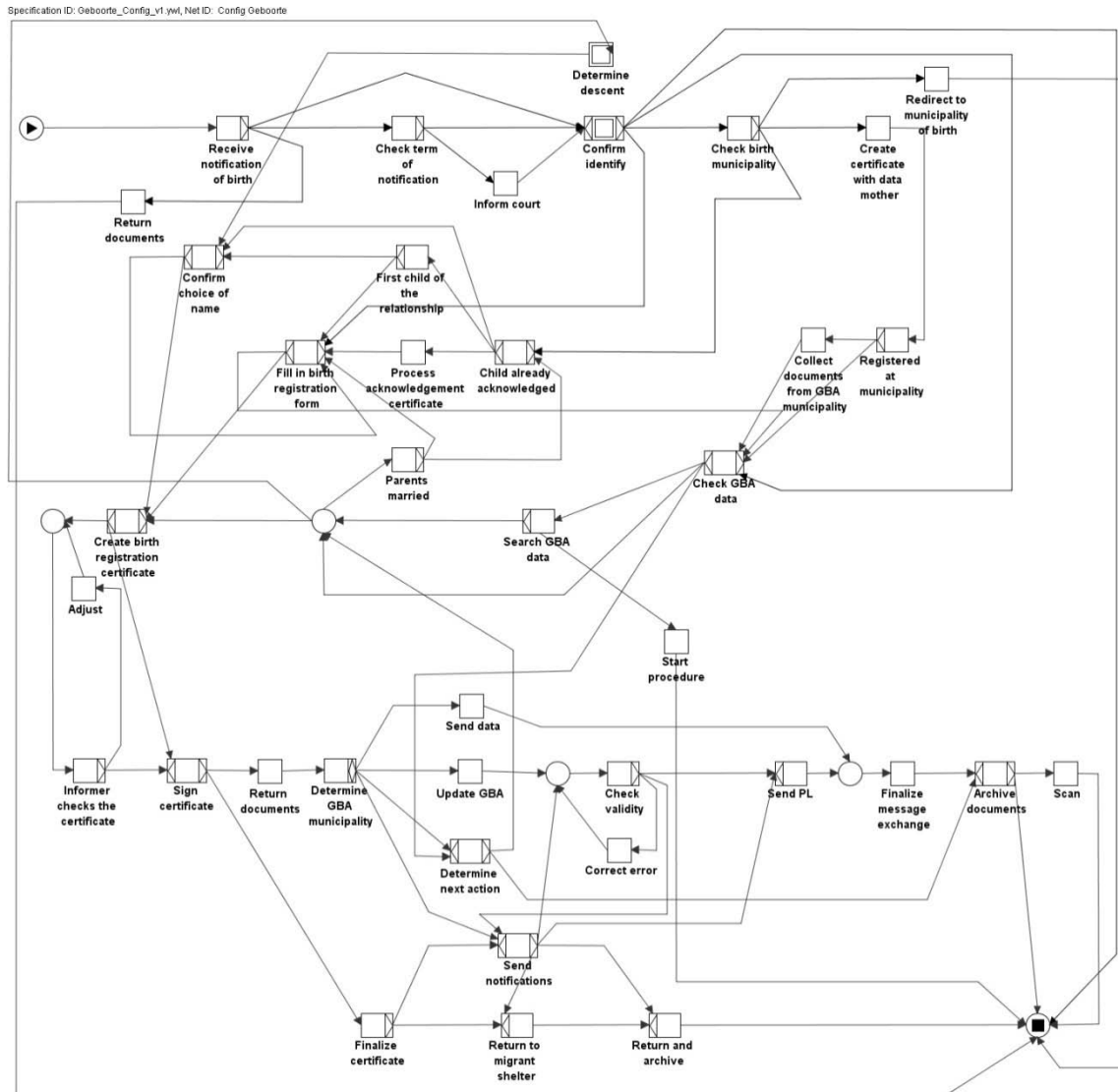


Fig. 3-16 Configurable birth workflow model

who are allowed to register the child; the parents, people who were present at the birth location, head/ owner of the institution the child was born, and an authorized civil servant. When the person cannot be categorized in one of those categories he/she is not allowed to register the child.

3.4.4 Control flow

To support the task *Receive notification birth* we create a check list for this task in YAWL by means of Boolean questions. Four Boolean questions in which every question represents a category and a fifth Boolean question in case the person is not allowed to register the child. The result of the Boolean questions determines what

the next task in the workflow should be. When for example the Boolean question *not authorized* is set to *true* the flow will continue towards *Return documents*, in all other cases it will continue to *Check term of notification* or *Confirm identity*. In Figure 3-17 the flow options are depicted. In the case of task *Receive notification birth* there are three flow possibilities; *Check term of notification*, *Confirm identity* and *Return documents*. For each flow possibility we specified in which case it should flow to this task. For example the predicate for the task *check term of notification* are depicted in Figure 3-18. Here it is stated that at least one of the four Boolean questions related to the categories should be answered *true* to continue to *Check term of notification*. The same predicate is used for *Confirm identity*. At this moment we don't know what the configuration is and therefore we don't know if the flow will continue to *Check term of notification* or *Confirm identity*. It is not allowed in process birth to execute *Check term of notification* or *Confirm identity* in parallel because *Receive notification* is an XOR split task. If the YAWL workflow model is not configured yet the workflow engine will select the first target task that has a predicate that holds. This means the order in which the *target tasks* in Figure 3-17 are depicted are of importance. In this case when the predicate of *Check term of notification* is *true* the flow will continue to this task and will not check the predicates of the other two target tasks. When the predicate of *Check term of notification* is *false* the next predicate will be checked until the YAWL engine finds a predicate to be *true*. If none of the predicates are *true* the last predicate will be selected as target task. In appendix 34 is depicted what *net variables*, *task variables*, and *task parameters* look like.

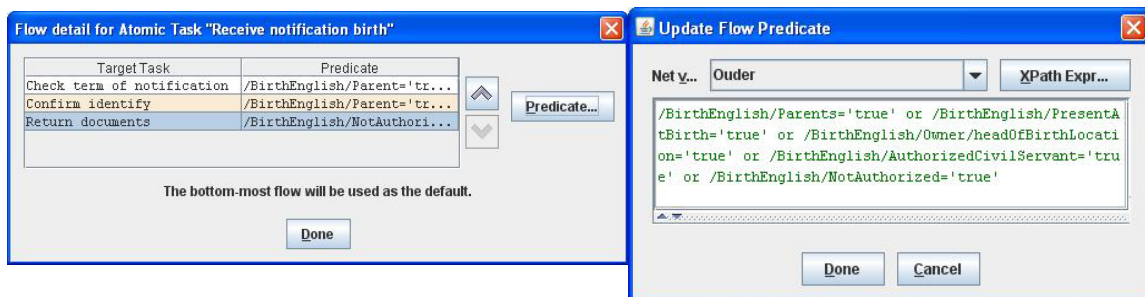


Fig. 3-17 The flow detail for atomic task "Receive notification birth"

Fig. 3-18 The flow predicate check term of notification

3.5 Create questionnaire

In the previous sections in chapter 3 we have discussed how we collected the data, created a combined business process model, and how we transformed the combined Protos business process model in to a combined YAWL workflow model. Several times we referred to the possibility of unwanted behavior. To eliminate the possibility of unwanted behavior and to prevent the confrontation of the user with a complex configurable model, we used the questionnaire approach from section 2.4. The description of the creation of a questionnaire in this paragraph is equal to *Phase 5* (Figure 3-19). The questionnaire is used to guide the user to the configuration that suites his situation. If we don't use the questionnaire, extensive modeling skills and

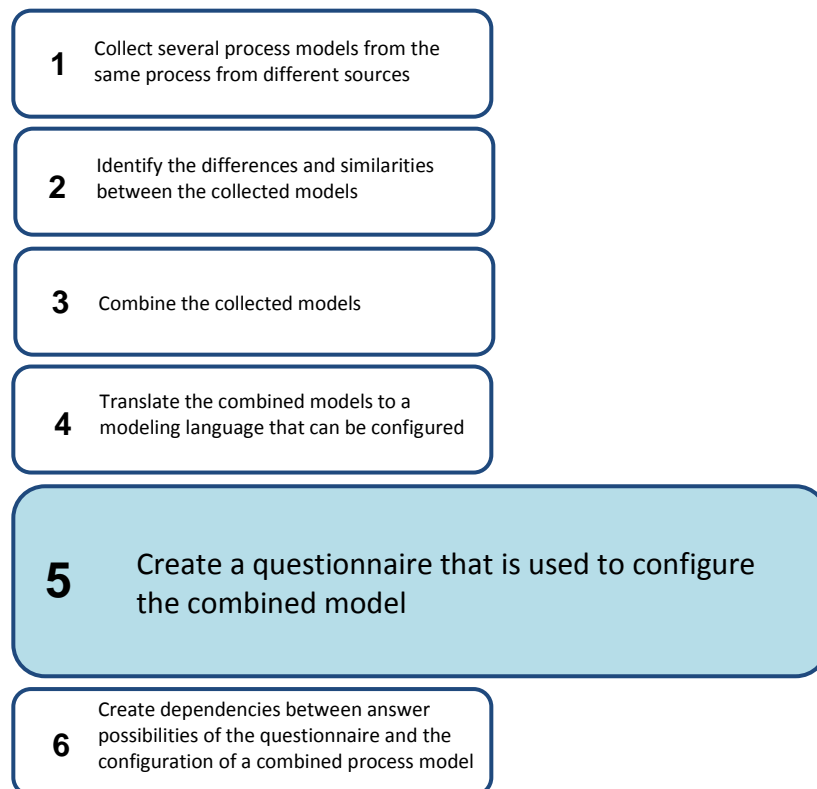


Fig. 3-19 Phase 5: Create a questionnaire

knowhow of the process would be needed to configure the configurable workflow model. In the next section we will describe how to identify the tasks that are configurable, because these tasks can have questions. Subsequently we describe how we specify the questions, followed by creating dependencies between questions and creating constraints.

3.5.1 Identifying the tasks that are configurable

Before we can start with creating questions for the questionnaire we have to identify which tasks can be configured. In principal every task can be configured but the result of the configuration must be a logical model. By a logical model we mean, the result of the execution of a business process or a workflow model is desirable. Therefore we will focus on the tasks that are logic to configure. In Section 3.2.2 we identified the differences between two *birth* business process models. Because in that section only two *birth* business process models are used we did not display all the differences that were identified in the combined *birth* business process model. These differences are depicted in appendix 35. According to the differences we will specify questions, because every difference implies a choice. As displayed in appendix 35 there are only six tasks that occurred in every individual *birth* business process model. This implies that the other 33 tasks do not occur in every individual *birth* business process model, and therefore are differences.

We assumed that the actions that are occurring in every individual *birth* business process model are mandatory. This implies that all similar actions/tasks are not configurable and therefore do not require a question.

3.5.1.1 Configurable clusters

The configurable *birth* workflow model that is created in section 3.4 has too many configurable tasks to create a clear questionnaire if for every individual configurable task a question is needed. Therefore we will look at configurable tasks that are related to other configurable tasks in every individual *Birth* business process model. When the tasks are related in the same way in every individual *Birth* business process model we assigned them to a cluster. The tasks in a cluster cannot be configured because we assume that if they occur in every individual *birth* business process model in the same formation, it is mandatory. By creating clusters instead of having a separate configurable task, we reduce the number of variation points. In Figure 3-20 the clusters that we created for the combined *birth* workflow model are depicted. The creation of these clusters is done by looking how a task is related to other tasks. For example, if the task *Check term of notification* occurs it is always followed by the XOR choice between *Confirm identity* and *Inform court*. These three tasks do not form a cluster yet because *Confirm identity* has also the possibility to have *Receive notification* as predecessor, instead of only *Check term of notification*

and *Inform court*. Therefore the *Confirm identity* task is not a part of the cluster. The same holds for the task *Receive notification of birth*. This task is always a predecessor of the task *Check term of notification*, but the task *Check term of notification* is not always a successor of *Receive notification of birth*. Therefore the task *Receive notification of birth* is not included in the cluster. Conversely the tasks *Check term of notification* and *Inform court* are always linked to each other and are therefore a cluster. Like the tasks that are similar in every individual business process model, we assumed that if the relation between tasks is the same in every individual business process model they are mandatory. We therefore preserve the original formation of certain tasks and wanted behavior.

The size of the clusters is determined by the number of variations in a model and the extensiveness of the configuration freedom. If it is allowed to configure within a cluster it implies that the cluster becomes smaller i.e. a cluster must always be in the same formation if this is not the case it is not a cluster. When there are more configuration possibilities in a model, there are less formations in a model that are always the same. Therefore the formation is determining the size of a cluster.

During the remainder of the thesis we use the term configurable task for the cluster, because every cluster is one configurable task.

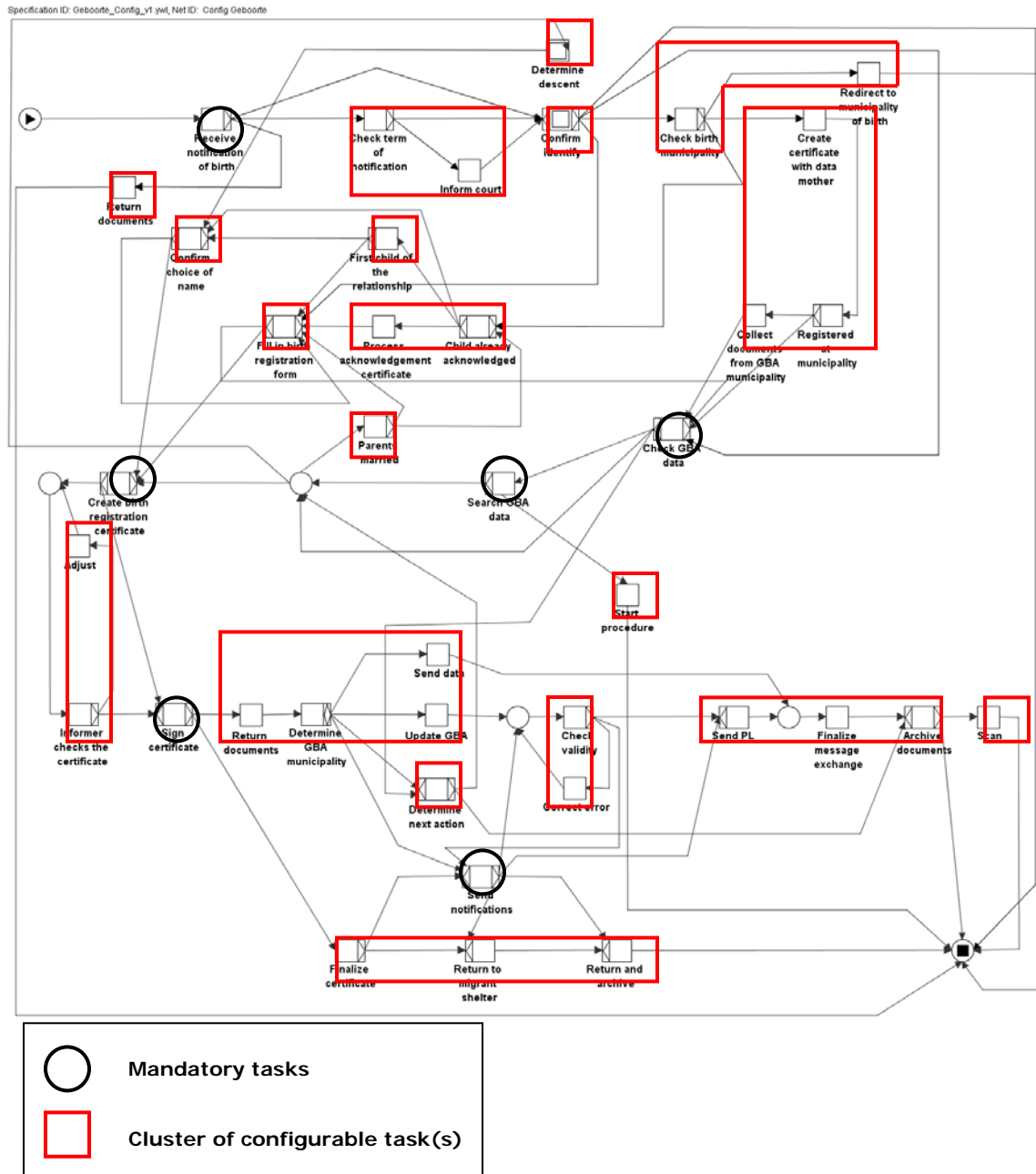


Fig. 3-20 Workflow model divided in clusters of tasks which are always executed in the same formation

3.5.2 Specifying questions for the questionnaire

Now we know which tasks can be configured we need to specify questions accordingly. We wanted to make sure that at least all individual *birth* business process models could be derived from the configurable *birth* workflow model by means of the questionnaire that we created. To achieve this, not every configurable task needs to have a unique question. Nevertheless when the decision is made to make a question for every individual task, it increases the complexity of the constraints and the mapping. There is a tradeoff between complexity of the constraints and the mapping, and the freedom of configuring every individual configurable task. With mapping we mean the translation of the answers of the questionnaire to a configured model (mapping will be discussed in section 3.6). When the complexity of the mapping and the constraints are too large it becomes error prone i.e. the probability of unwanted behavior increases and accordingly the chance of errors. The complexity increases when the number of constraints and mappings increase. There is no rule to what extend it is reasonable to have a question for every individual configurable task. If it is manageable to have a question for every individual configurable task, it can be done. It is also important to reason if it is useful to give the user much freedom. When there are too much questions it can also become user unfriendly. Because all those different aspects need to be taken into account it is difficult to find the correct tradeoff.

In section 3.5.2.1 we will explain how a question can be formulated which results in a task that is the subject of the question will be *blocked* or *activated*. The following section (3.5.2.2) describes how a question can indirectly configure a model.

3.5.2.1 Directly configuration by means of a questions

The user has an indication what he is configuring when a question is asked in the questionnaire. There is a relation between the question and the configuration of the model in a way that the subject of the question is related to according tasks. When a question only configures the tasks that are related to the subject it is a direct configuration. For example the cluster of configurable tasks *Check term of notification* and *Inform court* (section 3.5.1.1) occurs in one *Birth* business process model and checks if the child is registered within the first three days after the birthday. A question for the questionnaire, for this example, could be; *Do you want to have a task for checking the registration term?* The answer possibilities could be

Yes or No, where Yes activates, and No blocks the configurable task related to the term of notification of the birth. This question will directly configure the related configurable task. It is possible to create questions for all other configurable tasks in the same way. In this way the user knows what he is configuring. There is also a possibility that a question is not only configuring the related subject but also another part of the model. In the next section we will discuss this last mentioned type of question.

3.5.2.2 Indirectly configuration by means of a questions

Despite the fact that we already limited the freedom of configuring by guiding the configuration by the questionnaire approach and decreased the complexity of configuring by asking questions per configurable task, it is still possible that it is too complex. Another possibility to reduce the complexity is to derive configurations, that are not related to the subject of the question, from answers of a question. It can be the case that when a user decides to configure a task in a certain way, it also implies another configurable task should be configured. For example the task *Check birth municipality* has two possible configurations. It can be configured to flow to the task *Create certificate with data mother*, or to *Child already acknowledged*. A question for the configuration of this could be; *What should be the next task after checking if the child is born in the municipality?* The answer possibilities for this question are *Make birth document based on the data of the mother* and *Check if the child already is acknowledged*. If the user answers this question with *Check if the child already is acknowledged* this task will be activated and the task *Make birth document based on the data of the mother* will be blocked, by means of activating and blocking output ports of *Check birth municipality*. Besides the fact that the answer determines the configuration of the related task, we can also derive some other configurations of the answer. If we know that the task *Check birth municipality* will be followed by the task *Child already acknowledged*, we can derive that the task *First child of the relation* should be activated and the output port from the task *Child already acknowledged* towards the task *Confirm choice of name* should be blocked. The derivation of the configuring of the task *Child already acknowledged* is claimed by looking at the individual *birth* business process models. When the combination of *Check birth municipality* followed by *Child already acknowledged* is located in the individual process models, (this occurs in the business process model that is depicted

in appendix 4) we determine the tasks that always succeed this combination. Therefore we can state that the answer of the question (*What should be the next task after checking if the child is born in the municipality?*) is not only relevant for the direct related configurable tasks but also for indirect related configurable tasks. Therefore we don't have to create a separate question for the configurable tasks *Child already acknowledge* and *First child of the relationship*.

If a question with direct configuration also has indirect configuration it becomes a question with indirect configuration i.e. a question with indirect configuration has also direct configuration.

The indirect configuration decreases the freedom of the user to configure the model. Therefore it should be considered to what extend the indirect configuration should be used. For example, it is possible to derive one of the individual *birth* business process models, by asking one specific question. In this way the freedom of configuring is reduced to a minimum. The questionnaire creator must have a clear idea to what extend he allows the user to configure. By making use of direct and indirect configuration the most suited questionnaire can be created for the determined configurable freedom of the user.

3.5.3 Design dependencies between questions

If all questions are specified to configure the configurable tasks we need to create dependencies between the questions. The theory about creating dependencies is described in section 2.4.1. We give an example for a dependency between two questions that configure the *birth* configure workflow model. The question *Which task should follow up the identification task?* is fully depending on the question *Do you want to have a task for checking the registration term?* This dependency indicates that question *Which task should follow up the identification task?* can only be asked if question *Do you want to have a task for checking the registration term?* is answered. This makes it possible to keep the order of occurring of the questions manageable. Another possibility to use the dependencies is to link a question to a fact/answer. The answer possibilities for question *Which task should follow up the identification task?* are *Check if the child is born in your municipality*, *Fill in the birth registration form*, and *Check GBA data*. If the question is answered with *Check if the child is born in your municipality*, the question *What should be the next task after checking if the child is born in the municipality?* occurs. If the answer of question

Which task should follow up the identification task? is not Check if the child is born in your municipality the related question will not occur. When all questions are created and all dependencies are specified, the structure of question-fact for the birth questionnaire is displayed in Figure 3-21.

To make sure these dependencies and the questions will be displayed in the interface of the questionnaire in the right manner, constraints are needed.

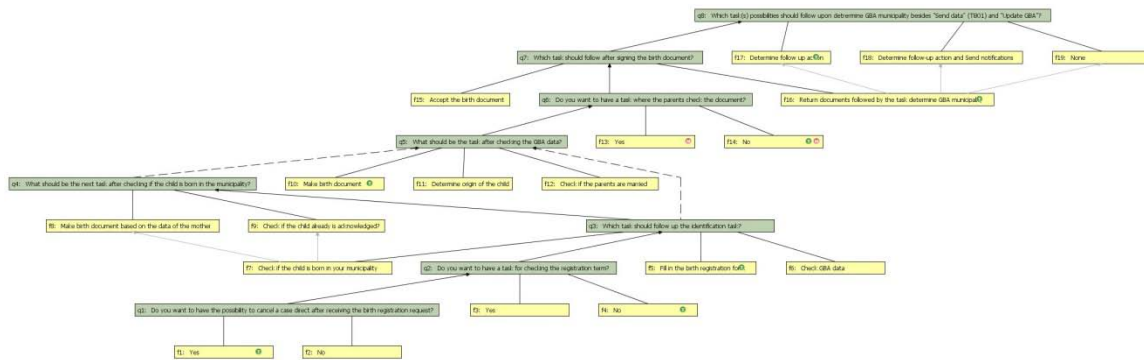


Fig. 3-21 Question-fact structure of the birth questionnaire

3.5.4 Creating constraints

The theory of creating constraints is described in section 2.4.1. The constraint of question q3 of Figure 3-21 is displayed in Table 3.8.

Behavior of q3	Constraint for q3
XOR behavior of q3	$(xor(f5,f6,f7))$
If f7 is true q4 succeeds	$(f7 \Rightarrow (xor(f8,f9)))$
If f5 is true q4 will not succeed	$(f5 \Rightarrow (-f8.-f9))$
If f6 is true q4 will not succeed	$(f6 \Rightarrow (-f8.-f9))$
If f5 is true q5 succeeds	$(f5 \Rightarrow (xor(f10,f11,f12)))$
If f5 is true q5 succeeds	$(f6 \Rightarrow (xor(f10,f11,f12)))$

Table 3.8 Constraints of question 3

When we combine these constraints, we will get the constraint for q3; $(xor(f5,f6,f7)) \cdot (f7 \Rightarrow (xor(f8,f9))) \cdot (f5 \Rightarrow (-f8.-f9)) \cdot (f6 \Rightarrow (-f8.-f9)) \cdot (f5 \Rightarrow (xor(f10,f11,f12))) \cdot (f6 \Rightarrow (xor(f10,f11,f12))) \cdot (f8 \Rightarrow (xor(f10,f11,f12)))$.

We created the constraints for the other questions in the same way. During the creation of these constraints we experienced that if the number of constraints is

increasing, it became more error prone. By reducing the number of questions by means of clustering (section 3.5.1.1) and creating indirect configuration (section 3.5.2.2), the structure of the question-fact tree became more orderly.

By creating the constraints we completed the creation of the questionnaire. First we created the questions, created the dependencies, and created the constraints. All these activities are created in the *Questionnaire designer*. The *Questionnaire designer* is the design application of the questionnaire and creates a file that can be imported into *Quaestio*. *Quaestio* is the interface of the questionnaire and is used by the user to answer the questions. By executing every step in section 3.5, we completed the creation of the questionnaire.

3.6 Mapping

After the creation of the questionnaire we are not finished with creating a guidance to configure a configurable workflow model. There is a crucial link missing between the configurable workflow model and the questionnaire and this is *Phase 6* (Figure 3-22). When the questionnaire is answered by a user, the configurable workflow model is not yet configured accordingly. By means of mapping, this missing link is created. The mapping indicates how tasks need to be configured according to the answer combinations the user supplied. The mapping is very important because it prevents

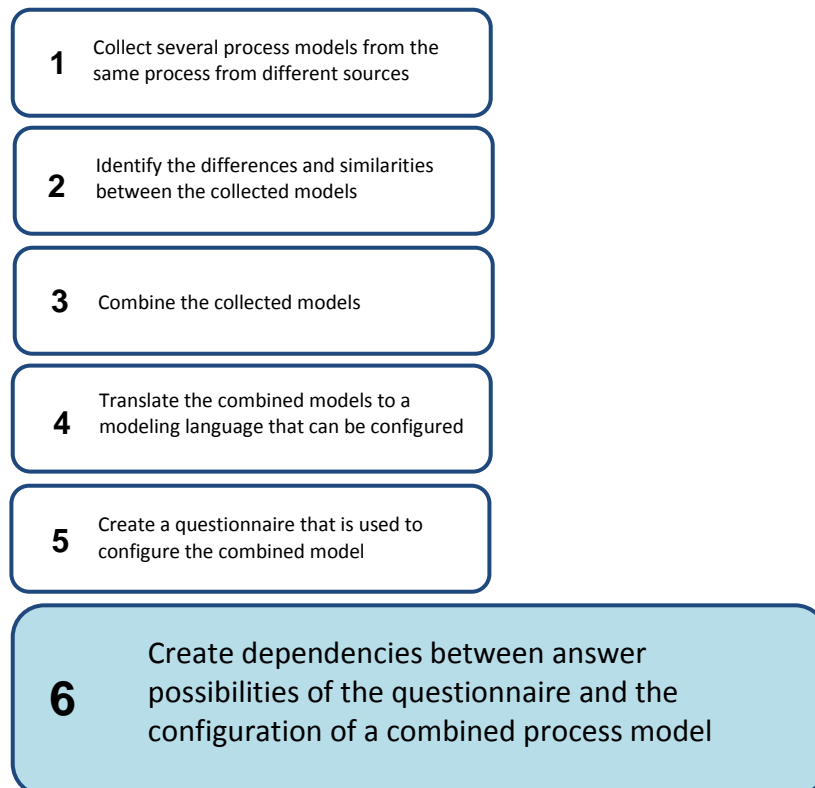


Fig. 3-22 Phase 6: Create a mapping

unwanted behavior. Conversely if the mapping is not done properly it will cause unwanted behavior.

In the next section we describe how we created the Excel compilation sheet for the configurable *birth* workflow model. Subsequently the transformation of the Excel compilation sheet in to the xml file will be described. We conclude with an explanation of how the created files should be used in combination with the *Configurator* and *Individualizer*.

3.6.1 Excel compilation sheet

The answer possibilities of the *birth* questionnaire, which are depicted in Figure 3-21, influence the configurable model. Identifying for each answer possibility the influence it has on the configurable workflow model, results in the mapping. We already identified clusters of configurable tasks (section 3.5.1.1), so we know that the tasks that are part of a cluster cannot be configured in a way the formation of the cluster changes. We can configure the input and the output flow of a cluster. In this way the configurable model can be configured.

The mapping of answers onto the model is done by using the relations that are identified by the direct configuration (section 3.5.2.1) and indirect configuration (section 3.5.2.2). Each of these relations belongs to a certain answer. All the answers and the questions are depicted in Figure 3-21. For every answer we identify what the configurable consequences are. We will use the example from section 3.5.2.1. This example is about the answer *f9* depicted in Figure 3-21. If *f9* is true, the output port of task *Check birth municipality* that flows to task *Create certificate with data mother*, must be blocked in the configurable *birth* workflow model (Figure 3-16). The output port from task *Check birth municipality* that flows to *Redirect to municipality of birth* must always be activated, because if the check that is performed is negative the municipalities must have the possibility to refuse the registration by sending the person to another municipality. The only configurable output port of the task *Check birth municipality* that is not configured yet in Figure 3-16 is the output port that flows into *Child already acknowledged*. This outflow port must be activated when answer *f9* is true. When the output ports of task *Check birth municipality* are configured like we just described, the flow succeeding task *Check birth municipality* has a XOR- choice. The flow can continue to task *Redirect to municipality of birth* or to *Child already acknowledged*. Depending on the decision

the workflow user makes, one of these two tasks will succeed. In section 3.5.2.2 we concluded that when task *Child already acknowledged* could succeed task *Check birth municipality*, the output ports of *Child already acknowledged* also must be configured. The output port towards the task *First child of the relation* should be activated and the output port towards the task *Confirm choice of name* should be blocked. The notations of these configurations that are related to answer *f9* are depicted in Table 3.9. We only notate the ports that need to be blocked or hidden when *f9* is *True* in Table 3.9. By notating the flow from task to task we indicate which port needs to be blocked or hidden. We notate for every answer what the consequence are in the same manner as in Table 3.9. The result of this is combined in one Excel sheet, the Excel compilation sheet (Appendix 36). The Excel compilation sheet is in Dutch because the original model is in Dutch to make it understandable for the municipalities. The Excel compilation sheet from appendix 36 depicts that answer *f9* has more configurable consequences than the one we mentioned in our examples. In the column of answer *f9 Blocked* is notated six times i.e. six ports are configured when *f9* is true. For every answer the Excel compilation sheet depicts which ports need to be blocked and for every port when it is blocked or hidden. This will be the starting point of the XML file that we will create in the succeeding section.

Flow	Fact: f9
Check_birth_municipality_5 → Create_certificate_with_data_mother_52	Blocked
Child_already_acknowledged_78 → First_child_of_the_relation_79	Blocked

Table 3.9 The notation of the configurations of fact *f9* in the excel compilation sheet

3.6.2 XML file

The XML file that we are creating based on the Excel compilation sheet is needed to import into the *Configurator*. The *Configurator* uses an XML file to map the answers that the user provide, on to the configurable workflow model. The mappings of answer *f9*, which are depicted in Table 3.9, will be transformed in the following XML:

```
<splits>
  <port sourceId="Check_birth_municipality_48"
  targetId="Create_certificate_with_data_mother_52">
    <value type="blocked" condition="f9"/>
  </port>
  <port sourceId="Child_already_acknowledged_78"
  targetId="First_child_of_the_relation_79">
    <value type="blocked" condition="f9"/>
  </port>
</splits>
```

The first line, `<splits>`, in the XML code indicates that an output port is configured, followed by a line that describes the port that needs to be configured. The `sourceId` indicates the task which port will be configured. The `targetId` indicates the destination task. The notations between quotes are the `engineIds` that can be located in the YAWL editor. It is important that these `engineIds` are notated without any errors, otherwise the *Configurator* cannot match the `sourceIds` and the `targetIds` of the mapping, with the `engineIds` of the configurable YAWL workflow model. By means of this information the *Configurator* can identify the port in the YAWL-xml file. Now the location is specified, the *Configurator* needs to know in which situation what needs to be configured. The situation is given by the `condition`, in this case if `f9` is true, and the configuration is given by `value type`, this case is `blocked`. There are three `value types`; `blocked`, `activated`, and `hidden`. This notation structure is repeated for every output port that can be configured. When all output ports that can be configured are specified in XML code, it is closed with `</splits>`.

The notation of the configuration of an input port is slightly different. The configuration of an input port is done in the following structure:

```
<joins>
  <port sourceId="null_140" targetId="Parents_married_109">
    <value type="activated" condition="-f10 ∧ ¬f11"/>
    <value type="blocked" condition="f10 ∨ f11"/>
  </port>
</joins>
```

The only differences that can be observed compared to the output notation, is that it starts with `<joins>` indicating that it is an input port, and there is explicitly specified when the port is activated. The notation of the output port only indicates when it needs to be blocked or hidden. We choose, if possible, to configure the output ports of the tasks, because the XML notation of the output port is shorter. It is not always possible to reach the desired behavior by configuring the output port. When a flow from a condition to a task needs to be blocked or hidden, there is no output port to configure. Conditions don't have ports, therefore the input port of the destination task needs to be blocked or hidden.

When the whole Excel compilation sheet from appendix 36 is transformed to XML, it will result in the XML code that is depicted in appendix 37.

4 Results

During the case study we did several observations related to the creation of the configurable workflow models. These observations give an indication of the possibilities and the difficulties of creating configurable workflow models. In section 4.2 these observations will be described followed by a description of the evaluation with stakeholders in section 4.3. But first we will give a general guidance of how we created the configurable workflow models.

4.1 Generalization

We applied the configurable approach from [6] during the case study for civil affairs processes at municipalities. The approach can also be used to create configurable models from other processes than civil affairs processes. Therefore, we have created a basic guidance for creating configurable process models. In Figure 4-1 are the six phases, that we used to create a configurable workflow model, depicted.

The first phase depicts how we collected the business process models for the business process that needs to be configurable. Several organizations, that are executing the same process, were visited to inventory how the process is executed. For every visited organization a business process model is created. The modeling language that is used to create the business process models can be several. During our case study we used Protos because the municipalities were familiar with this language. It is also possible to create the business process models directly in YAWL to avoid translation to YAWL in phase 4. When the selected modeling language is not Protos or YAWL we don't have a translation in this guidance.

When all business process models from different organizations were created, they must be compared (*Phase 2*). The techniques that are described in paragraph 3.2.1 and 3.2.2 are used to identify the similarities and the differences. When during the comparison appears that there are no or limited similarities between any business process model it is doubtful to combine the models into one configurable model. It is then wise to keep the business process models separated instead of combined. If the opposite occurs i.e. all the business process models are similar, it is also doubtful to combine them to one configurable model because there is almost no added value.

When the business process models are not similar or different, they can be combined. The combining of the business process models is done in *Phase 3*. The

techniques to combine the business process models are described in 3.3. During the third phase we merged the *actions* that are the same and added the different *actions* in the right place in the model as described in 3.3 until all models are integrated into one model. When all models were combined to one model, we executed a check to identify if all traces/flows that were possible in the individual business process models are also possible in the combined business process model.

When all traces/flows are present in the combined business process model, it can be transformed to YAWL (*Phase 4*). If the business process models are already created in YAWL and therefore the combined business process model is also in YAWL, the transformation to YAWL is not needed anymore. If the modeling language is not YAWL, a transformation to YAWL is needed. The transformation is needed because the application that configures the model based on the result of the questionnaire only works with YAWL at this moment. In our case Protos was used as modeling language the approach of 3.4 can be used to transform Protos manually into YAWL. If the modeling language is not Protos a translation from the specific modeling language into YAWL should be used. The translation from other modeling language to YAWL is not part of the thesis. When we did the translation we had a YAWL combined business process model. This model needs to be transformed into a workflow model. The transformation is done by adding data to the model, like described in 3.4.3. By adding data the workflow model knows when to execute a specific data element.

When all data that is needed for the workflow was specified the constraints needed to be specified. The constraints make sure that only workflows can be configured that have behavior that is wanted. The adding of the constraints transformed the combined YAWL workflow model into a configurable YAWL workflow model. The constraints specify the configurable possibilities and restrictions of the model. In paragraph 3.5.4 is described how constraints can be created. The creation of the constraints is part of *Phase 5* create questionnaire. The creation of the questionnaire is described in paragraph 3.5 and can be used as guidance.

When the configurable YAWL workflow model and the questionnaire were created the link between those two needed to be made. The link was made by an XML file. To support the creation of the XML we create an Excel file. In paragraph 3.6 is described how to create the Excel and the XML file. The XML file specifies what the consequences are of certain answer results of the questionnaire. When all files were

working properly they were loaded into the *Configurator* and subsequently in the *Individualizer* like described in paragraph 2.5.

When we executed every described phase it results in a configurable YAWL workflow model.

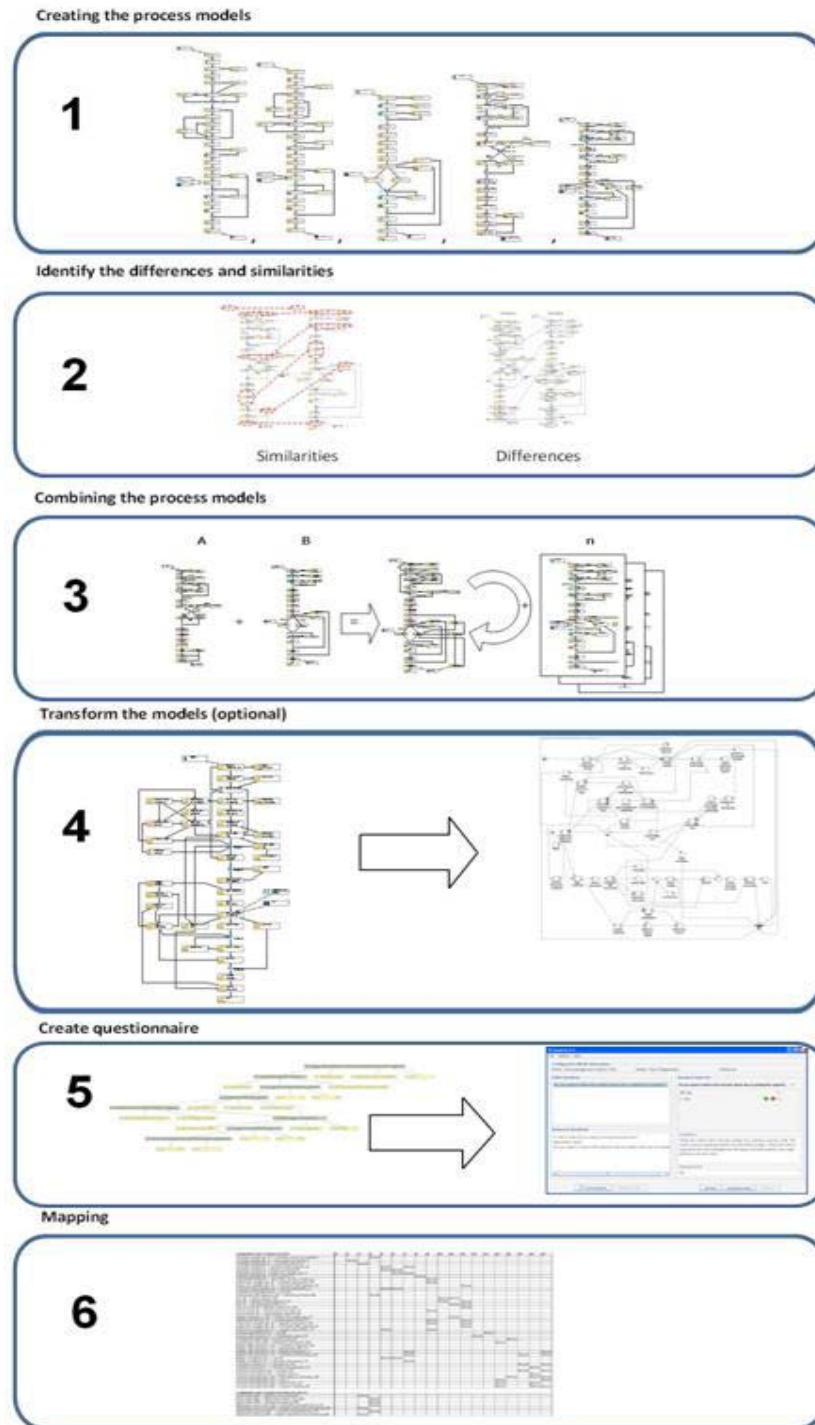


Fig. 4-1 Depiction of the generic guidance to create configurable YAWL workflow models

4.2 Observations

During the case study we had to master some difficulties. We look at chapter 3 from start to end to describe the observations we did during the creation of the configurable workflow model, starting with the data collection.

- *Data Collection* – All the municipalities were able to give a description of the selected process in a way we could create a business process model out of the information. Some municipalities supplied us with their business process models to support their descriptions of the processes, what improved the creation of the business process models. The data collection was straight forward and is not a bottleneck in the process of creating the configurable workflow model.

- *Combining models* – During the combining of the individual business process models to one combined business process model, we encountered the first challenges. Instead of only looking to the names of the task to identify if the tasks are the same or different, we needed to look into the descriptions of the task to identify if the tasks were really the same. It could be the names were the same but the content of the tasks was different or vice versa. Therefore we did not depend on the names of the tasks when we identified the differences and the similarities. We used categories of [30] to identify the similarities and differences, so we could check what type of similarity or difference it was. These categories created in our opinion a clearer identification of the tasks.

After the identification of the differences and the similarities we could combine the models. The combining of the models became harder as the number of tasks and conditions increased. Because of the combining of models more paths were possible. We needed to check for every individual business process model that was merged into the combined process model if the paths that were possible in the individual business process model were also possible in the combined business process model. The more paths an individual business process model contains the more likely it was to overlook a path. Therefore we introduced a simple check to support the combining of the models.

Another challenge was to model different orders of execution. Some situations occurred in which the order of tasks amongst individual business process models was

different. We presented three ways of modeling of an order of tasks to make it possible to derive different orders from the combined business process model.

- *Transform the models into YAWL* – The translation of the Protos business process models into the YAWL workflow models was straightforward. YAWL consists of all patterns that are used in Protos and this simplifies the translation. The tricky part is the creation of the data perspective. Especially the data that was part of a task, which consists of run-time and configuration-time decisions were tricky to create. When an OR-split has four outgoing arcs and two of these arcs are mandatory, we need to anticipate on the implications when one of the two remaining arcs is configured. Therefore we need to take into account the order of checking the conditions of the ports by the workflow engine.

The behavior of a configurable model is created by combining the behavior of the merged individual business process models. It can be the case that it is useful to add some behavior to the configurable model but only when it enriches it. A model is enriched when the probability of executing the extra behavior, is high.

- *Creating the Questionnaire* – The questionnaire is created, first of all to make the configuration more user-friendly, but also to control the configurable freedom of the user. It was hard to find a good tradeoff between the freedom of the configuration and the control of the configuration. We want to give the user a much freedom to configure but we want to make the possibility of unwanted behavior in a configured workflow model as small as possible.

The creation of a questionnaire with all its dependencies and constraints became more complex when the number of questions and answer possibilities increased. The complexity of the questionnaire needs to be manageable otherwise the questionnaire will have the possibility of errors. The complexity therefore also needs to be included in the decision of the structure, number of questions and answers. The trade between complexity and the freedom of configuring and the control of configuring, make a difficult to create a questionnaire.

- *Mapping* – The observation we did when we were creating the mapping was that we did not use the configuration possibility *hiding* frequently. In Figure 4-2 we depict an example of the situation we created when we combined the models. For example, a model *I* has the action *A*, followed by action *B*, followed by action *C*, while a model

// has the action *A* followed by action *C*, and does not have the action *B*. We combined the two models by drawing an arc from action *A* to action *C*. Instead we could also eliminate the arc from action *A* to action *C* and still have the possibility to configure the flow from action *A* to action *C*. By hiding the input port of action *B* we also have configured the flow from action *A* to action *C*.

The mapping determines which tasks need to be configured and when. This means the mapping is important, because it controls the configuration. The mapping

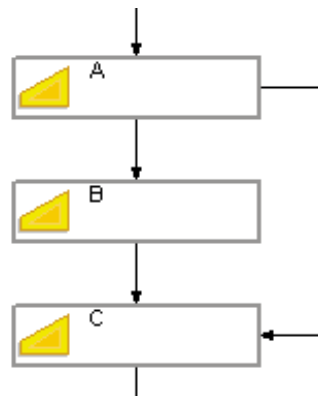


Fig. 4-2 Configuring the model by blocking the output port from *A* to *B* instead of hiding the input port of *B*, to create the situation in which *A* must be followed by *C*.

must eliminate the chance of unwanted behavior. To check if there is no unwanted behavior caused by the mapping, the questionnaire needs to be answered with several different answers. The result of the questionnaire must be mapped on to the configurable workflow model, by means of the *Configurator* and the *Individualizer*. The resulting configured workflow model must be checked if it is corresponding with the answers and if it does not have unwanted behavior. The more answer combinations a questionnaire has, the harder it is to guarantee that no unwanted behavior will occur in the configured models.

Despite of the challenges we encountered we were able to create for the *birth* business process a configurable workflow model. Not only for the *birth* business process we succeeded to create a configurable workflow model, but also for the three other processes, *Acknowledgement of an unborn child*, *Deceased*, and *Marriage*. For each of the configurable workflow models we were able to derive the original individual business process models, by means of answering the questionnaire. This indicates that it is possible to combine several individual business process models to

one configurable model, where by configuration the individual business process models can be derived.

The complex process of creating configurable models and the according questionnaire is helping to simplify the configuration by the user.

4.3 Evaluation with the stakeholders

Until now we described how we created the configurable workflow model and the questionnaire during our case study. But now that we have described our own observations, we want to know how the different stakeholders think of our approach. We asked five stakeholders for feedback:

- The end-user, in our case the municipalities
- Software developer
- Pallas Athena
- Consultancy firm
- NVVB

The five stakeholders are all related to our case study. Only one municipality evaluated our research instead of four municipalities because of lack of time. The selected municipality for evaluation can use the configurable workflow model and the questionnaire to specify their processes, the software developer can offer a customized implementation of their software to their customers (municipalities in our case), Pallas Athena can enrich their business process management software with the configurable approach, the consultant can apply the configurable approach on their collection of business process models of their customers, and the NVVB can use the approach to enrich their reference models.

We visited the five stakeholders and gave a presentation at four stakeholders about our work. We did not do a presentation at the municipality because it has no value to them to know how the technique behind the configurable approach is working. For them it is more important to know if the configurable approach is helping to create a process model that suits their needs. The only things we checked at the municipality were, if we could generate a model that satisfies their needs, and to know that they think about the questionnaire and creating configurable workflow models.

The presentation we did at four stakeholders ended with a discussion. We initiated the discussion by introducing questions sequentially. We introduced the questions sequentially and not in parallel to prevent the stakeholders from reacting

only on the questions they found interesting. The reaction on the questions indicates how they value our work. The three main questions that we used to initiate the discussion were:

- Is there a need for configurable models?
- Is it feasible to create configurable models?
- Are configurable models useful?

For every main question we created a list of sub questions in case the stakeholder did not really answer our question.

We will describe the results of the evaluation with the stakeholders in the following section. This will give an indication of the opinion of the stakeholders.

4.3.1 Pallas Athena

Pallas Athena identified that there is a need in the market for configurable models. The configurable approach could enrich their product by providing extra process model options. Instead of providing one standard solution that must be modified to the needs of the customer they could provide a solution with several configuration options.

For a recent project they completed, it could have been useful to use the configurable approach. In this particular project, which is linked to the introduction of a new law, there is one “one point of contact” workflow product. This requires from the municipalities that they restructure their business processes. The business processes that are related to this new law are highly standardized because they depend on the law and therefore are suitable for the configurable approach. The standardized business processes will not differ much from each other and the differences can be used as configuration options. Because it is a “one point of contact” workflow product, there is one main business process model with several related sub business process models. This whole package could be made into a configurable process model and could be easier to maintain and enrich the product Pallas Athena provides.

Pallas Athena was also very interested in process mining to create the models from the log files of a customer and compared those to similar processes at other customers. This will give a representation of the business process as it actually is performed at the customer. They would like to see the integration of mining in the configurable approach.

They were concerned if it can be guaranteed that all constraints are in the model and thus can be guaranteed no unwanted behavior will occur. The idea of the constraints linked with the questionnaire was positive in their opinion. This questionnaire could also be useful for clients who are not familiar with process modeling notation.

4.3.2 Software provider

The software provider we selected has over two hundred municipalities as client. They provide software that supports the processes at the municipalities. They were positive about the configurable approach even without seeing the working configurable YAWL workflow models and the questionnaires. The configurable YAWL workflow models and questionnaires are really an addition but the approach itself is already very interesting in their opinion. They were very positive about configuring process models and mentioned that it could be interesting for them to develop one standard process model with additional options that can be configured. This standard can be created by specifying with the constraints that this part cannot be configured and the additional options can be configured by means of the *Quaestio* tool. The software provider had some doubts if configuring the models could cause unwanted behavior in the resulted model, which is why they were really positive about the constraints that can be specified and used in the mapping and the questionnaire. The constraints are needed to specify which parts of a model are mandatory and which are optional.

They identified that it could also be nice to link the questionnaire answers to other configurable elements, like the configuration of software screens and windows as well as data fields. In this way they could customize the software for a customer by means of the questionnaire. The customization is often requested by customers and is at this moment expensive due to need of external consultants. The questionnaire also gives the customer the feeling of freedom instead of restriction. Another positive point is that they can show client's possibilities of how other clients use the model. In this way clients learn from other municipalities and the software provider gets to know what the most frequent used configurable options are. The difficulties they identified are related to the modeling level e.g. should models be defined on a high level or on a detailed level. They mentioned that some municipalities have software support when they identify a person while other

municipalities do the identification without software support. This should require a data configuration to enable or disable the software support of a certain action. The possibility to configure resources was a thing they were missing and would like to see if this would be possible in future.

4.3.3 Consultancy firm

The consultancy firm identified that the configurable approach could be useful for a recent project they performed for one of their clients. The client requested a world-wide role-out of new business processes. The policy the client has is that 80% of the processes must be the same in all countries and a 20% deviation caused by local legislation is allowed. The deviation has been manually modeled for every country by the consultancy firm while, if they had the possibility to use configurable approach they could have used the configurable approach to configure the 20% deviation. Not only for this specific client could the configurable approach be useful for the consultancy firm but for several. They visited their clients with their own best practice models to prevent that they had to start from scratch. Because there is not just one best-practice according to the consultancy firm, they frequently updated their best-practice models, but the changes were not stored. If the changes and updates are stored they can be used for the configurable approach to create configurable process models. They acknowledge that this would be a nice possibility but they are skeptical because there is no software support. In their opinion software support would make it more accessible for the consultants to use and maintain configurable process models. They also noticed that it can be tricky to use certain data for configurable models when it is used in several countries e.g. postal codes are different amongst countries and there is no standard so for every country a specific postal codes data element should be specified.

They identify that, besides secondary process in some industries the core processes are also so standardized that the configurable approach could be applied. The consultancy firm is also skeptical about the creation of configurable models. The creation of configurable models seems to require big efforts. Because the creation of configurable models is not commissioned by a client, the sponsoring should be funded internal. To get the internal support can be difficult. In their opinion it can be easier to store several variations of process models instead of one configurable model, only they immediately identify that it is hard to detect the variations between

the different process models. They identified a need for a tool that can automatically detect differences and so could support the creation of configurable process models.

In Table 4-1 we summarized the key results of the discussions with the stakeholders. The stakeholders were positive about the method of creating configurable models. During the presentation to explain the work we did, most of them could already identify a project that is or was suitable to apply the method. This indicates that there is a need for a method that we developed.

Despite the need for the configurable method, there were still some slight concerns on the method. These concerns were related to the completeness of the configurable model, no configuration support for resources, sponsoring of creating the models because of the big modeling efforts, and the identification of differences and similarities. Overall the concerns were in the minority and most reactions were positive.

4.3.4 NVVB

The visit to the NVVB was different from the other visits because we also needed to ask permission to publish their reference models. We presented our work to them and asked them their opinion. The configurable process model approach is in theory interesting for the NVVB because they create for a large number of municipalities reference models. These reference models are bought by the municipalities and adjusted to the situation of the buyer. The adjustment of the reference model is done by hand and is therefore time consuming. By using the configurable process model approach they can satisfy the needs of their customers i.e. the customer does not have to adjust the model by hand or at least the adjustment is less.

The NVVB was impressed by the ideas and the work we did. The idea they had was to use the configurable approach to monitor which configuration was used most for a model. The outcome of the monitoring can be used to create a reference model. On the other hand they were really protective about the use of their reference models and do not expect to use the configurable process model approach in the near future because of the translation of the scientific work to daily applications was difficult for them.

4.3.5 Municipality

The evaluation with the municipality was different from the evaluations with the other stakeholders, because the municipality is an end-user. The end-user is not concerned how the method is working and how it could be applied to their products. They can be the ones that use the questionnaire to configure a process model, which they want to purchase, to their situation. The most important factor for end-user is that the purchased product is satisfying their needs.

We executed with the municipality the configurations by means of the questionnaire for the four business processes. For all the business processes we were able to create a configured model that satisfied their needs (Appendices 36-39). They were positive about the method and the resulting models. The immediately resulting models were considered as a big advantage by the municipality. Another advantage that they mentioned was to have an indication of the processes execution of other municipalities. The configuration points are the result of different execution of processes amongst municipalities. Besides the positive reactions there were also some remarks from the municipality. The remarks concerned the way of asking the questions. We created the order of appearing of the questions according to the order of occurring of the configurable points in the configurable model. The municipality indicated, during answering, that they did not use one way of executing the model, but that they use the model as a guide.

If every step in the model is executed, it will result in a correct outcome, but other orders of execution of the business process model could also result in a correct outcome. They use the business process model for the audit and to support new employees. The experienced employee adapts the process immediately to situation that occurs at their desk and doesn't use a business process model as a guide. Because the interview partner was an experienced employee he immediately mentioned, when a question was asked, several activities that could be needed. The activities he mentioned were also part of the configurations but the questions that deal with these activities did not appear yet. Therefore we recommend displaying most of the questions, when an experienced user answers the questionnaire, in the questionnaire interface so the user has an indication which configuration subjects will be treated during the questionnaire. Because we displayed the questions sequentially the user was triggered to mention activities he thought did not consist in the questionnaire.

Interview Partner	Potential applications and advantages (+) as well as concerns (-)
Pallas Athena	<p>(+) Configurable Process models would have been useful for the development of a “one point of contact” workflow product developed for municipalities based on a new law that requires municipalities to re-structure the customer interaction of their business processes?</p> <p>(+) Potential applications are highly regulated, publicly documented and accessible, or non-core business processes like HR processes.</p> <p>(-) The integrated model must be complete. Is this possible and how can this information be derived from existing processes?</p>
Software provider	<p>(+) Questionnaire answers can be linked to other configurable elements, like the configuration of software screens and windows as well as data fields.</p> <p>(+) Configuration through questionnaires enables software providers to create applications that prevent users to fail during the process configuration.</p> <p>(+) A user likes to see in the questionnaire the configuration freedom she has rather than the limitations the configuration is subject to.</p> <p>(+) Clients often ask for software adaptation and modifications for a better support of their desired business processes which is currently expensive due to the need for external consultants. Currently, this often results into workarounds.</p> <p>(-) A configuration of the resources that are involved in a process is not possible.</p>
Consultancy Firm	<p>(+) Best-practice reference models are often not sufficient: there is no single best-practice.</p> <p>(+) It would have been useful, in a world-wide role-out of new business processes where it was a headquarter policy, that 80 % of the processes needed to remain conform to the global process while it was allowed to deviate by 20 % to make the process compliant to local regulations.</p> <p>(+) In some industries production processes are so standardized that the technique might even be applicable to core processes.</p> <p>(-) The creation of such models seems to require big efforts, sponsoring for this might be difficult to find.</p> <p>(-) The identification of variations between processes is difficult, i.e. tools are necessary for this.</p>
NVVB	<p>(+) Positive about configurable process models</p> <p>(+) Configurable process models can be useful to create a reference model</p> <p>(-) The translation of the scientific work to daily applications was difficult in their</p>

	situation.
Municipality	(+) The immediately resulting models were considered as a big advantage by the municipality. (+) Have an indication of the processes execution of other municipalities. (-) Asking questions in a sequential manner while they do not think sequential

Table. 4-1 The main comments of the interviewed stakeholders [46]

By displaying all questions the user has a better understanding if, in his opinion, the questions are complete. The questions we asked were sufficient to configure the model, only the order was confusing for the experienced employee.

5 Discussion

5.1 Discussion

At the start of the case study our objective was to develop a set of configurable workflow models for common processes in municipalities to evaluate the feasibility of configurable workflow models. We accomplished to create a set of configurable workflow models out of several common process models. The selected common processes of the civil affair department are *Marriage*, *Birth*, *Acknowledgement of an unborn child*, and *Deceased*. For every process we succeeded to develop a configurable workflow model. Despite the fact that the processes are standardized because they are guided by law, there was still enough variation amongst the processes. This indicates that for standardized processes of municipalities it is possible to create configurable workflow models. The challenges of creating configurable workflow models are located in indentifying the differences and similarities, modeling different order possibilities, defining the data flows, and to guarantee that all the individual models could be derived from the configurable workflow model. These challenges are tricky because they are all error prone. Therefore it is important to check these challenges thoroughly. The approach of combining models we used during the case study has as side effect: it will reduce the number of hidden tasks, but will increase the number of blocked tasks. The possibility to configure the configurable process models by means of questionnaires is feasible. All the initial individual business process models that were collected from the municipalities can be derived from the configurable workflow model by means of answering the questionnaire. The creation of the questionnaire is challenging because the right mix of configuration freedom vs. restriction needs to be identified. A lot of freedom means many possible outcomes. The more outcome possibilities the harder it is to make sure the configured model does not contain unwanted behavior. Therefore an easy solution is to limit the freedom of the user, but this is against the idea of configuring. These considerations are challenging and depend on the intentions of the creator of the questionnaire.

The representation of the questions in the questionnaire should be adapted to the person that is answering the questions. For example the experienced employee of the municipality preferred to know from the start which questions will be asked. It

is to guide an experienced employee through their own process in one specific manner, than an inexperienced employee.

The link between the answers of the questionnaire and the configurations are captured in the mapping. The mapping is essential to guarantee the answers are resulting in wanted behavior of the configured model. Therefore the mapping also should be checked thoroughly.

The stakeholders were all very interested in the approach of configuring process models. They identified several possibilities to apply the approach. There were current projects and future projects from the stakeholders that were suitable to apply the configurable approach. They saw opportunities for the configuration approach in the area of resources and data, and configuring interfaces of software. The configurable approach has potential to be adapted by industry.

5.2 Reflection

The time it took to create the configurable workflow models was underestimated. The knowledge we now have will prevent underestimating the time it takes to create configurable workflow models. The time consumption was mainly cost by two factors. The first factor was underestimating the creation of configurable workflow models. The underestimating was caused by the lack of knowledge about the challenges we would encounter. The second factor was the bad planning we made. The planning did not take into account political issues.

Another issue that could be improved the next time we create configurable workflow models is the collection. With the knowledge we now have we should collect the process models from the municipalities at the municipality by observing the execution of the processes. This will make sure we have the process models that are equal to the real situation.

The presentation of the order of the questions should be done in different fashion. We presented the questions sequential while the civil servant was already thinking about all the exceptions that where possible. Therefore we would present all questions at ones so he has an indication what configurable variations are present. The presentation should depend on the user because a user with less knowledge of the process model could prefer a sequential presentation.

At the end we could create, with gather knowledge we have, more efficient configurable workflow models.

5.3 Future Work

The case study to check the feasibility of the configurable approach had some limitations. By focusing on these limitations, the approach could be further improved. Therefore we suggest the following topics for further research.

During our case study we focused on the control flow perspective of the business processes and not on the resources. Like configuring the control flow of a model it could also be useful to configure the resources. In that way the roles that are authorized can be configured to the specifications of an organization. The same holds for the distribution of the work to employees by different levels of authorization. By focusing on this subject the configuration approach we used could be enriched.

Another possibility is to configure the data of a task. During the case study we configured the data flow for the whole model but not on the level of individual tasks. The data flow in our approach can only be configured by *blocking*, *hiding*, or *activating* of tasks. This causes the presence or non-presence of the data depending on the configuration of the tasks. It is correct but it could be wanted that the data for every individual task should be configurable. In the future it could be interesting to investigate the possibilities of configuring data per task.

During the case study we had some issues that could be improved by further tool support. For example the identification of the differences and the integration of the models could be supported by tooling. The check if all traces of an individual business process model are present in the combined model could also be supported by a tool. Further consistent identifiers for the mapping and the guarantee that the layout of the YAWL model isn't disturbed after the configuration could be improved. Likewise the creation of constraints of the questionnaire could be supported to make the approach more accessible for a larger group. These tool supports would improve the user friendliness.

During the case study we did not find the limit of the number of models that we combined. It would be nice to have an idea what the added value is of combining one hundred models instead of four models. Would the model increase every time a model is added or will it stay at the same size at a certain point because the model already consist all the variations? Is the limit of the number of models that can be

combined depending on the number of tasks or to what extent the model is standardized or even something else? By answering these questions with a research an indication to what extent the number of combined models has extra value can be answered.

References:

- 1 <http://www.bprbzk.nl/dsc?c=getobject&s=obj&objectid=529&!sessionid=1CW RZ5zCSrofBaqys3l14qVBDU@p5K78Ld!zmM!6Az1JlodrhoUhCp3M4aGxJh@O&!dsname=BPRextern>, 8th November 2008.
- 2 T.H. Davenport and J.E. Short. The New Industrial Engineering: Information Technology and Business Process Redesign. *Sloan Management Review*, 1990, pp11-27.
- 3 www.nnvb.nl, 2008
- 4 J. M. Küster, J. Koehler, K. Ryndina. Improving Business Process Models with Reference Models in Business-Driven Development. *2nd Workshop on Business Processes Design (BPD'06)*, LNCS, Springer-Verlag, September 2006.
- 5 Workflow Management Coalition. *The Workflow Reference Model (version 1.1)*. Technical Report TC00-1003, Workflow A Management Coalition, 1995
- 6 F. Gottschalk, W.M.P. van der Aalst, M.H. Jansen-Vullers, M. La Rosa. Configurable Workflow Models. *International Journal of Cooperative Information Systems (IJCIS)*, Vol. 17, No. 2, June 2008, pp. 177--221.
- 7 J. Mendling, H.A. Reijers, and Jorge Cardoso. What Makes Process Models Understandable? In G. Alonso, P. Dadam and M. Rosemann, editors, *Proceedings of the 5th International Conference Business Process Management (BPM 2007)*, *Lecture Notes in Computer Science* 4714, 48-63. Springer Verlag, Berlin, 2007.
- 8 C. Schell. The Value of the Case Study as a Research Strategy. January, 1992. Manchester Business School
- 9 Yin, Robert K. Yin. Case Study Research. Design and Methods. Third Edition. *Applied social research method series Volume 5*. Sage Publications. California, 2002.
- 10 Users manual Protos Classic 8.0, Apeldoorn: Pallas Athena, 2006

- 11 W.M.P. van der Aalst and A.H.M. ter Hofstede. YAWL: Yet Another Workflow Language. *QUT Technical report*, FIT-TR-2002-06, Queensland University of Technology, Brisbane, 2002.
- 12 H.M.W. Verbeek, W.M.P. van der Aalst, and A.H.M. ter Hofstede. Verifying Workflows with Cancellation Regions and OR-joins: An Approach Based on Relaxed Soundness and Invariants. *Computer Journal*, 50(3):294-314, 2007.
- 13 M.T. Wynn, D. Edmond, W.M.P. van der Aalst, and A.H.M. ter Hofstede. Achieving a General, Formal and Decidable Approach to the OR-join in Workflow using Reset nets. In G. Ciardo and P. Darondeau, editors, *Applications and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 423-443. Springer-Verlag, Berlin, 2005.
- 14 YAWL Editor 2.0 User Manual, 2008, YAWL Foundation
- 15 J. Estublier and R. Casallas. The Adele Software Configuration Manager. In *Configuration Management*, pages 99-139. John Wiley & Sons, 1994.
- 16 M. Svahnberg, J. van Gurp, and J. Bosch. A Taxonomy of Variability Realization Techniques. *Software Practice & Experience*, 35:705-754, 2005
- 17 M. La Rosa, Wil M.P. van der Aalst, M. Dumas and A. ter Hofstede. Questionnaire-based Variability Modeling for System Configuration. *Software and Systems Modeling*, 2008.
- 18 M. La Rosa, M. Dumas, A. H. M. ter Hofstede. Modelling Business Process Variability for Design-Time Configuration. In J. Cardoso, W. M. P. van der Aalst (editors), *Handbook of Research on Business Process Modeling*, IDEA Group – Information Science Reference, 2008.
- 19 <http://www.processconfiguration.com/tools.html>, februari 2009
- 20 L. Algermissen, P. Delfmann, and B. Niehaves. Experiences in Process-oriented Reorganisation through Reference Modelling in Public Administrations – The Case Study Regio@KomM. In *Proceedings of the 13th European Conference on Information Systems (ECIS)*, Regensburg, 2005.

- 21 O. Thomas, B. Hermes, and P. Loos. Towards a Reference Process Model for Event Management. In *Business Process Management Workshops*, volume 4928 of LNCS, pages 443-454, 2008.
- 22 Rafael Prikladnicki, Jorge Luis Nicolas Audy, Roberto Evaristo. A Reference Model for Global Software Development: Findings from a Case Study. *Global Software Engineering*, 2006. ICGSE apos;06. International Conference on Volume , Issue , Oct. 2006 Page(s):18 – 28 Digital Object Identifier 0.1109/ICGSE.2006.261212
- 23 A.-W. Scheer. *Business Process Engineering, Reference Models for Industrial Enterprises*. Springer-Verlag, Berlin, 1994.
- 24 S. Seidel, M. Rosemann, A.H.M. ter Hofstede, and L. Bradford. Developing Business Process Reference Models for the Screen Business – A Design Science Research Case Study. In *17th ACIS*, Adelaide, 2006.
- 25 F. Gottschalk, W.M.P. van der Aalst, M.H. Jansen-Vullers, "Configurable Process Models: A Foundational Approach", *Reference Modeling. Efficient Information Systems Design Through Reuse of Information Models*, July 2007, pp. 59--78.
- 26 Marcello La Rosa, Marlon Dumas, and A.H.M. ter Hofstede. Modelling Business Process Variability for Design-Time Configuration. Accepted for Jorge Cardoso and Wil van der Aalst (Editors), *Handbook of Research on Business Process Modeling*, IGI Global (accepted 26 October 2008).
- 27 27 F. Gottschalk, W.M.P. van der Aalst, M.H. Jansen-Vullers, "SAP WebFlow Made Configurable: Unifying Workflow Templates into a Configurable Model", *BPM 2007*, September 2007; *Lecture Notes in Computer Science*, Vol. 4714, pp. 262--270.
- 28 J. Becker, P. Delfmann, A. Dreiling, R. Knackstedt, and D. Kuropka. Configurative Process Modeling - Outlining an Approach to increased Business Process Model Usability. In *Proceedings of the 15th IRMA International Conference*, New Orleans, 2004. Gabler.

-
- 29 J. Becker, P. Delfmann, and R. Knackstedt. Adaptive Reference Modelling: Integrating Configurative and Generic Adaptation Techniques for Information Models. In J. Becker and P. Delfmann, editors, *Reference Modeling. Efficient Information Systems Design Through Reuse of Information Models*, pages 27-58. Springer, 2007.
- 30 R.M. Dijkman. A classification of differences between similar business processes. *Enterprise Distributed Object Computing Conference, 2007. EDOC '07. 11th IEEE International, 2007*, 37-47.
- 31 H. Frank, and J. Eder. Towards an Automatic Integration of Statecharts. In: *ER 1999*, LNCS 1728, Berlin, Germany, 1999, pp. 430-444.
- 32 G. Grossmann, Y. Ren, M. Schrefl, and M. Stumptner. Behavior Based Integration of Composite Business Processes. In: *BPM 2005*, LNCS 3649, Springer-Verlag, Berlin, Germany, 2005, pp. 186-204.
- 33 J. Mendling, and C. Simon. Business Process Design by View Integration. In: *BPM 2006 Workshops*, LNCS 4103, Springer-Verlag, Berlin, Germany, 2006, pp. 55-64.
- 34 G. Preuner, S. Conrad, and M. Schrefl. View Integration of Behavior in Object-Oriented Databases. *Data & Knowledge Engineering* 36(2), 2001, pp. 153-183.
- 35 M. Ehrig, A. Koschmider, A. Oberweis. Measuring Similarity between semantic Business Process Models, *CAN International Conference Proceeding, vol 247*, pp71-80, 2007.
- 36 B.F. van Dongen, R.M. Dijkman, and J. Mendling. Measuring Similarity between Business Process Models. In: *Proceedings of the 20th International Conference on Advanced Information Systems Engineering (CAiSE)*, Montpellier, France, pp. 450-464, 2008.
- 37 W.M.P. van der Aalst and T. Basten. [Inheritance of Workflows: An approach to tackling problems related to change.](#) Computing Science Reports 99/06, Eindhoven University of Technology, Eindhoven, 1999.

- 38 T. Basten and W.M.P. van der Aalst. [Inheritance of Behavior.](#)
Computing Science Report 99/17, Eindhoven University of Technology,
Eindhoven, 1999.
- 39 W.M.P. van der Aalst and T. Basten. [Identifying Commonalities and Differences in Object Life Cycles using Behavioral Inheritance.](#) In J.M. Colom and M. Koutny, editors, *Application and Theory of Petri Nets 2001*, volume 2075 of *Lecture Notes in Computer Science*, pages 32-52. Springer-Verlag, Berlin, 2001.
- 40 M. Rosemann and W.M.P. van der Aalst. A Configurable Reference Modelling Language. *Information Systems*, 2007, 32(1):1-23.
- 41 M. La Rosa, F. Gottschalk, M. Dumas, and W.M.P. van der Aalst. Domain-driven Reference Process Model Configuration. In *Proceedings of the BPM 2007 Workshops*, volume 4928 of *Lecture Notes in Computer Science*, pages 417-430, Berlin Heidelberg, 2008. Springer Verlag.
- 42 E. S. Raymond. The CML2 Language, 2000. <http://catb.org/esr/cml2/cml2-paper.html>.
- 43 K. Pohl, G. Böckle, and F. van der Linden. *Software Product-line Engineering Foundations, Principles and Techniques*. Springer, Berlin, 2005.
- 44 G. Halmans and K. Pohl. Communicating the Variability of a Software-Product Family to Customers. *Journal of Software and Systems Modeling* 2, 1 2003, 15-36.
- 45 L. Bradford and M. Dumas. Getting Started with YAWL. *Technical Paper, YAWL Foundation*, May 2007.
- 46 F. Gottschalk, T.A.C. Wagemakers, M.H. Jansen-Vullers, W.M.P. van der Aalst, M. La Rosa. Configurable Process Models: Experiences from a Municipality Case Study, *BETA Working paper*, 2008.

Glossary

Action

Naming of Protos for a description of a unit of work that may need to be performed as part of a process.

AND-join

All incoming ports must be triggered before the action/ task can be executed.

AND-split

All outgoing ports must be triggered.

Arcs

The depiction of the flow between several objects in a process model by means of an arrow/arc.

Blocking

Disabling the possibility of executing an action and all succeeding actions by blocking a port.

Build time

The period in which the model is created.

Business process model

When a business process is depicted by symbols it becomes a business process model.

Business process

A business process is a set of logically related tasks performed to achieve a defined business outcome.

Combined business process model

Multiple business process models combined into one model where the similar actions are merged.

Combined workflow model

Multiple workflow models combined into one model where the similar actions are merged.

Condition

Naming of YAWL for a state a process is in at a specific moment in time.

Configurable cluster

A combination of tasks that always occurs in the same composition and therefore this composition cannot be changed/ configured. The cluster can be configured by hiding, blocking, or enabling the entire composition.

Configurable process model

Is a process model that can be configured to a desired situation by blocking, hiding, or enabling of parts of the model within borders of the constraints.

Configurable tasks

These tasks are variation points of a configurable process model.

Configurable workflow model

Is a workflow model that can be configured to a desired situation by blocking, hiding, or enabling of parts of the model within borders of the constraints.

Configuration time

The period before run-time in which the model is configured to desired configuration.

Configured model

A configurable process model that is adapted/ configured to the desired configuration.

Deadlock

A deadlock is a state in which the flow cannot continue towards the next state or task and is not the final state.

Hiding

Disabling the possibility of executing an action but still have the possibility to execute all succeeding actions by hiding an input port.

Input port

The point of entering an action/ task by an arc. The input port can be enabled, blocked, or hidden.

Join

The gathering of tokens (or a token) through several incoming arcs into a task/ action.

OR-join

At least one incoming port must be triggered to execute the action/ task.. It can be that only one port is triggered, or all ports, or several ports.

OR-split

At least one outgoing port must be triggered. It can be that only one port is triggered, or all ports, or several ports.

Output port

The point of leaving an action/ task by an arc. The output port can be enabled or blocked but cannot be hidden.

Port

The connection between an arc and an action/ task that can be configured.

Protos

Is an application that supports the creation of business process models.

Run time

The period in which the model is used in an operational environment

Split

Dispense of tokens (or a token) through several outgoing arcs into a task/ action.

State

Naming of Protos for a state a process is in at a specific time

Task

Naming of YAWL for a description of a unit of work that may need to be performed as part of a process.

Token

A token is an indicator at which point the execution of a process is.

Trigger option

One of the possibilities that leads to the execution of an action; OR, XOR, AND.

Trigger

An event that leads to execution of an action.

Unwanted behavior

This is behavior of the model that is not intended.

Variation point

Part of a configurable process model that can be configured during configuration time.

Variation

Differences between process models that can be used to configure a configurable process model.

Workflow model

When a business process model is used to enable the automated execution in a workflow system it is called a workflow model.

Workflow system

A workflow system is concerned with the automation of procedures in which documents, information, or tasks are passed between participants according to a defined set of rules to achieve, or contribute to, an overall business goal.

Workflow

Workflow is a depiction of a sequence of operations, declared as work of a person, work of a simple or complex mechanism, work of a group of persons, work of an organization of staff, or machines.

XOR-join

Only one incoming port can be triggered to execute the action/ task.

XOR-split

Only one outgoing port can be triggered.

YAWL

Yet Another Workflow language: is a language to create workflow models.

Appendix 1 Questionnaire Interface

Until now the technique was discussed how to create the questionnaire by means of questions and facts but the user will have a different interface when he is answering the questions. In Figure app1-1 the interface of the questionnaire is depicted. There are four main frames in the layout of the questionnaire application (Quaestio). In the left top frame (Valid Questions) the questions are displayed that still have to be answered, the right top frame (Question Inspector) displays the answer facts that - can be selected and the right bottom frame (Guidelines) displays a description of the context of the question. The left bottom frame (Answered Questions) displays the questions that are answered but also questions that are not answered. For example the situation depicted in Figure 2-14 where question 2 won't be displayed if $f2$ is *false*, so if $f2$ is *false* question 2 won't be displayed but will be directly moved into the answered questions. In this way the user has an idea what consequences his choice has. In the *Question Inspector* frame the labels mandatory or true by default are also displayed. When all mandatory questions are answered, Quaestio will ask for the remaining questions if the default answers should be filled in automatically or manually wants to answer the remaining questions. This way the user has the choice to shorten the number of questions or to look into every question.

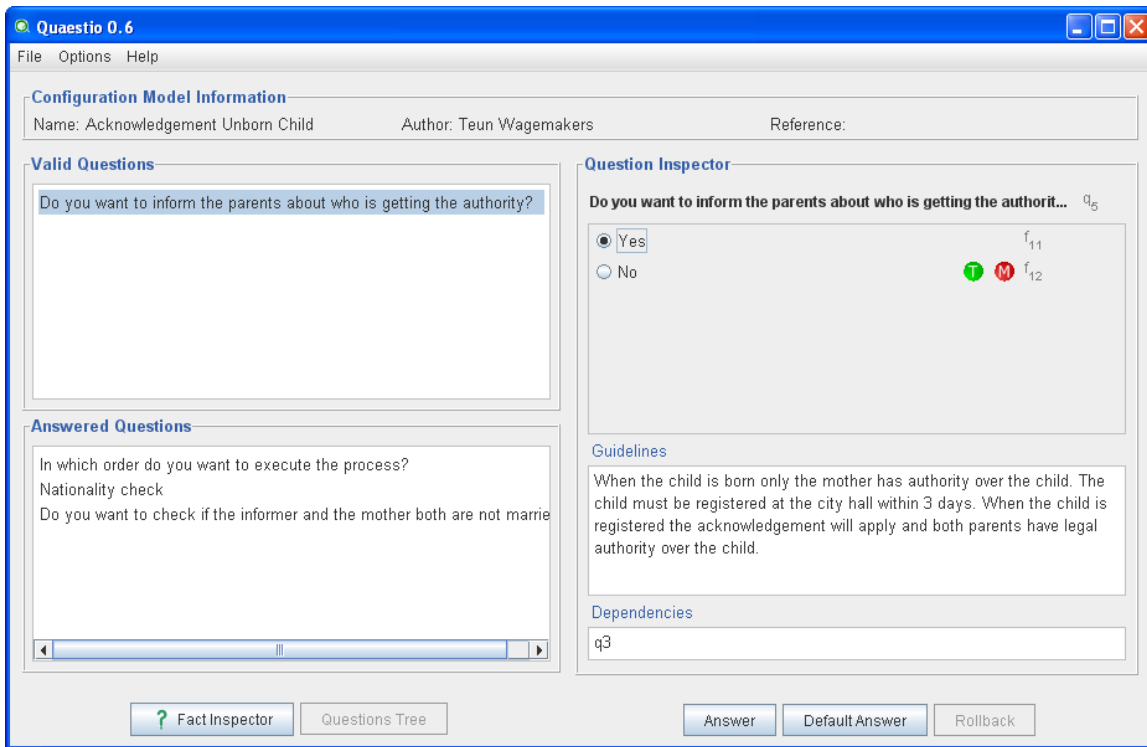


Fig. app1-1 A screenshot of Quaestio

Appendix 2 Using the Configurator and the Individualizer

The xml file that specifies the mapping (Section 3.6.2) is created, we can use it in the *Configurator*. The *Configurator* is depicted in Figure app2-1 and uses three files to configure the configurable workflow model. The first file is the *YAWL xml file of the configurable workflow model*. This file will be adjusted according to the answers the user gave in the questionnaire. The second file is the *mapping*. This is the xml file that we created in section 3.6.2 that specifies how a task should be configured. The last file is *configuration file*. This file is generated by *Quaestio* according to answers of the user. So by using the configuration file the *Configurator* knows which answers are given. The mapping will be consulted to look up which actions are needed for those tasks, and these tasks will then be applied on the configurable workflow model. The result of this is a file that is loaded into the *Individualizer*.

The *Individualizer* is depicted in Figure app2-2. The file that is generated by the *Configurator* will be loaded in to the *Individualizer*. The *Individualizer* will remove all redundant tasks, arcs, and conditions. The result is a YAWL xml file that is configured according to the answers of the user. This file can be loaded in to YAWL engine and is ready for use.

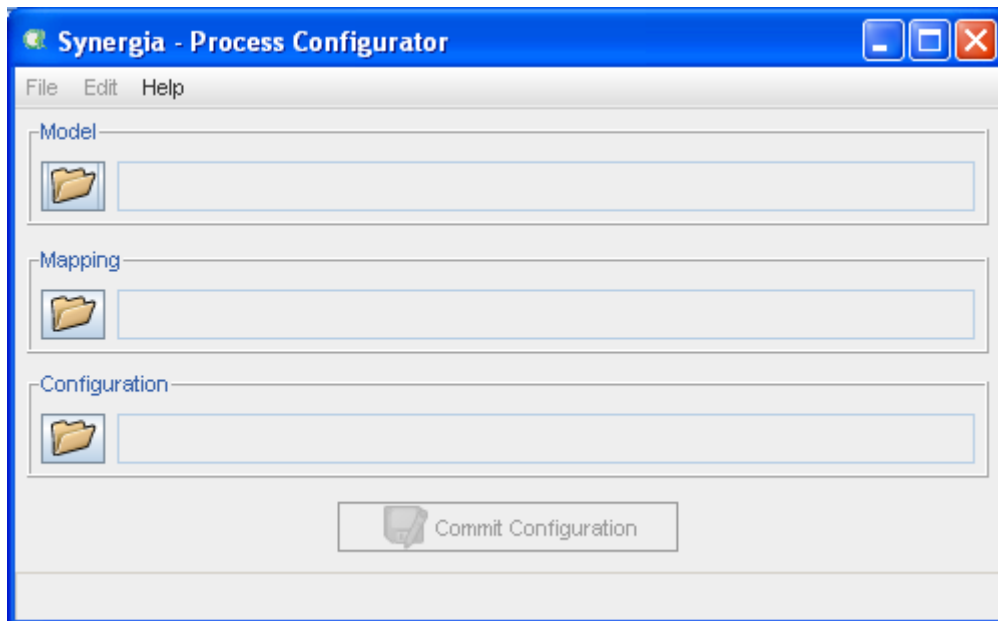


Fig. app2-1 Depiction of the Configurator. This is the application which configurates the configurable models

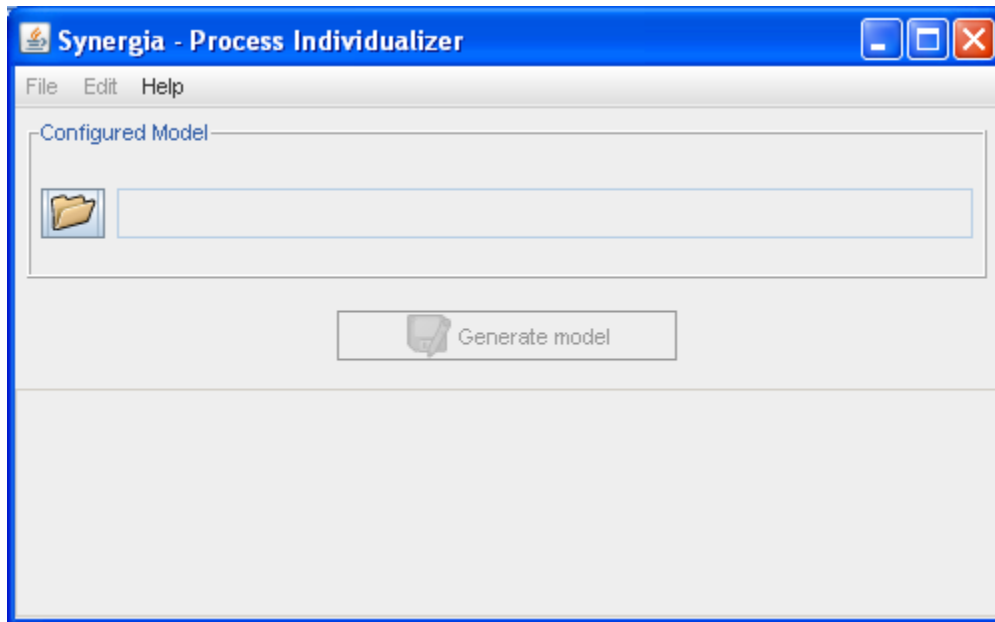
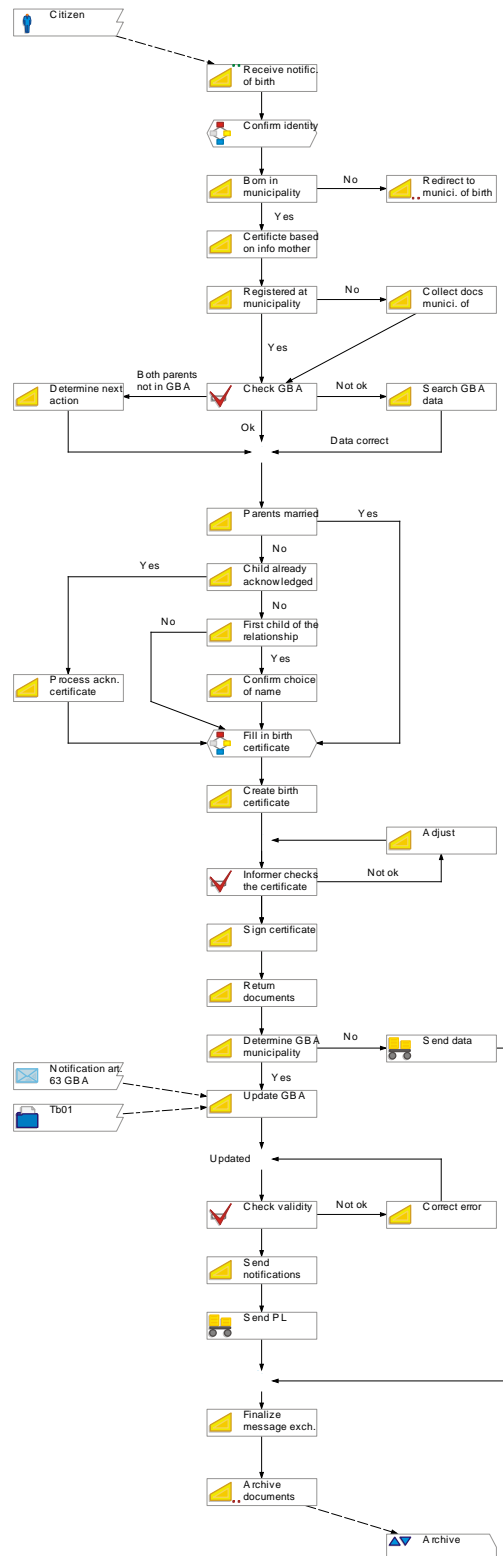


Fig. app2-2 The individualizer removes the redundant configurable process

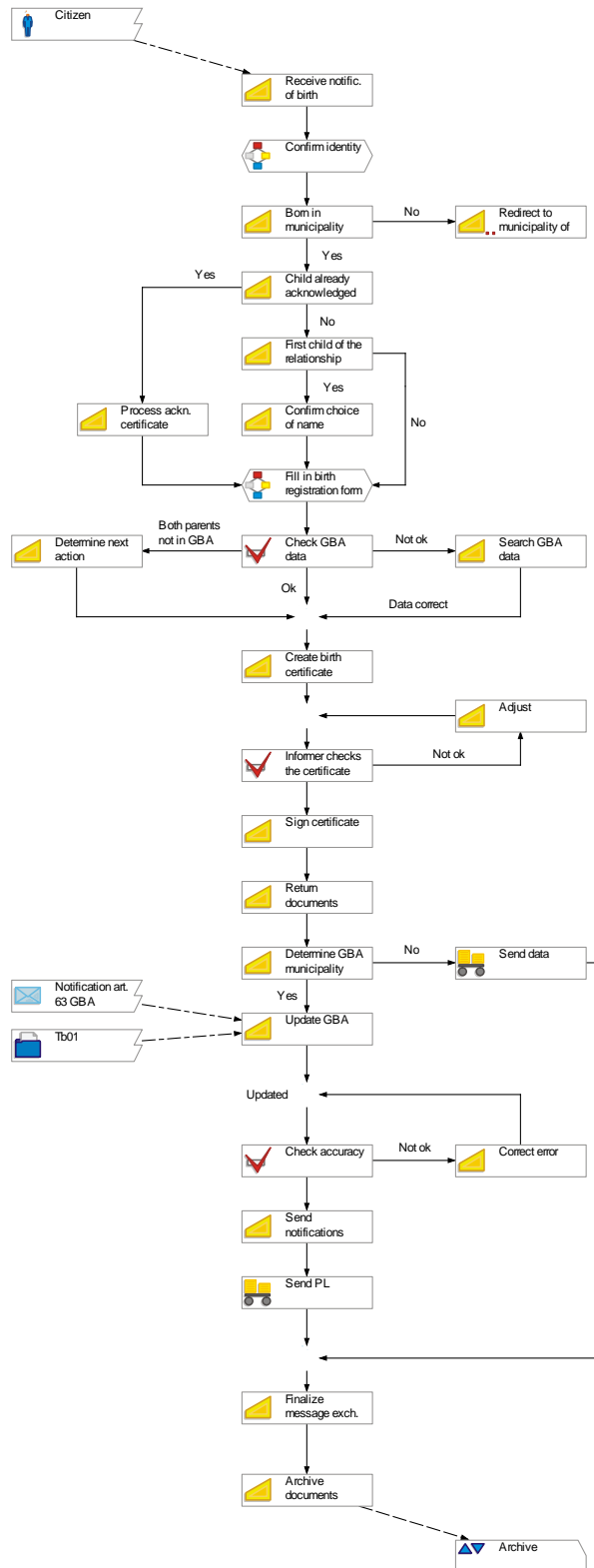
Appendix 3 individual *birth* business process model A

Small municipality

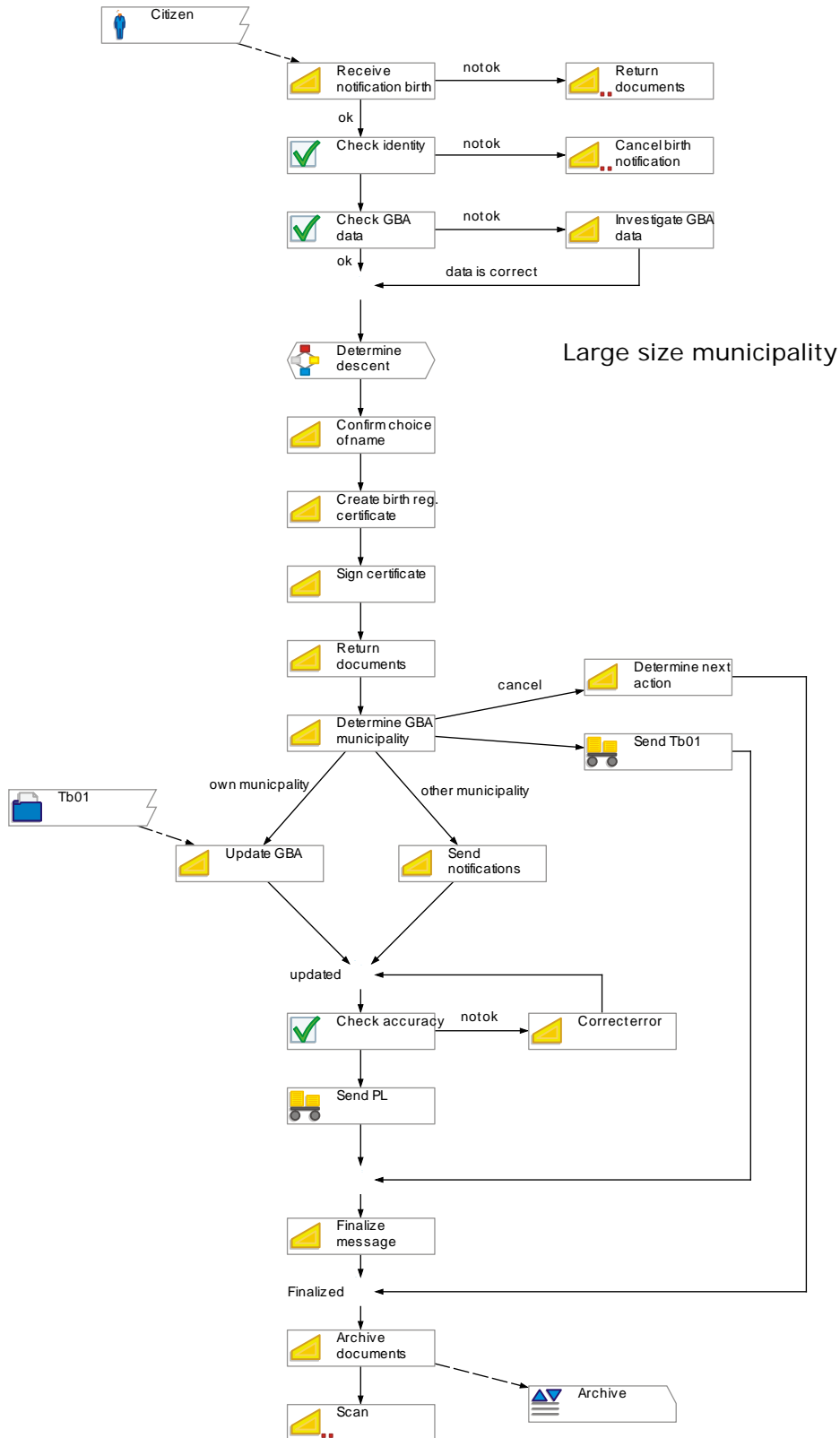


Appendix 4 individual *birth* business process model B

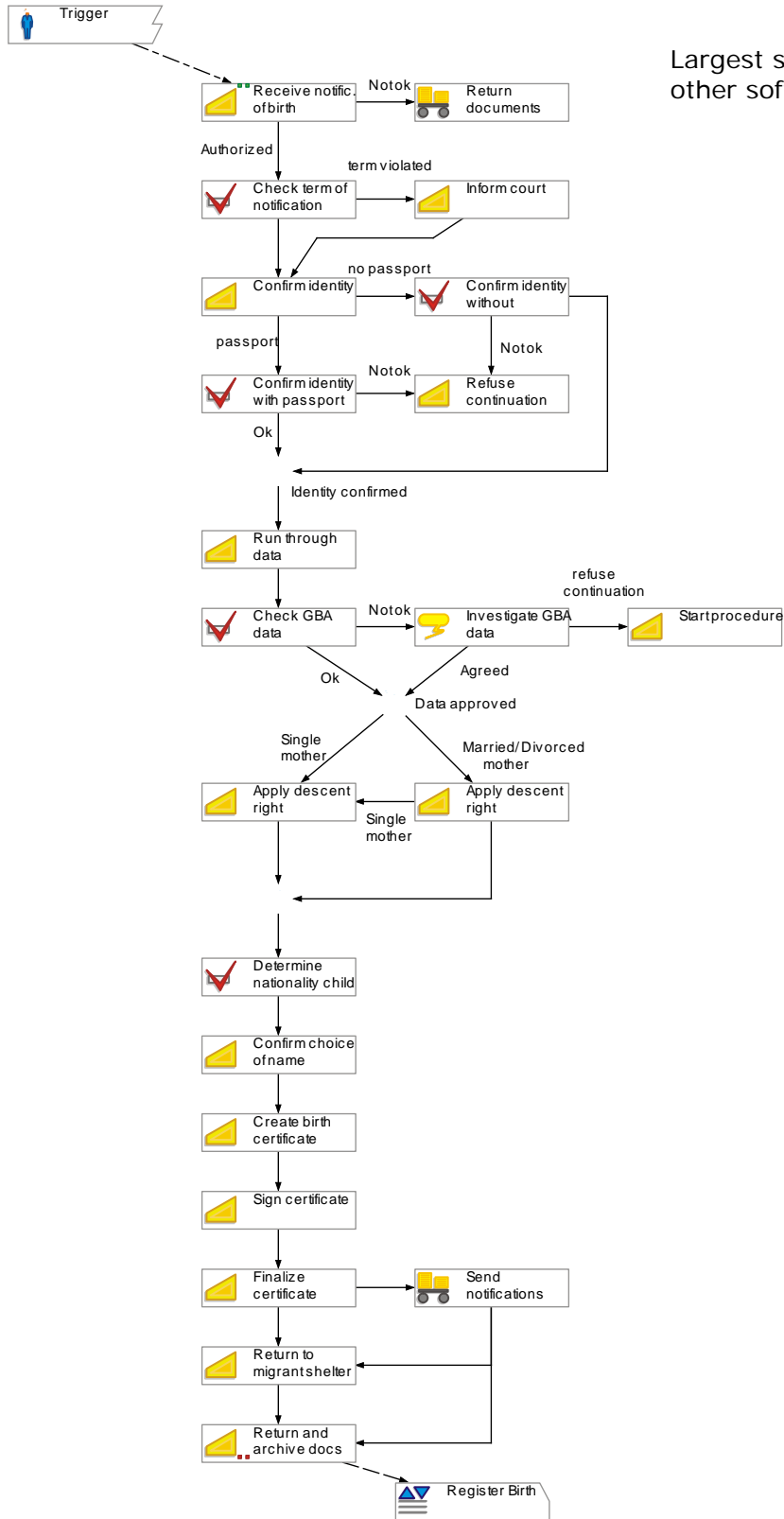
Medium size municipality



Appendix 5 individual *birth* business process model C

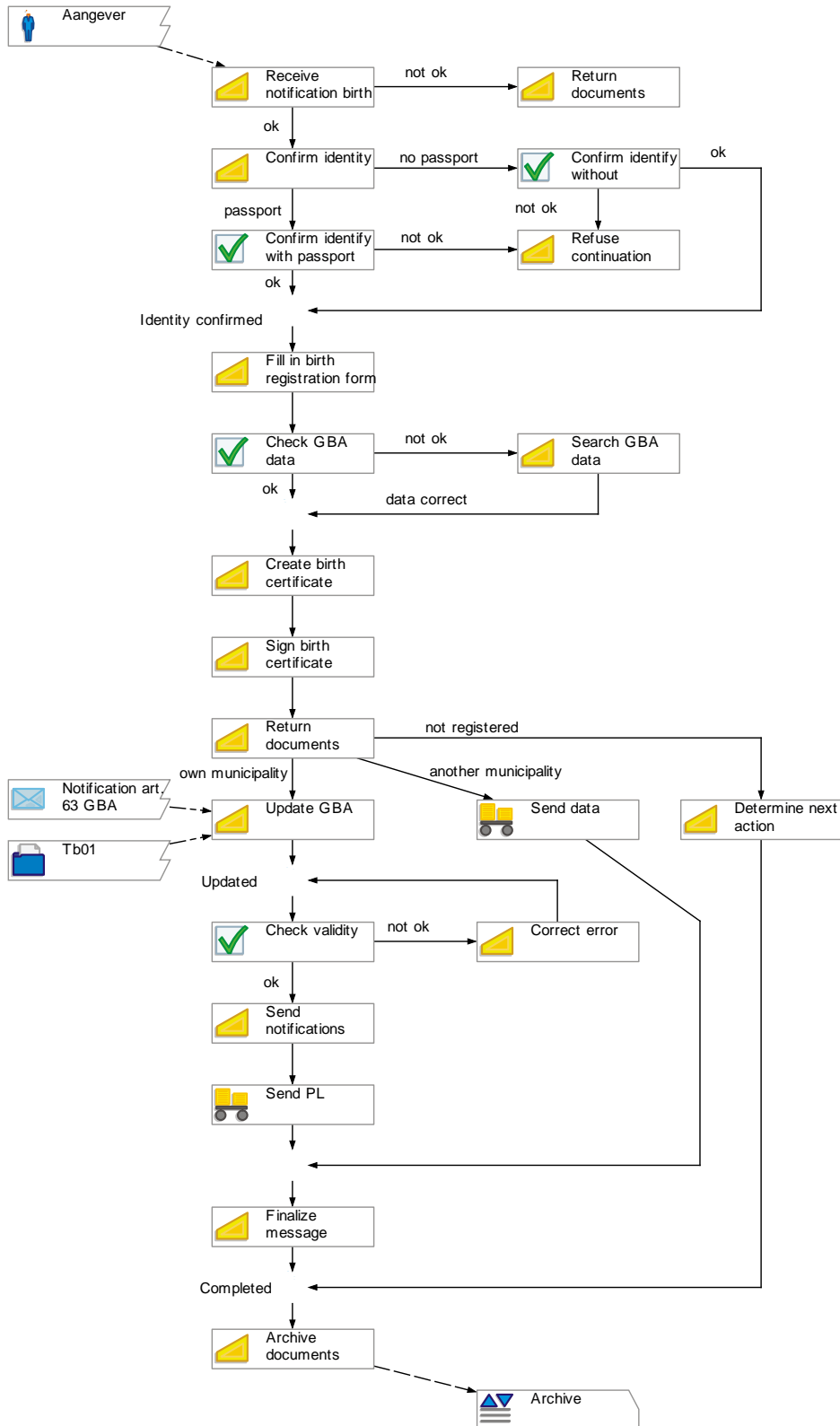


Appendix 6 individual *birth* business process model D



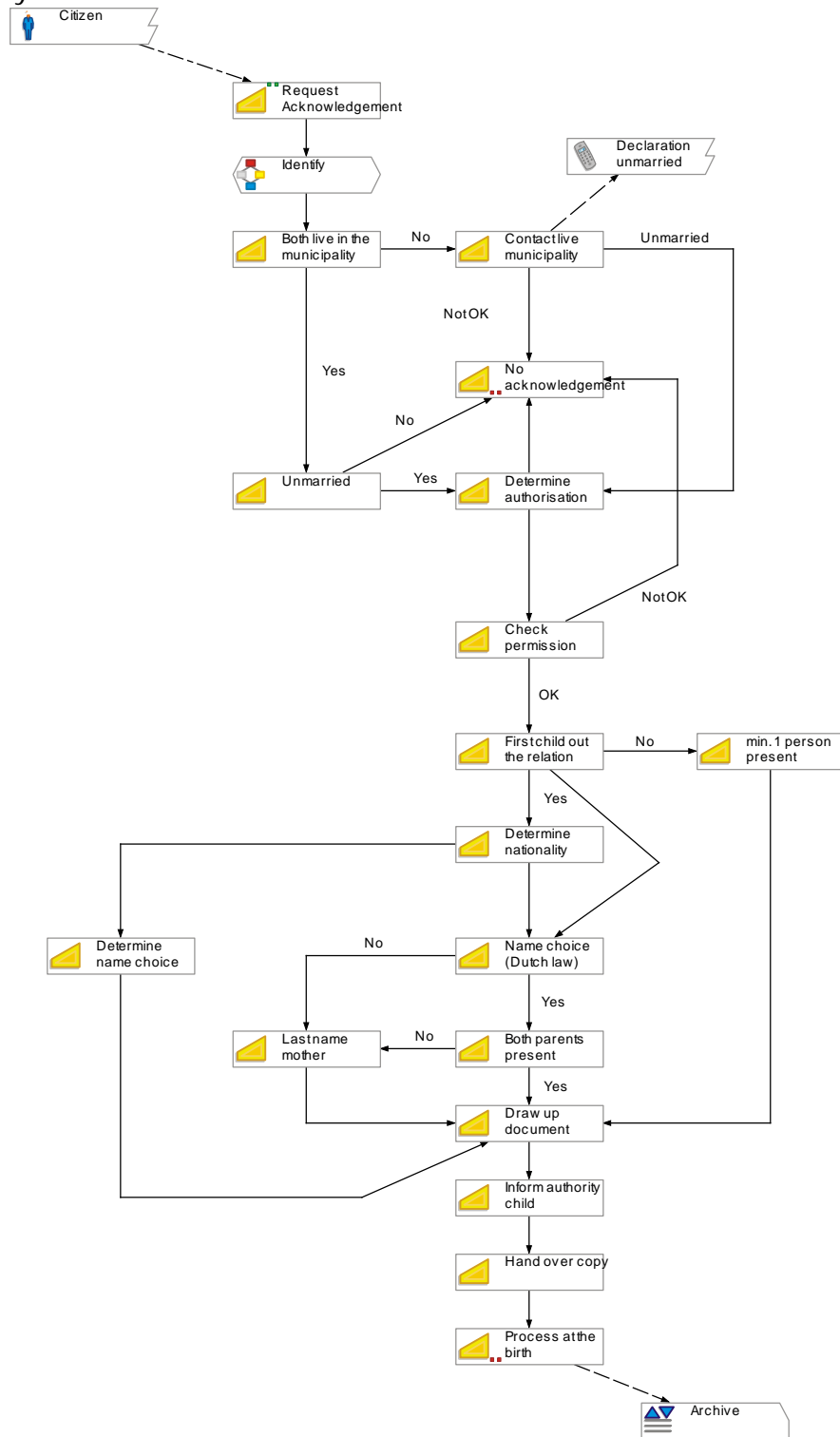
Largest size municipality and other software

Appendix 7 individual *birth* business process reference model



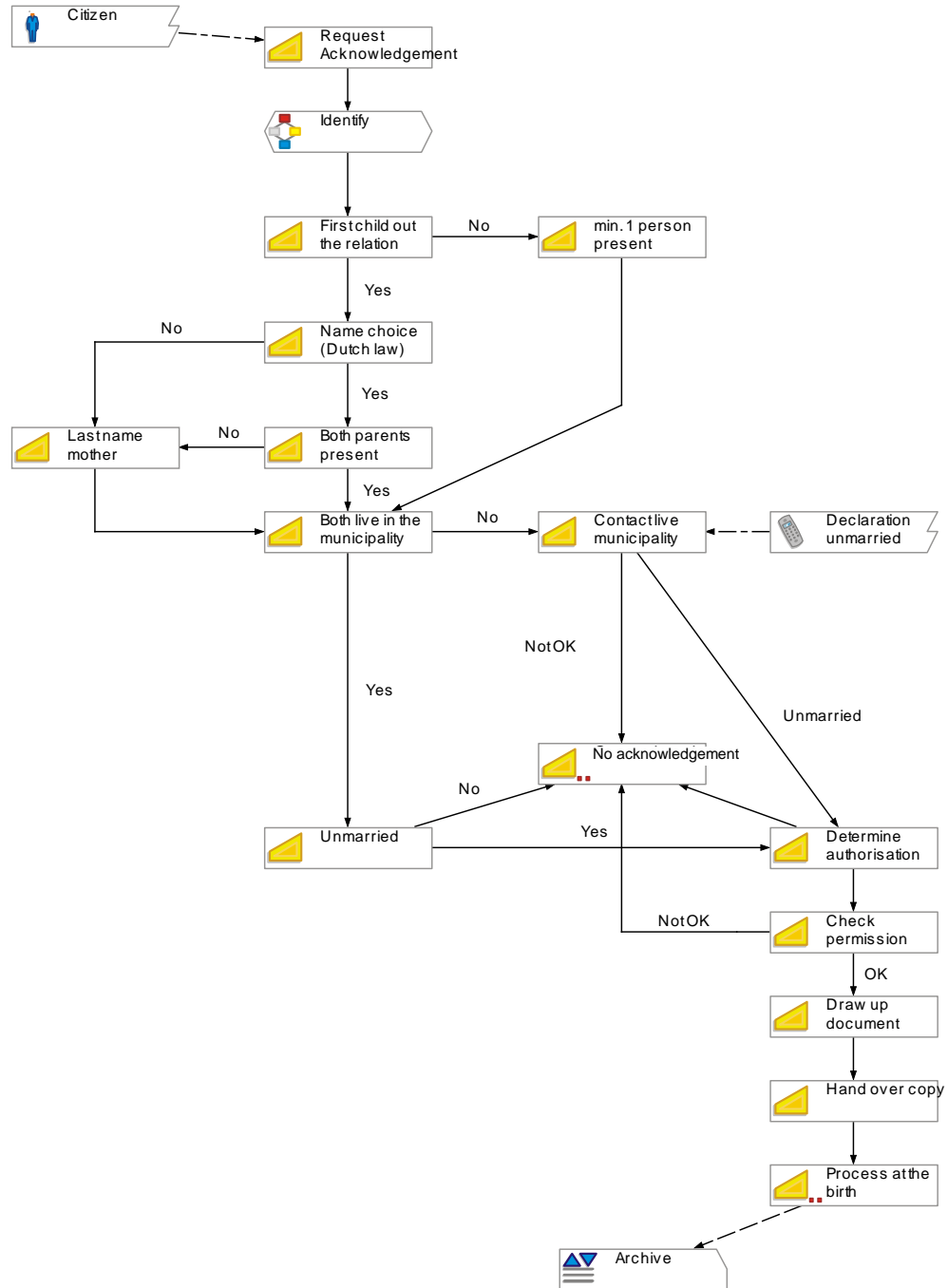
Appendix 8 individual *acknowledgement of an unborn child* business process model A

Small municipality

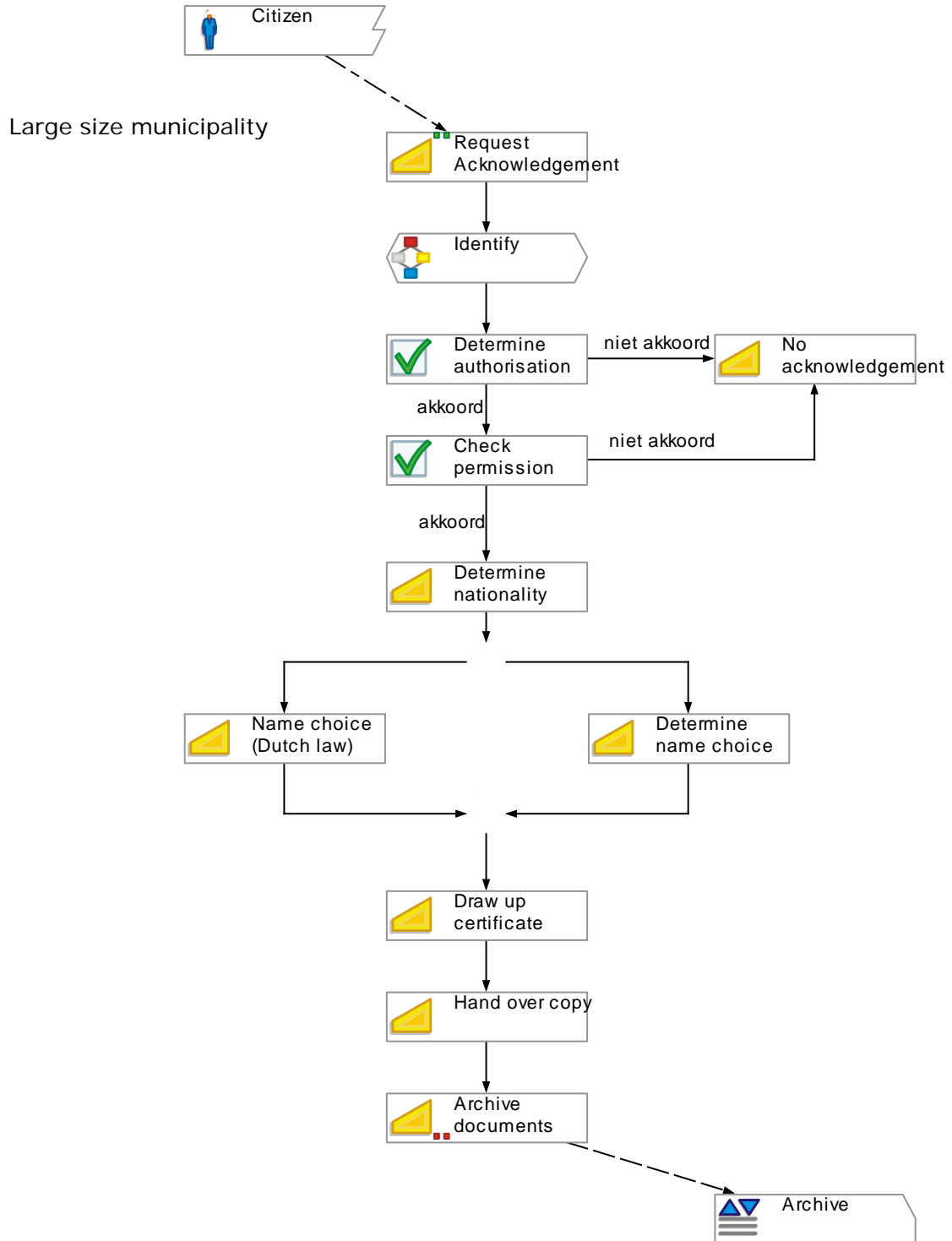


Appendix 9 individual acknowledgement of an unborn child business process model B

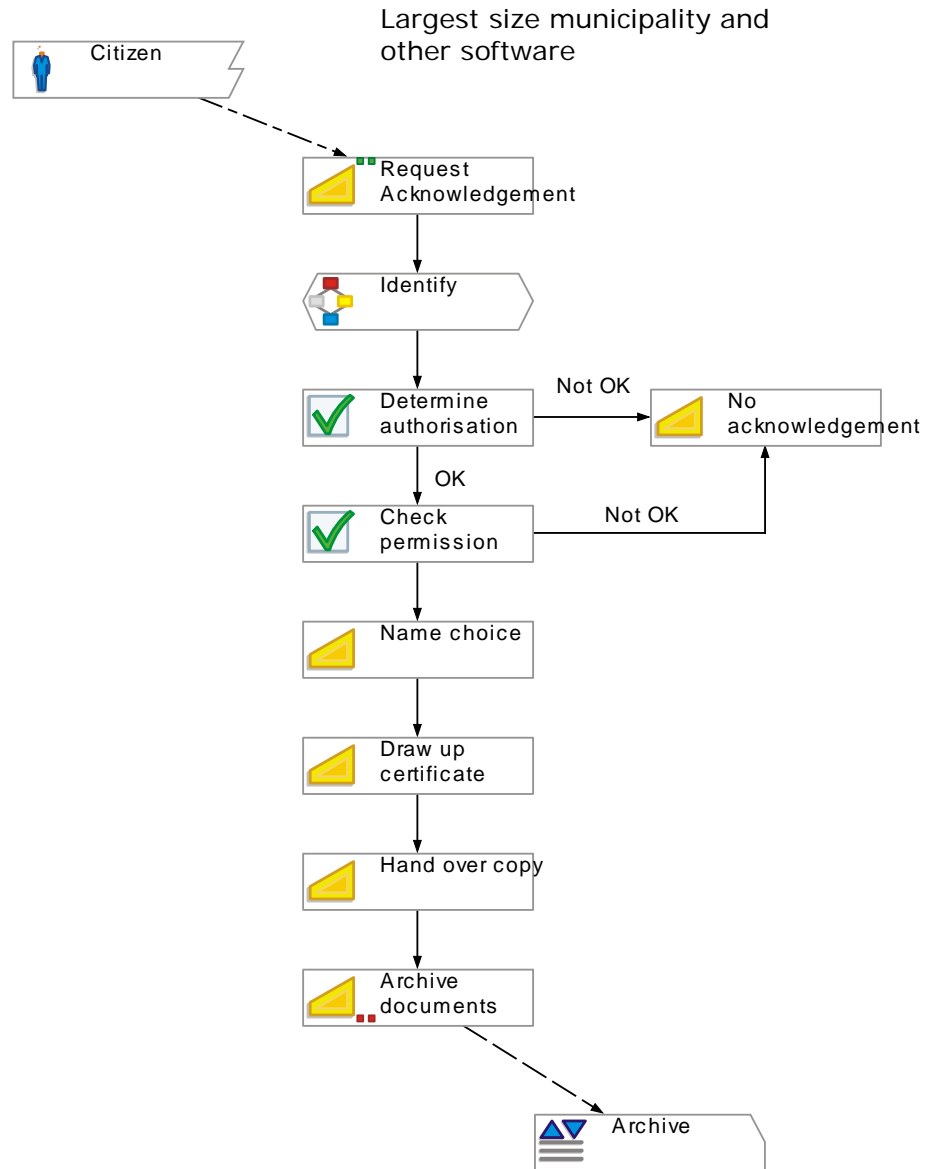
Medium size municipality



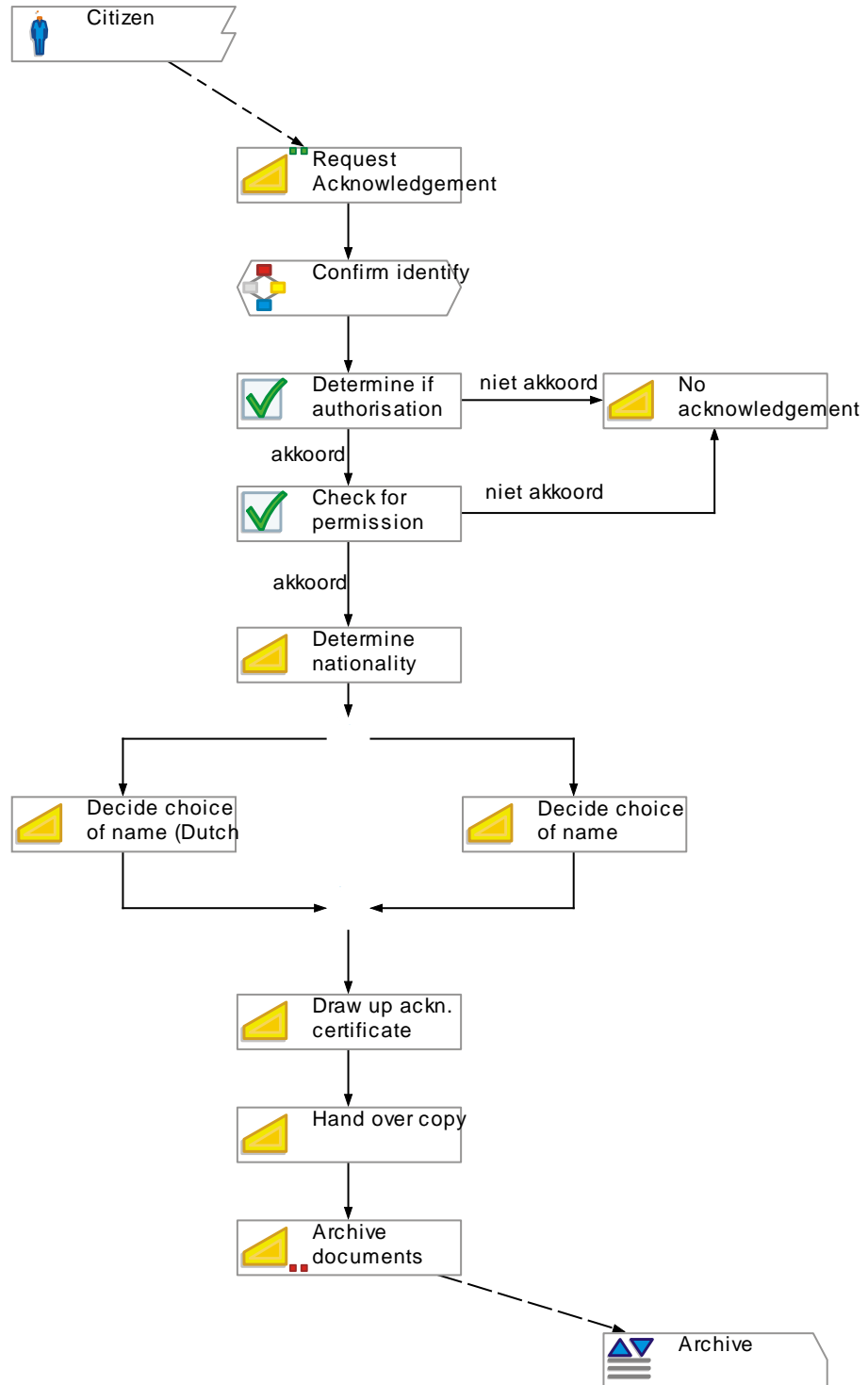
Appendix 10 *individual acknowledgement of an unborn child* business process model C



Appendix 11 *individual acknowledgement of an unborn child* business process model D

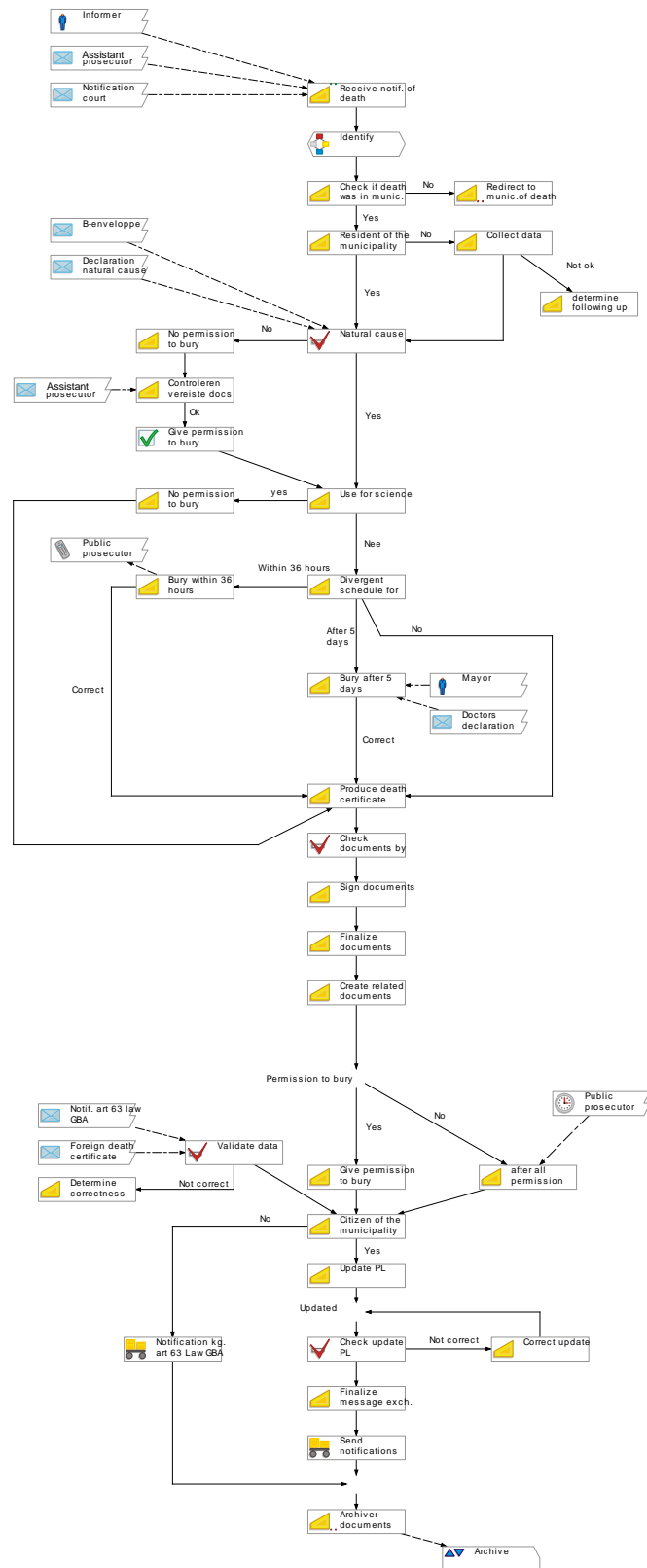


Appendix 12 individual acknowledgement of an unborn child business process reference model



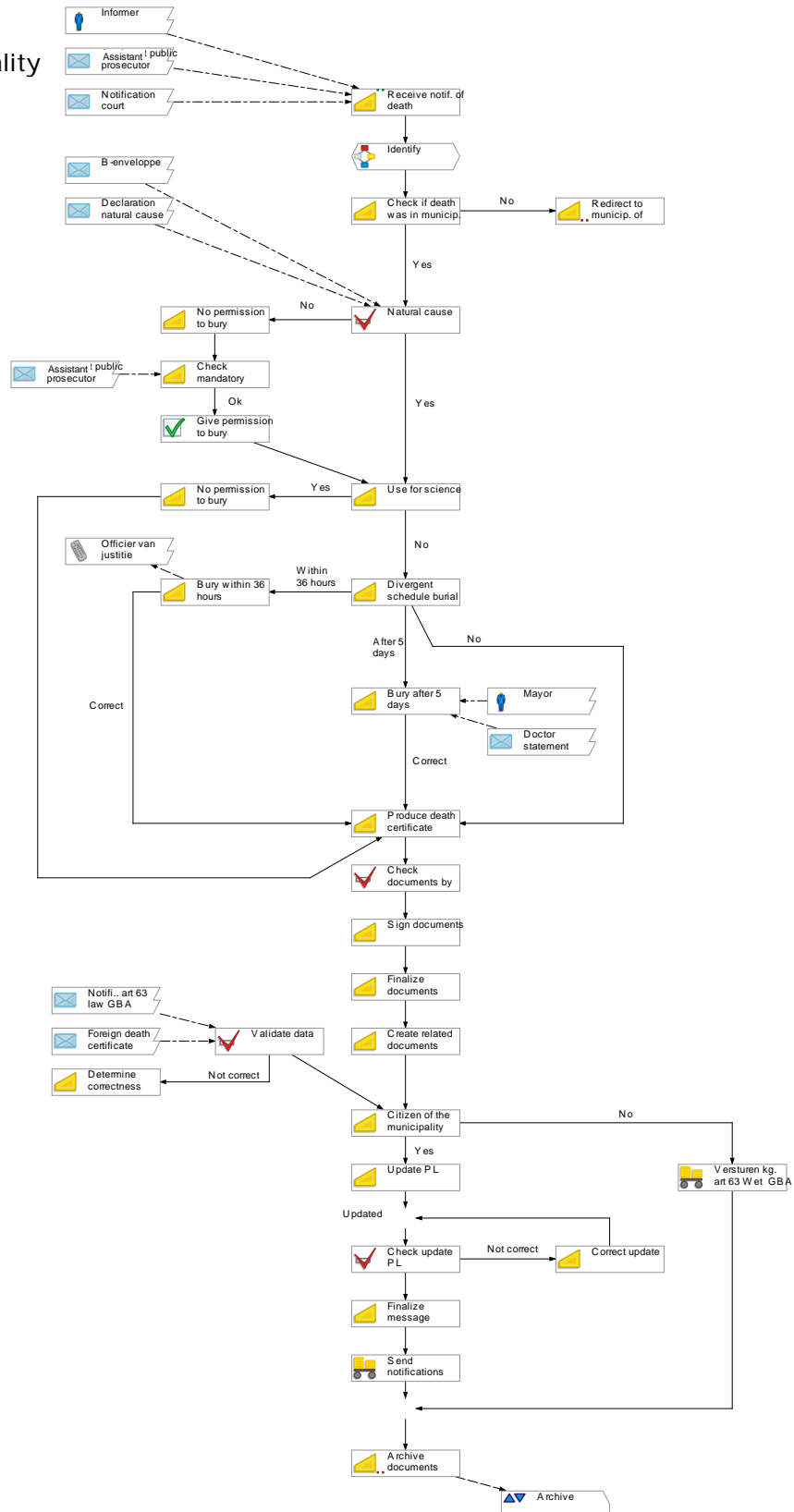
Appendix 13 individual *deceased* business process model A

Small municipality

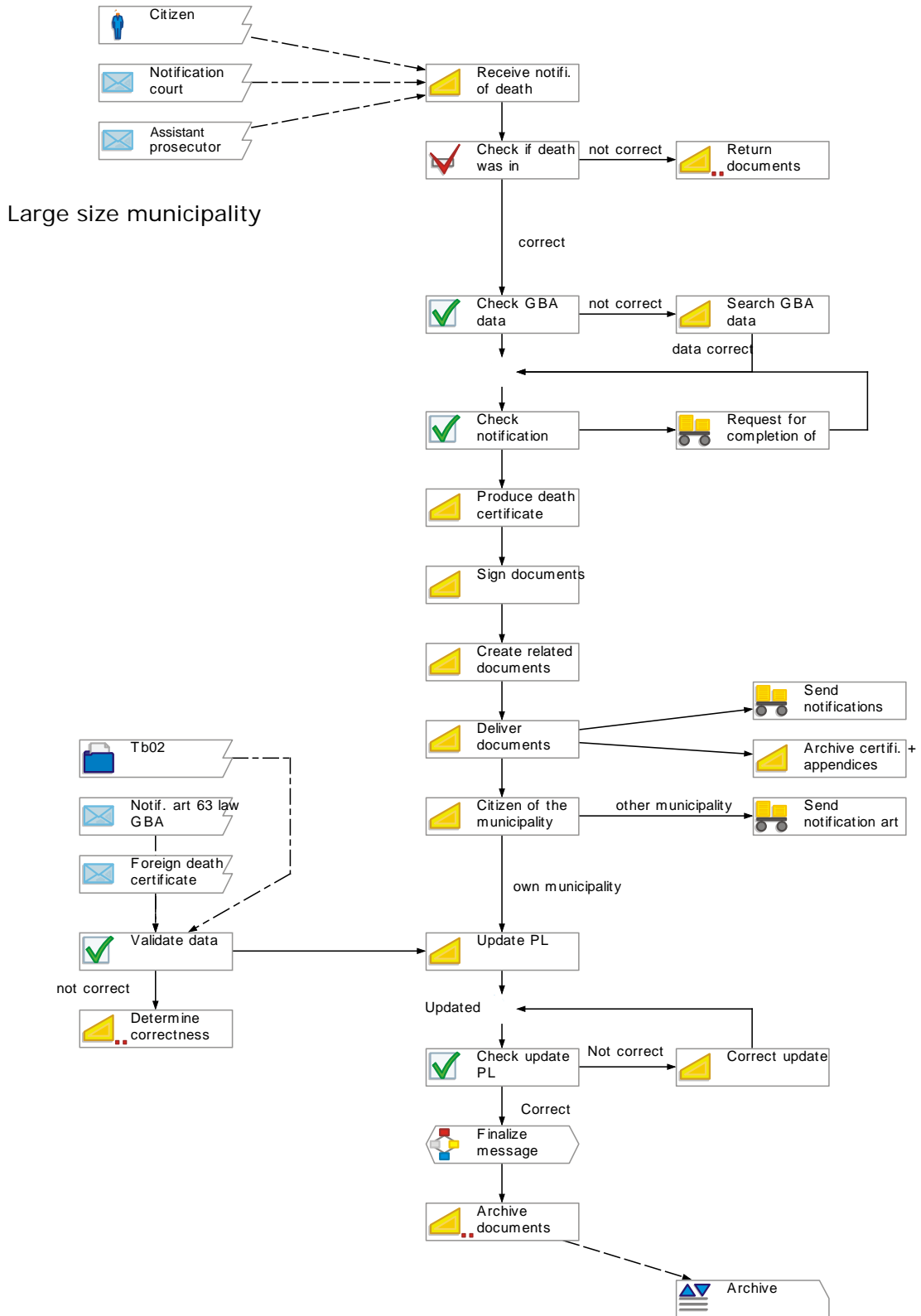


Appendix 14 individual *deceased* business process model B

Medium size municipality

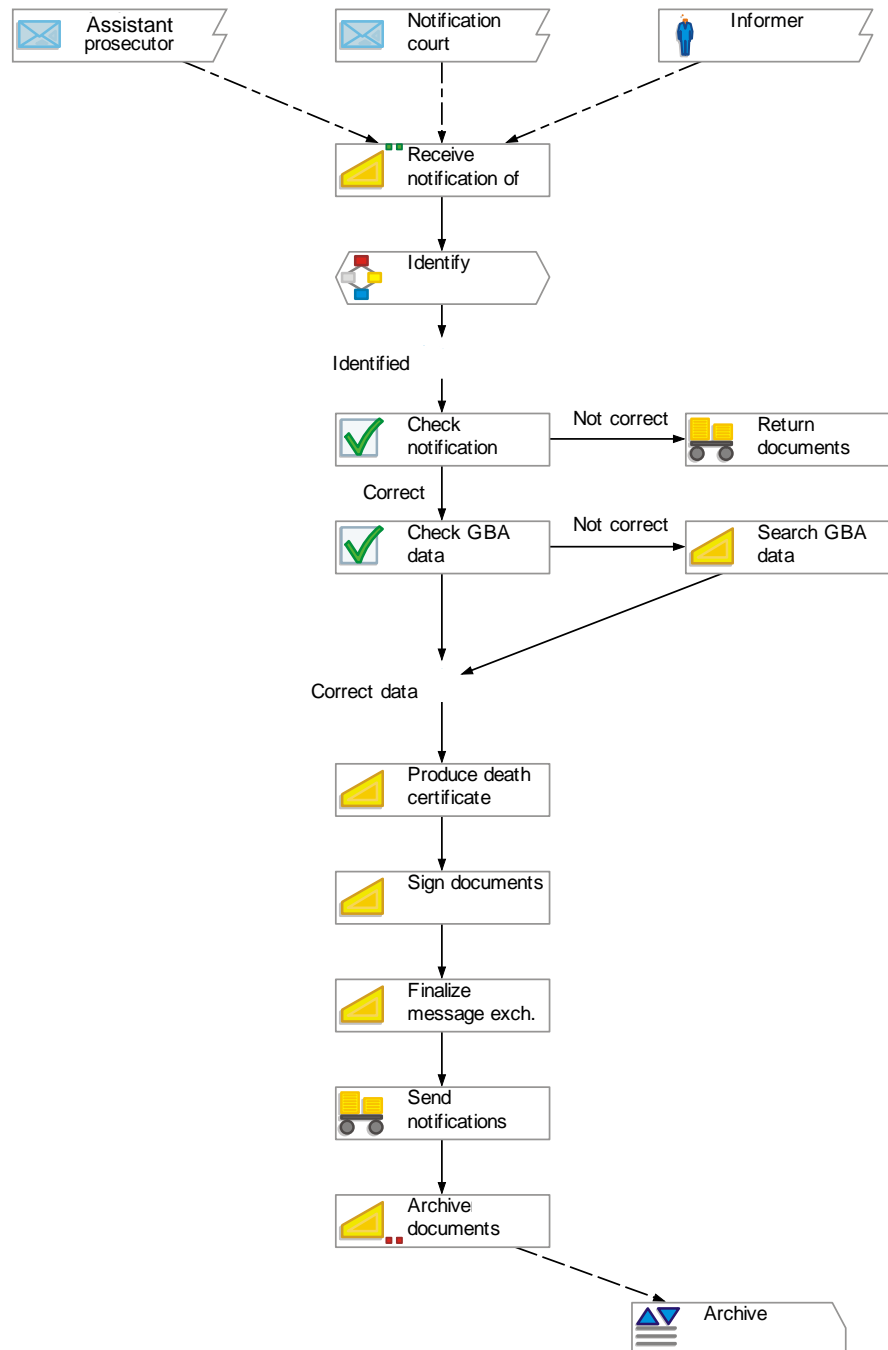


Appendix 15 individual *deceased* business process model C

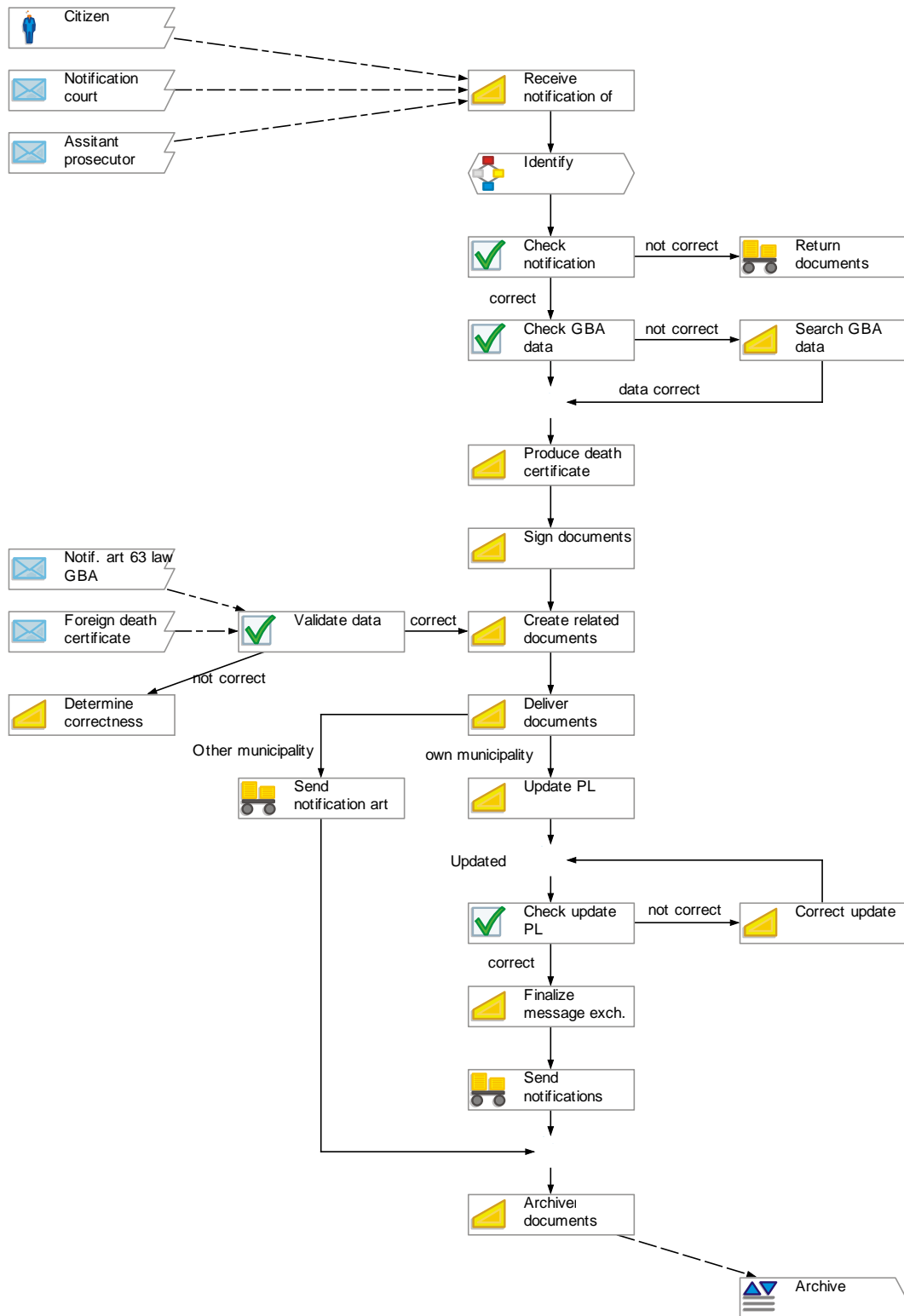


Appendix 16 individual *deceased* business process model D

Largest size municipality and other software

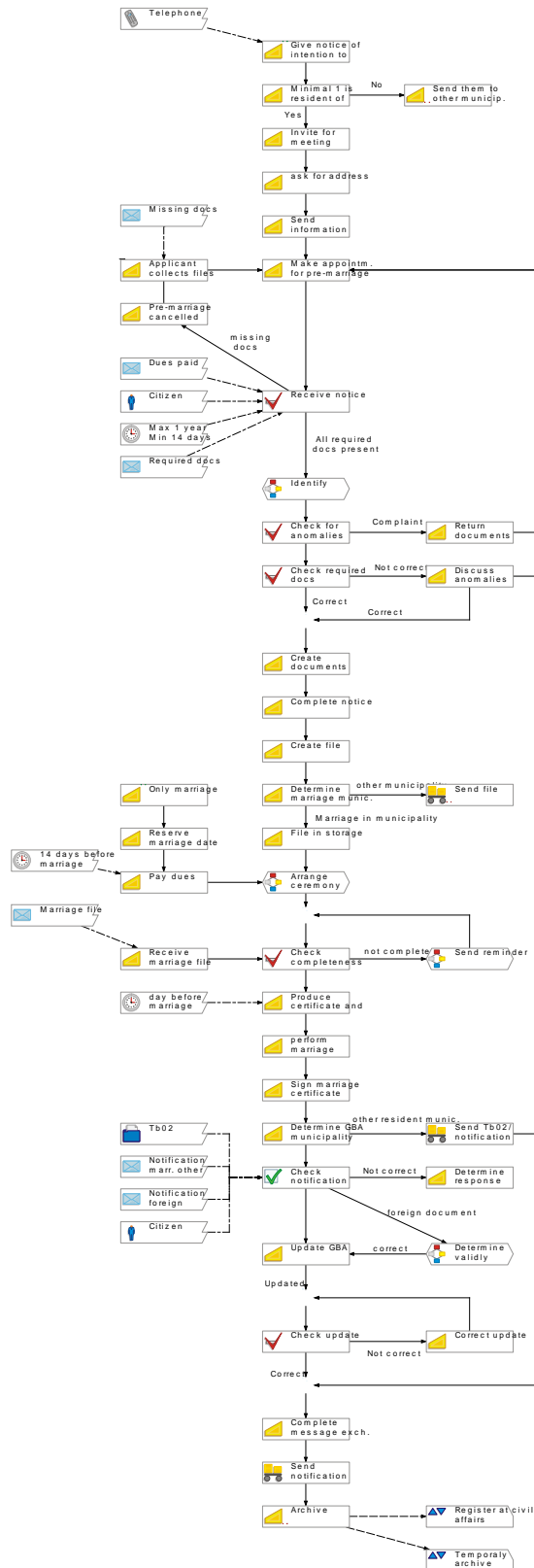


Appendix 17 individual *deceased* business process reference model



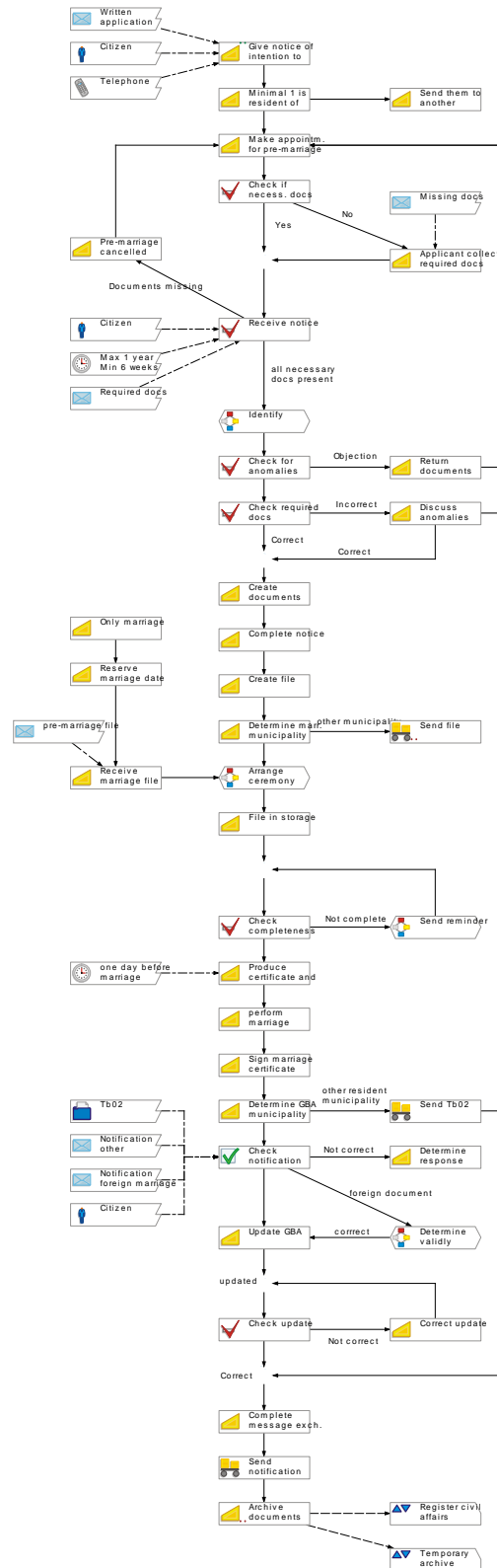
Appendix 18 individual *marriage* business process model A

Small municipality



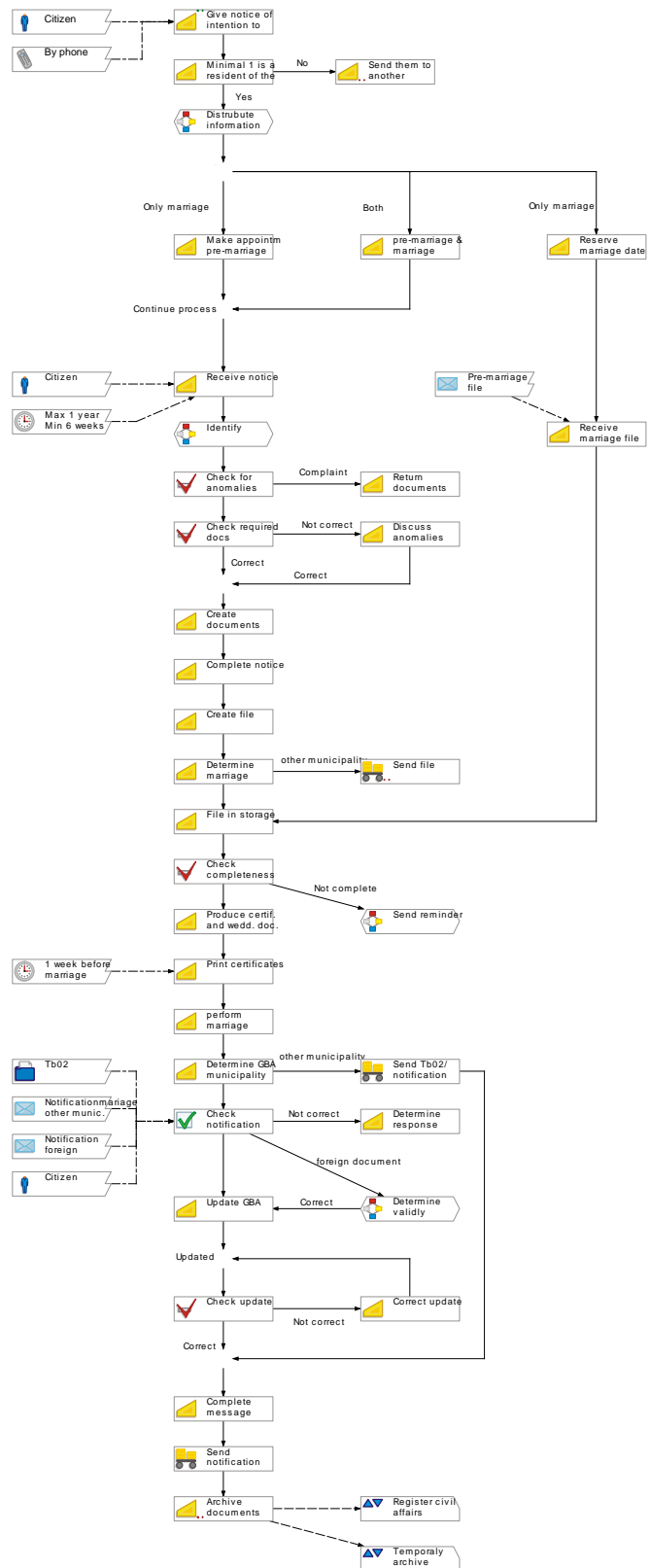
Appendix 19 individual *marriage* business process model B

Medium size municipality



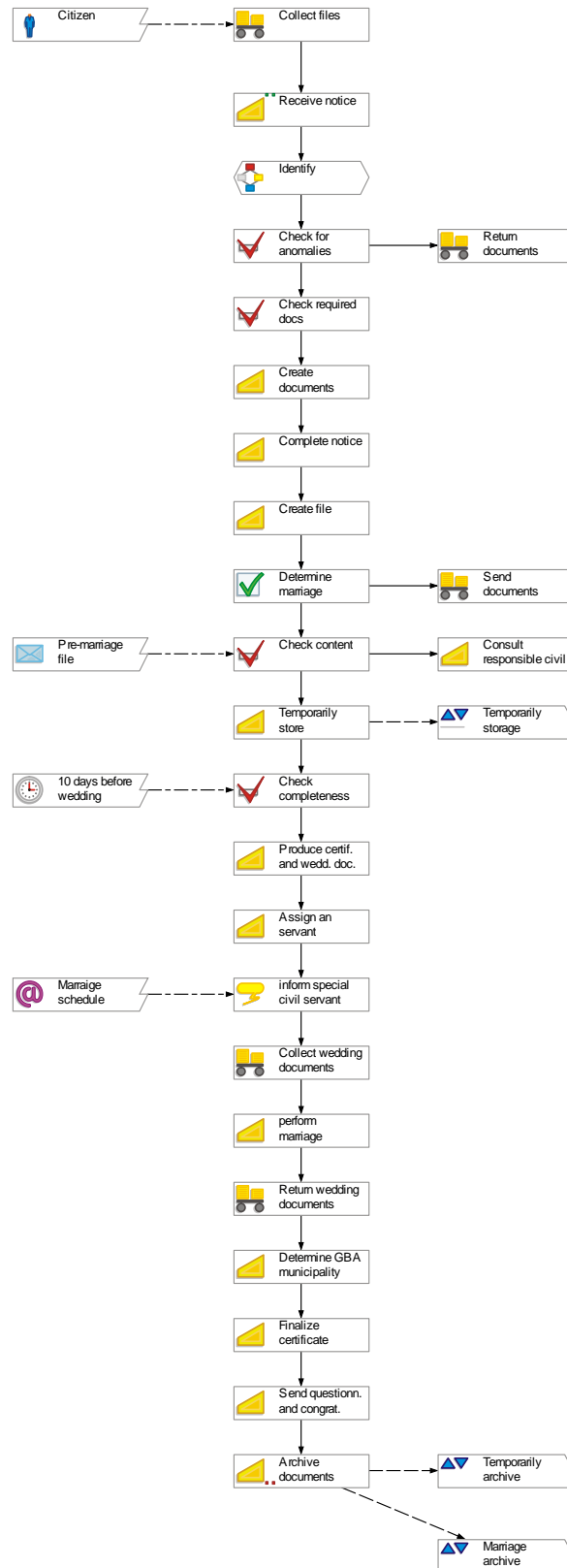
Appendix 20 individual *marriage* business process model C

Large size municipality

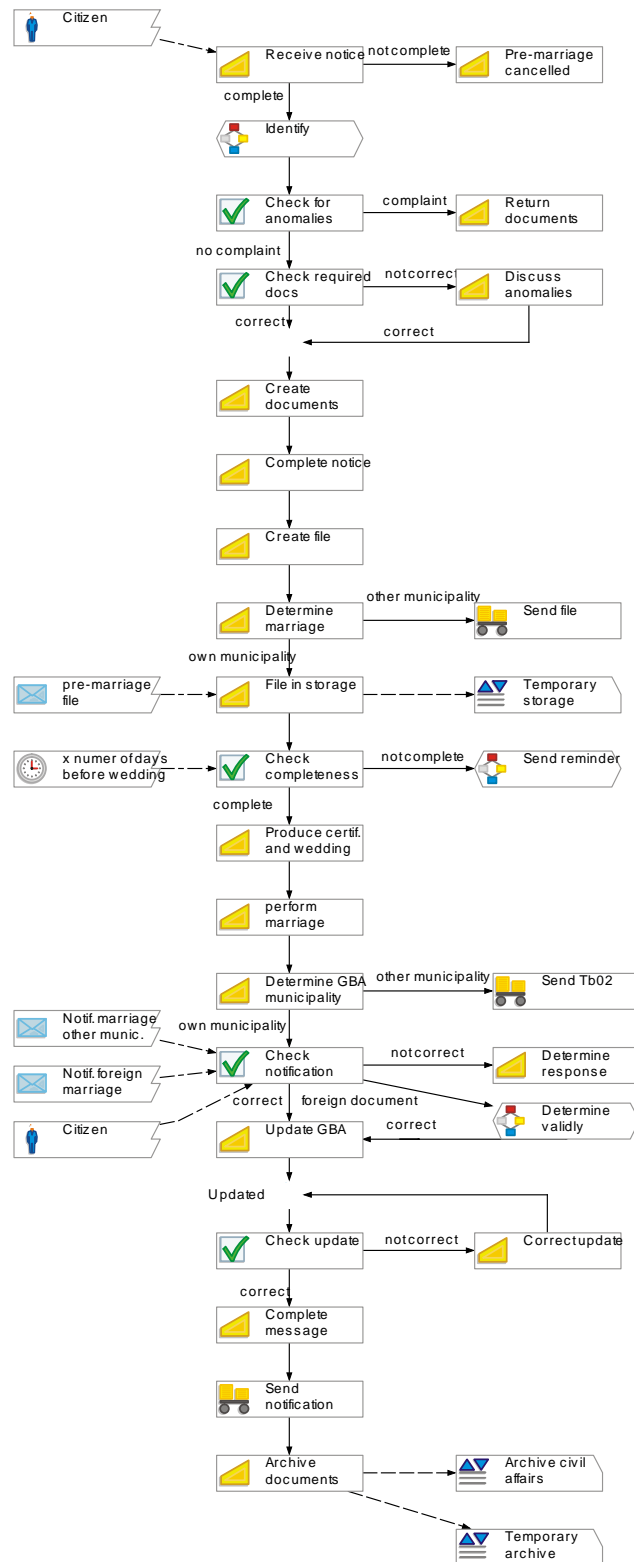


Appendix 21 individual *marriage* business process model D

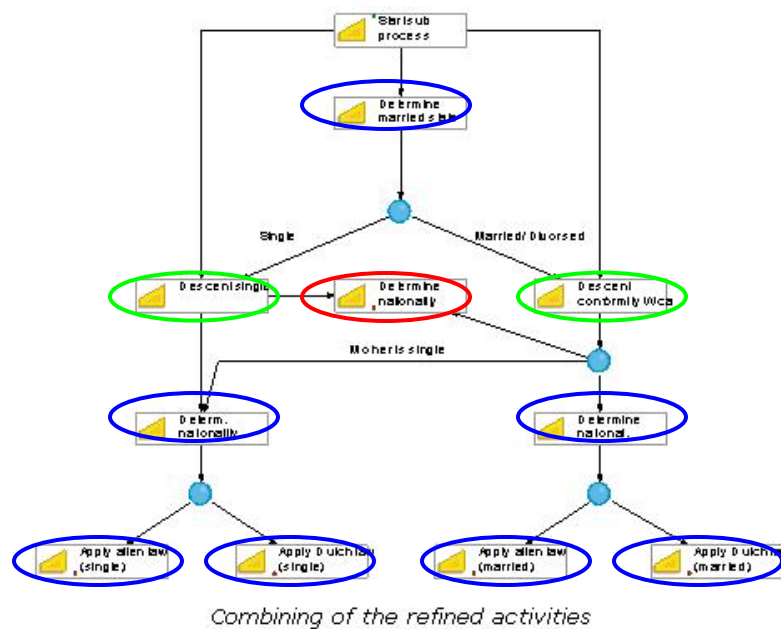
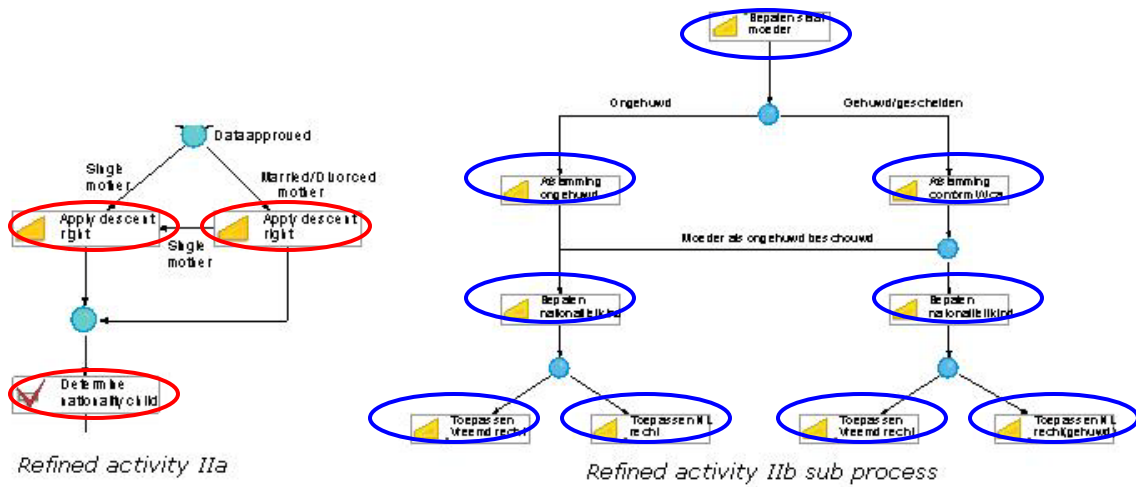
Largest size municipality and other software



Appendix 22 individual *marriage* business process reference model

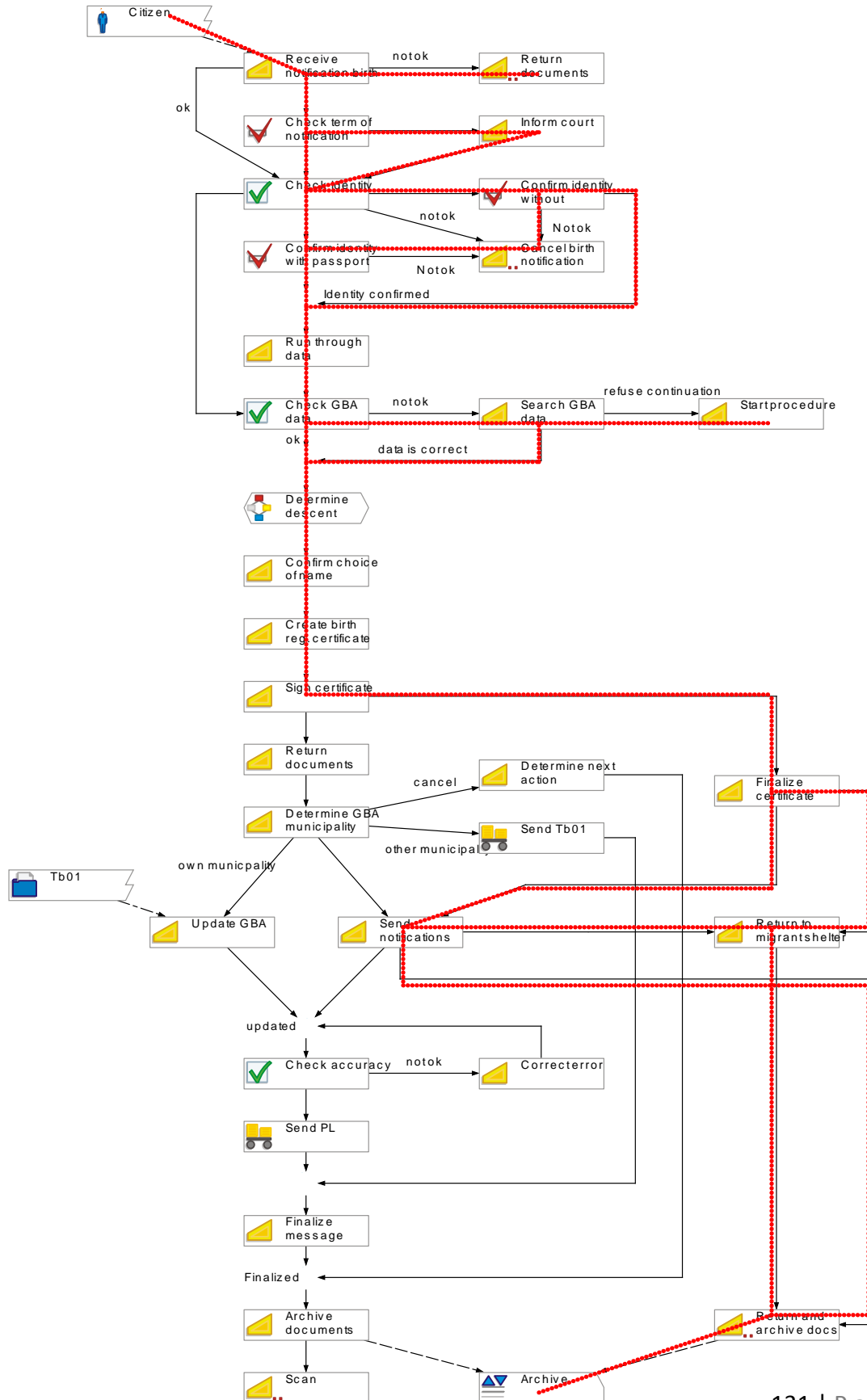


Appendix 23 Combining of a subprocess

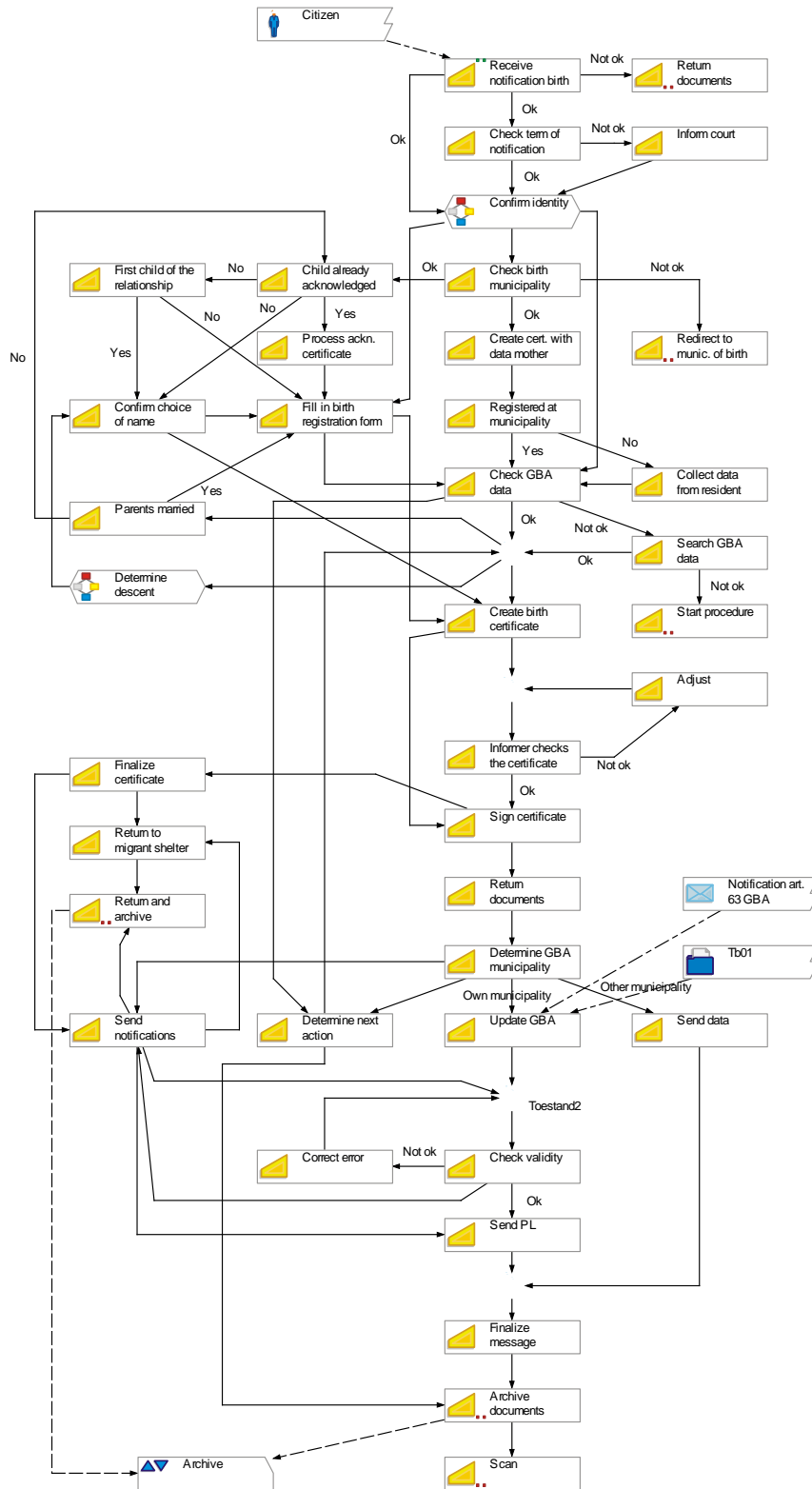


- The red circles are representing the actions of the *refined activity IIa*
- The blue circles are representing the actions of the *refined activity IIb*
- The green circles are representing the actions that both occur in the *refined activity IIa* and *IIb*.

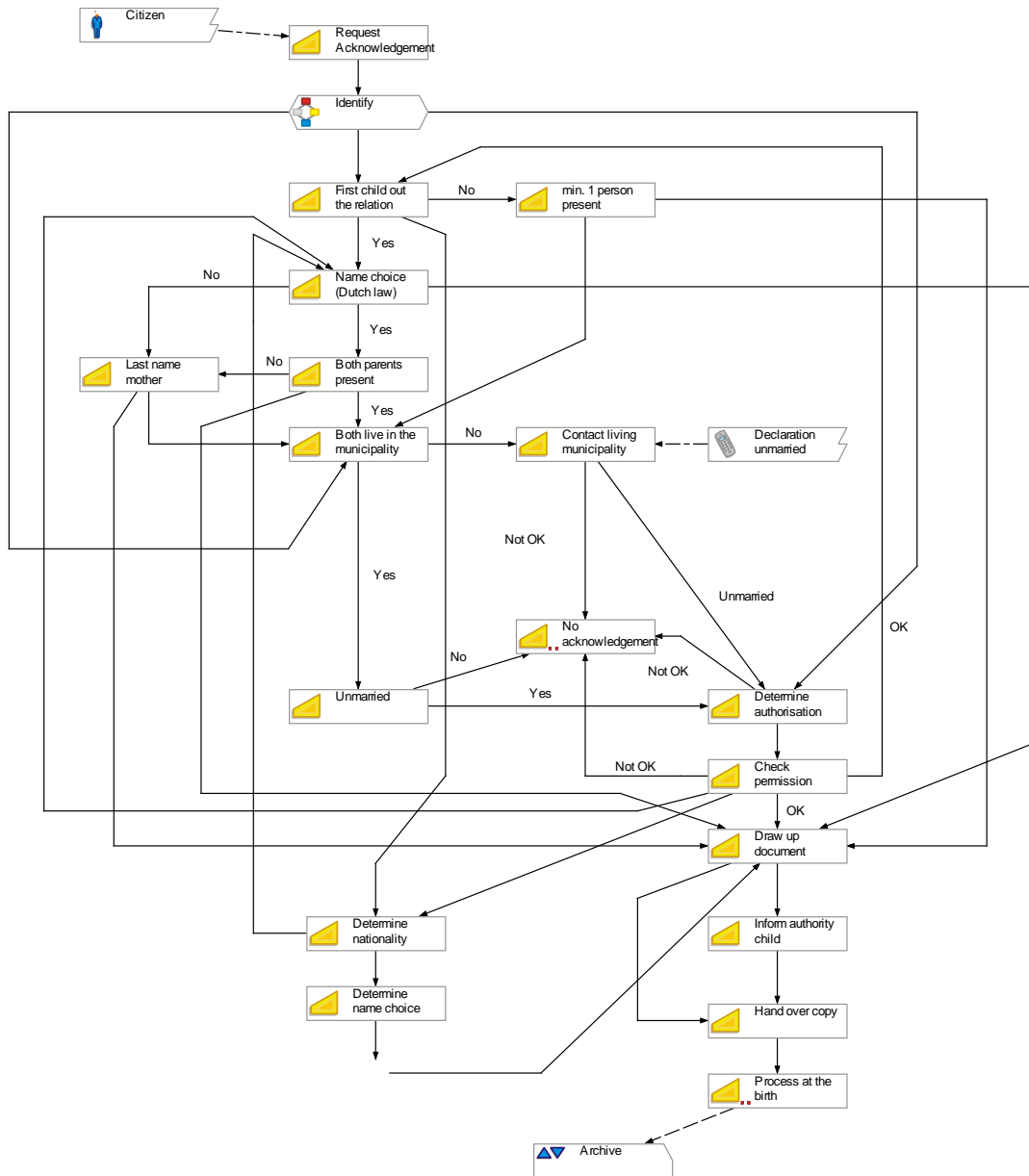
Appendix 24 trace check of the birth model of appendix 4



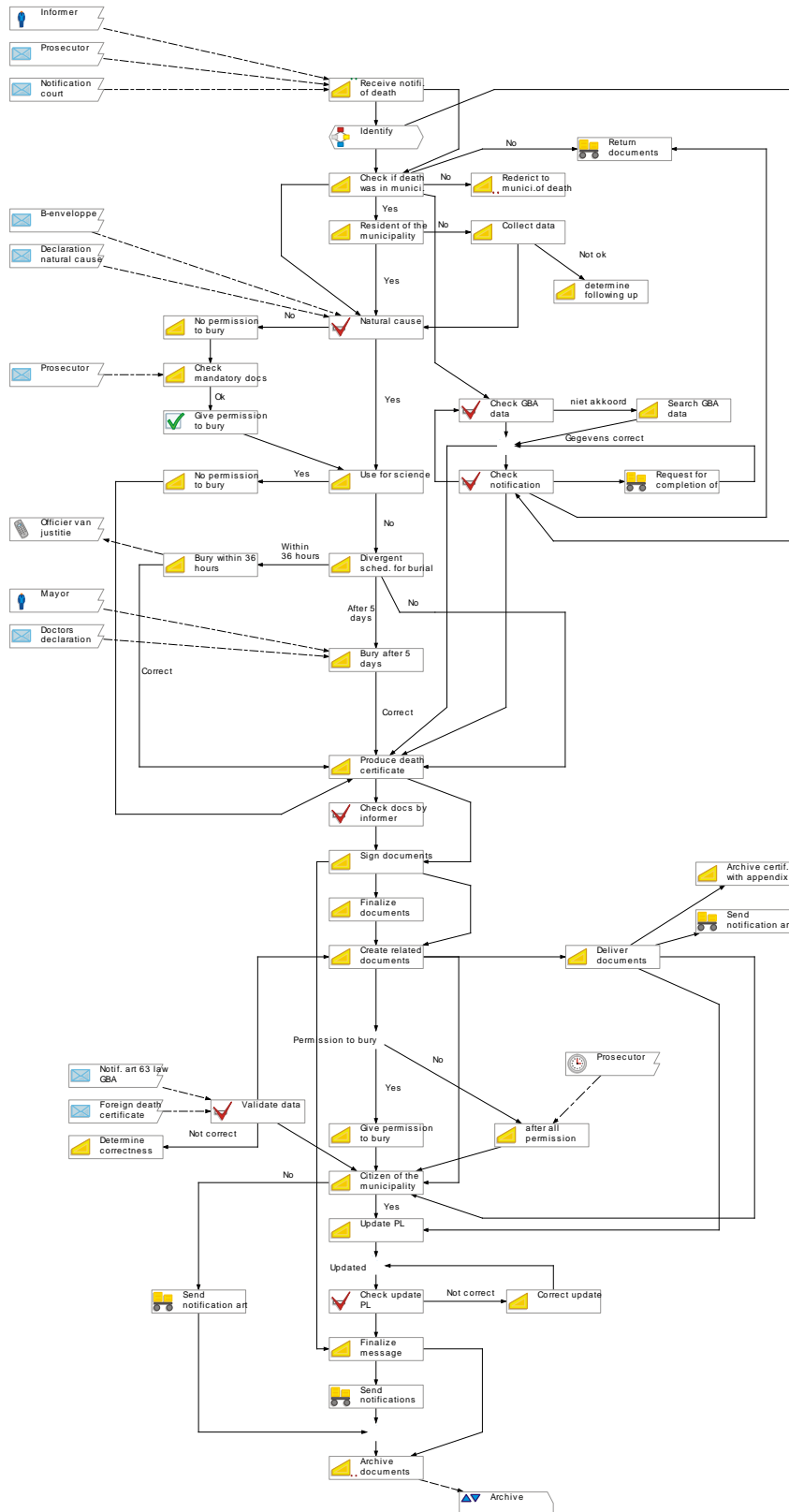
Appendix 25 configurable *birth* business process model



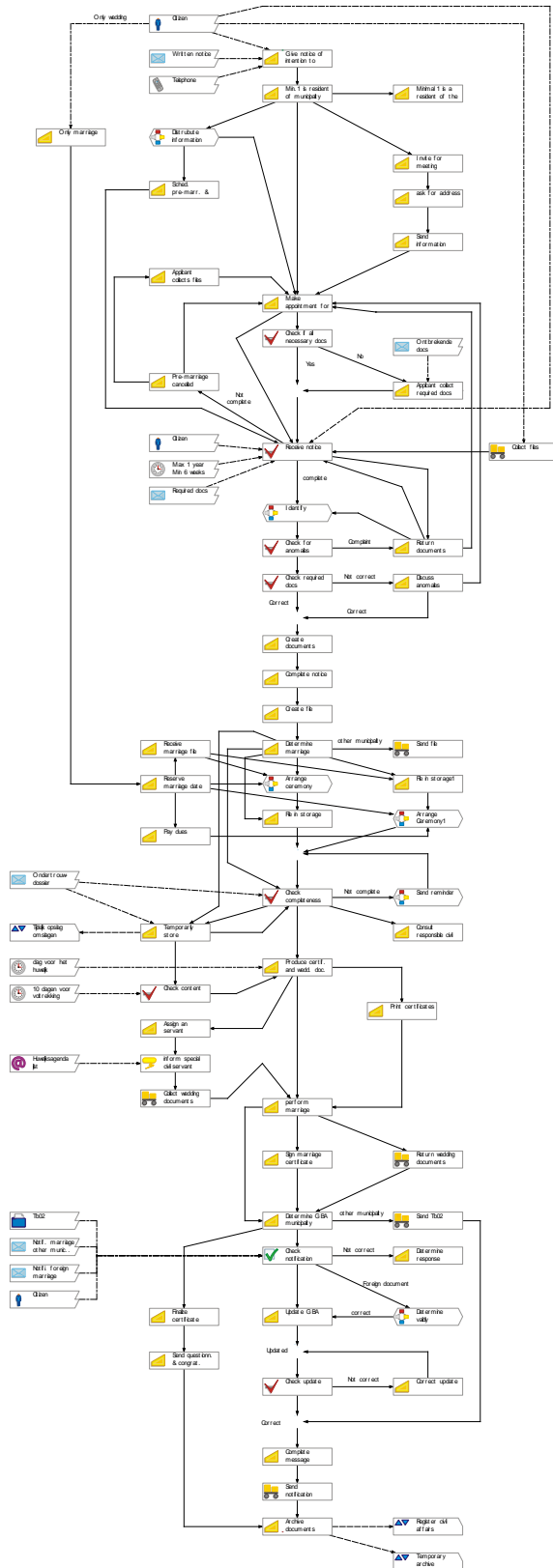
Appendix 26 configurable *acknowledgement of an unborn child* business process model



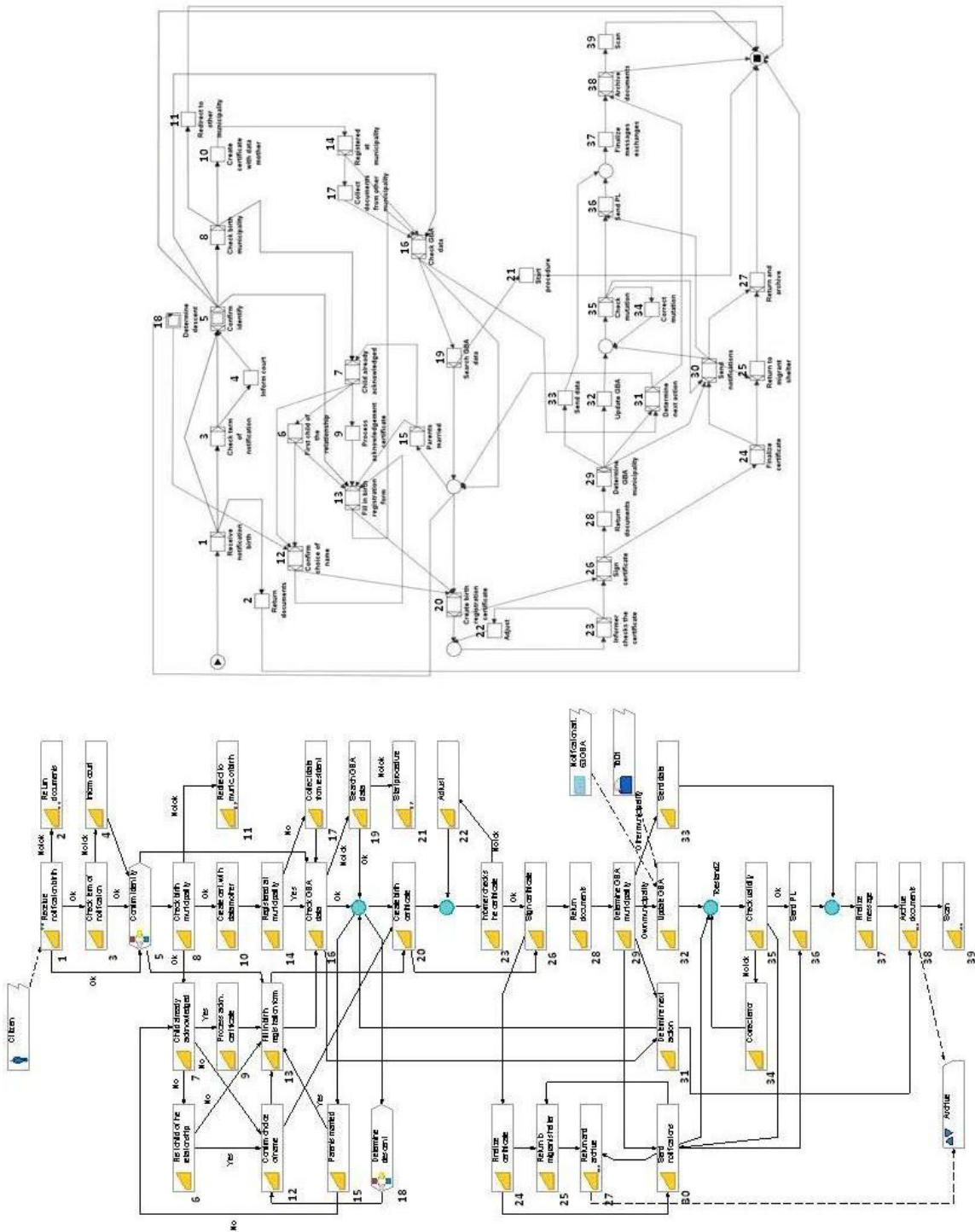
Appendix 27 configurable *deceased* business process model



Appendix 28 configurable *marriage* business process model



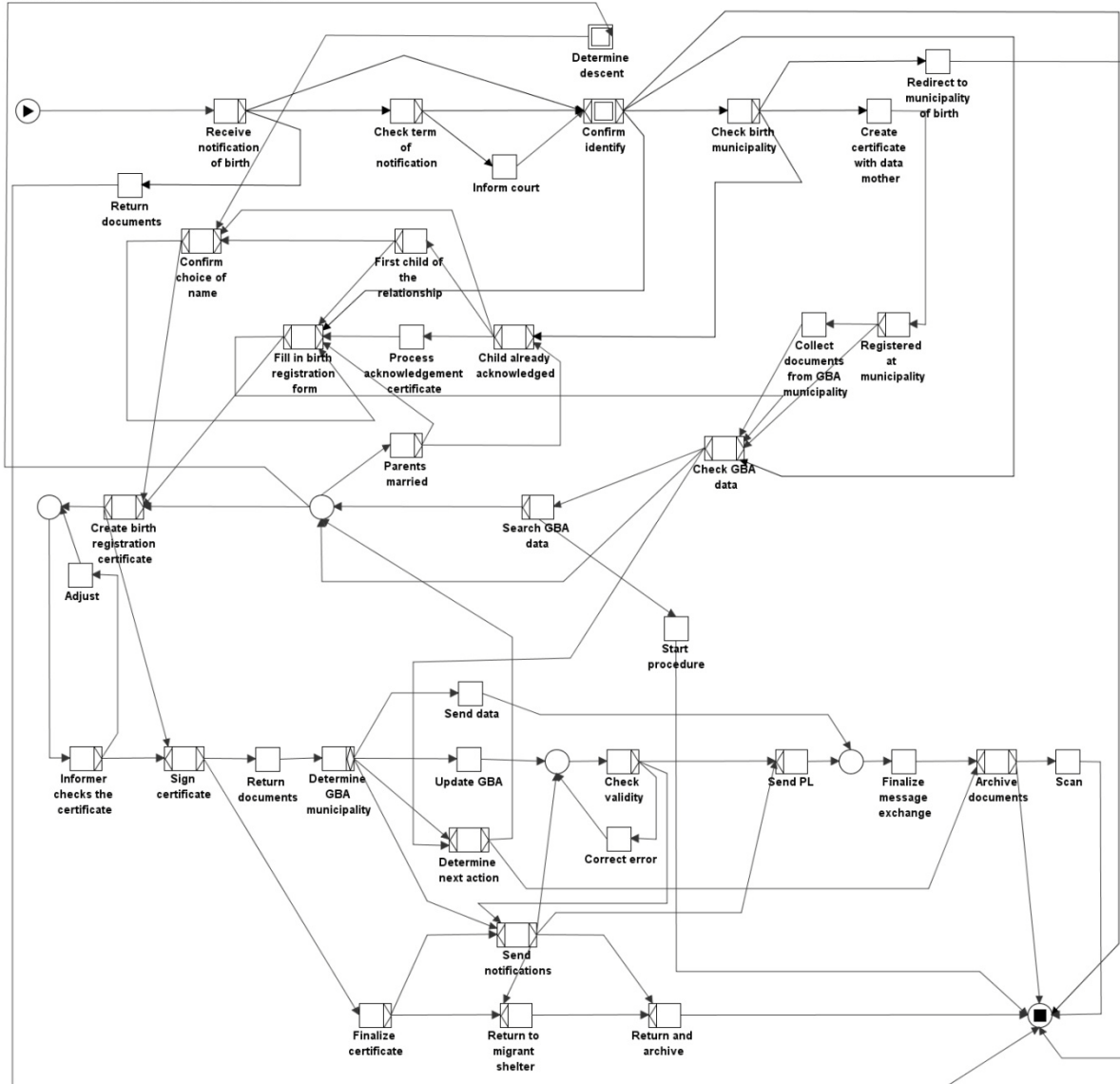
Appendix 29 translation from Protos to YAWL



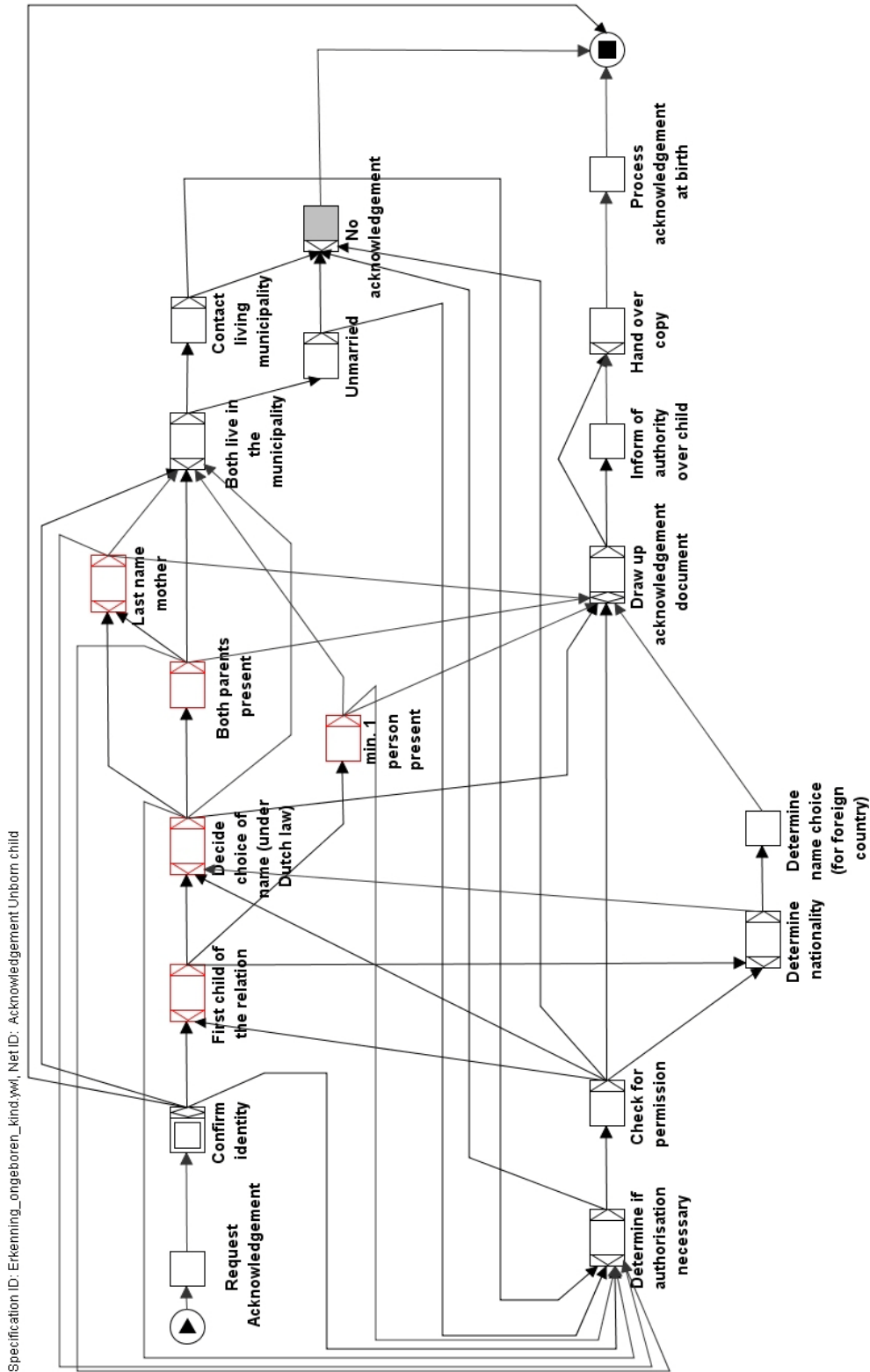
The numbers in the combined birth Protos business process model correspond with the numbers in the configurable birth YAWL workflow model and represent equal tasks/ actions

Appendix 30 Configurable *Birth* Workflow Model in YAWL

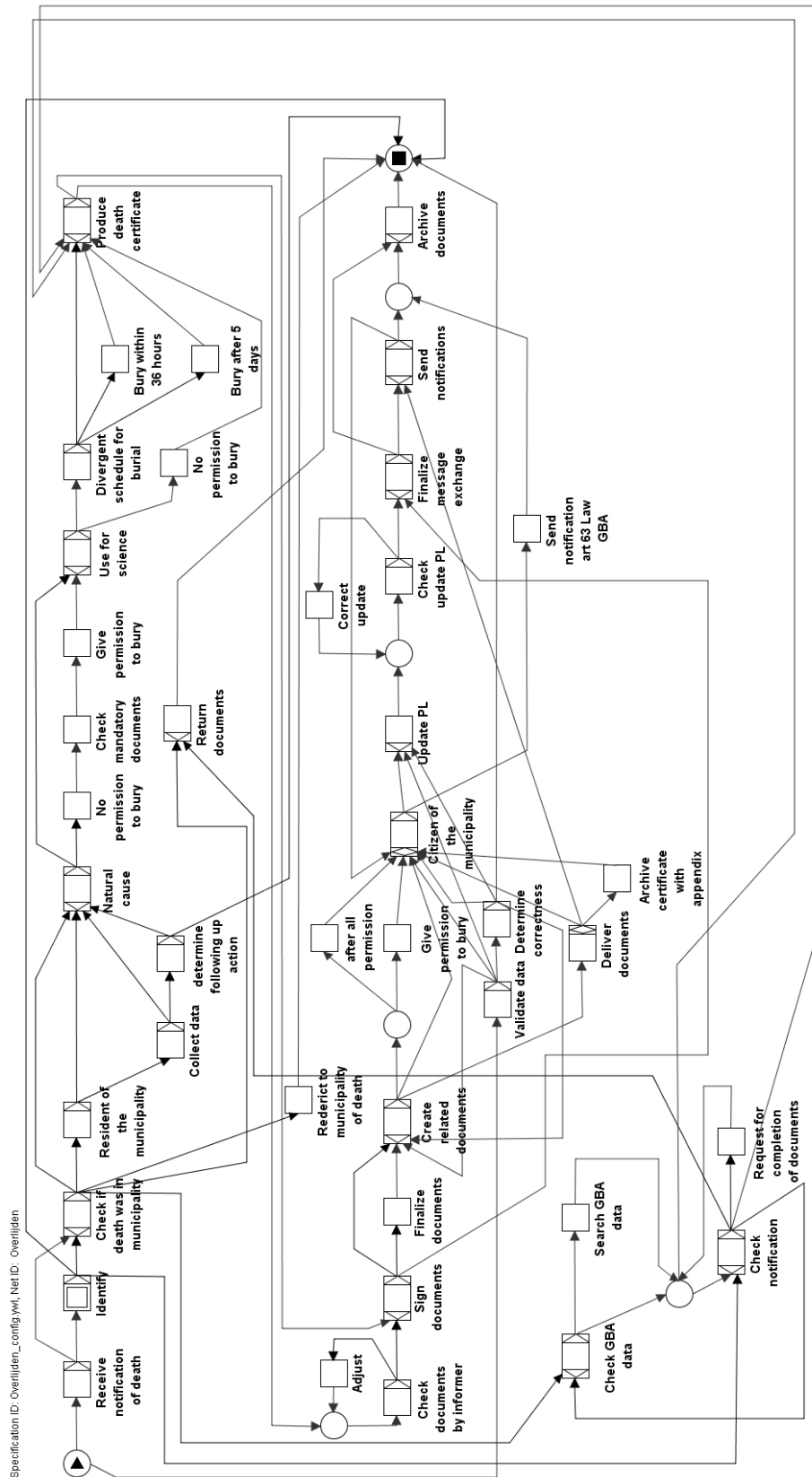
Specification ID: Geboorte_Config_v1.ywl, Net ID: Config Geboorte



Appendix 31 Configurable Acknowledgement of an unborn child Workflow Model in YAWL

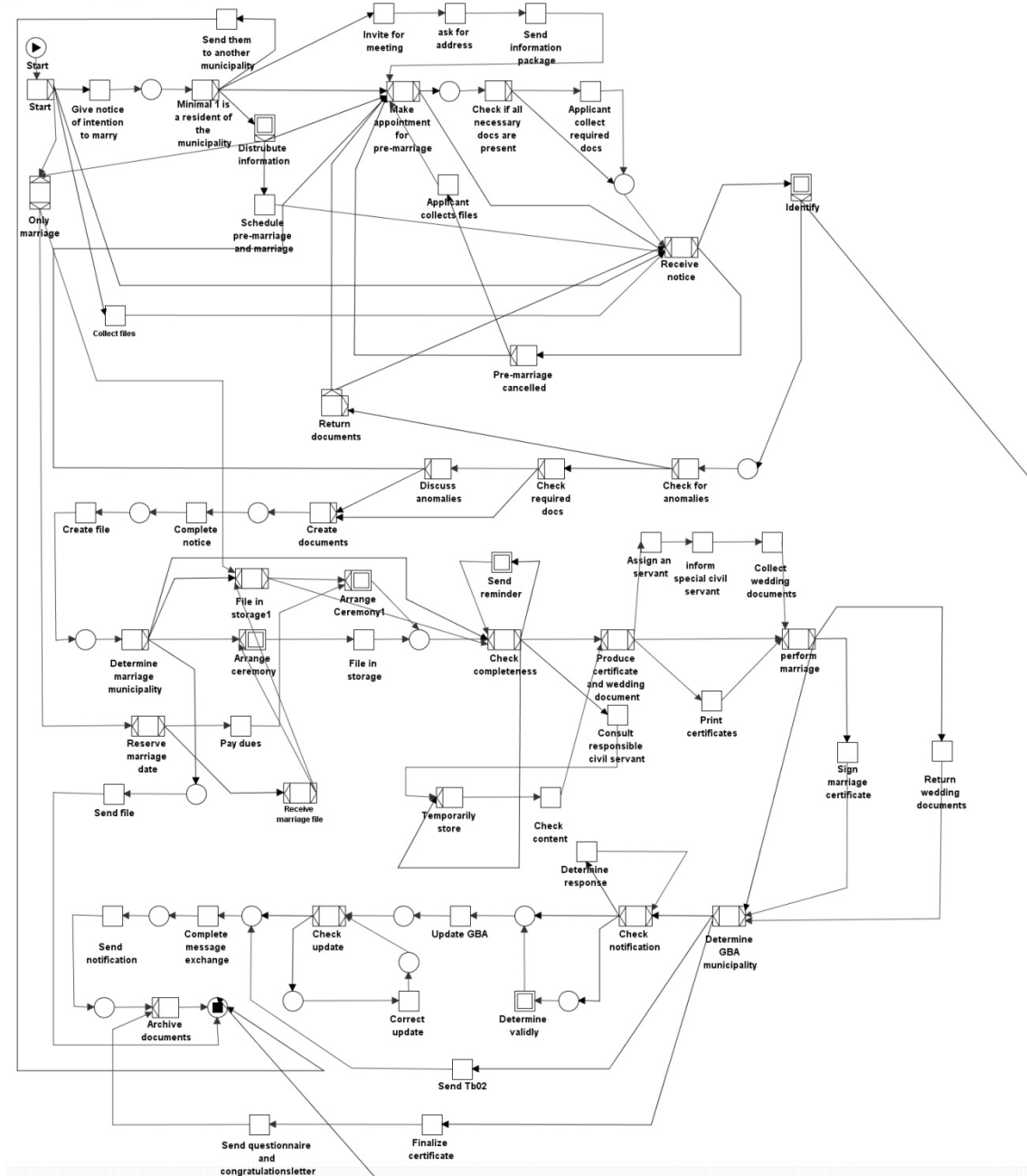


Appendix 32 Configurable *Deceased* Workflow Model in YAWL



Appendix 33 Configurable *Marriage* Workflow Model in YAWL

Specification ID: Huzen_huwelijk_wf, Net ID: Huwelijk Huzen



Appendix 34 Adding data to the YAWL model

To add data to a YAWL model we first need to specify the *net variables*. These variables are used to store information relating to the process Birth that tasks within the workflow may need to read or update. In Figure appendix 27-1 some net variables are depicted.

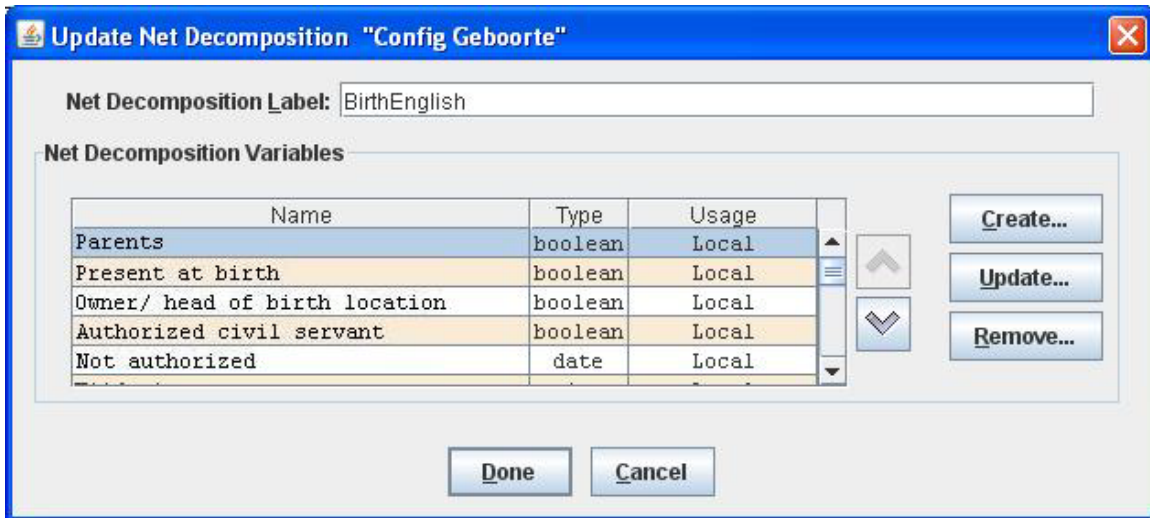


Figure appendix 32-1 net variables of the process Birth

Besides creation of net variables also task variables need to be specified for the process Birth. The task variables of a task store specific information relating to that task. Task variables have several uses. One use is to transferring information between workflow users and workflow engine. A second use is for passing data between web services and/or external code and/or applications that the running workflow engine invokes and the Net task resides in [14]. The task variables that we specified for the task *receive birth notification* are depicted in Figure appendix 27-2

After specifying the task variables we need to specify the parameters of the task. Both Input and Output Parameters can be assigned to any tasks to allow the passing of state between nets and their tasks, and between tasks and workflow engine users and web services [14]. In the case of *Receive notification birth*, the parameters are depicted in Figure appendix 27-3. In the figure no Input parameters are depicted because we don't need input from net variables, and *receive notification birth* is the first task in the process it cannot receive input from other variables in the net. The output is generated by the workflow engine users by answering the Boolean

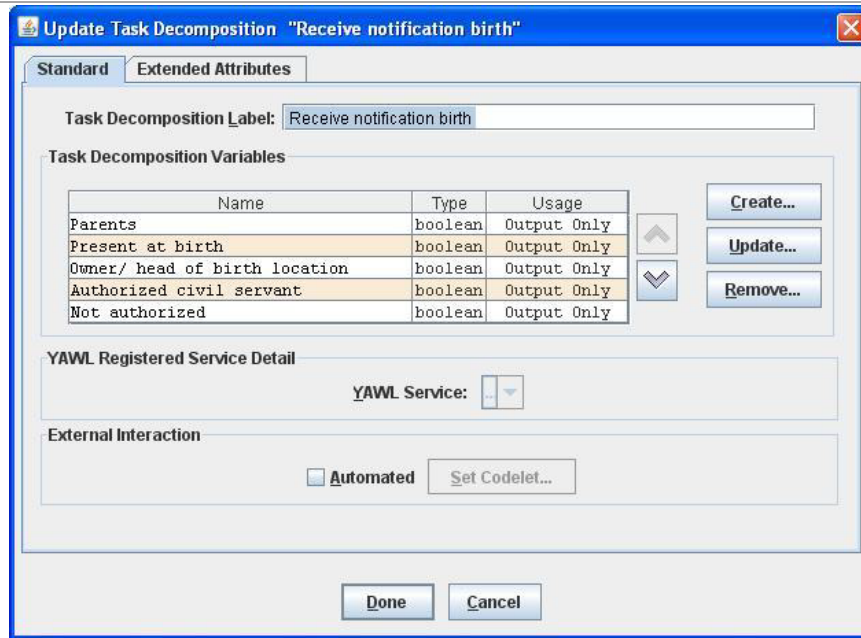


Figure appendix 32-2 Task variables of the receive birth notification task

questions. The results are stored in the net variables. Therefore there needs to be a link between the task variables and the net variables.

When all this steps are completed for all tasks in the workflow we can loaded in to workflow engine. The result is shown in Figure 2-5 in chapter 2.

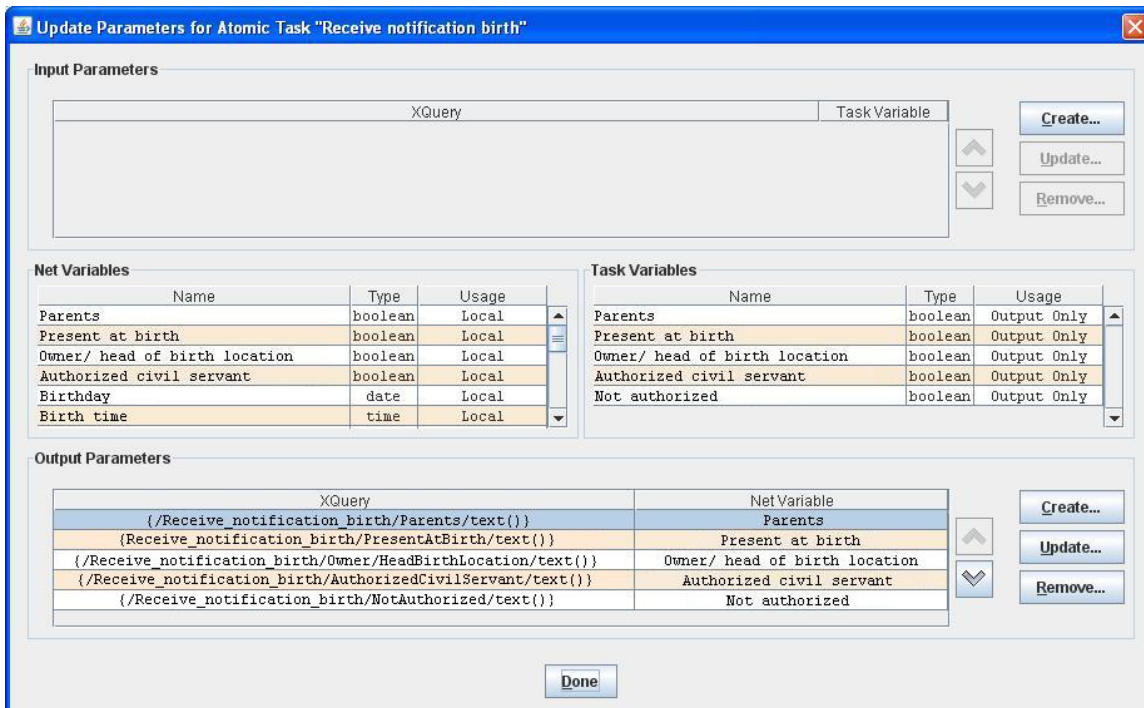
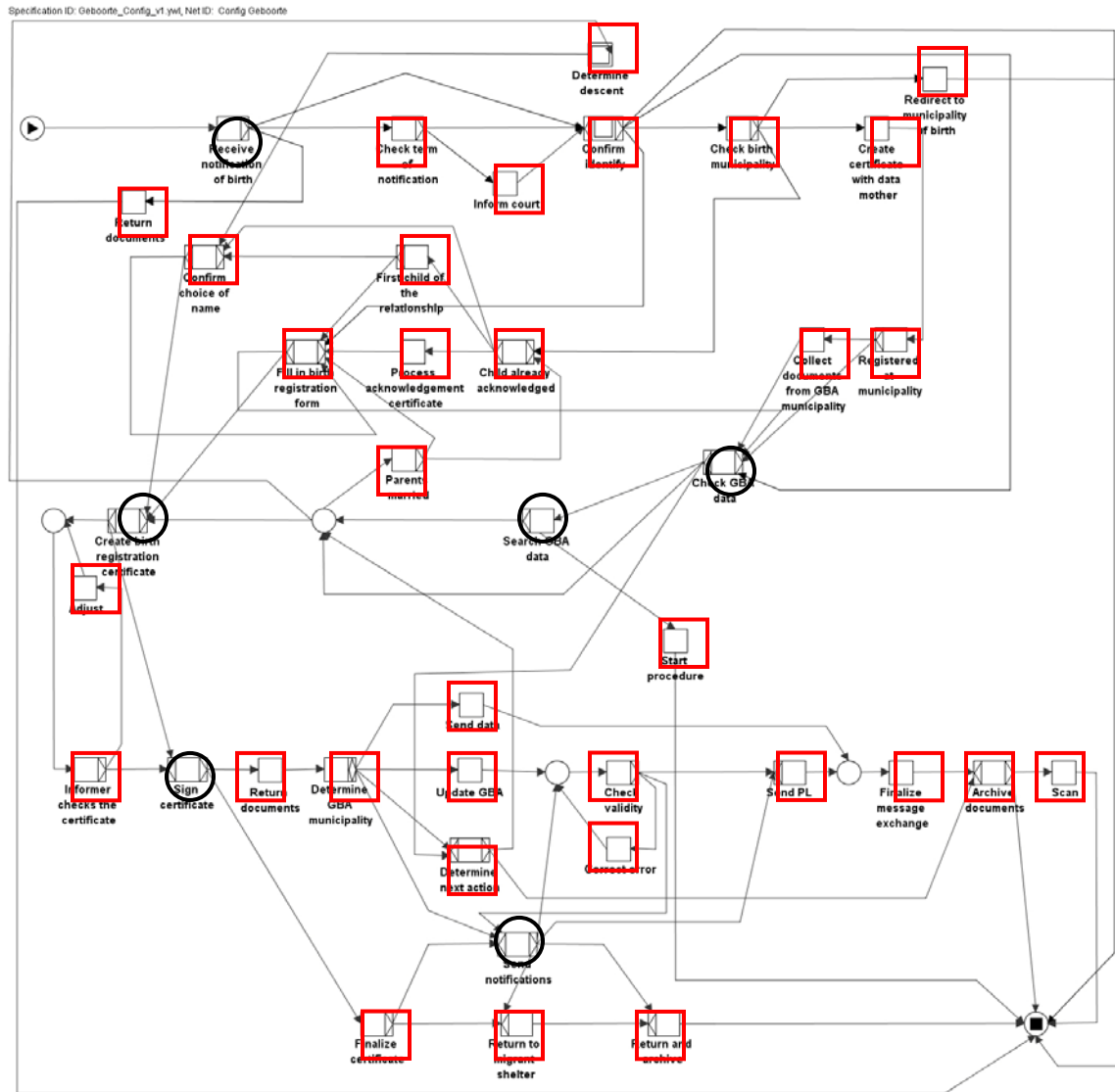




Figure appendix 32-3 task variables of the process Birth

Appendix 35 The variations in the Configurable *Birth* Workflow model



-  Occurring in all the models
-  Variations/ differences

Appendix 36 Excel compilation sheet

Configuration tasks of sub process Birth	f11	f12	f13	f14	f15	f16	f17	f18	f19	f20	f21	f22	f23	f24	f25	f26	f27	f28	f29	
Ontvangen_aangifte_geb_5 -> Controleren_termijn_van_aangifte_6																				
Ontvangen_aangifte_geb_5 -> Retourneren_documenten_3		Blocked																		
Ontvangen_aangifte_geb_5 -> Vaststellen_identiteit_8		Blocked																		
Vaststellen_identiteit_6 -> Controleren_GBA_gegevens_55					Blocked		Blocked													
Vaststellen_identiteit_8 -> Geboorte_gemeente_46					Blocked		Blocked													
Vaststellen_identiteit_8 -> Invullen_form_aangifte_geb_81																				
Geboorte_gemeente_48 -> Kind_al_erkent_78																				
Geboorte_gemeente_48 -> Akte_maken_op_basis_moeder_52																				
Invullen_form_aangifte_geb_81 -> opmaken_geboorteakte_172																				
Invullen_form_aangifte_geb_81 -> Controleren_GBA_gegevens_55																				
Controleren_GBA_gegevens_55 -> Bepalen_vervolgactie_211					Blocked		Blocked													
Controleren_GBA_gegevens_55 -> null_140					Blocked		Blocked													
Onderzoeken_GBA_gegevens_139 -> Opstarten_procedure_302																				
null_140 -> Ouders_ghenwd_109																				
null_140 -> Bepalen_astammingsrecht_331																				
null_140 -> opmaken_geboorteakte_172																				
Kind_al_erkent_78 -> Bepalen_naamskeuze_108																				
Kind_al_erkent_78 -> Eerste_kind_ut_de_relate_79																				
Kind_al_erkent_78 -> erkeningsakte_verreken_80																				
Bepalen_naamskeuze_108 -> Invullen_form_aangifte_geb_81																				
Bepalen_naamskeuze_108 -> opmaken_geboorteakte_172																				
Invullen_form_aangifte_geb_81 -> Bepalen_naamskeuze_108																				
Invullen_form_aangifte_geb_81 -> Controleren_GBA_gegevens_55																				
Invullen_form_aangifte_geb_81 -> opmaken_geboorteakte_172																				
opmaken_geboorteakte_172 -> null_265																				
opmaken_geboorteakte_172 -> Ondertekenen_akten_208																				
Ondertekenen_akten_208 -> Passeren_akte_327																				
Ondertekenen_akten_208 -> Retourneren_documenten_209																				
Bepalen_GBA_gemeente_1175 -> Verzetten_Gegevens_1176																				
Bepalen_GBA_gemeente_1175 -> Actualiseren_GBA_212																				
Bepalen_GBA_gemeente_1175 -> Bepalen_vervolgactie_211																				
Bepalen_GBA_gemeente_1175 -> Versturen_kennisgevingen_328																				
Bepalen_vervolgactie_211 -> null_140																				
Bepalen_vervolgactie_211 -> Archiveren_documenten_218																				
Controleren_mutatie_213 -> Toesturen_PL_216																				
Controleren_mutatie_213 -> Versturen_kennisgevingen_328																				
Archiveren_documenten_218 -> Scannen_219																				
Archiveren_documenten_218 -> OutputCondition_2																				
Versturen_kennisgevingen_328 -> Retourneren_en_archiveren_330																				
Versturen_kennisgevingen_328 -> null_312																				
Versturen_kennisgevingen_328 -> Toesturen_PL_216																				
Versturen_kennisgevingen_328 -> Aangeven_vestiging_329																				
Configuration tasks of sub process Determine descent																				
InputCondition_890 -> Bepalen_staet_moeder_892		Blocked																		
InputCondition_890 -> Afstamming_conform_Wca_893		Blocked																		
InputCondition_890 -> Afstamming_ongehuwd_894		Blocked																		
Afstamming_conform_Wca_893 -> Vaststellen_nationaliteit_kind_1101		Blocked																		
Afstamming_conform_Wca_893 -> Bepalen_nationaliteit_kind_ongehuwd_895		Blocked																		
Afstamming_ongehuwd_894 -> Vaststellen_nationaliteit_kind_1101		Blocked																		
Afstamming_ongehuwd_894 -> Bepalen_nationaliteit_kind_ongehuwd_896		Blocked																		

Appendix 37 XML file *Birth* mapping

```
<?xml version="1.0" encoding="UTF-8"?>
<tns:CM_Mapping xmlns:tns="http://www.fit.qut.edu.au/BPM/configuration/CM_Mapping"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.fit.qut.edu.au/BPM/configuration/CM_Mapping
CM_Mapping.xsd">
  <c-yawl>
    <splits>
      <port sourceId="Ontvangen_aangifte_geb._5"
targetId="Controleren_termijn_van_aangifte_6">
        <value type="blocked" condition="f4"/>
      </port>
      <port sourceId="Ontvangen_aangifte_geb._5" targetId="Retourneren_documenten_3">
        <value type="blocked" condition="f2"/>
      </port>
      <port sourceId="Ontvangen_aangifte_geb._5" targetId="Vaststellen_identiteit_8">
        <value type="blocked" condition="f3"/>
      </port>
      <port sourceId="Vaststellen_identiteit_8" targetId="Controleren_GBA_gegevens_55">
        <value type="blocked" condition="f5 V f7"/>
      </port>
      <port sourceId="Vaststellen_identiteit_8" targetId="Geboorte_gemeente_48">
        <value type="blocked" condition="f5 V f6"/>
      </port>
      <port sourceId="Vaststellen_identiteit_8" targetId="Invullen_form_aangifte_geb._81">
        <value type="blocked" condition="f6 V f7"/>
      </port>
      <port sourceId="Geboorte_gemeente_48" targetId="Kind_al_erkent_78">
        <value type="blocked" condition="f8"/>
      </port>
      <port sourceId="Geboorte_gemeente_48" targetId="Akte_maken_op_basis_moeder_52">
        <value type="blocked" condition="f9"/>
      </port>
      <port sourceId="Invullen_form_aangifte_geb._81" targetId="Bepalen_naamskeuze_108">
        <value type="blocked" condition="f9 V f12"/>
      </port>
      <port sourceId="Invullen_form_aangifte_geb._81"
targetId="Controleren_GBA_gegevens_55">
        <value type="blocked" condition="f12"/>
      </port>
      <port sourceId="Invullen_form_aangifte_geb._81" targetId="opmaken_geboorteakte_172">
        <value type="blocked" condition="f5 V f9"/>
      </port>
      <port sourceId="Controleren_GBA_gegevens_55" targetId="Bepalen_vervolgactie_211">
        <value type="blocked" condition="f5 V f6"/>
      </port>
      <port sourceId="Onderzoeken_GBA_gegevens_139" targetId="Opstarten_procedure_902">
        <value type="blocked" condition="f4"/>
      </port>
      <port sourceId="Kind_al_erkent_78" targetId="Bepalen_naamskeuze_108">
        <value type="blocked" condition="f12"/>
      </port>
      <port sourceId="Kind_al_erkent_78" targetId="Eerste_kind_uit_de_relatie_79">
        <value type="blocked" condition="f9"/>
      </port>
      <port sourceId="Bepalen_naamskeuze_108" targetId="Invullen_form_aangifte_geb._81">
        <value type="blocked" condition="f11"/>
      </port>
      <port sourceId="Bepalen_naamskeuze_108" targetId="opmaken_geboorteakte_172">
        <value type="blocked" condition="f9 V f12"/>
      </port>
      <port sourceId="opmaken_geboorteakte_172" targetId="null_265">
        <value type="blocked" condition="f14"/>
      </port>
      <port sourceId="opmaken_geboorteakte_172" targetId="Ondertekenen_akten_208">
        <value type="blocked" condition="f13"/>
      </port>
      <port sourceId="Ondertekenen_akten_208" targetId="Passeren_akte_327">
```

```

    <value type="blocked" condition="f16"/>
  </port>
  <port sourceId="Ondertekenen_akten_208" targetId="Retourneren_documenten_209">
    <value type="blocked" condition="f15"/>
  </port>
  <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Bepalen_vervolgactie_211">
    <value type="blocked" condition="f7 v f 19"/>
  </port>
  <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Versturen_kennisgevingen_328">
    <value type="blocked" condition="f7 v f 17 v f 19"/>
  </port>
  <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Bepalen_vervolgactie_211
Versturen_kennisgevingen_328 Aktualiseren_GBA_212 Verzenden_Gegevens_1176">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
    <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Bepalen_vervolgactie_211
Versturen_kennisgevingen_328 Aktualiseren_GBA_212">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
    <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Bepalen_vervolgactie_211
Versturen_kennisgevingen_328 Verzenden_Gegevens_1176">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
    <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Bepalen_vervolgactie_211
Aktualiseren_GBA_212 Verzenden_Gegevens_1176">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
    <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Versturen_kennisgevingen_328
Aktualiseren_GBA_212 Verzenden_Gegevens_1176">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
    <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Bepalen_vervolgactie_211
Versturen_kennisgevingen_328">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
    <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Bepalen_vervolgactie_211
Verzenden_Gegevens_1176">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
    <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Bepalen_vervolgactie_211
Aktualiseren_GBA_212">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
    <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Verzenden_Gegevens_1176
Versturen_kennisgevingen_328">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
    <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Aktualiseren_GBA_212
Verzenden_Gegevens_1176">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
    <port sourceId="Bepalen_GBA_gemeente_1175" targetId="Aktualiseren_GBA_212
Versturen_kennisgevingen_328">
    <value type="blocked" condition="f17 v f 18 v f 19"/>
  </port>
  <port sourceId="Bepalen_vervolgactie_211" targetId="null_140">
    <value type="blocked" condition="f5 v f 6"/>
  </port>
  <port sourceId="Bepalen_vervolgactie_211" targetId="Archiveren_documenten_218">
    <value type="blocked" condition="f7"/>
  </port>
  <port sourceId="Controleren_mutatie_213" targetId="Versturen_kennisgevingen_328">
    <value type="blocked" condition="f18"/>
  </port>
    <port sourceId="Controleren_mutatie_213" targetId="Toesturen_PL_216">
    <value type="blocked" condition="f17 v f 19"/>
  </port>
  <port sourceId="Archiveren_documenten_218" targetId="Scannen_219">
    <value type="blocked" condition="f17 v f 19"/>
  </port>

```

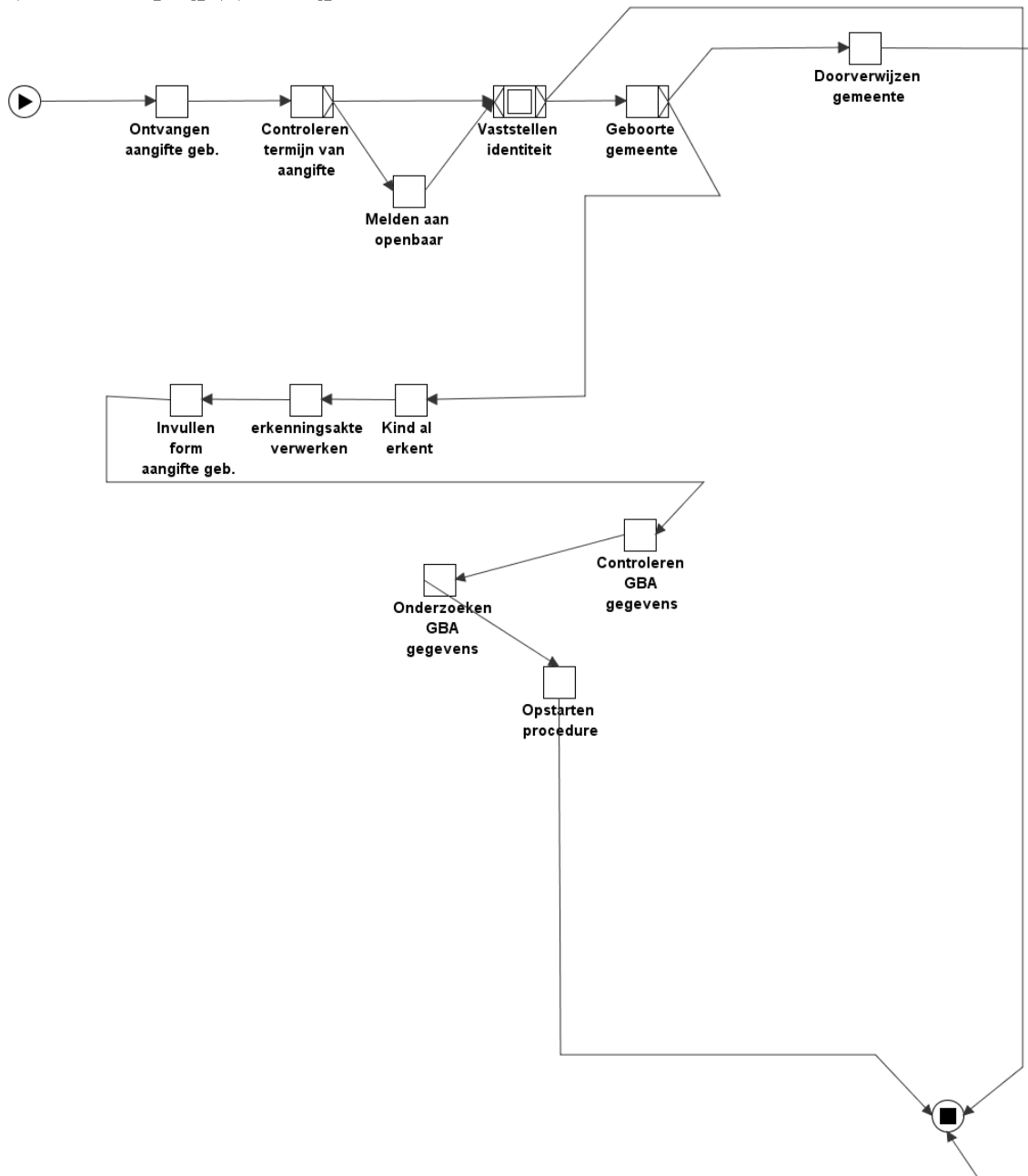
```

    <port sourceId="Archiveren_documenten_218" targetId="OutputCondition_2">
      <value type="blocked" condition="f18"/>
    </port>
    <port sourceId="Versturen_kennisgevingen_328"
targetId="Retourneren_en_archiveren_330">
      <value type="blocked" condition="f16"/>
    </port>
    <port sourceId="Versturen_kennisgevingen_328" targetId="null_312">
      <value type="blocked" condition="f15 v f17 v f19"/>
    </port>
      <port sourceId="Versturen_kennisgevingen_328" targetId="Toesturen_PL_216">
        <value type="blocked" condition="f15 v f18"/>
      </port>
    <port sourceId="Versturen_kennisgevingen_328" targetId="Aangeven_vestiging_329">
      <value type="blocked" condition="f16"/>
    </port>
      <port sourceId="Afstamming_conform_Wca_893"
targetId="Vaststellen_nationaliteit_kind_1101">
        <value type="blocked" condition="f4"/>
      </port>
      <port sourceId="Afstamming_conform_Wca_893"
targetId="Bepalen_nationaliteit_kind_gehuwd_895">
        <value type="blocked" condition="f3"/>
      </port>
      <port sourceId="Afstamming_ongehuwd_894"
targetId="Vaststellen_nationaliteit_kind_1101">
        <value type="blocked" condition="f4"/>
      </port>
      <port sourceId="Afstamming_ongehuwd_894"
targetId="Bepalen_nationaliteit_kind_ongehuwd_896">
        <value type="blocked" condition="f3"/>
      </port>
    </splits>
    <joins>
      <port sourceId="null_140" targetId="Ouders_gehuwd_109">
        <value type="activated" condition="¬f10 ∧ ¬f11"/>
        <value type="blocked" condition="f10 v f11"/>
      </port>
      <port sourceId="null_140" targetId="Bepalen_afstammingsrecht_331">
        <value type="activated" condition="¬f10 ∧ ¬f12"/>
        <value type="blocked" condition="f10 v f12"/>
      </port>
      <port sourceId="null_140" targetId="opmaken_geboorteakte_172">
        <value type="activated" condition="¬f11 ∧ ¬f12"/>
        <value type="blocked" condition="f11 v f12"/>
      </port>
      <port sourceId="InputCondition_890" targetId="Bepalen_staatsmoeder_892">
        <value type="activated" condition="¬f3"/>
        <value type="blocked" condition="f3"/>
      </port>
      <port sourceId="InputCondition_890" targetId="Afstamming_conform_Wca_893">
        <value type="activated" condition="¬f4"/>
        <value type="blocked" condition="f4"/>
      </port>
      <port sourceId="InputCondition_890" targetId="Afstamming_ongehuwd_894">
        <value type="activated" condition="¬f4"/>
        <value type="blocked" condition="f4"/>
      </port>
    </joins>
    <rems>
  </rems>
  <nofis>
</nofis>
</c-yawl>
</tns:CM_Mapping>

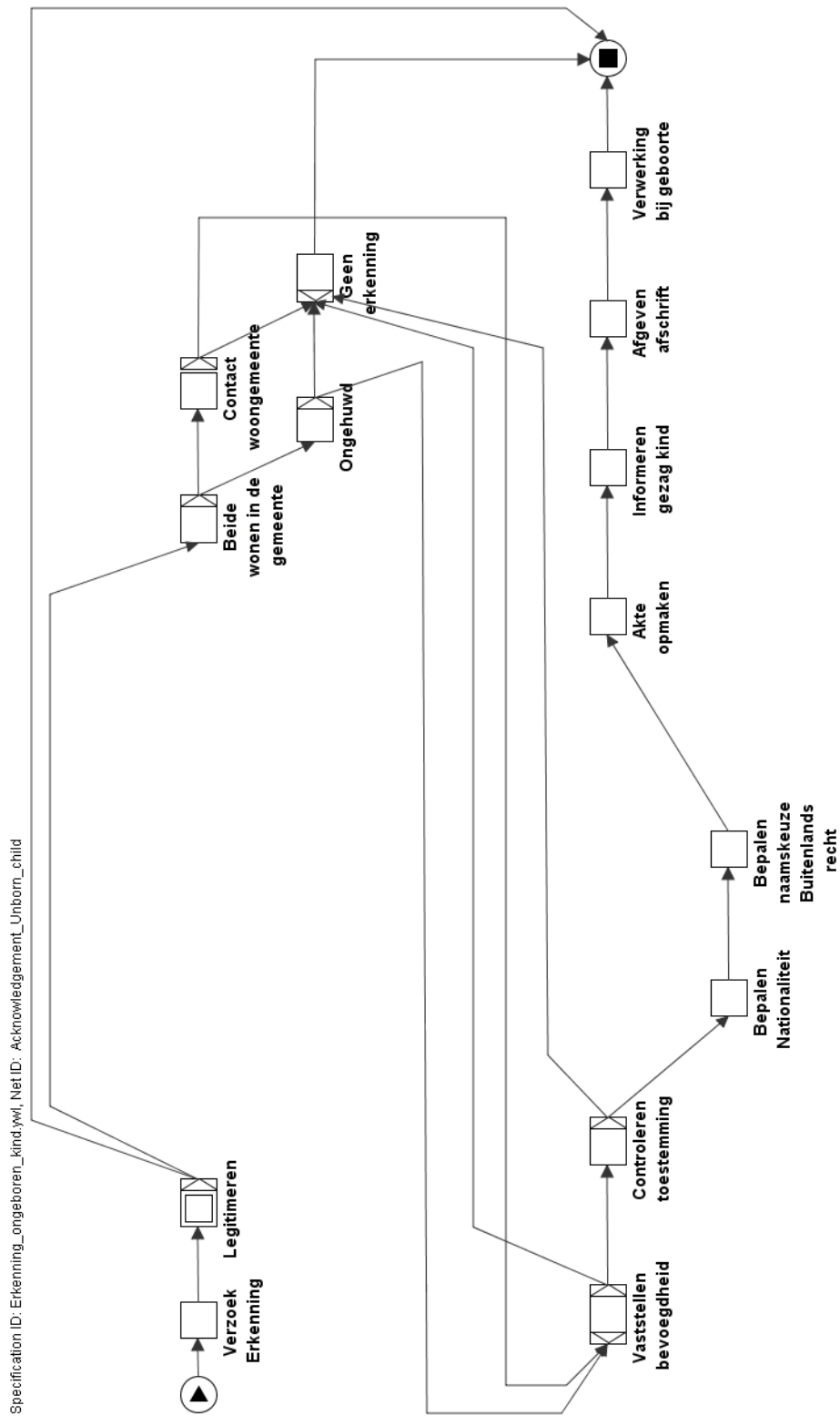
```

Appendix 38 Configured *birth* Workflow Model in YAWL

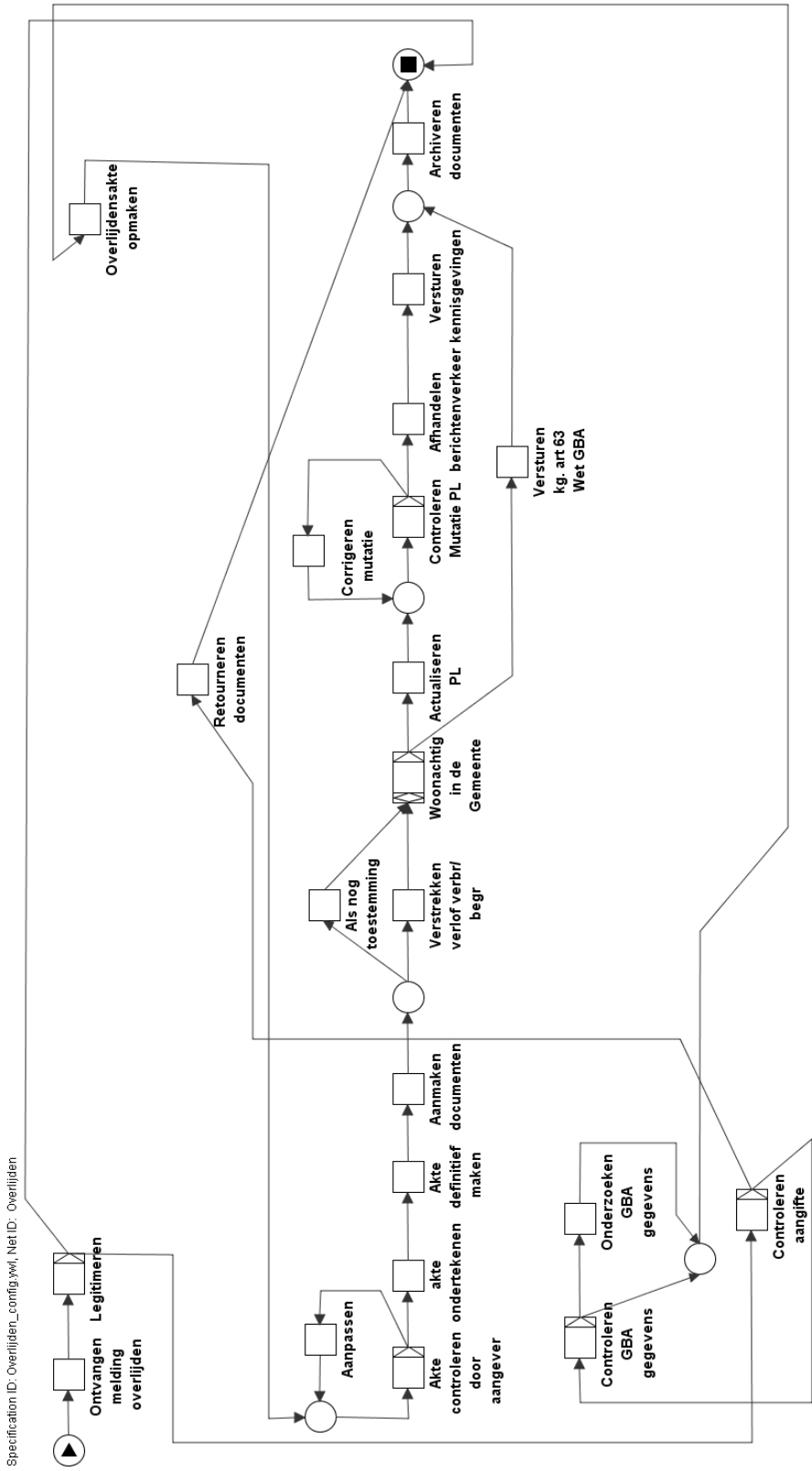
Specification ID: Geboorte_Config_v1.ywl, Net ID: Config_Geboorte



Appendix 39 Configured *Acknowledgement of an unborn child* Workflow Model in YAWL



Appendix 40 Configured *Deceased* Workflow Model in YAWL



Appendix 41 Configured *Marriage* Workflow Model in YAWL

Specification ID: Huizen_huwelijk.ywl, Net ID: Huwelijk_Huizen

