National University of Ireland, Galway

Doctoral Thesis

Business Capability-centric Management of Services and Process Models

Author:
Wassim Derguech

Supervisors:
Dr. Edward Curry
Dr. Sami Bhiri

Examiners
Prof. Jan Mendling
Prof. François Charoy
Prof. Dietrich Rebholz-Schuhmann

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

Insight Centre for Data Analytics

College of Engineering and Informatics

January, 2017

Declaration of Authorship

I, Wassim Derguech, declare that this thesis titled, "Business Capability-centric Management of Services and Process Models" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Wassim Derguech

January, 2017

"Education is not the learning of facts, but the training of the mind to think."

Albert Einstein

NATIONAL UNIVERSITY OF IRELAND, GALWAY

Abstract

Insight Centre for Data Analytics
College of Engineering and Informatics

Doctor of Philosophy

Business Capability-centric Management of Services and Process Models

by Wassim DERGUECH

With the advent of Industry 4.0, more and more companies are actively working on digitising their assets (i.e., services, processes, etc.) for better control, collaboration, modularity, analysis, etc. By 2020 more than 80% of companies will have digitised their business processes and value chains. This creates more services and processes, making their indexing, discovery, configuration, etc. more challenging. Thus, digitising assets needs a data model to describe them together with algorithms for indexing, discovery and configuration.

This thesis details a concept model for describing the business capability of services and business processes from a functional perspective in terms of what do they achieve together with related business properties. Furthermore, this work proposes the aggregation, indexing, discovery and configuration of services and business processes using the concept of business capability.

The first contribution of this thesis is a conceptual model for describing Business Capabilities. The model is implemented as a set of ontologies that can be used for creating semantic annotations of business process models or services. This model is verified using ontological evaluation by mapping its constructs with ontology constructs and verifying there is not an overload or semantic ambiguity. This method has been used for the assessment of relationships in the entity-relationship model. A feature comparison to existing models is performed with respect to three requirements: expressiveness, support for inferencing, use of ontologies and ability to model configuration options. Finally, interviews with domain experts were carried out revealing that the model is simple, easy to adopt and flexible enough to be extended for user requirements.

The second contribution is an abstraction technique that allows moving from an entire process model to its functional description by aggregating the business capabilities of the process elements into a single one. The idea of the algorithm is to traverse the model from an initial node to a final node. Each intermediate node introduces changes to the propagated business capability. The propagation steps are validated using formal semantics of process elements in Petri Nets.

The third contribution of this thesis is to explore the use of Formal Concept Analysis for providing efficient indexing and discovery of business capabilities described using the proposed model. This contribution is validated in a real world scenario for indexing sensor capabilities. A quantitative evaluation is conducted with synthetic sensor capabilities for indexing and discovering 5000 entities. The results show that the indexing and discovery time are less than 200 ms.

The fourth contribution of this thesis is to reduce the business process modelling effort when using configurable process models via an algorithm for creating business capability-annotated configurable business process models that captures configuration options in terms of business capability features. The merging operation realises a compression rate of 50% (by computing the number of nodes before and after the merging).

Dedicated to Mum, Dad, Mejda, Rayen and Ilyes

Acknowledgements

I would like to express my gratitude to my advisors Dr. Edward Curry and Dr. Sami Bhiri for their excellent guidance and precious advices along the PhD work. Their experiences, knowledge and working methodologies were essential for completing this work.

I would like to thank my graduate research committee members: Prof. Manfred Hauswirth, Prof. Stefan Decker, and Dr. Adegboyega Ojo. Their recommendations at various steps of the PhD were valuable for shaping further this research and directing it towards its completion. I am also very grateful to my examiners Prof. Jan Mendling, Prof. François Charoy, and Prof. Dietrich Rebholz-Schuhmann for their time to evaluate this work and their precious feedback and comments, and thanks to Dr. Matthias Nickles for chairing the process.

My thanks also go to Roghaiyeh Gachpaz Hamed, Arslane Chaouche, and Amadou Tangara who contributed to the extensions made to the business process modeling tool for integrating the contributions of this thesis.

I would like to forward my acknowledgement to the funding agencies who supported this work at different stages: Science Foundation Ireland, Enterprise Ireland, and the European Commission. I would also like to thank every member of my institute, the Insight Centre at NUI Galway, and formerly DERI. Fellow researchers, students, staff, and friends, all made this place an exceptional environment for pursuing my PhD venture. The great value of discussions, feedback, and support on every occasion have been priceless.

Very special and warm thanks go to my precious family: my mum, my dad and my brother for their continuous support. My dearest thanks are to my beloved wife, Majda Allani, who has been my source of inspiration, support, courage and determination along these years of study.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"The significant problems we have cannot be solved at the same level of thinking with which we created them."

Albert Einstein

1.1 Research Context

From a historical point of view, when people started to live in social groups, ideas of trading started to appear for exchanging goods and services. This evolved to more formal structures that we know as modern businesses. Businesses continuously optimise their products and services to satisfy their customers and increase their profit [44].

Everything that is carried out in business is driven by processes [44], called also business processes. A business process is a set of ordered activities and tasks that, once completed, achieve the organisation's goal [238]. They constitute valuable assets for businesses as they define and document the important activities, services and resources of an enterprise. The advent of computer systems with automated calculations, algorithms, databases, etc. made their adoption very quick by enterprises for managing their business processes, services and resources using information systems.

Dumas et al. visualise current information systems in multiple layers as shown in Figure 1.1 [65]. The first layer represents the operating system that manages the hardware and makes it run. The second layer represents the generic applications that can be found within multiple departments of the same enterprise. A database management system, a text editor, or a spreadsheet application are examples of such generic applications. The third layer represents the domain specific applications such as a call centre application

or human resource management application that are used within specific departments or enterprises. The fourth layer represents the tailor-made applications that are usually developed for a particular enterprise.

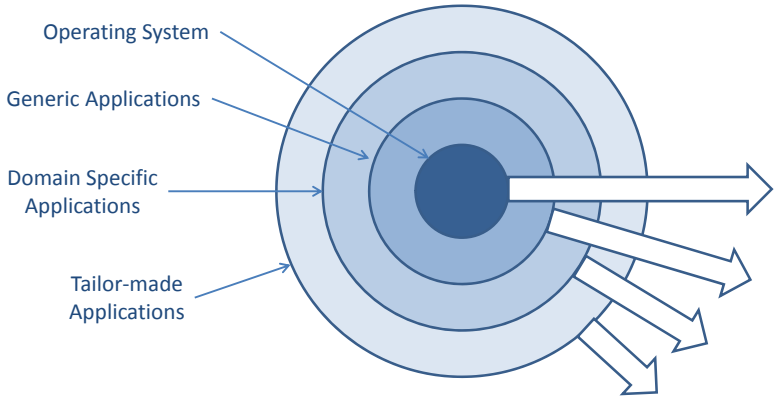


FIGURE 1.1: Trends in Information Systems [65]

In the sixties, the second and third layers shown in Figure 1.1 did not exist, and therefore, information systems were built directly on top of the operating systems making their functionalities very limited [65]. Most enterprise information systems were tailor-made applications and thus their number was increasing and more and more generic functionalities started to appear leading to a new trend of generic applications [65]. Consequently, nowadays, thanks to the second and the third layers and with the increasing performance of operating systems, enterprise information systems are gaining more features aligned with the design principles of Industry 4.0 [124] such as inter-operability, virtualisation, decentralisation, real-time data processing, service orientation and modularity [29].

Industry 4.0 came as natural trend joining intelligent analytics and cyber-physical systems brining new thinking of production management and factory transformation [127]. In this trend, appropriate sensor data together with historical data and contextual information constitute a "Big Data" space to serve intelligent analytics tools for conducting data integration, predictive analytics, data visualisation, etc. Such tools are valuable for enterprises to define the strategic innovation of their business capabilities and drive the development of their Information Systems [34, 127]. Within this trend, it is expected that, in five years, more than 80% of companies will have digitised their business processes and value chains [110]. This creates more services and processes, making their indexing, discovery, configuration, etc. more challenging. Thus, properly digitising those assets needs a proper data model to describe them, and proper algorithms for indexing, discovering and configuring them.

In this context, this thesis aims to propose a concept model for describing what services and business processes can do from a functional perspective (i.e., business capability): What do they achieve and what are the related business properties? Furthermore, this

work proposes the aggregation, indexing, discovery and configuration of services and business processes using the concept of business capability.

In order to highlight further the importance of the business capability in a corporate environment, I introduce three motivational scenarios in Section 1.2 that I will use in the rest of this chapter.

1.2 Motivational Scenarios

In this section I introduce three motivational scenarios that occur with Maria, a business analyst, who has recently joined a large corporate specialized in the management of seaports (i.e., management of Import and Export procedures, logistics services, etc.) to work on the verification and quality assurance of the services offered by the corporate. As a primary mission, Maria needs to review the corporate business processes and their documentation in order to verify their compliance with the corporate regulations and guarantee consistency between them. Maria has to report any issues to her direct manager and these issues will be later review by the IT department for further adjustment of the information system used by this corporate.

1.2.1 Scenario 1: What Do Business Processes Achieve?

At first, Maria needs to understand how business processes work: what tasks each process goes through? What does each task of the process achieve? What does the entire process achieve? etc.

Maria has been provided a set of business process models, similar to the one depicted in Figure 1.2, together with their documentation [1]. From this model, Maria was able to easily identify partially what this process achieves (i.e., its business capability) as it is relatively simple. However, what she misses from this model is the business parameters of each task such as the required documentation, the priority channel conditions, the means of notification, etc. This information can be found in the long 102 pages documentation of this process. Furthermore, the situation might be more complicated if Maria has to deal with more complex processes that involve more than 5 tasks and much more difficult to visualize. Such models, as the one depicted in Figure 1.3, are called 'spaghetti' models

[1] This process model depicts the process of cargo examination process at Davao City Seaport in Philippines using EPC notation. The process has 5 functions represented in rectangles: checking content, selectivity processing, detailed examination, scan x-ray and release and notification. The flow between these functions is controlled via the events shown in diamonds. This model is available at http://kjri-davao.com/?page=news&siteLanguage=English&address%20link=127&cat=Economics as accessed on 06-06-2014.

because of their complex structure that require dedicated tools for assisting business analysts in their management and understanding [58, 224].

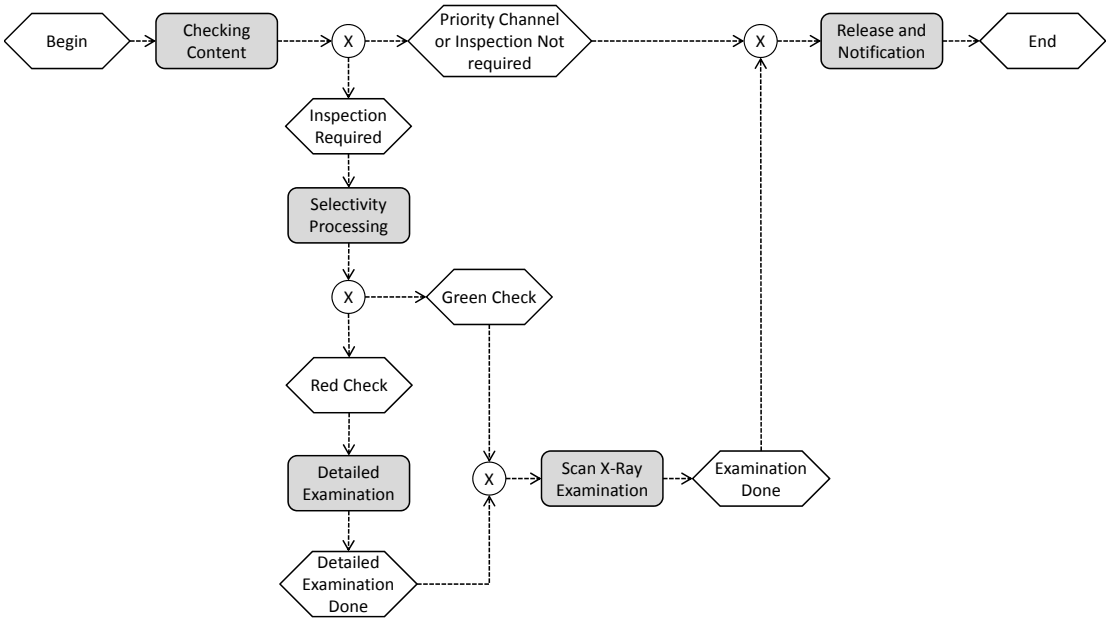


FIGURE 1.2: Cargo Examination Process at Davao City Seaport, Philippines

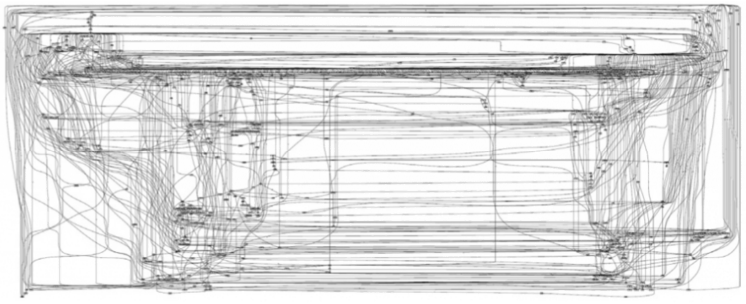


FIGURE 1.3: An Example of a 'spaghetti' Model [224]

Maria needs a solution to help her understand better and quickly both individual tasks and entire business processes that are running in her corporate. She needs to have a clear business centric description of what these business processes achieve (i.e., business capability) without having to refer to a large documentation in order to be able to verify their compliance with the regulations that are usually described using business terms.

1.2.2 Scenario 2: Discovery of Services and Business Processes

Within the same corporate, Maria is not only in charge of the management of core/-operational business processes (i.e., processes that create the core value stream of the company [238]) but also organizational business processes (i.e., that govern the operation of the system and support the core processes such as accounting, security, recruitment,

etc. [238]). Furthermore, operates in various countries. Each country has its own restrictions and requirements that generate multiple variants of operational as well as organizational processes. All these processes are stored in a large repository that Maria needs to query in order to select the processes and services that she needs to verify, understand, test, etc.

The company's repository of services and business processes is designed by John from the IT department. John designed this repository in a way that he can easily query the right process/service using IT capabilities. In his search queries he needs to define what Inputs, Outputs, Preconditions and Effect (IOPEs) are required by a certain process/service. The problem with this repository is that search queries are very tight to the actual IT implementations that Maria is not familiar with. Each time that Maria needs a particular process, she needs the assistance of John to define her search request.

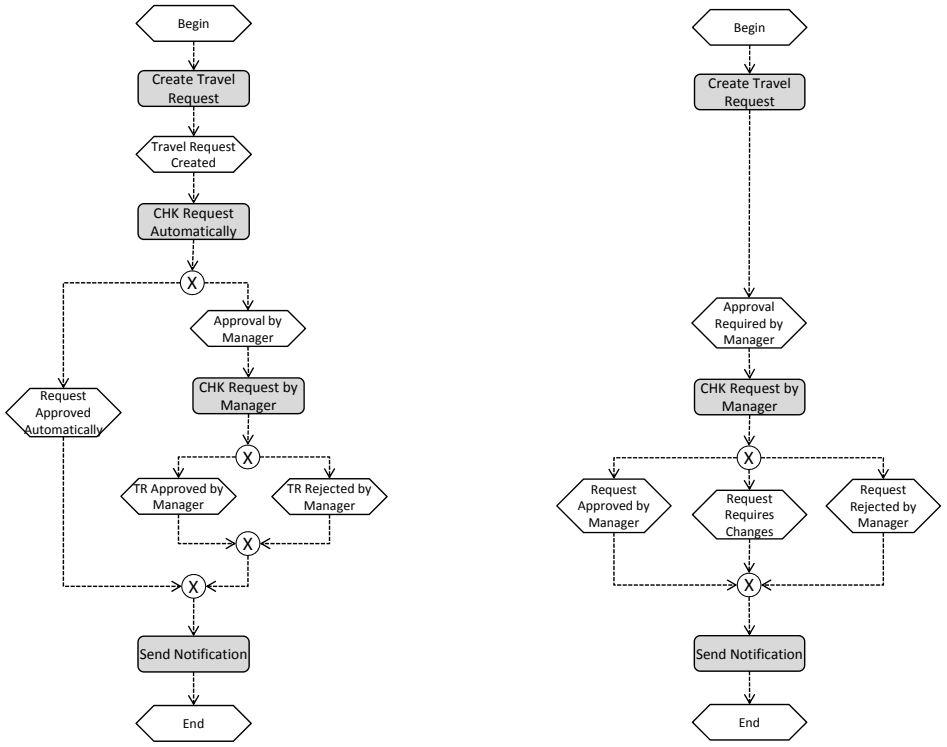
Ideally, Maria would have the possibility to discover the services/processes that she needs using their business capabilities featuring business terms that she is familiar with.

1.2.3 Scenario 3: Variability Management in Business Processes

As part of the portfolio of the organizational business processes of her corporate, Maria found two variants of the same process "*Approving Travel Requests*" that are depicted in Figure 1.4. Maria decided to merge those variants into a single reference process model as in essence both achieve the same goal with slight differences. Having a single model for the same process she guarantees that any future changes in the corporate regulations, are applied in all the variants.

In this particular situation, Maria was able to merge these models and create a unified reference model manually because these processes are relatively simple. However, in the absence of business capabilities in such models, she will be able to create reference models with variation points from a structural perspective only (i.e., branches of the model and the orders between the tasks). Understanding and managing these models will result into challenges similar to Scenario 1.

Furthermore, with more complex business process models such the one shown previously in Figure 1.3, a manual solution can be costly and error-prone. Indeed, La Rosa et al. [123] reported that three analysts spent 130 man-hour to manually create about 25% of a reference model of an insurance process by merging two of its variants. It is obvious that in these cases, an automation support is highly required.



(A) Automatically Approving Travel Requests (adapted from [195])

(B) Manually Approving Travel Requests (adapted from [196])

FIGURE 1.4: Two Business Process Variants from SAP Workflow Scenarios in Travel Management [197]

1.3 Challenges, Objectives and Requirements

The previously mentioned motivational scenarios were validated and adjusted by a domain expert who works as a project manager in a company specialized in the management of seaports in multiple countries. These scenarios reflect real-world situations with concrete challenges: the first challenge reflected in all these scenarios is tackling the description of services and business processes from a business perspective; the second challenge reflected in scenario 1 is the identification of aggregated business capabilities for an entire process or composed services; the third challenge reflected in Scenario 2 is the discovery of services and processes using business capabilities; and the fourth challenge reflected in scenario 3 is tackling the problem of designing reference process models capturing variation points in terms of business capabilities.

For each of the identified challenges, I set the objectives in this thesis and related requirements:

Challenge 1: Description of services and business processes from a business perspective:

The first objective of this thesis is to propose a conceptual model for describing business capabilities with respect to the following requirements:

- *Requirement 1: Expressiveness* - A business capability modelling language should be expressive enough to represent the meaning or the action behind the actual capability. Action's semantics should be explicitly defined and not relying on inferences and analysis of its effect. Furthermore, capabilities should be described independently from their implementations. This requirement was elicited from the following works:Sycara et al. [216], Oaks et al. [159], Semantic Web Services Models: WSMO [242] and OWL-S [232], and Semantic Annotation of Invocation Interfaces Models: SA-WSDL [198, 226] and SA-REST [88].
- *Requirement 2: Inferences* - Given a set of business capability descriptions, additional knowledge can be inferred such as identification of relationships between business capabilities, indexing, etc. Such features can be used to efficiently discover and compose capabilities. This requirement was elicited from the following works: Sycara et al. [216], Oaks et al. [159], and Semantic Web Services Models: WSMO [242] and OWL-S [232].
- *Requirement 3: Use of Ontologies* - A business capability description language should support the use of domain and common ontologies for specifying capabilities [216]. This requirement was elicited from the following works:Sycara et al. [216], Oaks et al. [159], Semantic Web Services Models: WSMO [242] and OWL-S [232], and Semantic Annotation of Invocation Interfaces Models: SA-WSDL [198, 226] and SA-REST [88].

Challenge 2: Identification of aggregated business capabilities:

The second objective of this thesis is to define an algorithm for computing the capability of an entire business process model given that all its tasks are annotated with their business capabilities. For this object, I consider the following requirement:

- *Requirement 4: Rich description* - The complexity of business process models is one of their biggest obstacles: their size and their complex structure make their management difficult for human users. Thus, abstraction and aggregation techniques are required. However, business process abstraction techniques limit the result of an aggregated process model to an abstract task described with a single label [72, 172, 179, 208, 209]. A single label is not sufficient to properly describe a business process [228] that should have a rich description of its business capability in order to give business experts a meaningful description of what a business process achieves [228]. This requirement

was elicited from the following works: Paolucci et al. [166] and Vulcu et al. [228].

Challenge 3: Discovery of services and processes using business capabilities:

The third objective of this thesis is to define a methodology for the discovery of services or business processes using their business capabilities rather than IT capabilities. The discovery solution should respect consider the following requirements:

- *Requirement 5: Ontology based discovery* - Searching business capabilities should rely on concepts from domain ontologies used in the descriptions of business capabilities without relying on keyword extraction from textual descriptions [159]. Relying on extracting key words from unstructured textual descriptions can lead to inconsistent results [2]. This requirement was elicited from the following works: Sycara et al. [216], Oaks et al. [159], and Semantic Web Services Models: WSMO [242] and OWL-S [232].
- *Requirement 6: Time Performance* - Searching for a service or a business process is often relying of reasoning that makes the discovery very slow [153]. A consequent requirement is to propose a solution that is quicker so that it can be adopted in large repositories of services or business processes. This requirement was elicited from the following works: Aznag et al. [9], Srinivasan et al. [211] and Mokhtar et al. [153].

Challenge 4: Designing reference process models capturing variation points in terms of business capabilities:

The fourth objective of this thesis is to define an algorithm for creating reference process models capturing the variation points in terms of business capabilities. The input for this algorithm is a set of process models annotated with business capabilities. The output is a business capability aware reference process model with configuration facilitates, called also configurable process model [184] where configuration options use business capability terms. The requirements of this algorithm are as follows:

- *Requirement 7: Integration of Business Capabilities* - Configurable process models are generally larger than regular models as they integrate multiple variants of the same process [123]. This makes their configuration difficult to handle by business experts [120], thus integrating the business properties in these models makes their configuration easier for these users [120]. This requirement was elicited from the following works: La Rosa [116, 119, 120]

- *Requirement 8: Execution Time* - The creation of configurable process models can be done by merging multiple variants of the same process. Manual creation is time consuming task and thus an automatic merging algorithm should be quick enough to give quick results for business experts. This requirement was elicited from the following works: Gottschalk [89], La Rosa et al. [120] and Assy et al. [7].
- *Requirement 9: Compression Rate* - Merging process models should identify common elements and merge them into a single element [120]. This results into reduced size of input process elements with a high compression rate if there is a high similarity between the variants [90]. This requirement was elicited from the following works: La Rosa et al. [120] and Assy et al. [7].

These challenges, objectives and requirements are summarised in Table 1.1.

TABLE 1.1: Challenges, Objectives and Requirements

Challenges	**Objectives**	**Requirements**
Description of services and business processes from a business perspective	Propose a conceptual model for describing business capabilities	1- Expressiveness 2- Inferences 3- Use of Ontologies
Identification of aggregated business capabilities	Define an algorithm for computing the aggregated capability of an entire business process	4- Rich description
Discovery of services and processes using business capabilities	Define a methodology for the discovery of services or business processes using business capabilities	5- Ontology based discovery 6- Time performance
Designing reference process models capturing variation points in terms of business capabilities	Define an algorithm for creating reference process models capturing the variation points in terms of business capabilities	7- Integration of Business Capabilities 8- Execution time 9- Compression Rate

Limits of the current approaches dealing with the mentioned challenges are discussed in the following sections.

1.4 The Problem of Capability Modelling

In current business process models, the functional perspective (also can be referred to in the literature as capability, functionality or business function) for each process activity is limited to its label [152]. A single label is not enough to describe properly

the capability of a particular process element (i.e., activity, fragment or entire process). Using labels only prevents stakeholders from easily and quickly understanding business processes or identifying the differences and commonalities between them in terms of business properties [151]. When required, stakeholders need to read the business process documentation in order to find out what a process element does, expressed in terms of business properties.

Information Systems' vendors such as IBM, Oracle, or SAP offer together with their solutions the related documentation that is usually (1) extremely large and (2) combines various levels of the technical implementation [99]. For example, searching in SAP ERP documentation requires in depth knowledge of a large and proprietary terminology [99].

Thus, defining capabilities with simple labels or textual descriptions does not fulfil any of the above mentioned requirements: *Expressiveness*, *Inferences* and *Use of Ontologies*.

The literature proposes various capability description approaches as part of efforts for describing related concepts such as business processes, services and search requests (WSMO [183], OWL-S [144], SA-WSDL and SA-REST [111, 125]). They primarily describe capabilities either as part of their implementations (i.e, invocation interface) or as part of other concepts (i.e., services). For all these approaches, the semantics of the action performed by services is derived through reasoning over its inputs, outputs, preconditions and effects (IOPE). The semantics of the action is not explicitly defined and thus the first requirement of *Expressiveness* is not fulfilled, however, they partially fulfil the *Inferences* and *Use of Ontologies* requirements. The main criticism towards these approaches comes from the fact that they mainly focus on modelling IT capabilities rather than business capabilities.

Oaks et al. [159] explores a frame-based modelling approach for describing service capabilities by using natural language constructs such as the action performed and associated parameters (i.e., temporal, location, etc.). Even though the action performed is captured in terms of action verbs, the associated business parameters remain as part of the service inputs, outputs, preconditions and effects (IOPE) which makes the *Expressiveness* requirement partially fulfilled. As the solution proposed by Oaks et al. does not go beyond classical IOPE-based capability descriptions, I consider that it has the same problem of semantic web service solutions and partially fulfils the *Inferences* requirement. Furthermore, this solution relies on language constructs rather than ontological concepts, it partially fulfil the *Use of Ontologies* requirement.

Table 1.2 summarizes contributions related to capability modelling that will be further detailed in Chapter 2. None of the analysed approaches fulfils the three identified requirements. Even though there are attempts to fulfil them, further efforts are required

to enhance these solutions. Therefore, *the first objective of this thesis is to propose a conceptual model for describing the functional perspective of activities (referred to* ***Business Capability****) that respects the three identified requirements: expressiveness, inferences and use-of-ontologies.* The proposed model will be implemented as a set of ontologies that can be used for creating **semantic annotations** of business process models or services (independently from their implementations).

TABLE 1.2: Comparative Analysis of Capability Modelling Approaches

Approach	**Examples**	**Expressiveness**	**Inferences**	**Use-of-Ontologies**
Activity Labels and Textual Descriptions	Major BPM solutions including IBM, SAP and Oracle	**Not Fulfilled**: Labels are not expressive; Textual Description very long and hard to read	**Not fulfilled**	**Not fulfilled**
Semantic Web Services modelling approaches	WSMO [183], OWL-S [144], SA-WSDL and SA-REST [111, 125]	**Not Fulfilled**: Action Semantics needs extensive reasoning	**Partially Fulfilled**: Inferences are used for composition only	**Partially Fulfilled**: The use of ontologies is limited to IOPE parameters
Frame-based Modelling	Oaks et al. [159]	**Partially Fulfilled**: Actions are explicitly captured but other parameters remain as part of IOPEs	**Partially Fulfilled**: Inferences are used for composition only	**Partially Fulfilled**: It uses language constructs, ontology constructs are optional

1.5 The Problem of Aggregation of Business Capabilities

A business process model can detail various elements: activities, data objects, control flow, etc. Therefore, not all the stakeholders are interested in all these details; e.g., the strategic management team is more interested in WHAT is being performed, however, the technical team is interested in HOW tasks are performed. Consequently, there is a lot of effort put towards finding the right details that need to be presented to the involved stakeholders. In this context, Eshuis et al. [72] suggests hiding unwanted process elements while preserving the entire process consistency, whereas Reichert et al. [179] presents an approach that allows for a customized representation of process models with respect to the user preferences, while Smirnov et al. [208, 209] and Polyvyanyy

et al. [172] propose to simply reduce the complexity of process models through abstraction techniques. Business process abstraction techniques consist of aggregating several process elements into a single abstract one [32, 172, 179, 208, 209, 228].

Existing solutions permit the representation of an entire process model at several levels of abstraction. The entire model can even be abstracted into a single activity with a description limited to a single label. Figure 1.2 depicts a process model for the examination procedure of an importation process. Using, for example, the approach proposed by Smirnov et al. [208], this example would be abstracted into one activity that will be presented by a single label (e.g., "Examination of cargo"). It is obvious that a single label does not carry enough information to adequately describe the semantics of the functionality of this entire process model and consequently the *Rich Description* requirement is **not fulfilled**. Business experts can refer to the documentation of this process model to get more detailed information, however, for this simple process with 5 tasks, a documentation of 102 pages is associated that requires a lot of time to read and understand.

Therefore, *the second objective of this thesis is to propose another abstraction technique that allows moving from an entire process model to its functional description by* ***aggregating all the capabilities*** *of the process activities. The resulting aggregated capability should feature functional domain properties, not limited to a single label and not overwhelming the reader with a huge documentation.*

1.6 The Problem of Capability-centric Indexing and Discovery of Services

Existing discovery techniques for semantic web services (WSMO [183], OWL-S [144], SA-WSDL and SA-REST [111, 125]) are difficult as they heavily rely on semantic reasoning [153]. Consequently, a number of researchers have been investigating more lightweight discovery approaches. I classify these contributions under two main categories. The first one is concerned with optimizing the semantic reasoning in order to infer relationships between concepts in ontologies [145, 212, 217]. The second one is concerned by indexing the set of service descriptions in order to reduce the search space and the number of semantic matching operations needed to match service descriptions and search requests [153].

The main idea of the first category consists of performing the reasoning operations during the service publishing phase [145, 212, 217]. When a service is introduced to a service registry, a set of operations are used for pre-computing and storing relevant

information related to concepts used for describing this service. The main issue here is that such an approach cannot be applicable in highly dynamic environments where services are continuously introduced or removed from the service registry. Thus, further optimizations are required to reach a good *Time Performance*.

The second category looks into reducing the matchmaking time by minimizing the search space of service descriptions. This can be achieved by classifying services with respect to concepts used for their descriptions [61, 109, 246, 247], or maintaining a numerical encoding index of services [23, 153]. Solutions from this category perform better than the first one in terms of *Time Performance* but they have to deal with the complexity of maintaining the indexing structure.

Table 1.3 summarizes contributions that looked into indexing and discovering services using their descriptions. These approaches will be further detailed in Chapter 2 with respect to the two requirements: Ontology base Discovery and Time Performance. Building upon the capability model that I propose in this thesis, I further investigate its use for effectively enabling the discovery of interconnected capabilities. My research falls under the second category where I explore the idea of building an indexing structure based on the concepts used for describing capabilities. Specifically, I propose to use Formal Concept Analysis (FCA) [80] for indexing services' capabilities and use this index for their discovery. FCA is a well-known mathematical classification tool used in various domains that allows the organisation of objects described via a set of attributes into a Concept Lattice. The main goal of using FCA is to benefit from its well defined mathematical foundations and predefined indexing and discovery algorithms.

TABLE 1.3: Comparative Analysis of Capability Indexing and Discovery Approaches

Approach	**Ontology Based Discovery**	**Time Performance**
Perform reasoning operations at publishing time [145, 212, 217]	**Fulfilled**: matchmaking operations use reasoning over ontological concepts	**Not fulfilled**: reasoning operations are costly
Indexing and classifying [23, 61, 109, 153, 246, 247]	**Fulfilled**: Indexes are built from ontological concepts	**Partially Fulfilled**: Maintaining indexes is costly and requires tailor-made solutions

Therefore, *the third objective of this thesis is to explore an efficient indexing and discovery of capabilities that are described using the model proposed.* The solution, needs to use ontological concepts for the discovery and performs well in terms of time performance.

1.7 The Problem of Reuse of Business Process Models

In this thesis, *I investigate the (re)use of reference process models and more specifically configurable process models.* Such models have been introduced by Rosemann and van der Aalst [184] with the idea of creating a model that **integrates multiple business process variants** and capturing differences between them via explicit **variation points**. Figure 1.4, shows an example of two business process variants that can be merged into the configurable process models depicted in Figure 1.5. Variation points represent where differences between variants occur and they constitute **configuration decisions** that need to made during the modelling phase in order to derive an **individualized model** (i.e., a process model). In other words, a configurable business process model is a reference model that can be tailored by end-users in order to meet their requirements and satisfy their business needs. Figure 1.5 depicts variation points as thick connectors, the configuration step consists of enabling or disabling incoming or outgoing arcs of these connectors.

Creating configurable process models is a tedious and time consuming task [123]. Thus, few attempts were proposed to semi/automatically create configurable models [50, 91, 123] using mining or merging techniques. The proposed solutions mainly focus on the control flow perspective while ignoring the other business process perspectives.

The second challenge with using configurable models is the configuration step. Indeed, many companies have specialized in developing standard solutions (similar to configurable models) that can be tailored to individual settings. Therefore, they spend a huge amount of time in creating their solutions as well as adapting a particular solution to specific requirements of organizations [41]. In the context of configurable models, the adaptation/configuration consists of a set of operations for **enabling or disabling several branches of the configurable model**. These operations are called **model configurations** [184]. La Rosa [116] admits that identifying **model configurations that reflect domain/business requirements** is a task that can neither be done by **modelling experts** alone nor by **domain (business) experts** alone. Both of them have to meet in order to agree and link model configurations to domain requirements.

Having this **explicit link** between model configurations and domain requirements, La Rosa [116] proposes to guide the configuration step by making domain experts answer simple domain-related questions that are mapped to model configurations and applied on the configurable model. This approach helps guiding the configuration step, however, it relies mainly on these meetings between modelling and domain experts to **continuously update (manually) the links between configuration options and business requirements**. This can be resolved by an **early integration of domain requirements**

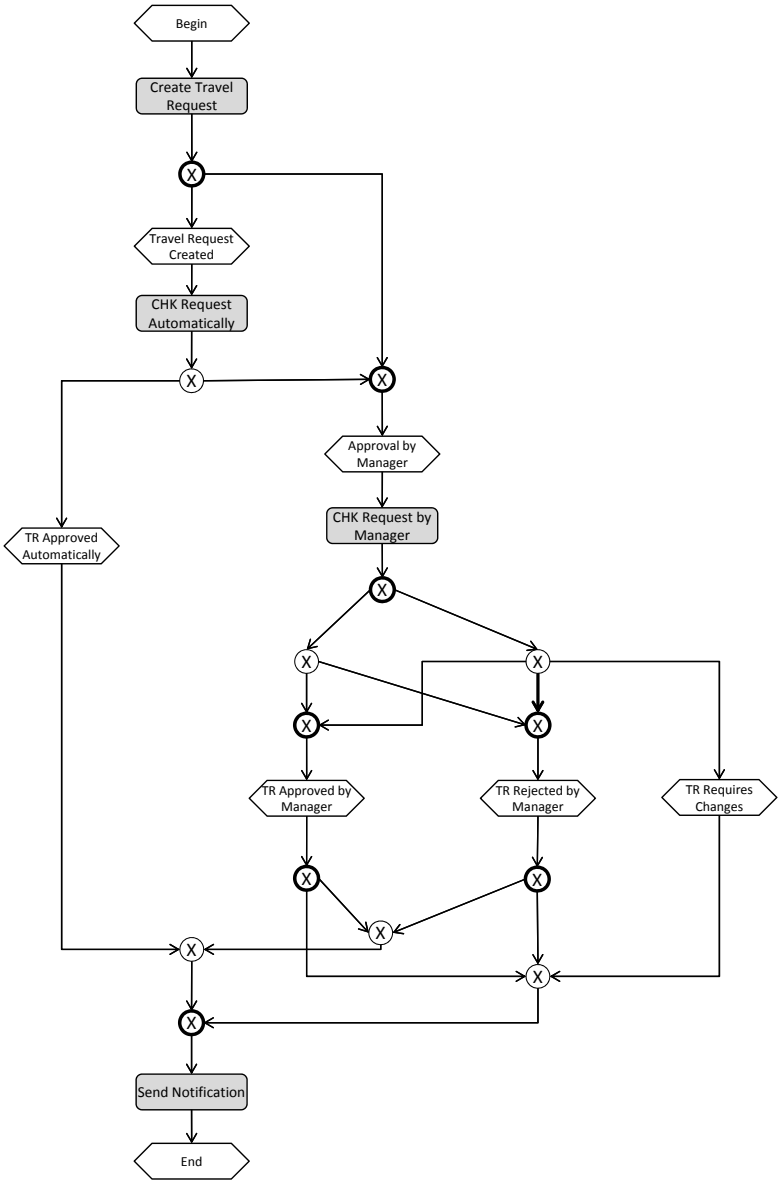


FIGURE 1.5: Configurable Business Process Model generated by Merging the two Models of Figure 1.4.

into business process models. When creating configurable models using this early integration of the business requirements into process models, one can derive configurable models that already have the explicit links between model configurations and domain requirements and consequently move the configuration step from manipulating model configurations to domain requirements.

Furthermore, capability descriptions in configurable process models can also be seen as a rich (and explicit) description of business processes by their properties and more specifically enriched properties at the structural configuration options. This could be considered a significant step forward and certainly supports the configuration step, i.e.

one can achieve incremental improvement of business processes and continued adaptation to demands.

Therefore, *the fourth objective of this thesis is to* ***reduce the business process modelling effort when using configurable process models*** *by (1) proposing an algorithm for creating capability-annotated configurable business process models from a set of capability-annotated business process variants and (2) guiding the configuration step by deriving capability-centric configuration options and applying them on the model.*

1.8 Proposed Approach

In order to tackle the previously mentioned research problems, I propose to use the following approach:

The problem of capability modelling (Section 1.4)

1. *Define a Business Capability Meta-Model*: I define a conceptual model for describing the actions of services and processes in a structured format. I use the term *Business Capability* to refer to this functional perspective [159]. The model should define business capabilities as *standalone entities* independent from their implementations and *feature business terms*. Business Capabilities should be *machine processable*: can be *indexed*, *searched*, and *compared* between each other. Business Capabilities and related concepts should be defined by domain experts (and optionally by modelling experts) and presented in domain-related ontologies (that I call capability domain ontologies) and serialized in a standard format to facilitate their portability.

2. *Implement the model as a set of vocabularies*: I have been particularly interested in investigating the use of semantic web technologies for implementing the model as a set of RDF vocabularies. The choice of the semantic web is motivated by the vision of better enabling *computers* and *people* to work in cooperation [20]. Within this vision and in the context of business process modelling, *people* represent *business experts* that are expressing their needs and requirements in terms of *Business Capabilities* and describing applications and processes from a *functional perspective*.

3. *Integrate business capabilities into process models:* I provide a tool support that allows stakeholders to annotate a particular process model using predefined business capabilities from a *capability domain ontology*. This requires the extension of the chosen business process modelling language serialization to integrate business

capability descriptions. I call the resulting models *Business Capability Annotated Business Process Models.*

4. *Evaluation of the model*: I perform three types of evaluations of the model:

   (a) *Ontological Evaluation*: The ontological evaluation of conceptual models consists of mapping the proposed conceptual model constructs to ontological concepts/constructs in order to assess the ability of the model to represent reality [244]. In this approach, the evaluation of the model is carried out through the verification of a set of rules insuring that a model does not generate any semantic ambiguity by avoiding construct overload and redundancy [231].

   (b) *Feature Comparison Evaluation*: A feature comparison evaluation of conceptual models consists of comparing multiple models and investigate how they represent the same problem based on a set of units of analysis [177]. In Section Section 2.2.1 of Chapter 2 I analyse related approaches using the requirements identified in this thesis as units of analysis. In Chapter 3 I show how the proposed model fulfils those requirements.

   (c) *Interviews with Domain Experts*: I chose to carry out semi-structured interviews [25, 62] with five domain experts that have strong background and are currently active in the area of service computing and information system design and development. The interviews were done after explanation of this thesis objective and details about service modelling approaches. The main targeted outcome of these interviews were to identify if these experts can confirm that the proposed model is good enough to model business capabilities and if it can be adopted in their working environment.

The problem of identifying the business capability of an entire business process model (Section 1.5)

1. *Define an algorithm for Business Capabilities Aggregation*: Given a business capability-annotated business process model, I define an algorithm that computes the capability of the entire business process model by aggregating the capabilities of all its elements. The defined algorithm starts from the initial node and fires all its subsequent nodes one by one until reaching the final node. Each node introduces changes on the aggregated capability. The algorithm assumes that the model is well structured and terminates when it reaches the final node to deliver the aggregated capability. Further details will follow in Chapter 4.

2. *Use of formal semantics*: I propose to use formal semantics of the business capability-annotated process model as a token game, similar to Petri Nets. Petri Nets are

a tool for modelling and studying a system in order to report on its dynamic behaviour [170]. A token is a theoretical concept that is used as an aid to define the behaviour of a process by firing its nodes. The Initial Node generates a token that traverses the sequence flows and passes through the routing nodes until reaching the Final Node [102]. I use the semantics of the various nodes as defined in Petri Nets [170].

3. *Tool Support*: The proposed business capabilities aggregation algorithm is implemented as an extended version of a business process modelling tool. This extension allows users to define the business capability of business process fragment defined by a start and end node of type event.

4. *Interviews with Domain Experts*: The idea of carrying interviews with domain experts consists of using questionnaires to gather their assessment, attitude, opinion, etc. on the proposed research (e.g., [25]). I carried out interviews with five domain experts that have strong background and are currently active in service computing and business process management activities. The interviews were done after explanation of this thesis objective and details about the business capability meta-model and the business process overview approach. The main targeted outcome of these interviews is to identify if these experts see that this research is relevant and its output can be used by their companies.

The problem of capability-centric indexing and discovery of services (Section 1.6)

1. *Capabilities Indexing and Discovery using FCA*: I propose to use *Formal Concept Analysis* (FCA) for indexing and discovering capabilities. FCA is a well-known mathematical classification tool used in various domains that allows organizing objects described via a set of attributes into a *Concept Lattice*. The idea is to start from a set of capabilities described using the proposed *conceptual model* and construct their corresponding concept lattice. The use of such structure reduces the search space for discovering capabilities and helps carry out a step by step search by navigating the lattice. Details about this approach are discussed in Chapter 5.

2. *Implementation and Use Case*: In order to evaluate the applicability of this approach in sensor services modelling, I create a Sensor Capability Ontology that I use in a use case scenario using a set of real world sensors deployed within the Linked Energy Intelligence (LEI) dataspace.

3. *Empirical Evaluation*: In order to evaluate the applicability of the proposed approach in highly dynamic environments, I carry out two experiments highlighting mainly the efficiency of using Formal Concept Analysis in terms of the number of concepts created in a concept lattice given a formal context and the time required to build it.

The problem of reuse of Reference Business Process Models (Section 1.7)

1. *Automatic creation of business capability-annotated configurable process models*: Given a set of business capability-annotated process variants, I define a merging algorithm that generates a capability-annotated configurable business process model. The resulting model should subsume the behaviour of all the input models, explicitly capture variation points in terms of domain requirements (not only model configurations) and allow stakeholders to trace back the origin of each business process element. Model configurations can be driven by the resulting configurable capabilities.

2. *Tool Support*: The proposed merging algorithm is implemented as an extended version of a business process modelling tool. This extension allows users to merge business capability-annotated process models and derive a business capability-annotated configurable process model.

3. *Evaluation*: I perform two types of evaluations of the model:

   (a) *Compression Rate and Time Evaluation*:I have manually created a test collection of business process variants that have been previously used by Gottschalk in his thesis [89]. They were subject of a case study [92] in which techniques for managing configurable process models were extensively tested in a real-world scenario. The process models used in this case study are four processes out of the five most executed registration processes in the civil affairs department of Dutch municipalities [89]. When merging those models, I evaluated the compression rate gained as well as the required execution time.

   (b) *Interviews with Domain Experts*: I carried out a second round of semi-structured interviews with the five domain experts that I interviewed with regard to the business capability aggregation work. These experts have strong background and are currently active in business process management activities. The main targeted outcome of these interviews is to identify if these experts see that the business capability-driven configuration of business process models is useful and can be adopted in their working environment.

1.9 Contributions

The research conducted in this thesis makes contributions to the area of service computing and business process modelling and in particular in the area of business capability-centric indexing and discovery of services and business capability-enabled management of business process models. The contributions brought by this thesis are summarized as follows:

- *A capability meta-model for defining domain capability ontologies (see Chapter 3).* a business capability is defined as an *action category* enriched with domain related features. The meta-model is implemented as a set of RDF vocabularies. Examples of domain capability ontologies are created for use cases and running examples in various chapters in the thesis, such as Import Procedures Capability Ontology (IMPC) in Chapter 4, Sensor Capability Ontology (SCO) in Chapter 5 and Business Travel Capability Ontology (BT) in Chapter 6. In terms of validation, with respect to Bunge's theory of ontology [30], the meta-model can be used for modelling reality by avoiding construct overload and redundancy. In a feature based-evaluation, the model is compared to other contributions proposed in the literature for modelling business capabilities to show that unlike those contributions, the proposed meta-model fulfils the set of identified requirements in Section 1.3. Finally, positive results from interviews with domain experts are noticed with minor issues regarding the implementation choice.

- *An algorithm to automatically generate the capability of an entire capability-annotated business process model (see Chapter 4).* When process models tend to be large, identifying their business capability becomes difficult for end-users. Chapter 4 proposes to compute automatically the aggregated capability of an entire business capability-annotated process model. The algorithm operates by propagating the capability of a start node through all the intermediate nodes until reaching the end of the process model. Each node introduces changes to the aggregated capability. Each transition in the model is formally verified using formal semantics using Petri Nets [170]. A proof of concept is developed and used in interviews with domain experts. The experts gave positive feedbacks on the proposed approach. It has been noticed that this can be used not only for showing the capability of the process model, but can also be further extended to generate a documentation on how the process operates.

- *The validation of the applicability of formal concept analysis for indexing and discovering efficiently capabilities (see Chapter 5).* The frame-based modelling approach proposed in Chapter 3 for modelling capabilities has the advantage of being

flexible enough to reuse exiting contributions that require attribute-based description of objects. To illustrate further this advantage, Chapter 5 explores the use of Formal Concept Analysis for indexing and discovering sensor capabilities. The applicability of this approach is validated via a use case scenario using a set of real world sensors deployed within the Linked Energy Intelligence (LEI) dataspace. LEI is an ecosystem where energy related data is made available and interlinked to support decision making and ultimately energy consumption friendly behaviour [37]. Furthermore, an empirical evaluation is carried out to verify the time performance of the proposed approach by measuring the size of the generated concept lattice for 5000 capabilities and its navigation time (i.e., time required to discover a capability by visiting all the nodes of the lattice).

- *An algorithm for automatically creating a capability-annotated configurable process model by merging a set of capability-annotated business process variants (see Chapter 6).* Early integration of business capabilities in process models allows to create configurable models with configuration options captured in terms of business capability features. The proposed algorithm allows to create such configurable models while guaranteeing that the resulting model subsumes the behaviour of all input models, each element of the model can be traced back to its origin and permits to generate input models as well as new ones. A proof of concept is designed to carry out empirical evaluations by measuring the time required to merge models as well as the compression rate gained. The configurable model is created in few milliseconds and reaching a compression rate of around 50% in terms of space required for saving the input models. Furthermore, I carried out interviews with domain experts revealing that the major challenge for business capability-driven configuration modelling to join industry is the fact that current industrial solutions are mature enough and hard to replace. Current solutions have been built over years of analysis, engineering and research that are proven to be effective. Replacing these solutions has never been taken as a serious option. However, features such as those proposed in this thesis can be seen as additional options to the current systems but a lot of adaptation work is required.

Resources that can be useful for the community in terms of tools, vocabularies and test collections have been developed in this thesis:

- A new version of a business process modeling tool [2] that includes the implementation of: (i) an annotation tool for integrating capability descriptions in business

[2]Original version of EPCTools: http://www2.cs.uni-paderborn.de/cs/kindler/Forschung/EPCTools/ as accessed on the 03/10/2013.
New version of EPCTools available at: http://wassimderguech.org/phd/

process models with respect to the proposed meta-model in Chapter 3, (ii) the capabilities aggregation algorithm proposed in Chapter 4 and (iii) the merging algorithm proposed in Chapter 6 for creating configurable process models.

- A set of RDF vocabularies [3] used in defining the capability meta-model introduced in Chapter 3.
- A test collection of business process models for municipalities, available in Appendix E.
- A test collection of business process models from the customs clearance domain, available in Appendix A.

1.10 Thesis Structure

Figure 1.6 sketches the structure of the rest of the chapters. First of all, **Chapter 2** reviews current research contributions related to capability modelling and discovery as well as capability-enabled management of business process models. It helps position the research in this thesis and highlight its contributions.

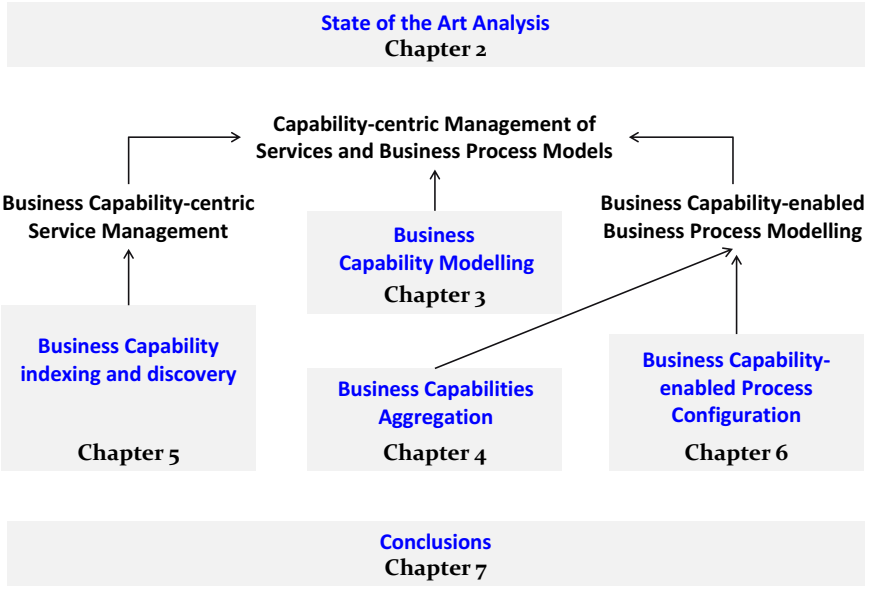


FIGURE 1.6: Thesis Structure

Chapter 3 deals with the problem of describing capabilities. It introduces formally the meta-model that I propose for designing domain capability ontologies.

Chapter 4 introduces the capabilities aggregation algorithm and its validation.

[3] These vocabularies are made publicly available and will be introduced in Chapter 3

Chapter 5 builds upon the capability model proposed in Chapter 3 for indexing and discovering capabilities using formal concept analysis.

Chapter 6 discusses an algorithm that takes as input a set of capability-annotated business process variants, merges them and provides as output a capability-annotated configurable business process model. The proposed algorithm is implemented and tested on a set of real world process models that are detailed in this chapter as well.

Chapter 7 concludes the thesis by summarizing its contributions and suggests future research directions.

1.11 Publications

The research carried out within this thesis led to the following set of publications grouped by topic:

On the modelling of capabilities

- Wassim Derguech and Sami Bhiri. Modelling, interlinking and discovering capabilities. In *ACS International Conference on Computer Systems and Applications, AICCSA 2013, Ifrane, Morocco, May 27-30, 2013*, pages 1–8, 2013
- Sami Bhiri, Wassim Derguech, and Maciej Zaremba. Web service capability meta model. In Karl-Heinz Krempels and José Cordeiro, editors, *WEBIST 2012 - Proceedings of the 8th International Conference on Web Information Systems and Technologies, Porto, Portugal, 18 - 21 April, 2012*, pages 47–57. SciTePress, 2012
- Sami Bhiri, Wassim Derguech, and Maciej Zaremba. Modelling Capabilities as Attribute-Featured Entities. In José Cordeiro and Karl-Heinz Krempels, editors, *Web Information Systems and Technologies - 8th International Conference, WEBIST 2012, Porto, Portugal, April 18-21, 2012, Revised Selected Papers*, volume 140 of *Lecture Notes in Business Information Processing*, pages 70–85. Springer, 2012

On the aggregation of capabilities

- Wassim Derguech and Sami Bhiri. Business Process Model Overview: Determining the Capability of a Process Model Using Ontologies. In Witold Abramowicz, editor, *Business Information Systems - 16th International Conference, BIS 2013, Poznań, Poland, June 19-21, 2013. Proceedings*, volume 157 of *Lecture Notes in Business Information Processing*, pages 62–74. Springer, 2013

On the discovery of capabilities

- Wassim Derguech, Sami Bhiri, Souleiman Hasan, and Edward Curry. Using Formal Concept Analysis for Organizing and Discovering Sensor Capabilities. *The Computer Journal*, 58(3):356–367, 2015
- Wassim Derguech, Souleiman Hasan, Sami Bhiri, and Edward Curry. Organizing Capabilities Using Formal Concept Analysis. In Sumitra Reddy and Mohamed Jmaiel, editors, *2013 Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, Hammamet, Tunisia, June 17-20, 2013*, pages 260–265. IEEE, 2013

On the design of configurable business process models

- Wassim Derguech and Sami Bhiri. Merging Business Process Variants. In Witold Abramowicz, editor, *Business Information Systems - 14th International Conference, BIS 2011, Poznan, Poland, June 15-17, 2011. Proceedings*, volume 87 of *Lecture Notes in Business Information Processing*, pages 86–97. Springer, 2011
- Wassim Derguech and Sami Bhiri. An Automation Support for Creating Configurable Process Models. In Athman Bouguettaya, Manfred Hauswirth, and Ling Liu, editors, *Web Information System Engineering - WISE 2011 - 12th International Conference, Sydney, Australia, October 13-14, 2011. Proceedings*, volume 6997 of *Lecture Notes in Computer Science*, pages 199–212. Springer, 2011

Case studies and applications

- Sana Baccar, Wassim Derguech, Edward Curry, and Mohamed Abid. Modeling and Querying Sensor Services Using Ontologies. In Witold Abramowicz, editor, *Business Information Systems - 18th International Conference, BIS 2015, Poznań, Poland, June 24-26, 2015, Proceedings*, volume 208 of *Lecture Notes in Business Information Processing*, pages 90–101. Springer, 2015
- Wassim Derguech and Sami Bhiri. Capability Modelling - Case of Logistics Capabilities. In Marcello La Rosa and Pnina Soffer, editors, *Business Process Management Workshops - BPM 2012 International Workshops, Tallinn, Estonia, September 3, 2012. Revised Papers*, volume 132 of *Lecture Notes in Business Information Processing*, pages 519–529. Springer, 2012

- Feng Gao and Wassim Derguech. Ubiquitous Service Capability Modeling and Similarity Based Searching. In *Web Information Systems Engineering - WISE 2011 and 2012 Workshops - Combined WISE 2011 and WISE 2012 Workshops, Sydney Australia, October 12-14, 2011 and Paphos, Cyprus, November 28-30, 2012, Revised Selected Papers*, pages 173–184, 2012

- Wassim Derguech, Feng Gao, and Sami Bhiri. Configurable Process Models for Logistics Case Study for Customs Clearance Processes. In Florian Daniel, Kamel Barkaoui, and Schahram Dustdar, editors, *Business Process Management Workshops - BPM 2011 International Workshops, Clermont-Ferrand, France, August 29, 2011, Revised Selected Papers, Part II*, volume 100 of *Lecture Notes in Business Information Processing*, pages 119–130. Springer, 2011

Other related publications

- Wassim Derguech. Towards a framework for business process models reuse. In Anne Persson, Boualem Benatallah, and Adnene Guabtni, editors, *Proceedings of the CAiSE Doctoral Consortium 2010*, volume 593, Hammamet, Tunisia, 2010. ceur-ws.org

- Wassim Derguech, Gabriela Vulcu, and Sami Bhiri. An Indexing Structure for Maintaining Configurable Process Models. In Ilia Bider, Terry A. Halpin, John Krogstie, Selmin Nurcan, Erik Proper, Rainer Schmidt, and Roland Ukor, editors, *BMMDS/EMMSAD*, volume 50 of *Lecture Notes in Business Information Processing*, pages 157–168. Springer, 2010. ISBN 978-3-642-13050-2

- Gabriela Vulcu, Sami Bhiri, Wassim Derguech, and María José Ibáñez. Semantically-enabled Business Process Models Discovery. *International Journal of Business Process Integration and Management*, 5:257–272, 2011

- Wassim Derguech and Sami Bhiri. Reuse-Oriented Business Process Modelling Based on a Hierarchical Structure. In Michael zur Muehlen and Jianwen Su, editors, *Business Process Management Workshops*, volume 66 of *Lecture Notes in Business Information Processing*, pages 301–313. Springer, 2010. ISBN 978-3-642-20510-1

Chapter 2

State of the Art Analysis

"If I have seen further it is by standing on the shoulders of giants."

Isaac Newton

This chapter constitutes the analysis of research works related to the main contributions of this thesis. It starts in Section 2.2 with a discussion of the contributions related to **capability modelling**. Followed by Section 2.3 that looks into contributions used for **determining the capability of a business process model**. Then Section 2.4 analyses research works proposed for **indexing and discovering capabilities**. And Section 2.5 analyses works related to capability-driven reuse of business process models. Finally, the Chapter is concluded in Section 2.6.

For each of these sections, we followed a methodology similar to the one proposed by Recker and Mendling [178]. It consists of identifying relevant research contributions then classifying them with respect to certain coding categories driven by the proposed technique or topic. Each of the contributions is then analysed individually and then a conclusion in drawn for each category.

2.1 Basic Concepts

The main concept used in this thesis is the *business capability*. This concept has been defined in the literature from various perspectives:

- *From an organizational and resource perspective: Organisational Capability*: the ability of organizations to efficiently use their resources (i.e, human capital, knowledge, available data, etc.) to generate value and achieve their objectives [3, 68].

- *From a control flow perspective: Planning Capability* : the way organizations achieve their goals by capturing explicitly process tasks and their temporal and logical order [238].
- *From a service perspective: IT Capability*: the effect of a service in terms of data generated or change of the world [160] that are explicitly represented in terms of Inputs, Outputs, Preconditions and Effects (IOPE for short).
- *From a functional perspective: Business Capability*: the action performed by a service, computer program, etc. that creates a value for the customers [159].

In this thesis, I consider a the business capability from a functional perspective. I argue that this concept is highly required for describing what is being achieved by enterprise services and business processes. In this context, I adopt the following definitions for these related concepts:

- *A business process*: is something that businesses go through every day in order to accomplish their mission [26]. They can be either primary processes (e.g., production processes, marketing, customer support) or support processes (e.g., travel request approval, HR processes such as payment of salaries, etc.).
- *A business process model*: a set of ordered activities and tasks that, once completed, achieve the organisation's goal [238]. Such models are an explicit representation of business processes. Usually captured as graphs with nodes representing activities, events, resources, etc. and edges for capturing the temporal and causal order.
- *Activity*: is a piece of work forming a single step within a business process [165]. Other synonyms for this concept used in the literature include: Step, Node, Task, Work Element, Function Item.

The object of this thesis is to propose the management of services and processes using their business capabilities. By management operations I refer to:

- *Annotation*: is the enrichment of current assets with additional information. In this thesis I use annotation for enriching activities, services and process models with their business capabilitities.
- *Indexing*: is the operation of organising a repository of items to optimise their discovery. I this thesis, the items to consider are business capabilities.

- *Aggregation*: is the grouping of multiple items into a single and more general item. This can apply to aggregating business process activities to abstract ones. Therefore, in this thesis, I investigate the identification of the business capability of aggregated activities.
- *Configuration*: is set of customisation steps that allow to move from a general item to a more specific one. In this thesis, configuration is defined as a phase for the customisation of reference process models. The configuration is done by manipulating certain parameters that are captured in terms of business capability properties.

2.2 Capability Modelling

A capability denotes what an action does either in terms of world effects or returned information [160]. The purpose of providing well defined capabilities of services or business processes is to allow end users to discover them with respect to the action they perform. This section examines service description languages and how they describe capabilities of services. I classify the contributions found in the literate in three families: Semantic Web Services models, Semantic Annotation of Invocation Interfaces models and Frame-based models. These three families are discussed in details using the units of analysis defined in Section 2.2.1.

2.2.1 Requirements as Units of Analysis

This section recalls the set of requirements for business capability modelling identified in Chapter 1. These requirements help in the analysis of the state of the art and identify the research gap that this thesis contributes to reduce. Recall:

- *Requirement 1: Expressiveness* - A business capability modelling language should be expressive enough to represent the meaning or the action behind the actual capability. Action's semantics should be explicitly defined and not relying on inferences and analysis of its effect. Furthermore, capabilities should be described independently from their implementations. This requirement was elicited from the following works: Sycara et al. [216], Oaks et al. [159], Semantic Web Services Models: WSMO [242] and OWL-S [232], and Semantic Annotation of Invocation Interfaces Models: SA-WSDL [198, 226] and SA-REST [88].
- *Requirement 2: Inferences* - Given a set of business capability descriptions, additional knowledge can be inferred such as identification of relationships between

business capabilities, indexing, etc. Such features can be used to efficiently discover and compose capabilities. This requirement was elicited from the following works: Sycara et al. [216], Oaks et al. [159], and Semantic Web Services Models: WSMO [242] and OWL-S [232].

- *Requirement 3: Use of Ontologies* - Describing business capabilities requires sharing of common understanding of the structure of this information and enable the reuse of its constructs among the involved stakeholders. In such context, the use of ontologies is key enablers. A business capability description language should support the use of domain and common ontologies for specifying capabilities [216]. This requirement was elicited from the following works:Sycara et al. [216], Oaks et al. [159], Semantic Web Services Models: WSMO [242] and OWL-S [232], and Semantic Annotation of Invocation Interfaces Models: SA-WSDL [198, 226] and SA-REST [88].

In addition to these requirements, I need to include another important aspect specific to my research: the creation of *configurable* capabilities. A configurable capability is an integrated representation of various capabilities that has explicit configuration options. These options allow to define new capabilities by choosing relevant ones.

In the following, I further refine these high level requirements and consider this list:

Expressiveness: Explicitly represent the action performed.

Expressiveness: Explicitly capturing functional and non-functional features related to the action performed.

Expressiveness: Ability to express these features using simple (e.g., integer, boolean, string) as well as complex types (e.g., conditional values, enumerations).

Inferences: Ability to explicitly identify relationships between capabilities based on their descriptions.

Use of Ontologies: Use of domain or general ontological concepts for describing business capabilities.

Configuration: Ability to describe configurable capabilities.

2.2.2 Semantic Web Services Models

Description

The first family of contributions for capability descriptions includes Semantic Web Services models (WSMO [242] and OWL-S [232]). Capability descriptions found in this family are split into information transformation and state of the world change captured as Input, Output, Preconditions and Effects (IOPE paradigm) [181].

WSMO stands for the Web Service Modeling Ontology that has been proposed as a meta-model for describing aspects related to semantic web services [242]. This model has emerged from the idea of integrating semantic web technologies and web services. It has been developed with respect to a set of design principles including: web compliance, ontology-use, strict decoupling, etc. In the following, I briefly describe the three top level WSMO model elements that are relevant to the context of this thesis:

- **Ontologies:** captures the formal descriptions of the information model of WSMO. The use of ontologies brings two main features: (1) shared conceptualisation and (2) formal semantics that are defined by the Web Service Modelling Language (WSML) [45].
- **Web Services:** are described via the functional *capability* they give access to as well as the required interface(s). WSMO describes the functional capability using (1) preconditions and assumptions that define that state of the world before execution and (2) postconditions and effects that define the state of the world after the execution. For example, in order to process a delivery, a precondition is that a valid zip code is available in the address and an assumption is that the address is correct.
- **Goals:** refer to the objectives that a requester wants to achieve with a certain service. WSMO goals are defined in terms of the required information and the change to the state of the following the execution of a service.

OWL-S is an ontology of service built on top of Web Ontology Language (OWL)[200]. Its aim is similar to any proposed structured languages: enhance automated, discovery, invocation and composition. A service in OWL-S is defined via the three top level concepts: ServiceProfile, ServiceGrounding and ServiceModel. In this thesis, the main concept of interest is the ServiceProfile.

- **ServiceProfile:** defines the functional *capability* of the service using human understandable parameters such as *serviceName* and *textDescription* and machine

processable parameters such as *hasInput*, *hasOutput*, *hasPrecondition* and *hasResult*. Other attributes are used for defining other aspects of the service but these details are outside of the scope of this work. The main feature to note here is that OWL-S and WSDL are very similar.

Critique

Both OWL-S and WSDL were designed at a time when extensive service descriptions were thought necessary to build a web service architecture. The major problem with these languages is that they have been extensively enriched making the description of the entire service in some cases complex [104]. Indeed, WSMO and OWL-S have been extended to various versions to meet multiple requirements: WSML-Core, WSML-DL, WSML-Flight, WSML-Rule and WSML-Full for WSMO and OWL Lite, OWL DL and OWL Full for OWL-S. Even though these extensions helped WSMO and OWL-S reach a high level of expressiveness for more sophisticated reasoning, they led to a more difficult descriptions for end-users and a more costly computational reasoning. Furthermore, describing what services would do upon the change of state of the world after its execution proved to be a much harder problem than the developers of WSMO and OWL-S anticipated [104]. Using this analysis in the context of this thesis, in the following, I further analyse these contributions with respect to the predefined requirements:

Expressiveness: Explicitly represent the action performed: partially fulfilled. Information transformation and state of the world changes are expressed in terms of axioms, consequently the explicit action performed is not captured. However, in OWL-S Profiles, a classification in a service taxonomy such as North American Industry Classification System (NAICS) [161] or United Nations Standard Products and Services Code (UNSPSC) [94] can be used to help identify the actual action being performed.

Expressiveness: Explicitly capturing functional and non-functional features:: partially fulfilled. Modelling capabilities as IOPEs does not feature in an explicit and easily accessible way domain features. Extracting and managing domain attributes requires some reasoning which can be time consuming and difficult to manage by end-users [104].

Expressiveness: Ability to express features using simple and complex types: fulfilled. Both WSMO and OWL-S have extensions to allow for describing complex types.

Inferences: Ability to explicitly identify relationships between capabilities: partially fulfilled. Even though in their specification documentation it is mentioned that both languages allow for explicitly modelling relations between service descriptions, I could not find any contributions that investigate this feature and its use remains obscure as argued by Kamaruddin et al. [104].

emphUse of Ontologies: Use of domain or general ontologies: fulfilled. Both languages use domain or general ontological concepts for describing capabilities.

Configuration: Ability to describe configurable capabilities: not fulfilled. There are no possibilities to create configurable capabilities, extensions of the languages are required.

2.2.3 Semantic Annotation of Invocation Interfaces Models

Description

The second family of related efforts concerns semantic annotations of invocation interfaces (SA-WSDL [198, 226] and SA-REST [88]). While these approaches do not directly target capability modelling, they attempt to provide alternative solutions to top-down semantic approaches (WSMO [242] and OWL-S [232]) by starting from existing descriptions such as WSDL [35] and annotating them with semantic information.

SA-WSDL stands for the Semantic Annotations for WSDL [198, 226] and is proposed as a simplified version of WSDL-S. Its idea is to start from existing service descriptions defined in WSDL and integrate some semantic annotations using new extensions. These extensions are confined to attributes of *modelReference* and two specializations of schemaMapping namely, *liftingSchemaMapping* and *loweringSchemaMapping*. The modelReference attribute can be used to annotate XSD complex type definitions, simple type definitions, element declarations, and attribute declarations as well as WSDL interfaces, operations, and faults. The liftingSchemaMapping can be applied to XML Schema element declaration, complexType definitions and simpleType definitions. While in WSDL-S the precondition and effect annotations where explicitly annotating a WSDL operation, in SA-WSDL the *modelReference* is proposed to define implicitly these preconditions and effects.

SA-REST stands for Semantic Annotation of Web Resources [88, 204]. SA-REST is designed to add further meta-data to REST API descriptions in HTML or XHTML. It gives the possibility to integrate in its annotations multiple sources of meta-data (i.e., ontologies, taxonomies, etc.). While this language is not exclusively intended to be used

for describing web services defined in standard web services languages such as WSDL, it provides a simple annotation mechanism using RDFa that can be used for this purpose [204].

Critique

It is clear that these contributions focused mainly on annotating the service interfaces rather than providing a comprehensive business *capabilities*. This is mainly perceived in the fact that there was no clear decisions regarding the attributes to be used in the *modelReference*. The researchers that worked on these contributions could have taken the decision to use an RDFS/OWL model that defines terms like "category", "precondition", "effect" etc. that would allow model references to be typed in a standard way. I carry out a further analysis of these contributions with respect to the predefined requirements:

Expressiveness: Explicitly represent the action performed: partially fulfilled. The specification of SAWSDL indicates a possible use of the interface *modelReference* for categorization [144] that might help introducing a natural language indication of the action being done by the proposed service.

Expressiveness: Explicitly capturing functional and non-functional features: not fulfilled. These approaches define a semantic description of syntactic interaction interfaces rather than concrete capabilities and thus do not explicitly capture domain features.

Expressiveness: Ability to express features using simple and complex types: fulfilled. Semantic annotations allow for describing complex types.

Inferences: Ability to explicitly identify relationships between capabilities: not fulfilled. Relations between interfaces descriptions are mainly used to determine potential interactions that can be used for composition.

Use of Ontologies: Use of domain or general ontologies: fulfilled. Both languages use domain or general ontological concepts for describing interaction interfaces but they lack some flexibility in choosing ontologies. Indeed, as stated by Lefort et al. [128], further work is required to upgrade SA-WSDL so that it can also let the end user select the service ontology they want if they are not satisfied by the definitions brought by the SA-REST or WSMO-Lite ontologies.

Configuration: Ability to describe configurable capabilities: not fulfilled. There are no possibilities to create configurable capabilities, extensions of the languages are required.

2.2.4 Frame-based Models

Description

The third family includes frame-based approaches for modelling capabilities. This is another way to describe capabilities featuring *functional declarations* that are different from the classical IOPEs. Functional declarations are investigated in details by researchers from the linguistics and natural language processing domain with the aim to give another view on the structure of sentences by describing verbs using "cases" contained in case frames [76]. Example of cases include: agent (who), location (where) and instrument (how) as declared by Fillmore [76].

The idea of modelling capabilities using frames has been used to describe the capabilities of software agents [239] and proves to be effective for enhancing agents communication and planning while facilitating human understanding of agents capabilities. In the same vision, Oaks et al. [159] used frame-based modelling for describing service capabilities. Oaks et al. proposed a comprehensive conceptual model that extends the IOPE paradigm with additional frames extracted from textual descriptions. Frames used by Oaks et al. are similar to those defined by Fillmore. It makes the model easy for humans to read and understand but machines won't be able to use this model to compose capabilities. Composition is still relying on the classical IOPE approach.

Critique

Frame-based modelling is suitable for human understanding. This was one of the main motivations for exploring this approach in agent-based systems and services descriptions [159, 239]. However, these contributions did not consider exploring relations between agent of service capabilities as they are mainly extending the classical IOPE paradigm. This analysis of this approach with respect to the identified requirements previously defined is as follows:

> *Expressiveness: Explicitly represent the action performed*: fulfilled. The model proposed by Oaks et al. [159] distinguished in particular the corresponding action verb of the capability description.
>
> *Expressiveness: Explicitly capturing functional and non-functional features*: not fulfilled. In addition to the classical IOPE paradigm, capabilities are described with an action verb and *informational attributes* (called roles in the paper [159]). These attributes are neither explicitly capturing functional and non-functional features of the action nor capturing domain-related properties.

Expressiveness: Ability to express features using simple and complex types: fulfilled. The proposed model is rich enough to model both simple and complex types.

Inferences: Ability to explicitly identify relationships between capabilities: partially fulfilled. Most of the elements in the model are defined in an information source such as WordNet (i.e., lexical-based descriptions). This allows the explicit declaration of relations between them such as synonymy, equivalence, etc. However, the authors do not investigate further this feature and do not show how these relations can be derived.

Use of Ontologies: Use of domain or general ontologies: partially fulfilled-:The proposed model allows using domain or general ontological concepts for describing capabilities, but this has not been explicitly shown in the proposed work. More efforts are put towards using lexical-based descriptions of capabilities.

Configuration: Ability to describe configurable capabilities: not fulfilled. There are no possibilities to create configurable capabilities, extensions of the model are required.

2.2.5 Summary and Discussion

This section reviewed related approaches that proposed service description models that can be used as alternatives/extensions to either simple labels and textual descriptions or to existing languages such as WSDL [35]. A summary of these approaches is available in Table 2.1. While all of the proposed approaches were mainly focusing on the use of ontological concepts in their descriptions and do not rely on keyword extraction during the discovery process, they either fail or partially fail in fulfilling the other requirements.

All the proposed approaches are reliable for carrying out machine processing operations such as composition and discovery. These solutions were proposed to avoid relying on simple labels or long textual descriptions in these operations. However, most of the proposed approaches do not go beyond the classical IOPEs. This requires search requests to define the state of the world before and after the execution of a service, something that has proven to be difficult [245] and requires additional abstraction efforts to make end users able to query services in a more user friendly manner [245].

The key points of this analysis are:

- **Explicit actions** even using simple lexical terms form a good basis for a capability description [159, 239]. This is the natural way human users define what a service

or application does [76]. Capturing these actions in a domain specific ontology helps improve their reuse by creating a common understanding on their semantics.

- Capability description models should be **open** to allow for more **flexibility** to include other ontological concepts and the way end-users adopt to describe their assets. This alleviates the need to to create agreements with all possible services stakeholders (including brokers and clients) [226]. A good example in here is the quick and high adoption of JavaScript Object Notation (JSON) as a lightweight format for exchanging and modeling structured data without strict restrictions on what attributes to use or any particular order that they should follow, etc [157].
- Enriching the action performed with explicit **functional and non-functional features** does not only refine further the action being carried out but also can be used to infer relations between capabilities [159]. These relations can create an indexing structure that is not exclusively built on the categorisation schema of lexical terms [153].
- There are no contributions investigating the description of configurable business capabilities.

2.3 Business Capability of a Business Process: Business Capabilities Aggregation

The previous section primarily focused on business capability modelling in the context of service computing. Most of the reviewed approaches were considering atomic services and consequently atomic capabilities. In an enterprise environment composed services and business processes also constitute valuable assets. In this section I analyse research contributions that compute the capability of an entire business process. I call this operation *capabilities aggregation*.

Business process models are central artifacts in Process Aware Information Systems. These models are being managed and maintained by several stakeholders with various needs. While a business process engineer is interested in a detailed business process with all its options and with the integration of all available enterprise views (e.g., organisational, functional, data, etc.), other stakeholders are interested in less detailed views over the entire business process model. For example an operations manager is more interested in the functional view, a data analyst is interested in the data flow and how data is controlled over the various steps of the process, etc. In this context, techniques for Business Process Model Abstraction (BPMA for short) can be applied to have a

TABLE 2.1: Comparative Analysis of Capability Modelling Approaches

Approach	**Expressiveness**			**Inferences**	**Use-of-Ontologies**	**Configuration**
	Explicitly represent the action performed	**Explicitly capturing functional and non-functional features**	**Express features using simple and complex types**	**Identify relationships between capabilities**	**Use of domain or general ontologies**	**Describe configurable capabilities**
Semantic Web Services Models: WSMO [242] and OWL-S [232]	**Partially fulfilled** as in OWL-S Profiles one can use categories of services using taxonomies such as the [161] or UNSPSC [94] (but remains not explicit action description)	**Partially fulfilled** as IOPEs do not feature in an explicit and easily accessible way domain features . Additional effort towards the extraction of these features is required e.g., [104]	**Fulfilled** as both WSMO and OWL-S have extensions to allow for describing complex types	**Partially fulfilled** as both languages claim to have support for creating relations between services, but I could not find any work that used this feature. Both languages are proposed to enable automation of discovery.	**Fulfilled** as both languages use domain or general ontological concepts	**Not fulfilled**
Semantic Annotation of Invocation Interfaces Models: SA-WSDL [198, 226] and SA-REST [88]	**Partially fulfilled** as the *modelReference* in SAWSDL can optionally be used for categorization [144]	**Not fulfilled** as these approaches describe interaction interfaces rather than concrete capabilities	**Fulfilled** as semantic annotations allow for describing complex types	**Not fulfilled** as relations between interfaces descriptions are used to determine potential interactions that can be used for composition	**Fulfilled** as both languages use domain or general ontological concepts	**Not fulfilled**
Frame-based Models: Oaks et al. [159]	**Fulfilled** as the model proposed by Oaks et al. [159] distinguished the action verb of the capability	**Not fulfilled** as the model simply adds to the classical IOPE an action verb and *informational attributes* that are neither explicitly capturing functional and non-functional features nor capturing domain-related properties	**Fulfilled** as the proposed model is rich enough to model both simple and complex types	**Partially fulfilled** as the model proposes to use relations between action verbs in terms of synonymy, equivalence, etc. But this has not been validated/tested	**Partially fulfilled** as the model allows using domain or general ontological concepts however more efforts are put towards using lexical-based terms	**Not fulfilled**

quick view of the essential elements of process models depending on the required level of detail that is implemented via two operations: elimination and aggregation [210].

Four categories of BPMA are analysed with respect to the proposed **technique** and the **structure of aggregated capability**. Wthin the context of this work, I keep referring to *Requirement 4: Rich description* from Chapter 1.

Recall:

> *Requirement 4: Rich description - Business process abstraction techniques limit the result of an aggregated process model to an abstract task described with a single label [72, 172, 179, 208, 209]. A single label is not sufficient to properly describe a business process [228] that should have a rich description of its business capability in order to give business experts a meaningful description of what a business process achieves [228]. This requirement was elicited from the following works: Paolucci et al. [166] and Vulcu et al. [228].*

2.3.1 Elimination of Activities

Description

Reichert et al. propose Proviado Framework (depicted in Figure 2.1 for visualising large process models in a reduced format depending on the stakeholders' needs and profile. The proposed architecture depicted in Figure 2.1 requires a visualisation model that captures the elements to keep, hide, etc (b). Structural and notation preferences are then mapped (c,d,e) to the generic process model (a) leading to a customised and personalised visualisation (f).

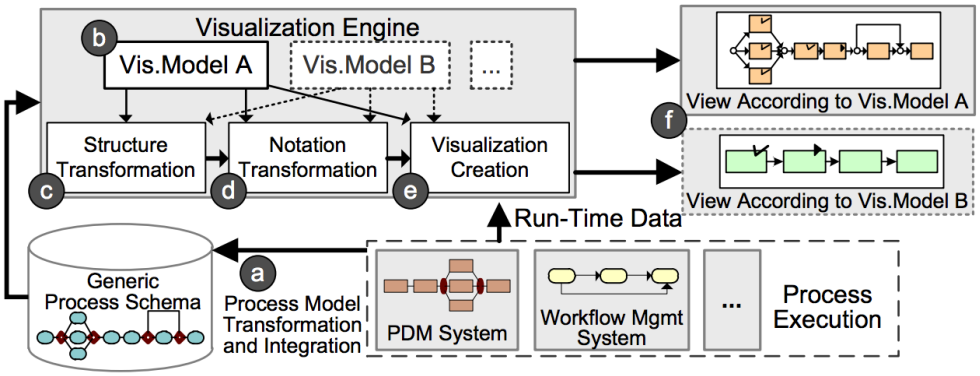


FIGURE 2.1: Proviado Framework [179]

The structural transformations can be implemented through elimination techniques by omitting unwanted model elements [210] without necessarily altering the behaviour of

the original model. Reichert et al. [179] call such techniques "schema reduction". For example, the reduction operation `RedActivity` (see Figure 2.2.b) is done by removing an activity and its incoming and outgoing arcs. Then a new arc between the previous and following nodes of the removed activity is added. As shown in Figure 2.2.a, if multiple activities need to be removed, a decomposition of the reduction operation is necessary to remove one activity at a time. For reducing conditional branching (Or/Xor branching) advanced operations are provided. The reduction algorithm also removes automatically duplicate arcs, connectors that do not imply any routing information and consecutive connectors as shown respectively in Figure 2.2.c, 2.2.d and 2.2.e.

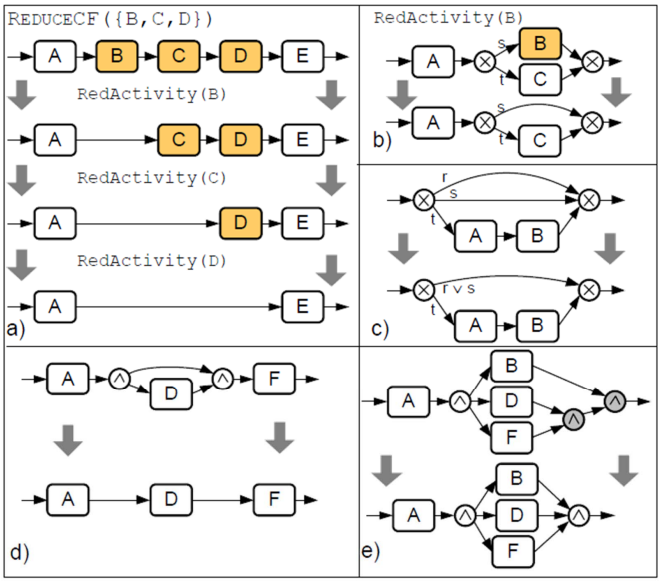


FIGURE 2.2: Business Process View based on Schema Reduction [179]

Critique

From a technical perspective, the proposed framework assumes a complete and correct visualisation model that is used by the visualisation engine to carry out the transformation operations. This is useful and valid assumption to automate the transformation operations, however, if a new change occurs on the process model, the corresponding visualisation models for each stakeholder needs to be updated. Authors did not report on how to handle the propagation of changes from process models to visualisation models.

The applied reduction technique operates by removing business process nodes including activities without creating any aggregated elements (e.g., abstract process node or task). Consequently, it does not generate any aggregated capabilities or labels. Even though a lot of effort was put towards preserving the structure of the resulting model view [179, 210], reductions of activities always comes with an information loss, namely the

business capabilities of the entire business process. Consequently, *Requirement 4: Rich description* is not fulfilled.

2.3.2 Aggregation using Structural Patterns

Description

As part of Proviado framework depicted in Figure 2.1, in addition to the schema reduction technique, Reichert et al. [179] introduce the "schema aggregation" technique that is widely used in the literature. For example Eshuis et al. [72] use an aggregation technique for controlling the public visibility of business process models without revealing some private activities. The object is to create from an entire business process model a public view that can be exchanged with other collaborators without necessarily revealing sensitive process parts. The proposed technique consists of asking the process provider to select the elements that he wants to be part of the public view. The other activities are hidden by aggregating them into single elements. Eshuis et al. focused mainly on ensuring that unrevealed process parts remain private, thus, they did not capture explicit structural reduction patterns. Furthermore, they did not discuss how the aggregated activities' capability or even label is generated. I assume that the resulting activity label is manually entered by the process provider after execution of the aggregation algorithm.

Critique

Polyvyanyy et al. [172] use the same technique for reducing the complexity of large EPCs. While Reichert et al. [179] defines model-specific aggregation operations that are part of the visualisation model, Polyvyanyy et al. [172] explicitly detail generic structural reduction patterns. Figure 2.3 depicts three of these patterns: the sequential, block and loop aggregation patterns. Applying this technique to a real world example reached a compression rate of 50% [172]. However, authors do not generate any label or capability of the aggregated function items and consequently, *Requirement 4: Rich description* is not fulfilled.

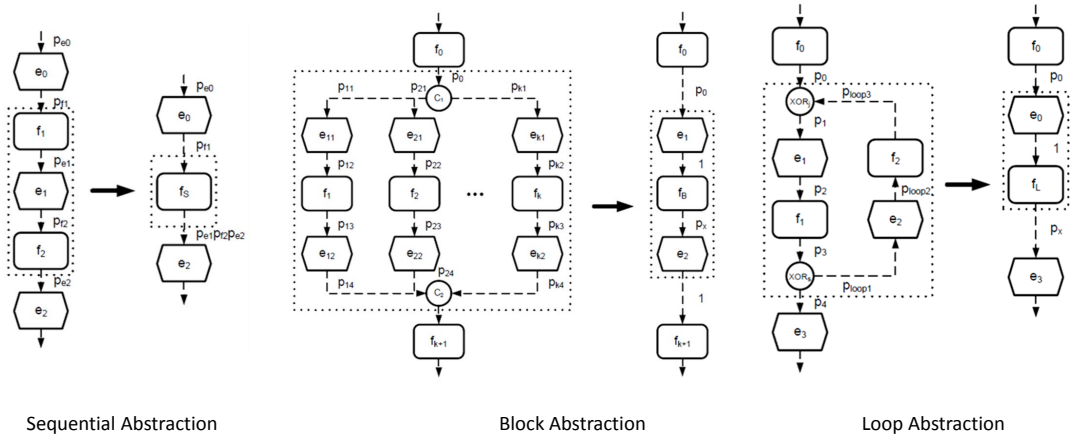


FIGURE 2.3: Business Process Aggreagtion using Structural Patterns [172]

2.3.3 Aggregation based on Semantic Similarity

Description

Rather than looking to the structure of the business process elements, Smirnov et al. [209] looked into the semantic similarity between process elements in order to detect the relevant aggregation candidates. The particularity of this approach is the type similarity measures that are used. The assumption of Smirnov et al. [209] on which they built their approach is that in industrial settings, process models are annotated with non-control flow elements such as the exchanged data items, the underlying service, the associated role, etc. In this context, the idea of the authors is: activities associated with such non-control flow elements are semantically related. Consequently, they use this measure to select potential aggregation candidates.

Critique

It is obvious that this approach would fail in case of the absence of the non-control flow annotations. However, this is not a big issue as annotations can be generated automatically from execution logs or process documentation.

From a technical perspective, this approach is applicable for simple and sequential process models. For complex control flow patterns, more investigations for improving this approach are needed. I can perceive two main cases where the approach might fail. The first case appears when the semantically related activities are spread across the model. The abstraction of these activities might generate semantic ambiguities amongst the remaining tasks. The second case appears when there is a conflict of annotations (e.g., an

activity is equally annotated with two roles). In this case a manual decision is required for choosing the most suitable aggregation.

With respect to *Requirement 4: Rich description* , here again the authors do not discuss how the labels of the aggregated activities are generated. The authors recognise the importance of this problem and consider it as part of their future work. Indeed, the authors plan the use of ontologies for the identification of a proper activity label based on meronymy relations between ontological concepts that is discussed in Section 2.3.5.

2.3.4 Lexical Relationships between Words

Description

Leopold et al. [132] propose an abstraction technique that aims to lift a process description from an entire process model to a single name/label. Similar to my vision, authors identify their approach as a way to *"ease the naming task for users who are interested in creating more abstract views on process models than are readily available to them"* [132]. This work makes use of theories of naming and exploratory research to derive names of process models. The idea of the contribution is sketched in Figure 2.4. During a first phase, process models are annotated with tags that identify start and end events, dominant elements, and main activities. A second phase uses these annotations to analyse and construct a list of candidate names using lexical relations between activity labels of the process model. The third step performs a ranking of a list of names that are returned to the end-user.

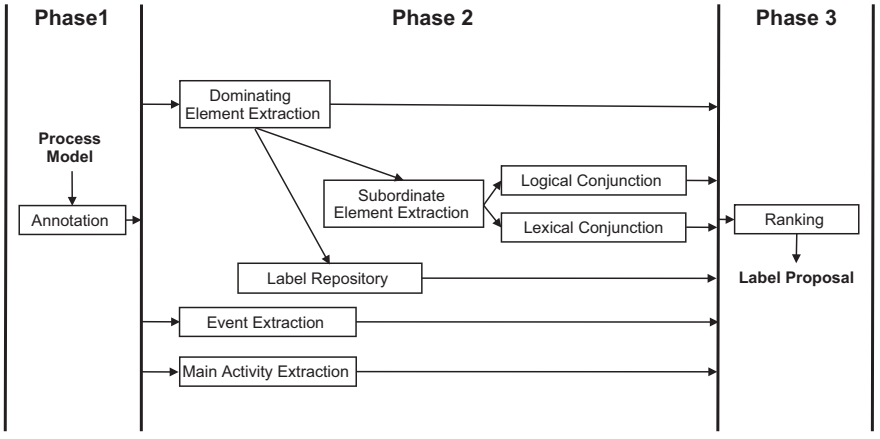


FIGURE 2.4: Business Process Abstraction using a Lexical Relations between Words [132]

Critique

The main advantage of this approach is that it allows to give a quick overview of the actions performed by a process through the analysis of its activity labels. Even though a case study shows that participants were very positive towards the resulting names, I would argue that limiting an entire process model to a single label is an issue. A single label does not feature properly the business parameters associated to the returned name. Furthermore, the identification of names highly depends on the annotation step that needs to properly determine the main activities and the dominant elements of the process model. Nevertheless, this is a promising technique that partially fulfills *Requirement 4: Rich description.*

2.3.5 Meronymy-based Aggregation

Description

Meronymy (part-of) relations between activity labels is investigated in Smirnov et al. [208] in order to capture the granularity of relationships between activities at several levels of abstractions. Meronymy or composition relation between activity labels can be explicitly captured in ontologies or taxonomies such as NAICS [161], UNSPSC [94] or the MIT Process Handbook [146]. Meronymy relations capture multiple levels of abstractions as shown in the meronymy tree example in Figure 2.5. The selection of the most relevant aggregation candidate can be driven by such a structure. In the example depicted in Figure 2.5, both proposed candidates can lead to a correct aggregation, however, with the use of the meronymy tree example provided, the first candidate is selected. The authors elaborate an advanced technique for scoring aggregation candidates even when the activities labels are not taken from the same meronymy tree.

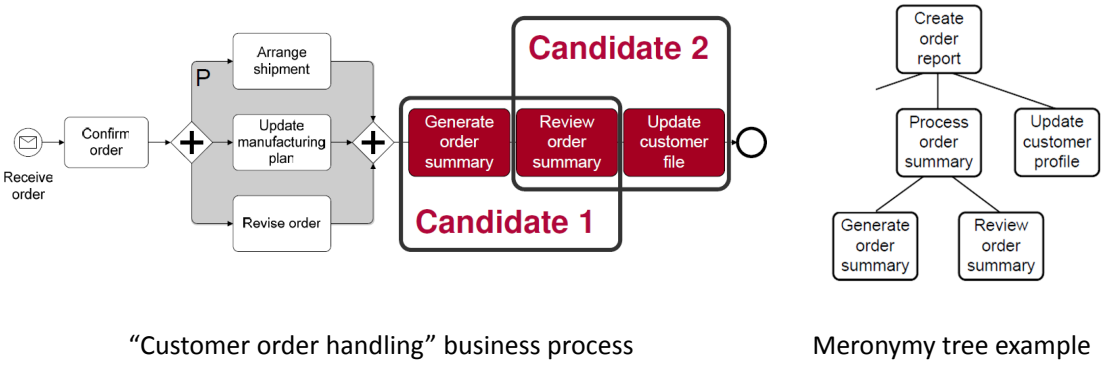


FIGURE 2.5: Business Process Abstraction using a Meronymy Tree [208]

Critique

This contribution builds on the aggregation of activities based in their semantic similarity (discussed in Section 2.3.3). It goes one step further with the identification of a candidate name of the aggregated activities. The main advantage of this approach is that it allows to generate the right label of the aggregated activities which is their lowest common ancestor in the meronymy tree. However, a single label does not properly describe the business capability of an entire process [228]. The authors also recall the importance of providing a richer description that is required to give business experts a comprehensive description of the business process [166, 228]. Consequently, *Requirement 4: Rich description* is partially fulfilled.

2.3.6 Propagation of IOPEs

Description

None of the previously reviewed works gave the importance of the capability of the aggregated activity. They either do not even address the issue of generating the label of the aggregated activity or do not go beyond a textual label entered manually or taken from an ontology/taxonomy. In a joint work with Vulcu et al. [228], I have proposed to represent business process models at several levels of abstractions. The intention was to compute aggregations of business processes or process fragments in order to match them with a business process search request. The model used for describing search requests included in addition to the structural aspect, the functional aspect in terms of Input, Output, Precondition and Effect parameters in addition to other quality of service properties. A business process model (either graph-based or block-based) can be represented in a structural hierarchy (see Figure 2.6). Each node of this hierarchy is further annotated by its capability and QoS parameters. These parameters are computed using a set of propagation functions over the composite capabilities and QoS parameters that are heavily relying on reasoning.

Critique

The major issue with this work is the use of the IOPEs for modelling business capabilities. The aggregated IOPEs in large business processes are costly to compute and result in large logical formulas that cannot be interpreted by human users. The propagation technique is similar to token propagation game in PetriNets [170]. I use the same idea for generating aggregated business capabilitites in Chapter 4. Even though the resulting

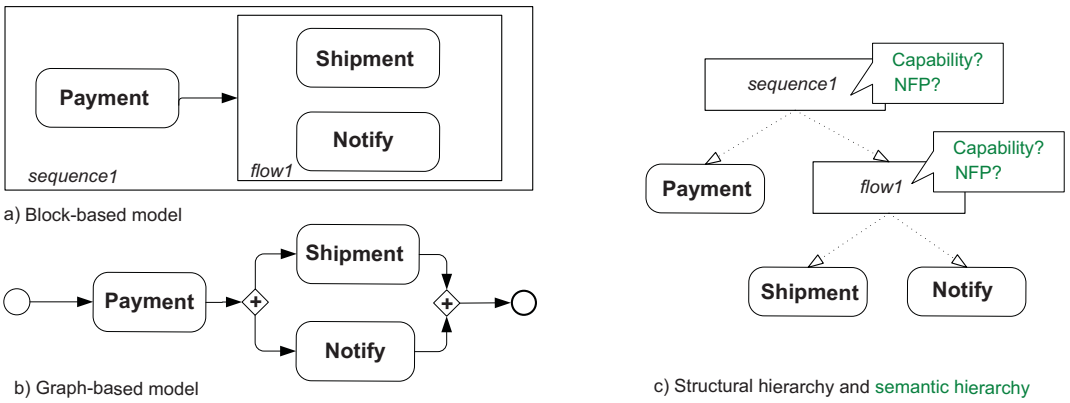


FIGURE 2.6: Block-based and Graph-based Business Process Models with their Structural and Semantic Hierarchies

description fulfils *Requirement 4: Rich description*, the resulting business capability does not fulfil *Requirement 1: Explicitly represent the action performed* and *Requirement 2: Explicitly capturing functional and non-functional features.*

2.3.7 Summary and Discussion

Except for the work carried out by Vulcu et al. [228], all the other approaches fail to compute the capability of the aggregated activities. Most of the reviewed approaches, that are summarised in Table 2.2, focus on the aggregation technique while ignoring the modelling of the functional aspect. This problem comes in essence from the fact that current business process models limit the capabilities of their tasks to simple textual labels.

The key points of the analysis are:

- The use of meronymy trees can easily drive the decision for defining a proper label for an aggregated set of activities [208].
- Using propagation functions for computing the aggregated capability is powerful and simple technique to implement [228].

2.4 Business Capability Indexing and Discovery

Section 2.2 focused on analyzing languages and conceptual models proposed to indirectly model business capabilities. The primary purpose of explicitly modelling capabilities is to enhance and automate their discovery and selection. However, the employment

TABLE 2.2: Comparative Analysis of Business Process Abstraction Techniques focusing on the Aggregation of Capabilities

Aggregation Technique	**Aggregated Capability**	**Limitations**	**Requirement 4: Rich Description**
Elimination of Activities [179]	No capability	Reduction of activities always comes along with information loss.	Not fulfilled
Structural Patterns [72, 172, 179]	No capability	Aggregated activities are manually labelled.	Not fulfilled
Similarity Measures [209]	No capability	If process models do not integrate non-control flow elements, the approach fails.	Not fulfilled
Lexical Relationships between Words [132]	Process name generated using lexical relations between activity names	Capability limited to a name	Partially fulfilled
Meronymy Trees [208]	Activity labels generated from the meronymy tree	Capabilities are limited to activity labels	Partially fulfilled
Propagation of IOPEs [228]	Complete capability using IOPE	Capabilities are expressed in terms of IOPEs that are costly to compute and difficult to read by human users.	Fulfilled

of semantic technologies and related tools for service discovery is particularly costly in terms of computational resources and not intended for use in highly dynamic and interactive environments [153]. Therefore, indexing/organizing capabilities to enhance their discovery is required. This section analyses approaches that propose to use indexing structures for enhancing business capability discovery.

For each of the approaches that I am analysing in this section, I consider the following two requirements from Chapter 1:

- *Requirement 5: Ontology based discovery* - Searching business capabilities should rely on concepts from domain ontologies used in the descriptions of business capabilities without relying on keyword extraction from textual descriptions [159]. Relying on extracting key words from unstructured textual descriptions can lead to inconsistent results [2]. This requirement was elicited from the following works: Sycara et al. [216], Oaks et al. [159], and Semantic Web Services Models: WSMO [242] and OWL-S [232].

- *Requirement 6: Time Performance* - Searching for a service or a business process is often relying of reasoning that makes the discovery very slow [153]. A consequent requirement is to propose a solution that is quicker so that it can be adopted in large repositories of services or business processes. This requirement was elicited

from the following works: Aznag et al. [9], Srinivasan et al. [211] and Mokhtar et al. [153].

Along these requirements, in my analysis, I highlight the adopted indexing **mechanism**, the underlying capability **modelling language** as well as its **limitations**. Note that the approaches under this category complement the previously discussed approaches in Section 2.2 and they cover mainly the requirement **CapR5: Index and Search** under the **Inferences** requirement.

2.4.1 Inheritance Between OWL-S Services

Description

The discovery of semantic web services suffers primarily from the large repositories together with the required costly reasoning operations. Ferndriger et al. [74], propose to enhance this task by introducing inheritance between OWL-S services. Their specification denotes the possibility to define service profiles' hierarchies similar to object-oriented inheritances. Inheritance relationships between services are proposed to find service substitutes by exploring services that are higher in the hierarchy. Similar to the object-oriented concepts, a sub service may be used to substitute its super service for automated, dynamic service discovery and composition [74]. Inheritance is also useful for creating new service profiles as a subclass of an existing profile. This makes the new service inherit the properties defined in the superclass profile. Elenius et al. [71] propose to capture such hierarchies in a visual editor for an OWL-S service description editor without discussing how these hierarchies can be created.

Critique

Introducing inheritance was a natural decision to enhance service discovery operation [74]. Services and relations are defined in OWL language and consequently fulfils *Requirement 5: Ontology based discovery.* However, little research has been carried out to further investigate and determine inheritance between web services [104], consequently I cannot further comment on *Requirement 6: Time Performance.*

2.4.2 Topic Extraction and Formal Concept Analysis

Description

Aznag et al. [9] investigated the use of formal concept analysis as an indexing tool using topics extracted from service descriptions using SA-WSDL. Starting from a set of service descriptions, their algorithm converts them into a so called "service transaction matrix" that captures for each service the relevant textual concepts used in its description. This matrix is further refined with probabilistic clustering of the textual concept in order to extract a set of topics. The result of this analysis generates the correlated topic model that holds for each service the topics it belongs to with certain probabilities. In this work, formal concept analysis is used exclusively for clustering the extracted topics in order to make the discovery easier when using a concept lattice. The use of formal concept analysis in this approach is its wide adoption as a well established mathematical theory of concepts and concept hierarchies that makes the service discovery much easier [9].

Critique

Topics used in the concept lattice are textual concepts that are extracted from the textual description of services and consequently this approach does not fulfil *Requirement 5: Ontology based discovery*. With respect to *Requirement 6: Time Performance*, the authors did not perform any evaluation of the time required to create the cluster of services. Nevertheless, they indicate that the query response time varies between 300 and 3000 milliseconds with a test collection of 1088 services. Given that the discovery operation using formal concept analysis is a simple tree parsing operation, it has a linear complexity depending on the number of concepts in the created lattice (tree).Furthermore, in formal concept analysis, the creation of the concept lattice is the most expensive operation [214], thus this can lead to the conclusion that the construction time of the entire cluster could be in the order of seconds.

2.4.3 Reasoning-based Matching

Description

Srinivasan et al. [211] looked into enhancing the indexing of a UDDI registry of services described in OWL-S during the service advertisement phase. Assuming that capabilities of services are described using predefined ontologies, they use a matching degree between

inputs and outputs of services with concepts from these ontologies in order to identify a correct clustering of service descriptions into predefined ontological clusters similar to NAICS [161].

Critique

Requirement 5: Ontology based discovery is fulfilled, as this approach relies exclusively on clusters that are constructed from hierarchical ontological concepts similar to NAICS [161]. However, the problem with this approach is that it relies heavily on reasoning and pre-computing information required for the search request which is costly (*Requirement 6: Time Performance*). This has been further confirmed by the authors through the performance measures they carried out. The OWLS/UDDI approach takes more than 4000 milliseconds for inserting 50 advertisements into the registry which was 6 to 7 times slower than using a classical UDDI approach. However, the authors argue that this time is not very important as the advertisement operation can be done offline and more time can be saved during the discovery phase without giving any quantifications.

2.4.4 Numerical Encoding of Ontological Concepts

Description

Mokhtar et al. [153] optimize the indexing of service descriptions by avoiding semantic reasoning and by using a numeric coding scheme, a widely adopted method for enhancing the performance of ontology processing. Mokhtar et al. propose that a service registry can be clustered using a predefined ontology or taxonomy such as NAICS [161] or UNSPSC [94] where each ontological concept is encoded by an interval of numbers. These intervals are defined using a linear inverse exponential function in a way that one interval can be contained in other ones without overlap creating a subscription relation. For example, in order to model subconcept relations between WiFi and Wireless, WiFi can be coded by the interval [0, 0.1] and Wireless by the interval [0,1] (e.g., [0, 0.1] is contained in [0,1]). One can add to this example another concept Bluetooth that is a subconcept of Wireless by assigning the interval [0.2,0.3] to Bluetooth ([0.2, 0.3] is contained in [0,1] where [0, 0.1] and [0.2, 0.3] do not overlap).

Critique

Similar to Srinivasan et al. [211], Mokhtar et al. [153] rely exclusively on clusters that are constructed from hierarchical ontological concepts similar to NAICS [161], consequently,

Requirement 5: Ontology based discovery is fulfilled. With respect to *Requirement 6: Time Performance*, and compared to the performance of the work of Srinivasan et al. [211], Mokhtar et al. achieves much better results as the required time for encoding and advertisement does not exceed 450 milliseconds for 50 service descriptions. This performance is achieved with the assumption that the used ontologies for service classification are encoded similarly, the reasoning operation are reduced to a comparison of codes/intervals. In such case, to infer that a concept c1 subsumes another concept c2, one needs to evaluate if their corresponding encoding interval of c1 is contained in the encoding interval of c2. This restricts the system to use classification ontologies that do not evolve frequently otherwise service advertisements and requests need to periodically check and update their encoding intervals when needed.

2.4.5 Summary and Discussion

In summary, most of the analysed approaches rely on indexing service descriptions using exiting taxonomies such as NAICS [161], UNSPSC [94] or the MIT Process Handbook [146]. This supports the idea of using ontologies as a common conceptualisation and shared understanding among service providers, registry hosts and service requesters. However, the use of these ontologies makes the indexing heavily reliant on reasoning, a task that can be costly. The literature proposes multiple techniques that either used reasoning or propose alternative solutions, relevant contributions were discussed in this section and summarized in Table 2.3.

Table 2.3: Comparative Analysis of Capability Indexing Approaches

Approach	**Requirement 5: Ontology based discovery**	**Requirement 6: Time Performance**	**Limitations**
Inheritance between OWL-S services: Elenius et al. [71]	Not fulfilled	N/A	- no clear methodology on how a hierarchy is created
Topic extraction and Formal Concept Analysis: Aznag et al. [9]	Fulfilled	size: 1088 services, query response time between 300 and 3000 ms	- topic extraction and correlations are based on a probabilistic system; - the solution needs further optimizations; - FCA is exclusively used for topic clustering
Reasoning-based matchmaking: Srinivasan et al. [211]	Fulfilled	size: 50 services, index construction + advertisement time: $\sim$ 4 s	- service advertisement operation is costly and heavily relying on reasoning
Numerical encoding of ontological concepts and codes comparison: Mokhtar et al. [153]	Fulfilled	size: 100 services, index construction + advertisement time: $\sim$ 500 ms	- requires periodic updates of the codes of the clustering ontology; - slow registry maintenance for large repositories

The key findings of this analysis are:

- Indexing or clustering of service descriptions using ontologies is widely adopted [9, 153, 211].
- Maintainability of the indexing structure is critical to the applicability of the proposed approach [153, 211].
- The benefits of reusing existing techniques such as Formal Concept Analysis for creating or maintaining the indexing structure is widely accepted [9].

2.5 Reuse of Process Models Driven by Business Capabilities

In this section I review current research contributions related to the topic of reuse in business process modelling. I investigate if current approaches consider business capabilities of activities or entire business processes in their work. This analysis helps identify where business capability descriptions can be used to enhance the reuse of business process models.

This section is organised according to Figure 2.7, that is: Section 2.5.1 and 2.5.2 respectively outline two main categories of reuse-oriented business process modelling techniques: (i) using *Business Process Repositories* and (ii) using *Reference Business Process Models*.

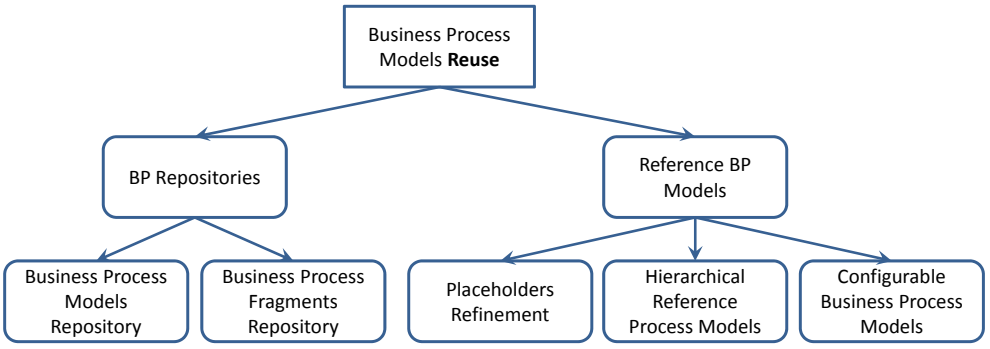


FIGURE 2.7: Classification of Reuse-oriented Business Process Modelling Approaches

First, Section 2.5.1 investigates various implementations of business process repositories that permit either to discover an entire business process model or to discover business process building blocks that can be used later for composition.

Second, Section 2.5.2 investigates three implementations of reference process models either by refining placeholders (in Section 2.5.2.1), using hierarchical reference models (in Section 2.5.2.2) or customising configurable models (in Section 2.5.2.3).

The requirements identified in Chapter 1 target primarily the configuration based-modelling of business processes. These requirements are also used in analysing other techniques as follows:

- *Requirement 7: Integration of Business Capabilities* - Reuse oriented modelling of business processes requires the management of multiple variants of the same process [123]. This makes their management difficult to handle by business experts [120], thus integrating the business properties in these models makes their discovery and configuration easier for these users [120]. This requirement was elicited from the following works: La Rosa [116, 119, 120].

- *Requirement 8: Execution Time* - This requirement targets reference business process modelling techniques. Indeed, the manual creation of reference process models is time consuming task and thus an automatic approach should be quick enough to give quick results to business experts. This requirement was elicited from the following works: Gottschalk [89], La Rosa et al. [120] and Assy et al. [7].

- *Requirement 9: Compression Rate* - Managing multiple variants of the same business process should consider common elements and avoid redundancy, especially in large business process repositories [120]. This results into reduced size of input process elements with a high compression rate if there is a high similarity between the variants [90]. This requirement was elicited from the following works: La Rosa et al. [120] and Assy et al. [7].

2.5.1 Business Process Models Repository

In the first part of the analysis, I study related work in the area of business process model discovery. The literature distinguishes two main research directions related to business process discovery. The first one consists of detecting the business process activities and their ordering by analysing business process execution logs, that is business process mining [31, 220]. The second one consists of querying a business process repository in order to find a relevant business process model satisfying particular needs. This work focuses on the second research direction. Specifically, I examine existing approaches for managing business process model repositories.

2.5.1.1 The Process Variant Repository

Description

The Process Variant Repository [136–138] or PRV for short, defines a repository of both business process models and associated "*preferred work practices*". A preferred work practice is a process variant that is captured from the process execution logs and is suitable for a particular situation. Each process model is stored with its historical information about the execution instances in order to achieve new operational goals in similar situations.

PVR provides a support for querying business process models and their variants where a query is a partial or complete description of a process variant. On the basis of similarity metrics, the authors measure the equivalence and subsume relations [192] between the process query and the stored processes using reduction techniques in graphs. The results are then ranked based on these similarity values.

Critique

With respect to the identified requirements, the key points of analysis are as follows:

- *Requirement 7: Integration of Business Capabilities*: not fulfilled. Even though the authors provide a rich model for storing business process variants, it was scoped only toward the structural aspect and consequently do not consider the integration of business capabilities in the search requests or in the stored business process models.
- *Requirement 8: Execution Time* : not applicable.
- *Requirement 9: Compression Rate*: not fulfilled. The PVR focus is more on providing a discovery mechanism while ignoring any challenges related to maintainability and particularly to managing common process parts. Business process variants are stored individually without performing any compression.

2.5.1.2 BP-Suite

Description

BP-Suite is a tool-set for querying BPEL-based business process repositories. It consists of three query subsystems: (1) BP-QL [17] is used to query business process specifications

(which is the system related to this work); (2) BP-Mon [18] is used for monitoring process instances at run-time and (3) BP-Ex [12] allows for querying business process execution logs.

The focus of BP-QL is to use XQuery [230] to discover business processes given a structural pattern. Entries of the repository (i.e., business processes) are described using AXML [1], an abstraction of BPEL. The proposed language represents business processes as graphs, i.e., with nodes and links between them. Since the BPEL specification is also XML-based, an obvious question is why not query it directly? The answer to this question, according to the authors [16], is ease of use. Indeed, the BPEL format is complex and extremely inconvenient for querying.

Critique

The authors claim that their query building mechanism is user friendly as it is similar to those used by commercial vendors for the design of BPEL processes. However, it is important to notice that "*BPEL more closely resembles a programming language than a modeling language*" [221] which requires some learning. This makes the proposed approach helpful for reducing the learning curve of non-experts.

With respect to the identified requirements, the key points of analysis are as follows:

- *Requirement 7: Integration of Business Capabilities*: not fulfilled. Since the authors do not report on the integration of business capabilities, I assume that this requirement is not fulfilled.
- *Requirement 8: Execution Time*: not applicable.
- *Requirement 9: Compression Rate*: not fulfilled. The BP-Suite focus is more on providing a user friendly discovery mechanism while ignoring any challenges related to maintainability and particularly to managing common process parts. Business process variants are stored individually without performing any compression.

2.5.1.3 Semantic Business Process Repositories

Description

In this section I review four repositories of business process models that use semantics.

First, the Semantic Business Process Repository, or SBPR [140], describes business processes using ontologies such as: process, organizational and business function (i.e.,

business capability) ontologies. They use relational databases to store these descriptions. A reasoner such as Integrated Rule Inference System - IRIS [1] is supposed to be integrated with the semantic business process repository to reason over the business processes described using ontologies.

Second, while the framework for querying business process models proposed by Markovic et al. [143] also uses ontologies for describing business process models, Sakr and Awad [194] use ontologies only in the query matching process and tackle the problem of applying different terminologies when modelling processes. The former [143] uses Web Service Modeling Ontology (WSMO) [182] for describing functional and non-functional related properties and a process algebra, pi-calculus, for the structural properties of a business process model. They use Web Service Modeling Language (WSML) logical expressions as a query language and ontological reasoning for query answering. Whereas the latter [194] relies mainly on activity labels for describing functional properties and uses BPMN-Q [8] for querying business process models with an underlying classical database management system.

Last, the *oryx* [46] extension for semantically-enabled business process discovery [228] also proposes the use of ontologies for modelling and storing business process models. The authors propose in this work an ontology for describing graph-based and block-based business processes while capturing their functional (i.e., Input, Output, Precondition and Effect) and non-functional properties at multiple levels of abstraction.

Critique

With respect to the identified requirements, the key points of analysis are as follows:

- *Requirement 7: Integration of Business Capabilities*: partially fulfilled. Each of the examined solutions highlight the importance of describing the business capability using domain ontologies. However, they do not go beyond using simple labels of the classical Input, Output, Precondition and Effect approach (see analysis in Section 2.2).
- *Requirement 8: Execution Time*: not applicable.
- *Requirement 9: Compression Rate*: not fulfilled. The reviewed solutions investigate the use of ontologies for storing and querying business process models. They use graphical querying mechanisms for supporting users and avoiding learning a complex querying language. However, none of them deals with how to efficiently store process variants: compression is out of their scope.

[1] http://www.iris-reasoner.org/

2.5.1.4 APROMORE

Description

APROMORE (Advanced PROcess MOdel REpository) [122] is a recently proposed process models repository supporting multiple modelling languages including EPC, BPMN, Protos, WF-Nets, YAWL, and WS-BPEL. It manages company specific process models, reference models and process patterns. The strength of this repository is that it builds on a large set of existing contributions in terms of approaches and techniques which have been adapted and incorporated as evaluation, comparison, management and presentation functionalities.

APROMORE is open to integrate multiple contributions related to the management and maintainability of business process repositories. Examples of such contributions include the detection of clones [67, 218] and errors [150] in the repository.

Critique

With respect to the identified requirements, the key points of analysis are as follows:

- *Requirement 7: Integration of Business Capabilities*: not fulfilled. Even though APROMORE considers managing various aspects of business process models: activity, control flow, data and resource, it fails to properly represent their **functional aspect**. The capabilities of tasks/activities are limited to their labels which makes, the functional retrieval of process models limited to label matching (syntactically). In the best cases, this can be improved by introducing similarity measures covering various interrelated terminologies (e.g., synonyms).
- *Requirement 8: Execution Time*: not applicable.
- *Requirement 9: Compression Rate*: fulfilled. In order to overcome the problem of resource efficiency and propose a suitable compression of the stored business process variants, APROMORE proposes the integration of merging and individualisation features which relate to the area of configurable process models [184]. Those feature will be discussed in details later in this chapter.

2.5.1.5 Business Process Models Repositories: Summary and Discussion

The reviewed business process models repositories share in essence the same objective: discovering a business process model by querying a repository and selecting the most

suitable one. As depicted in Figure 2.8, this technique involves a process variant repository and two kinds of stakeholders: (i) a process modeller and (ii) a business expert. The process modeller is responsible of regularly updating the process variant repository. The business expert has to query this repository in order to find out a particular business process variant. Learning a customized query language for retrieving a suitable business process model is not at all user-friendly. This motivated current approaches to propose graphical querying languages and interfaces for end users.

With respect to *Requirement 7: Integration of Business Capabilities*: The proposed solutions mainly rely on the labels of activities/task of the business process models in order to determine their function. The use of ontologies for semantic annotation is also proposed for enhancing the similarity measuring [70] or for querying purposes [140, 228].

Requirement 8: Execution Time is not applicable to these solutions.

For *Requirement 9: Compression Rate* : This requirement is needed in order to avoid duplication of common process parts and ensure consistency (i.e, every change of a process model has to be propagated in all similar models) and correct (i.e., without clones and errors). As these solutions do not consider managing common process parts as single elements, additional maintainability effort for ensuring a clean repository such as the detection of clones [67, 218] and errors [150] is required.

A key point of this analysis is: Even though it is well recognized that process variants share some commonalities, this has not been taken into account in these approaches. In

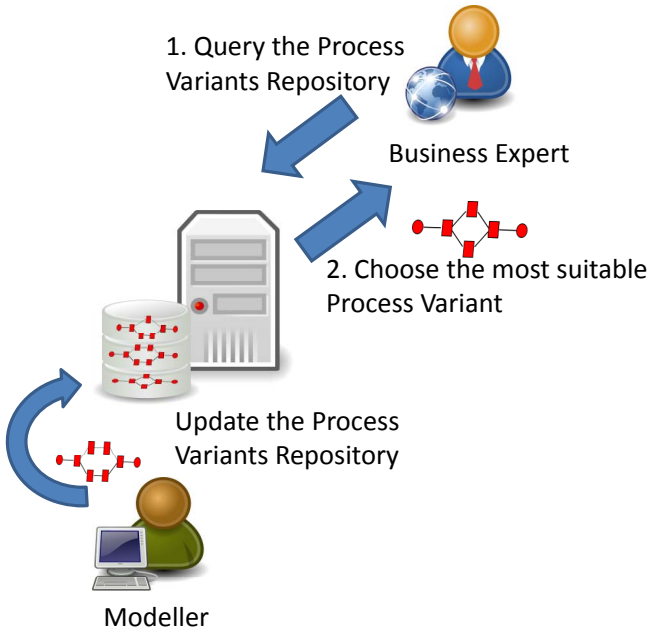


FIGURE 2.8: A Process Variants Repository for Reusing Business Process Models

fact, each process variant is stored as a standalone entity. Consequently, this method suffers from resource redundancy because it does not consider common parts of process models which are duplicated in each entry of the repository. This can be resolved by storing business process building blocks instead of entire models. These building blocks can be later retrieved and aggregated in order to construct a business process model [141, 201]. As depicted in Figure 2.9, the business expert will have to query the required building blocks and aggregate them in order to derive his entire business process model. Modelling business process models from building blocks still requires some modelling skills but can be reduced by using dynamic composition [207].

2.5.2 Reference Business Process Modelling

In the second part of the analysis, I study three implementations of reference business process modelling techniques. A reference process model is a generic model that can be tailored to specific needs and adapted to various situations. Stakeholders benefit from these models by avoiding the need to create a model from scratch and use the reference model as a starting point. The main challenge with such solutions is that a reference model has to be properly managed in order to help deriving a proper process variant.

In this section I review three implementations of reference process models:

1. Using reference models with placeholders that need to be refined during the modelling phase (detailed in Section 2.5.2.1).
2. Using hierarchical reference process models that capture variability at various levels of abstraction (detailed in Section 2.5.2.2).

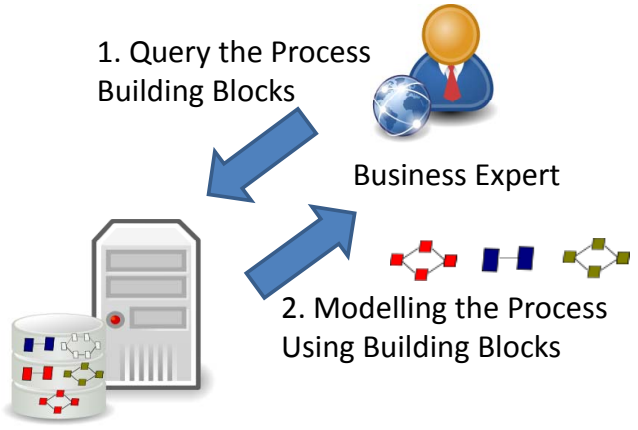


FIGURE 2.9: Using Process Building Blocks for Modelling Business Processes

3. Using configurable business process models that allow the customisation of process models by enabling or disabling some branches of the model (detailed in Section 2.5.2.3).

2.5.2.1 Placeholders Refinement: Late modelling

Description

Creating a model with a *placeholder*, or a *pocket of flexibility*, as introduced by Sadiq et al. [189], provides the means for creating flexible business process models. The idea is to create a partially completed business process model with placeholders that require late modelling. The late modelling allows business processes to be tailored either to a process model during the modelling phase or to individual instances at runtime [15, 234].

During the late modelling users can refine the placeholders using their own modelling skills. They can be assisted either with a set of activities and/or constraints as it has been highlighted by Sadiq et al. [189]. The authors also distinguish three options for implementing late modelling:

- *Option 1: Reference Process Model.* Placeholders may be defined without any constraints or predefined activities.
- *Option 2: Reference Process Model + Set of Activities.* Placeholders may be defined using the predefined set of activities without any constraints.
- *Option 3: Reference Process Model + Set of Activities + Set of Constraints.* Placeholders may be defined from the predefined set of activities under the given set of constraints.

In [190], Sadiq et al. propose an implementation of option 3 for late modelling. Figure 2.10 illustrates the proposed approach. This example defines a set of activities and constraints that are needed to define the placeholder (i.e., task B) of the process model. At runtime, the placeholder / pocket of flexibility is defined for a given process instance based on tacit knowledge.

Critique

With respect to the identified requirements, the key points of analysis are as follows:

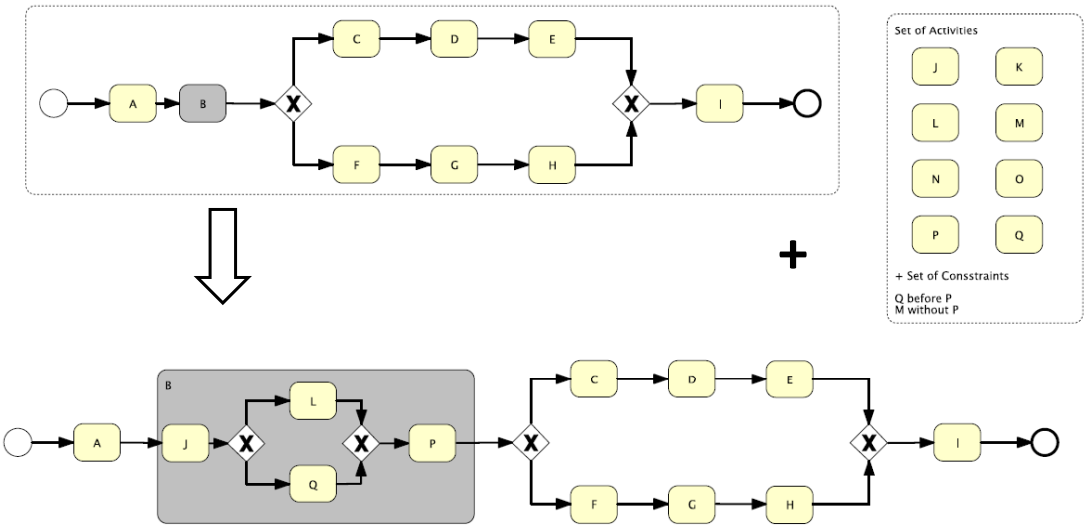


FIGURE 2.10: Using Placeholders for Managing Business Process Variants (adapted from [234])

- *Requirement 7: Integration of Business Capabilities*: not fulfilled. Most of the reviewed approaches in this category do not consider this requirement and keep relying on the labels of activities for representing business capabilities.
- *Requirement 8: Execution Time*: not applicable. Even though it has been noticed that there is a need to help users create sound and correct models [168, 222], I could not find any contribution that creates and updates such reference process models in order to comment on this requirement. Nevertheless, more effort has been put in maintainability, the literature proposes various algorithms for checking the satisfiability of the constraints [139, 169] used with the predefined set of activities.
- *Requirement 9: Compression Rate*: fulfilled. In essence the use of a reference model guarantees that duplicate process elements are merged together, ensuring a high compression rate.

2.5.2.2 Hierarchical Reference Process Models

Description

In most cases, business process models tend to be very large and are more and more difficult to manage by end-users. Reducing the complexity of large models can be achieved by representing them at different levels of detail. The general idea is to reduce the complexity of business processes and reveal to the end user a partial model by applying abstraction techniques similar to those previously reviewed in Section 2.3. This fosters

the reuse of similar process fragments as well as reducing inconsistency problems. In this context, some researchers tried to manage reference process models at various levels of abstraction while explicitly capturing variation points. The object of this section is to review the proposed approaches that study such models, i.e., hierarchical reference process models.

Razavian and Khosravi [176], propose a variability modelling method which is specifically designed for the component and connector view of UML 2. The authors introduce multiple mechanisms for modelling variation points depending on the variable element (component, connector or interface). Variation points are presented at various levels of abstraction by having optional or alternative architectural elements. An example is shown in Figure 2.11 where the top level component "UI Manager" can be further refined to one of the two associated variants: "JavaScript UI Manager" and "HTML UI Manager". Each element is annotated by specific stereotypes: the variation point is marked by $<< alt\_vp >>$ and its lower level sub processes express all details related to higher level activities and variabilities residing in them and they are annotated by $<< variant >>$.

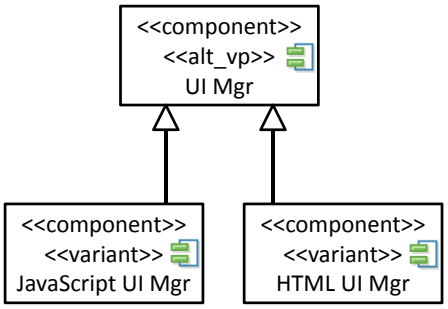


FIGURE 2.11: "UI Manager" Variation Point using Hierarchical Representation [176]

Baran et al. [13] investigated the use of hierarchical reference business process models using BPMN. Such models are created in a two-step operation. First, the proposed algorithm transforms the input BPMN models into two-level hierarchical models. The authors use a very simple abstraction technique that takes as input a BPMN model and the set of interlinked high-level and low-level tasks and delivers the corresponding hierarchical model. Second, the BPMN models are merged into a single one that requires additional transformations to become well formed.

I have also explored the use of hierarchical reference process models [48, 54]. I proposed the use of an indexing structure for representing process models at different levels of abstractions as depicted in Figure 2.12. I used the concept of abstract tasks for capturing variation points, it is marked with a "*" at the end of the task label (see "Customer Registration*" on Figure 2.12). An abstract task can be refined/concreted by selecting

one of its concrete alternatives which are associated to it via dotted lines. In addition to this customised notation, I proposed an algorithm for updating the reference model by inserting a new node (either a task or sub-process). The work that I proposed has not been further investigated.

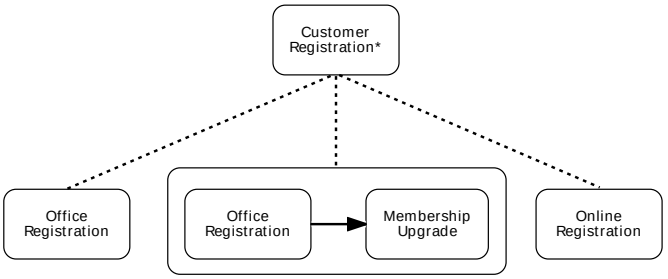


FIGURE 2.12: A Hierarchical Indexing Structure for Modelling one Variation Point of a Process for Registering a Customer to an Insurance Contract [48]

Critique

With respect to the identified requirements, the key points of analysis are as follows:

- *Requirement 7: Integration of Business Capabilities*: not fulfilled. The reviewed approaches in this category do not consider this requirement and keep relying on the labels of activities for representing the business capabilities. Only Razavian and Khosravi [176] recognise the importance of this view and prospect explicitly interlinking model and business variation points.
- *Requirement 8: Execution Time*: fulfilled. All the studied approaches offer the required methods for creating reference process models. However, none of them has done an evaluation regarding the execution time of their algorithms. I assume this requirement is fulfilled as an automation support to reduce manual efforts has been proposed.
- *Requirement 9: Compression Rate*: fulfilled. In essence the use of a reference model in general, and hierarchical model in particular, guarantees that duplicate process elements are merged together, ensuring a high compression rate.

2.5.2.3 Configurable Business Process Models

Description

A configurable business process model [184] is the result of merging process variants into a single model. This model can be tailored to the analysts' needs by enabling or disabling

different branches of the configurable model. Figure 2.13 depicts, in the left-hand side, two variants of the same business process. These two variants reflect two common tasks (i.e., Task A and B), however after this, each variant ends with a different task (i.e., C or D). This difference introduces the choice between the task C or D that represents a variability which is depending on various indicators, e.g., cost, quality of service, user preference, etc.

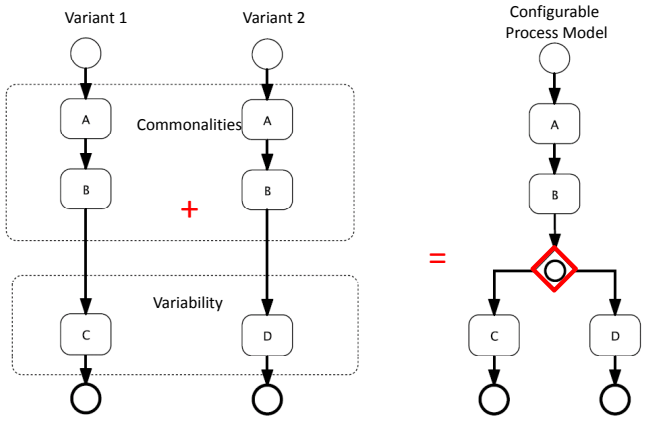


FIGURE 2.13: Configurable Business Process Model (adapted from [117])

The right-hand side of the Figure 2.13 shows the configurable process model which is a merger between the two process variants. The variation point is represented by a configurable gateway: an inclusive split gateway marked with a thick red border. Unlike a "normal" BPMN gateway, it does not represent a choice or a parallel split, instead, it represents a design choice that needs to be made by an analyst to adapt the configurable process model to a particular requirement. In this example the configurable gateway captures the fact that one needs to choose whether to select one path (i.e., task C) or the other (i.e., task D), or possibly both.

In this case, the modelling phase consists of enabling or disabling different branches of the configurable process model. This allows customization of the configurable process model by choosing the right variant. However, the main weakness of this solution is that it does not allow the business users to understand the relationship each variant has with the business domain. There are two important challenges for adopting these models: (1) automation support for creating a configurable model and (2) assisting end-users during the configuration phase.

Gottschalk [89] proposed in his thesis to provide an algorithm for automatically creating configurable process models by merging a set of input variants. This contribution as well as other merging algorithms will be analysed in detail in Chapter 6.

La Rosa recognises the need to make the configuration phase user-friendly and proposes a questionnaire-driven configuration [116, 118]. The proposed approach is sketched on

Figure 2.14. A process modeller has to define the configurable process model and meet with the domain expert in order to define domain constraints (i.e., business capabilities) and their mapping to the model. The configuration is then performed via an interactive questionnaire. The domain expert's answers are then mapped to the configurable model in order to "individualize" it in a process model.

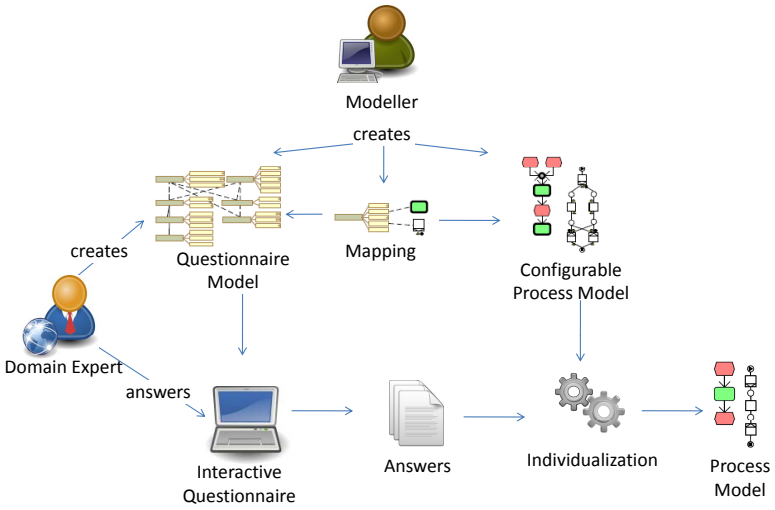


FIGURE 2.14: Questionnaire-driven Approach for Configurable Business Process Modelling [116]

Critique

The solution provided by La Rosa [116] offers a suitable methodology for user-centric process modelling. However, adding a new variant in this framework, requires another meeting between the modeller and the domain expert in order to (i) add a variant to the process model, (ii) define its mapping with the domain constraints and (iii) update the questionnaire model. From this arises a need to improve this solution by providing an automation support for maintaining the configurable process model and the questionnaire model.

With respect to the identified requirements, the key points of analysis are as follows:

- *Requirement 7: Integration of Business Capabilities*: partially fulfilled. La Rosa captures business capabilities as a set of questions and answers that domain experts are familiar with. However, these questions and answers are manually linked to the model after its creation.
- *Requirement 8: Execution Time*: fulfilled. Several contributions for automatically creating a configurable business process model from a set of process variants are

proposed in the literature. Those contributions guarantee a reduced execution time compared to manual creation.

- *Requirement 9: Compression Rate*: fulfilled. In essence the use of a reference model in general, and configurable process model in particular, guarantees that duplicate process elements are merged together, ensuring a high compression rate.

2.5.3 Reference Business Process Modelling: Summary and Discussion

This chapter analysed reuse in the context of business process modelling. The results of the analysis are summarised in Table 2.4 that clearly shows that while the approaches considered in this analysis exhibit a heterogeneous set of methods for reuse and variability management in process models, none of them addresses all the identified requirements.

Various contributions regarding maintaining those reference models were discussed from different perspectives such as providing standard CRUD operations on a repository of models [140, 194, 228] to clones or error detection [67, 150, 218] and automatically creating configurable process models [89]. I propose my contribution to cover the maintainability aspect in Chapter 6 by providing a new algorithm for creating a capability-annotated configurable process model.

The reviewed solutions are motivated by the user support for modelling business processes by enhancing querying a repository [16, 46] or customizing a reference model [116]. A task that is difficult and has been extensively discussed in the literature. Even though La Rosa [116] proposes a prevalent solution to this problem, it still suffer for a major shortcoming: the need for an extensive manual matching between the model and the domain constrains. I propose in Chapter 6 to integrate domain properties (business capability) to the process models before the creation of configurable process models in order to automatically generate configuration options.

Most of the analysed approaches focus on the structural aspect of business process models which led them to limit the descriptions of the functional aspect (i.e., capability) to either task labels or an accompanied textual descriptions. It is important, though, to highlight that some of these contributions recognise the importance of this functional view and represent it using semantic annotations [140, 143, 194, 228] or capturing it in a separate structure [116, 176]. I will discuss further this aspect in Chapter 3 where I will investigate a new vision for modelling this aspect.

From a compression perspective, it is obvious that most of the approaches that use a single structure are covering this aspect. However, the approaches that favour the

TABLE 2.4: Comparative Analysis of Techniques of Reuse in Business Process Modelling

Approach	**Requirement 7: Integration of Business Capabilities**	**Requirement 8: Execution Time**	**Requirement 9: Compression Rate**
The Process Variant Repository [136–138]	Single labels and textual descriptions	Not Applicable	Not Fulfilled - stores duplicate models
BP-Suite [12, 17, 18]	Single labels and textual descriptions	Not Applicable	Not Fulfilled - stores duplicate models
Semantic Business Process Repositories [140, 143, 194, 228]	Labels, textual descriptions and semantic annotations taken from domain related ontologies	Not Applicable	Not Fulfilled - stores duplicate models
APROMORE [122]	Single labels and textual descriptions	Not Applicable	Fulfilled only when using merged models
Placeholders Refinement: Late Modelling [189, 190]	Single labels and textual descriptions	Not Fulfilled - not algorithms available for automatic creation	Fulfilled - all models are merged into a reference model
Hierarchical Reference Process Models [13, 48, 54, 176]	Explicitly captures the business view but not linked to the process model [176]	Partialy Fulfilled - algorithms for automatic creation are available but not evaluated in terms of execution time	Fulfilled - all models are merged into a reference model
Configurable Business Process Models [89, 116, 118, 184]	Explicitly captures the business and the model constraints; however, their linking is manually made through interviews between domain experts and modelling experts [116]	Fulfilled - algorithms for automatic creation are available and evaluated in terms of execution time	Fulfilled - all models are merged into a reference model

use of business process repositories fail unless they consider managing business process fragments rather than entire models [141, 201].

The key findings of the analysis are:

- In business process modelling, most of the contributions did not investigate the integration of the business capability into business process models.
- The integration of business capabilities in business process models can improve the customization of configurable business process models by automatically generating configuration options in terms of domain parameters (i.e., business capabilities).

2.6 Conclusion

In this chapter, I reviewed major contributions that tackle directly or indirectly the capability-driven management of services and business process models. The analysis of each area of interest, helped identify the following limitations:

- *Business Capability Modelling*: Currently, capabilities are expressed in terms of IOPEs to describe the state of the world before and after the execution of a service or business process. In most of the cases a capability is limited to single label and textual description or confused with its implementation (i.e., invocation interface). It has been proven that such descriptions are hard to define by human users either for creating new capabilities or formulating their search requests. In Chapter 3 I explore another vision for modelling capabilities as a set of actions and related features defined in a set ontologies.
- *Business Capabilities Aggregation*: Similar to indexing capabilities, due to the current modelling languages of capabilities, the aggregation operation is costly and hard to read by human users. Chapter 4 constitutes my contribution to resolve these issues.
- *Business Capability Indexing and Discovery*: Due to the current description languages of capabilities, their indexing and discovery remains a costly task as it is heavily reliant on reasoning. Using the capability model I propose in Chapter 3, I explore in Chapter 5 an indexing and discovery mechanism of capabilities.
- *Capability-driven Reuse of Business Process Models*: Currently, the customization of configurable process models requires an extensive manual work. In Chapter 6, I show how the early integration of capabilities in business process models can

automate the generation of configuration options that make the customization of configurable models easier.

Chapter 3

Towards a Structured Business Capability Description

"I have an unusual type of thinking. I have no visual memory whatsoever. Everything is conceptual to me."

Craig Venter

3.1 Introduction

The concept of capability has been defined by Dutta et al. [68] and Amit and Schoemaker [3], from organizational and data perspectives, as the ability of organizations to efficiently use their resources (i.e, human capital, knowledge, available data, etc.) to generate value and achieve their objectives. BPM [238], defines capabilities, from a control flow perspective, as the way (i.e, HOW) organizations achieve their goals by capturing explicitly process tasks and their temporal and logical order. And from a functional perspective, OASIS Reference Model [160] considers capabilities as the effect of a service in terms of data generated or change of the world and Oaks et al. [159] defines capabilities as what a service does in terms of action performed that creates a value for the customers.

Considering the functional perspective, the definition given by OASIS Reference Model [160] is tight to the actual implementation of a service and thus defines *IT capabilities* while Oaks et al. [159] try to give a more *business/functional capability* definition. However, the capability conceptual model proposed by Oaks et al. [159] does not go beyond the use of the effect of services for deriving the semantics of the action performed.

Furthermore, OASIS Reference Model [160] considers the concept of a service as a core element that enables a requester to access and achieve a particular *business capability*. Within this vision, we notice that the concept of service has evolved from the notion of remote invocation interface (such as WSDL [35]) to a more comprehensive entity [227]. The invocation interface is only one aspect of the whole service description. Another core aspect of a service, is the notion of **business capability** which describes what a service can do.

The notion of business capability is a fundamental concept not only for SOA (Service Oriented Architecture) but also for enterprise information systems. The ARIS architecture [199] recognizes the importance of the functional perspective in enterprise information systems and considers it as one of its views. The concept of capability is the glue point between services and business processes. A service gives access to a certain business capability which can be achieved by a business process. Despite its importance, this concept has not drawn the attention from the research community as it deserves. Current approaches for capability modeling were part of efforts to describe related concepts such as business processes, service descriptions and search requests.

In the literature, three families of approaches can be defined that tackled the problem of business capability modelling either directly or indirectly. The first family includes semantic Web services models (WSMO [183] and OWL-S [144]) which model capabilities as Input, Output, Preconditions and Effects (IOPE paradigm). Modelling business capabilities following the IOPE paradigm that does not feature in an explicit and easily accessible way domain-specific parameters.

The second family of related efforts concerns semantic annotations of invocation/interaction interfaces (SA-WSDL and SA-REST) [111, 125]. While these approaches do not directly target business capability modelling, they attempt to provide alternative solutions to top-down semantic approaches (WSMO and OWL-S) by starting from existing descriptions such as WSDL and annotate them with semantic information. These approaches define a semantic (business) description of syntactic interaction interfaces rather than actual capabilities.

The third family includes frame-based approaches for modelling capabilities. Oaks et al. [159] provided a comprehensive overview of related approaches and proposed a model for describing service capabilities following the same principle. The proposed model distinguishes in particular the corresponding action verb and informational attributes (called roles in the paper [159]) in addition to the classical IOPE. While this model makes a step beyond the classical IOPE paradigm, the semantics of capabilities remain defined via the IOPE paradigm and therefore has the same problems as the first family of approaches described above.

To give a high level definition, a business capability is a descriptor (or a specification) of constraints (i.e. similar to IOPEs in classical approaches) which can be a stand-alone entity, but which could lead into a "service-level agreement". In this work, I share the same vision of Oaks et al. [159] and argue that it is very hard to model business capabilities in terms of IOPEs only. Thus I propose in this chapter a novel business capability meta-model that focuses on the aspects of interest which characterise the carried out action. The model uses a frame-based representation of business capabilities as it is an expressive and flexible means to represent business capabilities [239, 240]. The proposed conceptual model is detailed in Section 3.2. A possible realization of the model is shown in Section 3.3. The model has been validated in Section 3.4 using ontological and feature-based evaluations as non-empirical methodologies and semi-structured interviews with domain experts as empirical evaluation. Finally, Section 3.5 concludes the chapter and identifies future perspectives.

3.2 Business Capability Conceptual Model

3.2.1 Capabilities as Property-featured Entities

I propose to model a business capability as a **category** (of functionalities) enriched by (zero or many) **functional or non-functional features** (see the concept of property entry below). These features refine the given **category** by giving more details about **aspects of interest** of the corresponding **action**.

More formally, in the proposed model capabilities are defined as a `Category` and a set of `property entries` (see Definition 2). A `property entry` as described in Definition 1 (see also Example 1) is a couple *(property, value)* where *property* is a **domain-specific functional feature** or a **domain-independent non-functional property** and *value* is the value or the possible *values* that a property can have. Both *property* and *value* refer to ontological terms.

Definition 1 (Property Entry, Property Declaration and variantOF).

- A `property entry` *(P, v)* is specified with respect to a `property declaration` defined in a shared ontology.
- A `property declaration`, $d = (P,V,R)$, defines (i) a property P as a relevant functional or non-functional feature of the capabilities of a given domain, (ii) V the most general value (super class) that property entries defined according to d can have and (iii) a set of relations R that tell when a value v_1 is more specific than a value v_2 with respect to the semantics/meaning of the property P.

- Let v_1 and v_2 be two values and R a set of specification relations. v_1 `variantOF` v_2 if $\exists r \in R$ such that v_1 `r` v_2.

Example 1 (Property Entry and Property Declaration). This example shows a single property declaration from the shipping domain [1]. A shipping capability includes as part of its description the property *From* that indicates where (location) the shipment is taken from. This property is defined with respect to the property entry d_{From} = *(From, GeographicalLocation, LocatedIn)* where *From* is the actual property, *GeographicalLocation* is the most general value of this property and *LocatedIn* is the specification relation that exists between possible values of this property. $FROM_{EU}$ = *(From, Europe)* and $FROM_{IE}$ = *(From, Ireland)* are two property entries defined with respect to d_{From}. Please note the relation *Ireland* ***LocatedIn*** *Europe*.

Definition 2 (Business Capability). A couple *Cap = (Category,Properties)* is a business capability, where:

- *Category*: This concept is similar to [159] that defines, in a natural language, what is the action being described. Different to [159], I consider the category as a concept from a domain related ontology that comes form a shared agreement on its semantics. A category is a specific property that is present in all business capability descriptions via the property **achieves**.
- *Properties*: Represents a set of pairs *(Property, Value)* that correspond to the set of features of the business capability as shown in Definition 1.

A business capability can be linked to the so called *"case frames"* in linguistics as it is offered by FrameNet project [11]. In linguistic study, the capability of an action is defined by an *action verb* that is quite similar to the *action category* in my proposed model. Then several dimensions may extend that verb by giving more details about the carried work and related aspects. While in FrameNet these dimensions are predefined such as: agentive, dative, objective, etc. in my case I do not impose any predefined dimensions. However, it is possible to establish links between both models to generate structured business capabilities from textual descriptions using linguistic case frames. Such idea has been discussed by Leopold et al. [131], Rahm and Bernstein [175] and Mendling and Simon [149] for automatically annotating process activities with their action verbs derived from the textual documentation. One step further, Gao and Bhiri [81] explore the idea of mapping case frame dimensions to capability properties to generate a full structured capability using the model proposed in this thesis.

[1] Please note that the notation used in this example is not formal, formal descriptions of the used concepts will be shown in Section 3.3

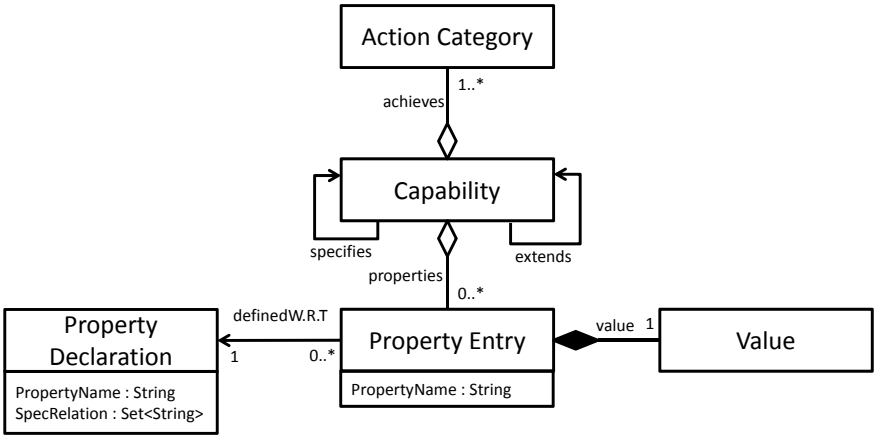


FIGURE 3.1: Business Capability UML Class Diagram

In order to give an object oriented conceptual view of the proposed model, I refer to the UML class diagram shown in Figure 3.1. This diagram contains the following classes:

- *Business Capability* is the class that captures the business capability (see Definition 2). This class is composed of at least 1 *Action Category* and, optionally, multiple *Property Entries*.
- *Action Category* is the class that represents the category of the business capability (see Definition 2).
- *Property Entry* is the class that represents a property entry that has a name and a *Value* (see Definition 1). A property Entry is defined with respect to a *Property Declaration* (the connection between both classes the the same property name).
- *Property Declaration* is the class that represents the property declaration (see Definition 1). It has a property name, the most general *value* and a set of specification relations (see Definition 1).
- *Value* is the class that represents the value of a property (see Definition 1). A separate class for values is needed to create complex types that are depicted in Figure 3.2. These types are:
  - *EnumerationValue* extends the class *Value* to represent enumerations of other values (see relation *hasElement* in Figure 3.2).
  - *RangeValue* extends the class *Value* to represent ranges of other values presented as minimum and maximum (see relations *hasMin and hasMax* in Figure 3.2).
  - *DynamicValue* extends the class *Value* to represent a dynamic value that is evaluated with respect to an Expression (see relation *hasEvaluator* in Figure 3.2).

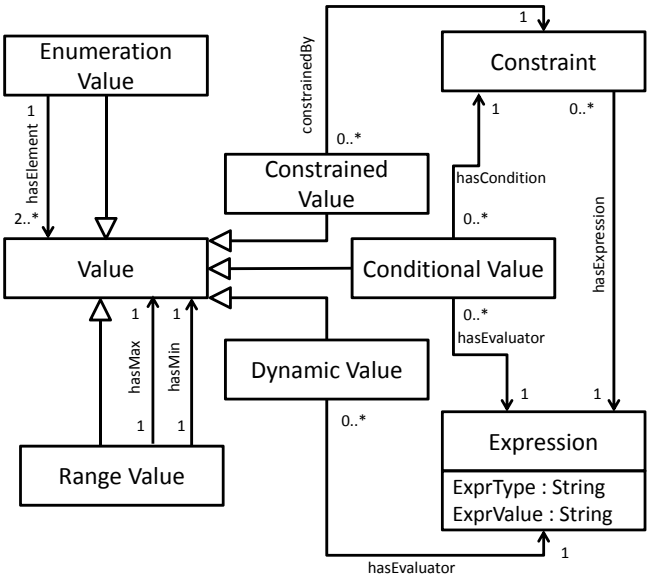


FIGURE 3.2: UML Class Diagram for the Possible Values for "Property Entry"

- *ConstrainedValue* extends the class *Value*, it has a value only if a certain *Constraint* is valid (see relation *constrainedBy* in Figure 3.2).
- *ConditionalValue* extends the class *Value*, it has a value only if a certain *Constraint* is valid (see relation *hasCondition* in Figure 3.2). Additionally, its value is computed dynamically with respect to an Expression (see relation *hasEvaluator* in Figure 3.2).
- *Constraint* is the class that represents a constraint that is defined via an *Expression* (see relation *hasExpression* in Figure 3.2).
- *Expression* is the class that represents an expression, it has a type and a value (both of type String).

The idea of modelling capabilities as a set of features was highly influenced by the frame-based modelling paradigm. Indeed, the conclusions of Wickler [239, 240] after an extensive analysis of multiple modelling mechanisms and languages suggests that frame-based descriptions of capabilities in the context of software agents were the most expressive and flexible means. Additionally, using the model suggested in this chapter one can describe capabilities independently from their actual implementations by highlighting the action being performed with a set of related properties. Contrary to the Input, Output, Precondition and Effect paradigm, it features the functional (business) and non-functional characteristics which end-users are mostly interested in and which are specified in their requests. This is natural for users to describe their needs. For example, users need a *service that* ***ships packages from*** *an address* ***to*** *another*. This

is expressed in the model via the action category "ship" and the properties "package", "from" and "to".

3.2.2 Specification and Extension Relations Between Capabilities

Another benefit of using frame-based modelling of capabilities is the possibility to infer potential relationships between them by analyzing their features (or properties). This has particular interested in relation to *specifications* and *extensions* that may exist between business capabilities. These relations are captured in Figure 3.1 as "*specifies*" and "*extends*" between capabilities and are described respectively in Definitions 3 and 4.

Please note:

- For abbreviation purposes I say that a certain business capability *cap* has a property *pr*.
- I refer to the property *pr* of *cap* by *cap.pr*.
- I refer to the set of properties of *cap* by *cap.properties*.
- I say that two capabilities $\mathcal{C}_1$ and $\mathcal{C}_2$ (or more) share the same property *pr* if both of them have the property *pr* (but possibly with different values).

Definition 3 (specifies). Given two capabilities $\mathcal{C}_1$ and $\mathcal{C}_2$, $\mathcal{C}_1$ `specifies` $\mathcal{C}_2$ if (i) all the properties of $\mathcal{C}_2$ are also properties of $\mathcal{C}_1$ (In other terms $\mathcal{C}_1$ inherits all the properties defined in $\mathcal{C}_2$), (ii) for every shared property *pr*, the value of $\mathcal{C}_1.pr$ is either equal to or `variantOf` the value of $\mathcal{C}_2.pr$, and (iii) there exists at least one shared property *pr'* such that the value of $\mathcal{C}_1.pr'$ `variantOf` $\mathcal{C}_2.pr'$. (see `variantOF` in Definition 1).

Definition 4 (extends). Given two capabilities $\mathcal{C}_1$ and $\mathcal{C}_2$, $\mathcal{C}_1$ extends $\mathcal{C}_2$ if (i) $\mathcal{C}_1$ has all the properties of $\mathcal{C}_2$ and has additional properties, and (ii) for every shared property *pr*, the value of $\mathcal{C}_1.pr$ is equal to the value of $\mathcal{C}_2.pr$.

Fine-grained relations can be defined based using these relations. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two capabilities such that $\mathcal{C}_2$ specifies $\mathcal{C}_1$, and let *pr* be a shared property, we say that $\mathcal{C}_2$ specifies $\mathcal{C}_1$ on *pr*, denoted $\mathcal{C}_2$ specifies_pr $\mathcal{C}_1$ iff the value of $\mathcal{C}_2$.pr is `variantOf` the value of $\mathcal{C}_1$.pr.

The relations `specifies` and `extends` (fine or coarse-grained) enable organizing a repository of capabilities as a hierarchy [52, 83] as shown in Example 2 and illustrated on Figure 3.3 and 3.4.

Example 2 (Hierachy of Capabilities). Figure 3.3 and 3.4 show an example of a hierarchy of business capability descriptions[2]. *Capability A* is the root of these hierarchies, it represents an abstract business capability description for shipping goods from any source and destination at an international scale. This business capability can be extended either to *Capability B* or *Capability C*. Both extend the initial business capability by 1 or 2 attributes; fine-grained relations can be seen in Figure 3.4. As an example of specification relation between capabilities, I refer to the link between *Capability D* and *Capability B* in Figure 3.3. *Capability D* specifies *Capability B* as it becomes a European shipping capability instead of International. This fine-grained semantics of the specification relation is further shown in Figure 3.4. It is also clear that *Capability E* extends *Capability D*.

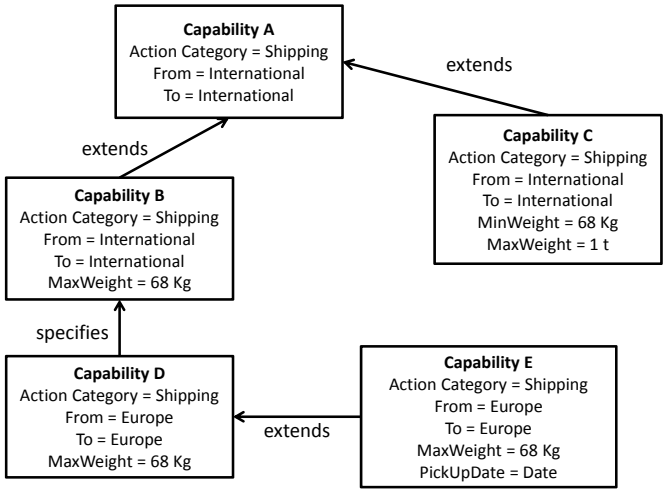


FIGURE 3.3: Example of Shipping Capabilities Hierarchy using Coarse-grained Relations

Please note that the hierarchies depicted in Figure 3.3 and 3.4 are simple and can be easily created manually. However, when it comes to large set of capabilities, more dedicated algorithms for creating optimal hierarchies are needed [52].

3.2.3 Configurable Business Capabilities

The goal of configuration in general is to arrange functional units according to their nature, number and characteristics in order to provide a tailored hardware, software or documentation [101]. The configuration of a business capability consists of arranging its properties and values in order to define a desired business capability. Giving the user the freedom to arrange a business capability's properties might lead to incomplete or inconsistent business capability descriptions (e.g., missing properties, wrong values, etc.).

[2]Please note that the notation for this example is not formal, formal descriptions of the used concepts will be shown in Section 3.3

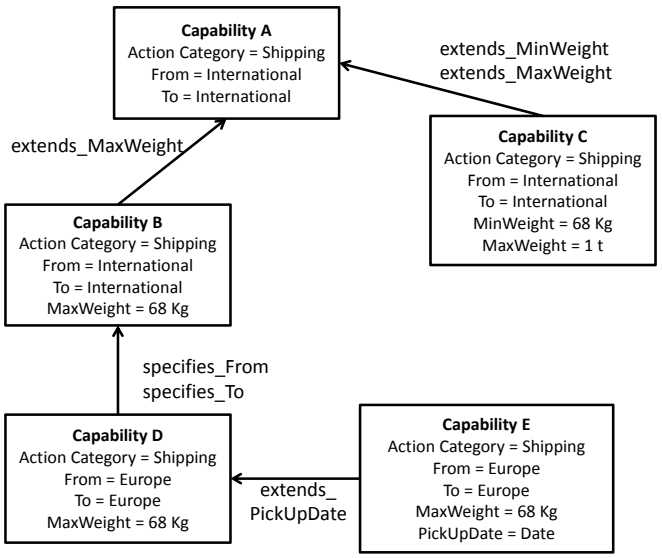


FIGURE 3.4: Example of Shipping Capabilities Hierarchy using Fine-grained Relations

To avoid this problem, predefined configurable business capabilities can be proposed with explicit configuration options. This section defines formally a configurable business capability and shows how it is captured in the proposed conceptual model.

While frame-based modelling have been successfully used for representing and understanding structured entities [155], it can also be used for modelling configuration options among entities. To this end, a configurable business capability can be used as a initial entity and operates as a reference business capability. Each configuration step allows to further refine this reference business capability. A configurable business capability adds the possibility to remove a particular property if it is optional (i.e., represents an explicit configuration point for optional properties) or select the value of a particular property if multiple values are proposed (i.e., represents an explicit configuration point for a property value). This can be translated in the meta-model as shown in Figure 3.5: an additional attribute (namely Configurable) that indicates if a particular property is configurable and an additional attribute (namely Configurable) that indicates if a particular property value is configurable. Definition 5 formally defines a configurable business capability.

Definition 5 (Configurable Business Capability). A triple *CCap = (Category,Properties,CPr, CVPr)* is a configurable business capability, where:

- *Category* and *Properties* are defined as in Definition 2.
- *CPr* is a function that indicates if a particular property $pr \in Properties$ is configurable. $CPr : pr \rightarrow CPr(pr)$
  $pr \mapsto \{true, false\}$

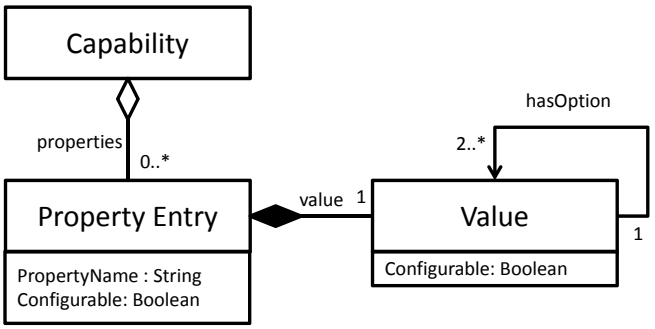


FIGURE 3.5: Configurable Business Capability Meta-Model

- $CVPr$ is a function that indicates if a particular property has a configurable value.
  $CVPr : pr \rightarrow CVPr(pr)$
  $pr \mapsto \{true, false\}$

To summarize the conceptual effort in proposing a business capability meta-model, Figure 3.6 shows the complete UML class diagram that integrates all the parts that have been discussed in this Section.

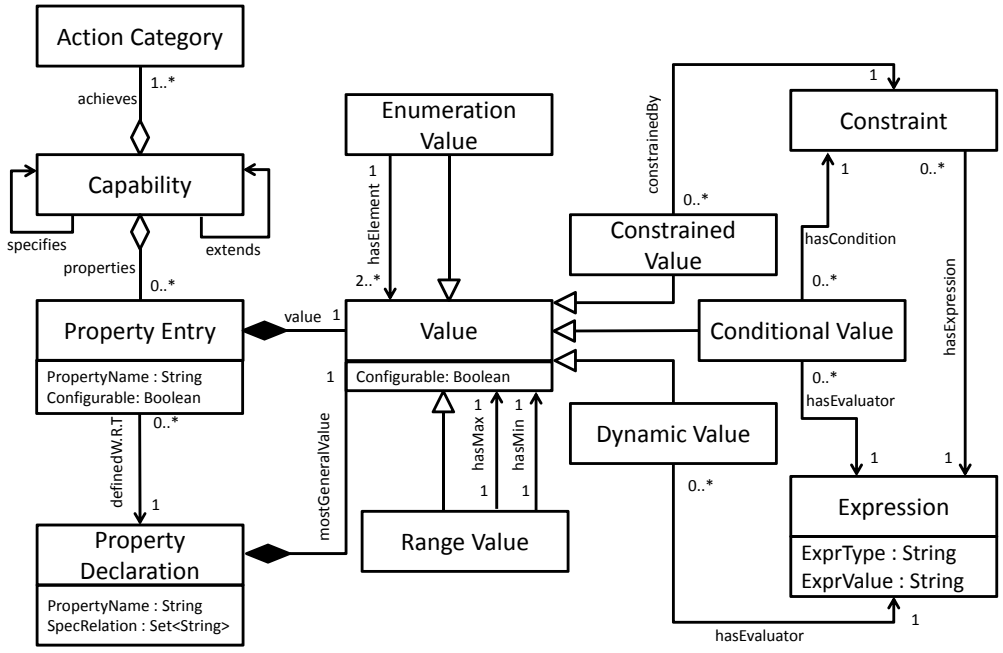


FIGURE 3.6: UML Class Diagram: Complete Business Capability Meta-Model

The proposed conceptual model allows to model high level business capabilities in a particular domain that can be tailored to specific use cases. Similar to domain ontologies which define shared concepts and shared attributes/properties, high level business capabilities in a given domain can also be defined as an ontology where an agreement about their meaning is reached and shared. Like any other ontology concepts, these

business capabilities can be reused to define other ones. I implemented this model as a set of ontologies that are detailed in the following section.

3.3 Realisation

A recent trend in the adoption of Linked Data [24, 203] principles, and a growing amount of data sets specified in RDF. It is clear that Linked Data provides a best practice for publishing structured data on the Web. Linked Data is published using RDF where URIs are the means for referring tp various entities on the Web giving the possibility to interlink them. Currently, organizations are highly interested in publishing their data in RDF [126] as well as various public vocabularies (ontologies) are being released. Consequently, I have chosen to implement the proposed business capability meta-model in RDF and make use of Linked Data principles in order to define business capabilities, categories and properties as well as their values.

As illustrated in Figure 3.7, I distinguish four levels of ontologies for implementing the proposed business capability model:

- *Meta-Model*: is the lowest level that defines the required classes and properties for defining action categories (see Section 3.3.1) and property declarations (see Section 3.3.2).
- *Categories and Properties*: this level defines the set of categories (see Section 3.3.1) and properties (see Section 3.3.3) related to particular domain.
- *Domain Ontology*: is the actual business capability domain ontology. It creates abstract business capabilities that associate for each action category the possible set of properties (see Section 3.3.3).
- *Business Capabilities*: at this level business capabilities are created with respect to the business capability domain ontology.

3.3.1 Action Categories

The action category meta-model proposes the set of classes and properties for defining the actions being performed in a particular domain. Figure 3.1 did not give any details about the *ActionCategory* class as it can be defined based on the implementation choices.

After analysing existing actions categories such NAICS [161], UNSPSC [94] and the MIT Process Handbook [146], I propose to model action categories and relations between them

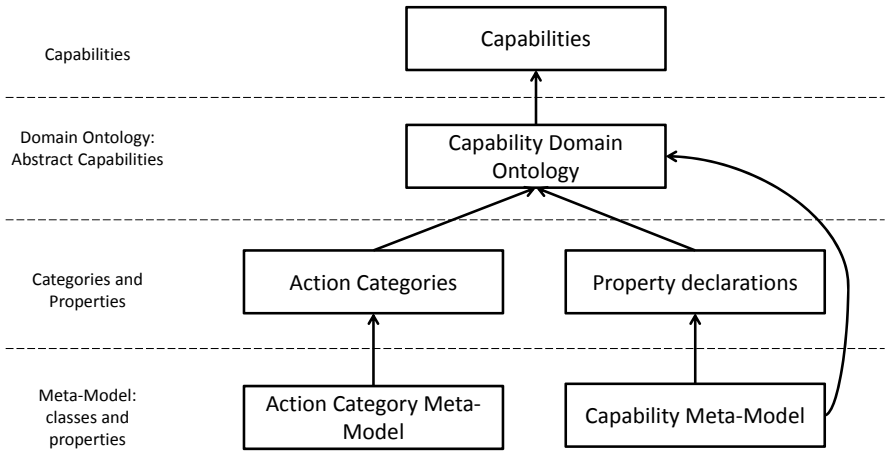


FIGURE 3.7: Four Levels of Ontologies: From Meta-Model to Actual Business Capabilities

(i.e., Meronymy and generalisation relations). Meronymy (part-of) relations between action categories are used in NAICS [161] and the MIT Process Handbook [146] and also investigated in [208] in order to capture granularity relations between actions at several levels of abstraction. The generalisation relation is also used in NAICS [161] to represent that an action can be more specific or general than another one (e.g., "book accommodation" is more general than "book hotel").

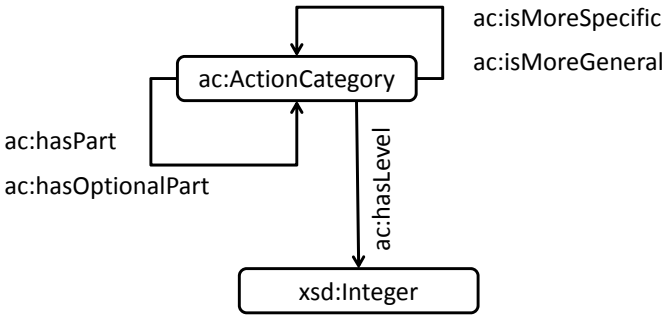


FIGURE 3.8: Action Category Meta-Model

Figure 3.8 illustrates the proposed meta-model for action categories. I use in this ontology the prefix *ac* for the namespace `http://vocab.deri.ie/ac`[3]. This ontology has only one rdfs class (rdfs:Class) and five rdf properties (rdf:Property). *ac:ActionCategory* is the class of action categories. The properties *ac:hasPart* and *ac:hasOptionalPart* are used to create a meronymy hierarchy of action categories. An action is performed only if all its parts are performed. I use the property *ac:hasLevel* to assign the level (xsd:Integer) of an action category within a meronymy hierarchy of action categories. The top of the hierarchy starts with 0. The properties *ac:isMoreGeneral* and *ac:isMoreSpecific* (inverse relations) are also used to create a specification hierarchy of action categories.

[3]Please note that I also use the prefix *xsd* for the namespace `http://www.w3.org/2001/XMLSchema#`

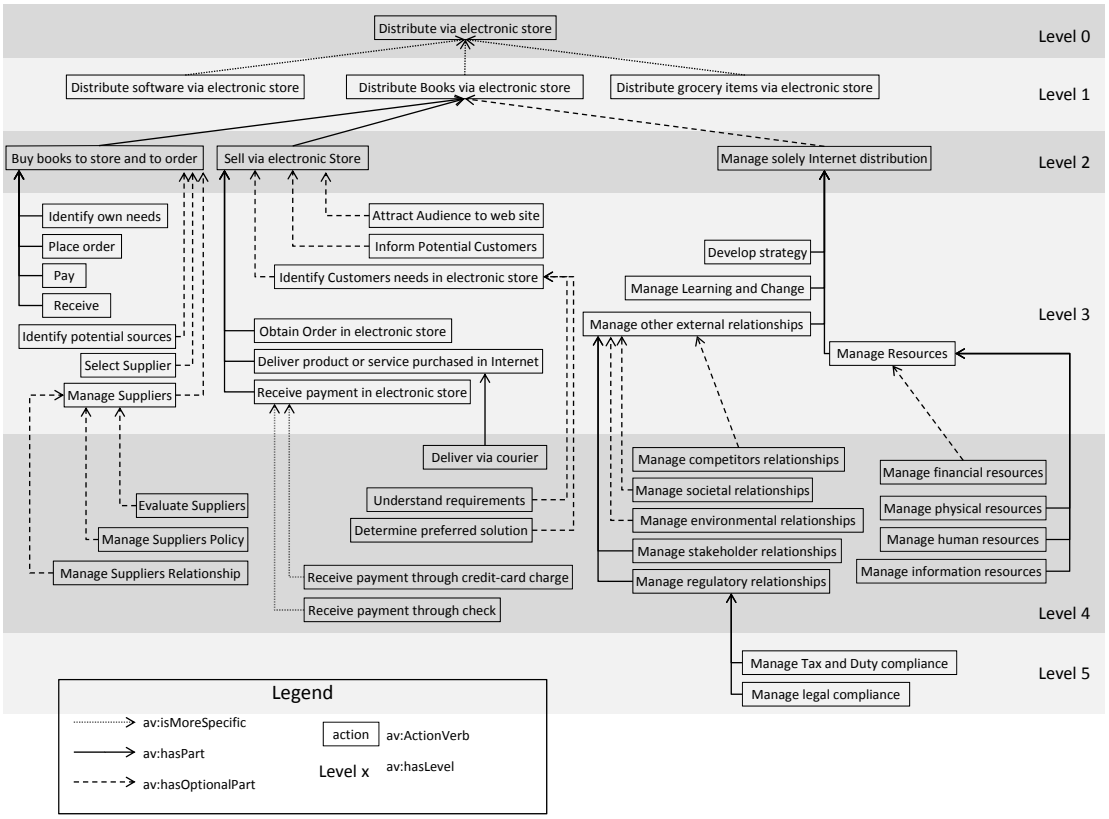


FIGURE 3.9: Example of Action Categories from the Distribute via Electronic Store Domain [96]

```
1  @prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
2  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
3  @prefix ac: <http://vocab.deri.ie/ac#>.
4  @prefix deso:  <http://.../AContology/businessTravel#>.
5
6  deso:distributeBooksViaElectronicStore a ac:ActionCategory;
7          ac:hasLevel ''1''^^xsd:Integer;
8          ac:hasPart deso:buyBooksToStoreAndToOrder, deso:sellViaElectronicStore;
9          ac:hasOptionalPart deso:manageSolelyInternetDistribution;
10         rdfs:label ''Distribute books via electronic store''^^xsd:String.
```

LISTING 3.1: Action Categories Ontology Snippet from the Distribute via Electronic Store Domain

This meta-model is used to create **domain specific action categories** (Called in this thesis "*action categories ontology*") and explicitly capture composition and generalization relations between them. To illustrate how it can be used, I translated an example of the action categories ontology from the MIT Process Handbook [146] of the *Distribute via Electronic Store* domain [96]. The resulting ontology is shown in Figure 3.9 and Listing 3.1 shows the action category *deso:distributeBooksViaElectronicStore* using N3 representation. This action category ontology has been manually generated, however, it is important to notice that the content of MIT Process Handbook [146] can be automatically parsed to build action categories.

3.3.2 Business Capability Meta-Model

Listing 3.2 shows the concept *cmm:Capability* as an *rdfs:Class* and *cmm:PropertyValue* as an equivalent class to *owl:Thing* because it needs to be the most general class to allow for the reuse of existing vocabularies for possible attribute values. For example using *vcard* open vocabulary for defining addresses. Then, the property *cmm:achieves* allows linking a *cmm:Capability* to its corresponding *ac:ActionCategory*. Furthermore, the property *cmm:property* allows to create domain specific properties that will be created as an *rdfs:subProperty* of *cmm:property* and will be interpreted as properties of a business capability. Finally, the property *cmm:hasMostGeneralValue* is used to define its most general value during the property declaration.

Please note that Listing 3.2 is not a complete listing of the Business Capability Meta-Model, further details can be found in the online version[4].

```
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix owl:  <http://www.w3.org/2002/07/owl#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix cmm:  <http://vocab.deri.ie/cmm#>.

cmm:Capability a rdfs:Class.

cmm:PropertyValue owl:equivalentClass owl:Thing.

cmm:achieves a rdf:Property;
                        rdfs:domain cmm:Capability;
                        rdfs:range ac:ActionCategory.

cmm:property a rdf:Property;
                        rdfs:domain cmm:Capability;
                        rdfs:range cmm:PropertyValue.

cmm:hasMostGeneralValue a rdf:Property;
                        rdfs:domain cmm:property;
                        rdfs:range cmm:PropertyValue.
```

Listing 3.2: Business Capability Meta-Model Snippet

3.3.3 Property Declarations and Business Capability Domain Ontology

One can define a domain specific business capability ontology by modelling its action category and properties. I discuss in this section the various property types that our model supports. I will use the Distribute via the Electronic Store domain (its set of action categories has already been discussed in the previous section) for defining the property declarations that are related to the action categories shown in Figure 3.9.

[4]Online version of cmm available at: http://vocab.deri.ie/cmm as accessed on 06/06/2015

Listing 3.3 shows the N3 description of the business capability *desco:deliverViaCourrier* (see Line 5). Line 6 links this business capability to its action verb from the Action Categories of the domain which is *deso:deliverViaCourrier*. The rest of the listing illustrates how two properties *desco:from* and *desco:to* are defined. Both properties are *rdfs:subProperty* of *cmm:property* (Line 10 and 12).

```
1  @prefix vcard: <http://www.w3.org/2006/vcard/ns#>.
2  @prefix cmm:   <http://vocab.deri.ie/cmm#>.
3  @prefix desco:  <http://.../DContology/businessTravel#>.
4
5  desco:deliverViaCourrier a cmm:Capability;
6          cmm:achieves deso:deliverViaCourrier;
7          desco:from vcard:VCard;
8          desco:to vcard:VCard.
9
10 desco:from rdfs:subProperty cmm:property;  rdfs:range vcard:VCard.
11
12 desco:to rdfs:subProperty cmm:property;  rdfs:range vcard:VCard.
```

LISTING 3.3: Shipping Domain Ontology Snippet

The range of the properties *desco:from* and *desco:to* is *vcard:VCard* to refer to an address. More complex and detailed property types are required for modelling more advanced properties. As depicted in Figure 3.2, I consider five classes used for describing the values of a business capability properties.

Using these classes separately or in combination, a business capability can specify (i) the possible values properties can have, and (ii) how to compute their values. Before detailing these classes, I introduce the concepts of *Constraint* and *Expression* to which some attribute values may refer.

3.3.3.1 Constraint and Expression

A constraint can specify the possible values an attribute can have. The class *Constraint* represents all constraints. The class *Expression* enables expressions including the value of a given constraint. The class *Expression* has two attributes/properties, *ExprType* which specifies the type of the expression and *ExprValue* which defines the expression itself. The type of the expression, *ExprType*, indicates how to build the corresponding queries during a matching process. Currently, the only type of expression my meta-model supports is SPARQL (queries).

Listing 3.4 shows an example for expressing a constraint on the weight of the package. The constraint *PackgConstraint* is defined in line 1. This constraint has an expression of type SPARQL. The value of the constraint expression (line 5) indicates that the weight of the package has to be lower than or equal to 50 Kg.

```
1 desco:PckgConstraint a cmm:Constraint; cmm:hasExpression desco:PckConstraintExpr.
2
3
4 desco:PckgConstraintExpr  a  cmm:Expression;  cmm:exprType "SPARQL";
5             cmm:exprValue "?weight =< 50? && ?weightUnit = dbpedia:KG".
```

LISTING 3.4: Example of a Package Constraint

3.3.3.2 Constrained Value

The class *ConstrainedValue* defines the possible values a property can have by specifying a set of constraints on its value. As depicted in Figure 3.2, a *ConstrainedValue* is constrained by a set of constraints. Listing 3.5 shows how *desco:deliverViaCourrier* can specify that it can deliver packages of weight under 50 Kg. The value *X*, of the property *Item*, is a *ConstrainedValue* (lines 1 and 3). X is constrained by the constraint *PckgConstraint* (line 5) which was detailed in Listing 3.4.

```
1 desco:deliverViaCourier  desco:Item  :X.
2
3 :X  a  desco:Package, desco:ConstrainedValue;
4       desco:hasWeight [desco:hasValue ?weight; ship:hasUnit ?weightUnit];
5       cmm:constrainedBy  desco:PackgConstraint.
```

LISTING 3.5: Example of a Courier Constrained Value

3.3.3.3 Dynamic Value

A *DynamicValue* defines how to compute the value of a property where the value depends on (i) consumer provided properties, (ii) dynamic values, or (iii) hidden variables. As shown in Figure 3.2, a *DynamicValue* refers to an expression that defines how to compute it.

Listing 3.6 shows an example of how to compute the shipping price. The value *Y*, of the property *price*, is a *DynamicValue*. It has as evaluator an expression *PriceExpression* (lines 4) which is a SPARQL expression (line 6). Line 7 specifies the formula for computing the price based on the weight of the package.

```
1 desco:deliverViaCourrier  desco:price  :Y.
2
3 :Y  a  desco:ShippingPrice, cmm:DynamicValue;  desco:hasValue  ?price;
4    desco:hasUnit  dbpedia:USD;  cmm:hasEvaluator desco:PriceExpression.
5
6 desco:PriceExpression a cmm:Expression;  cmm:hasType  "SPARQL";
7                 cmm:exprValue  "?price := fn:ceiling(?weight)*5.5+41".
```

LISTING 3.6: Example of a Dynamic Value for computing the shipping price of a Package

3.3.3.4 Conditional Value

A *ConditionalValue* assigns a value to the corresponding property if a certain condition holds. As shown in Figure 3.2, a *ConditionalValue* has a condition expressed as a constraint and an element which corresponds to the relevant property value.

Listing 3.7 gives an example showing how to specify a shipping price when the target country is a European country. The value Y, of the property *price*, is a *ConditionalValue* (lines 3). It assigns the Property Value, *EuropeanPrice*, when the *PriceCondition* holds (line 4). *PriceCondition* is a *Constraint* which requires that the target country is in Europe (Lines 7-9). *EuropeanPrice* is a *DynamicValue* (line 11) and has as evaluation expression *PriceExpression* which was detailed in Listing 3.6.

```
1  desco:deliverViaCourrier  desco:price  :Y.
2
3  :Y  a  desco:ShippingPrice, desco:ConditionalValue; desco:hasValue ?price;
4         cmm:hasCondition desco:PriceCondition; cmm:hasElement :EuropeanPrice.
5
6
7  desco:PriceCondition a cmm:Constraint;
8       cmm:hasExpression [cmm:hasType "SPARQL"; cmm:hasValue "?trgCountry
9                          skos:subject dbpedia-cat:European_countries"].
10
11 desco:EuropeanPrice a cmm:DynamicValue; cmm:hasEvaluator desco:PriceExpression.
```

Listing 3.7: Example of a Conditional Value

3.3.4 Tool Support

As it is difficult for new users to write RDF annotations for describing the business capability of their services or business processes. I have implemented an extension of EPCTools, a business process modelling tool using Event-driven Process Chains (EPC), in order to assist users in the annotation operation. Figure 3.10 shows screenshots of the extended version of EPCTools. When users want to annotate a particular task of a model, they need to right click it and choose *Capability* from the contextual menu (see 1 on Figure 3.10). After loading the required ontologies, they have to select the action category for the current task (see 2 in Figure 3.10). Finally, they have to choose what properties associated to the action category they want to define in the business capability (see 3 in Figure 3.10).

The extended version of EPCTools offers two options to save the model either in its existing EMPL serialisation with additional tags for the business capability or to export the entire model as RDF. Please note that the primary purpose of this research is not to propose this UI, it has been developed only for testing the applicability of the proposed model. A proper evaluation of the model itself is carried out in the following section.

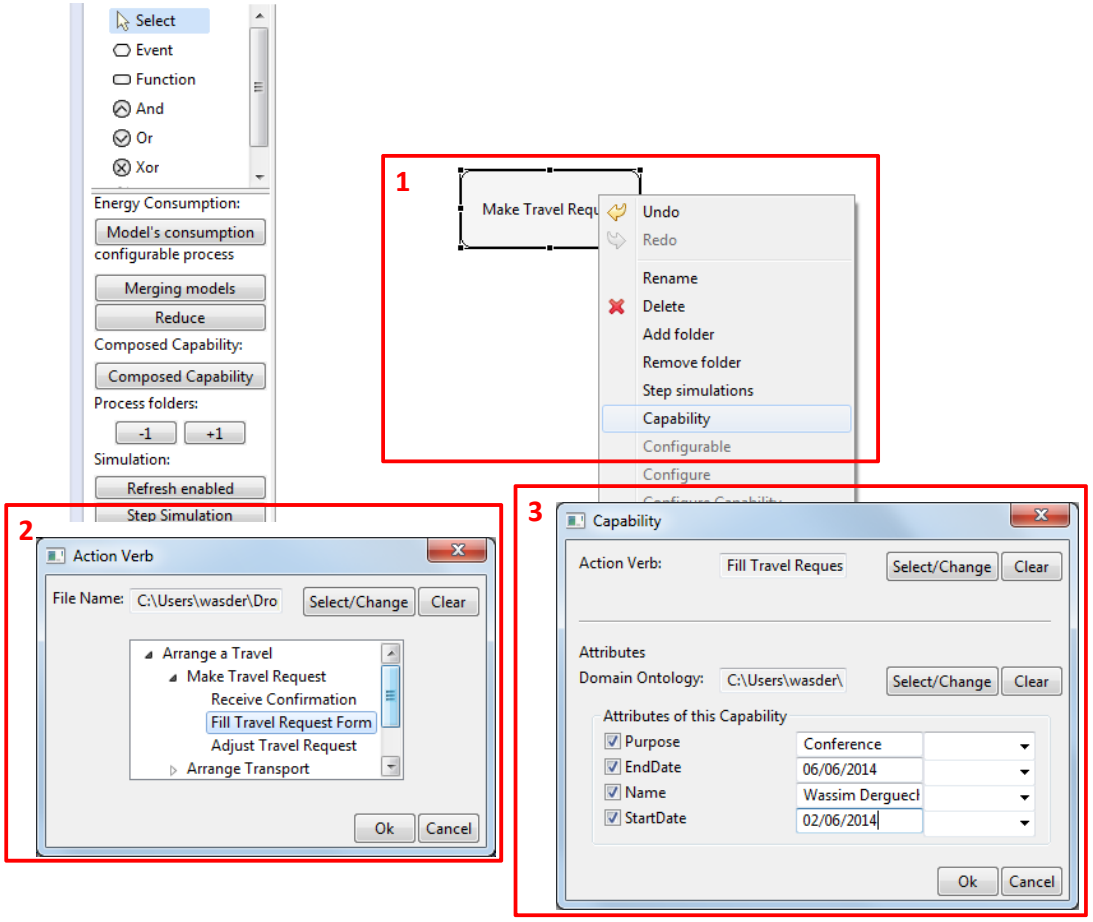


FIGURE 3.10: Extended Version of EPCTools that Supports the Annotation of Business Process Tasks with their Business Capabilities

3.4 Evaluation of the Meta-Model

Various evaluation techniques have been proposed in the literature for assessing the appropriateness of a modelling approach. Recker [177] and Siau and Rossi [206] discuss various evaluation methods for conceptual modeling languages in order to assess their applicability in given modeling contexts towards more rigor in conceptual modelling research [177]. Table 3.1 and 3.2 list these evaluation methodologies as well as a discussion on their applicability in this work.

After verification of the applicable evaluation methodologies, I propose to validate the business capability model proposed in this chapter using non-empirical evaluation methods using (1) ontological evaluation (see Section 3.4.1) and (2) feature evaluation (see Section 3.4.2) and an empirical method using semi-tructured interviews with domain experts that I am reporting in Section 3.4.3.

TABLE 3.1: Overview of the Empirical Methods of Evaluation of Conceptual Models (table modified from [177] and [206])

Evaluation Method	**Comment**
Evaluation methods used in this chapter	
Survey: using questionnaires to gather human assessment, attitude, opinion, etc. on the proposed model (e.g., [25]).	Both structured and semi-structured interviews/questionnaires can be used for assessing conceptual models. Structured interviews or questionnaires are applicable in a large scale contexts where multiple users can be interviewed [62]. In this research, target users are business experts that are using process modelling tools in their work. Such profiles are not easy to approach and large numbers of interviewees cannot be easily found. Consequently, I aimed to use semi-structured interviews with reduced number of domain experts [62]. Section 3.4.3 reports on results of these interviews.
Evaluation methods NOT used in this chapter	
Laboratory experiment: comparing the proposed model with different models with respect to observed metrics such as cost, execution time, etc (e.g., [14]).	In this chapter, the model has not been used in a context where observable quantitative metrics can be used. This evaluation method will be applied in Chapter 5 where metrics such as execution time and search space size can be observed.
Case study: observing a group of users while using of the model and investigate results without intervening (e.g., [77, 92]).	This method is applicable if the model could have been brought into a end-users environment. Unfortunately, I did not have access to such opportunity.
Action research: testing of the idea developed in an academic environment in real world situations with the presence of the researchers (e.g., [154]).	Same as *Case study*.

TABLE 3.2: Overview of the Non-empirical Methods of Evaluation of Conceptual Models (adapted and modified from [177])

Evaluation Method	**Comment**
Evaluation methods used in this chapter	
Ontological evaluation: mapping the modelling language constructs with ontology constructs and verify that the model is able to represent reality (e.g., [244]).	This method is not specifically designed for evaluating ontology-based models/languages. Indeed, it has been used by Wand et al. [231] for the assessment of relationships in the Entity-Relationship model. This approach has been adapted in this work. Details for this evaluation are discussed in Section 3.4.1.
Feature comparison: comparing the proposed model to others using a set of predefined requirements (e.g., a checklist of features) (e.g., [243]).	I have already discussed in Section 2.2 of Chapter 2 related modelling contributions using the modelling requirements identified in this thesis. Furthermore, in Section 3.4.2 I will discuss the proposed model with respect to those requirements.
Evaluation methods NOT used in this chapter	
Paradigmatic analysis: analyzing the underlying assumptions of methods, for example the views and intentions of modelling methods (e.g., [100]).	Most of the literature contributions have been focusing on the modelling of **IT capabilities** and linking capabilities to their actual implementations as part of services descriptions or search requests. Different from this view, in this thesis, I am modelling **Business Capabilities** for describing actions of IT assets or human agents. This different view has been discussed along the entire thesis.
Meta modelling: comparing the proposed model with different models with respect to their language constructs to assess analogies and dissimilarities between them (e.g., [97]).	Most of the contributions found in the literature focus on modelling IT capabilities rather than Business Capabilities. Comparing their language constructs will not result in a valuable analysis, this makes the method not applicable.

TABLE 3.2: Overview of the Non-empirical Methods of Evaluation of Conceptual Models (adapted and modified from [177])

Evaluation Method	**Comment**
Metrics approach: comparing the proposed model with different models using a set of method metrics such the number of constructs, complexity of the expressions, etc (e.g., [185]).	In addition to the fact that other models are focused on IT capabilities while I focus on business capabilities, this method applies best when the models to compare are implemented using the same language. The use of RDF as an underlying implementation language is not commonly used by the reviewed models. Thus, the method is not applicable in this work.
Contingency identification: identifying particular cases where the proposed model works while other models fail (e.g., [43]).	Case studies and empirical tests are required to identify such contingencies. Besides, models evaluated using this method are generally researched for solving those particular cases or contingencies. This is not the case in this research.
Approaches based on cognitive psychology: investigating the impact of the cognitive psychology theories after adoption of the model (e.g., [180]).	I have excluded this method as it requires extensive knowledge about cognitive psychology theories.

3.4.1 Ontological Evaluation of the Business Capability Model

3.4.1.1 Introduction

The ontological evaluation of conceptual models consists of mapping the proposed conceptual model constructs to ontological concepts/constructs in order to assess the ability of the model to represent reality [244]. However, mapping the modelling language constructs to ontological concepts can be subjective especially when we want to identify intrinsic and non-intrinsic attributes or classes and kinds, consequently this can lead to a subjective evaluation. To avoid this issue, Wand et al. [231] propose to use a set of generic conceptual modeling constructs (i.e., instance, class and attribute) instead of the ontological constructs defined by Bunge [30]. In this approach, the evaluation of the model is carried out through the verification of a set of rules insuring that a model does not generate any semantic ambiguity by avoiding construct overload and redundancy.

These rules are not related to a particular conceptual model and can be applicable to any model that is using generic ontological concepts. Therefore, I evaluated the proposed business capability conceptual model using this methodology.

3.4.1.2 Evaluation Steps

The first step of the evaluation consists of mapping the business capability conceptual model constructs to the generic conceptual model constructs proposed by Wand et al. [231] (i.e., instance, class and attribute). Table 3.3 shows the mapping that is used for the second step of this evaluation that consists of verifying that the model respects a set of rules defined by Wand et al. [231].

TABLE 3.3: Mapping the Business Capability Conceptual Model Constructs to Generic Conceptual Modeling Constructs and Ontological Constructs

Business Capability Conceptual Model Construct	**Generic Conceptual Modeling Construct [231]**	**Ontological Construct [30]**
Capability	Class	Class
Action Category	Class	Class
achieves	Attribute	Connection Attribute
specifies	Attribute	Attribute representing a mutual property
extends	Attribute	Attribute representing mutual property
Property Entry	No direct representation	Property
Property Name	Attribute	Attribute representing intrinsic property
Configurable	Attribute	Attribute representing intrinsic property
Property Declaration	Class	Class
SpecRelation	Attribute	Attribute representing intrinsic property
Value	Class	Class
Range Value	Class	Class
hasMin	Attribute	Attribute representing intrinsic property
hasMax	Attribute	Attribute representing intrinsic property

Evaluation Rule 1. ***"Things are represented only as instances. Instances should represent only things."*** [231]

In the business capability modelling approach proposed in this work as well as in other modelling paradigms (e.g., Object Oriented Modelling), *things* represent objects representing instances of classes. Rule 1 implies that *things* cannot be instances of attributes or other instances. This is true for all the attributes shown in Table 3.3 except for *Property Entry*. In Wand et al.'s constructs, there is no direct representation of properties. The model that includes such constructs should define how it can be interpreted. In this work, for creating a domain specific business capability property, the model suggests to create an instance of a *Property Entry* that links a *Capability* instance to a certain *Value*. For this reason, in the implementation of this business capability meta-model, the *Property Entry* is defined as an *rdf:Property* (see *cmm:property* in Listing 3.2) and all business capability properties are created as *rdfs:subProperty* of *cmm:property* (see *desco:from* in Listing 3.3).

Evaluation Rule 2. ***"Both simple and composite things should be represented using the same construct (entity, object)."*** [231]

The business capability meta-model proposed in this thesis does not differentiate between simple and composite business capabilities. Both can be presented as instances of the class Capability.

Evaluation Rule 3. ***"A class or a kind of thing is defined in terms of a given set of attributes and relationships; that is, intrinsic attributes and mutual attributes."*** [231]

Instances of all classes in the business capability meta-model are defined by a set of attributes that are derived from intrinsic as well as mutual properties of the proposed model.

Evaluation Rule 4. ***"An aggregate type/class must have properties in addition to those of its component types/classes."*** [231]

This rule does not apply in the case of the business capability meta-model proposed here as there are no aggregated classes in the model.

Evaluation Rule 5. ***"All attributes and relationships in a class represent properties of things in the class."*** [231]

Instances of the classes proposed in the business capability meta-model are created as objects with a set of properties. These properties are either intrinsic or derived from mutual relation. Additional instances of business capabilities have additional attributes that are derived from the *Property Entry*.

Evaluation Rule 6. ***"Null attributes have no meaning."*** [231]

The business capability meta-model does not allow the creation of instances without attributes. Most importantly, it does not allow the creation of a business capability without at least specifying its *Action Category* (i.e., an attribute generated from a mutual property).

Evaluation Rule 7. ***"The same construct should be used to represent a binary relationship and a higher-order relationship."*** [231]

This rule is already covered by the mapping proposed by Wand et al.. Indeed, both binary and higher-order relationships are mapped to Attributes. Furthermore, the business capability meta-model proposed in this thesis does not have any higher-order relationships.

3.4.1.3 Discussion

The above mentioned rules (i.e., rule 1 to 7), as articulated by Wand et al. [231], verify that a general conceptual model can be used for modelling reality by avoiding construct overload and redundancy; which is the case with the business capability meta-model proposed in this thesis.

Researchers such as Bunge [30], Wand et al. [231] and Weber et al. [236] evaluate conceptual models from a realistic point of view. In other words, they assume that conceptual models should be designed to represent things that "exist in the world". The proposed business capability meta-model is, effectively, representing real things, i.e., actual actions performed by services, processes or human agents. The main observation is that business capabilities are not tangible objects and might need more flexibility to the give the designer the possibility to model these actions as he perceives them. Weber et al. [236] is in favor of such flexibility and argues that ontological foundations for information systems design might be subject to the perception of the designer. This is also aligned with paradigmatic approaches where conceptual models can capture different aspects of things depending on the intended perception and use of the conceptual models.

One of the advantages of using an ontological model for evaluation is that it is derived from a philosophical foundation [206]. This method can be challenging and difficult to apply with complex models, especially when using the ontological analysis of Bunge [30]. This can be simplified by using the method proposed by Wand et al. [231] to avoid situations where the designer needs to differentiate between a "kind" and a "class". The evaluation of the proposed business capability meta-model using this method was relatively simple as the model defines 14 constructs. This made the verification of the rules of Wand et al. [231] straightforward.

Even though in this work, I showed that the proposed business capability meta-model is "suitable to represent reality", further assessments of the generated business capabilities are required to ensure that the (1) they are consistent, (2) have an intuitive appeal to the end users, (3) they can be used to represent various domains and (4) can be used in empirical evaluations.

3.4.2 Feature Comparison Evaluation of the Business Capability Model

A feature comparison evaluation of conceptual models consists of comparing multiple models and investigate how they represent the same problem based on a set of units of analysis [177, 206]. This comparison has already been carried out in Chapter 2. In this section, I will further discuss how the proposed model satisfies the units of analysis used in Section 2.2.1 of Chapter 2. This evaluation methodology has also been carried out by Oaks for the evaluation of the proposed business capability meta-model in her thesis [158].

> ***Expressiveness*** - *Explicitly represent the action performed*: the actions performed by a business capability are captured via the *rdfs property* ***cmm:achieves***. The action categories are defined in an ontology of actions with composition and specification relations between them. This model can be further enriched by exploring other relations such as *synonymy*.
>
> ***Expressiveness*** - *Explicitly capturing functional and non-functional features related to the action performed*: contrary to the IOPE paradigm, my model expresses a business capability with a set of properties that can be both functional and non-functional. The related properties of a particular high level business capability are captured also in a domain specific ontology.
>
> ***Expressiveness*** - *Ability to express these features using simple as well as complex types*: a property value of a business capability can be assigned any simple value such as string, integer or an address vcard (see Listing lst:SDOnto). Furthermore,

the proposed business capability meta-model proposes a set of advanced types such as *conditionalValue*, *enumerationValue*, etc. (see Figure 3.2).

Inferences - *Ability to explicitly identify relationships between business capabilities based on their descriptions*: the proposed model identifies two relations to capture specification (see Definition 3) and extension (see Definition 4) relations between business capabilities. This chapter did not propose any algorithms for extracting these relations. Chapter 5 uses this feature to build an indexing structure of business capabilities based on their properties.

Use of Ontologies - *Use of domain or general ontological concepts for describing business capabilities*: actual business capabilities are derived from high level ones that are defined in domain specific ontologies. In the examples shown in this chapter, I illustrated the use of general and domain specific ontological concepts.

Often, feature comparison evaluation is seen a very subjective. Usually, researchers develop their own checklist of features to be used in this evaluation [206]. To avoid this problem, I used a predefined list of features/requirements identified by Sycara et al. [216] and used by Oaks et al.[159].

3.4.3 Interviews with Domain Experts

3.4.3.1 Introduction

Examples of empirical evaluations of conceptual models include surveys [25, 62], laboratory experiment [14], case studies [77, 92] and action research [154]. Each evaluation method can be applied in a particular case (e.g., laboratory experiment [14] requires putting the model in practice and observe performance metrics). At this stage of my research, conducting a survey [25, 62] is the most suitable method (see Table 3.1).

In this part of the evaluation, I chose to carry out semi-structured interviews with five domain experts that have strong background and are currently active in the area of service computing and information system design and development.

The objective of this evaluation is to assess the intuitive appeal of the proposed meta-model for describing the business capabilities of process activities. In this evaluation a design science methodology was followed [241] together with formal guidelines for conducting and reporting case study research [187].

The interviews were done after explanation of this thesis objective and details about service modelling approaches. The main targeted outcome of these interviews were to

identify if these experts can confirm that the proposed model is good enough to model business capabilities and if it can be adopted in their working environment.

3.4.3.2 Participants

For this semi-structured survey, five participants were recruited from different levels of expertise in the area of service computing and information systems design and development. The age group of these participant is 30-50 years old and their professional background includes a minimum of 5 years experience and are currently active in their field. Their profiles include:

- two project managers (P1 and P2): leading teams of developers of information systems for the management of seaports in different countries.
- two service providers and consumers (P3 and P4): working as consultants in the area of telecommunication. One of them is also a manager of his own start-up offering automated post services.
- and one IT engineer (P5): working in the same start-up as the main developer of the provided service.

3.4.3.3 Approach

The approach used for this evluation follows the case study research process proposed by Runeson and Höst [187]:

- *Case study design*: the objective of the evaluation is to assess the intuitive appeal of the proposed model. Interviews run individually using online conferencing tool. Each interview took about 1 hour for each participant.
- *Preparation for data collection*: The discussions were semi-structured to give the participants the freedom to give additional comments and get as much feedback as possible from them. Collecting evidence: The structure of the interviews was as follows [5]:
    1. 5 minutes discussion about the profile of the participant and his knowledge about services and business processes modelling languages.

[5]Please note that the durations used here are approximative. Some of the interviews run for few minutes more or less for each section.

2. 15 minutes presentation of the business capability meta-model introduced in this work with open discussion on each component and its use.
3. 15 minutes for creating simple business capability ontology in RDF
4. 10 minutes demo and interaction with the tool support
5. 15 minutes discussion about the proposed approach and modelling language

- *Analysis of collected data*: A post interview analysis of the collected feedback is reported in Section 3.4.3.4.
- *Reporting*: A discussion of the resulting feedback is summarised in Section 3.4.3.5 and shared among the participants.

3.4.3.4 Results

The following outcomes were identified:

General comments on the experience of the experts in capability modeling

- All the experts are familiar with all the standardized service description languages that have been discusses in Chapter 2: WSMO [242], OWL-S [232], SA-WSDL [198, 226] and SA-REST [88].
- Each of these experts has extensively worked with at least one of these languages.
- It is highly agreed by P1, P2, P3 and P4 that all these languages focus primarily on the technical aspect without considering the business aspect.
- None of these experts is still using any of these languages because they are complicated to understand and get familiar with.
- Developers need to read a lot about the use of ontologies, rules, reasoners, etc. An average user cannot easily adopt such technologies.
- They all prefer to have their own custom made modelling of their assets.
- These experts are developing REStful APIs that exchange data using JSON format and for them using JSON for their services description was a natural choice.
- The use of JSON gives them the flexibility they need to identify their own properties and values.

Feedback on the business capability modelling approach

- These experts confirmed that modelling of business capabilities the way I am proposing in this thesis seems to be quit close to their vision and simple to implement.

  "*Frame-based modelling is same as key value pairs. This is exactly how we model objects in JSON.*" (P5)

- Designing sample capabilities by P2 and P5 was easy and intuitive after a short tutorial.

- None of the experts is in favour of the idea of exclusively modelling business capability properties.

- They are interested in including more implementation/technical properties.

Feedback on the implementation of the business capability meta-model

- The experts (P2, P3 and P4) were not necessarily in favour of using RDF as an underlying implementation language of such model. After discussing JSON-LD, they were convinced that it is a better alternative.

  "*I think RDF is not the best implementation language.*" (P2)

- The tool support was very useful to show how the model can be used to annotate business process models.

3.4.3.5 Discussion

Surveys and interviews are often characterized with a high degree of representativeness compared to experiments [206]. However, they exhibit a low level of control over extraneous factors such as the influence of the background of the participants to their answers. To limit this factor, in this evaluation, the choice of these particular participants was made for two main reasons. First of all, both managers helped previously with the validation of the three motivational scenarios from Chapter 1. Consequently, they are familiar with this research and particularly with the motivation of modelling business capabilities. Second, the diversity in these profiles guarantees different points of view:

- Managers have a global view over the entire lifecycle of business processes.

- Consultants have multiple interactions with end-users as they, regularly, give trainings and information sessions to end-users.
- Engineers have a strong technical background in the development of services and business process management tools.

The important outcomes from these interviews can be summarised as follows:

> All the expert agree that current modelling languages do not give much importance to business capabilities. The proposed model in this work comes as an addition rather than a substitution to current models for describing a different view of enterprise information systems.
>
> Frame-based modelling is good modelling paradigm towards ease of use and intuitiveness of the models. However, the use of RDF as underlying realisation language is not the best option. The tool support was very helpful to hide its complexity.
>
> Some of the experts suggest to include technical aspects in the current model to serve as bridge between both business and IT perspectives. This recommendation is aligned with my vision in this thesis. The business capability is one of the aspects of a service description that can be further extended.

3.5 Summary

In this chapter, I defined a meta-model for describing Business Capabilities using frame-based paradigm by featuring an action category and related properties. This Model permits the description of business capabilities independently from their actual implementations by highlighting the action being performed with a set of related properties.

While process models explicitly capture the involved activities and workflows together with organisation-specific resources, the proposed model focuses at providing an abstract representation of what these processes achieve or the outcome that customers or collaborators need. Within the same organisation, there may be several workflows for specific outcomes (e.g., by rearranging activities and resources), but on a broader scale the organisation would not expose the different workflows, but would only show the business capabilities if offers. The model proposed can serve this need, however, further steps are required: identification of the business capability of an entire process, indexing a repository of business capabilities and cutomization of process models.

In the following chapters, I will use this business capability modelling approach for aggregating business capabilities to determine the business capability of an entire business process model in Chapter 4, indexing and discovering business capabilities in Chapter 5, and use the concept of configurable business capability for driving the customization of configurable business process models in Chapter 6.

Chapter 4

Aggregation of Business Capabilities: Determining the Business Capability of a Process Model

"There are no facts, only interpretations."

Friedrich Nietzsche

4.1 Introduction

Process Aware information Systems [64] allow to manage and execute business processes involving several components on the basis of process models. These models constitute a central element that is being shared among various stakeholders.

A business process model can detail various elements: activities, data objects, control flow, etc. Not all the stakeholders are interested in all these details; e.g., the strategic management team is more interested in WHAT is being performed, however, the technical team is interested in HOW tasks are performed. Consequently, there is a lot of effort put towards finding the right details that need to be presented to the involved stakeholders. For example, when it comes to privacy concerns when presenting processes to business partners, Eshuis et al. [72] suggest hiding unwanted process elements while preserving the entire process consistency, whereas Reichert et al. [179] present an approach that allows for a customized representation of process models with respect to the

user preferences, while Polyvyanyy et al. [172] propose to simply reduce the complexity of process models.

Whether the aim is hiding details for privacy reasons or providing different process views, the biggest obstacle remains the complexity of process models. As processes tend to be large in size and complex in structure, abstraction techniques help transforming them from more to less detailed ones.

Business Process Models Abstraction (BPMA for short) is a promising technique that allows for a seamless navigation from a detailed process model to an abstract one. Two strategies can be used in BPMA. The first one consists of leaving out unwanted components of the model [72]. Users can for example visualize only the elements they are interested in. With such solution, only essential elements are kept, however, the entire overview of the process model is partially presented. The second one consists of aggregating several components into a single abstract one [32, 172, 179, 208, 209, 228]. For example, the activities "book flight" and "book hotel" can be aggregated into an abstract activity "arrange trip". Consequently, an entire process model can be represented at several levels of abstraction. It can even be abstracted into a single activity. In such settings, current solutions limit the result of aggregation to a single label which gives a shallow representation of the business capability of the entire process.

This shallow representation of the aggregated business capability is due to the fact that current process models do not properly describe business capabilities as it has been discussed in Chapter 2. This can be resolved if each task of initial business process models is annotated with its business capability using a rich description such as the one proposed in Chapter 3. Proper aggregation method can be defined in order to determine the aggregated business capability of the entire process model.

The contribution of this chapter is an **aggregation algorithm** that computes the business capability of an entire process model. The proposed algorithm requires the input model to be **business capability annotated**, i.e., a control flow where each activity is annotated by its business capability considering the conceptual model defined in Chapter 3. Section 4.3 gives a formal description of such a model. The aggregation algorithm will be presented in Section 4.4. Section 4.5 reports on the developed tool support and interviews carried out with domain experts to discuss the applicability of the proposed approach in a corporate environment before concluding in Section 4.6.

4.2 Motivating Example

Figure 4.1 depicts an example containing a process model for the examination procedure of an importation process[1]. Using the approach proposed by Smirnov et al. [208], this example would be abstracted into one activity that will be presented by a single label (e.g., "Examination of cargo"). It is obvious that a single label does not carry enough information to adequately describe the semantics of the functionality of this entire process model.

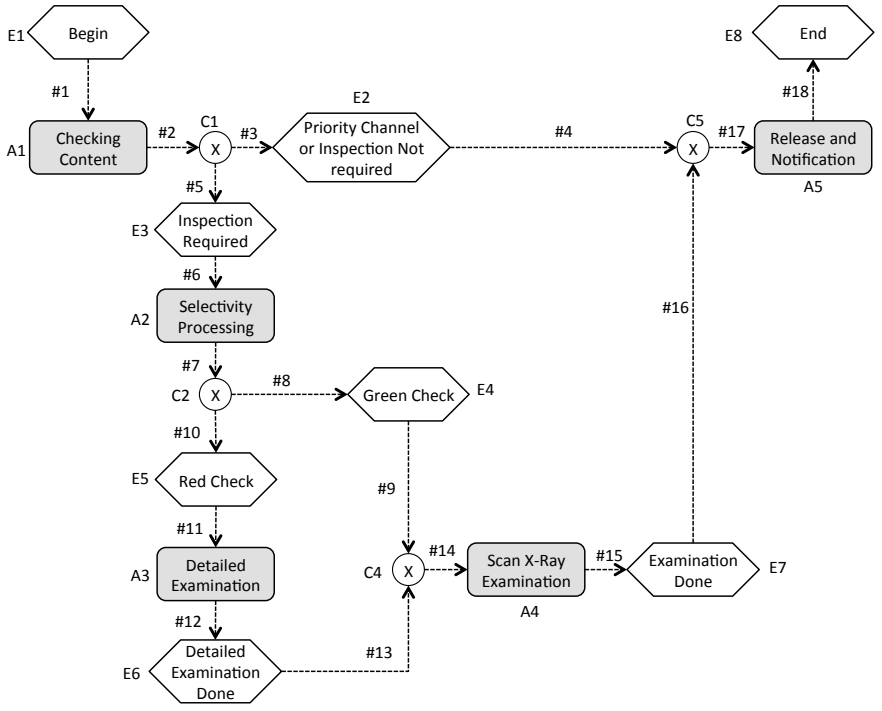


FIGURE 4.1: Examination of Cargo Procedure at Davao City Seaport in Philippines

This chapter proposes another technique that allows moving from an entire process model to its functional description by **aggregating all the business capabilities** of the process elements into a single **aggregated business capability**. A business capability should feature functional domain properties and not be limited to a single label. The business capability of the process model depicted in Figure 4.1 should report that after checking the content of the cargo (Activity A1) a decision on physical inspection is made. If the cargo goes through a priority channel or if a physical inspection is not required (Event E2), then it is directly released (Activity A5) without inspection, otherwise a physical inspection (Event E3) is required. In this case, a red (Event E5) or green (Event E4) check is performed depending on the results of the selectivity processing (Activity A2). A red check goes through a detailed examination of goods (Activity A3)

[1]This model is available at http://kjri-davao.com/?page=news&siteLanguage=English&address%20link=127&cat=Economics as accessed on 06-06-2014.

and an X-ray scan (Activity A4), however, a green check needs only an X-ray scan (Task T4).

4.3 Business Capability-annotated Business Process Model

In the area of business process modelling various efforts, either from industry or academia, have proposed modelling languages such as Event-driven Process Chains (EPC) [147, 148], Business Process Modelling Notation (BPMN) [27], and Unified Modelling Language Activity Diagram (UML AD) [186, 188]. This thesis aims to abstract from any of these notations to model business processes without any ties to existing business process modelling languages. The advantage of this abstraction is to make the thesis contributions easily applicable to other modelling languages. In this thesis, a business process model is presented as a directed graph and formally described in Definition 6.

Definition 6 (Business Capability Annotated Business Process Graph). A *Business Capability Annotated Business Process Graph* is a directed graph $G =< N, C, A, T, Cap, Cond >$, where N is a set of work nodes including $InitialNode$, $FinalNode$ and $IntermediateNode$ that are both Event and Activity Nodes; C is a set of graph connectors: i.e., ANDsplit, ANDjoin, ORsplit, ORjoin, XORsplit, and XORjoin and A is a set of directed arcs for interconnecting all the graph nodes. T is a type function, it associates with each node its respective type (i.e., a string to indicate: activity, event, XorSplit, etc.). Cap is an annotation function that associates with each Activity Node n a tuple Cap(n) = (ActionCategory (n), Properties(n)). $Cond$ is an annotation function that associated with each event node n a condition c (i.e., Cond(n)=c).

For a Business Capability Annotated Business Process Graph G, its set of work nodes is denoted N_G. Each work node n in N_G has a type depending on the modelling language being considered. For example, in BPMN and EPC there are two types of nodes: Events and Functions. In N_G there are two particular nodes: $InitialNode$ and $FinalNode$ for marking the beginning and the end of the business process. These nodes have dedicated graphical representations in BPMN specification [163] while in EPC these are regular events without incoming arcs for the $InitialNode$ and without outgoing arcs for the $FinalNode$.

Connectors, also known as routing nodes, of a Business Process Graph are denoted C_G. Each c in C_G can be either a split of join connector. Split connectors have a single input arc and multiple output arcs while join connectors have multiple input arcs and a single output arc. A split connector indicates that (i) the flow of activities continues into multiple parallel branches (i.e., in case of an ANDSplit), (ii) a choice has to be made

towards one possible active branch (i.e, in case of a XORSplit) or (iii) multiple branches can be activated (i.e, in case of an ORplit) after this node. A join connector indicates that the process has to wait until (i) all the branches (i.e., in case of an ANDJoin), (ii) exactly one branch (i.e, in case of a XORJoin) or (iii) multiple branches (i.e, in case of an ORJoin) are activated before this node.

Each of the nodes of an Annotated Business Process Graph are interconnected with directed arcs denoted A_G. These arcs define either causal or temporal relations between these nodes. For a node $n \in N \cup C$, $\bullet n$ and $n\bullet$ denote respectively the set of input and output arcs of n. Syntactic restrictions on possible arcs between nodes can be imposed based on the modelling language. For example, in EPC, arcs cannot exist between two functions or events.

In order to get the business capability of an annotated business process graph, the function Cap_G is used, it associates for each Activity Node (e.g., function node in EPC) its business capability with respect to the conceptual model defined in Chapter 3.

I capture the proposed business process graph of definition 6 as an RDF vocabulary called BANG [2] shown in Figure 4.2. In this vocabulary, I create *bang:BusinessProcessGraph* for representing G, its set of work nodes N_G and routing nodes C_G are respectively represented as the RDF classes *bang:WorkNode* and *bang:Connector*. Work nodes are further refined into *bang:EventNode* and *bang:ActivityNode*. A conditional *bang:EventNode* is defined with *cmm:Condition* via the property *cmm:hascondition*. Connectors are presented as classes depending on their types: *bang:ANDSplit*, *bang:ANDJoin*, *bang:ORSplit*, *bang:ORJoin*, *bang:XORSplit* and *bang:XORJoin*. Arcs A_G are defined via the RDF class *bang:Arc*; for each arc, the source and destination are captured respectively with the RDF properties *bang:hasIn* and *bang:hasOut*. Finally, the function Cap_G is presented via the RDF property *bang:hasCapability*.

The model defined here can be used for modelling business processes while capturing domain features in terms of business capabilities for its different activities. This model does not capture any formal aspects or technical realizations. Furthermore, this model can be tailored to other existing modelling languages. In the rest of this chapter EPC is used as underlying modelling language. An EPC (see Figure 4.1) is a directed graph using three types of nodes interconnected with directed arcs. These nodes are:

- **Functions**: represented as rounded rectangle, and correspond to the tasks that need to be performed when the process is executed.

[2] Available at: http://vocab.deri.ie/bang

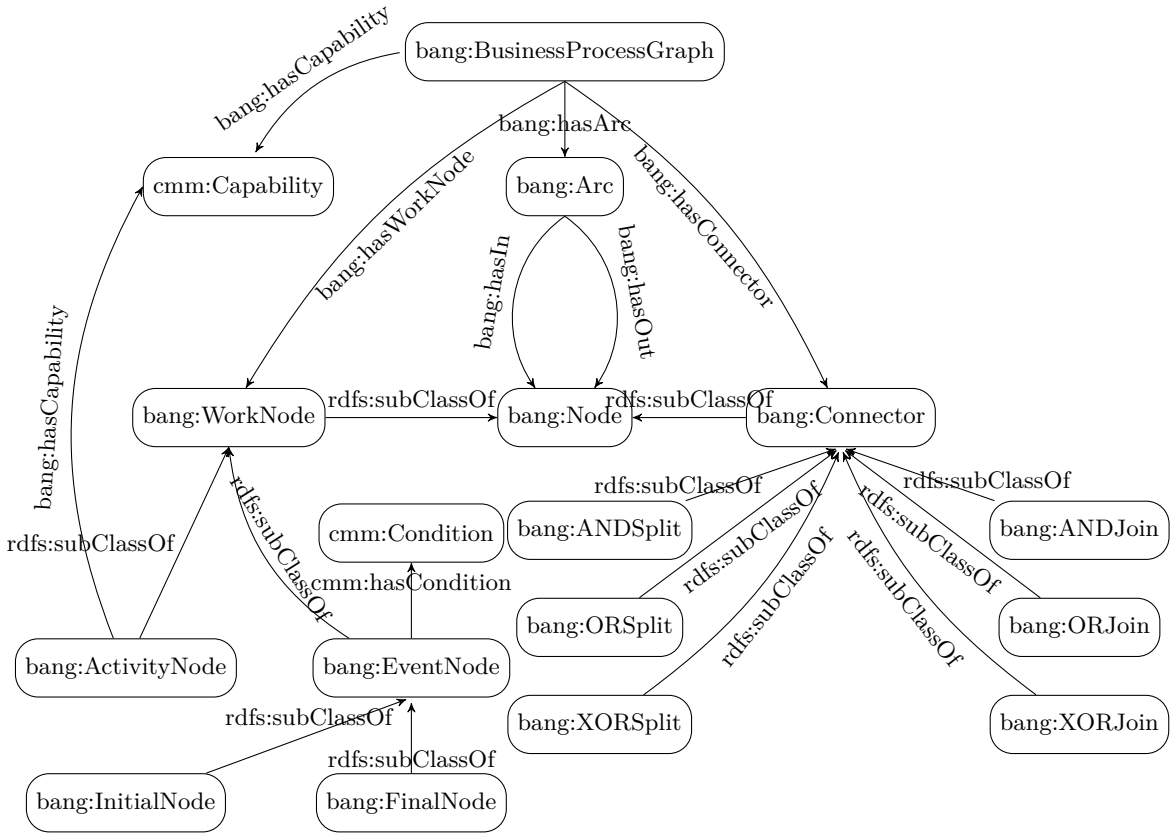


FIGURE 4.2: RDF Vocabulary for Annotated Business Process Graph: BANG

- **Events**: represented as hexagons, they describe under what circumstances a function works or the state in a function results. Event nodes are used for presenting preconditions that need to be satisfied to perform a particular function; this means that events are triggers for functions. Each function requires one or more succeeding and preceding events.
- EPC includes three kinds of **connectors**: OR-connector (V), XOR-connector (X) and AND-connector. Connectors are used to introduce parallel and alternative branches (in case of split connectors); or merging multiple branches (in case of join connectors).

Table 4.1 shows a direct mapping that I use between EPC items and the proposed BANG vocabulary.

Using this mapping, I create the *RDF* representation expressed in N3 of the example shown in Figure 4.1. The result is shown on Listing 4.1. This model illustrates a business process model for the examination procedure of an importation process. The model is referred as *model:CHKbpm* in line 6 of Listing 4.1. The first step of this business process consists of checking the content of the cargo and decide if a physical

TABLE 4.1: Mapping Items from Event-driven Process Chains (EPC) to BANG Concepts

EPC items	**BANG vocabulary concepts**
Function	bang:ActivityNode
Event	bang:EventNode
And connector (split)	bang:ANDSplit
And connector (join)	bang:ANDJoin
Or connector (split)	bang:ORSplit
Or connector (join)	bang:ORJoin
Xor connector (split)	bang:XORSplit
Xor connector (join)	bang:XORJoin

inspection is required. If the cargo goes through a priority channel or if a physical inspection is not required, then it is directly released without inspection. Otherwise, a red or green check is selected for examining the cargo. A red check goes through a detailed examination of goods and an X-Ray scan, however, a green check needs only an X-ray scan. Each function of the EPC model of Figure 4.1 is annotated with its business capability described in Table 4.2. The first line of this table describes the business capability of the first function that consists of Checking Content (i.e., cmm:achieves imp:checkingContent) of the cargo (i.e., impc:cargo xsd:String) and take a decision regarding the inspection (i.e., impc:examDecision). The last column of this table shows the RDF N3 presentation of these business capabilities.

Please note that the *BANG* vocabulary does not cover the description of any graphical characteristics. In fact, graphical aspects are dependant on the modelling languages as well as the used tools, dimensions, etc. My primary focus is on representing business process model components (activities, their business capabilities, and events) and the routing information between them (arcs and connectors). Consequently, BANG on its own does not impose any syntactic restrictions on business process graphs. In other words, there is no guarantee that a given graph described in BANG is well structured and respects other business process modelling notations restrictions.

4.4 Aggregation of Business Capabilities

4.4.1 Determining the Action Category of an Aggregated Business Capability

The action category is a mandatory property in the business capability description. Its value is taken from an Actions Ontology that is also used for determining the action category of aggregated business capabilities. An aggregated business capability has as

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix cmm:   <http://vocab.deri.ie/cmm#>.
@prefix bang:   <http://vocab.deri.ie/bang#>.
@prefix model:   <http://.../models/BusinessProcessModels/CHKContent#>.

model:CHKbpm  a bang:BusinessProcessGraph ;
        bang:hasWorkNode model:checkingContent, model:selectivityProcessing,
                         model:detailedExamination, model:scanXRay,
                         model:releaseAndNotification, model:begin,
                         model:PIRequired, model:redCheck, model:greenCheck,
                         model:DetailedExaminationDone, model, ExaminationDone,
                         model:inspectionNotRequired, model:end;
        bang:hasArc model:arc1, model:arc2, model:arc3,
                         model:arc4, model:arc5, model:arc6,
                         model:arc7, model:arc8, model:arc9,
                         model:arc10, model:arc11, model:arc12,
                         model:arc13, model:arc14, model:arc15,
                         model:arc16, model:arc17, model:arc18.

model:selectivityProcessing a bang:ActivityNode;
        rdfs:label ''Selectivity Processing'';
        bang:hasCapability   model:Phil_Cap_SelectivityProcessing.

model:Phil_Cap_SelectivityProcessing a cmm:Capability;
        cmm:achieves imp:SelectivityProcessing;
        impc:hasCargo xsd:String;
        impc:hasTypeOfCheck model:PhilTypeOfCheckValue.

model:PhilTypeOfCheckValue a impc:TypeOfCheck, cmm:EnumerationValue;
        cmm:hasElement impc:RedCheck;
        cmm:hasElement impc:GreenCheck.

model:begin a bang:InitialNode;
        rdfs:label ''Begin''.

model:arc1 a bang:Arc;  bang:hasIn model:begin;
                bang:hasOut model:checkingContent.
```

LISTING 4.1: RDF N3 representation of of the Cargo Examination Process depicted in Figure 4.1 using BANG vocabulary (Figure 4.2)

action category corresponding to the Lowest Common Ancestor (LCA) of the action verbs of its components.

In this work, I created an Actions Ontology for Import procedures [3] (available in Appendix C) that is illustrated in Figure 4.3 [4] (available in Appendix D). Using this ontology for determining the action category of the aggregated business capability of the entire process model depicted in Figure 4.1 consists of looking for the LCA of all the action verbs of tasks of that process model : *LCA(Checking Content, Selectivity Processing, Detailed Examination, Scan X-Ray) = Examination of Cargo.*

Ideally, all the action categories used in the model are taken from the same actions ontology like in this running example. For various reasons, modelers can use actions taken from different action ontologies. Instead of searching for the LCA in a single actions ontology, one needs to take into account all the possible ontologies used in

[3] http://vocab.deri.ie/imp

[4] The actions ontology is created with the assistance of domains experts that have validated it after several interviews.

TABLE 4.2: Business Capability Annotations of the Functions of the Cargo Examination Process Depicted in Figure 4.1

Activity	**Description**	**Business Capability in RDF N3**
Checking Content	This activity consists of **checking the content** of the **cargo** in order to take a **decision** about the necessity of a physical check.	model:Phil_Cap_CHKContent a cmm:Capability; cmm:achieves imp:CHKContent; impc:hasCargo xsd:String; impc:hasExamDecision model:PhilExamDecisionValue. model:PhilExamDecisionValue a cmm:EnumerationValue; cmm:hasElement imp:PriorityChannel; cmm:hasElement imp:PhysicalInspectionNotRequired; cmm:hasElement imp:PhysicalInspectionRequired;
Selectivity Processing	This activity consists of **selecting** the **type of check** that needs to be done.	model:Phil_Cap_SelectivityProcessing a cmm:Capability; cmm:achieves imp:SelectivityProcessing; impc:hasCargo xsd:String; impc:hasTypeOfCheck model:PhilTypeOfCheckValue. model:PhilTypeOfCheckValue a impc:TypeOfCheck, cmm:EnumerationValue; cmm:hasElement impc:RedCheck; cmm:hasElement impc:GreenCheck.
Detailed Examination	This activity consists of performing a **detailed examination** of the **cargo**.	model:Phil_Cap_DetailedExamination a cmm:Capability; cmm:achieves imp:DetailedExamination; impc:hasCargo xsd:String; impc:hasExamType imp:detailed.
Scan X-Ray Examination	This activity consists of performing an **X-Ray scan** of the **cargo**.	model:Phil_Cap_ScanXRayExamination a cmm:Capability; cmm:achieves imp:ScanXRay; impc:hasCargo xsd:String; impc:hasExamType imp:X-Ray.
Release and Notification	This activity consists of releasing the **Cargo** and sending a **notification message** to the concerned person.	model:Phil_Cap_ReleaseandNotification a cmm:Capability; cmm:achieves imp:ReleaseAndNotification; impc:hasCargo xsd:String; impc:hasMSG xsd:String.

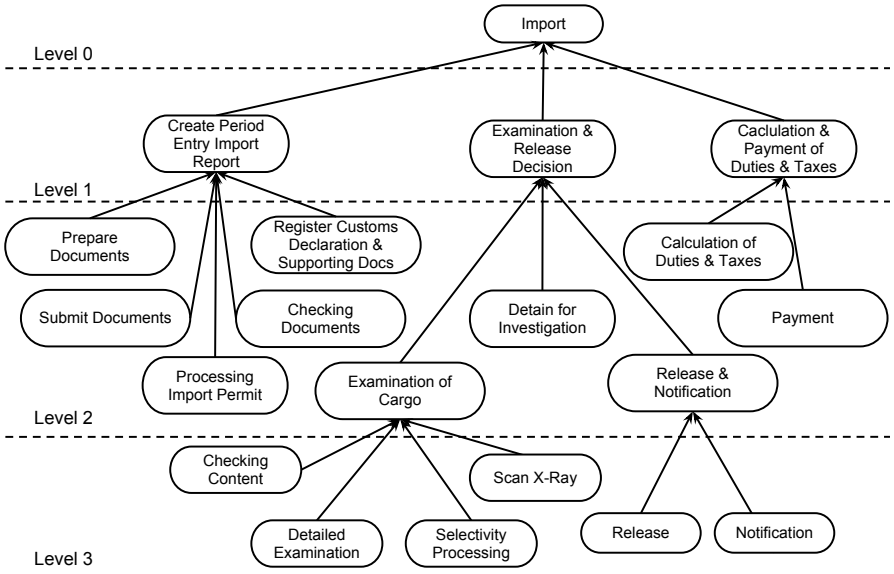


FIGURE 4.3: Actions Ontology of the Import Domain

assigning action categories to the capabilities of a process model. In such a case a more elaborated method as presented by Smirnov et al. [208] is needed.

4.4.2 Determining the Set of Properties of an Aggregated Business Capability

In order to propose a correct propagation algorithm, I assume that the input process model does not have any loops and is well structure. In a well-structured process model every split connector has a corresponding join connector, whereas both connectors bound a process model fragment with one entry node and one exit node [106]. I propose to use the formal semantics of the business capability annotated business process graph as a token game, similar to Petri Nets [170].

A Petri Net is a tuple (P, T, F), where P is a finite set of places (representing events in EPC), T is a finite set of transitions ($P \cap T = \emptyset$) (representing functions in EPC) and F $\subseteq$ (P x T) $\cup$ (T x P) is a set of arcs (flow relations). Petri nets can be used to represent dynamics of business process models by using token propagation to verify if models are regular, sound and well-structured [219].

A token is a theoretical concept that is used as an aid to define the behaviour of a process by firing its nodes. The *Initial* place generates a token that traverses the sequence transitions and passes through all the places until reaching the *Final* place [102]. In this case, an EPC is transformed into a petri net using transformation rules depending on the type of EPC nodes. While function and event nodes are transformed into transitions and places respectively, mapping connectors is more complex. This depends on the type of connector and its linked nodes [219]. Figure 4.4 and 4.5 show, respectively, the mapping of split and join connectors to Petri Nets proposed by van der Aalst [219].

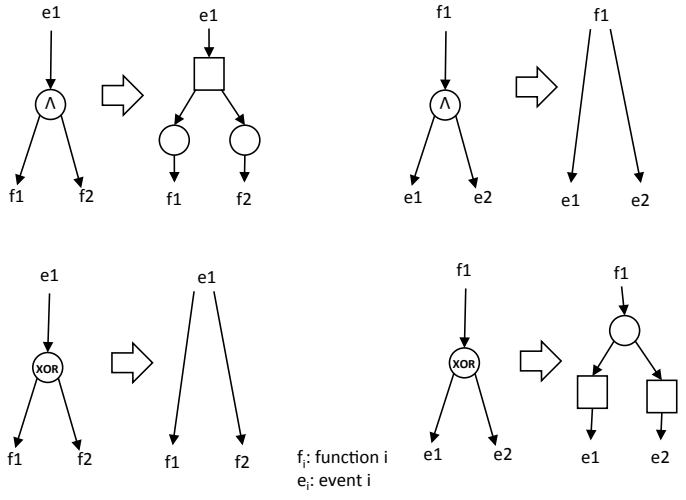


FIGURE 4.4: Mapping EPC Split Nodes to Petri Nets [219]

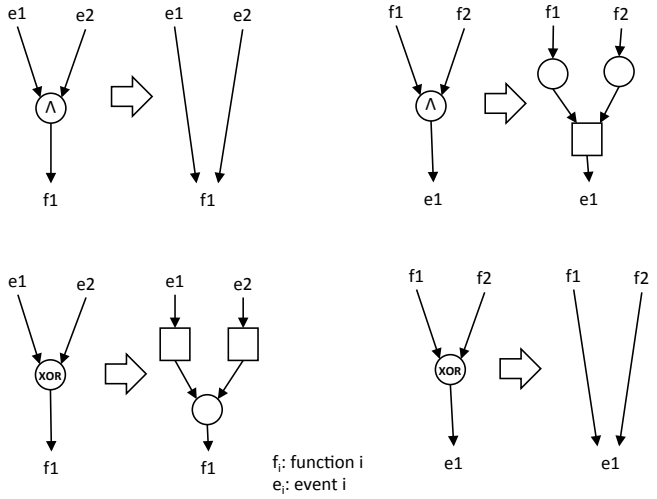


FIGURE 4.5: Mapping EPC Join Nodes to Petri Nets [219]

The idea of the business capability aggregation algorithm is similar, it starts from the *InitialNode* then fires all the nodes one by one and propagates the subsequent properties until it reaches the *FinalNode*. Each node introduces some changes on the set of propagated properties. The propagated properties at a particular node are marked on its outgoing arcs. Knowing that event nodes can be conditional nodes, they also affect the propagated properties, I also mark the propagation of conditions as they can have effects on subsequent properties. The annotation of arcs with propagated properties and conditions is called propagation result (see Definition 7).

Definition 7 (Propagation Results). A *Propagation Result* is a couple $PR =< PP, PC >$, where PP and PC are two sets of Propagated Properties and Conditions respectively. $\forall a \in A$ $PP(a)$ and $PC(a)$ refer respectively to the propagated properties and conditions at an arc a.

The propagation of properties and conditions is then guided by the traversal of tokens in the petri nets representing the business process model. However, classical petri nets allow only the modelling of states, events, synchronisations, etc. and are not able to model data objects such as the properties of capabilities. To solve this issue colored or typed petri nets [223] have been introduced as an extension to classical petri nets where tokens represent objects (e.g., data item) in the system. Tokens represent 'colors' or set of properties. At each transition, a token is produced with respect to the consumed tokens. More concretely, a transition represents a relation between input and output tokens. The introduced propagation algorithm in this chapter defines the relations of these transitions. The idea of this algorithm has been used by Vulcu et al. [228] for

propagating IOPEs of process models to determine their IT capabilities. Similarly, Weber et al. [235] used precondition and effect propagation to verify the soundness of business process models.

Initialisation step

The first step of the business capability aggregation algorithm starts by initializing the annotations of edges with the initial propagation results with respect to Definition 8.

Definition 8 (Initialisation of the propagation). Let $G =< N, C, A, T, Cap, Cond >$ be a business capability annotated business process graph (see Definition 6), $\mathcal{P} = \bigcup_{\forall n \in N/T(n)=ActivityNode} Properties(n)$ and $\mathcal{CO} = \bigcup_{\forall n \in N/T(n)=EventNode} Cond(n)$. The initialisation function is defined as follows: $Prop_0 : A \rightarrow (\mathcal{P} \bigcup \{\perp\}, \mathcal{CO} \bigcup \{\perp\})$ such that $\forall a \in A$:

- $Prop_0(a) = (\{\}, \{\})$ if $a = InitialNode\bullet$
- $Prop_0(a) = (\{\perp\}, \{\perp\})$ otherwise (the symbol $\{\perp\}$ means that the value is unknown)

Note that Definition 8 uses $Properties(n)$ to refer to the set of properties of the business capability of the Activity Node n.

The initialisation of the propagation assigns the value ({},{}) to the outgoing arc of the $InitialNode$ and the value ({⊥},{⊥}) to the other arcs. These notations are interpreted as follows: if an arc $a \in A$ is annotated by ({},{}), then the propagation result from the $InitialNode$ until this arc is empty. If an arc $a \in A$ is annotated by ({⊥},{⊥}), then the propagation result has not yet been defined for that arc.

Going back to the running example depicted in Figure 4.1, the initialisation step makes $Prop_0(\#1)$ = ({},{}) and all the other arcs will be initialized to ({⊥},{⊥}).

Correctness of the initialisation step: The initialisation step simply places the token at the $InitialNode$, obviously no propagations have started and consequently it is correct.

In definition 7, the second term of the propagation results (i.e., PC) holds the conditions that are introduced by event nodes at a conditional branching. Assuming that our input model is well structured, each conditional branching starts by a split connector followed by conditional events and ends by a corresponding join connector. In this situation, each node on the path from the split connector to the corresponding join connector is fired only if the condition of the split event is satisfied. In addition, multiple embedded cases of conditional branching can exist in a model. For this reason and for making the

propagation results easier to compute, PC contains the conjunction of all conditions that need to be satisfied on each path.

Figure 4.6 [5] highlights two cases of conditional branching in the running example: (C1-C5 and C2-C4). C2-C4 is a conditional branching that is embedded in C1-C5. The propagation of conditions results in:

- $Prop_C(\#4) = (\{\bot\}, \{\text{“Priority Channel or Inspection Not required”}\})$
- $Prop_C(\#11) = (\{\bot\}, \{\text{“Inspection Required” AND “Red Check”}\})$

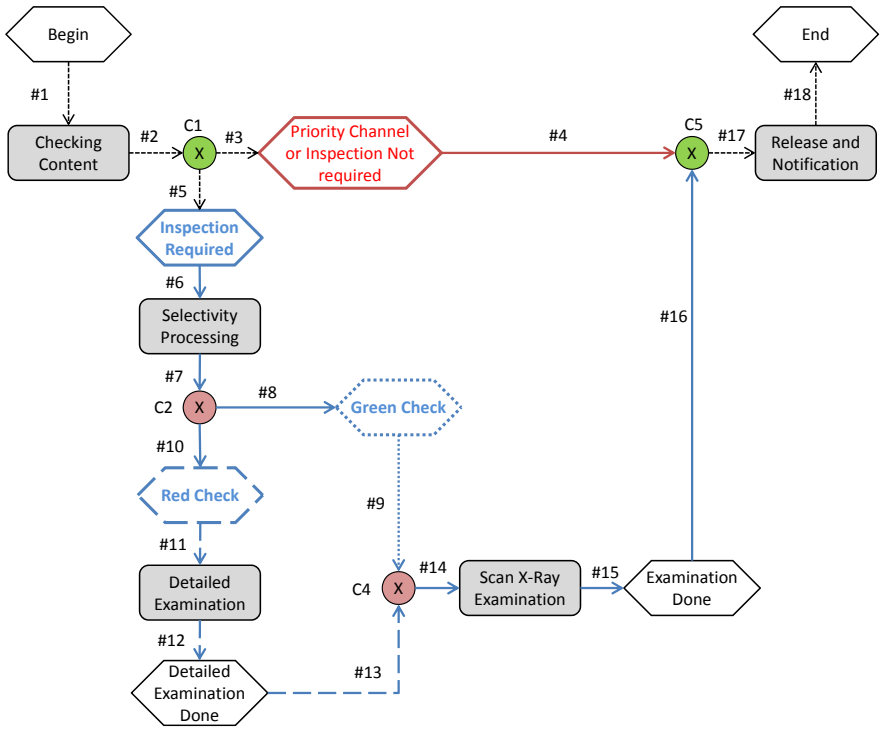


FIGURE 4.6: Process Model of the Cargo Examination Process of Figure 4.1 with Corresponding split-join Connectors

Correctness of the conditions propagation step: The propagation of conditions does not affect the set of properties of the aggregated business capability. It is only used for annotation purposes for facilitating the computation of conditional values of the propagated attributes and consequently it does not introduce any errors to the propagation result. Furthermore, only well structured models are considered in this work, this makes the propagation of conditions limited by the corresponding split-join nodes.

[5] Each corresponding pair of connectors share the same color (Green or Red). Each path from one connector to its corresponding one is highlighted either by a color (red or blue) or pattern (plain or dotted lines). These colors are simply used for visualisation purposes and dot not have any semantics in the modelling language used.

After the initialisation steps (initialisation and propagation of conditions), the algorithm starts the computation of the first term of the propagation results (i.e., PP) at each arc. This is done via the propagation of properties after firing one node at a time. Each node n might introduce some changes on the set of propagated properties from its incoming arc(s) $\bullet n$ and propagates them on its outgoing arc(s) $n\bullet$. This makes the propagation of properties different from a simple $\cup$ operation between the properties. For this reason, the operator $\uplus$ is introduced to represents the aggregation function applied when propagating the set of properties. This function depends on the control flow pattern being considered, the property type and its value. As described in Definition 9, if the fired node n is an Activity Node then the algorithm computes the $\uplus$ of the propagated properties from $\bullet n$ with the properties of the fired node n. If the fired node n is a join node then the algorithm computes the $\uplus$ of all the properties from all the incoming arcs $\bullet n$ of the fired node. If the current node n is an event or a split then there are no changes on the set of properties from $\bullet n$ and they are propagated as they are on the outgoing arcs $n\bullet$. The $\uplus$ operator will be discussed later in this section.

Definition 9 (Propagation Functions). Let the two functions $Prop_k, Prop_{k+1} : C \rightarrow \mathcal{A} \bigcup \{\perp\}$. $\forall n \in N, Prop_{k+1}$ is the propagation of $Prop_k$ at the node n iff: $\forall\ c_{in} \in \bullet n, Prop_k(c_{in}) \neq \{\perp\}$. $\forall\ c_{out} \in n\bullet$

1. if n is an Activity Node: $Prop_{k+1}(c_{out}) = Prop_k(c_{in}) \uplus Properties(n)$
2. if n is an ANDjoin, ORjoin or XORjoin: $Prop_{k+1}(c_{out}) = \uplus_{\forall c_{in} \in \bullet n} Prop_k(c_{in})$
3. n is an Event Node, ANDsplit, ORsplit or XORsplit: $Prop_{k+1}(c_{out}) = Prop_k(c_{in})$

There exist in the literature several attempts to determine the aggregation function for computing quality of service parameters (e.g., execution time, latency, cost, etc.) of composed web services using control flow patterns [103, 105]. The major aggregation functions used in such contributions are summation, average, maximum, etc. where all the values of an attribute are considered in the computed value. However, this cannot be the case in this work. Indeed, if a propagated property has more than one value, the propagation function should consider either all the values or only one of the alternatives that is described via the $\uplus$ operator.

To select the right values for the aggregation I defined a control mechanism based on categorization of the properties (i.e., each property is tagged by a category). Each category helps determine the required aggregation function. If one needs to determine the aggregation function applied on a property, they simply need to indicate its category. In the following, I present the set of categories that I take into consideration:

- *Dominant*: the value of a property of this category cannot change. During the aggregation operation if only one of the alternative values is dominant, then its value is the only one to consider. If multiple alternatives are dominant, then the property value becomes an enumeration of all the dominant values.
- *Composed*: the value of a property of this category depends on a function. Its aggregation consists of updating this function.
- *Passive*: the value of a property of this category can be overridden by any other value if it has a superior category (i.e., Dominant or Composed).

It is important to note that there is a superiority order between these categories: *Dominant* > *Composed* > *Passive*. These categories help to determine the right aggregation function from this list:

- *Copy*: this function simply copies the property without applying any changes.
- *Override*: this function overrides the value of the property and considers only the superior category (*Dominant* > *Composed* > *Passive*).
- *Enumerate*: this function makes the property value an EnumerationValue and lists the possible values.
- *Conditional*: this function transforms the property value into a ConditionalValue.
- *Composition*: this function is applied on properties where a formula is needed to compute its value. The composition function consists of determining the new function of the aggregated property.

The following sections discuss the right functions to apply when firing each type of nodes of the model.

4.4.2.1 Fired Node is *Activity Node*

In order to determine the right aggregation function applied to compute the propagation of properties when firing an Activity Node n, refer to Table 4.3. Each column corresponds to the category of the property $p \in Properties(n)$ (i.e., dominant, passive and composed). Each line corresponds to the category of the same property $p \in Prop_k(\bullet n)$. Each cell defines the right aggregation function that is needed.

Table 4.3 defines the aggregation functions when firing an Activity Node where $PC(\bullet n) = \{\perp\}$. In other words, the condition propagation at the input arc of the node n is

TABLE 4.3: Required Aggregation Function when Firing an Activity Node

	$p \in Properties(n)$ of category **Dominant**	$p \in Properties(n)$ of category **Passive**	$p \in Properties(n)$ of category **Composed**	$p \notin Properties(n)$
$p \in Prop_k(\bullet n)$ of category **Dominant**	Enumeration(p) of category Dominant	Override(p) of category Dominant	Override(p) of category Dominant	Copy(p) of category Dominant
$p \in Prop_k(\bullet n)$ of category **Passive**	Override(p) of category Dominant	Enumeration(p) of category Passive	Override(p) of category Composed	Copy(p) of category Passive
$p \in Prop_k(\bullet n)$ of category **Composed**	Override(p) of category Dominant	Override(p) of category Composed	Composition (p) of category Composed	Copy(p) of category Composed
$p \notin Prop_k(\bullet n)$	Copy (p) of category Dominant	Copy(p) of category Passive	Copy (p) of category Composed	

empty. If this is not the case (i.e., $PC(\bullet n)! = \{\perp\}$), after applying the pre-mentioned aggregation function from Table 4.3, the result becomes a *Conditional Value*, where the condition is $PC(\bullet n)$. If a property $p \in Properties(n)$ and $p \notin Prop_k(\bullet n)$ then the propagated attribute will have a ConditionalValue on its copied value (i.e., line 4 of Table 4.3).

Correctness: The state of propagation of properties after firing an activity node is similar to the state of execution of a process after that activity. It can be modelled with a Petri Net with the position of tokens in places before and after the transition that represents the corresponding activity [219]. In formal semantics using Petri Nets, an Activity Node removes one token from one of its incoming places and generates one token on one of its outgoing places. Furthermore, if tokens are *typed or colored* (case of colored Petri Nets), the generated tokens at this transition can change their color with respect to a predefined function. In this work, the propagated properties are colored tokens. The firing of an activity corresponds to the transition step, the new colors of the tokens are the results of the propagation function selected from Table 4.3. Furthermore, I consider, in this work, only well formed models, Activity Nodes have only one incoming and one outgoing arc. Then the set of properties from the incoming arc will be taken by the Activity Node and after performing the required propagation operation it places another set of properties on its outgoing arc (see Figure 4.7). The new state reports properly on the right propagation of properties only if all possible cases of propagations are considered which is the case as per Table 4.3.

Continuing with the running example of Figure 4.1, the second iteration of the algorithm consists of firing the Activity Node *Checking Content*. As the arc #1 is annotated by ({},{}), that means $\forall p \in Properties(CheckingContent), p \notin Prop_1(\#1)$. According to Table 4.3, the required aggregation function is copy (p). The result of this iteration is reflected on the arc #2 that is annotated by the properties of the Activity Node *Checking Content*.

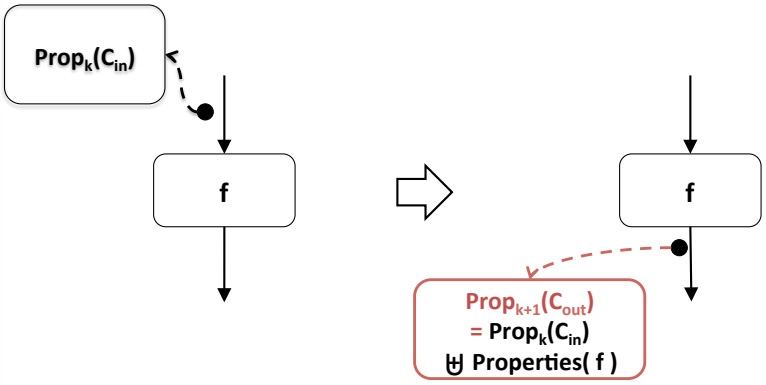


FIGURE 4.7: Properties Propagation when Firing an Activity Node

During the fourth iteration, the fired node is the Activity Node *Selectivity Processing*. This Activity Node introduces the attribute *TypeOfCheck*. According to Table 4.3, the aggregation function should be Copy(TypeOfCheck) as it is the case in the second iteration. However, the arc #6 is guarded by the condition *ExamDecision = imp:PhysicalInspectionRequired* which imposes the aggregation function *Conditional()* that makes the attribute TypeOfCheck a *ConditionalValue* where the condition is ExamDecision = imp:PhysicalInspectionRequired and its value would be an enumeration of RedCheck and GreenCheck.

4.4.2.2 Fired Node is an Event or *Split* Node (i.e., ANDsplit, ORsplit or XORsplit)

If the fired node n is an event, ANDsplit, an ORsplit or a XORsplit, the aggregation function is always a Copy(p). In other words, each property $p \in Prop_k(\bullet n)$ is copied to all its outgoing arcs. More formally: $\forall c \in n\bullet, Prop_k(c) = Prop_k(\bullet n)$.

Correctness (firing an event): Events are passive nodes, they simply pass the token as it is to the outgoing arc (see Figure 4.8).

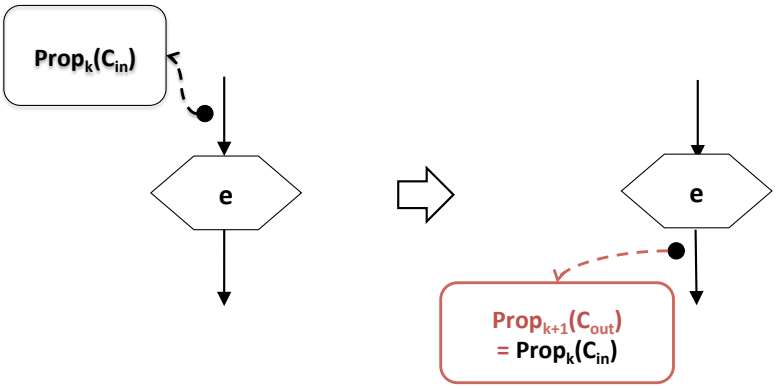


FIGURE 4.8: Properties Propagation when Firing an Event node

Correctness (firing a split node): According to the informal semantics of the split nodes, split nodes remove one token from their incoming arc and place one token on one of their outgoing arcs. To formally represent this in Petri Nets [219], multiple cases needs to be considered depending on the context (event to function, or function to event) and the routing operation of the node (or, xor, and) (see Figure 4.4). As in this work, the propagation of properties needs to reach all the nodes of the graph model (i.e., considering all possible execution cases), then the set of properties has to be propagated on all the outgoing arcs of the split node which is similar to the semantics of ANDSplit node in Petri Nets. Furthermore, split nodes are only connectors and do not introduce any changes on the propagated attributes and thus validates the use of the copy function. In this case, transitions introduced during the transformation of connectors to Petri Nets (see Figure 4.9 and 4.10) are simply passing the token without transformations.

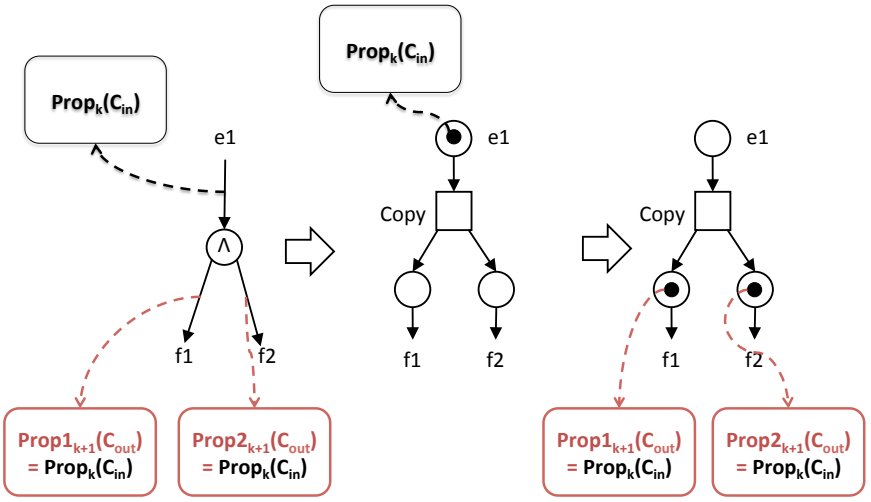


FIGURE 4.9: Colored Token Propagation when Firing an AndSplit

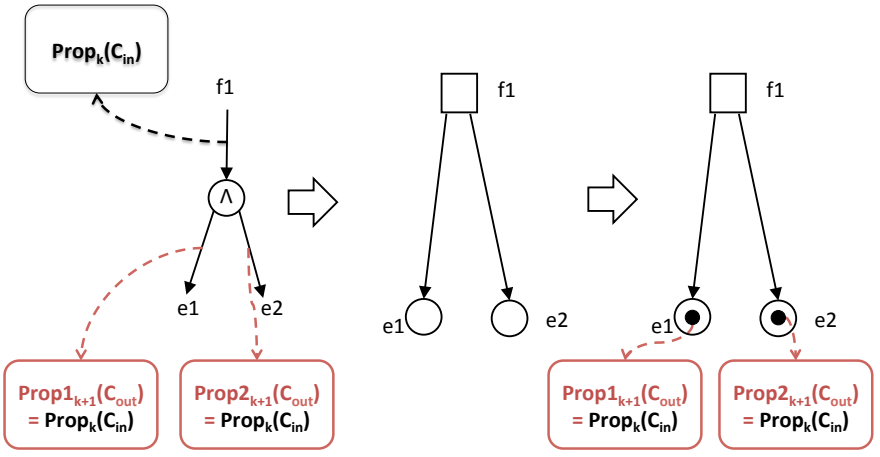


FIGURE 4.10: Colored Token Propagation when Firing an AndSplit

The third iteration of the algorithm when propagating attributes for the example of Figure 4.1 consists of firing the first XORsplit (i.e., C1). The operation here is a simple

copy operation. Both arcs #3 and #5 are now annotated with a copy of the properties from the arc #2.

4.4.2.3 Fired Node is a *Join* Node (i.e., ANDjoin, ORjoin or XORjoin)

The aggregation function depends on the category of the properties $Prop_k(\bullet n)$.

- If exactly 1 property $p \in \bigcup_{c \in \bullet n} Prop_k(c)$ is of category **Dominant** (or **Composed**)
  - This property value overrides all the other alternative values and the resulting property is of category **Dominant** (or **Composed**)
- If there are several properties $p \in \bigcup_{c \in \bullet n} Prop_k(c)$ of category **Dominant** (or **Composed**)
  - The propagated property value will be an enumeration of all the alternative values and the resulting property is of category **Dominant** (or **Composed**)
- If there is no property $p \in \bigcup_{c \in \bullet n} Prop_k(c)$ is of category **Dominant** or **Composed** (Only **passive** attributes)
  - The propagated property value will be an enumeration of all the alternative values and the resulting property is of category **Passive**

Correctness: The informal semantics of the join nodes in EPC is as follows: a join node removes one or many tokens from their incoming arcs and places one token on their outgoing arc depending on the operator and linked nodes. In here formal semantics are required to specify the exact incoming arc to be considered (especially in the case of OR connectors) [219] (see also Figure 4.5). This semantics is valid for monitoring the execution of the graph model by activating the outgoing arc of the connector depending on the used operator. In the case of business capability propagation, all incoming branches should be considered as all of them influence the resulting business capability. In this case, all the properties coming from all incoming arcs need to be reflected on the outgoing arc. Figure 4.11 shows the token propagation strategy used in the algorithm: at a certain join connector, input tokens are consumed and an output token is generated based on the operator discussed previously. Furthermore, as properties can be duplicated from different arcs, the priority control using "dominant", "composed" and "passive" categories automates the selection of their values.

After nine iterations of the propagation algorithm on the running example, the node to be fired is the last XORjoin node. An Override aggregation function is applied on

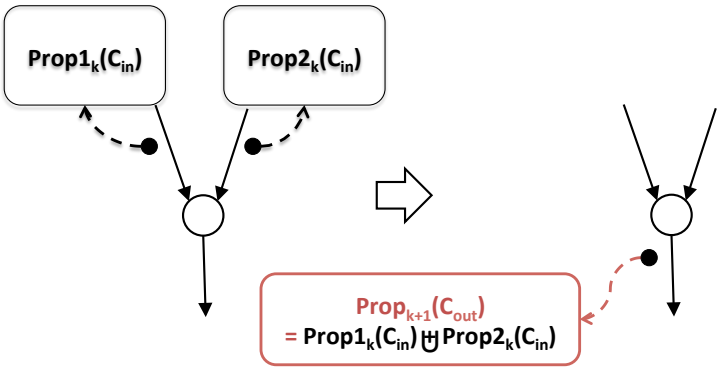


FIGURE 4.11: Properties Propagation when Firing a Join node

the properties from #4 and #16 which is actually in this case a simple union operation (because the shared properties have the same values).

Listing 4.2 represents the set of properties of the aggregated business capability of the entire process models of the running example. Together with the ActionVerb *Examination of Cargo*, this business capability is interpreted as follows: This business capability allows to **examine a cargo**, where an **examination decision** determines if the cargo has to be checked; if a **physical inspection is required** then an **X-Ray scan** is performed; if a **physical inspection is required**, and the **type of check is a Red Check** then a *detailed examination* is done.

```
:Phil_Cap_RunningExample a cmm:Capability;
cmm:achieves imp:ExaminationOfCargo;
impc:hasCargo :Phil_Cargo;
impc:hasExamDecision :Phil_ExamDecision;
impc:hasExamType :Phil_ExamType;
impc:hasTypeOfCheck :Phil_TypeOfCheck.

:Phil_ExamDecision a impc:ExamDecision, cmm:EnumerationValue;
cmm:hasElement :PhysicalInspectionRequired;
cmm:hasElement  impc:PhysicalInspectionNotRequired;
cmm:hasElement impc:PriorityChannel.

:Phil_ExamType a impc:ExamDecision, cmm:EnumerationValue;
cmm:hasElement [     a cmm:ConditionalValue;
                              cmm:hasCondition impc:PhysicalInspectionRequired;
                              cmm:hasCondition impc:RedCheck;
                              cmm:hasValue impc:Detailed. ];
cmm:hasElement [     a cmm:ConditionalValue;
                              cmm:hasCondition impc:PhysicalInspectionRequired;
                              cmm:hasValue impc:X-Ray. ].

:Phil_TypeOfCheck a impc:TypeOfCheck, cmm:ConditionalValue;
cmm:hasCondition  impc:PhysicalInspectionRequired;
cmm:hasValue  [ a cmm:EnumerationValue;
                cmm:hasElement impc:RedCheck;
                cmm:hasElement impc:GreenCheck.].
```

LISTING 4.2: Aggregated Business Capability of the model depicted in Figure 4.1

4.5 Tool Support and Evaluation

4.5.1 Tool Support

The proposed business capabilities aggregation algorithm has been implemented as an extended version of EPCTools. This extension allows users to define the business capability of a business process fragment defined by a start and end node of type event (see area 1 in Figure 4.12). The result is shown to the user as a list of properties hiding all the complexity of RDF as a set of action verbs (see area 2 in Figure 4.12) and set of properties (see area 3 in Figure 4.12). This extension offers the possibility to export the resulting aggregated business capability either in a separate file or the entire business process model using BANG vocabulary shown in Section 4.3.

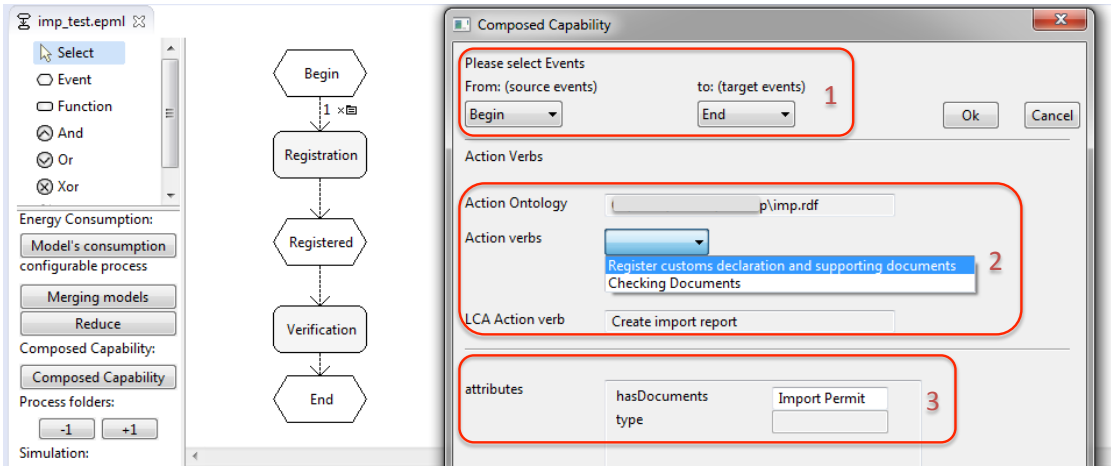


FIGURE 4.12: EPCTools Extension Implementing the Capability Aggregation Algorithm

Please note that the object of this research is not to provide a fully evaluated user interface. It has been only developed to show the applicability of this approach, to carry out some manual verifications of the results and to be used as a visual support for doing interviews with domain experts. Indeed, apart from this running example, I manually tested this algorithm on a set of process models from the customs clearance processes, namely import procedures. The test collection that I have considered in this work includes ten business processes that are available in Appendix A. They describe guidance on the basic regulatory requirements that all importers must consider when they plan to import goods. The import customs clearance involves various steps from submission of import documents until the release of the imported goods. These models were manually annotated using the Import Capabilities Domain Ontology (IMP) available in Appendix D. Further to the manual tests, I carried out interviews with domain experts that are reported in the following section.

4.5.2 Interviews with Domain Experts

4.5.2.1 Introduction

The idea of carrying interviews with domain experts consists of using questionnaires to gather their assessment, attitude, opinion, etc. on the proposed research (e.g., [25]). It is possible to use either structured or semi-structured interviews/questionnaires depending on the type of research and availability of interviewees. Indeed, structured interviews or questionnaires are applicable in a large scale contexts where multiple users can be interviewed [62]. In this research, target users are business experts that are using process modelling tools in their work. Such profiles are not easy to approach and large numbers of interviewees cannot be easily found. Consequently, I aimed to use semi-structured interviews with reduced number of domain experts [62].

In this evaluation, I carried out interviews with five domain experts that have strong background and are currently active in service computing and business process management activities. Their profiles include two information systems architects, one project manager, one IT engineer and one consultant and training expert. I target these four types of stakeholders as each of them has his own perspective and usage of services and business process models.

The objective of this evaluation is to assess the usefulness and intuitive appeal of the proposed aggregation algorithm for identifying the business capabilities of a process model. In this evaluation a design science methodology was followed [241] together with formal guidelines for conducting and reporting case study research [187].

The interviews were done after explanation of this thesis objective and details about the business capability meta-model and the business process overview approach. The main targeted outcome of these interviews is to identify if these experts see that this research is relevant and its output can be used by their companies.

4.5.2.2 Participants

For this semi-structured survey, five participants were recruited from different levels of expertise in the area of service computing and information systems design and development. The age group of these participant is 30-50 years old and their professional background includes a minimum of 5 years experience and are currently active in their field. Their profiles include:

- two system architects (P1 and P2): working as designers of information systems for clients of a multinational company.

- one project manager (P3): leading teams of developers of information systems for the management of seaports in different countries.
- and one IT engineer (4): working as developer in start-up offering automated post services.
- one information system consultant (P5): working as a consultant and trainer in the area of business process management.

4.5.2.3 Approach

The approach used for this evluation follows the case study research process proposed by Runeson and Höst [187]:

- *Case study design*: the objective of the evaluation is to assess the usefulness and intuitive appeal of the proposed aggregation algorithm for identifying the business capabilities of a process model. Interviews run individually using online conferencing tool. Each interview took about 1 hour for each participant.
- *Preparation for data collection*: The discussions were semi-structured to give the participants the freedom to give additional comments and get as much feedback as possible from them. The structure of the interviews was as follows [6]:
  1. 5 minutes discussion about the profile of the participant and his knowledge about services and business processes modelling languages.
  2. 15 minutes presentation of the business capability of atomic and aggregated tasks and the propagation algorithm introduced in this chapter with open discussion.
  3. 15 minutes for manually defining the aggregated capability of the cargo examination model used in this chapter.
  4. 10 minutes demo and interaction with the tool support.
  5. 15 minutes discussion about the proposed business capability aggregation approach.
- *Analysis of collected data*: A post interview analysis of the collected feedback is reported in Section 4.5.2.4.
- *Reporting*: A discussion of the resulting feedback is summarised in Section 4.5.3 and shared among the participants.

[6] Please note that the durations used here are approximative. Some of the interviews run for few minutes more or less for each section.

4.5.2.4 Results

The following outcomes were identified:

Feedback on the business capability modelling approach and its adoption

- The 5 experts agree that the business capabilities modelling proposed in this thesis looks a promising direction towards end-users understanding. *Actions* are what users do in their processes and properties use business terms that these experts are familiar with. Two of the experts highlight that the main advantages that the model brings is the simplicity and extensibility.

  "*It looks like your model can also capture other aspects!*" (P1)

  "*You are proposing a simpler view of what the model achieves.*" (P5)

- The project manager (P3) highlights the fact that a new approach that describes any aspect of business processes is welcome as long as it reuses their existing taxonomies and ontologies. This is in favor of the proposed model as it does not impose any restrictions regarding the list of properties to be used.

- All the experts find that the adoption of this work into their information systems is possible as long as long as it can be adapted to their modelling and annotation techniques.

Feedback on the use of ontologies in industry

- Ontologies and taxonomies are already in use and constitute valuable assets for the companies where these experts work. All the experts use ontologies.

Feedback on the business capability aggregation approach

- Regarding the aggregation work, P1, P2, P3 and P4 find it as a useful feature for users that are developing multiple service-based business processes. It allows not only identifying the business capability of the model but also any other aspect of interest and visualize the parameters used or required by the process.

  "*I can add here another property regarding the data format used in one activity to make sure that it is communicated to the following activities as a requirement.*" (P4)

- The consultant and training expert (P5) finds that the results of aggregation can be further used for documentation purposes. This can help delivering business processes and services documentation quickly.
- P1, P3 and P4 see that abstraction and aggregation techniques are already in use in their working environment. However, they do not use rich descriptions of models and services and thus do not have rich abstracted service or process descriptions.

Feedback on the tool support

- While testing the proposed tool support, all the experts find that it is simple to use and intuitive. It is important to note that in the implemented prototype I used only primitive types.
- Experts do not see using simple types as a major issue as they find that properties are more important to visualize than their values.

  "*At this stage I don't care about the values used, I give more importance to the parameters themselves!*" (P2)

4.5.3 Discussion

It is agreed among the interviewed experts that business process models are central artifacts in Process Aware Information Systems. These models are being managed and maintained by several stakeholders with various needs. The experts also confirmed that business process abstraction is required for having a rapid overview of process models. Furthermore, overviews with rich representation of the business capability of an abstract model gives a better understanding of the corresponding process. A rich model can then easily be transformed into a complete documentation using Natural Language Generation techniques [129, 132].

Business Process Abstraction can be technically implemented via two operations: elimination and aggregation [210]. Elimination omits unwanted model elements [72], however, aggregation makes a process model more coarse-grained [32, 172, 179, 208, 209, 228]. In the best case, the aggregation operation allows to transform an entire process model into a single high-level activity. In my discussions with domain experts, we highlighted the fact that elimination techniques are good only for hiding parts without effectively removing them from the model otherwise the semantics of the entire model can be altered. Experts acknoledge the wide support of abstraction in process modelling languages such as BPMN and EPC that allow to represent aggregated tasks for encapsulating process

fragments. On the other hand, there are little efforts for automatically generating these abstracted process fragments including a rich description of their business capabilities.

Further to these discussions, I was particularly interested in exploring with the experts two main abstraction techniques that are highly related to my work: Meronymy-based abstraction of business process models [208] and Precondition and Effect propagation for process abstraction [228].

Meronymy (part-of) relations between activity labels is investigated by Smirnov et al. [208] in order to capture granularity relation between activities at several levels of abstraction. I currently, use a similar approach for detecting the action verb of the aggregated business capability of the entire process model. The interviewed experts find that this work easy to implement but limits the aggregated activity into a simple label which is not enough to describe the business capability of the resulting aggregated tasks. Benefiting from this approach and using it in my work was a legitimate decision to further extend the resulting label with business properties.

While discussing the contribution of Vulcu et al. [228], the experts consider that it provides more elaborated model properties than other abstraction techniques. By model properties, experts refer to Inputs, Outputs, Preconditions and Effects as well as Quality of Service properties. The experts point the weakness of this effort in providing complex logical expressions in the propagation results that need extensive analysis and a dedicated reasoner to interpret the precondition and effect. Moving from representing business capabilities using Preconditions and Effects to structured business capability removed the complexity of the results and makes the aggregated business capabilities easier to read and interpret.

4.6 Summary

In this chapter, I defined a business capabilities aggregation algorithm that takes as input a business capability annotated process model and returns its aggregated business capability. The algorithm assumes that the input model is well structured and annotated using the business capability meta model introduced in Chapter 3. The limitation that input models should be well-structured can be resolved. Indeed, most models from practice can be easily made structured using graph parsing techniques. Respective techniques have been formalized and implemented in libraries [173, 174, 225].

The algorithm operates in two steps. First, it determines the action category of the input model using the lowest common ancestor of the action categories of its activities. Then, it propagates the properties of capabilities starting from an initial to a final

node. The idea of the propagation is inspired from the token game similar to Petri Nets [170] using formal semantics of each node of the model. Streit et al. [213] propose an alignment between control-flow of EPC and PetriNets from an end-user perspective. Their experiment illustrates the intention to use EPC notation is higher. Therefore, using EPC for process models in the interviews with domain experts was chosen.

On its own, this work helps defining a high level description of business process models leaving out all the structural aspects while focusing on the business capability. The proposed approach can also be extended with schema reduction techniques [225] for representing process models at multiple levels of abstraction. This guarantees the optimisation of process models while providing a rich description of aggregated activities.

As the business capability description is a structured entity defined in RDF, further use cases can extend this work, for example documentation of a business process, comparison and configuration of business processes. In Chapter 6, I will use this work for deriving configuration options, featuring business capability properties, in order to help end-users in the customization of configurable business process models.

Chapter 5

Indexing and Discovering Capabilities

"An index is a great leveller."

George Bernard Shaw

5.1 Introduction

With the trend of Industry 4.0 [124], with the advent of the Internet of Things [95], and with the decreasing costs, and increasing capabilities of sensors and smart devices, modern businesses are integrating more and more live data into their business processes [142]. New challenges facing modern business processes include dynamic and efficient discovery of resources such as data sources and services [95]. Indeed, such processes rely on sensor data to provide necessary business intelligence to support decision making. A possible use case can be a smart building within an energy management application where a decision support model is used to control the supply and demand of energy.

The main source of information used for decision support models within smart buildings is the sensors. An efficient decision support model in such a context requires that sensor data is provided correctly and timely. However, accidents may occur at any time. For example a sensor may become unresponsive or source of data errors. In these cases, the decision support model should provide suggestions to use another source of data. This can be simplified if sensors are properly described and organised. Creating explicit links between sensors helps to discover similar ones and consequently facilitate balancing observations from one sensor to the other.

Given the dynamicity of the sensor environment, the diversity of their features and of user requirements, finding appropriate sensors having the required capabilities or replacing faulty ones constitute a challenging task especially in medium and large scale areas. Efficiently describing and indexing sensors in smart environments is essential to deliver a rapid adaptation to errors and the availability of data.

In this chapter, I present an approach for indexing sensor services based on their capabilities. I describe sensor capabilities using the capability meta-model introduced previously in Chapter 3. Then, I apply Formal Concept Analysis [80] (FCA for short) for indexing sensor services based on these capabilities. FCA is a well-known mathematical classification tool used in various domains that allows organizing objects described via a set of attributes into a Concept Lattice.

The remainder of the chapter is organised as follows. Section 5.2 revisits the theoretical foundations of FCA and shows how to apply it for indexing sensors based on their capabilities into a concept lattice. It also shows how this concept lattice can be used to discover sensors and relationships between sensor properties. Section 5.3 details the evaluation of this work. First, section 5.3.2, introduces the Linked Energy Intelligence dataspace which constitutes the use case for organizing sensor capabilities using FCA. Then section 5.4 details two experimentations for verifying the applicability of the proposed approach. Finally, section 5.5 draws conclusion and details future work.

5.2 Formal Concept Analysis for Organizing Sensor Capabilities

The approach in this work utilises Formal Concept Analysis [80] (FCA for short) to better organize a repository of capabilities in order to make their discovery more efficient. In this section, I define the theoretical foundations of FCA while applying it on sensors capabilities. I use FCA in Section 5.2.1 for creating a concept lattice, a structure that allows for indexing sensor capabilities. Then in Section 5.2.2, I discuss how a discovery mechanism can be implemented using this concept lattice.

5.2.1 Creating the Concept Lattice

FCA is a technique that has evolved from mathematical lattice theory and has been used for data analysis across several domains. Examples include organizing web search results into concepts based on common topics, gene expression data analysis, information retrieval, understanding and analysis of source codes, etc. [19]. It represents a powerful

tool for identifying meaningful relationships within a set of objects that share common attributes. It provides a theoretical model to build from a *formal context* (see Definition 10) a partially ordered structure called a *concept lattice*.

Definition 10 (Formal Context). A *formal context* $\mathcal{FC}$ is a triplet $< X, Y, R >$ where X and Y are non-empty sets and $R \subseteq X * Y$ is a binary relation between X and Y.

For a formal context $\mathcal{FC}$, elements $x \in X$ are referred to as objects and elements $y \in Y$ are called attributes. $< x, y > \in R$ denotes that the object x has the attribute y.

In this work, the formal context is defined via the set of sensors as well as their respective descriptions. Table 5.1 (called cross-table) will be used in this section as a running example which describes the relationship between the objects (i.e., sensors 1 to 5 represented by the table rows: X ={Sensor 1, Sensor 2, Sensor 3, Sensor 4, Sensor 5}) and their descriptions (i.e., attributes represented by the table columns: Y ={Active, Storage Option, Digital Display, Accessible}, in Table 5.1). This example considers the following four attributes:

- *Active* that indicates if the sensor is in operation;
- *Storage Option* that indicates if the sensor has the possibility to store data on it;
- *Digital Display* that indicates if the sensor is equipped with a digital display for displaying the data; and
- *Accessible* that indicates if the sensor is located in an accessible area.

TABLE 5.1: Data Table with Binary Attributes for Sensors

Objects	Active	Storage Option	Digital Display	Accessible
Sensor 1	X	X	X	X
Sensor 2	X		X	X
Sensor 3		X	X	X
Sensor 4		X	X	X
Sensor 5	X			

Another fundamental concept in FCA is the *Formal Concept*. This concept is defined in Definition 11.

Definition 11 (Formal Concept). A *formal concept* in $< X, Y, R >$ is a pair $< E, I >$ of $E \subseteq X$ (called *extent*) and $I \subseteq Y$ (called intent) such that $Att(E) = I$ and $Obj(I) = E$.

$Att(E)$ is an operator that assigns subsets of X to subsets of Y, such that $Att(E)$ is the set of all attributes shared by all objects from E. $Obj(I)$ is an operator that assigns

subsets of Y to subsets of X, such that $Obj(I)$ is the set of all objects sharing all the attributes from I.

From Definition 11, one can conclude that a concept $C =< E, I >$ is created by getting objects from E sharing the same attributes from I. For example, the shaded rectangle in Table 5.2 represents a formal concept $< E_1, I_1 >=<${Sensor 1, Sensor 2, Sensor 3, Sensor 4}, {Digital Display, Accessible}$>$ because $Att(E_1) = \{DigitalDisplay, Accessible\}$ and $Obj(I_1)=$={Sensor 1, Sensor 2, Sensor 3, Sensor 4}.

TABLE 5.2: Data Table with Shaded Srea Representing an Example of Formal Concept

Objects	Active	Storage Option	Digital Display	Accessible
Sensor 1	X	X	X	X
Sensor 2	X		X	X
Sensor 3		X	X	X
Sensor 4		X	X	X
Sensor 5	X			

From a *formal context* $\mathcal{FC} =< X, Y, I >$ one can deduce a set of formal concepts that can be ordered with respect to a subconcept ordering. Definition 12 formally introduces the subconcept ordering.

Definition 12 (Subconcept Ordering). Having two formal concepts $< E_1, I_1 >$ and $< E_2, I_2 >$ from $\mathcal{FC} =< X, Y, R >$, $< E_1, I_1 > \leq < E_2, I_2 > \iff E_1 \subseteq E_2$ ($\iff I_2 \subseteq I_1$).

Let's consider the following formal concepts from the example in Table 5.1:

$< E_1, I_1 >=<${Sensor 1, Sensor 2, Sensor 3, Sensor 4}, {Digital Display, Accessible}$>$

$< E_2, I_2 >=<${Sensor 1, Sensor 2, Sensor 4}, {Digital Display, Accessible}$>$

$< E_3, I_3 >=<${Sensor 1, Sensor 2}, {Active, Digital Display, Accessible}$>$

$< E_4, I_4 >=<${Sensor 1, Sensor 2, Sensor 5}, {Active}$>$

Then

$< E_3, I_3 > \leq < E_1, I_1 >$, $< E_3, I_3 > \leq < E_2, I_2 >$,

$< E_3, I_3 > \leq < E_4, I_4 >$ and $< E_2, I_2 > \leq < E_1, I_1 >$.

The set of ordered formal concepts derived from a formal context is called a *concept lattice* which is another important notion in FCA. A concept lattice can be represented into a graph such as the one depicted in Figure 5.1[1]. In this figure, the concept extent

[1] All concept lattices in this paper are created using Conexp [202].

near the bottom of the lattice contains only *Sensor 1* since the corresponding intent is related to the biggest number of attributes. The top concept contains all the sensors and its intent corresponds to no attribute. This makes the concept less interesting as it allows for all possible combinations of attributes.

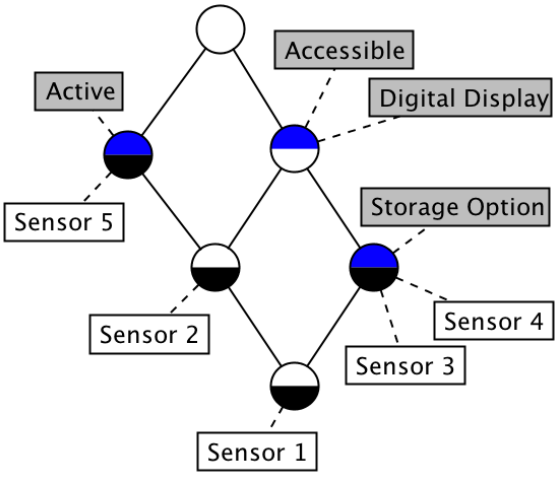


FIGURE 5.1: Concept Lattice of the Example of the Table 5.1.

So far, I considered binary attributes (i.e., either the object has or has not that attribute). However, in real settings when describing capabilities, there are also multi-valued attributes. Consider Table 5.3, this table contains an additional attribute *Observed Phenomenon*. This attribute reports whether the sensor is an *Energy* consumption sensor, *Light* detection sensor, *Temperature* sensor or a *Motion* sensor. In this case, we need to transform this multi-valued attribute into a binary attribute.

TABLE 5.3: Data Table with a Multi-valued Attribute of Sensors

Objects	Active	Storage Option	Digital Display	Accessible	Phenomenon Observed
Sensor 1	X	X	X	X	Energy
Sensor 2	X		X	X	Energy
Sensor 3		X	X	X	Light
Sensor 4		X	X	X	Temperature
Sensor 5	X				Motion

For the usage of FCA, transforming and preprocessing the data displayed in Table 5.3 is needed. One possible way consists of using a scaling method. Scaling is a transformation method that converts a multi-valued attribute into a context. Table 5.4[2] represents the transformation of the multi-valued attribute *Phenomenon Observed* into a context.

[2]Please note that PO stands for Phenomenon Observed

TABLE 5.4: Data Table with a Scaled Multi-valued Attribute for the Phenomenon Observed

Objects	PO: Energy	PO: Light	PO: Temperature	PO: Motion
Sensor 1	X			
Sensor 2	X			
Sensor 3		X		
Sensor 4			X	
Sensor 5				X

After the application of FCA on the tables, the resulting lattice is depicted in Figure 5.2.

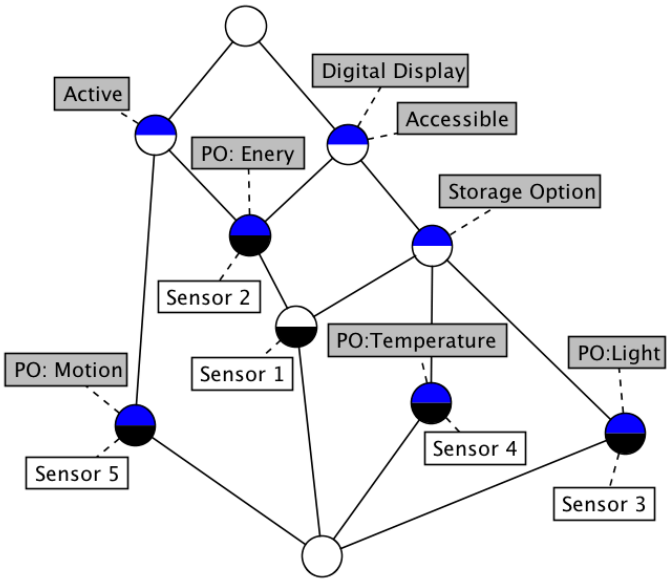


FIGURE 5.2: Concept Lattice of the Example in Table 5.3 of Sensors with Phenomenon Observed

This concept lattice is an indexing structure, it allows the organization of sensor capabilities in a tree. This structure can serve for the discovery of sensors as described in the following sub-section.

5.2.2 Concept Lattice for Sensor and Knowledge Discovery

In the following, I show the usefulness of using FCA for indexing sensor descriptions via two scenarios. The first is the discovery of sensors and the second is the discovery of implications between sensor attributes.

I propose algorithm 1 for the discovery of sensors satisfying a set of attributes. It takes as input the concept *Lattice* and a set of *attributes* representing the query. Suppose

that the input Lattice is the one depicted in Figure 5.3 and the input attributes are *"PO:Energy"*, *"Digital Display"* and *"Storage Option"*. The operation of Algorithm 1 is as follows:

1. Lines 1-4: find the set of formal concepts with an intent that contains the input attributes. The result of this step as shown in Figure 5.3, is the set of formal concepts **FC1**, **FC2** and **FC3**.
2. Line 5: find the Highest Common Subconcept of the concepts identified in the first step. In Figure 5.3, this can be determined by following the lines down from FC1, FC2 and FC3 and stopping where they meet. The result is **FC4**.
3. Lines 6-10: collect the set of potential candidates of the query. Every object in the formal concept identified in step 2 as well as all its subconcepts down to the bottom of the lattice are potential candidates for the input query. In Figure 5.3, starting from FC4, *"Sensor 1"* is the only result for our input query as there are no subconcepts of FC4 with non-empty extent.

Algorithm 1: Sensor Discovery Algorithm using Formal Concept Analysis

Input: Lattice L: A concept lattice that represents the indexing structure.
Attributes ATTS: the list of attributes of the search request.
Result: List of sensors that satisfy the search request.

```
1  Concepts ← null;
2  foreach (Attribute ∈ ATTS) do
3  |  Concepts.add(L.findConceptWithAttribute(Attribute));
4  end
5  Concept ← FindHCSubC(L, Concepts);
6  SubConcepts ← Concept.getSubConcepts();
7  Sensors ← null;
8  while SubConcepts.size() ≠ 0 do
9  |  OneConcept ← SubConcepts.getConcept();
   |  Sensors.addAll(OneConcept.getObjects());
   |  SubConcepts.addAll(OneConcept.getSubConcepts());
   |  SubConcepts.remove(Concept);
10 end
11 return Sensors
```

The proposed algorithm relies mainly on the explicit relations between formal concepts. This is useful to discover sensors that share similar attributes. For example, if one of the motion sensors M is not active anymore, it is possible to use one of the other motion sensors in its equivalence class or discover sensors that share its attributes (i.e., attributes of the sensor M) by using them as input for Algorithm 1.

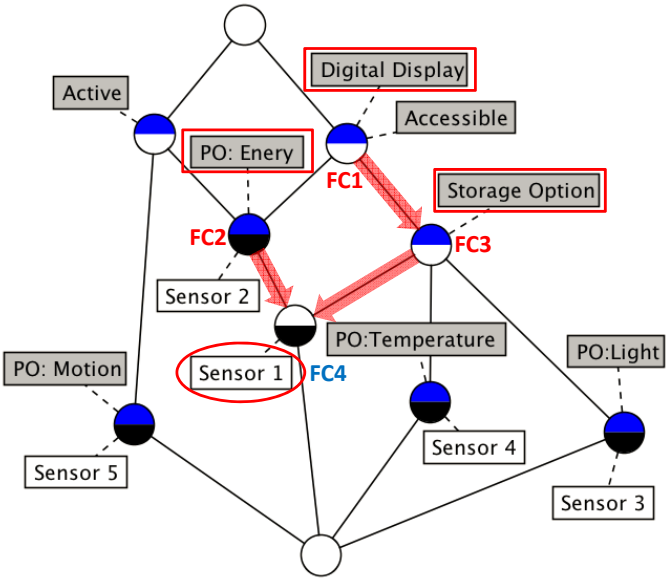


FIGURE 5.3: Discovering Sensors with same Attributes in the Concept Lattice

In the context of the replacement of a sensor, this method reduces the change time [3] considerably to simplify parsing of the lattice until it reaches the required equivalence class and select one of its sensors rather than performing a full search over the set of all the available sensors. It helps avoid having empty results. In fact, during the navigation of the concept lattice, if the users cannot find the equivalence class that satisfies their request, they can adapt it according to the visited nodes of the lattice. This allows the user to relax their query by reducing the attributes initially identified in the request.

The other advantage of using FCA is the presence of the explicit subconcept relationship between equivalence classes. This allows the discovery of additional knowledge among the objects' attributes that are analyzed (i.e., sensor attributes). Indeed, as depicted in Figure 5.4, one can discover implications such as: every sensor that has a *"Storage Option"* is also *"Accessible"* and has a *"Digital Display"*. In other words: *"Storage Option"* implies *"Accessible"* and *"Digital Display"*.

To conclude, it is important to notice that the use of FCA permits the creation of a concept lattice uniformly. In other words, it always creates the same structure with the same input objects. This has the advantage of creating a deterministic discovery algorithm, as there is no need to use any heuristic for parsing this indexing structure. This chapter focuses mainly on the creation of the concept lattice and the study of its applicability for indexing a set of sensors capabilities. I have used FCA in real settings for organizing sensors capabilities and the experimental settings are described in detail in Section 5.3.

[3] Change time: the required time for selecting a replacement sensors for the disabled one.

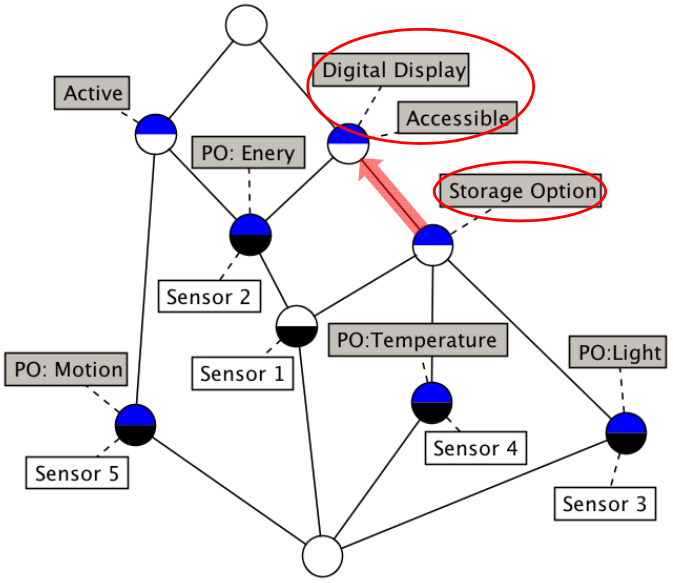


FIGURE 5.4: Example of Implication: Every Sensor that has a *"Storage Option"* is *"Accessible"* and has a *"Digital Display"*

5.3 Implementation and Use Case

In order to evaluate the approach proposed in this chapter, I developed multiple modules for creating a concept lattice starting from an RDF description of sensor services. The workflow as well as the data exchanged between the various modules is shown in Figure 5.5.

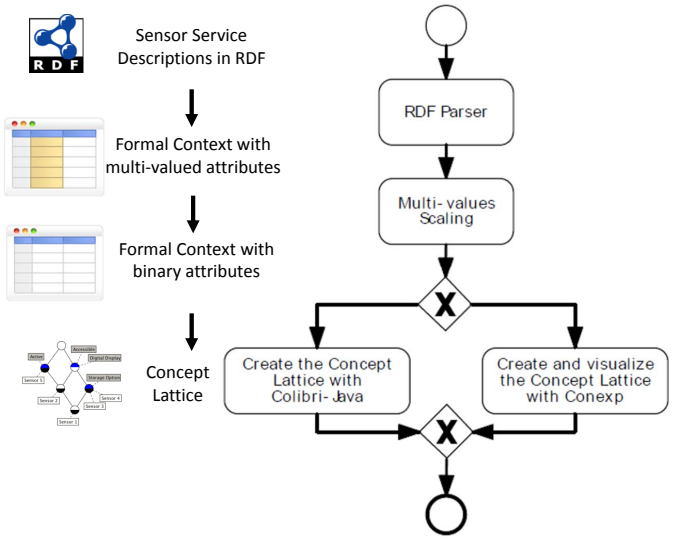


FIGURE 5.5: Creating a Concept Lattice from RDF Descriptions of Sensor Services

The first developed tool is the RDF parser. It is a Java application that uses Apache Jena Framework, a Java RDF and Semantic Web library. This module takes as input an RDF file and produces a text file containing a formal context as a table with multivalued attributes. This module can be further improved to include data from an RDF store and not simply an RDF file. It is custom made for this application so it can parse only sensor descriptions defined using the Sensor Capability Ontology that respects the capability meta-model proposed in Chapter 3. Details about the Sensor Capability Ontology are in Section 5.3.1.

The second module of the prototype is another Java application that performs the scaling operation described in Section 5.2. It starts by checking all the attributes that are not boolean (see Listing 5.2 for an example) then it considers each of its values as a separate attribute and assigns a boolean value (i.e, true) to the corresponding objects. The output of this module is a textual file compatible with the Conexp tool format [202] that is used in the following step.

The third step of this work consists of creating the concept lattice with one of the following options: (i) Using Conexp [202] for the creation and visualization of the concept lattice. (ii) Using Colibri-Java [93] for the creation and the analysis of the resulting concept lattices. I use the first option for applying the proposed approach on a real world scenario that will be described in detail in Section 5.3.2 and the second option for carrying out further statistical analysis on the use of FCA in indexing synthetic sensors descriptions in Section 5.4.

5.3.1 Sensor Capability Ontology

The sensor capability ontology extends the capability model shown in Chapter 3. Listing 5.2 gives a snippet in N3 format of our RDF Sensor Capability Ontology which is also illustrated graphically in Figure 5.6. The class of all sensor capabilities `SensingCapability` is defined as sub class of the concept `Capability` (Listing 5.2, line 9). As all sensors are supposed to provide data about a phenomenon they are observing, the action category *sensing* is defined and assigned to this class as a value of the property `cmm:achieves` (Listing 5.2, line 11 - 14).

The second step consists of defining features of interest for sensor capabilities. As presented above, these features are specified as property declarations. I distinguish between valued and non-valued features. Non-valued features are features which are either present/fulfilled or not present such as `hasStorageOption`. The range of these properties is `boolean`. I define the following property declarations (see Figure 5.6).

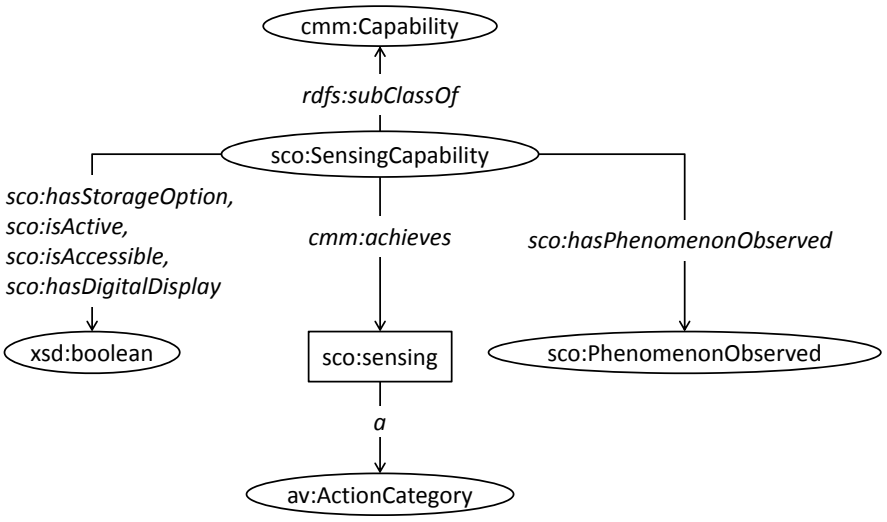


FIGURE 5.6: Sensor Capability Ontology

- *sco:isActive* (line 18) is defined as a property that has a boolean value. It reports if the sensor is active.
- *sco:hasStorageOption* (line 22) is defined as a property that has a boolean value. It reports if the sensor has any storage option. This property can be modified to report on the size of the storage capacity of the sensor.
- *sco:isAccessible* (line 26) is defined as a property that has a boolean value. It reports if the sensor is accessible. This helps to take decisions to physically move and check the status of the sensor or read directly from its digital display if it has one.
- *sco:hasDigitalDisplay* (line 30) is defined as a property that has a boolean value. It reports if the sensor has a digital display that a user can read from.
- *sco:hasPhenomenonObserved* (line 34) is defined as a property that has a string value. It reports on the phenomenon that the sensor is observing. A listing of possible values is defined as a Datatype (line 39).

A sensor capability is created as an instance of (*rdf:type*) *sco:SensingCapability* with concrete values of its predefined properties. Listing 5.1 presents an example of a temperature sensor capability *:TemperatureSensorCapability123* reporting that it is an active and accessible temperature sensor with a digital display and it does not have a storage option.

Compared to existing approaches, this capability model presents several advantages. Mainly, it explicitly captures domain-specific functional properties which describe and characterise the carried action according to the aspects of interest to end-users and

```
@prefix sco:    <http://vocab.deri.ie/sco#>.

:TemperatureSensorCapability123 a sco:SensingCapability;
                sco:isActive "true"^^xsd:boolean ;
                sco:hasStorageOption "false"^^xsd:boolean ;
                sco:isAccessible "true"^^xsd:boolean ;
                sco:hasDigitalDisplay "true"^^xsd:boolean ;
                sco:hasPhenomenonObserved "Temperature"^^sco:PhenomenonObserved.
```

LISTING 5.1: Snippet of a Temperature Sensor Capability in the sensor capability ontology

targeted applications. Indeed, these properties are defined in domain-specific ontologies with respect to specific engineering tasks. Moreover, this domain-specific capability model is easily extensible. If a new property is required for describing a particular sensor aspect/characteristic, it simply needs to be defined as a new *cmm:PropertyDeclaration* with its corresponding domain and range. Finally, this feature-based model enables new techniques for indexing and discovering services as experimented in this chapter.

5.3.2 Use Case Application

This section illustrates a use case scenario using a set of real world sensors deployed within the Linked Energy Intelligence (LEI) dataspace. LEI is an ecosystem where energy related data is made available and interlinked to support decision making and ultimately energy consumption friendly behaviour [37]. Such data is provided by real-time data sources such as sensors as well as relatively static background knowledge such as building plan and occupancy. The LEI dataspace has been realized at the INSIGHT Centre for Data Analytics (previously known as the Digital Enterprise Research Institute (DERI)) at the National University of Ireland, Galway (NUIG).

INSIGHT @ NUIG is a premier research institute with approximately 130 research students and staff with a worldwide reputation in its area. It is based in a dedicated building with 2190 sq. m of space, comprising 22 unit offices, 160 open plan workspaces, 1 large 80-seat conference room with audio visual and video conferencing facilities, 4 meeting rooms, 3 kitchens, 1 air conditioned data centre with backup generator, 1 sensor network laboratory, a 30 person café, and Ireland's National Museum of Computing History.

There are various sources of power consumption in INSIGHT @ NUIG such as Heating, Ventilation and Air Conditioning (HVAC) systems, lights, and electronic devices. The building provides a first-class technical infrastructure to its researchers.

The INSIGHT @ NUIG building has been retrofitted with energy sensors to monitor the consumption of power within the building. In total there are over 50 fixed energy

```
@prefix sco: <http://vocab.deri.ie/sco#>.
@prefix cmm: <http://vocab.deri.ie/cmm#>.
@prefix av: <http://vocab.deri.ie/av#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#>.

sco:SensingCapability rdfs:subClassOf cap:Capability .

sco:sensing a av:ActionCategory ;
        rdfs:comment "Measuring a physical quantity and converts
        it into a signal which can be read by an observer."^^xsd:string ;
        rdfs:label "Sensing"^^xsd:string .

sco:SensingCapability cmm:achieves sco:sensing .

sco:isActive a cmm:PropertyDeclaration;
        rdfs:domain sco:SensingCapability;
        rdfs:range xsd:boolean.

sco:hasStorageOption a cmm:PropertyDeclaration;
        rdfs:domain sco:SensingCapability;
        rdfs:range xsd:boolean.

sco:isAccessible a cmm:PropertyDeclaration;
        rdfs:domain sco:SensingCapability;
        rdfs:range xsd:boolean.

sco:hasDigitalDisplay a cmm:PropertyDeclaration;
        rdfs:domain sco:SensingCapability;
        rdfs:range xsd:boolean.

sco:hasPhenomenonObserved a cmm:PropertyDeclaration;
        rdfs:label "hasPhenomenonObserved";
        rdfs:domain sco:SensingCapability;
        rdfs:range sco:PhenomenonObserved.

sco:PhenomenonObserved a rdfs:Datatype;
        rdfs:comment "An observed phenomenon can be light,
        motion, temperature, etc." ;
        owl:onDatatype xsd:string;
        owl:withRestrictions ("Light"^^xsd:string "Motion"^^xsd:string
        "Temperature"^^xsd:string "Energy"^^xsd:string).
```

Listing 5.2: Snippet of SCO: Sensor Capability Ontology

consumption sensors covering office space, café, data centre, kitchens, conference and meeting rooms, computing museum along with over 20 mobile sensors for devices, light and heaters energy consumption as well as light, temperature and motion detection sensors. A building-specific aspect of the dataspace has been presented in [38] with a sensor network-based situation awareness scenario presented in [98]. In total, this work used a total number of 78 sensors.

These sensors are described via a set of attributes:

- *Active*: This attribute reports whether the sensor is in operation.
- *Observed Phenomenon*: we have four observed phenomena which are *"energy and power consumption"*, *"motion"*, *"light"* and *"temperature"*. This attribute is a

multivalued attribute that needs to be scaled using the transformation previously shown on Table 5.4.

- *Protocol*: This attribute indicates the protocol used by the sensor. We have in our selection of sensors two possible protocols: UDP used by electricity and power consumption sensors and CoAP used by other sensors. This is a multi-valued attribute that has to be scaled.

- *Electricity Phases*: This attribute reports on the electricity phases used by the sensor, we have in our use case two options: 3-phases and 1-phase sensors. Again this is a multi-valued attribute that has to be scaled.

- *Location*: even though this attribute is not an intrinsic property of the sensor, we have used it because it is important information that is required for processing the data provided by the sensor. This is also a multi-valued attribute that enumerates the locations of the sensors, e.g., 1st floor: west wing, ground floor: canteen, etc. that needs to be scaled.

As previously mentioned, the advantage of the capability model of the previous chapter is that it can easily be extended to add more domain specific attributes. The current use case requires additional attributes: Protocol, Elasticity Phases and Location that are added to the domain ontology shown in Listing 5.2. Current changes to this domain ontology are shown in Listing 5.3. This listing uses geo as a namespace for referring to an existing RDF vocabulary for representing information about spatially-located things, using WGS84 as a reference datum [229]. It is also possible to customize further this attribute for example to the rooms vocabulary [39] if locations of sensors are limited to predefined rooms.

All the sensor capabilities were automatically generated from an Excel file containing the original descriptions that were manually checked. Manually checking RDF descriptions was possible as the number of sensors used was limited. I have not carried out any evaluation of the developed RDF parser, because it is custom made for the used data set and conceptual model. The correctness of the algorithm I applied for the RDF parser is out of the scope of this chapter, however, the data has been manually verified after parsing and scaling.

The resulting concept lattice from Conexp [202] is depicted in Figure 5.7. The top concept in this lattice represents the set of all active sensors <{Sensor 1, Sensor 2, ... Sensor 78}, {Active}>. This formal concept contains in its extent all the sensors of the dataset because they are all active. One can see in this concept lattice several formal concepts that represent the set of motion sensors <{Sensor 61,... Sensor 66}, {OP:

```
@prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>.

sco:hasProtocol a cmm:PropertyDeclaration;
        rdfs:domain sco:SensingCapability;
        rdfs:range sco:Protocol.

sco:Protocol a rdfs:Datatype;
        rdfs:comment "A protocol can be either COAP or UDP" ;
        owl:onDatatype xsd:string;
        owl:withRestrictions ("UDP"^^xsd:string "COAP"^^xsd:string).

sco:hasElasticityPhases a cmm:PropertyDeclaration;
        rdfs:domain sco:SensingCapability;
        rdfs:range sco:ElasticityPhases.

sco:ElasticityPhases a rdfs:Datatype;
        rdfs:comment "Elasticity Phases considered are 3-phaes or 1-phase" ;
        owl:onDatatype xsd:string;
        owl:withRestrictions ("3-phases"^^xsd:string "1-phase"^^xsd:string).

sco:hasLocation cmm:PropertyDeclaration;
        rdfs:domain sco:SensingCapability;
        rdfs:range geo:SpatialThing.
```

Listing 5.3: Snippet of SCO (Sensor Capability Ontology) with the required extensions for the LEI Use Case

Motion}>, the formal concept for temperature sensors <{Sensor 67,... Sensor 72}, {OP: Temperature}> and the light sensors <{Sensor 73,... Sensor 78}, {OP: Light}>. These three formal concepts are all subconcepts of the concept <{Sensor 61, ... Sensor 78}, {1st Floor:East Wing}>. This helps to deduce that all motion, temperature and light sensors are in the same location, i.e., 1st Floor : East Wing.

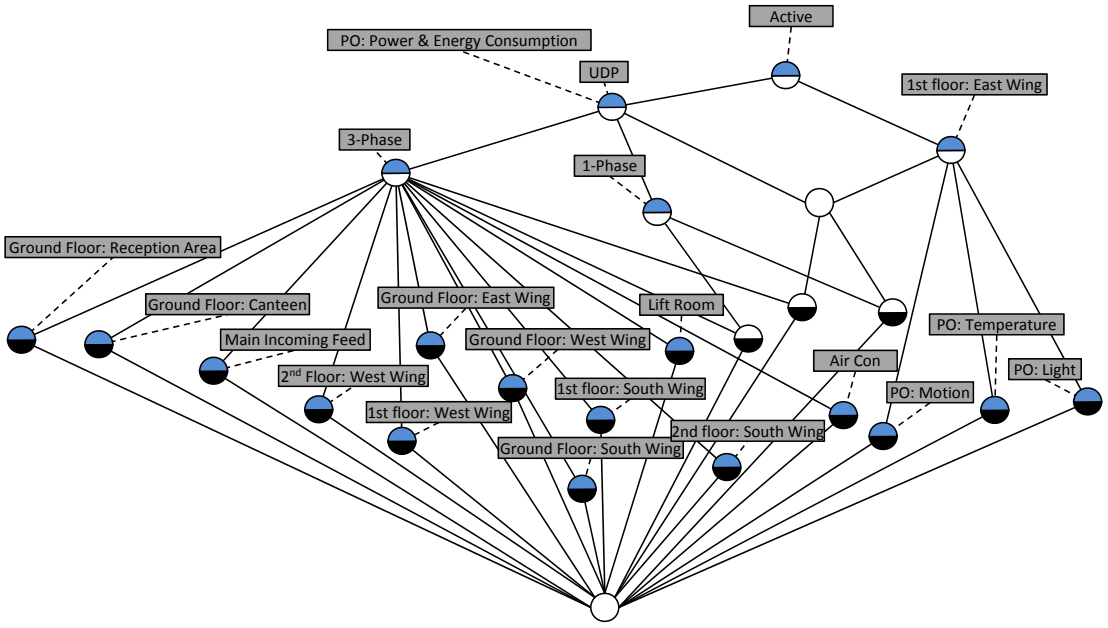


Figure 5.7: Concept Lattice of the LEI use case

5.4 Evaluation

In order to evaluate the applicability of the proposed approach in highly dynamic environments, I carried out two experiments highlighting mainly the efficiency of using Formal Concept Analysis in terms of the number of concepts created in a concept lattice given a formal context and the time required to build it. Throughout this evaluation, I reused an existing implementation of FCA in Java, namely Colibri-Java [93]. Colibri-Java is a library that offers the required tools from the preparation of the context to the creation of a concept lattice that has been experienced in [134].

5.4.1 Experiment 1: Context size vs. Lattice size

The object of this first experiment is to analyse the size of the generated concept lattice with respect to its original formal context and find out the limits of using FCA in this domain application.

During this experiment, I wanted to verify the correlation between the context size and the corresponding lattice size. I randomly created multiple sets of sensor capabilities. For each set I generated its corresponding concept lattice. In terms of attributes, I considered a total number of 16 attributes for describing each sensor capability with three different coverage ratios ranging from 0% to 60%, 50% or 30% (i.e., each sensor capability has between [0,10], [0,8] or [0,5] attributes respectively).

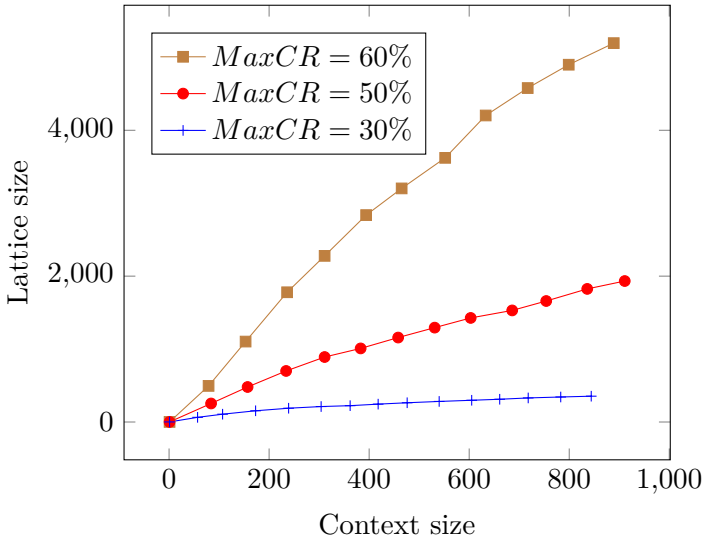


FIGURE 5.8: Context size vs. Lattice size vs. Maximum Coverage Ratio (Max CR)

Figure 5.8, shows the results of this evaluation. The horizontal axis presents the size of the original formal context that varies from 0 to 1000 objects and the vertical axis

presents the size of the corresponding concept lattice. From this figure one can clearly notice that concept lattices grow considerably in size with respect to their context. This is recognised as one of the problems of using FCA. Indeed, in the worst case, the size of the concept lattice is exponential in the size of the formal context [36]. Hence, the literature proposes methods for restructuring this concept lattice by hiding/aggregating some of its nodes in order to be able to visualize them [36]. This leads to the conclusion that if we want to consider applying FCA in an environment with a large number of sensors, it would not be very easy for an end user to visualize the different concept classes generated. Thus this is a limitation from a visualization perspective, and even in related indexing approaches, the visualisation of the indexing structure has never been a target. On the same figure, we can also notice the impact of the coverage ratio. We see that the concept lattice gets larger when large contexts have bigger coverage ratio. This means that when describing sensors, we have to avoid over describing them and carefully choose the most discriminating attributes.

It is important to notice that a discriminating attribute might generate a big number of scaled values. If a certain attribute has a big number of possible values that need to be scaled, each value will generate a new attribute in the considered context. This can be seen as a problem in increasing the number of attributes and consequently generating new equivalent classes from the context and requires building the entire lattice again when a new attribute value is introduced. At the time, in the same context, the new considered value contributes to reducing the coverage rate of attributes on the description of objects. This has a direct consequence on the reduced number of the lattice size as it has been shown in Figure 5.8.

5.4.2 Experiment 2: Lattice size vs. Construction Time

The object of the second experiment is to measure the required time for creating a concept lattice with respect to its size in order to verify the applicability of the approach at a large scale (big number of sensor capabilities) and dynamic environments.

Please note that for these experiments, I ignored the required time for creating a context starting from the RDF descriptions of sensor capabilities. It focuses only on the computation time required for the creation and parsing of a concept lattice. In other words, these experiments focus on the third step of diagram depicted in Figure 5.5.

I randomly created multiple sets of sensor capabilities with a fixed coverage ratio of the attributes between 0% and 50% (each object has between 0 and 8 out of a total of 16 attributes). For each set I generated its corresponding concept lattice and measured the required construction time.

Figure 5.9 shows the results of this evaluation. The horizontal axis presents the size of the concept lattice and the vertical axis present the required time for its construction. From this figure we can clearly see that the required construction time grows exponentially depending on the size of the concept lattice. However, for a concept lattice with over 5000 concepts, the construction time is still less than 200 milliseconds. This time can be considered acceptable in small or medium size buildings where decision making can be postponed until the data has been updated within a few seconds. Nevertheless, in highly sensitive environments, even a few milliseconds can have a huge impact.

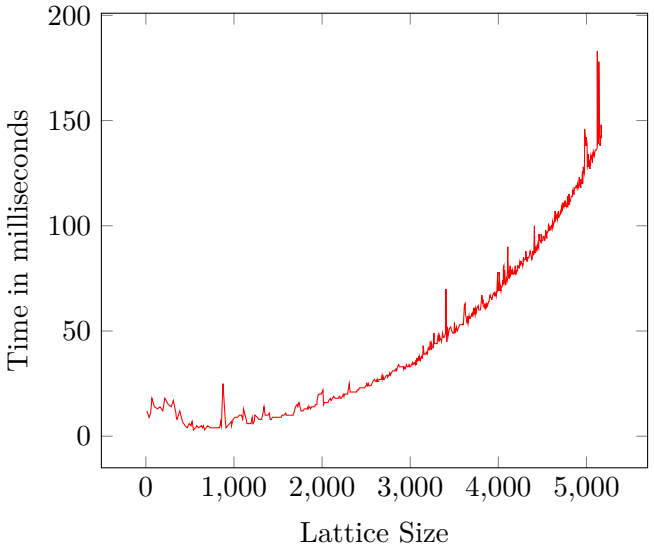


FIGURE 5.9: Lattice size [0-5000] vs. Construction time in Milliseconds

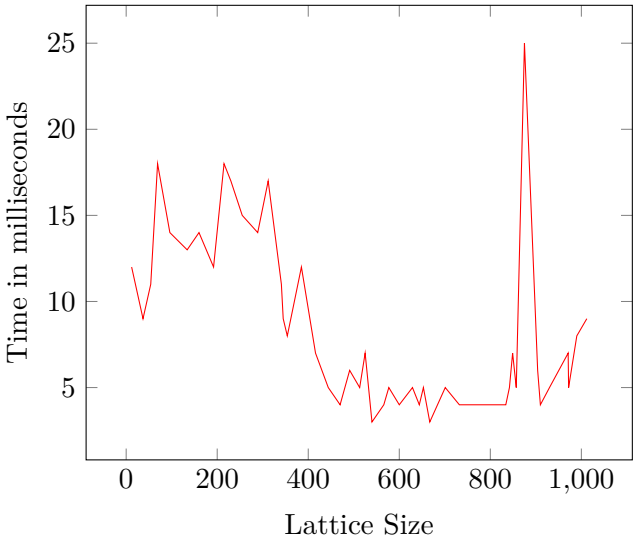


FIGURE 5.10: Lattice size [0-1000] vs. Construction time in Milliseconds

The focus of this work is on applying FCA in environments similar to INSIGHT @ NUIG where the number of concepts does not exceed 1000. In such settings, as we can see

from Figure 5.10, the maximum construction time can reach only 25 milliseconds. Even though the main criticism towards using FCA in this case is the fact of reconstructing the concept lattice for any change in the environment (e.g., a new sensor, change in a sensor attribute, etc.), this remains acceptable with such low construction time.

Comparing the efficiency of the approach with respect to the approaches analysed in Section 2.4 of Chapter 2, I refer to Table 5.5. Each line of this table recalls the approach used, the indexing mechanisms and the time performance as indicated by the authors in their papers. This table shows clearly that the proposed approach outperforms the others because it does not use any reasoning for indexing the set of input capabilities.

TABLE 5.5: Comparing Time Performances of Indexing Approaches

Indexing Mechanism	**Time Performance**
Inheritance between OWL-S services: Elenius et al. [71]	N/A
Topic extraction and Formal Concept Analysis: Aznag et al. [9]	size: 1088 services, query response time between 300 and 3000 ms
Reasoning-based matchmaking: Srinivasan et al. [211]	size: 50 services, index construction + advertisement time: $\sim$ 4 s
Numerical encoding of ontological concepts and codes comparison: Mokhtar et al. [153]	size: 100 services, index construction + advertisement time: $\sim$ 500 ms
Capabilities Indexing using Formal Concept Analysis	size: 1000 capabilities, index construction + parsing time: $\leq$ 25 ms

A major concern in using FCA is it application in a context of higher order of magnitude for the analysis and indexing of big numbers of business capabilities. FCA is known to be memory and compute heavy technique [5]. In small cases, such as the context of this thesis, the performance factor can be ignored as the computation time can be insignificant. However, in a big data context, this approach would completely fail because the time required to identify the concepts and creating the lattice may take several hours. Incremental concept lattice creation can help in this direction. Indeed, Godin et al. [87] propose another concept lattice creation algorithm that has the worst-case time complexity quadratic in the number of concepts. This algorithm is based on the use of an efficiently computable hash function f (which is actually the cardinality of an intent) defined on the set of concepts [87].

5.5 Conclusion

In this chapter, I used Formal Concept Analysis (FCA) for indexing a set of sensor services that are initially described using the conceptual model of Chapter 3. This chapter shows the applicability of this approach in indexing a set of 78 sensors used in a smart building energy management system. The object was to show that by using a set of capabilities (i.e., decribptions of services, sensors, business processes, etc.) one can

use existing tools for indexing and discovery. The use of FCA shows that even though the insertion of a new entry in the set of input capabilities requires the rebuilding of the entire indexing structure (i.e., concept lattice), the construction time remains less than a few milliseconds and consequently still possible to use with a relatively small repository (i.e., around 5000 entries).

The indexing technique used in this chapter does not differentiate between simple or composite capabilities. Indeed, simple and composite capabilities are modelled similarly and consequently the use of FCA remains possible. The main difference with composite capabilities is the use of multiple action categories, this property will be considered as any other attribute and requires dedicated scaling operations.

Chapter 6

Using Business Capabilities in the Design of Configurable Business Process Models

"I saw the angel in the marble and carved until I set him free."

Michelangelo

6.1 Introduction

This chapter, explores the idea of *early integration of capabilities in business process variants in order to create capability-annotated configurable business process models that can be tailored by manipulating capability properties.* In order to achieve this objective, this chapter proposes a novel algorithm for merging capability-annotated business process variants. The algorithm constitutes a contribution towards resolving the challenge of creating configurable business process models. Furthermore, the resulting configurable model will capture its configuration options in terms of differences between capabilities rather than model restrictions. This vision contributes also to challenge of configuration of configurable business process models as the configuration phase becomes the operation of manipulating capability properties, concepts that end-users are familiar with.

The remainder of this chapter is organized as follows: Section 6.2 analyses the research problem. Section 6.3 further describes the concept of configurable business process models and introduced the formal definition of a capability-annotated configurable business

process model. Section 6.4 introduces a running example that will be used in the rest of the chapter. Section 6.5 presents the merging algorithm. Section 6.6 reports on the implementation and validation of the algorithm. Section 6.7 analyzes the related work and Section 6.8 concludes this chapter.

6.2 Problem Analysis

A configurable business process model is a reference model that can be tailored by stakeholders in order to meet their requirements and satisfy their business needs [184]. The management of such models, brings two main challenges:

- **Creation** of configurable business process models: The basis of a configurable business process model is the integration of multiple behaviors of business processes into a single model. These behaviors are captured in *various* business process models that are called *business process variants* [184]. Configurable models can be created by *merging/aggregating* these variants [184]. Manual creation of configurable process models is tedious, time-consuming and error-prone task. It requires the identification of common process parts, merging them and explicitly representing differences between models in terms of configuration options. The literature provides several approaches to overcome this challenge [50, 91, 123], the main issue with such approaches is that the resulting configurable models capture their configuration options in terms of model restrictions that are difficult to manipulate by end-users during the configuration phase.
- **Configuration** of configurable business process models: this phase consists of enabling/disabling several branches of the model through manipulating configuration options. This phase is difficult and requires advanced modelling skills for identifying and selecting the configuration options. Furthermore, the users cannot determine the impact (i.e., what functionality are they enabling or disabling from the configurable model) of each configuration decision they take unless they manually trace each branch of the configurable node and determine the functionality resulting from each of them. This can be resolved by creating an explicit link between the model configurations and the domain requirements and lifting the configuration phase from manipulating model restrictions to domain requirements. La Rosa [116] proposed to model domain requirements as a set of questions with answers explicitly linked to configuration options. In this case, the configuration phase consists on answering these domain related questions. Even though this solution helps in guiding the configuration, it requires a lot of manual work for creating these questions and linking them to the model restrictions.

The contribution of this chapter is an algorithm that allows merging a pair of business capability-annotated process variants given as input and delivers a business capability-annotated configurable process model. Several methods have been proposed to merge business process variants such as [90, 115, 121, 123], their main weakness resides in the fact that they do not consider tasks capabilities for matching business process tasks. They rely exclusively on the task labels for this operation. In contrast to existing proposals, this chapter uses capabilities for matching similar tasks in different models. The resulting configurable model is also annotated with capabilities that can be used in the configuration and individualization steps [89, 116, 184].

In order to evaluate the merging algorithm proposed in this chapter, two main metrics are considered: **time** required for merging business process models and the **compression rate** gained after the merging operation. These two metrics have been used by La Rosa et al. [123] for evaluating their business process merging algorithm.

- **Time**: for organisations, time is important and should not be spent on manual creation of configurable models. La Rosa et al. [123] mentioned that it took a team of five analysts and 130 man-hour to merge *manually* 25% of an end-to-end process model. Therefore, an automation support for merging business process variants is needed to help saving time and money.
- **Compression rate**: the compression of a repository of business process variants into a single configurable model has multiple benefits: guaranteeing consistency between business process models, avoiding business process clones [67], etc.

This chapter evaluates also the proposed algorithm with respect to a set of requirements that have been used previously in the literature:

1. **[Behaviour Subsumption]** The merged model should allow for the behavior of all the original models. Traditionally, the merging operation is manually made by business analysts which comes with the risk that some aspects of the original models are accidentally neglected [90]. With automation support for merging process variants, this risk can be minimized considerably.
2. **[Traceability]** Each element of the merged process model should be easily traced back to its original model [121, 123]. A business analyst needs to understand what the process variants share, what are their differences, etc. This can be made possible if they can trace back to the variant from which an element originates.
3. **[Deriving Original Models]** Business analysts should be able to derive the input models from the merged process model [121, 123].

6.3 Capability-annotated Configurable Business Process Model

6.3.1 Configurable Business Process Model

Reference process models describe proven practices for a specific industry. They are often aligned with emerging industry-specific and cross-industry standards [28, 75]. One of the scenarios of use for reference process modelling is reference process model customization [113]. It begins with a reference process model that provides configuration facilities that can be configured to specific needs of an enterprise e.g., by refining business rules or enabling/disabling some activities. Such reference models are called configurable business process models [184].

Configurable process models are constructed via the aggregation of several variants of a process model [184]. Such models are considered for example when companies become the subject of acquisitions and mergers, in the case of improvement of existing business processes, or simply when different business analysts define their customized process models for achieving the same business goal. These business process models are called *business process variants*. Since these models achieve in essence the same business goal, the variants slightly differ from each other in their structure [133]. Therefore, managing the variants can be made easier by handling the common parts just once and not independently for each variant separately.

Figure 6.1 shows an example of three business process variants for organizing a trip using Event-driven Process Chain (EPC) [147, 148] which is one of the widely used business process modelling languages. The variation between these models is on the means of transport required. Indeed, the first variant (a) requires booking a flight, this applies when one needs to take a plane to reach the destination; the second (b) requires either taking a bus or a train as means of transport to reach the destination; and the last variant (c) does not require any means of transport, this can be the case when using the company's or own means of transport. All these variants share the same first function (i.e., Make Travel Request) and the last function (i.e., Book Hotel). In the context of configuration-based modeling, these common parts can be merged together while capturing explicit variation points among the other parts.

Business Process Management (BPM for short) [238] is an approach that focuses on the automation of business processes with the use of information technology [73] following a four phases lifecycle : modelling, implementation, execution and analysis (see Figure 6.2 (a)).

La Rosa [116] proposed an updated version of the classical BPM lifecycle where the modeling phase becomes split into two phases (see Figure 6.2 (b)). The *configurable*

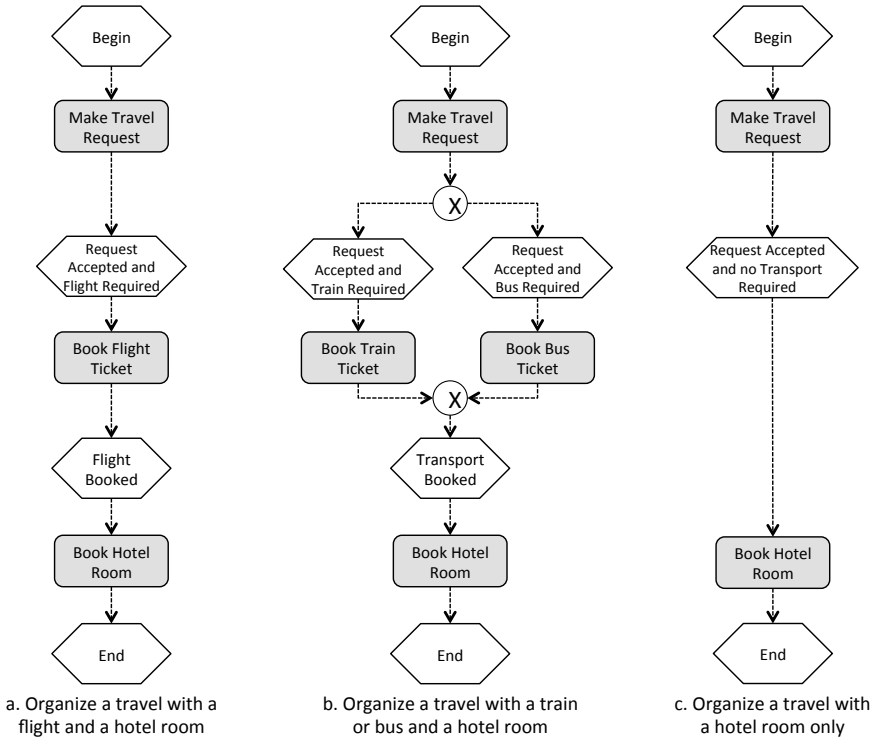


FIGURE 6.1: Three Business Process Variants for Organizing a Trip

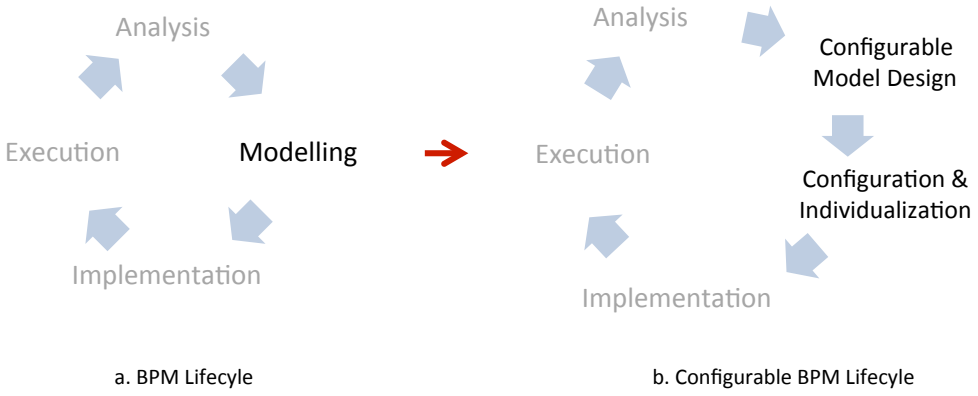


FIGURE 6.2: Configurable Business Process Lifecycle [116]

model design phase consists of creating the configurable model. This can be done either via mining techniques [91] or merging business process variants [50]. This chapter investigates and contributes to the second approach by proposing a novel algorithm for merging business process models. Figure 6.3 shows a configurable business process model created by merging the variants shown in Figure 6.1 using C-EPC notation [184]. In this C-EPCs, variation points are denoted by XOR connectors with thick borders. This is simply a visual indication to differentiate between regular connectors and configurable connectors. In fact, regular connectors represent choices that need to be made

during the execution of the business process, called run-time choices. However, configurable connectors represent choices that need to be made during the configuration of the configurable business process model, called configuration choices.

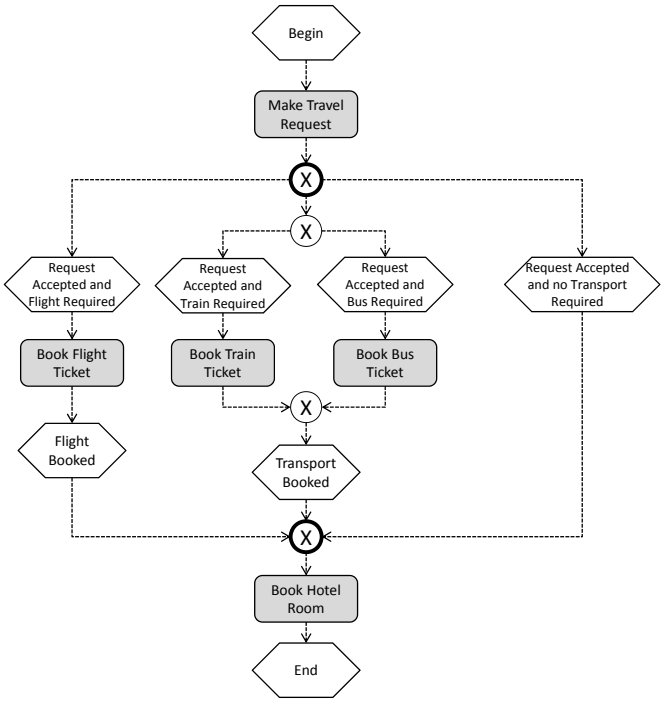


FIGURE 6.3: A Merged Configurable Business Process Model for Organizing a Trip

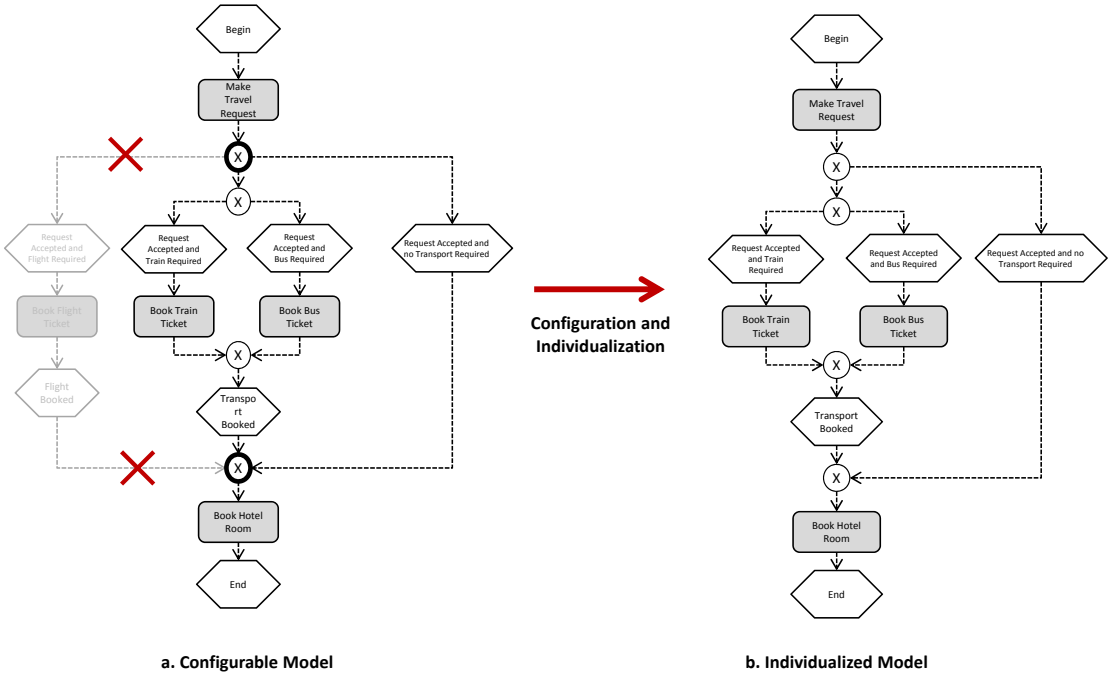


FIGURE 6.4: Removing the "Book Flight Ticket" Function from the Configurable Business Process Model for Organizing a Trip

During the *configuration and individualization* phase, several branches of the configurable model are enabled or disabled for removing undesired process behaviors. This can be done by removing/keeping the output or input arcs or the configurable connectors. The example shown in Figure 6.4 removes the "Book Flight Ticket" from the configurable model (a) in order to generate the individualized model (b). Note that the individualized model (b) did not exist previously in the set of input business process variants. The use of configurable business process models does not only allow the extraction of existing models but also the creation of new variants that did not exist previously.

The *implementation* and the *execution* phases result into a set of execution traces of business processes. Execution traces are captured in business process logs that can be later analysed during the *analysis* phase either for the continuous improvements of business process models [33], the generation of configurable process models [89] or mining configurable process fragments [6]. In Figure 6.5, execution traces are visualized as instances of the individualized model.

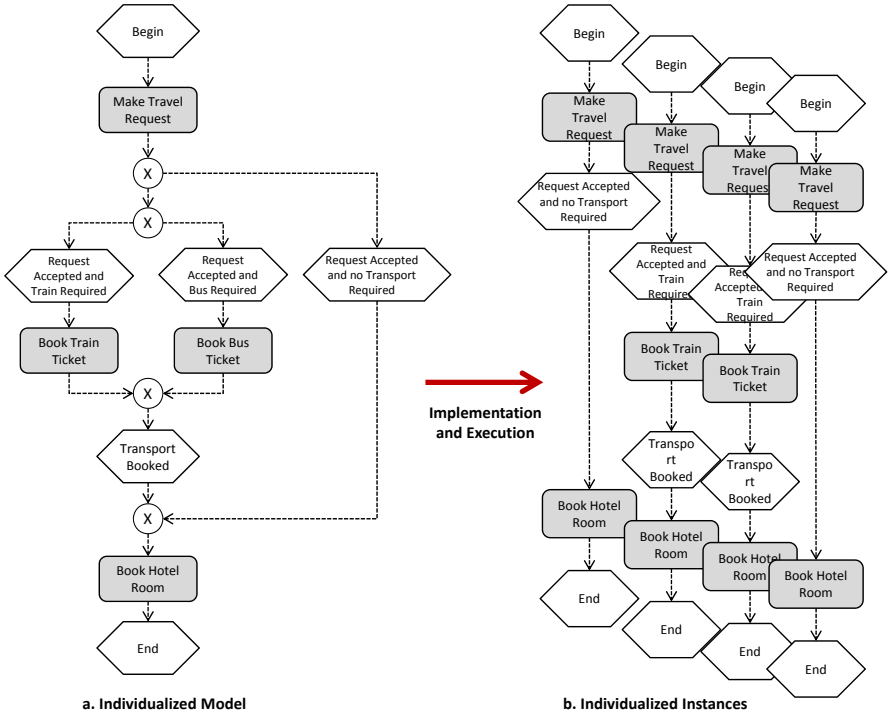


FIGURE 6.5: Instances of the Individualized Model are Generated during the Execution Phase.

6.3.2 Capability-annotated Configurable Business Process Model

This chapter uses EPC notation for illustrating basic business processes and C-EPCs [184, 233] for configurable business processes. C-EPC stands for Configurable EPC. It

is an extended version of EPC where some *connectors* can be marked as configurable. A configurable connector can be configured by reducing its incoming branches (in the case of a join) or its outgoing branches (in the case of a split) [121]. The result will be a regular connector with a reduced number of incoming or outgoing branches. Functions and events can also be configured by adjusting their labels. Additionally, functions can be set to enabled, skipped or conditionally skipped. In this chapter, I add another configuration dimension to function nodes based on their capabilities. The capability of a function can be configured by adding, removing or changing any of its properties with respect to the capability domain ontology.

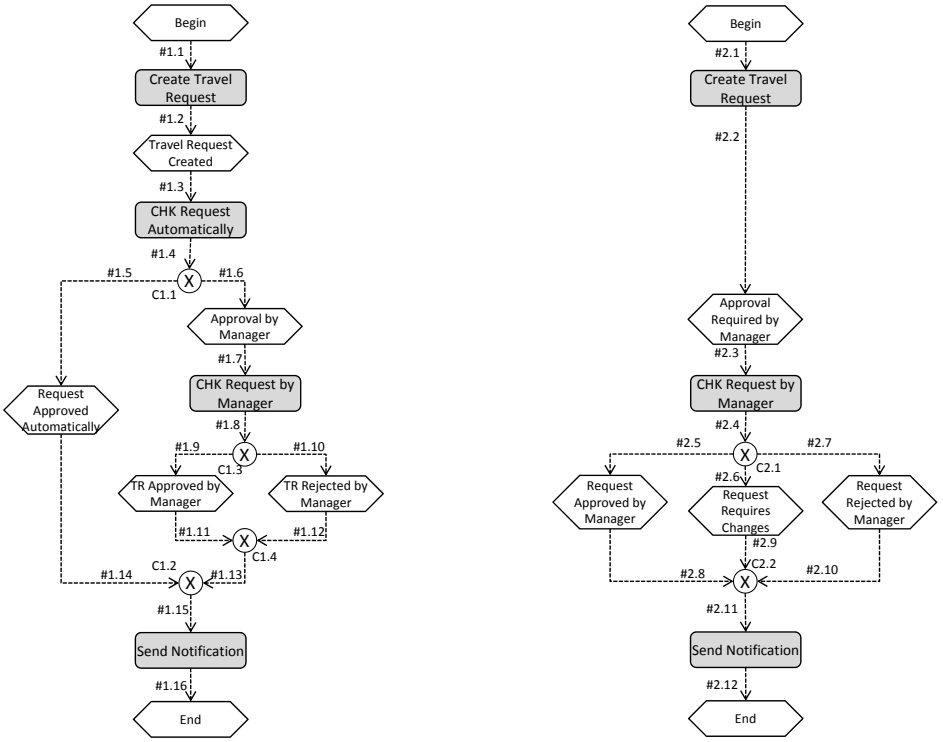
Recall, this chapter's contribution is a merging algorithm that takes as input a set of capability-annotated business process models and generates a capability-annotated configurable business process model. Input models are formally presented as directed graphs that were formally described in Definition 6 [Capability-Annotated Business Process Graph] (see Section 4.3 from Chapter 4). On top of this definition, Definition 13 formally describes a capability-annotated configurable business process graph that is used to formally describe the output of the proposed merging algorithm.

Definition 13 (Capability-Annotated Configurable Business Process Graph). A *Capability-Annotated Configurable Business Process Graph* is a directed graph G=<N, C, A, T, Cap, CN, CC, Tag>, where N, C, A, T and Cap are as specified in Definition 6: N is a set of work nodes that are both event and activity nodes; C is a set of graph connectors: i.e., ANDsplit, ANDjoin, ORsplit, ORjoin, XORsplit, and XORjoin; A is a set of directed arcs for interconnecting all the graph nodes; T is a type function, it associates to each node its respective type (i.e., a string to indicate: activity, event, XorSplit, etc.); and Cap is an annotation function that associates to each activity node n a tuple Cap(n) = (ActionCategory (n), Properties(n)).

$CN \subseteq N$ is the set of configurable nodes. $CC \subseteq C$ is the set of configurable connectors. Tag is an tagging function that associates for each item in N, C and A the identifier of the model it originated from.

The tagging function Tag is used in order to be able to trace back the origin of each item of the configurable business process graph (see Requirement **Traceability**). La Rosa et al. [123] use traceability also as a requirement during the creation of configurable process models. Knowing the origin of each item helps end-users, during the configuration phase, to know in what context (i.e., original model) a particular function/event has been used.

6.4 Running Example



(A) Automatically Approving Travel Requests (adapted from [195])

(B) Manually Approving Travel Requests (adapted from [196])

FIGURE 6.6: Two Business Process Variants from SAP Workflow Scenarios in Travel Management [197]

The running example depicted in Figure 6.6 presents two business process variants that follow the EPC notation [1]. These process models are taken from SAP Workflow Scenarios in Travel Management [197], they describe two travel request approval processes: automatically (see Figure 6.6a that is referred as *SAP_TR_A*) and manually (see Figure 6.6b that is referred as *SAP_TR_M*). These models involve four functions: "*Create Travel Request*", "*CHK Request Automatically*", "*CHK Request by Manager*" and "*Send Notification*".

- "*Create Travel Request*" consists of filling a form with the details of the travel request. This function appears in *SAP_TR_A* and *SAP_TR_M* and in both variants it is triggered with the same start event (i.e., "*Begin*").
- "*CHK Request Automatically*" is an automated process for approving a travel request with respect to the requested budget for the travel. This function appears only in *SAP_TR_A*. It is triggered by the event "*Travel Request Created*".

[1] Please note that, for presentation purposes, identifiers for arcs and connectors are added which is not part of the original EPC notation.

TABLE 6.1: Listing of the Nodes of SAP_TR_A with their Types and Business Capabilities

Node: *n*	**Type: *T(n)***	**Business Capability: *Cap(n)***
Begin	InitialNode	*not applicable*
Create Travel Request	Function	:CreateTravelRequest_Cap_A a cmm:Capability ; cmm:achieves bt:FillTravelRequestForm ; bt:name xsd:String ; bt:destination dbo:City ; bt:departureDate xsd:Date ; bt:returnDate xsd:Date ; bt:budget xsd:Double ; bt:purposeOfTravel xsd:String .
Travel Request Created	Event	*not applicable*
CHK Request Automatically	Function	:CHKRequestAutomatically_Cap_A a cmm:Capability ; cmm:achieves bt:CheckTravelRequestAutomatically ; bt:budgetLimit xsd:Double .
Request Approved Automatically	Event	*not applicable*
Approval by Manager	Event	*not applicable*
CHK Request by Manager	Function	:CHKRequestByManager_Cap_A a cmm:Capability ; cmm:achieves bt:CheckTravelRequestByManager ; bt:decision bt:accept, bt:reject.
TR Approved by Manager	Event	*not applicable*
TR Rejected by Manager	Event	*not applicable*
Send Notification	Function	:SendNotification_Cap_A a cmm:Capability ; cmm:achieves bt:SendNotification ; bt:notificationMessage xsd:String .
End	FinalNode	*not applicable*
C1.1	XORSplit	*not applicable*
C1.2	XORJoin	*not applicable*
C1.3	XORSplit	*not applicable*
C1.4	XORJoin	*not applicable*

- "*CHK Request by Manager*" asks the manager to decide about the travel request, it is triggered when the "*Approval by Manager*" is required. In *SAP_TR_A* the

TABLE 6.2: Listing of the Nodes of SAP_TR_M with their Types and Business Capabilities

Node	**Type**	**Capability**
Begin	InitialNode	*not applicable*
Create Travel Request	Function	:CreateTravelRequest_Cap_M cmm:achieves bt:FillTravelRequestForm ; bt:name xsd:String ; bt:destination dbo:City ; bt:departureDate xsd:Date ; bt:returnDate xsd:Date ; bt:budget xsd:Double ; bt:purposeOfTravel xsd:String .
Approval Required by Manager	Event	*not applicable*
CHK Request by Manager	Function	:CHKRequestByManager_Cap_M a cmm:Capability ; cmm:achieves bt:CheckTravelRequestByManager ; bt:decision bt:accept, bt:reject, bt:adjust.
Request Approved by Manager	Event	*not applicable*
Request Rejected by Manager	Event	*not applicable*
Request Requires Changes	Event	*not applicable*
Send Notification	Function	:SendNotification_Cap_M a cmm:Capability ; cmm:achieves bt:SendNotification ; bt:notificationMessage xsd:String ; bt:meansOfCommunication vcard:Email .
End	FinalNode	*not applicable*
C2.1	XORSplit	*not applicable*
C2.2	XORJoin	*not applicable*

manager can either approve or reject the travel request, this results into two respective events: "*TR Approved by Manager*" or "*TR Rejected by Manager*". In *SAP_TR_M* the manager can also ask for more clarifications or make changes to the travel request and this is shown via the event "TR Requires Changes".

- "*Send Notification*" consists of sending a notification to the requester. This function appears in *SAP_TR_A* and *SAP_TR_M* and in both variants it terminates the business process.

More formally, and with respect to Definition 6 introduced in Chapter 4, both business process models are defined as follows:

- $SAP\_TR\_A = < N_{SAP\_TR\_A}, C_{SAP\_TR\_A}, A_{SAP\_TR\_A}, Cap_{SAP\_TR\_A} >$, where $N_{SAP\_TR\_A}$, $C_{SAP\_TR\_A}$ and $Cap_{SAP\_TR\_A}$ are shown in Table 6.1 and $A_{SAP\_TR\_A} = \{\#1.n \ / \ n \in [0, 16]\}$ as shown in Figure 6.6a (e.g., #1.1 = (Begin,Create Travel Request) and #1.4 = (CHK Request Automatically, C1.1)).
- $SAP\_TR\_M = < N_{SAP\_TR\_M}, C_{SAP\_TR\_M}, A_{SAP\_TR\_M}, Cap_{SAP\_TR\_M} >$, where $N_{SAP\_TR\_M}$, $C_{SAP\_TR\_M}$ and $Cap_{SAP\_TR\_A}$ are shown in Table 6.2 and $A_{SAP\_TR\_M} = \{\#2.n \ / \ n \in [0, 12]\}$ as shown in Figure 6.6b (e.g., #2.1 = (Begin,Create Travel Request) and #2.4 = (CHK Request by Manager, C2.1)).

It is important to note that the original models [197] were incomplete and not well structured. They have been manually adapted to ensure that there are no deadlocks, dead-end paths, incomplete terminations, etc. [191]. Additionally these models were not annotated with any capability, the capabilities of each function item has been manually created using the capability meta-model introduced in Chapter 3.

6.5 Merging Business Capability-annotated Process Models: The Merging Algorithm

This section presents a novel algorithm for creating configurable business process models by merging pairs of business process variants. The input of this algorithm is a pair of *configurable business process models* and the output is a *configurable business process model*. If the input models are not configurable, it starts by transforming them into *configurable models* that mainly assures that the models' items are annotated with the identifier of the model they originate from in order to fulfill the traceability requirement [Traceability] (see Section 6.1).

The assumptions for this algorithm are as follows:

1. For both input models, every function item is annotated with its capability using the meta-model introduced in Chapter 3.
2. Both models are annotated with concepts from the same ontologies (i.e., same actions ontology and same capability domain ontology) and use the same language. In the absence of this requirement, an alignment of the used ontologies [193] or a cross-lingual comparison of business terms [164] is required.

3. Both models are well structured: there are no deadlocks, dead-end paths, incomplete terminations, etc. [191].

The Merging algorithm can be split into three steps:

1. *Merging both business processes' items*: first, match and merge each event and function item of a first model with its corresponding item of the second model. During this step, capabilities of functions are merged for creating configurable capabilities. This is followed by integrating the rest of the models' items (i.e., connectors and arcs) into the resulting model without any matching step. This step is detailed in Section 6.5.1.

2. *Post-processing the merged business process graph*: the previous step provides a business process graph that does not respect the modelling languages syntactic rules. The object of this step is to detect modelling problems and correct the resulting model. This step is detailed in Section 6.5.2.

3. *Reduction of the configurable business process graph*: when resolving syntactic problems, the Merging algorithm will create additional routing nodes that generate several connector chains. This step aims to reduce connector chains for a more compact configurable business process graph. Details are discussed in Section 6.5.3.

6.5.1 Merging Business Processes' Items

The Merging algorithm requires as input two configurable business process graphs. If the input models are not configurable, they need to be transformed to be compliant with Definition 13. Given the business process graphs depicted in Figure 6.6, this step needs to transform them into Configurable Business Process Graphs as per Definition 13. This is a trivial operation because for both models $CN = \emptyset$ and $CC = \emptyset$. The function Tag simply consists of tagging of each item in both models with their respective model's identifier. Both models become configurable models with single variants and without any configurable nodes.

6.5.1.1 Merging Events

The following step of the Merging algorithm consists of matching each *event* and *function* from both input models. The object of this operation is to identify similar events and functions in order to merge them into a single node. A straight forward solution to this

step can be carried out by imposing the use of common labels for events and functions in all variants. This is a valid assumption when all business process variants are created within the same modelling environment with a well defined organizational taxonomy that defines a controlled vocabulary to design business processes using the concepts of information entity, business process, organizational unit, actor, business schedule and business goal [167].

In the absence of an agreement on a common taxonomy in process design, modelers may agree on some terminology represented in a large corpus of text used within their business environment. Such corpus can be used to construct distributional models of meaning to generate semantic similarity and relatedness between the used terms. Semantic similarity between terms is based on the co-occurrence of terms in similar contexts in the corpus [79]. Two of the most powerful distributional semantic models are the Latent Semantic Analysis (LSA) [63] and the Explicit Semantic Analysis (ESA) [79]. In both approaches a large corpus of text documents is indexed to extract statistical properties about terms. Wikipedia is a good example of corpus initially used by ESA. Upon such indices, a semantic similarity/relatedness measure is operated. In such context, Freitas et al. [78] proposed an approximate query processing approach for databases based on distributional semantics and validated it within a natural query scenario over graph databases. In this chapter, I use an existing tool for computing matching scores between event labels that uses ESA-based over a domain independent corpus (Wikipedia).

Table 6.3, shows the scores of matching events from $G_{SAP\_TR\_A}$ to $G_{SAP\_TR\_M}$. Each cell represents the score that is computed as follows: the similarity between two event labels El_1 and El_2 is the average of the similarities between each pair of words (W_1, W_2) such that $W_1 \in EL_1$ and $W_2 \in EL_2$. For example, the matching score between "Approval by Manager" and "Approval Required by Manager" is 0.347. This score is computed after removing the stop-word "by" and computing the similarity between six possible pairs of words from these labels (e.g., Similarity("Approval", "Approval") = 1, Similarity("Approval", "Required") = 0.042, Similarity("Approval","Manager")=0.005) [2]. Note that for each line in Table 6.3, the two highest matching scores for each pair of event labels are highlighted. If for a line the score is 1, this means that the labels are identical, then obviously they merge together without further verifications, otherwise, the user can be asked to select the most appropriate label ordered with the highest score.

The matching score is used mainly for helping the user to select the best matching between events. Even though the matching score has been used to fully automate the merging of events in prior research [121, 123], I prefer giving the user the possibility to take the final decision. Indeed, in some cases, one can find a better matching of events

[2]Please note that scores are computed after stemming.

TABLE 6.3: ESA-based Matching Scores Matrix for Events from the Running Example of the two Business Process Models for Organizing a Trip (see Figure 6.6)

	Events from SAP_TR_M					
Events from SAP_TR_A	Begin	Approval Required by Manager	Request Approved by Manager	Request Requires Changes	Request Rejected	End
Begin	1	0.016	0.005	0.021	0.003	0.046
Travel Request Created	0.010	0.015	0.121	0.179	0.174	0.005
Request Approved Automatically	0.005	0.126	0.232	0.186	0.183	0.003
Approval by Manager	0.005	0.347	0.340	0.026	0.018	0.008
TR Approved by Manager	0.005	0.347	0.340	0.028	0.018	0.008
TR Rejected by Manager	0.004	0.181	0.179	0.018	0.259	0.010
End	0.046	0.007	0.006	0.005	0.004	1

according the score while the actual matching is different (see the matching score of "TR Approved by Manager" and "Approval Required by Manager" in Table 6.3). It is also possible to find high matching scores while there is no actual correct matching (see the matching score of "Request Approved Automatically" and "Request Approved by Manager" in Table 6.3).

After choosing the best matching between pairs of events, each pair of events is merged into a single one with the most appropriate label that the user can select. For traceability purposes, it is possible keep both labels in the merged event with additional tagging of the origin of each of them. However, this work keeps only one of the labels.

Figure 6.7 highlights the matching of events of the input models proposed in the running example in Figure 6.6. Each event from the first model share the same color with its corresponding event from the second model. Events kept in white do not have any corresponding event in the other model.

The matching operation used in this work can be further investigated and contribute to the currently growing body of research in the area of process model matching [4, 59, 60, 107, 108, 112, 130, 237]. Various solutions were introduced for matching process elements from different perspectives ranging from structural aspects analysis to the identification of activity labels.

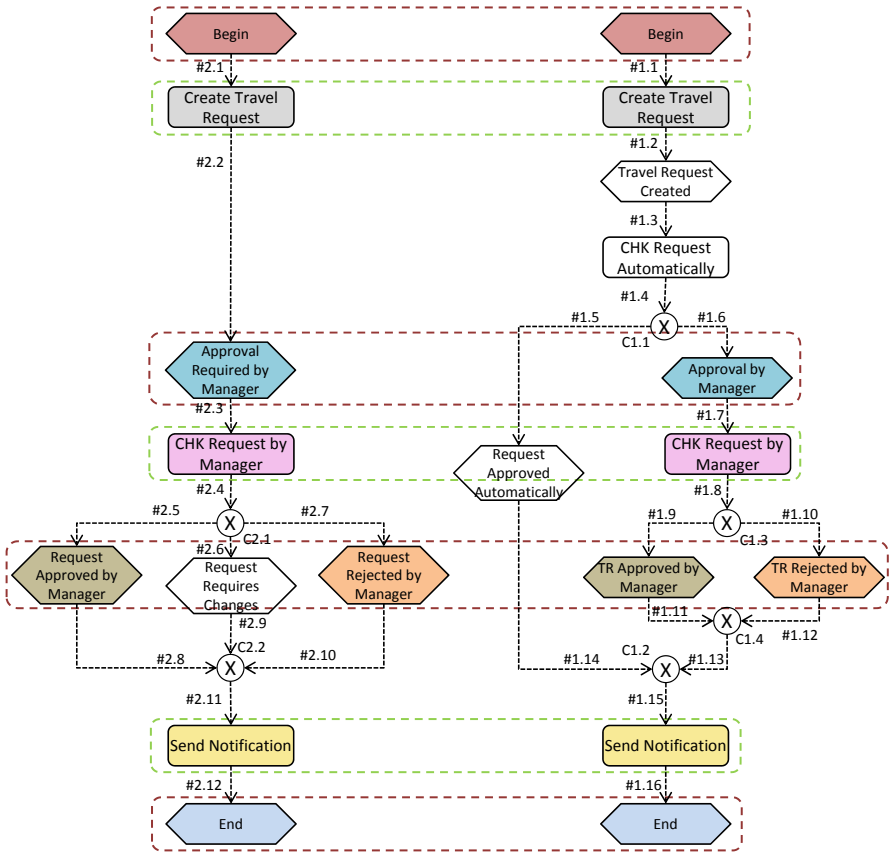


FIGURE 6.7: Matching Events and Functions from both Model Variants

6.5.1.2 Merging Functions

A primary assumption of the Merging algorithm imposes that both input models are annotated with their capabilities. This makes the matching of function items simpler than event items. Indeed, similar functions in essence achieve the same action and consequently should have the same *action category* of their capabilities. The matching of function items is simply done via comparing their action categories. However, the resulting merged function item should consider all differences between the capabilities from original models. Consequently, the merging of function items is a two-step operation that first identifies the corresponding items (see Figure 6.7 as example) then the second generates their merged capability.

For determining the resulting merged capability, the algorithm covers all possible cases:

- Both function items have the same capability: the resulting merged capability remain as it is (see Figure 6.8a).
- Both capabilities share the same property but with different values: the resulting merged capability is a **Configurable Capability** with a property that has a

Configurable Value (see Figure 6.8b). In all cases the configurable value is an **Enumeration** of both options originating from input capabilities.

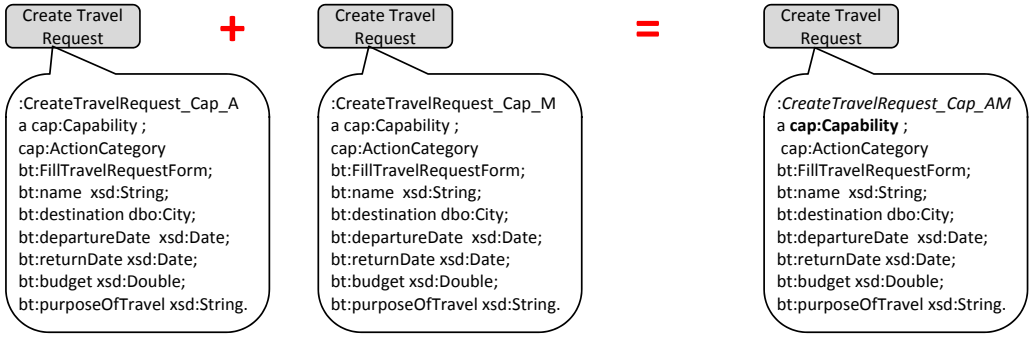
- One of the capabilities has one additional property: the merged capability is a **Configurable Capability** that has a **Configurable Property**: i.e., the additional property (see Figure 6.8c).

6.5.1.3 Merged Business Process Graph

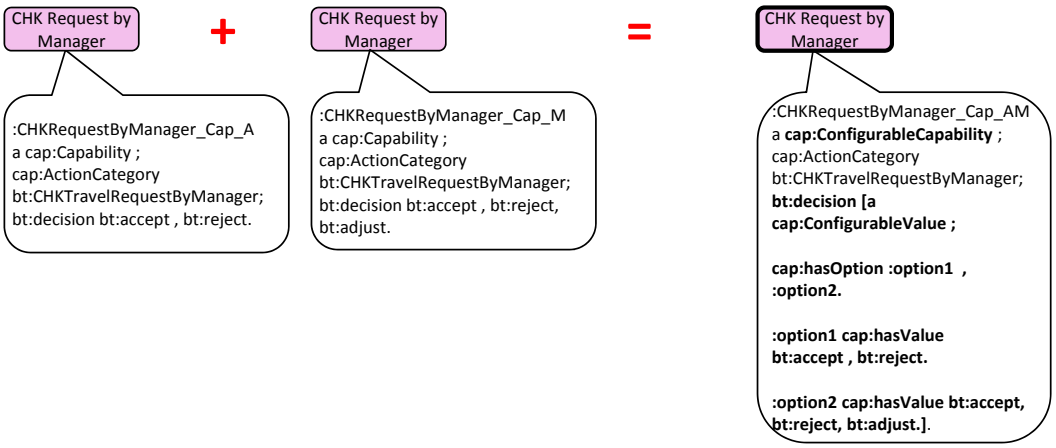
Note that the previous matching steps relate only to events and functions. Connectors from the first model can also be matched to connectors from the second one as it has been proposed by La Rosa et al. [121, 123]. The matching operation is done via a context similarity score by considering the connector neighborhood (i.e., incoming and outgoing nodes). This operation is not necessary and adds more complexity to the matching operation when connector chains appear in the model (i.e., various consecutive connectors).

The following step consists of creating the integrated configurable business process graph denoted $CG{=}< N_{CG}, C_{CG}, A_{CG}, T_{CG}, Cap_{CG}, CN_{CG}, CC_{CG}, CO_{CG}, Tag_{CG} >$ (see Definition 13). Let $G1{=}< N_{G1}, C_{G1}, A_{G1}, T_{G1}, Cap_{G1}, CN_{G1}, CC_{G1}, CO_{G1}, Tag_{G1} >$ and $G2$ be two input configurable business process graphs, the resulting $CG{=}< N_{G2}, C_{G2}, A_{G2}, T_{G2}, Cap_{G2}, CN_{G2}, CC_{G2}, CO_{G2}, Tag_{G2} >$ is constructed as follows:

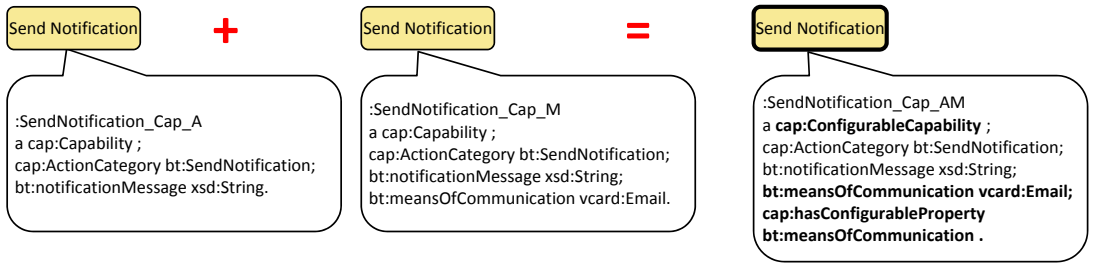
- $N_{CG} = N_{G1} \oplus N_{G2}$: the set of nodes of the configurable graph is the merger of nodes of both input models. More specifically, N_{CG} contains the results of merging events and functions as see in the Section 6.5.1.1 and 6.5.1.2 respectively, as well as the events and functions that are not common to $G1$ and $G2$.
- $C_{CG} = C_{G1} \cup C_{G2}$: the set of connectors of the configurable graph is the union of connectors of both input models without any merging.
- $A_{CG} = A_{G1} \cup A_{G2}$: the set of arcs of the configurable graph is the union of arcs from both models while considering updating sources and destinations of arcs with respect to the merged events and functions.
- T_{CG} and Cap_{CG} are two functions that report on the type of the node and the capabilities of the function items respectively.
- $CN_{CG} = CN_{G1} \cup CN_{G2} \cup CN_{G12}$: contains the list of function items that have configurable capabilities either originally defined in input models (i.e, $CN_{G1} \cup CN_{G2}$) or resulting from the merging of function items (i.e, CN_{G12}).



(A) Merging "Create Travel Request" Functions: Regular Capability



(B) Merging "CHK Request by Manager": Configurable Capability with Configurable Property Value



(C) Merging "Send Notification": Configurable Capability with Configurable Property

FIGURE 6.8: Merging Functions and Creating Configurable Capabilities

- $CC_{CG} = CC_{G1} \cup CC_{G2}$: contains the list of all configurable connectors from the original models. In the running example there are no configurable connectors, consequently at this stage $CC_{CG} = \{\}$.
- The tagging function Tag_{CG} will assign for each element of the graph the identifiers of the model they originated from (e.g., id(G1) and id(G2)).

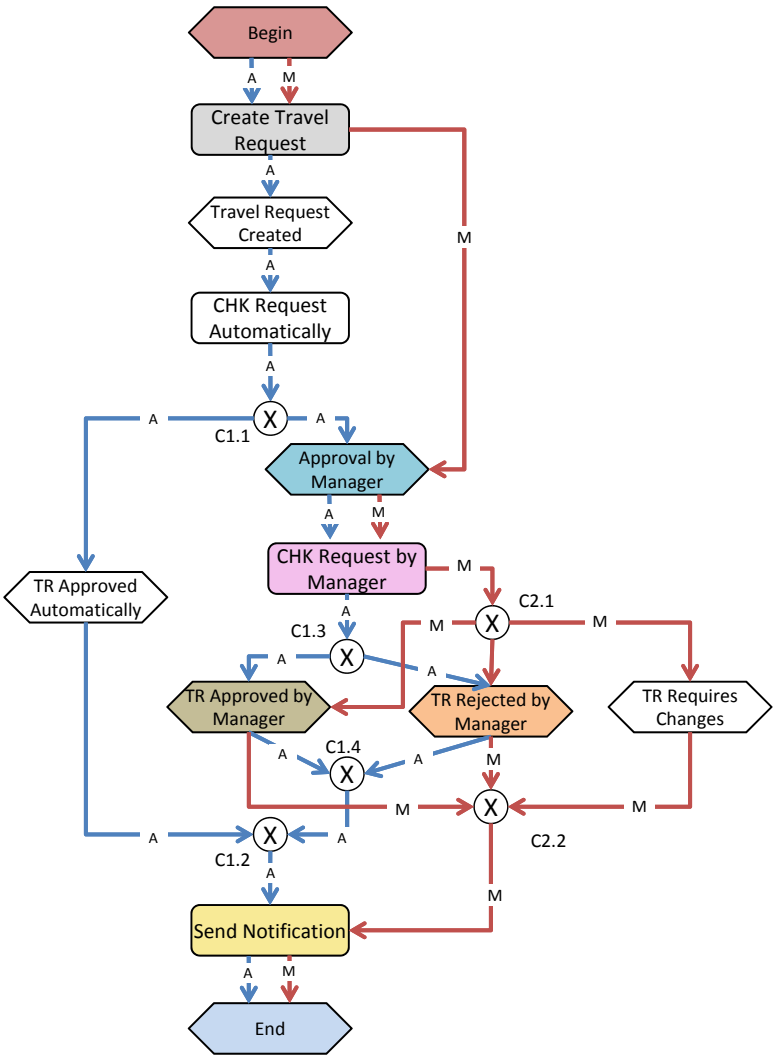


FIGURE 6.9: Merging Items from both Model Variants

Figure 6.9 illustrates the resulting configurable business process model after the merging operation. Only the tags on the arcs are shown in this Figure. One can easily notice that this model is not well structured: there are (1) duplicate arcs (e.g., from "Begin" and "Create Travel Request"), and (2) events/function nodes with multiple incoming/outgoing arcs. A post-processing step is required to resolve these issues.

6.5.2 Post-processing the Merged Business Process Graph

The resulting configurable graph CG needs some post-processing in order to be well structured and respect the set of requirements imposed by the modelling notation (EPC in this case) [147]. After the merging operation two requirements are violated. These requirements are:

1. for each $n \in N_{CG}$: $|\bullet n| \leq 1$. This requirement means that each event/function item must have at most one input.

2. for each $n \in N_{CG}$: $|n \bullet| \leq 1$. This requirement means that each event/function item must have at most one output.

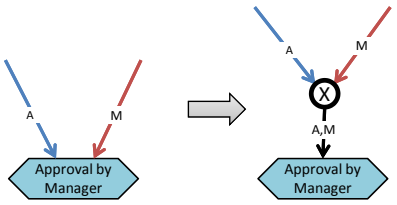
To ensure that each event/function item has a single entry, Algorithm 2 operates as follows: if a work node has more than one input, it creates a configurable connector (XOR-Join) that becomes the new destination of all input edges of that work node (i.e., lines 7 to 12). Finally, it creates a new edge from the new configurable connector to the work node that previously had more than one entry (i.e., lines 16 to 20).

Algorithm 2: Single Entry: Ensuring that Each Work Node has a Single Entry

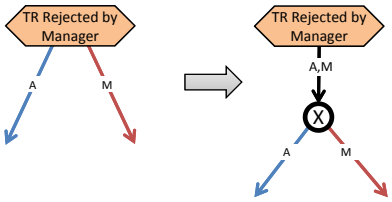
Input: Graph G: A graph that represents a configurable process graph.

```
1  begin
2      foreach n in N_G do
3          Tags ← {};
4          if |•n| > 1 then
5              CreateNewCXOR(CXOR);
6              CXOR.Type ← XORjoin;
7              foreach a ∈ A_G do
8                  if a.Destination == n then
9                      a.Destination ← CXOR;
10                     Tags.add(a.Tag);
11                 end
12             end
13             CXOR.Tag ← Tags;
14             C_G.add(CXOR);
15             CC_G.add(CXOR);
16             CreateNewArc(Arc);
17             Arc.Source ← CXOR;
18             Arc.Destination ← n;
19             Arc.Tag ← Tags;
20             A_G.add(Arc);
21         end
22     end
23 end
```

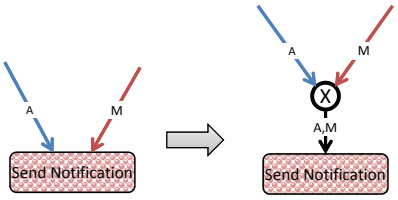
Figure 6.10a and 6.10c illustrate how this transformation is done. The left hand side of Figure 6.10a depicts two input arcs for the event *"Approval by Manager"* which has been changed in the right hand side of this figure by inserting a configurable XOR connector and an arc from this connector to *"Approval by manager"* that is tagged with all tags of original arcs (i.e., "'A,M").



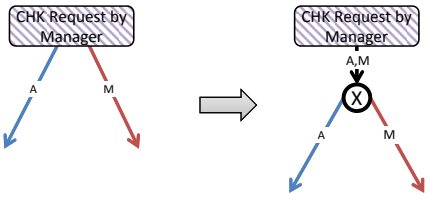
(A) Event node with Single Entry



(B) Event node with Single Entry



(C) Function node with Single Entry



(D) Function node with Single Entry

FIGURE 6.10: Introducing Configurable Connectors to make Work Nodes with a Single Entry and a single Exit

A similar algorithm operates to ensure that each event/function item has a single exit. Figure 6.10b and 6.10d illustrate how this transformation is done. The left hand side of Figure 6.10b depicts two output arcs for the event *"TR Rejected by Manager"* which has been changed in the right hand side of this figure by inserting a configurable XOR connector and an arc from *"TR Rejected by Manager"* to this connector that is tagged with all the tags of the original arcs (i.e., "'A,M").

At this level, the merged process model is completely constructed and Figure 6.11 depicts the resulting model. However, during this post-processing step, several configurable connectors have been inserted. This may lead to the appearance of several connector chains that make the model more complex to read. The following section shows how to reduce these connector chains in order to obtain a reduced configurable process model.

6.5.3 Reduction of the Configurable Business Process Graph

This section presents two rules that help reduce connector chains. These rules should reduce the business process graph while preserving its behaviour. Reducing a business process graph consists of deleting some routing nodes which are not mandatory. These reduction rules are: (i) merging consecutive split/join connectors and (ii) removing trivial connectors.

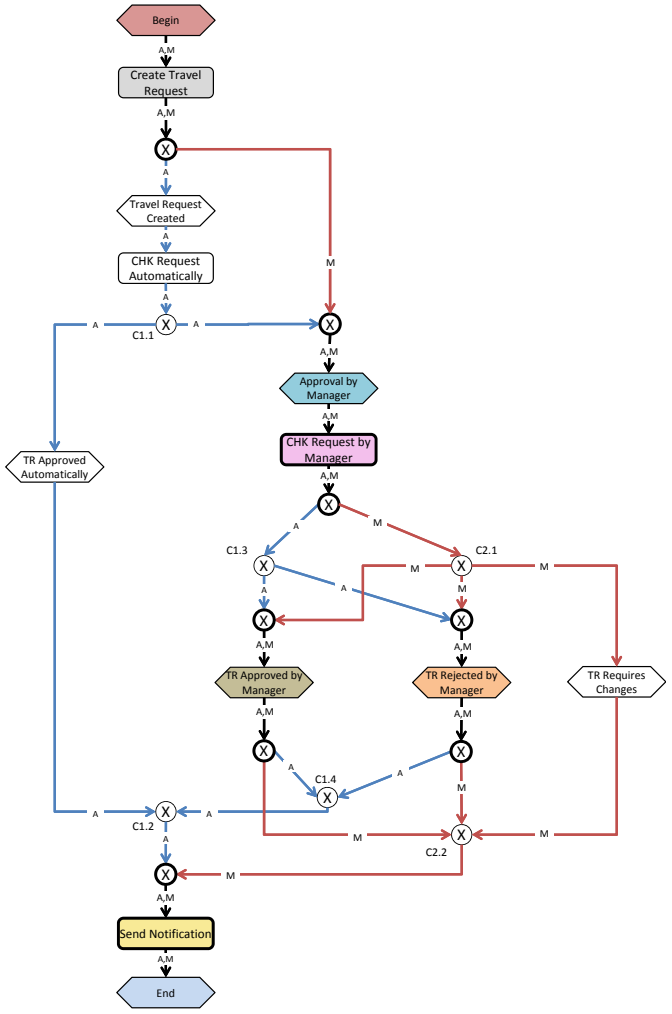


FIGURE 6.11: Correct Configurable Model after Post-processing Step

6.5.3.1 Merge Consecutive Split/Join Connectors

Algorithm 3 identifies and merges consecutive split/ join connectors into a single connector. It starts by parsing all the arcs of CG. If the source and the destination of an arc are two connectors of the same type (i.e., join or split) (i.e., line 2), then the algorithm fetches all the consecutive connectors having that type in order to create a set of connectors that have to be reduced (i.e., lines 8 to 15). The reduction of this set of connectors is made by creating a new configurable connector that replaces them. If one of the connectors that needs to be reduced is either an AND or an OR (i.e., line 17), then the new connector must be a configurable OR that keeps a trace back to the original operator with the identifier of the process where it originates from (i.e., line 22) otherwise it is a configurable XOR (i.e., line 3). Then, the algorithm continues to parse the remaining arcs to detect input/output arcs of the current connectors in order to link them to the new connector (i.e., lines 23 to 30). At the end of the algorithm, line 33 permits to merge arcs having the same source and destination.

Algorithm 3: Merge consecutive connectors

Input: Graph G: A graph that represents a configurable process model.

```
1  begin
2      foreach a in A_G do
3          Operator ← "XOR"; Tag ← ∅;
4          if a.Src ∈ C_G and a.Dest ∈ C_G then
5              if a.Src.Type == a.Dest.Type then
6                  ToReduce.addAll({a.Src,a.Dest}); Tag.addAll({a.Src.Tag, a.Dest.Tag};
7              end
8              foreach a2 in A_G do
9                  if a.Src ∈ ToReduce and a2.Dest ∉ ToReduce and
                   a2.Dest.Type == a.Src.Type then
10                     ToReduce.add (a2.Dest); Tag ← a2.Dest.Tag;
11                 end
12                 if a.Dest ∈ ToReduce and a2.Src ∉ ToReduce and
                   a2.Src.Type == a.Src.Type then
13                     ToReduce.add (a2.Src); Tag ← a2.Src.Tag;
14                 end
15             end
16             foreach a2 in A_G do
17                 if a2.Src ∈ ToReduce and (a2.Src.Op == "AND" or
                   a.Src.Op == "OR") or a2.Dest ∈ ToReduce and
                   (a2.Dest.Op == "AND" or a.Dest.Op == "OR") then
18                     Operator = "OR";
19                 end
20             end
21             CreateNewConfigurableConnector(CC);
22             CC.Type ← a.Src.Type; CC.Op ← Operator; CC.Tag ← Tag;
23             foreach a3 ∈ A_G do
24                 if a3.Src ∈ ToReduce then
25                     a3.Src ← CC;
26                 end
27                 if a3.Dest ∈ ToReduce then
28                     a3.Dest ← CC;
29                 end
30             end
31         end
32     end
33     MergeEdgesWithSameSourceSameDestination(G);
34 end
```

In Figure 6.12a (as it appears in this use case), connectors $CG1$, $C1.3$, and $C2.1$ are three consecutive split connectors which are merged into CG in Figure 6.12b. In Figure

6.12b there are two arcs from CG to $CG2$ with tags "A" and "M" which respectively originate from (Figure 6.12a) the arc from $CG1$ to $C1.3$ and the arc from $CG1$ to $C2.1$. These two arcs are merged into a single one with the tag "A,M" in Figure 6.12c.

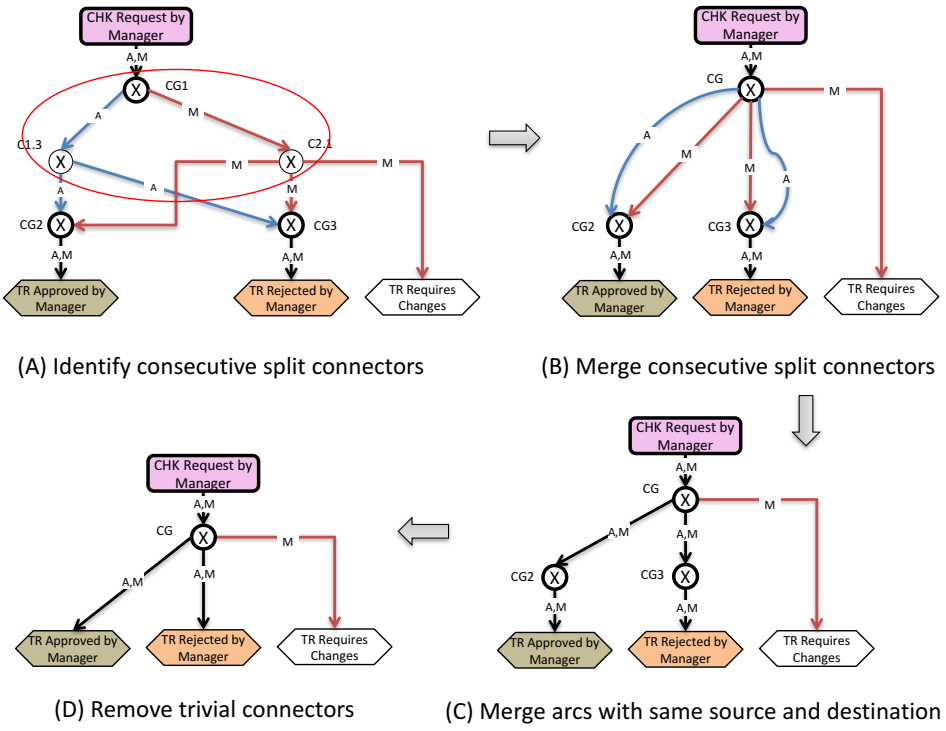


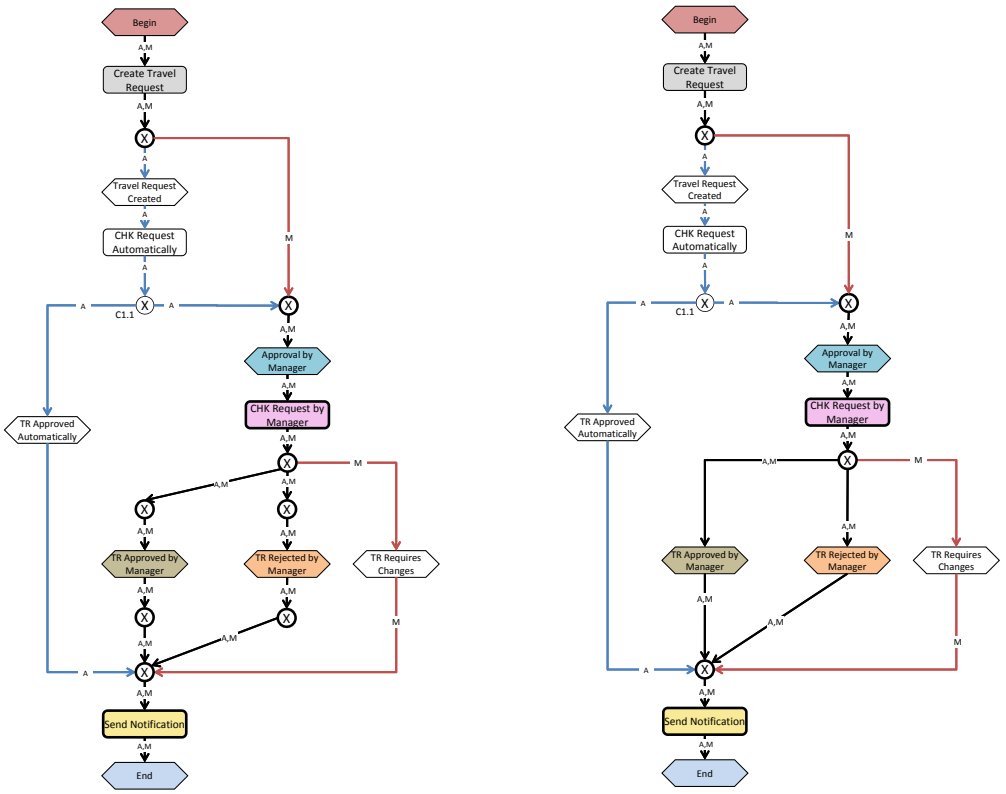
FIGURE 6.12: Reducing a Business Process Graph by Merging Consecutive Split Connectors

6.5.3.2 Remove Trivial Connectors

A trivial connector is a connector that has only one input and one output arc. It does not provide any useful routing information. Thus, it can be removed without altering the process behaviour. In the running example, Figure 6.12c depicts two trivial connectors (i.e., $CG2$ and $CG3$) that are removed in Figure 6.12d. Figure 6.13b depicts the resulting optimal configurable process model of the running example of this chapter.

6.5.4 Complexity Analysis

The proposed Merging algorithm in this chapter is linear depending on the size of the largest input business process graph. The first step of the algorithm consists of transforming both input models into two configurable models is a simple revisit to all the models' nodes and thus has the complexity of $O(|N|)$ where $|N|$ represents the number of work nodes of the largest input model. Without using the semantic similarity



(A) Configurable Model after Reducing Connector Chains

(B) Configurable Model after Removing Trivial Connectors

FIGURE 6.13: Resulting Configurable Models after Post-processing and Reduction steps

between event nodes, their merging starts by matching from both models the corresponding events which is bound to the number of nodes and thus has the complexity of $O(|N|^2)$. With respect to matching and merging function items, the matching operation is similar to the events matching and consequently has the complexity of $O(|N|^2)$; the merging step involves the merging of their corresponding capabilities that is bound to the number of properties p a capability can have which corresponds to the complexity of $O(|P|^2)$ where P is the maximum number of properties of a given capability. Consequently the complexity of the function items matching and merging is $O(|N * P|^2)$. The merging of arcs is bound to the number of arcs of the largest model which also corresponds to the number of nodes of the model and thus the complexity is $O(|N|)$. The post-processing steps has the complexity $O(|N|)$ as it is a simple loop over the nodes of the merged model while the reduction step has the complexity $O(|N|^2)$ as at each node, all neighbour are visited.

In the worst-case, the complexity of this merging algorithm is $O(|N * P|^2)$, where $|N|$ represents the number of nodes of the largest model and $|P|$ represents the number of properties of the largest capability used in the input models. This is the complexity

of the matching and merging of functions that dominates the complexity of the other steps.

6.6 Tool Support and Evaluation

6.6.1 Overview

As proof of concept, the proposed algorithm has been implemented as an extension of EPCTools [156], Section 6.6.2 reports on this extension . This tool has been used, in Section 6.6.3, to carry out further evaluations for measuring the compression rate gained by using this tool for merging a set of business process models and assess the required execution time. Furthermore, interviews with domain experts have been carried out in order to have a feedback on this contributions from practitioners in 6.6.4.

6.6.2 Tool Support

The presented business process merging algorithm has been implemented as an extension of EPCTools [156]. EPCTools is an open source initiative toward a tool for Event Driven Process Chains (EPCs) that supports the tool independent EPC interchange format EPML [148] implemented as an Eclipse Plug-in.

As shown in Figure 6.14, after opening one of the two process models, the user has to click on the "Merging models" button (see 1 in Figure 6.14), then a new dialog window is open, the user selects the second process model and clicks on ok, in this step the new configurable process is created. The user can optionally decide to apply the reduction step by selecting the "Reduce" button (see 2 in Figure 6.14).

The tool support is a proof of concept that has been implemented to carry out compression rate and execution time evaluations as well as to be used for interviews with domain experts. No further evaluations regarding the user interface and how the user interacts with this tool have been carried out. The user experience evaluation might be influenced by the modelling environment and is not relevant in the context of the contribution of this research.

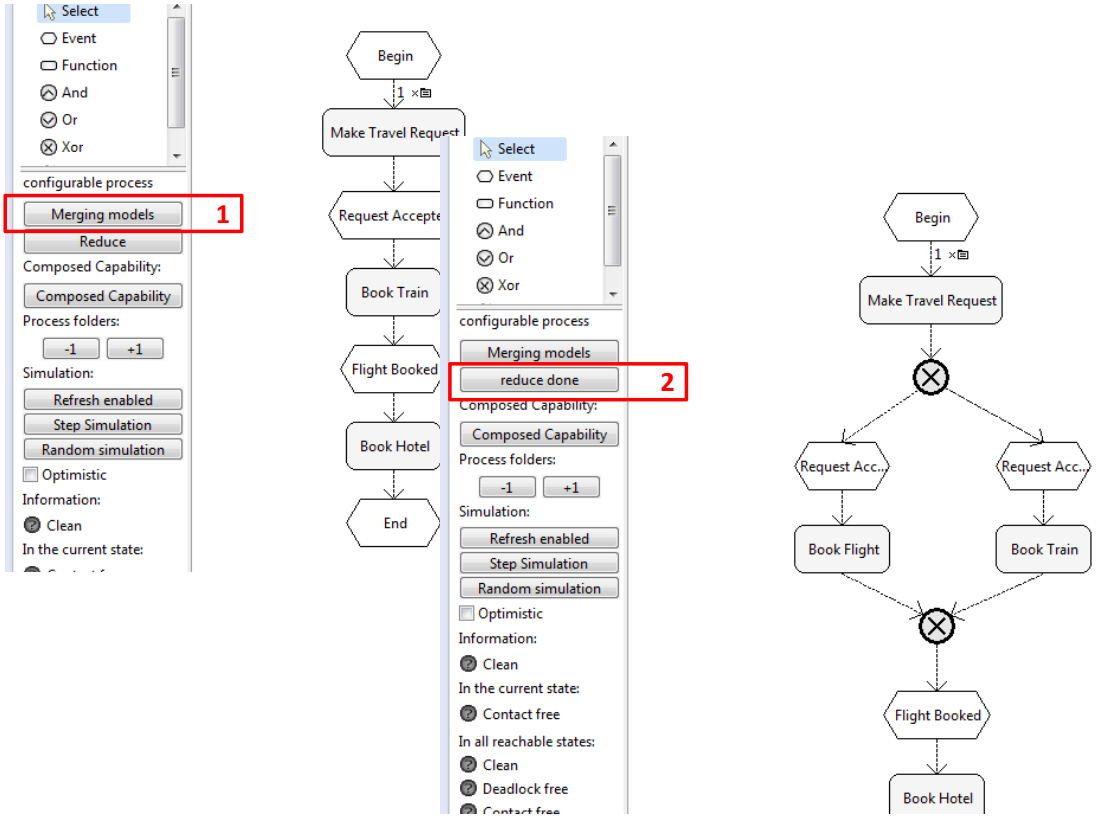


FIGURE 6.14: Extended Version of EPCTools that Supports the Creation of Capability-annotated Configurable Business Process Models

6.6.3 Compression Rate and Time Evaluation

6.6.3.1 Methodology

The objective of the compression rate evaluation is to highlight the benefit of merging business process variants into a single configurable business process model by avoiding duplicate process elements in process repositories. Furthermore, as for organisations time is important and should not be spent on manual creation of configurable models, this evaluation shows how quick the merging algorithm delivers configurable process models.

The evaluation of compression rate and execution time has been carried out as follows:

1. A test collection of real-world business process models have been manually created.
2. Each of the input models have been quantified in terms of number of process elements (i.e., events, functions and connectors).
3. Using the tool support, I have created configurable process models from the input models.

4. Each resulting configurable process model have been quantified in terms of number of process elements.

5. Measuring the compression rate by comparing the sizes of the input models and the output configurable model.

6. Measuring the execution time of the merging process.

Please note that the execution of the merging steps has not been interrupted with any manual task. In this regard the following actions have been taken:

- Event matching has been based on exact matching of events (in order to reduce manual input from te end-user).
- All the model variants are merged at once (instead of merging each pair one by one manually).
- The reduction step has been carried out automatically after merging (no manual decision regarding the reduction step).

6.6.3.2 Test Collection

I have manually created a test collection for evaluating the proposed merging algorithm. The process variants that I used in the experiment are those used by Gottschalk in his thesis [89]. They were subject of a case study [92] in which techniques for managing configurable process models were extensively tested in a real-world scenario. The process models used in this case study are four processes out of the five most executed registration processes in the civil affairs department of Dutch municipalities [89]:

- *P1: Acknowledging an unborn child*: This process is executed when a man wants to register that he is the father of an unborn child in case he is not married to his pregnant partner.
- *P2: Registering a newborn*: This process describes the steps for registering a newborn and get his birth certificate.
- *P3: Marriage*: This process describes all the steps required before getting married in a Dutch municipality.
- *P4: Decease*: This process describes the steps required by relatives to burry the deceased and get a death certificate.

The process variants considered in this evaluation are initially available in Protos[3]. Each process has five process variants. Consequently, a total of 5 x 4 = 20 process models were considered in this work (similar to the case study [92]). I have manually translated these models into EPC and used the extended version of EPCTools (see Section 6.6) for merging them in order to create configurable process models for each process.

6.6.3.3 Observations

During the merging steps, two metrics were observed: process models sizes (before, and after the merging) and the execution time of the merging steps. These metrics are shown in Table 6.4.

TABLE 6.4: Results of Merging Registration Processes of Dutch Municipalities

Process Number	**Input size (Number of Nodes)**	**Output size before reduction**	**Output size after reduction**	**Exec. Time (ms)**
P1	190 (29+56+52+29+24)	131 (31%)	71 (62%)	157
P2	347 (63+84+73+57+70)	276 (20%)	180 (48%)	235
P3	507 (76+127+127+114+63)	298 (41%)	214 (57%)	407
P4	355 (56+111+91+67+30)	266 (25%)	160 (54%)	282

Table 6.4 shows the size of the input and output models (size in terms of number of EPC nodes). Column one states the four processes considered here (P1: Acknowledging an unborn child, P2: Registering a newborn, P3: Marriage and P4: IDecease). Column two shows the size of the input models, entries of this column present the sum of the number of nodes of each variant as it is mentioned between parenthesis. Columns three and four show the size of the output models before and after the reduction step of the proposed algorithm which represent the size of the constructed configurable process model. The percentage value between parenthesis shows the compression rate gained from the creation of the configurable process models. Column five shows the execution time in milliseconds needed for merging the input process models.

6.6.3.4 Discussion

The reduction approach can gain around 50% in terms of space for storing several process variants. Besides this space gain, we can see that in a few milliseconds a set of five process

[3]Protos is part of Pallas Athena's BPM toolset BPM|one.

variants can be automatically merged which would take much longer for a business analyst to perform the task manually.

The compression rates are considerably higher after the reduction step. This step removes only connectors that are created for ensuring that events and functions have a single input and single output (see Section 6.5.2). In fact, generated connector chains can be reduced into a single connector without losing any routing information as per the reduction step discussed earlier in Section 6.5.3.

In general, compression rates are high because most of the process models share various process elements. Indeed, all the used variants, are from various Dutch municipalities that are initially defined from a high level reference model [92]. Depending on the population and the available resources of each municipality, few process tasks are either skipped or replaced by other ones. This keeps most of the process functions sequentially aligned. Consequently, the merged model observe a big number of common functions and events.

6.6.4 Interviews with Domain Experts

6.6.4.1 Introduction

In this part of the evaluation, I carried out a second round of semi-structured interviews with the five domain experts that I interviewed with regard to the business capability aggregation work of Chapter 4. These experts have strong background and are currently active in business process management activities. Their profiles include two information systems architects, one project manager, one IT engineer and one consultant and training expert. I target these four types of stakeholders as each of them has his own perspective and usage of business process models.

The objective of this evaluation is to assess the usefulness of the proposed merging algorithm for designing configurable process models integrating business capabilities of process activities. In this evaluation a design science methodology was followed [241] together with formal guidelines for conducting and reporting case study research [187].

The interviews were done after explanation of this thesis objective and details about configuration-based modelling. The main targeted outcome of these interviews is to identify if these experts see that the business capability-driven configuration of business process models is useful and can be adopted in their working environment.

6.6.4.2 Participants

This is a reminder about the profiles of the five participants that were recruited for this semi-structured interview. These five experts were from different levels of expertise in the area of service computing and information systems design and development. The age group of these participant is 30-50 years old and their professional background includes a minimum of 5 years experience and are currently active in their field. Their profiles include:

- two system architects (P1 and P2): working as designers of information systems for clients of a multinational company.
- one project manager (P3): leading teams of developers of information systems for the management of seaports in different countries.
- and one IT engineer (4): working as developer in start-up offering automated post services.
- one information system consultant (P5): working as a consultant and trainer in the area of business process management.

6.6.4.3 Approach

The approach used for this evluation follows the case study research process proposed by Runeson and Höst [187]:

- *Case study design*: the objective of the evaluation is to assess the usefulness of the proposed merging algorithm for designing configurable process models integrating business capabilities. Interviews run individually using online conferencing tool. Each interview took about 1 hour for each participant.
- *Preparation for data collection*: The discussions were semi-structured to give the participants the freedom to give additional comments and get as much feedback as possible from them. The structure of the interviews was as follows [4]:
    1. 5 minutes discussion about the profile of the participant and his knowledge about reference process modelling and particularly configuration-based business process modelling.

[4]Please note that the durations used here are approximative. Some of the interviews run for few minutes more or less for each section.

2. 15 minutes presentation of the business capability-driven design and configuration of configurable business process models.
3. 5 minutes for manually merging a pairs of small business process models. The models used are the two variants of the process of organising a trip depicted in Figure 6.15
4. 10 minutes for manually merging a pairs of larger business process models. The models used are the two variants of the process of importation from the customs clearance procedures as this was their domain of expertise.
5. 10 minutes demo and interaction with the tool support.
6. 15 minutes discussion about the contribution of this chapter.

- *Analysis of collected data*: A post interview analysis of the collected feedback is reported in Section 6.6.4.4.
- *Reporting*: A discussion of the resulting feedback is summarised in Section 6.6.4.5 and shared among the participants.

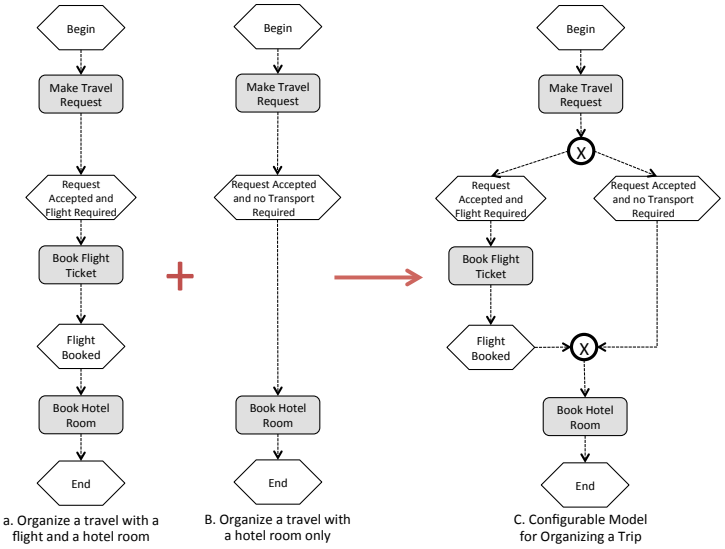


FIGURE 6.15: A Configurable Process Model for Organizing a Trip Created by Merging two Simple Variants

6.6.4.4 Results

Key results from these interviews are as follows:

General comments on the experience of the experts in using reference process modeling approaches

- All of the experts except the IT engineer (P4) are aware of reference process modelling in general and configuration-based modelling in particular.

  "*Of course we are familiar with reference process modelling. A lot of our missions consist of configuring our system to clients needs.*" (P5)

- Most of the configuration tools they used were focused on IT configurations, as stated by P1, P3 and P5.

- The only non-IT related configuration option that P5 encountered was the role assignment for tasks. For example, a particular task can be achieved by project managers and can be configured to other roles such as budget holder, director, etc.

Feedback on the business process merging approach

- The manual creation of pairs of small process models by all the experts was quick for small models and done within the 5 minutes slot given for this task

- The manual creation of pairs of larger process models was not complete within the 10 minutes slots given for this task.

Feedback on the use of the business capabilities in configurable process models

- The use of a single business capability ontology to annotate business process variants was pointed as weak point of this research by P5. Using multiple ontologies is more likely a common practice for these experts.

Feedback on the use of the tool support

- Using the tool support to merge the models that they have manually created was highly accepted.

- Fully automated merging is not always useful (P2 and P3), it is better to consider human intervention to validate some merging decisions.

  "*The tool is useful for guaranteeing a rapid merging of models. However, it is good to give the user the possibility to take some of the merging decisions.*" (P3)

- Manual changes of the resulting configurable model are also pointed as needed by P2, P3 and P4.

6.6.4.5 Discussion

Most of the current solutions that these experts use are configuration-driven but not from a business capability perspective. Configurations consist of setting the communication protocols, the form fields that need to be available in the process tasks, the monitoring indicators, etc. This confirms the fact that current information systems are mainly focused on the IT engineering part of business processes and not targeting other aspects such as the business capability. The consultant and training expert (P5) finds that this is a major issue with customers that adopt for the first time their information system. For this reason an extensive training period is required in order to make business experts more familiar with the vocabulary used by the solution provided. These solutions are not flexible enough to integrate changes that customers want to integrate. It is the customer that has to adapt his work to the solution rather than the other way round.

All the interviewed experts positively perceive the usefulness of the configuration-base modelling approach in order to design business process models, however, they see that the major problem is how to create these models and make them easy to manage by end-users. After showing the proposed solutions of this research, the experts agree that this is one possible solution but remains limited in terms of using business capability-annotations using a unique ontology. This work can be further extended to include multiple ontologies for annotating business process models.

The automatic merging was a valuable addition from these experts point of view. Some of them (P1, P2 and P5) had to create manual merging of business process models in other context and they recognise that manual creation is time consuming and does not necessarily guarantee a correct result. Those that did not experience this in their working environment (P3 and P4) also adhere to this point views after trying the manual merge of the large models during the interview.

Testing the tool reveals that it is simple to use but very EPC focused. Applying the same approach to other languages can be a good addition.

The experts pointed the need to have human interventions during the merging operation. Validating the merging of functions or events for some cases might need the expert's decision. Even after the completion of the merging process, some manual changes can be required for more flexible and tailored configurable model. Indeed, the creation of configurable process models is not exclusively done by merging process variants, other techniques such as mining process logs can be used [7, 91]. It is important to point that the resulting configurable model after merging the process variants can be tailored to include other configuration options and execution paths. A future direction to look

at, in the context of this research, is a hybrid solution that uses both process merging techniques together with process mining for the design of configurable process models.

The major challenge for business capability-driven configuration modelling to join industry is the fact that current industrial solutions are mature enough and hard to replace. Current solutions have been built over years of analysis, engineering and research that are proven to be effective. Replacing these solutions has never been taken as a serious option. However, features such as those proposed in this thesis (i.e., process annotations with capabilities and configuration of processes using their business capabilities) can be seen as additional options to the current systems but a lot of adaptation work is required.

6.7 Comparative Analysis with Related Approaches

6.7.1 Units of Analysis

This evaluation aims to compare the merging algorithm proposed in this chapter with other related business process merging approaches. As units of Analysis, I am using the requirements introduced in Section 6.2. The following is a recall of these requirements:

1. **[Behaviour Subsumption]** The merged model should allow for the behavior of all the original models [90].
2. **[Traceability]** Each element of the merged process model should be easily traced back to its original model [121, 123].
3. **[Deriving Original Models]** Business analysts should be able to derive the input models (as well as new ones) from the merged process model [121, 123].

The comparative analysis carried out in this Section focuses exclusively on business process modelling approaches. Other contributions for merging multiple perspectives of process models [149] or merging database schemas [175] and Object and Class Diagrams [162] are not considered in this analysis.

6.7.2 Related Approaches

Process Merging for Version Control

In a collaborative business process modelling environment, multiple stakeholders can be involved in the design of business process models. Starting from a common version of

a certain model, different users can adapt it to meet their needs resulting into multiple versions of the same model. At some point, when each of the stakeholders want to commit his version to a common repository, each of the new versions needs to be merged in order to generate a new common version of a certain model. This is the research context that motivates the work carried out by Gerth and Luckey [84]. In their previous works [85, 86, 114, 115], the authors propose a formalism to detect equivalent business process models based on the detection of equivalent fragments contained in these models. The objective it to detect and resolve version conflicts during the merging of process variants. Authors here refer to their existing tool support for model merging in IBM WebSphere Business Modeler [115]. The merge procedure defined is not intended to be fully automated, it is rather developed for reducing the number of false-positive differences and conflicts in models management. The resulting model is obtained after selecting a set of change operations and applying them on the current model. This new model is called the merged model that becomes the new common version of the process model.

Critique

Authors did not give attention to the first requirement of behaviour subsumption. Indeed, the resulting model can exclude the behaviours of input models in the resulting model after a stakeholder validation. Given the motivation of this work, i.e., consolidating multiple process version into a single common reference model, this approach does not satisfy neither the second requirement of keeping track on the origin of the element of the reference model nor the third requirement of derining input mdoels from the merged one.

Merging Event Driven Process Chains

Gottschalk et al. [90] define an approach exclusively intended for merging models following the EPC notation. This approach consists first of transforming EPCs into a so called abstraction of EPCs, namely function graphs. The second step is the combination of these function graphs by means of set union operations. Finally, they transform back the combined function graph into an EPC. The object in their approach is not to create a configurable EPC, there are no configurable connectors introduced which would allow for extracting one of the original models.

Critique

Gottschalk et al. [90] use behaviour preserving set union operations over function graphs in order to satisfy the first requirement of behaviours subsumption. However, this approach does not allow for the second (i.e., Traceability) or the third (i.e., Deriving Original Models) requirements introduced in Section 6.7.1. Indeed, the generated merged models do not allow to trace back where an element of the model originates from. As well as they do not provide any possibility to configure the obtained model in order to derive one of the input models.

Merging Process Graphs to create Configurable Models

La Rosa et al. [121, 123] propose a technique that satisfies the three requirements. Their technique starts by computing a similarity measure between nodes of pairs of process models. Then, given a mapping between different elements of the original models, they propose a merge operator that computes the Maximum Common Regions (MCR) and then links elements of the second models, which are not in the MCR, to the MCR of the first model. Similar to the approach presented, they use arc annotations to allow for tracking the origin of an element.

Critique

In this work, the mapping between function items exclusively relies on their labels using approximate semantic matching between them, while the approach presented in this chapter considers capabilities and domain ontologies. Furthermore, the proposed algorithm has a complexity of $O(|N|^3)$ for merging only one pair of process models [123] where $|N|$ is the number of nodes of the largest model. However, the merging algorithm proposed in this chapter has a reduced complexity of $O(|N * P|^2)$, where $|N|$ represents the number of nodes of the largest model and $|P|$ represents the number of properties of the largest capability that is much less than the size of an average process model.

Summary

A summary of the discussed approaches is show in Table 6.5. The first three lines of this table list the details of previously reviewed approaches while the last line concerns the proposed Merging Algorithm of this chapter. The proposed Merging Algorithm satisfies the three pre-mentioned requirements as follows:

1. **[Behaviour Subsumption]** All the operations of the merging algorithm do not remove any work node (i.e., events and function items). Furthermore, the order between these work nodes is preserved along the merging steps. The only removal is carried out during the reduction step when removing trivial connectors. Trivial connectors do not introduce any routing information and consequently they can be removed without altering the behaviour of the model.

2. **[Traceability]** The initial step of the algorithm starts by tagging each element of the input process models with the identifier of their corresponding model. This is maintained via the function Tag that returns the identifier of the model where an item originates from.

3. **[Deriving Original Models]** The main target of the merging model is to provide models that can be tailored for generating either input model or new ones. The concept of configurable models fulfils this requirement and more precisely via the use of configurable connectors and function items. A possible way to assist users in deriving one of the input models is to keep from the merged model only items tagged with the original model that the user wants to extract.

TABLE 6.5: Comparative Analysis of Business Process Merging Approaches

Approach	Behavous Supsumption	Traceability	Deriving Original Models
Process Merging for Version Control [84–86, 114, 115]	(-) The merged model does not necessarily include the behaviours of all input models.	(-) The output model allows to roll back to the immediate previous version of the model and not to other ones.	(-) The output model is not configurable.
Merging Event Driven Process Chains [90]	(+) The combination of function graphs does not alter the bahaviour on input models.	(-) The is not traceability to to any of the input models.	(-) The output model is not configurable.
Merging Process Graphs to create Configurable Models [121, 123]	(+) The fusion of maximum common regions and applied reduction operations are behaviour preserving.	(+) The process arcs are tagged with input models identifiers.	(+) The generated model is configurable and allows to generate either original or new models.
The Merging Algorithm	(+) All the operations of the merging algorithm are are behaviour preserving.	(+) All the process elements are tagged with input models identifiers.	(+) The generated model is configurable and allows to generate either original or new models.

6.7.3 Discussion

From these related approaches, we can distinguish other units of analysis that help further distinguish the merging algorithm of this chapter:

- *[Target modelling language]* All of the reviewed approaches including the proposed merging algorithm of this chapter except [90] make use of abstraction into business

process graphs so that minor changes can be applied to the proposed approaches to be applicable on other modelling languages.

- *[Matching of process elements]* Gerth et al. [84–86, 114, 115] focus on matching change operations using exact matching of labels of model elements. Gottschalk et al. [90] relies on exact matching of process elements. Only La Rosa et al. [121, 123] and the proposed merging algorithm use approximate semantic matching of labels of events. Furthermore, the proposed merging algorithm uses ontologies and capabilities for matching functions of process models.

- *[Complexity]* It is only discussed by La Rosa et al. [123], they mention that their algorithm has a complexity of $O(|N|^3)$ for merging only one pair of process models where $|N|$ is the number of nodes of the largest model. The algorithm proposed in this chapter has a complexity of $O(|N * P|^2)$, where $|N|$ represents the number of nodes of the largest model and $|P|$ represents the number of properties of the largest capability used in the input models.

6.8 Summary

This chapter discussed the early integration of capabilities in business process models in order to use them to create capability-annotated configurable business process models. The chapter proposes an algorithm that takes as input a set of capability-annotated business process models and outputs the corresponding capability-annotated configurable model. The resulting model should subsume the behaviour of the input models and allow to derive either of them or other new models. This feature is fulfilled by the concept of a configurable model that has been extended in this chapter with configurable capabilities. The resulting capabilities can be used at a later step for driving the configuration of such models.

The Merging algorithm has been evaluated using real world business process variants that have been manually created and annotated. Two main dimensions were considered for the evaluation: time and compression rate. These two metrics were used by previous researchers to evaluate similar contributions. Results show that the Merging algorithm insures a compression rate of 50% within a short execution time in the order of a few milliseconds.

Furthermore, capability descriptions in configurable process models can also be seen as a rich (and explicit) description of business processes by their properties and more specifically enriched properties at the structural configuration options. This could be considered a significant step forward and certainly supports the configuration step, i.e.

one can achieve incremental improvement of business processes and continued adaptation to demands.

Chapter 7

Conclusions

"The important thing is not to stop questioning."

Albert Einstein

7.1 Thesis Summary

With the advent of Industry 4.0, more and more companies are actively working on digitising their assets (i.e., services, processes, etc.) for better control, collaboration, modularity, analysis, etc. By 2020 more than 80% of companies will have digitised their business processes and value chains. This creates more services and processes, making their indexing, discovery, configuration, etc. more challenging. Thus, properly digitising those assets needs a proper data model to describe them, and proper algorithms for indexing, discovering and configuring them. In this context, this thesis proposes a concept model for describing the business capability of services and business processes from a functional perspective in terms of what do they achieve together with related business properties. Furthermore, this work proposes the aggregation, indexing, discovery and configuration of services and business processes using the concept of business capability.

Therefore, the first objective of this thesis was to propose a conceptual model for describing Business Capabilities. The proposed model is implemented as a set of ontologies that can be used for creating semantic annotations of business process models or services. In this work, I consider a business capability as standalone entity that can exist outside the scope of service descriptions or invocation interfaces. A service, a computer program, a business process or even a manual task can be described using the business capability concept where an explicit link can be created between them. In a very simple definition,

I consider a business capability as a set of actions enriched with zero or many properties. Properties allow to refine further the action that is taken for a domain related ontology.

The model was validated via Bunge's theory of ontology [30] and via interviews with domain experts.

The modelling of capabilities as a set of actions and properties is inspired by the frame-based modelling approaches that have been proven to be effective in practice with languages such as JSON. It is simple, relies mainly on a shared agreements on the semantics of the used actions and properties that are defined in common ontologies. The other advantage of using frame-based modelling is the possibility of indexing, searching and aggregating capabilities without heavily relying on reasoning which is the major issue with current approaches. Detailed analysis of current modelling approaches was carried out Section 2.2 of Chapter 2 and the details of the capability meta-model were discussed in Chapter 3.

The second objective of this thesis was to propose an abstraction technique that allows moving from an entire process model to its functional description by aggregating the business capabilities of the process elements into a single one. This feature is necessary when a user wants to have an overview of the capability of an entire business process model rather than a single activity. Various approaches reviewed in Section 2.3 of Chapter 4 looked into this problem from different perspecives such are removing unwanted process elements, abstraction and aggregation of activities or propagation of capabilities. These contributions either fail in proposing a complete capability description or trigger information loss or result in a complicated capabilities. The approach that I proposed in Chapter 4 relies on the same capability meta-model of Chapter 3. An aggregated capability results also in a set of actions and properties that domain experts are familiar with. The proposed aggregation algorithm is based on the token propagation game similar to Petri-Nets which gives a clear formal background of each propagation operation.

The third objective of this thesis was to explore the use of Formal Concept Analysis for providing efficient indexing and discovery of business capabilities that are described using the proposed model. Rather than inventing new a set of indexing and discovery algorithms, I reused Formal Concept Analysis (FCA for short) as a mathematical classification tool. Using FCA has the advantage of benefiting from already well established indexing and search algorithms that I have adopted for discovering a set of service descriptions (i.e., capabilities). In the evaluation of this work, I was observing the time required to create an index of synthetic capabilities with different configurations. Results of the evaluations show that the approach is effective and performs better than the related approaches. Details about reviewed related indexing and discovery approaches is given in Section 2.4 of Chapter 2.

The fourth objective of this thesis was to reduce the business process modelling effort when using configurable process models by proposing an algorithm for creating business capability-annotated configurable business process models that capture configuration options in terms of business capability features. First of all, in Section 2.5 of Chapter 2 I analyzed reuse-oriented business process modelling techniques in order to find how capabilities were used and how can the business capability model contribute to this area. Configuration-based modelling was a suitable starting point. The idea is to start from a reference process model (called a configurable model) and tailor this model to meet the end-user needs by enabling or disabling several branches of the model. The current state of configurable model requires an extensive process modelling techniques to carry out proper configurations (captured in terms of model restrictions and parameters) that reflect the business needs of the end-user. I suggest in Chapter 6 the early integration of business capability descriptions of activities in process models and create configurable models that capture configuration options from a functional perspective (capabilities parameters) in order to shift the configuration from manipulating the model directly to manipulating the parameters of its capabilities. My proposed solution has been tested on real world business process models from municipalities.

7.2 Contributions

The core contributions of this thesis are summarized as follows:

- *A capability meta-model for defining domain capability ontologies* (see Chapter 3): the thesis presented a meta-model for creating domain specific capabilities. This model has been serialized as a set of RDF vocabularies in order to guarantee its openness and portability. The approach has been subject to various publications [10, 21, 22, 51, 52, 82].

- *An algorithm to automatically generate the aggregated capability of an entire capability-annotated business process model* (see Chapter 4): I presented a propagation algorithm that starts from a capability-annotated business process model and delivers its entire capability. The algorithm is formally verified with formal semantics using Petri Nets [170] and implemented in a tool to support the validation by interviewing domain experts.The algorithm has been subject to one publication [53].

- *The validation of applicability of formal concept analysis for indexing and discovering efficiently capabilities* (see Chapter 5): I applied Formal Concept Analysis as a mechanism for indexing a set of capabilities. I validated this approach on a

real world Linked Energy Intelligence scenario to evaluate its applicability. Additionally, I carried out efficiency evaluation by measuring the size of the generated concept lattice for 5000 capabilities and its construction and navigation time (i.e., time required to create a concept lattice and discover a capability by visiting all the nodes of the lattice). The results of the evaluations showed that the approach outperformed related works in terms of time performance (creating the indexing structure in less than 200 ms) as it does not rely on any reasoning. The approach has been subject to various publications [56, 57].

- *An algorithm for automatically creating a capability-annotated configurable process model by merging a set of capability-annotated business process variants* (see Chapter 6): Using the proposed capability model. I designed an algorithm that can merge a set of capability-annotated business process models and create a capability-annotated configurable business process model. The generated model captures the configuration options in terms of capability properties that domain experts find easy to manage. The algorithm has been tested on real world business process models. In terms of quantitative analysis of this work, the time required to merge the input models is few milliseconds and the compression rate gained to is around 50%. Interviews with domain experts helped also validate this contribution and critically analyse it. The experts confirm that current configuration options are focused on technical parameters, the use of business capabilities can be added as another dimension to target business users and used for training and documentation purposes. The approach has been subject to various publications [49, 50, 55].

Other contributions of this thesis in terms of research, tools, vocabularies and test collections include:

- *Literature review and Gap analysis* (see Chapter 2): a related work analysis of contributions that are either directly or indirectly targeting capability-based service and business process research. More specifically it targeted contributions that focused primarily on modelling capabilities then indexing, discovering and aggregating them. An analysis of related work for the problem of reuse in business process modelling was performed, specifically the use of capabilities in those works.
- A new version of EPCTools [1] that includes the implementation of: (i) an annotation tool for integrating capability descriptions in business process models with

[1] Original version of EPCTools: http://www2.cs.uni-paderborn.de/cs/kindler/Forschung/EPCTools/ as accessed on the 03/10/2013.
New version of EPCTools available at: http://wassimderguech.org/phd/

respect to the proposed meta-model in Chapter 3, (ii) the capabilities aggregation algorithm proposed in Chapter 4 and (iii) the merging algorithm proposed in Chapter 6 for creating configurable process models.

- A set of RDF vocabularies [2] used in defining the capability meta-model introduced in Chapter 3.
- A test collection of business process models for municipalities, available in Appendix E.
- A test collection of business process models from the customs clearance domain, available in Appendix A.

7.3 Critique and Limitations

7.3.1 Capability Meta-Model

In Chapter 3, I defined a meta model for describing Capabilities using the frame-based paradigm by featuring domain specific actions and properties. This Model permits the description of capabilities independently from their actual implementations by highlighting the actions being performed with a set of related properties. This idea focused on describing what can be done rather than on the change of the world in terms of Input, Output, Precondition and Effect (IOPE for short).

The main advantages of modelling capabilities as such are:

First, contrary to the IOPE paradigm it features the functional (business) and non-functional characteristics which end-users are mostly interested in and which are specified in their requests. This constitutes a natural way on how users describe their needs, for example, users need a *service that* ***ships packages from*** *an address* ***to*** *another*. This is expressed in my model via the action category "ship" and the properties "package", "from" and "to".

Second, the meta model defines semantic links between Capabilities in terms of specifications and extensions. By using these relations Capability owners/providers can rapidly and easily define new Capabilities by reusing previous definitions. In addition, these relations define a cloud of Capabilities where navigation techniques can be developed as an alternative to goal based discovery techniques.

A cloud of capabilities description can be easily queried using SPARQL. I use RDF as a lightweight language for describing and linking capabilities descriptions whereas one can

[2] These vocabularies are made publicly available and will be introduced in Chapter 3

use SPARQL for advanced querying including the usage of SPARQL as a rule language [171].

Furthermore, in the proposed model, I introduced the concept of an action category for describing the actual action being performed by a capability using a natural language indication. Indeed, I did not want to tie the use of this concept to a simple literal or a verb. This concept is not the actual label that a capability can achieve, it is a concept from a domain ontology that might be accompanied by a label which is an expression describing the actual service action in a natural language. It is also possible to enrich this label by other related terms such as synonyms. Natural Language Processing [215] can be applied together with WordNet verb synsets for generating possible synonyms to generate a particular label for the action category concept of the capability description.

Finally, since the meta model is RDF based it can be easily extended, while preserving the property-featured principle, by considering other types of properties (such as optional and mandatory properties) and other types of property values.

However, I provide with this model a coarse-grained semantics of the capability effect or changes on the state of the world. While this coarse-grain semantics is adequate for conducting discovery, it is insufficient when automated chaining is required such as in composition and planning scenarios. By coarse grain semantics I mean defining shared agreement on coarse grain entities such as capability in my case. Automated chaining requires finer-grained semantics. For that purpose, I consider a domain related ontology that defines a detailed (i.e., fine grained) semantics of the abstract capabilities. This domain related ontology taken together with some relations between attributes can help to determine the fine grained semantics of a capability description. This particular idea of generating fine-grained semantics from a detailed domain ontology is part of future investigations.

7.3.2 Capabilities Aggregation

Chapter 4 presented one possible technique that allows to derive a high level process description in terms of a capability from a detailed model. I assumed for this work that each activity of the input model is annotated with its capability using the conceptual model proposed in Chapter 3. At some point this assumption might be questionable. Actually, it is very hard to find process models that semantically annotated with the required information (e.g., capabilities in my case). This is a common problem for any work that requires semantically annotated process models. Even though I provided the required tools to perform such annotations, manual effort remains high.

7.3.3 Capability Indexing and Discovery

I proposed in Chapter 3 to model capabilities as property-featured entities where each property reports on a particular characteristic of a described capability. The conceptual model is flexible enough to consider even non-functional properties to include for instance quality of service properties. In Chapter 5, I applied this approach for modelling sensor capabilities, from a real world dataspace, featuring both functional and non-functional properties. On top of these sensor capabilities descriptions, I used Formal Concept Analysis (FCA) for indexing them. The resulting indexing structure is called concept lattice that can serve for several use cases for example the discovery of a replacement sensor. Using FCA in such use case is recommended only if the number of objects (i.e., sensors) is not very big. Actually, this constitutes the major disadvantage of my approach.

The experiments show that reconstruction of the entire concept lattice with 5000 sensors does not exceed a few milliseconds. I conclude that it is acceptable to reconstruct the entire concept lattice when dealing with dynamic environments when a change in the environment has been detected. This method can be efficient but it is very costly and could not be very useful in highly dynamic environments when several sensors could be added and removed frequently or where the values of properties might change after a short period. A future research direction can investigate the required algorithms for updating this indexing structure (i.e., removing or adding a sensor description) in order to deal with dynamic changes in the environment.

The current version supports only primitive types of properties (e.g., boolean, integer, float and string). However, in Chapter 3 the conceptual model supports other complex property types such as conditional values, enumeration values, dynamic values, etc. Currently I am simply transforming any property entry by a substitute property of type boolean to report if a capability has a property or not. This method can be useful for a rapid construction of the concept lattice with less effort in the scaling operation but this comes along with information loss in terms of expressiveness of the concept lattice.

7.3.4 Capability-driven Customization of Configurable Business Process Models

Chapter 6 presented a novel algorithm that allows for merging a collection of capability-annotated business process models in order to create a capability-annotated configurable process model. Even though it used EPC to illustrate the running example and the results of the steps of this algorithm, the approach is not exclusively made for EPCs.

Indeed, I represent a process variant as a business process graph which has been used throughout the steps of the merging algorithm.

The algorithm ensures that the resulting capability-annotated configurable model includes the behaviours of the original business process variants by considering nodes with identical action categories and preserving the status of routing nodes.

7.4 Future Research Directions

The work carried out in this thesis opens various research questions that can be part of future works. These research directions include:

On the business capability meta-model: In my research I explored some of the foundations and methodologies required to describe business capabilities. Some of the design and implementation choices were driven by the context of my work. For example the specification and extension relations that I identified in Chapter 3 are mostly influenced by the indexing technique explored in Chapter 5. Further research can dig deeper into the conceptual mode itself by considering the following directions:

- In the current state, specification and extension relations define coarse-grained relations between business capabilities. They simply report whether a capability has more properties than another and whether some of the shared properties have more specific values. They do not indicate what are these properties. This information can speed up the discovery process and decision making if one needs to replace a service by another by simply looking at the relations between their capabilities without necessarily referring the actual capabilities or their respective documentation. This can be done by extending current specification and extension relations to what I call extended or fine-grained relations. An extended relation specifies which attribute the basic relation, it derives from.
- Rather than using only specification and extension relations, one can investigate other relations that might be useful for creating the business capabilities hierachy/cloud such as: *share*, *shareSame*, *shareDifferent*, *differMore* and *differLess*. While some of these relations do not bring much information on themselves, they can be used to compute other relations. This moves the current vision of indexing entities in a graph with simple relations between entities to more refined relations that can be interpreted with various aspects of interest. This can help connecting various data sets of business capabilities across multiple authorities without

necessarily having a centralised management. This vision can be aligned with the Linked Open Data Cloud (LOD Cloud) [40] to be become LOD Cloud of Business Capabilities.

On the transition From/To Documentation to/from business capabilities: While carrying out interviews with domain experts, one of the suggestions was the automatic generation of the documentation of processes and services using the business capability description. This idea can be further explored in both ways: (1) generating documentation from business capabilities as well as (2) generating capabilities from textual documentation or IT specifications (e.g., WSDL files). Thus, two research directions can be explored:

- The business capability as implemented in this thesis is well structured. It explicitly lists for its properties various types of values: conditional, range, enumeration, etc. Each of these types has a clear definition on its semantics. In this regard, Natural Language Generation (NLG) techniques can be used to provide a textual description of each of these values. This can be extended to the entire capability description and generate a textual description that serves as a documentation of services and processes. This can also explore another dimension of human understanding by providing customised summary of the capability using natural language.

- It is also possible to use Natural Language Processing (NLP) techniques for providing suggestions of business capabilities. This can be used either to fully automate the annotations of services and processes with their capabilities or propose autocompletions during their manual annotations. For example, a business capability can be linked to the so called *"case frames"* in linguistics as it is offered by FrameNet project [11]. In linguistic study, the capability of an action is defined by an *action verb* that is quite similar to the *action category* in my proposed model. Then several dimensions may extend that verb by giving more details about the carried work and related aspects. While in FrameNet these dimensions are predefined such as: agentive, dative, objective, etc. in my case I do not impose any predefined dimensions. However, it is possible to establish links between both models to generate structured business capabilities from textual descriptions using linguistic case frames. Such idea has been discussed by Leopold et al. [131], Rahm and Bernstein [175] and Mendling and Simon [149] for automatically annotating process activities with their action verbs derived from the textual documentation.

One step further, Gao and Bhiri [81] explore the idea of mapping case frame dimensions to capability properties to generate a full structured capability using the model proposed in this thesis.

On the aggregation of business capabilities: During the evaluation of the capability aggregation approach, it has been pointed that one of the weaknesses of the approach is that it relies on using a single capability ontology for the entire process. A solution to this issue can take part of future research directions:

- The work proposed provides the intended results under the assumption of using the same capability ontology for annotating the entire process. Beyond this assumption, a revision of how the action categories as well as the related properties are aggregated. It is common that each company creates its own ontology of actions when conceptualizing its own capabilities models. When these models are shared within other partners, they should subsequently consider the original ontology of actions and capabilities and not only the one they have. For example, several ontologies of actions can be proposed for the same domain which imposes defining a more advanced method for finding the right action categories of the entire model. Furthermore, the propagation approach for identifying the properties of an aggregated capability also needs to be refined.

On the indexing and discovery of business capabilities: Besides looking into other techniques for indexing repositories of capabilities, improving the current approach can be further explored by investigating the following issues:

- In formal concept analysis, a formal context considers only single-valued attributes. In case of multi-values attributes, a scaling operation is required for transforming them into multiple single-valued attributes. In this thesis I used only simple types of service capabilities that can be easily scaled from multi-valued to single-valued attributes. However, the model permits the modelling of complex values such range and conditional values. Investigating scaling operations for covering these complex attribute types is required in order to consider them in the indexing approach using Formal Concept Analysis.
- One of the major issues with using Formal Concept Analysis is the need to rebuild the entire concept lattice when a new entry is considered in the formal context. Even though my experiments show that the time required to build the entire structure is in the order of milliseconds with 5000 entities, this might be a problem when we move to higher order of magnitude such a web-scale level. Strategies that

can be explored in future works include either (1) building multiple lattices and interlink them: lattice of lattices; or (2) providing the necessary algorithms for maintaining the capabilities lattice through incremental built of the lattice [87].

On the use of business capabilities in process configuration: The use of capabilities in the configuration of reference process models as discussed in Chapter 6 is intended towards the design of configurable models that capture variation options in terms of capabilities properties. Future work in the configuration phase is needed and ideas of explorations are as follow:

- The configuration-based modelling introduces two steps in the modeling phase of business process management: (1) design of configurable models and (2) configuration and individualisation. The work in this thesis contributes to the first step, while the second step has not been tackled. Future works should include (1) formally defining of the configuration and individualisation phase, (2) identifying the configuration dependencies in order to direct the user to a starting configuration point and take subsequent configuration decisions, (3) controlling the configuration steps to ensure correct results, (4) enhancing the user experience during the configuration, (5) recommending possible configurations via process mining techniques, etc.
- The entire thesis focused on the modelling of business capabilities using a well structured model. The model can be further extended to model other aspect of the information systems: roles, resources, costs, etc. These aspects can also be used for driving the configuration of business processes and identify the impact of configuration decision when for example changing roles, substituting available resources, changing a certain supplyer, etc.

Appendix A

Business Process Models for Import Procedures

This appendix contains the business process models used in the evaluation of Chapter 4. These models have been manually created from online resources. Originally, the models were presented using non standard flowcharts describing steps that need to be taken followed by a detailed textual descriptions.

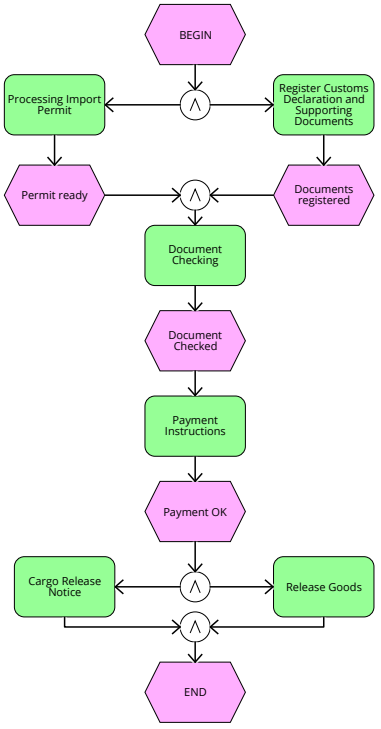


FIGURE A.1: Import Procedure by TRADE VAN

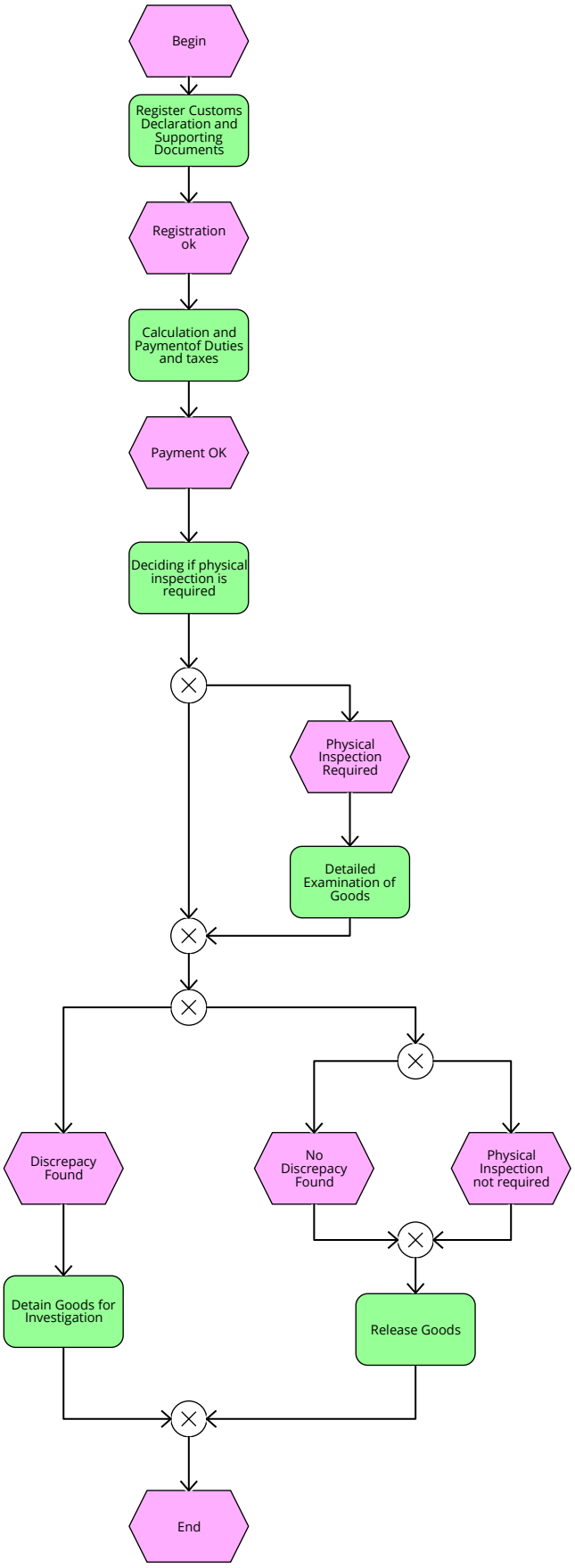


FIGURE A.2: Import Procedure in Cambodia

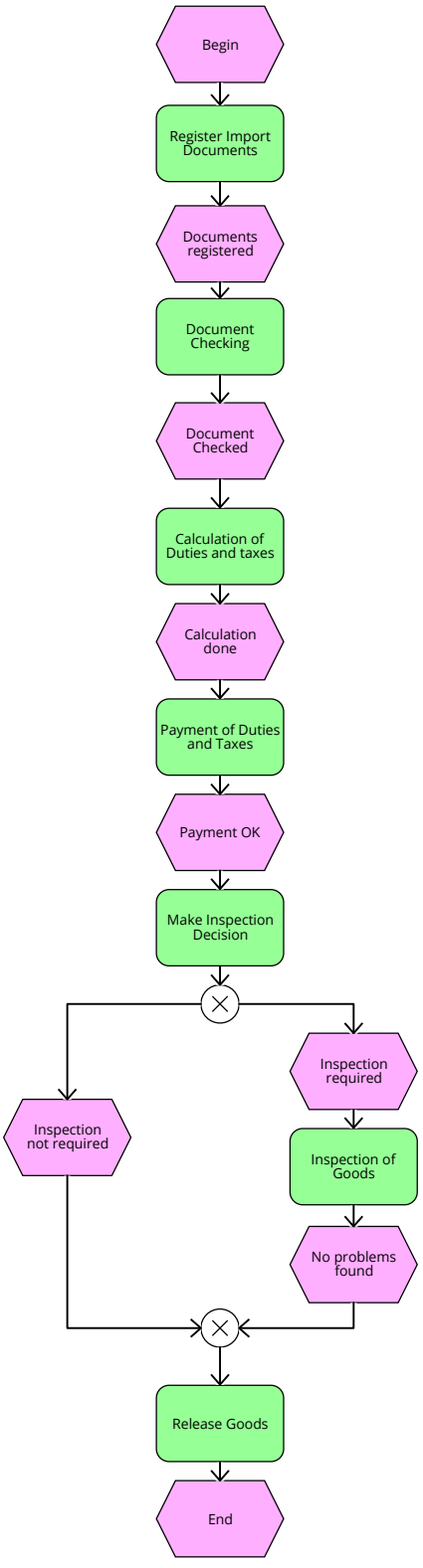


FIGURE A.3: Import Procedure by TILC in China

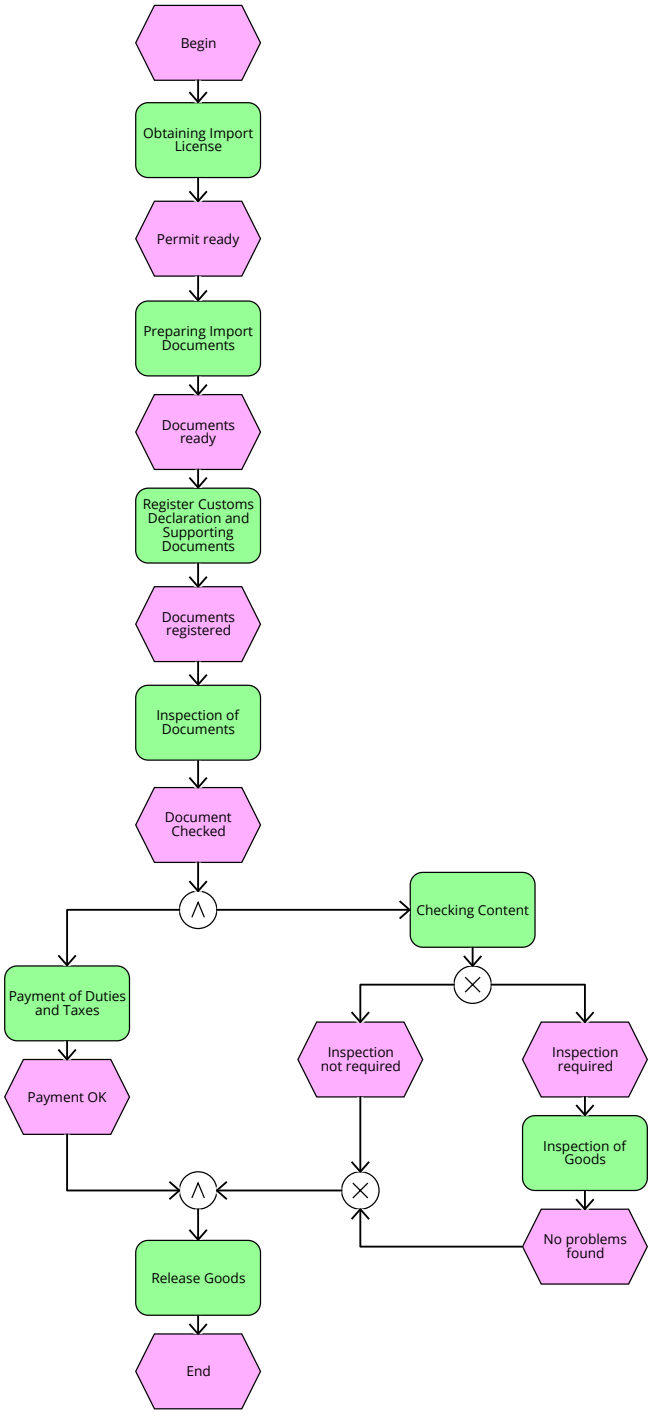


FIGURE A.4: Import Procedure in Tereshia

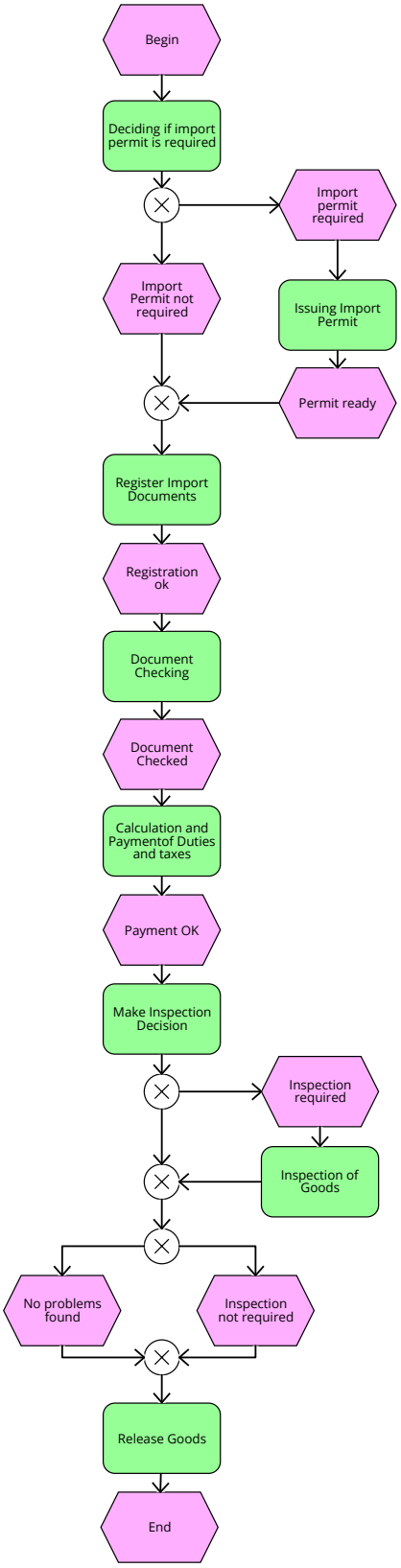


FIGURE A.5: Import Procedure in Port Klang

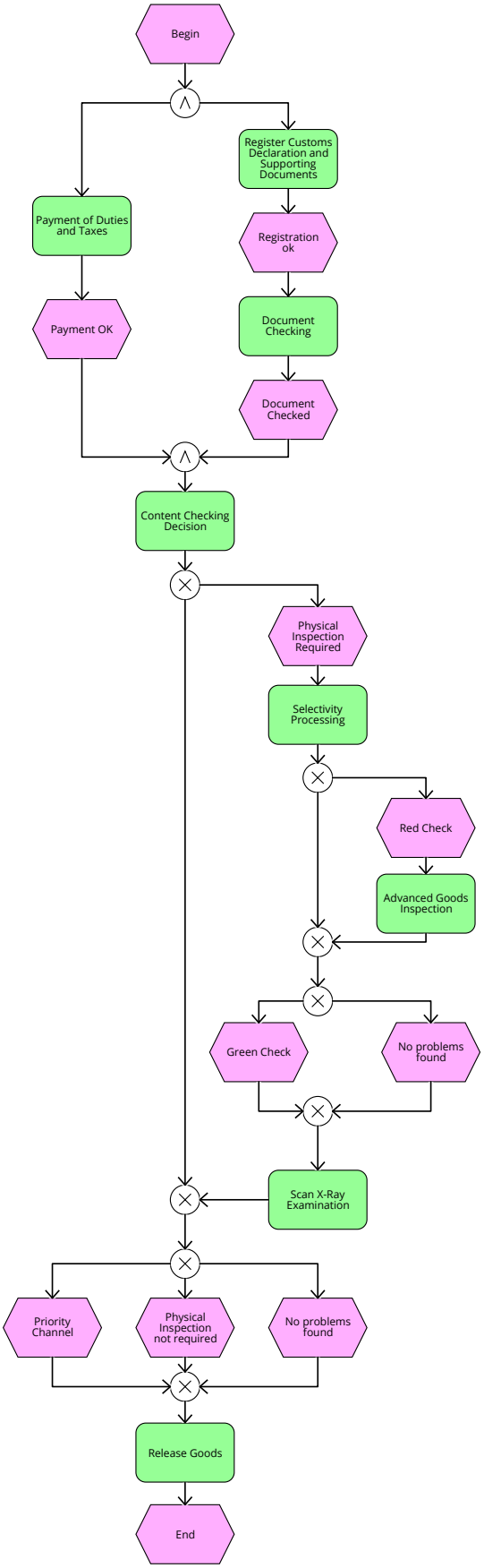


FIGURE A.6: Import Procedure in Philippines

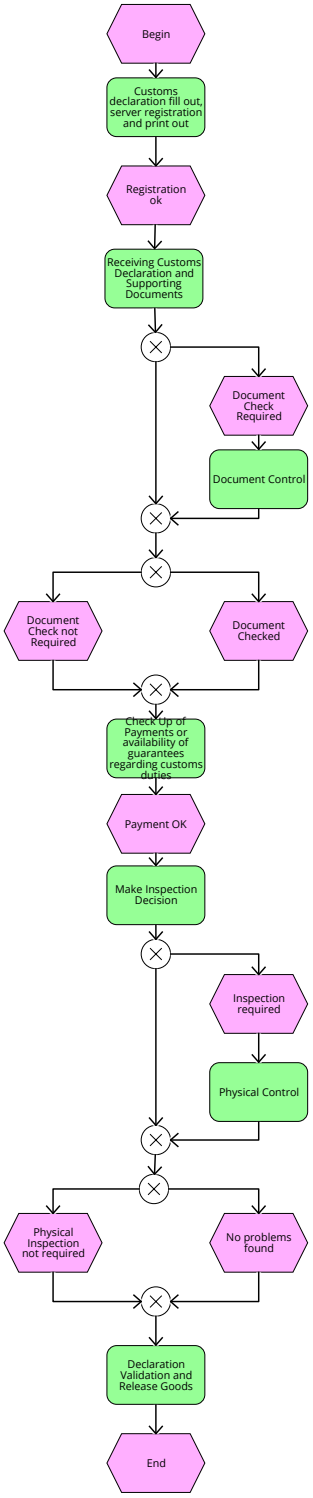


FIGURE A.7: Import Procedure in Moldova

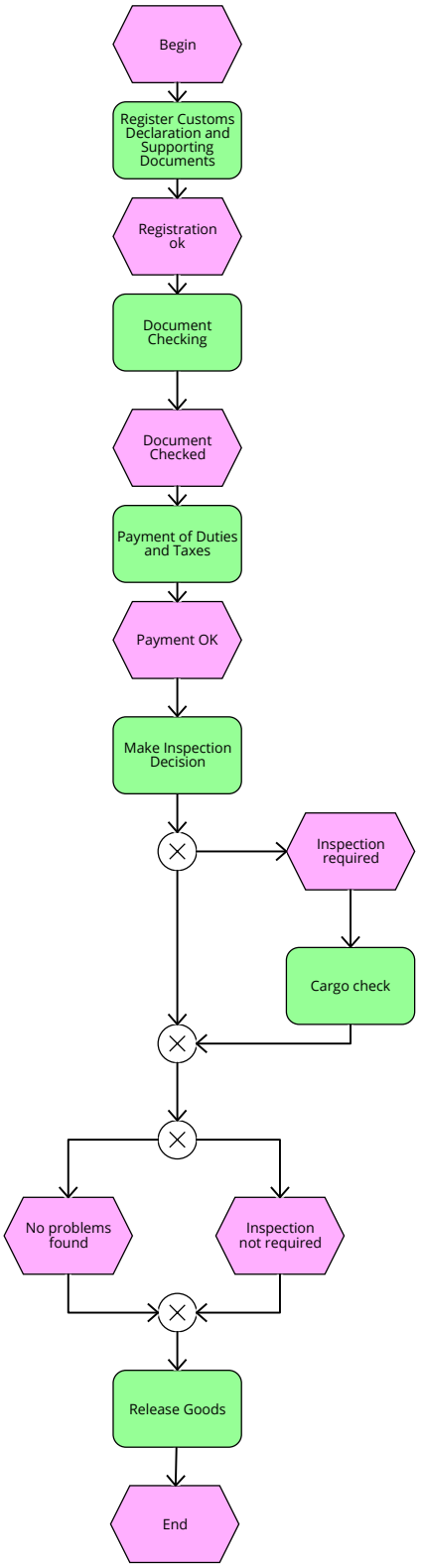


FIGURE A.8: Import Procedure in Sihanoukville seaport inShanghai, China

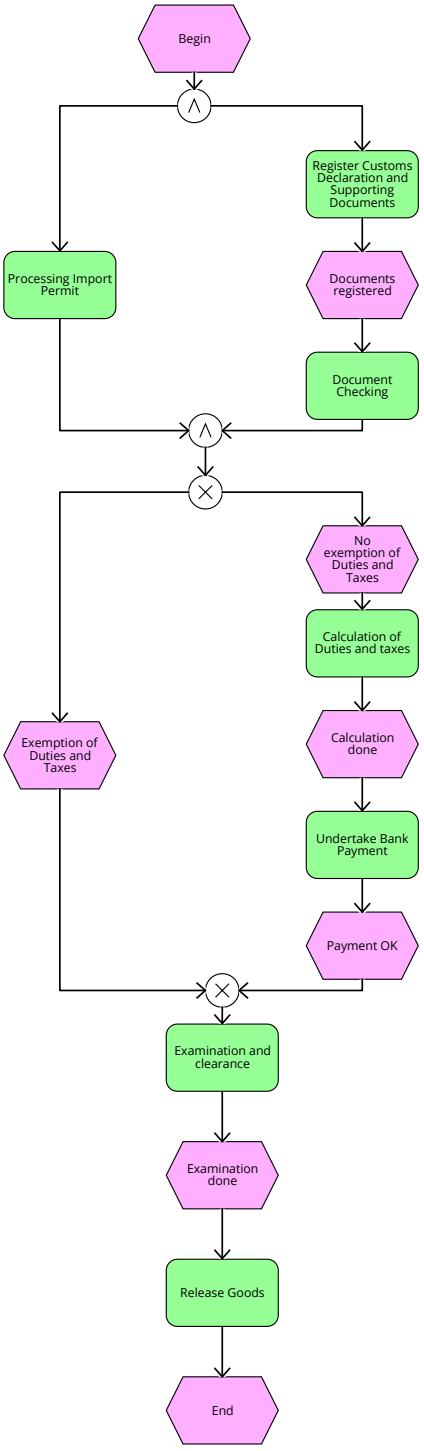


FIGURE A.9: Import Procedure in Bangladesh

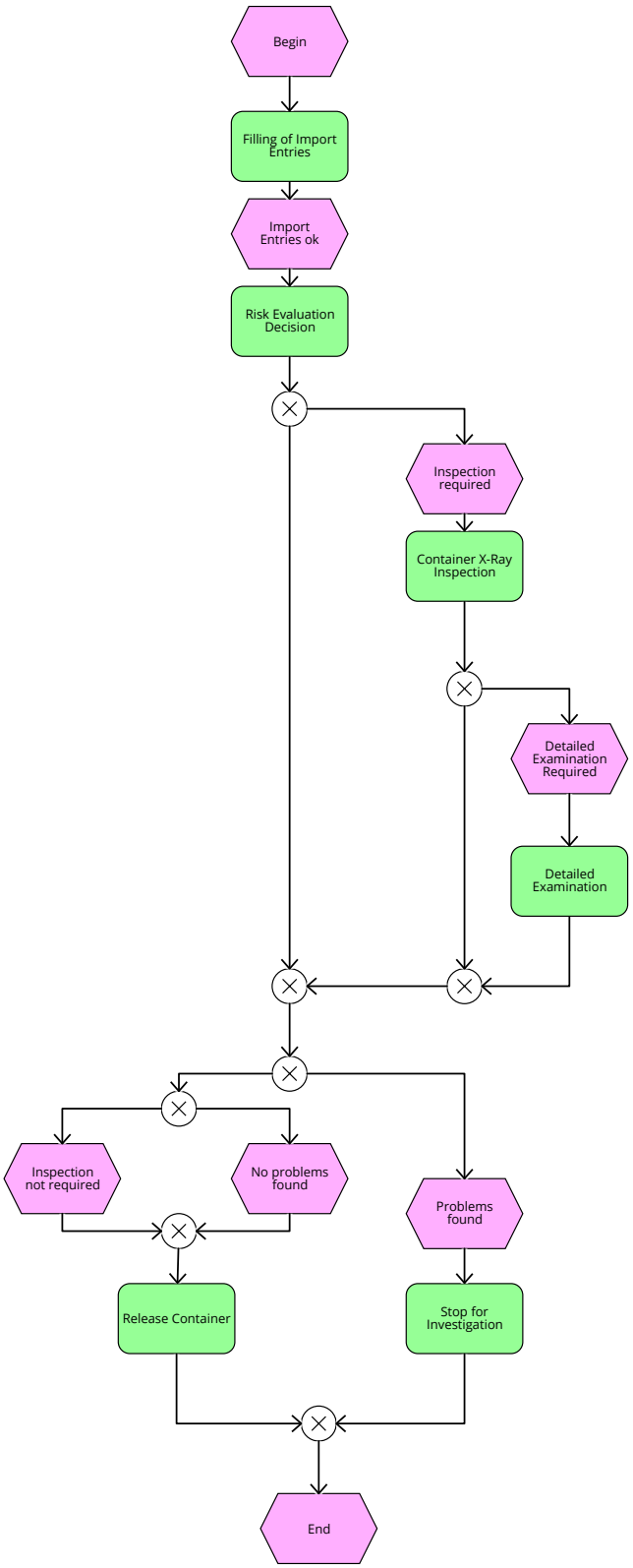


FIGURE A.10: Import Procedure by ASYCUDA

Appendix B

Capability Meta Model

This appendix contains RDF/N3 listing (Listing B.1) of the Capability Meta-Model (CMM). CMM permits describing capabilities as an action category and a set of properties. Additionally, this schema defines the possible property value types that can be considered when modelling capabilities: ConditionalValue, ConstrainedValue, EnumerationValue, DynamicValue and RangeValue.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix cmm: <http://vocab.deri.ie/cmm#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .


cmm:Capability a rdfs:Class, owl:Class;
    rdfs:label "Capability" .

cmm:Constraint a rdfs:Class, owl:Class;
   rdfs:label "Constraint" .

cmm:PropertyValue a rdfs:Class, owl:Class;
    rdfs:label "PropertyValue";
    rdfs:comment "This concepts defines the possible
                  values a particular property can have." .

cmm:ConditionalValue a rdfs:Class, owl:Class;
    rdfs:label "ConditionalValue";
    rdfs:subClassOf cmm:PropertyValue .

cmm:EnumerationValue a rdfs:Class, owl:Class;
    rdfs:label "EnumerationValue";
    rdfs:subClassOf cmm:PropertyValue .
```

```
28
29 cmm:Expression a rdfs:Class, owl:Class;
30     rdfs:label "Expression" .
31
32 cmm:ConstrainedValue a rdfs:Class, owl:Class;
33     rdfs:label "ConstrainedValue";
34     rdfs:subClassOf cmm:PropertyValue .
35
36 cmm:DynamicValue a rdfs:Class, owl:Class;
37     rdfs:label "DynamicValue";
38     rdfs:subClassOf cmm:PropertyValue .
39
40 cmm:RangeValue a rdfs:Class, owl:Class;
41     rdfs:label "RangeValue";
42     rdfs:subClassOf cmm:PropertyValue .
43
44 cmm:hasExpression a rdf:Property;
45     rdfs:label "hasExpression";
46     rdfs:domain cmm:Constraint;
47     rdfs:range cmm:Expression .
48
49 cmm:achieves a rdf:Property;
50     rdfs:label "achieves";
51     rdfs:domain cmm:Capability .
52
53 cmm:hasMax a rdf:Property;
54     rdfs:label "hasMax";
55     rdfs:domain cmm:RangeValue;
56     rdfs:range cmm:PropertyValue .
57
58 cmm:property a rdf:Property;
59     rdfs:label "property";
60     rdfs:domain cmm:Capability;
61     rdfs:range cmm:PropertyValue .
62
63 cmm:hasMin a rdf:Property;
64     rdfs:label "hasMin";
65     rdfs:domain cmm:RangeValue;
66     rdfs:range cmm:PropertyValue .
67
68 cmm:constrainedBy a rdf:Property;
69     rdfs:label "constrainedBy";
70     rdfs:domain cmm:ConstrainedValue;
71     rdfs:range cmm:Constraint .
72
73 cmm:specifies a rdf:Property;
74     rdfs:label "specifies";
75     rdfs:domain cmm:Capability;
76     rdfs:range cmm:Capability .
77
78 cmm:exprValue a rdf:Property, owl:DatatypeProperty;
79     rdfs:label "exprValue";
80     rdfs:domain cmm:Expression;
81     rdfs:range xsd:string .
82
```

```
83  cmm:hasEvaluator a rdf:Property;
84      rdfs:label "hasEvaluator";
85      rdfs:domain
86          cmm:ConditionalValue,
87          cmm:DynamicValue;
88      rdfs:range cmm:Expression .
89
90  cmm:hasElement a rdf:Property;
91      rdfs:label "hasElement";
92      rdfs:domain cmm:EnumerationValue;
93      rdfs:range cmm:PropertyValue .
94
95  cmm:hasCondition a rdf:Property;
96      rdfs:label "hasCondition";
97      rdfs:domain cmm:ConditionalValue;
98      rdfs:range cmm:Constraint .
99
100 cmm:extends a rdf:Property;
101     rdfs:label "extends";
102     rdfs:domain cmm:Capability;
103     rdfs:range cmm:Capability .
104
105 cmm:exprType a rdf:Property, owl:DatatypeProperty;
106     rdfs:label "exprType";
107     rdfs:domain cmm:Expression;
108     rdfs:range xsd:string .
```

Listing B.1: RDF N3 representation of the Capability Meta-Model

Appendix C

Import Actions Ontology

This appendix contains RDF/N3 listing (Listing C.1) of the Import Actions Ontology (IMP) depicted in Figure C.1.

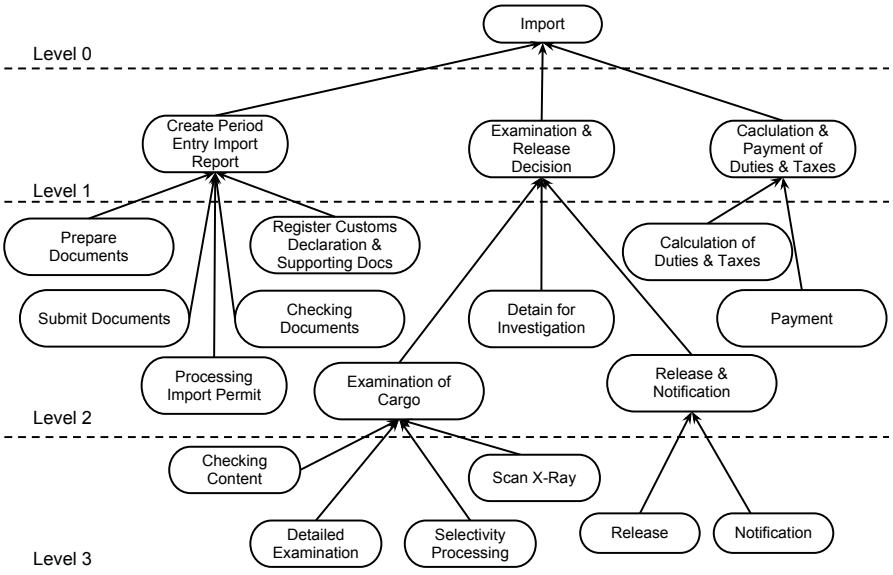


FIGURE C.1: Actions Ontology of the import domain

```
@prefix imp: <http://vocab.deri.ie/Imp#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix av: <http://vocab.deri.ie/av#> .

imp:Importation
rdf:type av:ActionVerb ;
rdfs:comment "The importation consists of transactions in goods (cargo) from a
    foreign country"^^xsd:string ;
```

```
av:hasLevel 0 ;
av:hasPart imp:CreateImportReport , imp:CalculationAndPaymentOfDutiesAndTaxes , imp:ExaminationAndReleaseDecision ;
skos:prefLabel "Import"^^xsd:string .

imp:CreateImportReport
rdf:type av:ActionVerb ;
rdfs:comment "Creating an Import report summarizes information about imports This task includes processing the import permit, registering import declaration, preparation, submission and checking of import documents."^^xsd:string ;
av:hasLevel 1 ;
av:hasPart imp:RegisterCustomDeclarationAndSupportingDocs , imp:CheckingDocs ;
av:hasOptionalPart imp:ProcessingImportPermit , imp:PrepareDocs , imp:SubmitDocs ;
skos:prefLabel "Create import report"^^xsd:string .

imp:ProcessingImportPermit
rdf:type av:ActionVerb ;
rdfs:comment "Processing Import Permit consists of obtaining the required license for being able to import goods. If an organization posses already an import permit, there is not need to apply for it again. Government agencies do not need to process an import permit."^^xsd:string ;
av:hasLevel 2 ;
skos:prefLabel "Processing import permit"^^xsd:string .

imp:PrepareDocs
rdf:type av:ActionVerb ;
rdfs:comment "Preparing the required documents for importing a cargo (Forms, licenses, etc.)."^^xsd:string ;
av:hasLevel 2 ;
skos:prefLabel "Prepare documents"^^xsd:string .

imp:SubmitDocs
rdf:type av:ActionVerb ;
rdfs:comment "Submitting import documents either online or in person depending on the available means."^^xsd:string ;
av:hasLevel 2 ;
skos:prefLabel "Submit documents"^^xsd:string .

imp:RegisterCustomDeclarationAndSupportingDocs
rdf:type av:ActionVerb ;
rdfs:comment "Registration of the customs declaration and the supporting documents."^^xsd:string ;
av:hasLevel 2 ;
skos:prefLabel "Register customs declaration and supporting documents"^^xsd:string .

imp:CheckingDocs
rdf:type av:ActionVerb ;
rdfs:comment "Checking the validity and completeness of import documents."^^xsd:string ;
av:hasLevel 2 ;
skos:prefLabel "Checking Documents"^^xsd:string .

imp:CalculationAndPaymentOfDutiesAndTaxes
```

```
54 rdf:type av:ActionVerb ;
55 rdfs:comment "The calculation is dependent upon commodity and duty regimes
       allowable by country and location of import. The payment is dependent upon
       the available means."^^xsd:string ;
56 av:hasLevel 1 ;
57 av:hasPart imp:CalculationOfDutiesAndTaxes , imp:Payment ;
58 skos:prefLabel "Calculation and payment of duties and taxes"^^xsd:string .
59
60 imp:CalculationOfDutiesAndTaxes
61 rdf:type av:ActionVerb ;
62 rdfs:comment "The calculation is dependent upon commodity and duty regimes
       allowable by country and location of import."^^xsd:string ;
63 av:hasLevel 2 ;
64 skos:prefLabel "Calculation of duties and taxes"^^xsd:string .
65
66 imp:Payment
67 rdf:type av:ActionVerb ;
68 rdfs:comment "The payment is dependent upon the available means (i.e., cash,
       cheques, money transfer, etc...)."^^xsd:string ;
69 av:hasLevel 2 ;
70 skos:prefLabel "Payment"^^xsd:string .
71
72 imp:ExaminationAndReleaseDecision
73 rdf:type av:ActionVerb ;
74 rdfs:comment "Examination of the cargo and deciding if it is going to be released
        or detained for investigation"^^xsd:string ;
75 av:hasLevel 1 ;
76 av:hasPart imp:ReleaseAndNotification ;
77 av:hasOptionalPart imp:ExaminationOfCargo , imp:DetainForInvestigation ;
78 skos:prefLabel "Examination and release decision"^^xsd:string .
79
80 imp:ExaminationOfCargo
81 rdf:type av:ActionVerb ;
82 rdfs:comment "The examination of the cargo is dependent upon the nature of the
       goods being imported. By the change the content, a decision is made to select
        the type of required examination."^^xsd:string ;
83 av:hasLevel 2 ;
84 av:hasPart imp:CHKContent , imp:DetailedExamination ;
85 av:hasOptionalPart imp:SelectivityProcessing , imp:ScanXRay ;
86 skos:prefLabel "Examination of Cargo"^^xsd:string .
87
88 imp:CHKContent
89 rdf:type av:ActionVerb ;
90 rdfs:comment "Checking the the content consists of verifying the list of goods
       being imported, the nature of the importer, etc. in order to decide wether an
        examination is necessary."^^xsd:string ;
91 av:hasLevel 3 ;
92 skos:prefLabel "Checking Content"^^xsd:string .
93
94 imp:SelectivityProcessing
95 rdf:type av:ActionVerb ;
96 rdfs:comment "Selectivity processing consists of deciding what type (i.e.,
       detailed, lab analysis, x-ray scan, etc.) of examination is required."^^xsd:
       string ;
97 av:hasLevel 3 ;
```

```
skos:prefLabel "Selectivity processing"^^xsd:string .

imp:DetailedExamination
rdf:type av:ActionVerb ;
rdfs:comment "A detailed examination may include a lab analysis, opening the
    cargo, etc."^^xsd:string ;
av:hasLevel 3 ;
skos:prefLabel "Detailed examination"^^xsd:string .

imp:ScanXRay
rdf:type av:ActionVerb ;
rdfs:comment "A detailed examination may include a lab analysis, opening the
    cargo, etc."^^xsd:string ;
av:hasLevel 3 ;
skos:prefLabel "Scan XRay"^^xsd:string .

imp:DetainForInvestigation
rdf:type av:ActionVerb ;
rdfs:comment "In case of a problem the cargo might be detained for investigation
    ."^^xsd:string ;
av:hasLevel 2 ;
skos:prefLabel "Detain for investigation"^^xsd:string .

imp:ReleaseAndNotification
rdf:type av:ActionVerb ;
rdfs:comment "Releasing the cargo and sending a notification to the importer."^^
    xsd:string ;
av:hasLevel 2 ;
av:hasPart imp:Release ;
av:hasOptionalPart imp:Notification ;
skos:prefLabel "Release and notification"^^xsd:string .

imp:Release
rdf:type av:ActionVerb ;
rdfs:comment "Releasing the cargo."^^xsd:string ;
av:hasLevel 3 ;
skos:prefLabel "Release"^^xsd:string .

imp:Notification
rdf:type av:ActionVerb ;
rdfs:comment "Sending a release notification to the importer."^^xsd:string ;
av:hasLevel 3 ;
skos:prefLabel "Notification of Release"^^xsd:string .
```

Listing C.1: RDF N3 representation of the Capability Meta-Model

Appendix D

Import Capabilities Domain Ontology

This appendix contains RDF/N3 listing (Listing D.1) of the Import Capabilities Domain Ontology (IMPC).

```
@prefix impc: <http://vocab.deri.ie/ImportCapability#> .
@prefix cap: <http://vocab.deri.ie/cap#>.
@prefix imp: <http://vocab.deri.ie/Imp#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix av: <http://vocab.deri.ie/av#> .
@prefix owl: <http://www.w3.org/2002/07/owl#>.

impc:ProcessingImportPermit a cap:Capability ;
cap:hasActionVerb imp:ProcessingImportPermit;
impc:hasImportPermit impc:ImportPermit.

impc:ImportPermit a impc:Document;
rdfs:label "Import Permit" ;
rdfs:comment "This document is required for the release for free circulation of
    the good. Also referred as Import License.".

impc:Document a rdfs:Class;
rdfs:label "Document";
rdfs:comment "This class refers to any kind of document.".

impc:PrepareDocuments a cap:Capability ;
cap:hasActionVerb imp:PrepareDocs;
impc:hasDocuments impc:Documents.

impc:Documents a cap:EnumerationValue;
```

```
cap:hasElement impc:BillOfLading, impc:ImportPermit, impc:Invoice, impc:
    DeclarationForm, impc:TradeAgreement, impc:DeliveryOrder, imp:OtherDocuments.

impc:BillOfLading a impc:Document;
rdfs:label "Bill of Lading";
rdfs:comment "A document issued by a carrier which details a shipment of
    merchandise and gives title of that shipment to a specified party.".

impc:Invoice a impc:Document;
rdfs:label "Invoice";
rdfs:comment "The invoice on which the Customs value of the goods is declared.".

impc:DeclarationForm a impc:Document;
rdfs:Label "Declaration Form";
rdfs:comment "Customs declaration form for import.".

impc:TradeAgreement a impc:Document;
rdfs:Label "Trade Agreement";
rdfs:comment "Preferential trade agreements or other reliefs from duty including
    for example bills of lading.".

impc:DeliveryOrder a impc:Document;
rdfs:label "Delivery Order";
rdfs:comment "A document from a consignor, a shipper, or an owner of freight
    which orders the release of the transportation of cargo to another party (i.e
    ., the importer).".

imp:OtherDocuments a impc:Document;
rdfs:label "Other Documents";
rdfs:comment "Any other document that can be useful for the importation (e.g.,
    Priority Statement).".

impc:PackingList a impc:Document;
rdfs:label "Packing List";
rdfs:comment "A document contains the list of imported items.".

impc:SubmitDocuments a cap:Capability;
cap:hasActionVerb imp:SubmitDocs;
impc:hasDocuments impc:Documents.

impc:RegCustomsDeclation a cap:Capability;
cap:hasActionVerb imp:RegisterCustomDeclarationAndSupportingDocs;
impc:hasDocuments impc:Documents.

impc:CHKDocs a cap:Capability;
cap:hasActionVerb imp:CheckingDocs;
impc:hasDocuments impc:Documents.

impc:CreateIMPReport a cap:Capability;
cap:hasActionVerb imp:CreateImportReport;
impc:hasDocuments impc:Documents.

impc:CalculationOfDT a cap:Capability;
cap:hasActionVerb imp:CalculationOfDutiesAndTaxes;
impc:hasFee impc:Fees.
```

```
77
78 impc:Fees a rdf:Class;
79 impc:hasAmount impc:Amount;
80 impc:hasCurrency impc:Currency.
81
82 impc:Amount a rdf:Class;
83 rdfs:label "Amount";
84 rdfs:comment "This represents the amount of Duties and Taxes.".
85
86 impc:Currency a xsd:string;
87 rdfs:label "Currency";
88 rdfs:comment "The currency used for the fees.".
89
90 impc:PaymentOfDT a cap:Capability;
91 cap:hasActionVerb imp:Payment;
92 impc:hasFee impc:Fees.
93
94 impc:CalculationAndPaymentOfDT a cap:Capability;
95 cap:hasActionVerb imp:CalculationAndPaymentOfDutiesAndTaxes;
96 impc:hasFee impc:Fees.
97
98 impc:CHKContent a cap:Capability;
99 cap:hasActionVerb imp:CheckingContent;
100 impc:hasCargo impc:Cargo;
101 impc:hasDecision impc:ExamDecision.
102
103 impc:Cargo a rdf:Class.
104
105 impc:Goods a rdf:Class.
106
107 impc:contains a rdf:Property;
108 rdfs:label "contains";
109 rdfs:domain impc:Cargo;
110 rdfs:range impc:Goods.
111
112 impc:hasCategory a rdf:Property;
113 rdfs:label "hasCategory";
114 rdfs:domain impc:Goods;
115 rdfs:range xsd:string.
116
117 impc:ExamDecision a cap:EnumerationValue;
118 cap:hasElement impc:PhysicalInspectionRequired;
119 cap:hasElement impc:PhysicalInspectionNotRequired;
120 cap:hasElement impc:PriorityChannel.
121
122 impc:PhysicalInspectionRequired a rdf:Class;
123 rdfs:label "Physical Inspection Required".
124
125 impc:PhysicalInspectionNotRequired a rdf:Class;
126 rdfs:label "Physical Inspection Not Required".
127
128 impc:PriorityChannel a rdf:Class;
129 rdfs:label "Priority Channel".
130
131 impc:Selectivity a cap:Capability;
```

```
cap:hasActionVerb imp:SelectivityProcessing;
impc:hasCargo impc:Cargo;
impc:hasCHKType impc:TypeOfCheck.

impc:TypeOfCheck a cap:EnumerationValue;
cap:hasElement impc:RedCheck;
cap:hasElement impc:GreenCheck.

impc:RedCheck a rdf:Class;
rdfs:label "Red Check";
rdfs:comment "This type of check means that a detailed examination of cargo is
    necessary.".

impc:GreenCheck a rdf:Class;
rdfs:label "GreenCheck";
rdfs:comment "This type of check means that a simple examination of cargo needs
    to be performed. By simple emanation we mean for example an X-Ray scan of the
     cargo.".

impc:DetExamination a cap:Capability;
cap:hasActionVerb imp:DetailedExamination;
impc:hasCargo impc:Cargo;
impc:hasExamType impc:DetailedExam.

impc:DetailedExam a rdf:Class;
rdfs:label "Detailed Examination";
rdfs:comment "This indicates that the examination type is detailed which include
    physical inspection, lab analysis etc.".

impc:ScanExamination a cap:Capability;
cap:hasActionVerb imp:ScanXRay;
impc:hasCargo impc:Cargo;
impc:hasExamType impc:ScanExam.

impc:ScanExam a rdf:Class;
rdfs:label "Scan X-Ray Examination";
rdfs:comment "This indicates that the examination type is scan X-Ray inspection
    .".

impc:ExamOfCargo a cap:Capability;
cap:hasActionVerb imp:ExaminationOfCargo;
impc:hasCargo impc:Cargo;
impc:hasDecision impc:ExamDecision;
impc:hasCHKType impc:TypeOfCheck;
impc:hasExamType impc:TypeOfExam.

impc:TypeOfExam a cap:EnumerationValue;
cap:hasElement impc:DetailedExam;
cap:hasElement impc:ScanExam.

impc:Investigation a cap:capability;
cap:hasActionVerb imp:DetainForInvestigation;
impc:hasCargo impc:Cargo.

impc:ReleaseCargo a cap:Capability;
```

```
182 cap:hasActionVerb imp:Release;
183 impc:hasCargo impc:Cargo.
184
185 impc:NotifyImporter a cap:Capability;
186 cap:hasActionVerb imp:Notification;
187 impc:hasMsg xsd:string.
188
189 impc:ReleaseAndNotify a cap:Capability;
190 cap:hasActionVerb imp:ReleaseAndNotification;
191 impc:hasCargo impc:Cargo;
192 impc:hasMsg xsd:string.
193
194 impc:ExamAndRelease a cap:Capability;
195 cap:hasActionVerb imp:ExaminationAndReleaseDecision;
196 impc:hasCargo impc:Cargo;
197 impc:hasDecision impc:ExamDecision;
198 impc:hasCHKType impc:TypeOfCheck;
199 impc:hasExamType impc:TypeOfExam;
200 impc:hasMsg xsd:string.
201
202 impc:Import a cap:Capability;
203 cap:hasActionVerb imp:Importation;
204 impc:hasDocuments impc:Documents;
205 impc:hasCargo impc:Cargo;
206 impc:hasFee impc:Fees;
207 impc:hasDecision impc:ExamDecision;
208 impc:hasCHKType impc:TypeOfCheck;
209 impc:hasExamType impc:TypeOfExam;
210 impc:hasMsg xsd:string.
```

Listing D.1: RDF N3 representation of the Import Capability Domain Ontology: IMPC

Appendix E

Business Process Models for Municipalities

This appendix contains the business process models used in the evaluation of Chapter 6. These models have been manually created from those used by Gottschalk in his thesis [89]. They were subject of a case study [92] in which techniques for managing configurable process models were extensively tested in a real-world scenario. The process models used in this case study are four processes out of the five most executed registration processes in the civil affairs department of Dutch municipalities [89]:

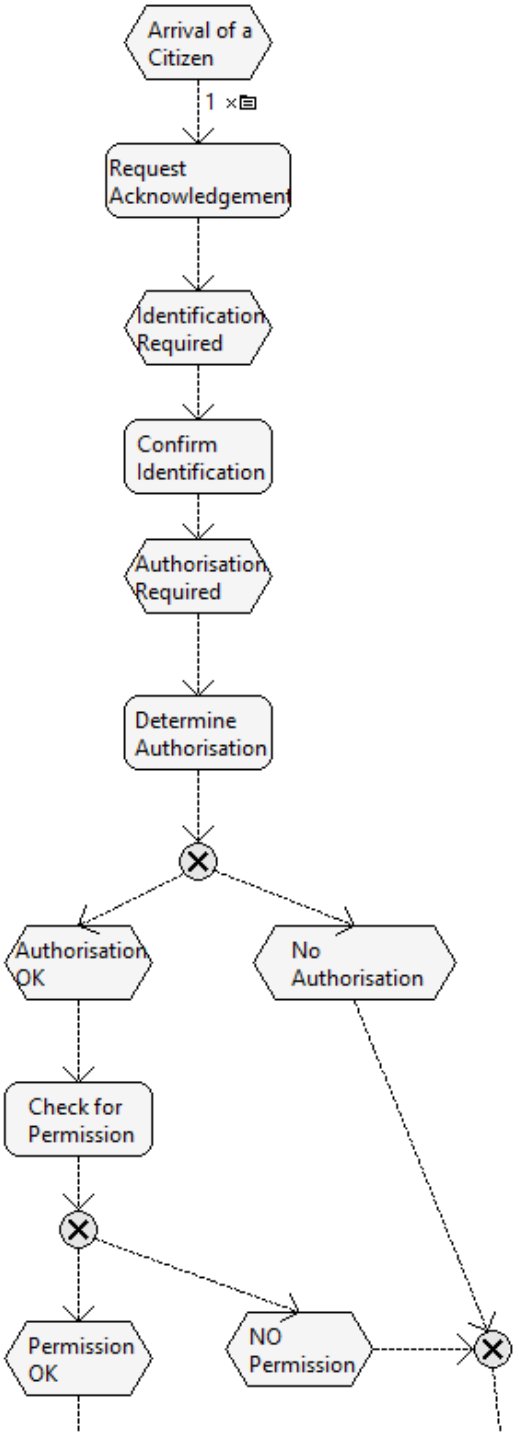
- *P1: Acknowledging an unborn child*: This process is executed when a man wants to register that he is the father of an unborn child in case he is not married to his pregnant partner.
- *P2: Registering a newborn*: This process describes the steps for registering a newborn and get his birth certificate.
- *P3: Marriage*: This process describes all the steps required before getting married in a Dutch municipality.
- *P4: Decease*: This process describes the steps required by relatives to burry the deceased and get a death certificate.

The process variants listed here were initially available in Protos[1]. Each process has five process variants. Consequently, a total of 5 x 4 = 20 process models were considered in this work (similar to the case study [92]). These models have been manually translated into EPC.

[1]Protos is part of Pallas Athena's BPM toolset BPM|one.

E.1 Acknowledging an Unborn Child

(A) Acknowledging an Unborn Child - Variant 1 - Part 1/2



(B) Acknowledging an Unborn Child - Variant 1 - Part 2/2

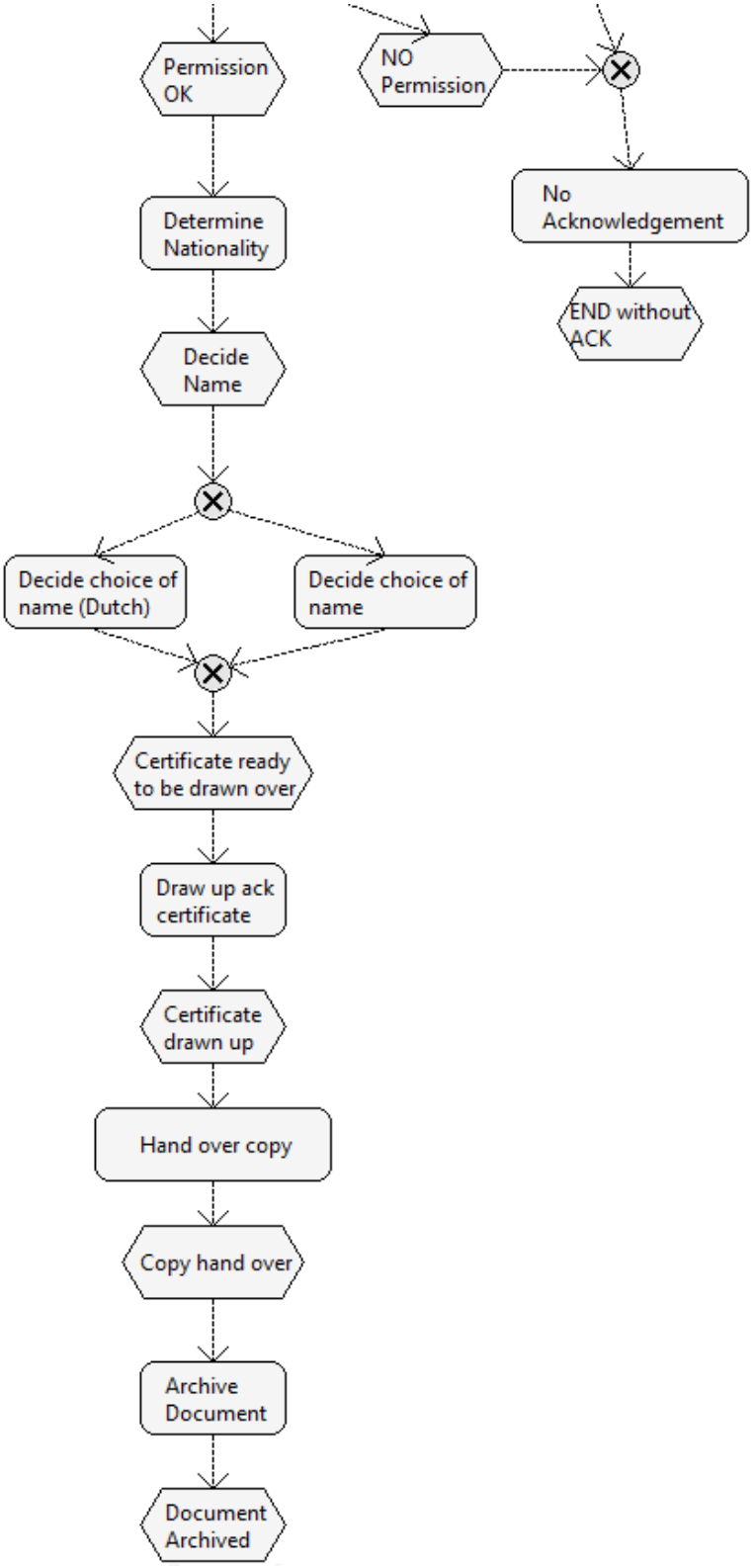
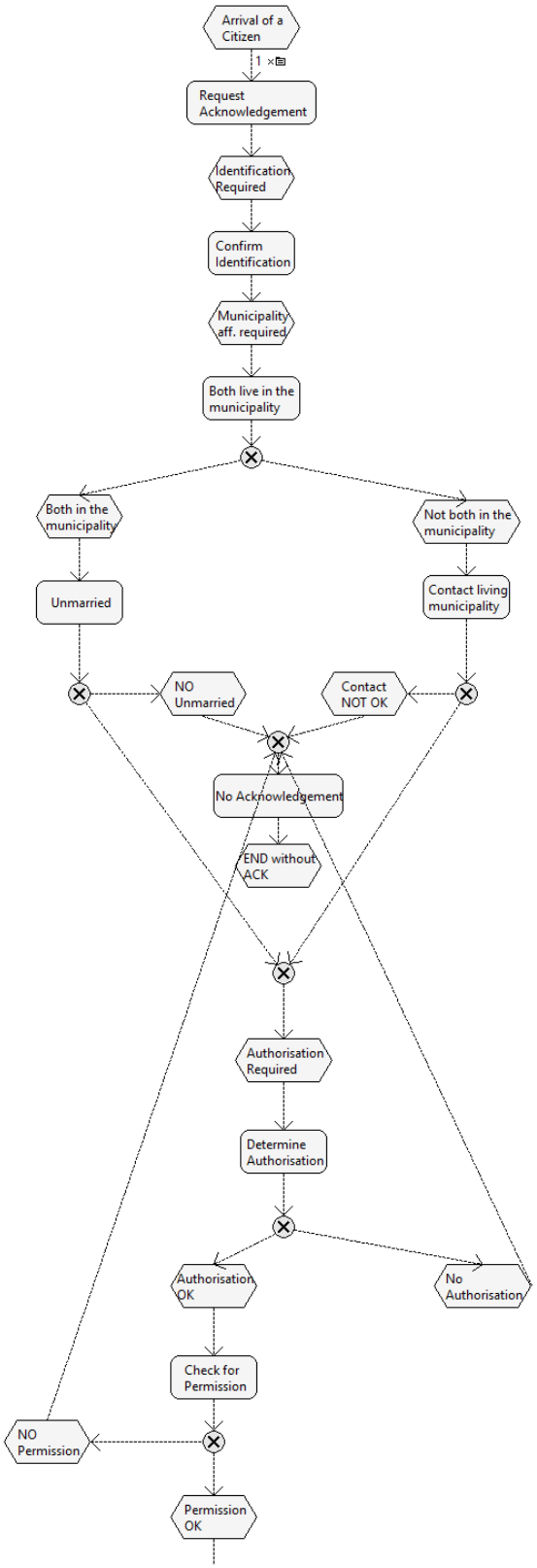


FIGURE E.1: Acknowledging an Unborn Child - Variant 1

(A) Acknowledging an Unborn Child - Variant 2 - Part 1/2



(B) Acknowledging an Unborn Child - Variant 2 - Part 2/2

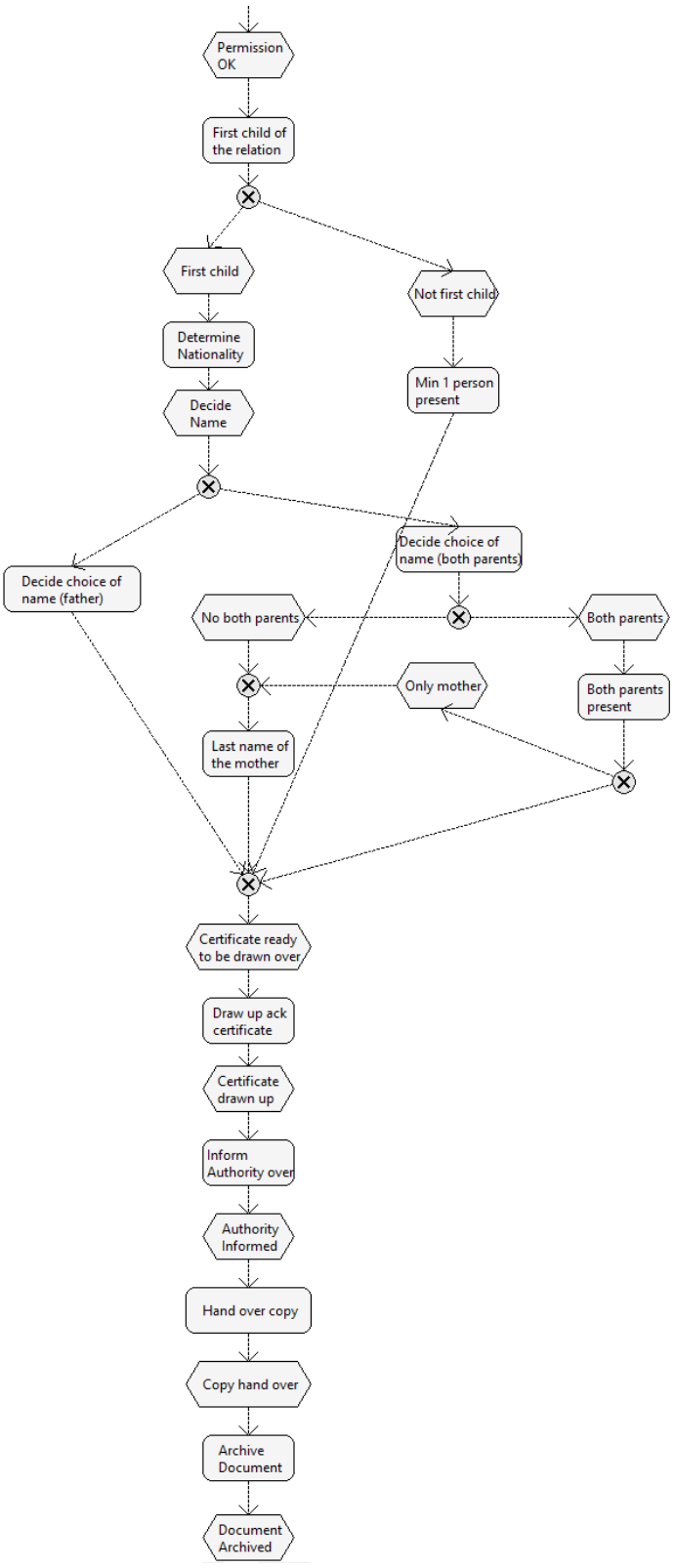
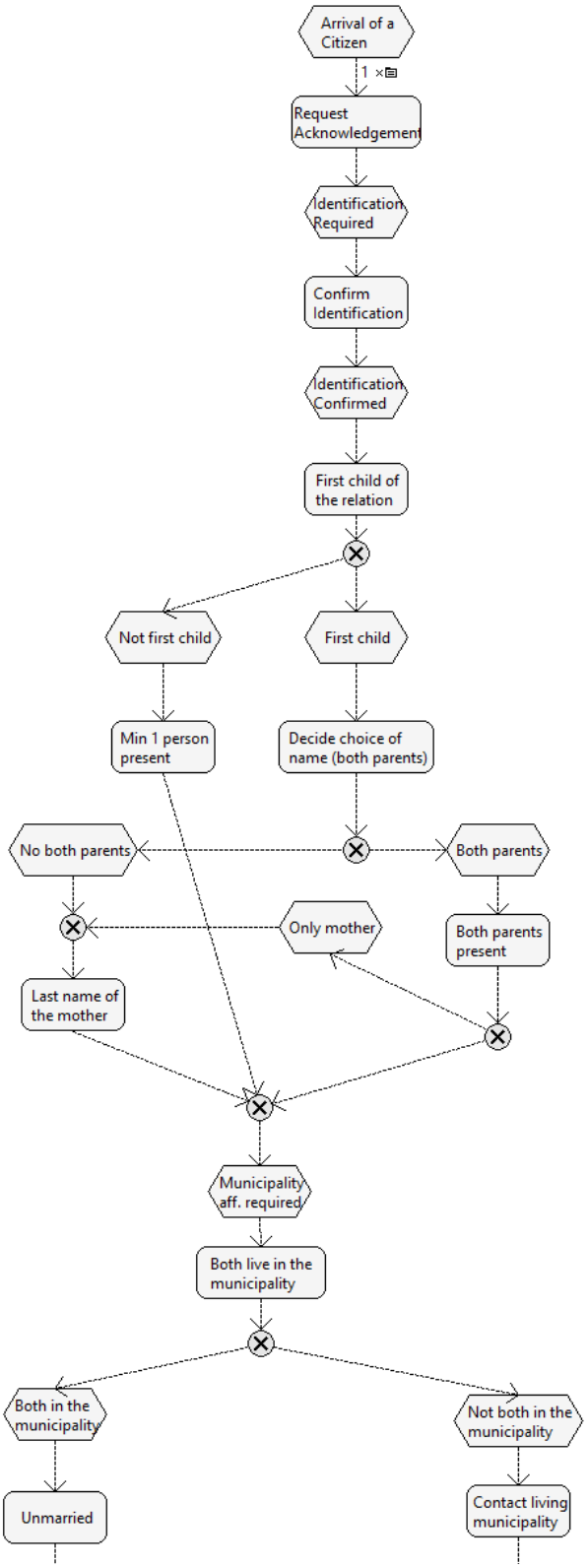


FIGURE E.2: Acknowledging an Unborn Child - Variant 2

(A) Acknowledging an Unborn Child - Variant 3 - Part 1/2



(B) Acknowledging an Unborn Child - Variant 3 - Part 2/2

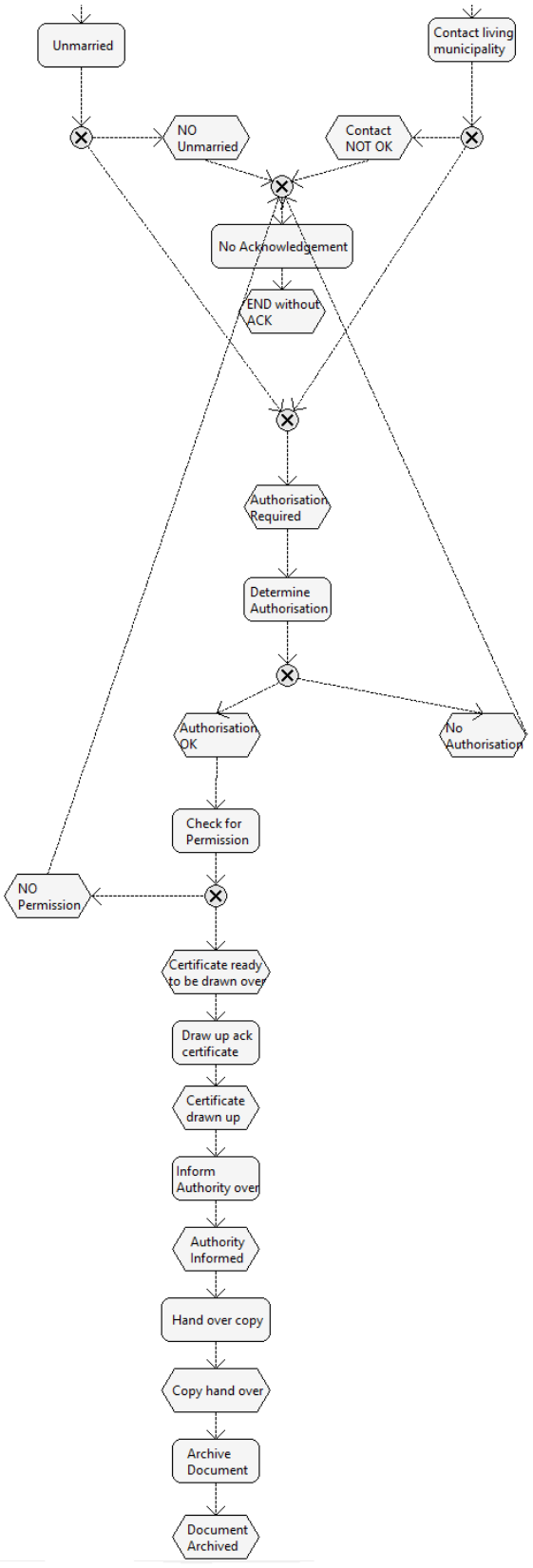


FIGURE E.3: Acknowledging an Unborn Child - Variant 3

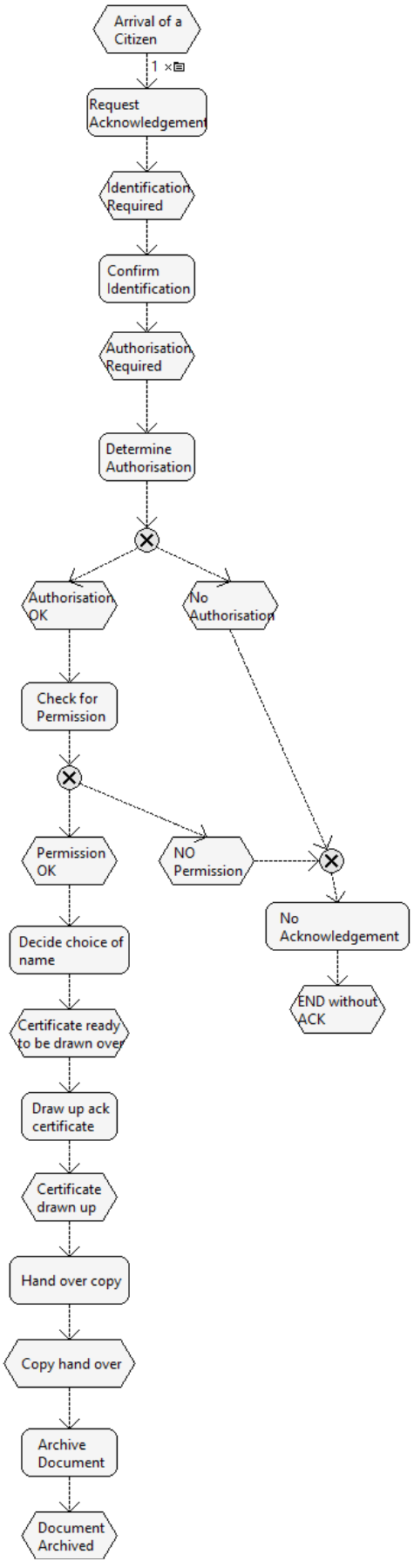


FIGURE E.4: Acknowledging an Unborn Child - Variant 4

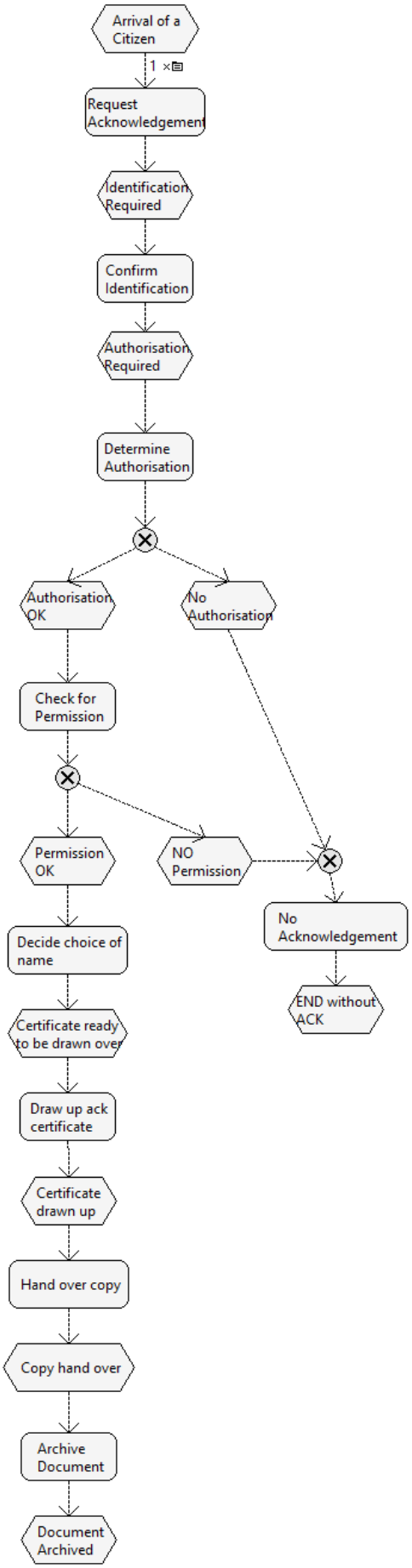
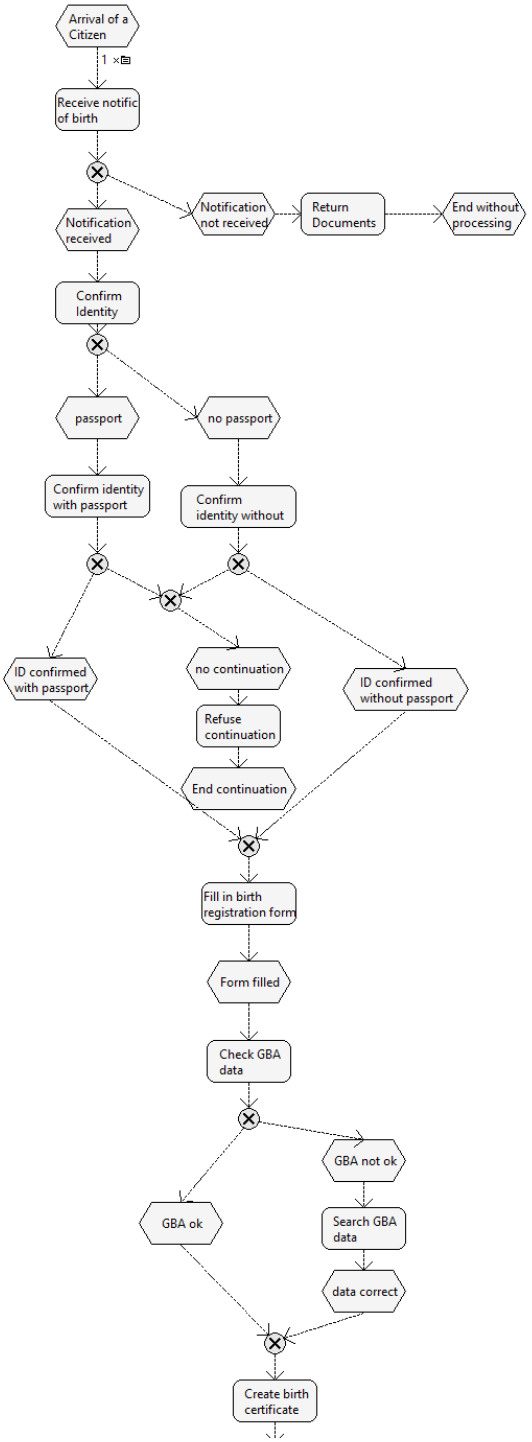


FIGURE E.5: Acknowledging an Unborn Child - Variant 5

E.2 Registering a Newborn

(A) Registering a Newborn - Variant 1 - Part 1/2



(B) Registering a Newborn - Variant 1 - Part 2/2

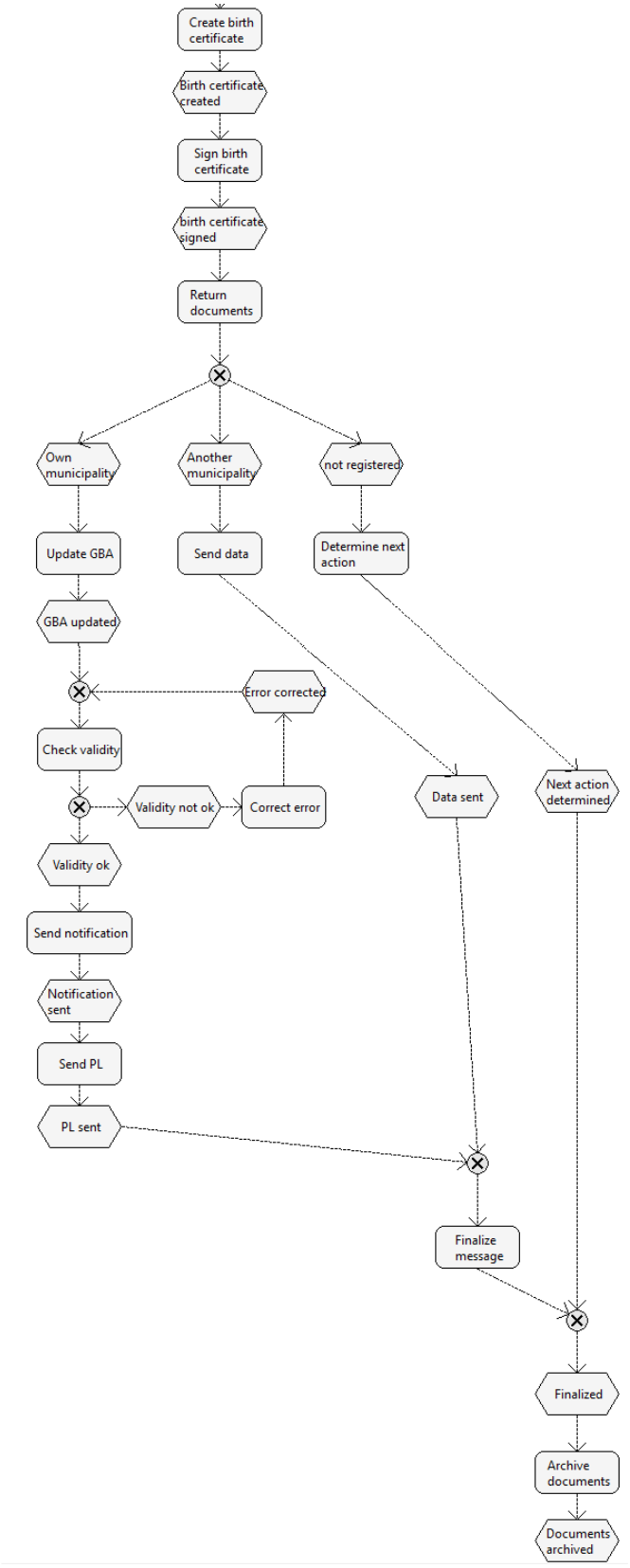
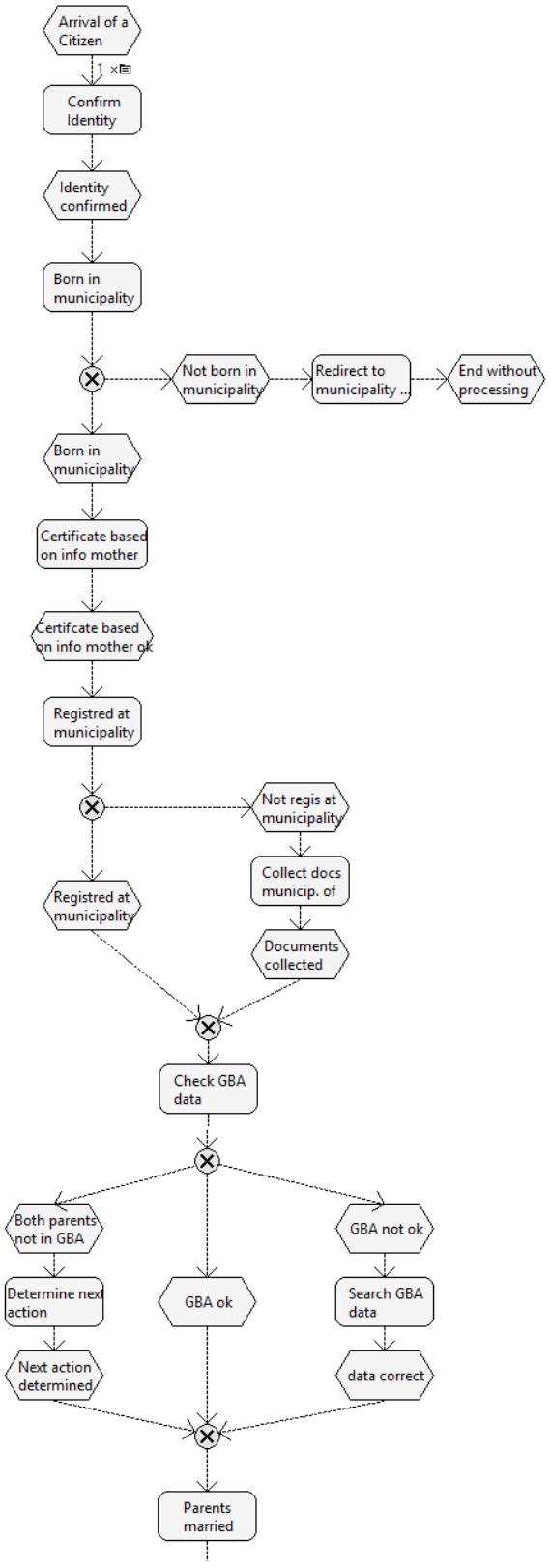
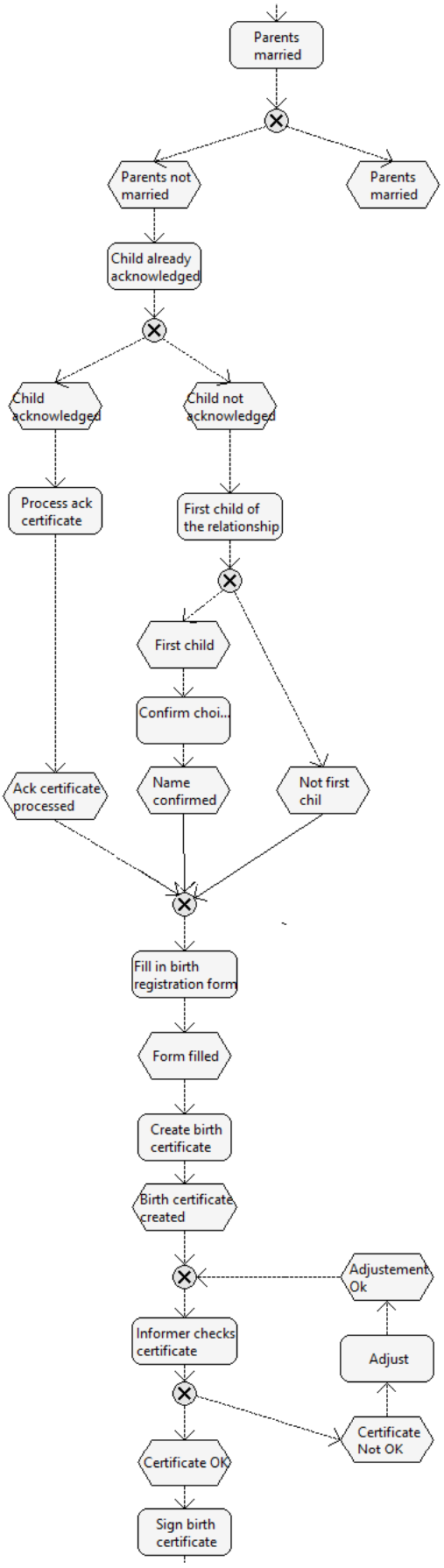


FIGURE E.6: Registering a Newborn - Variant 1

(A) Registering a Newborn - Variant 2 - Part 1/3



(B) Registering a Newborn - Variant 2 - Part 2/3



(c) Registering a Newborn - Variant 2 - Part 3/3

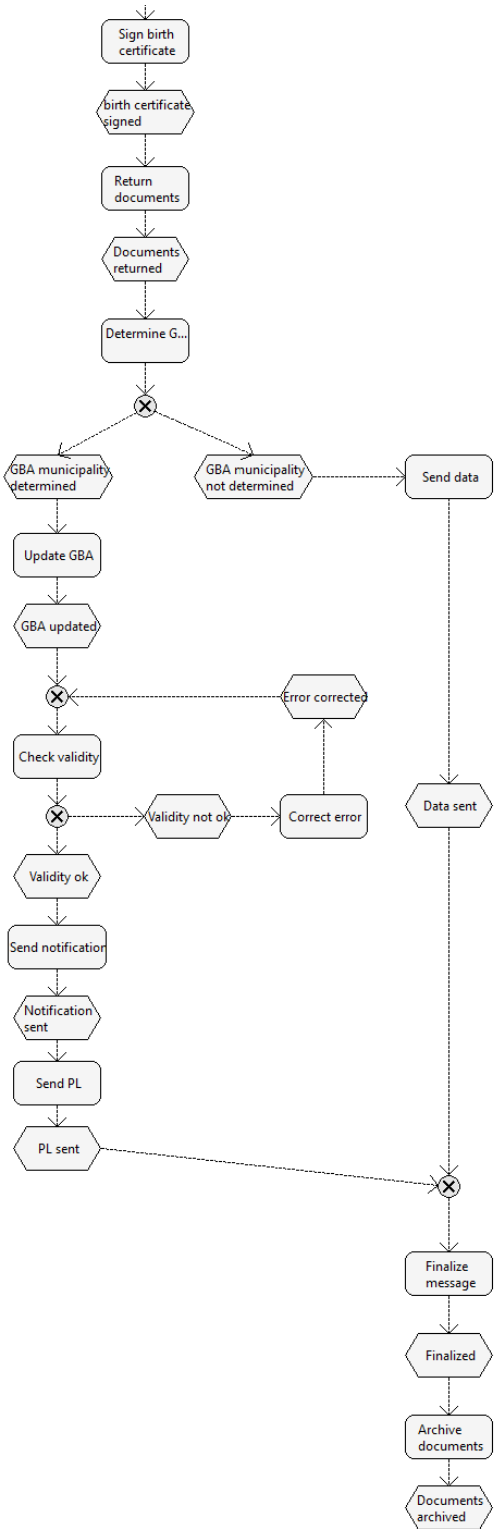
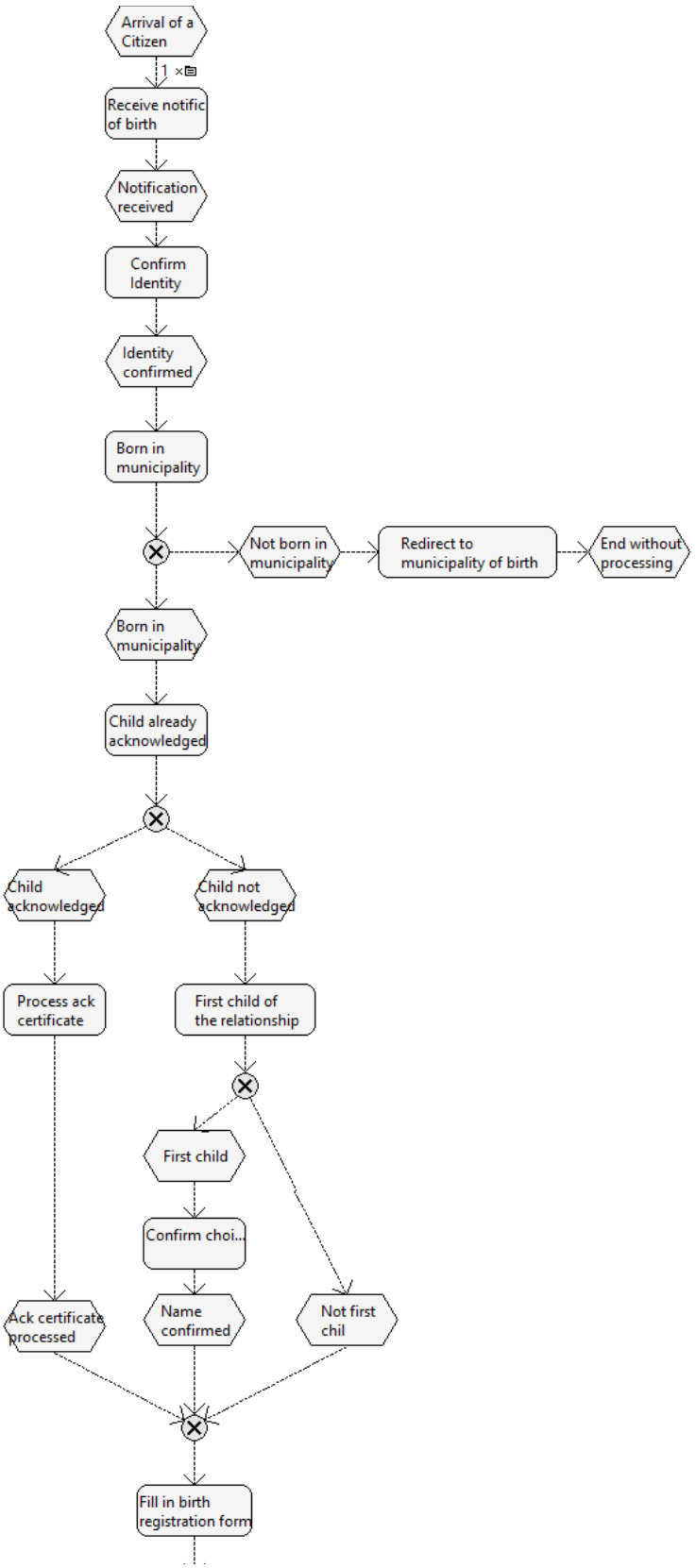
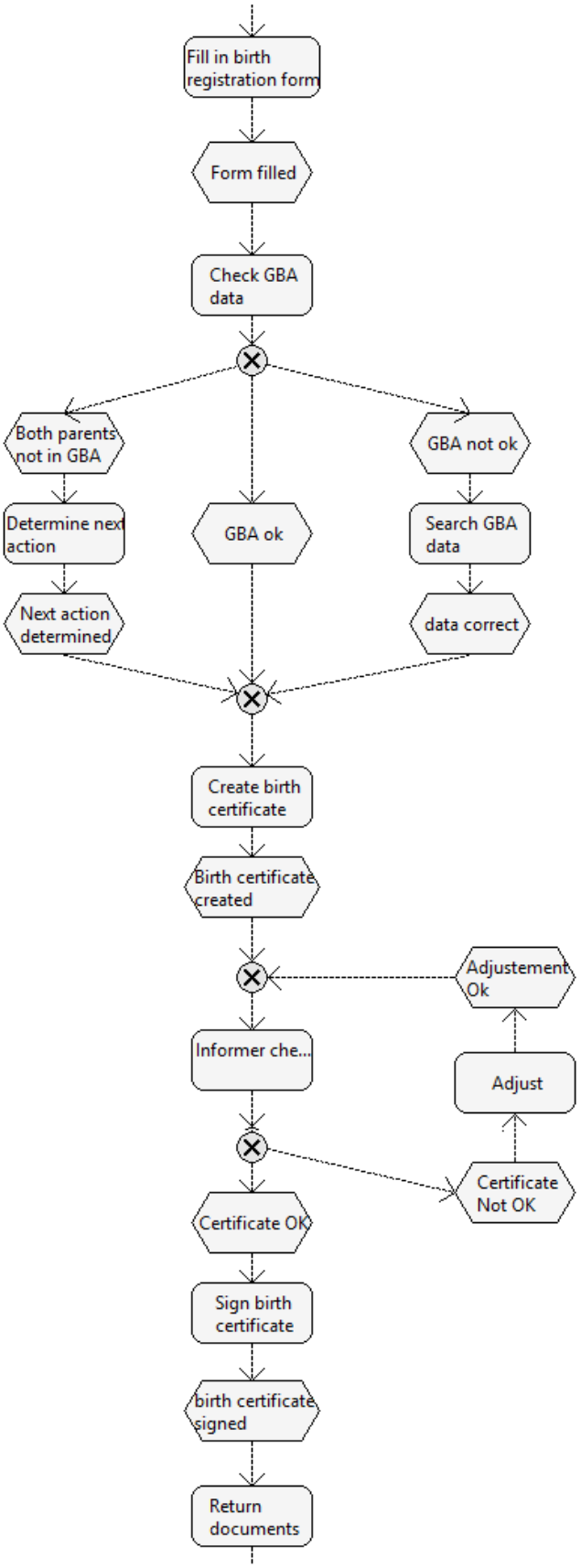


FIGURE E.7: Registering a Newborn - Variant 2

(A) Registering a Newborn - Variant 3 - Part 1/3



(B) Registering a Newborn - Variant 3 - Part 2/3



(c) Registering a Newborn - Variant 3 - Part 3/3

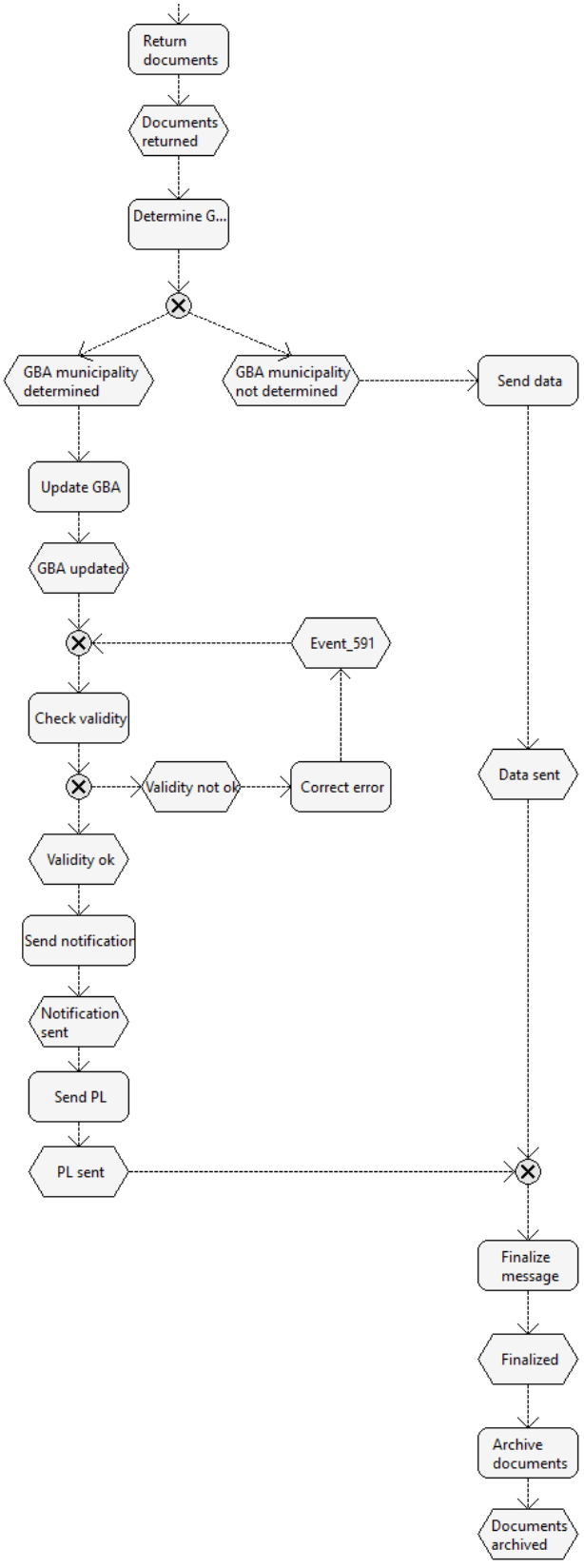
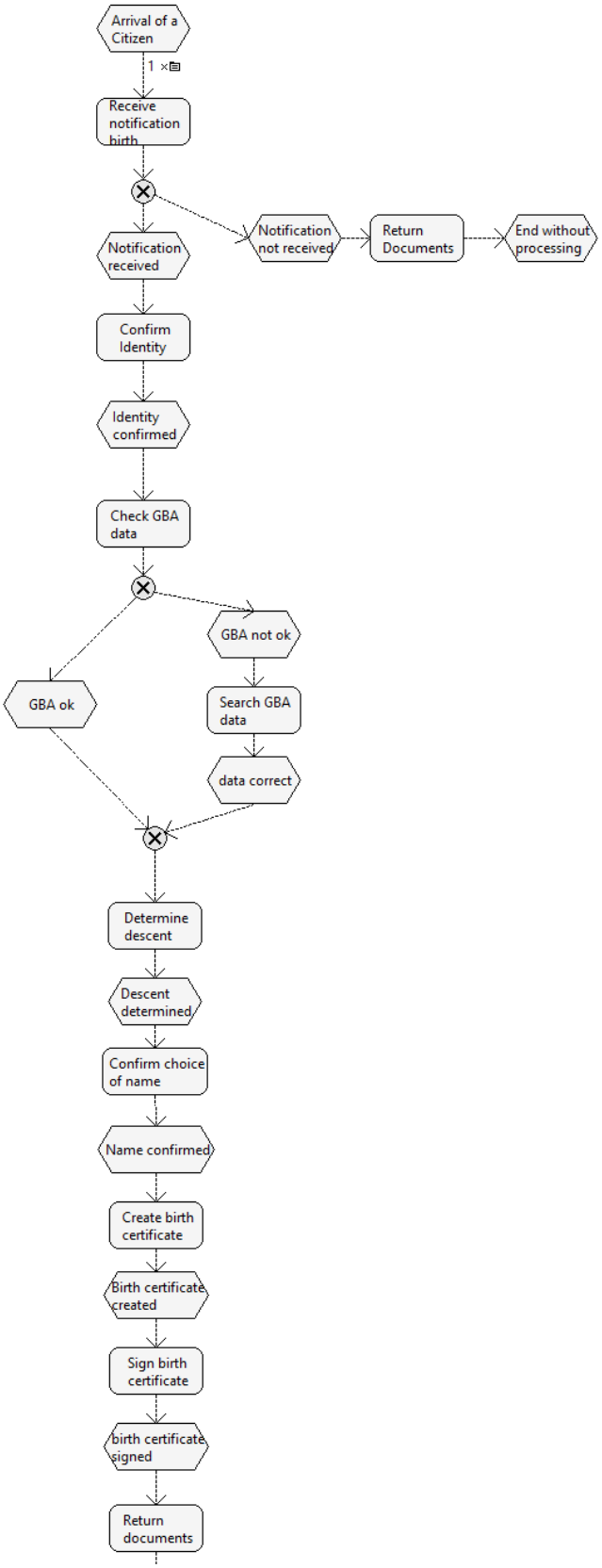


FIGURE E.8: Registering a Newborn - Variant 3

(A) Registering a Newborn - Variant 4 - Part 1/2



(B) Registering a Newborn - Variant 4 - Part 2/2

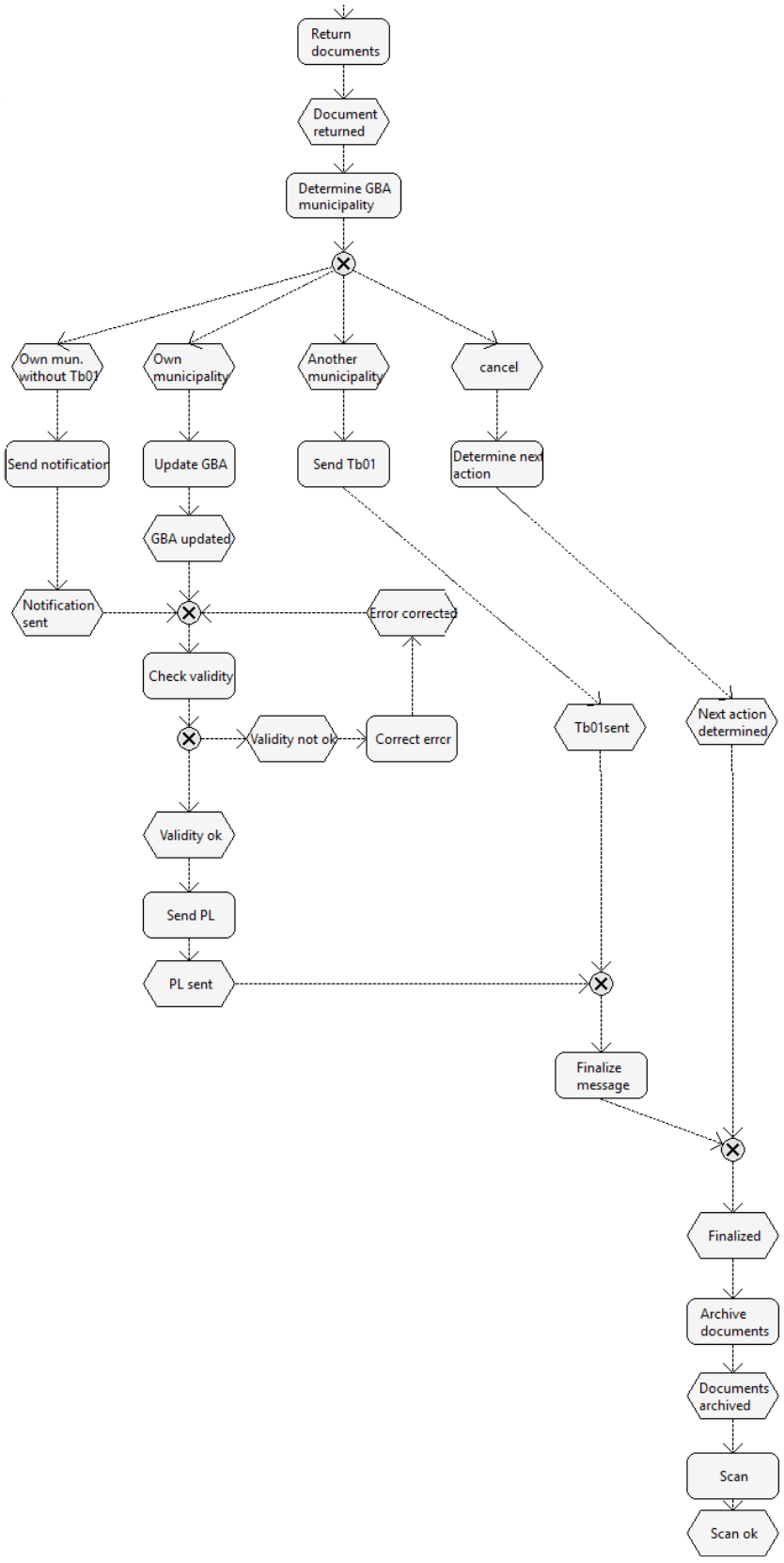
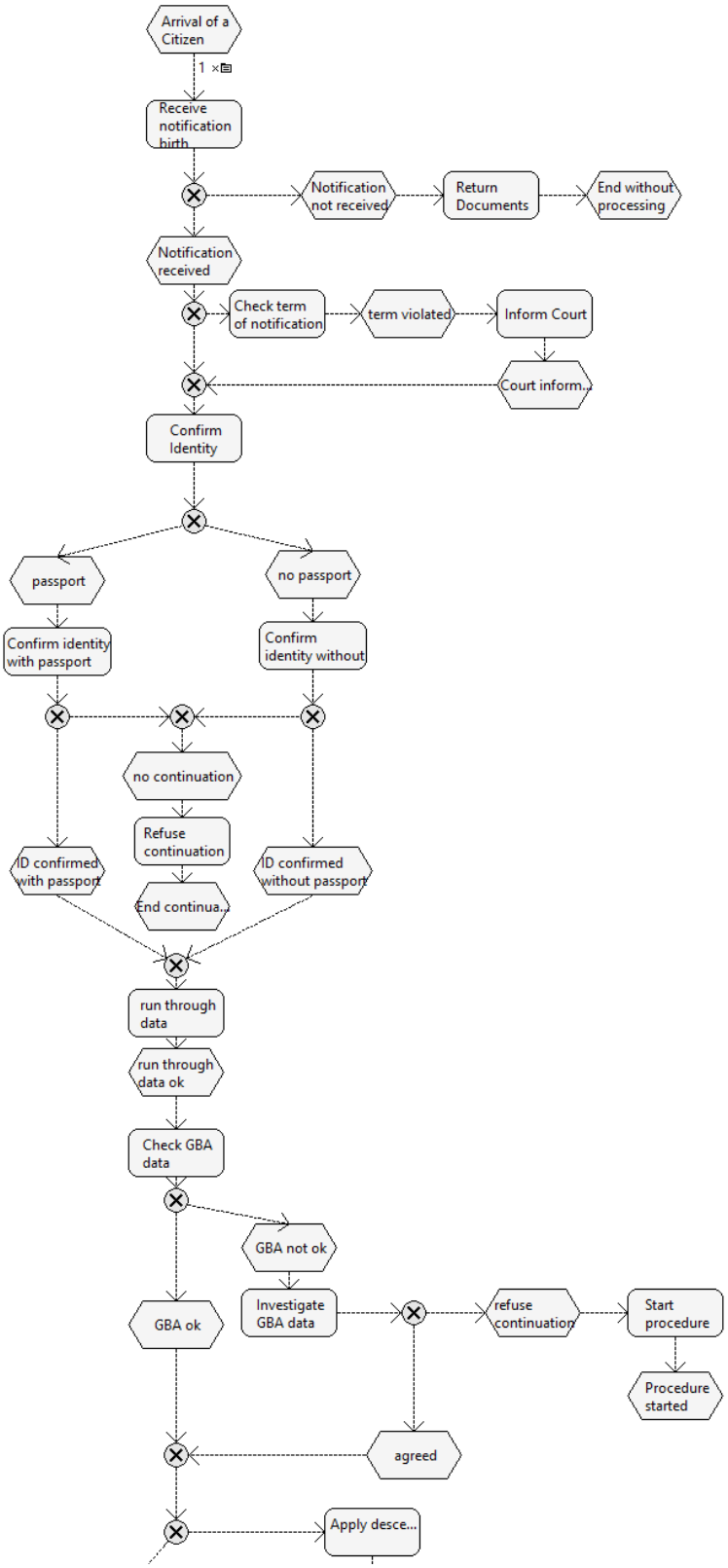


FIGURE E.9: Registering a Newborn - Variant 4

(A) Registering a Newborn - Variant 5 - Part 1/2



(B) Registering a Newborn - Variant 5 - Part 2/2

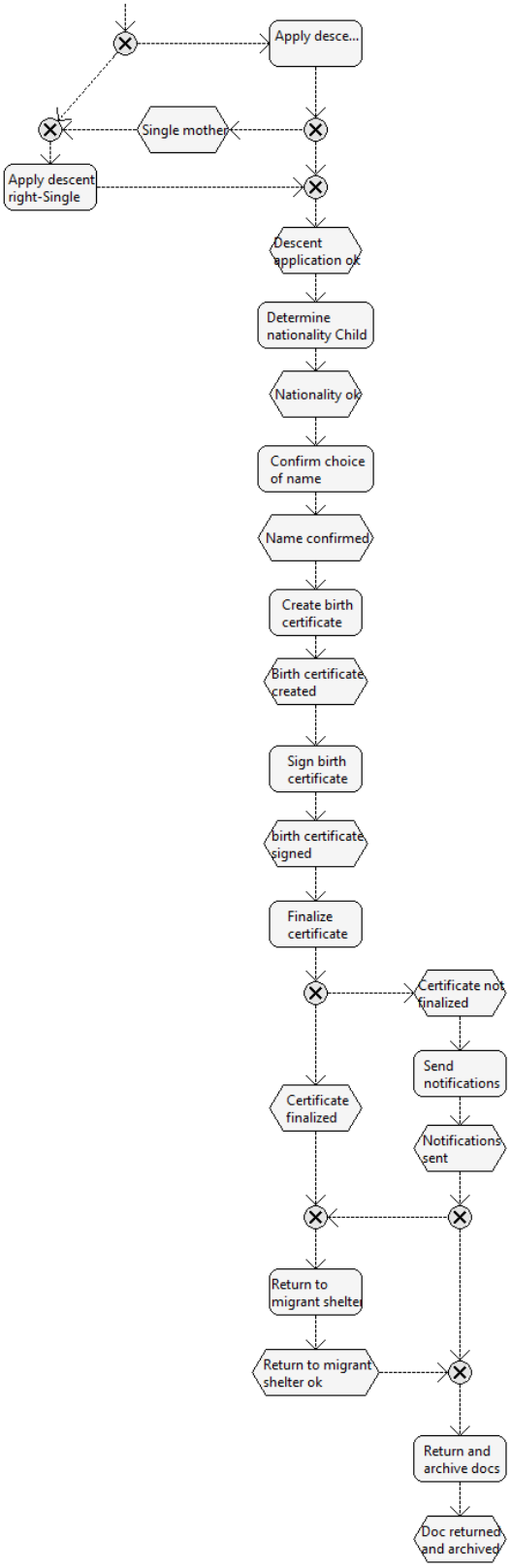
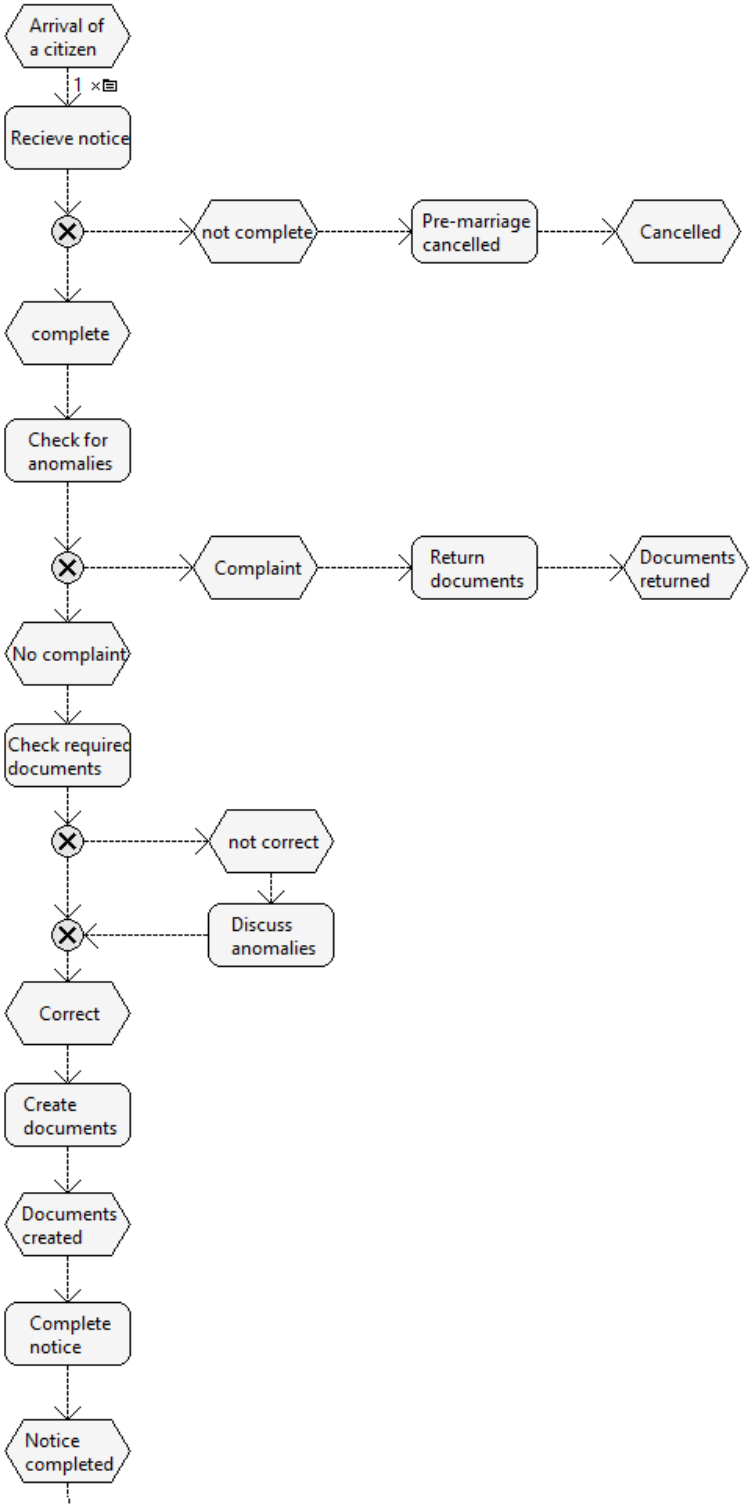


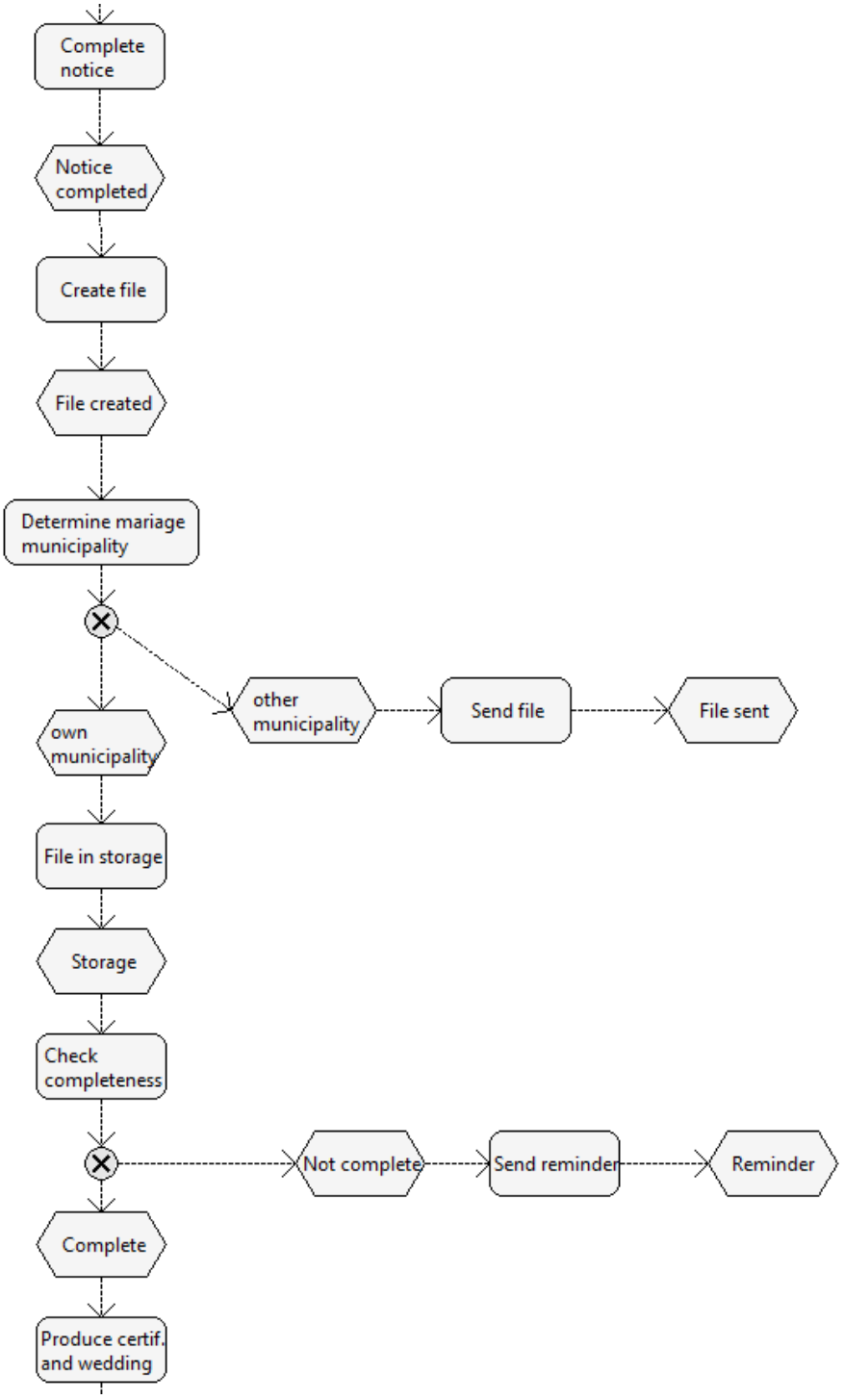
FIGURE E.10: Registering a Newborn - Variant 5

E.3 Mariage

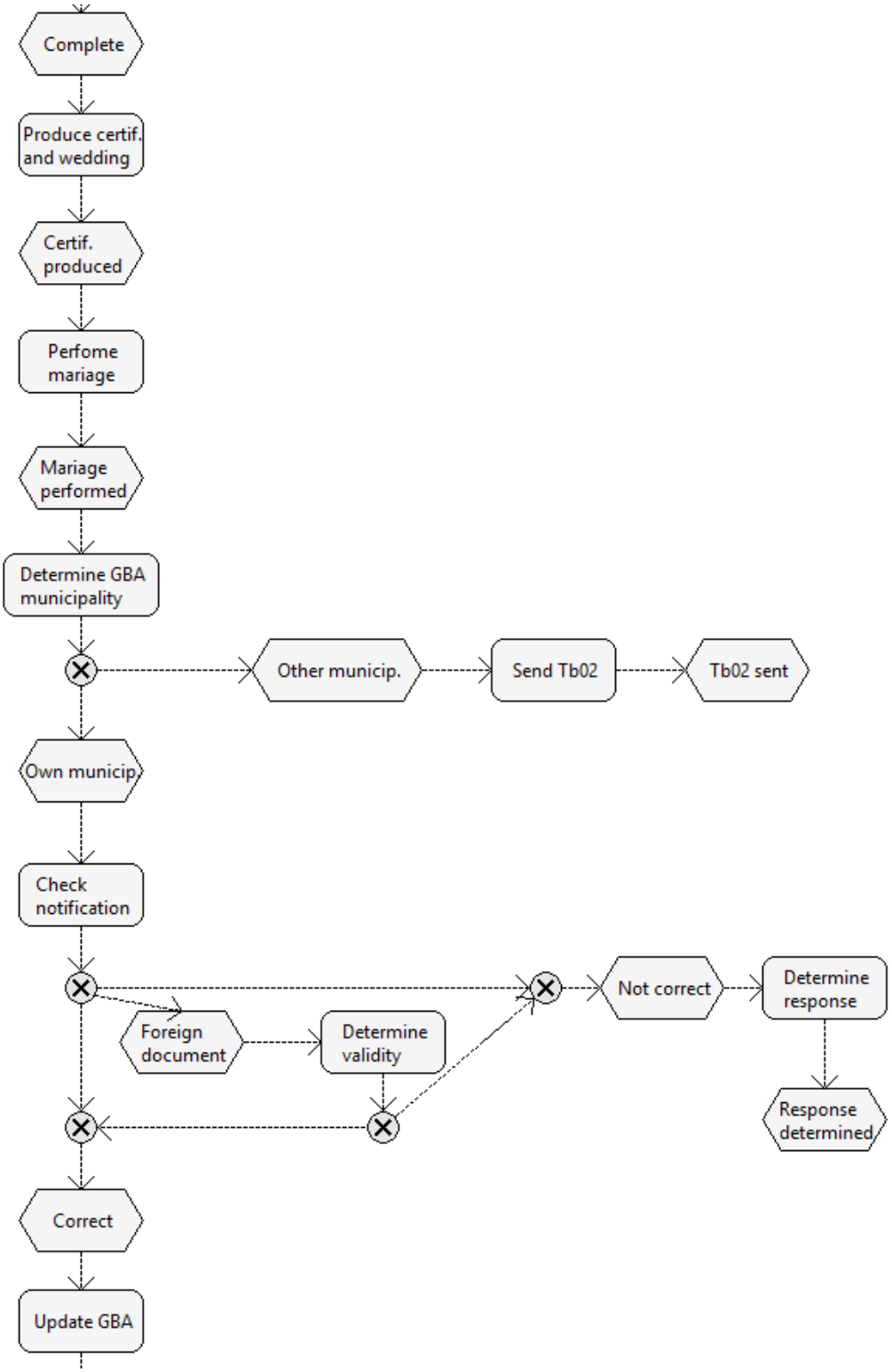
(A) Mariage- Variant 1 - Part 1/4



(B) Mariage- Variant 1 - Part 2/4



(c) Mariage- Variant 1 - Part 3/4



(D) Mariage- Variant 1 - Part 4/4

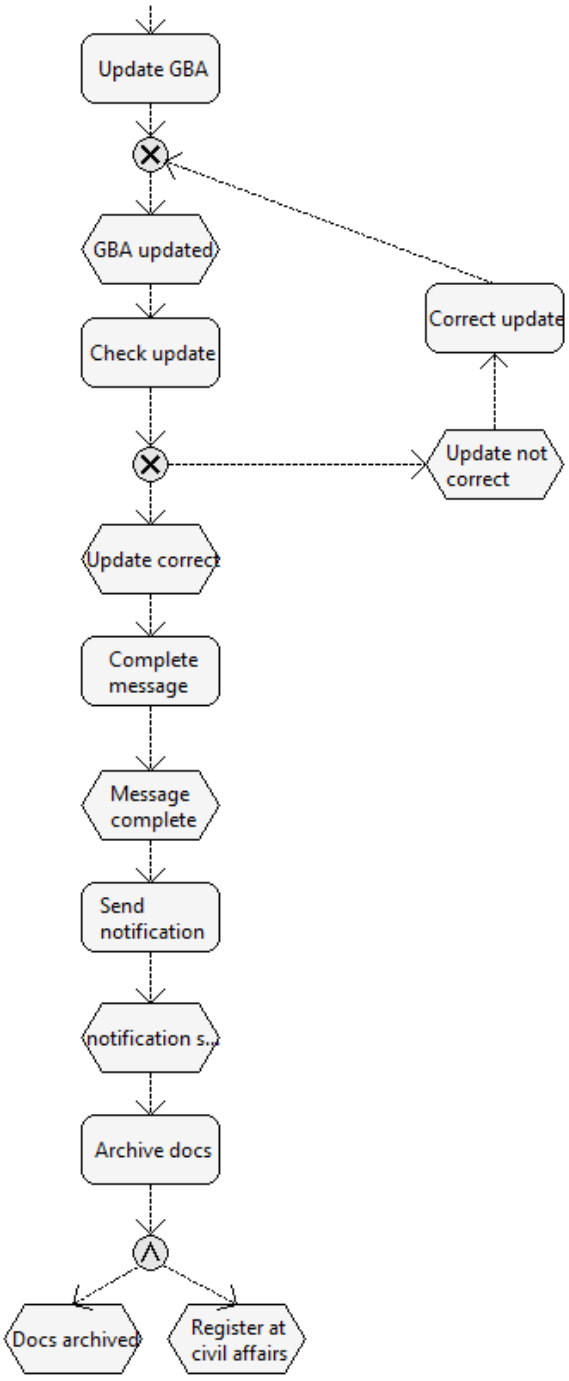
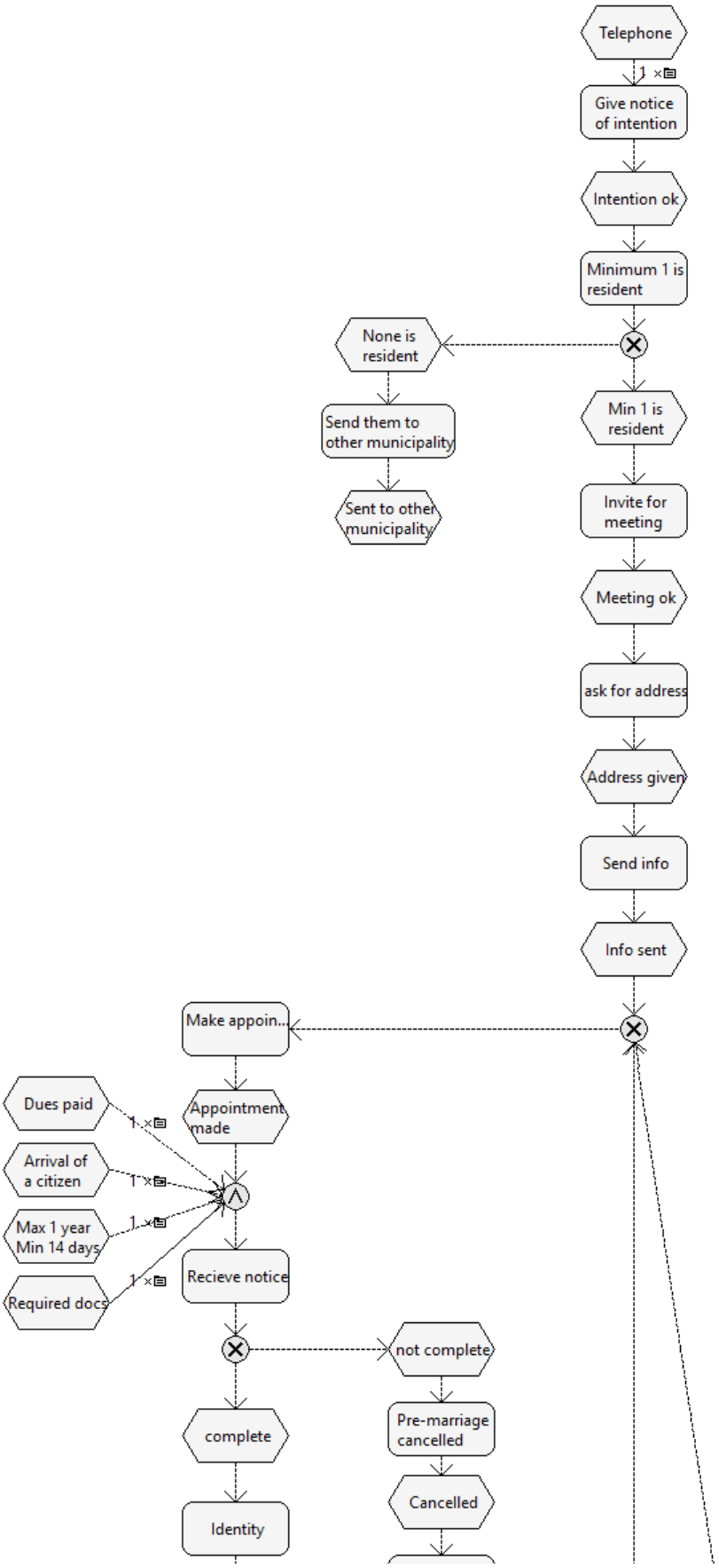
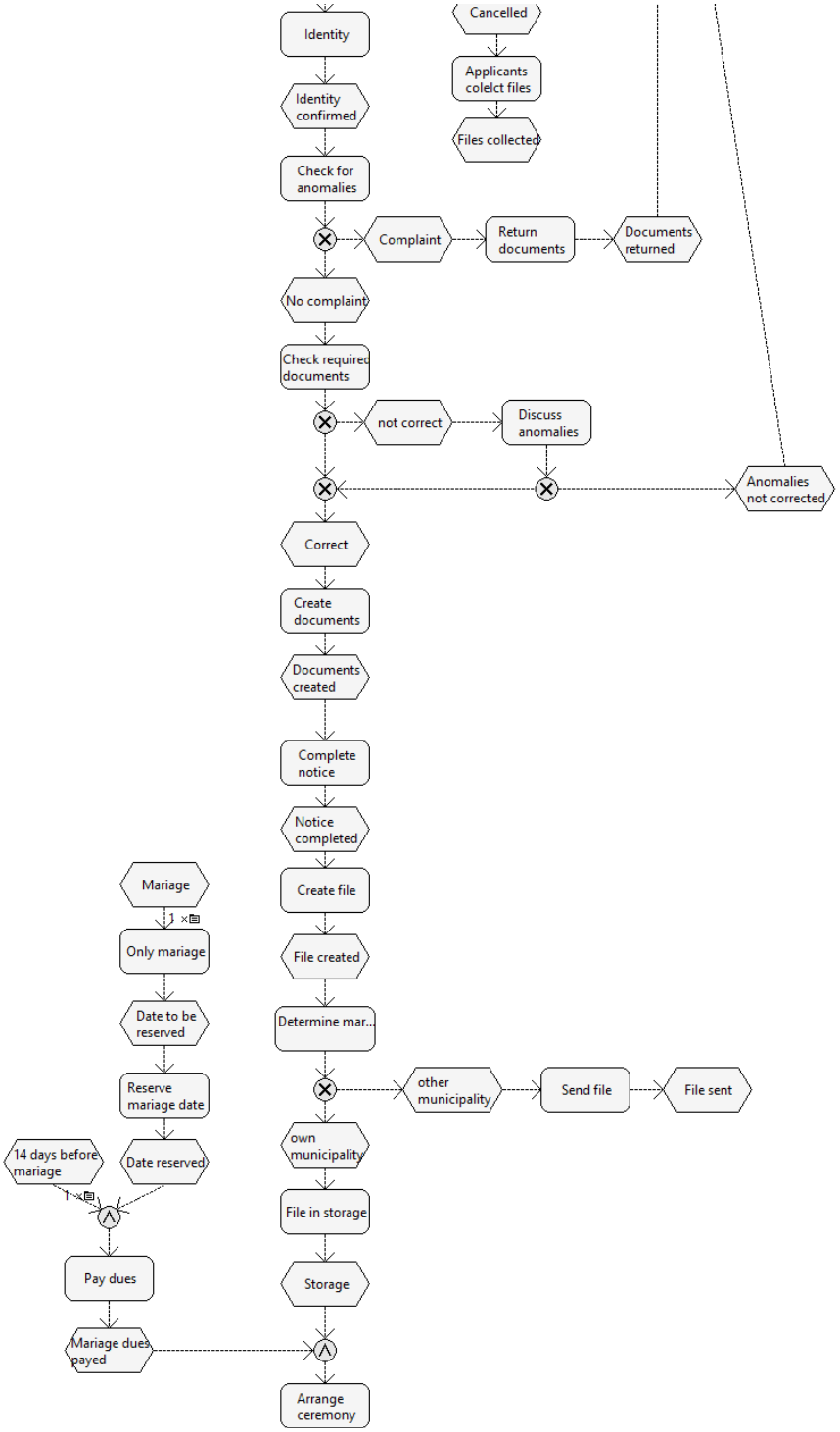


FIGURE E.11: Marriage - Variant 1

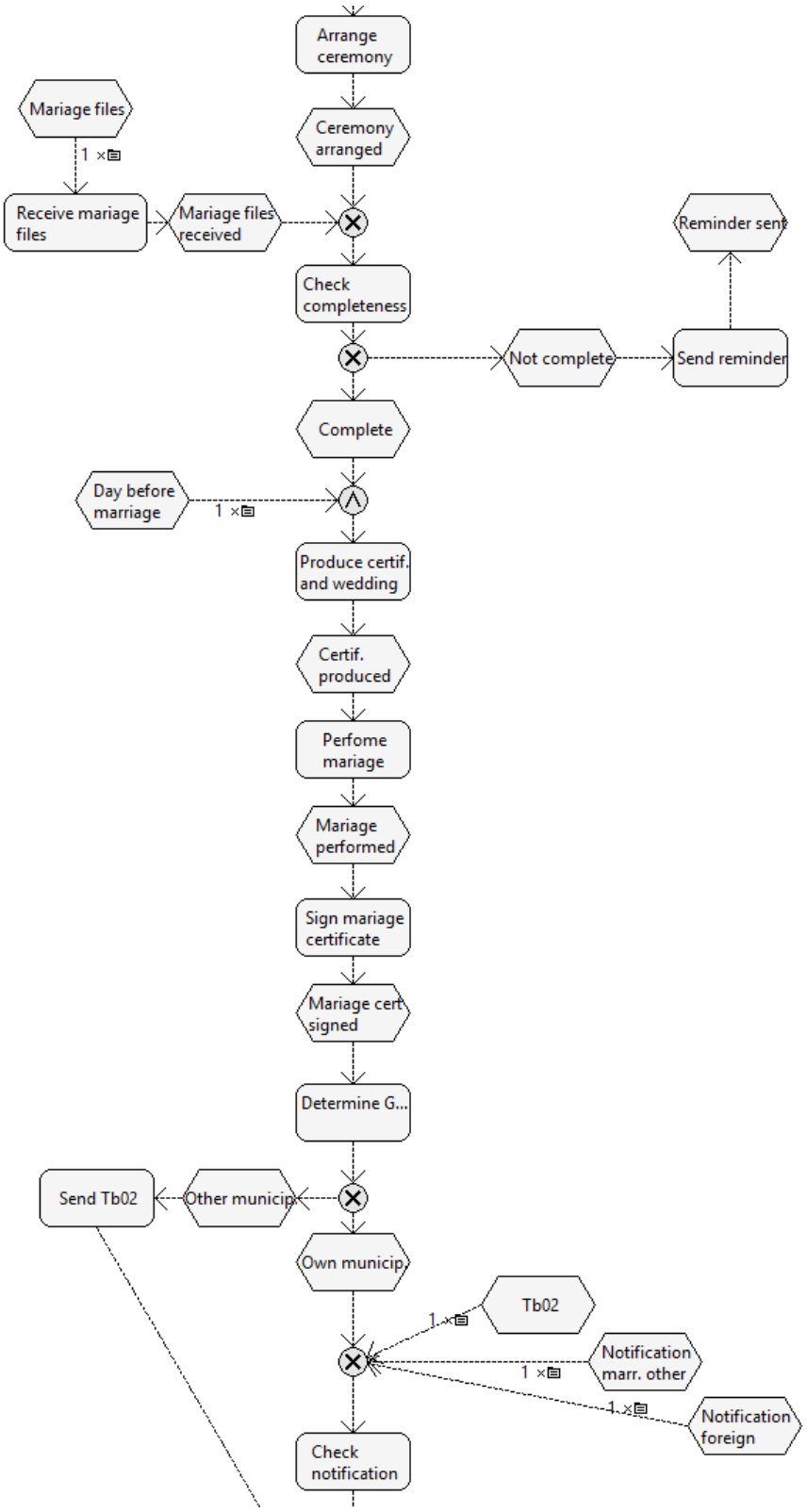
(A) Mariage- Variant 2 - Part 1/4



(B) Mariage- Variant 2 - Part 2/4



(c) Mariage- Variant 2 - Part 3/4



(D) Mariage- Variant 2 - Part 4/4

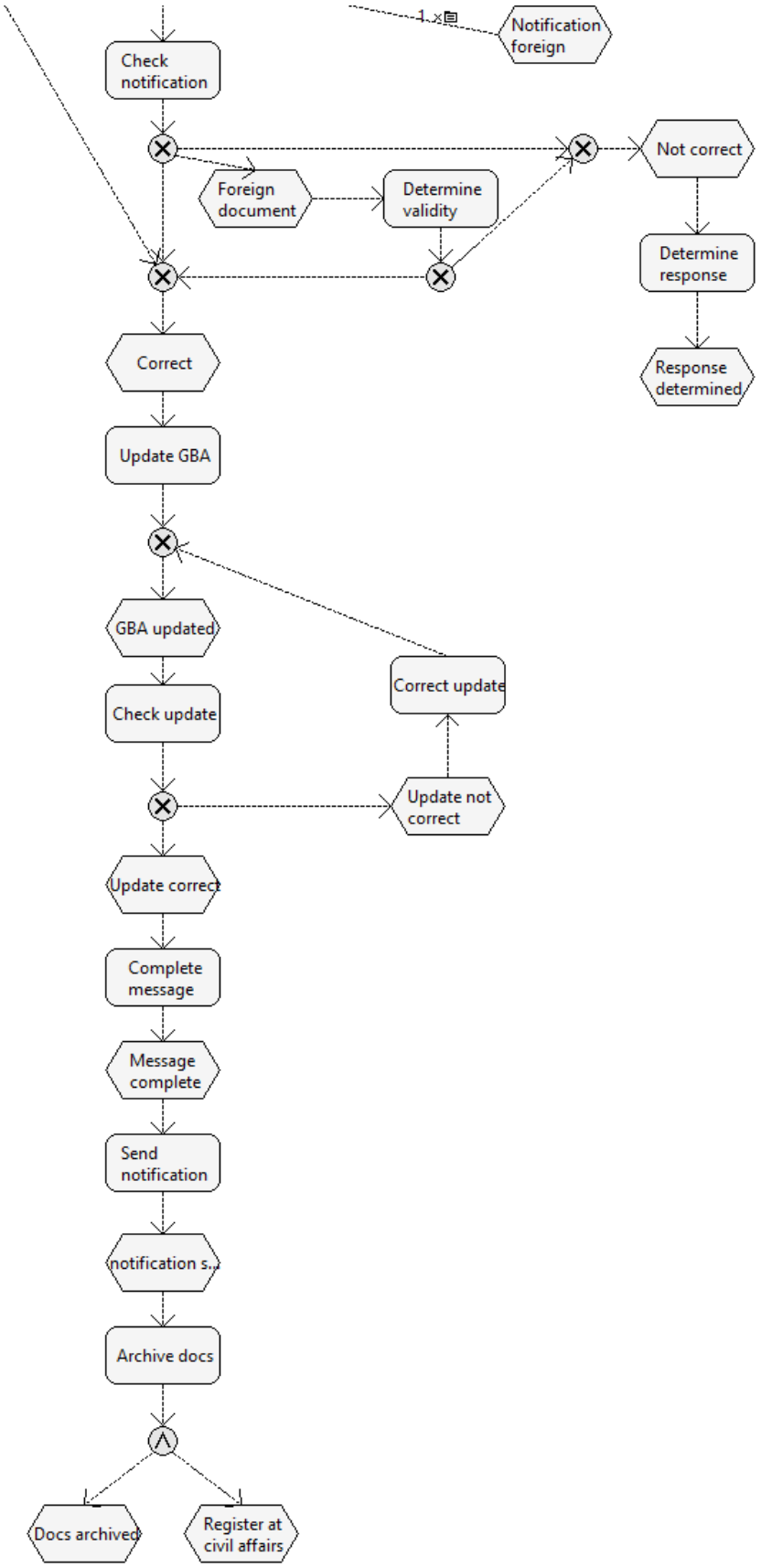
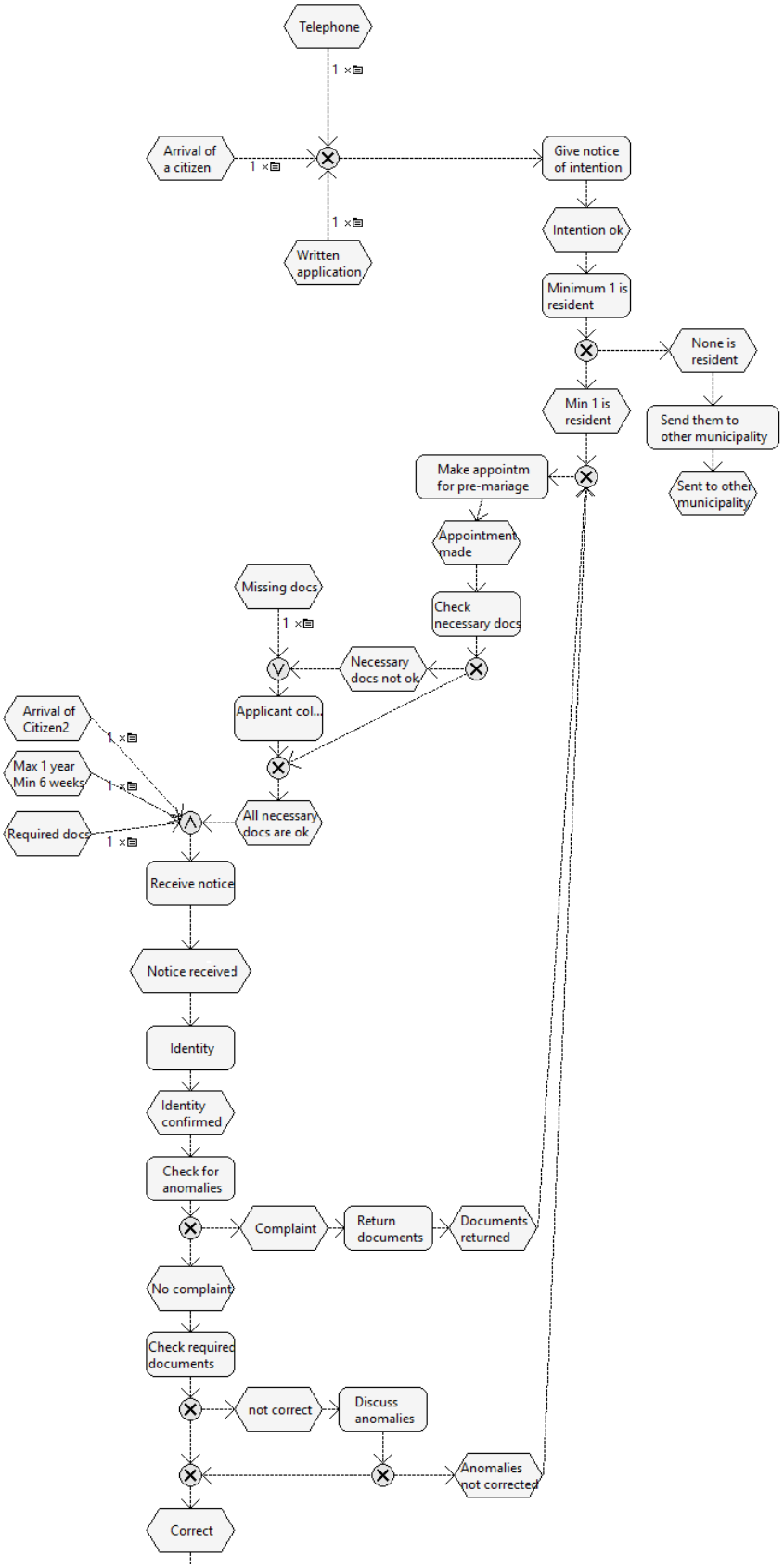
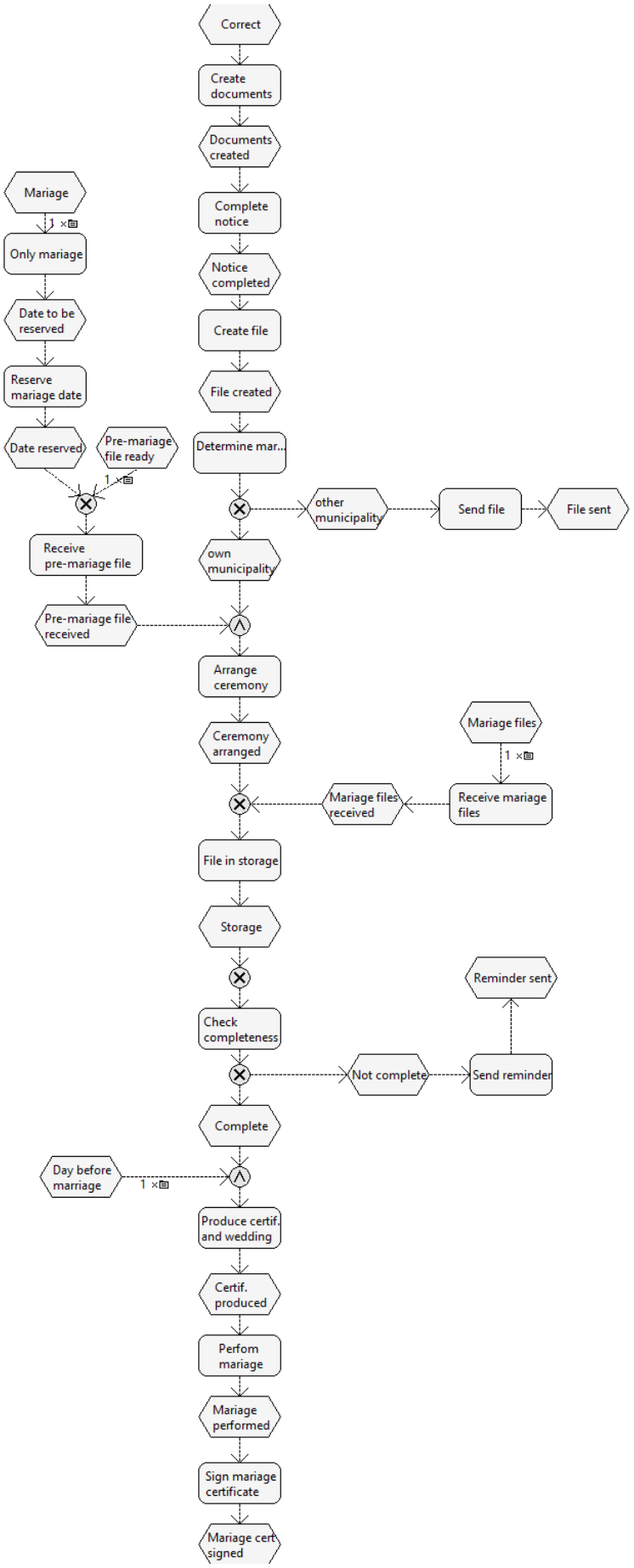


FIGURE E.12: Marriage - Variant 2

(A) Mariage- Variant 3 - Part 1/3



(B) Mariage- Variant 3 - Part 2/3



(c) Mariage- Variant 3 - Part 3/3

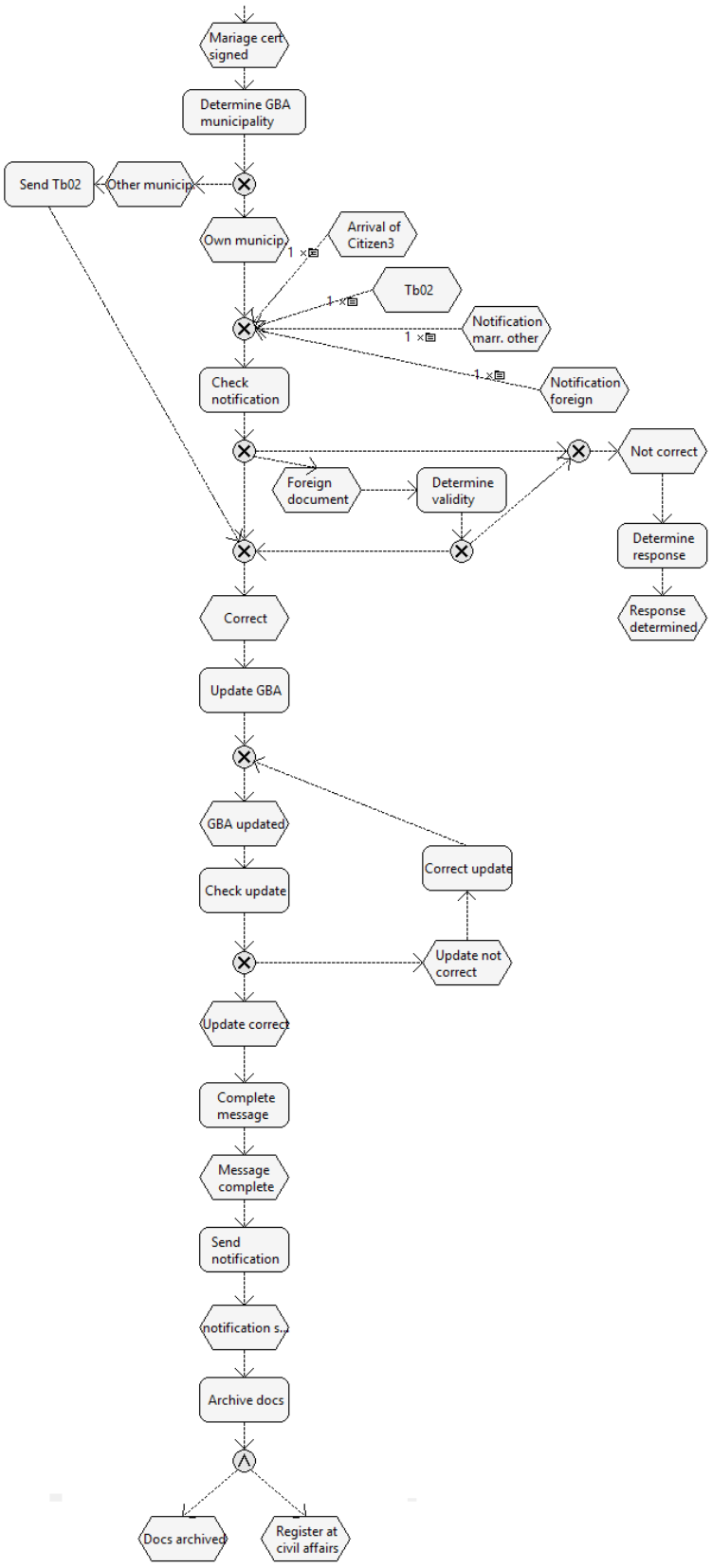
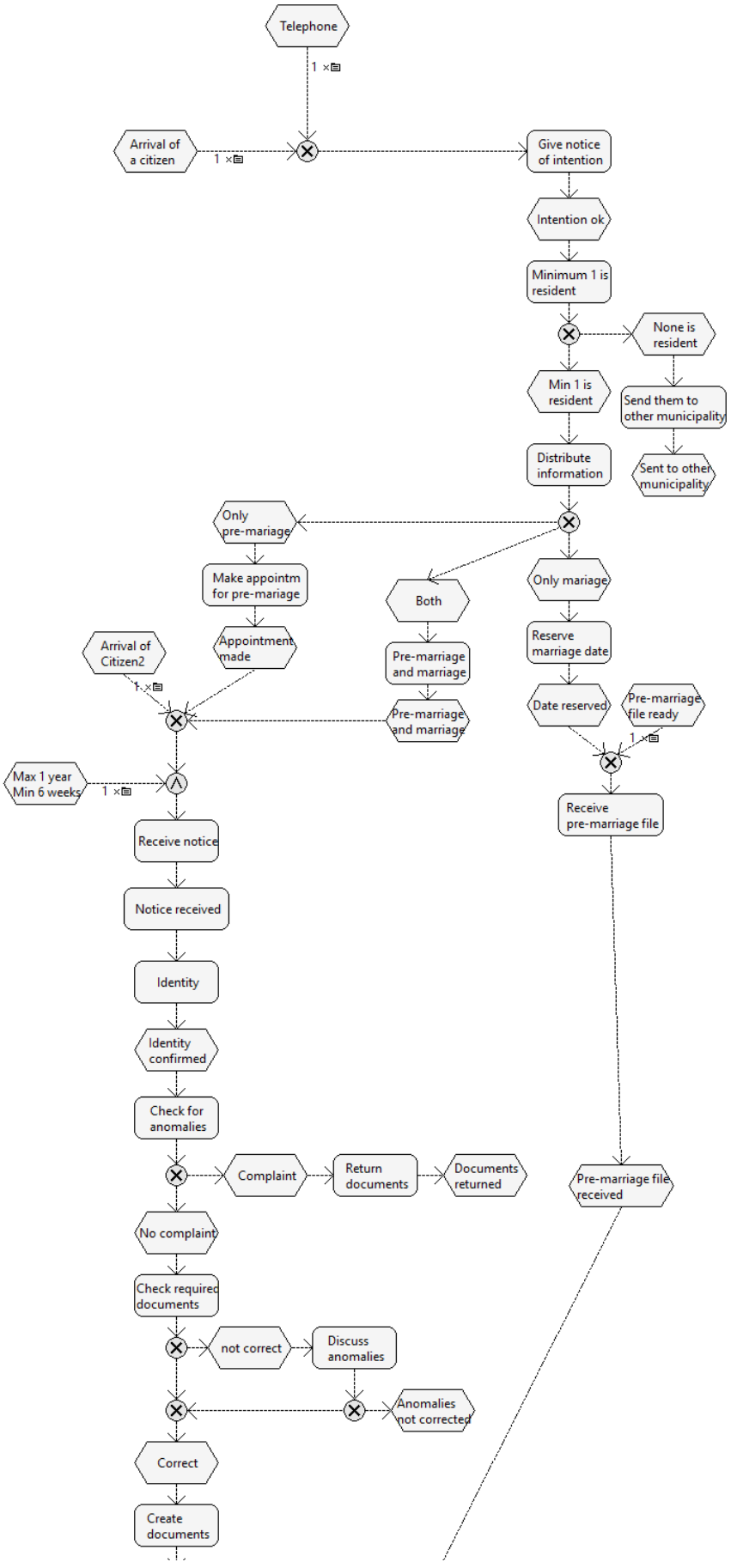
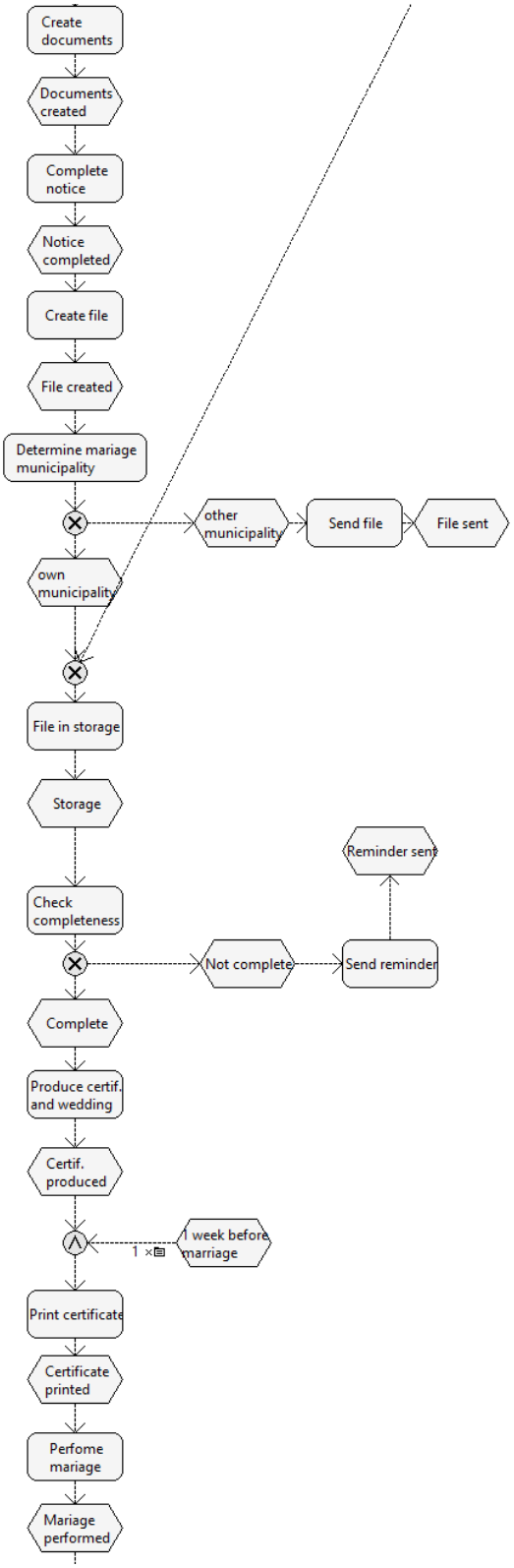


FIGURE E.13: Marriage - Variant 3

(A) Mariage- Variant 4 - Part 1/3



(B) Mariage- Variant 4 - Part 2/3



(c) Mariage- Variant 4 - Part 3/3

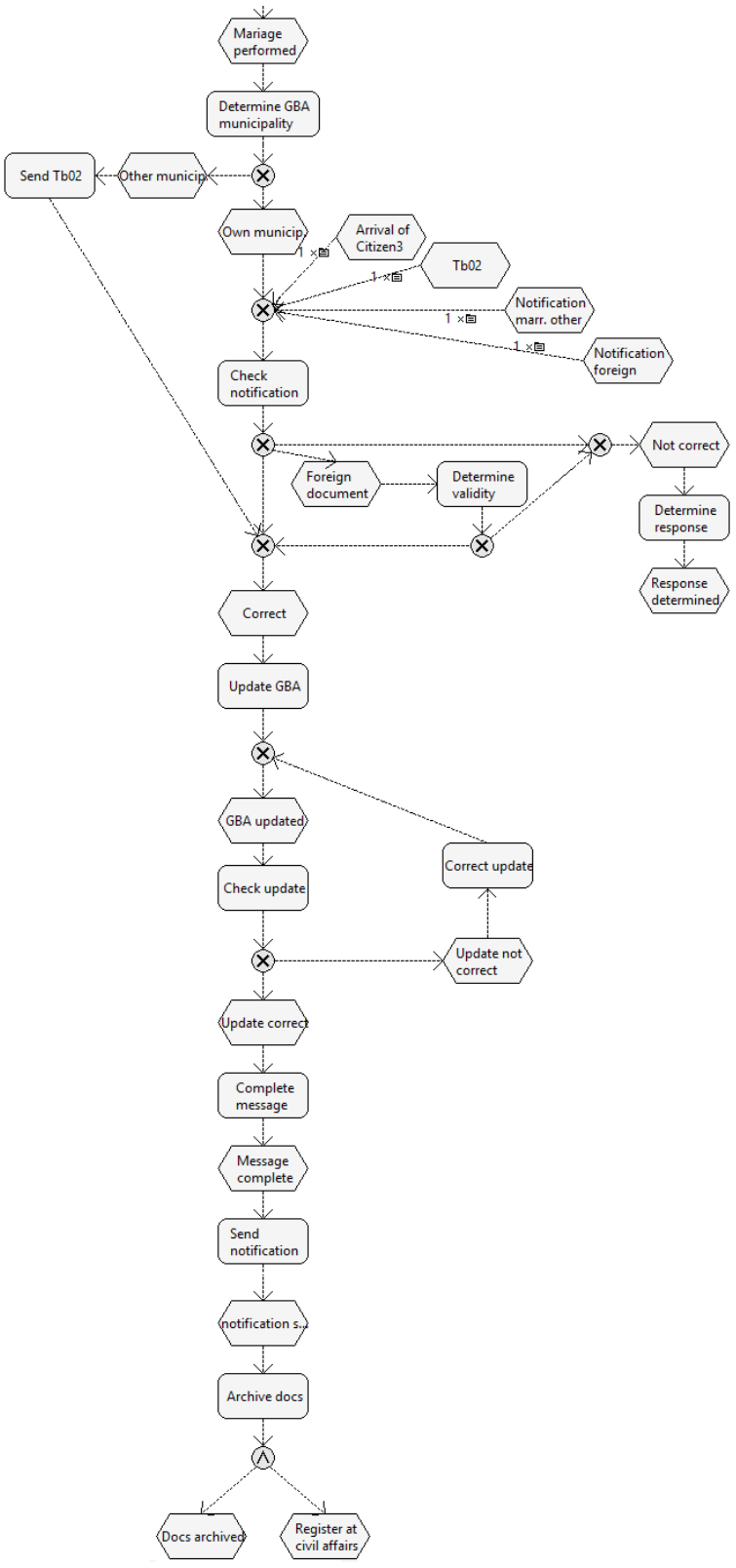
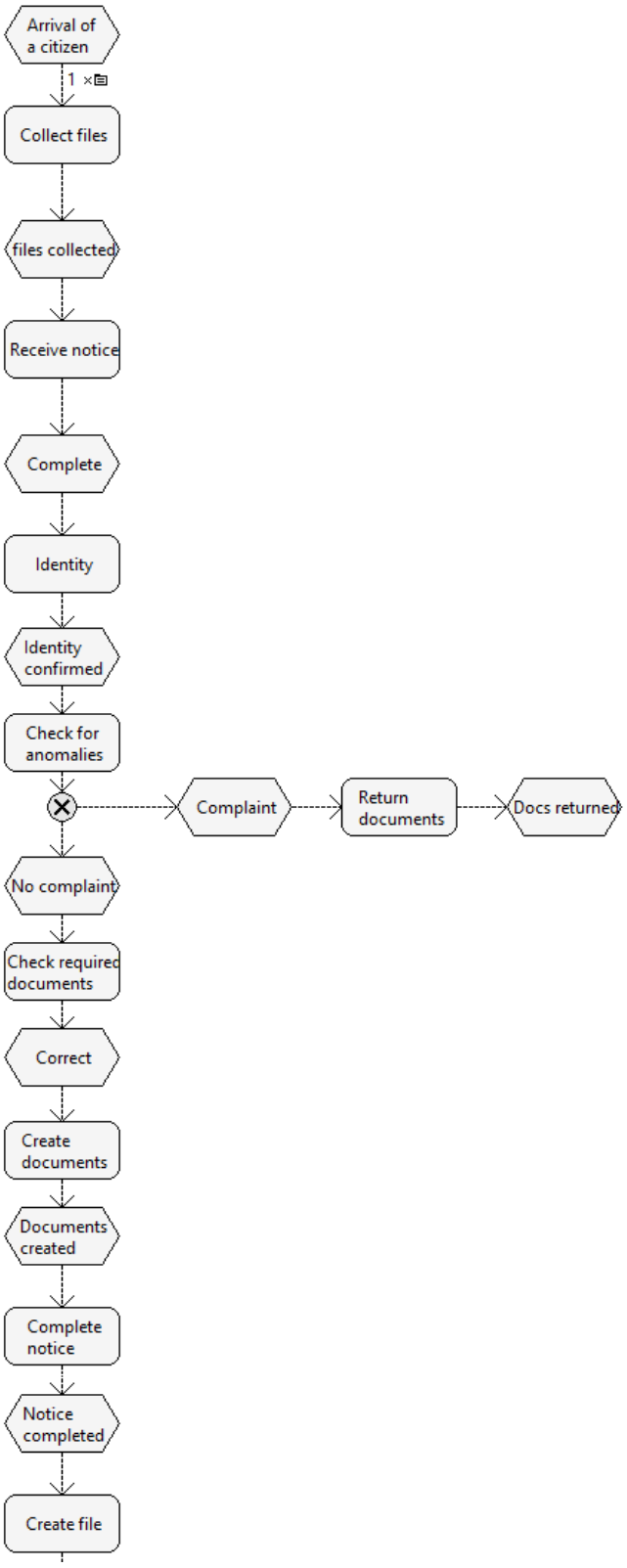
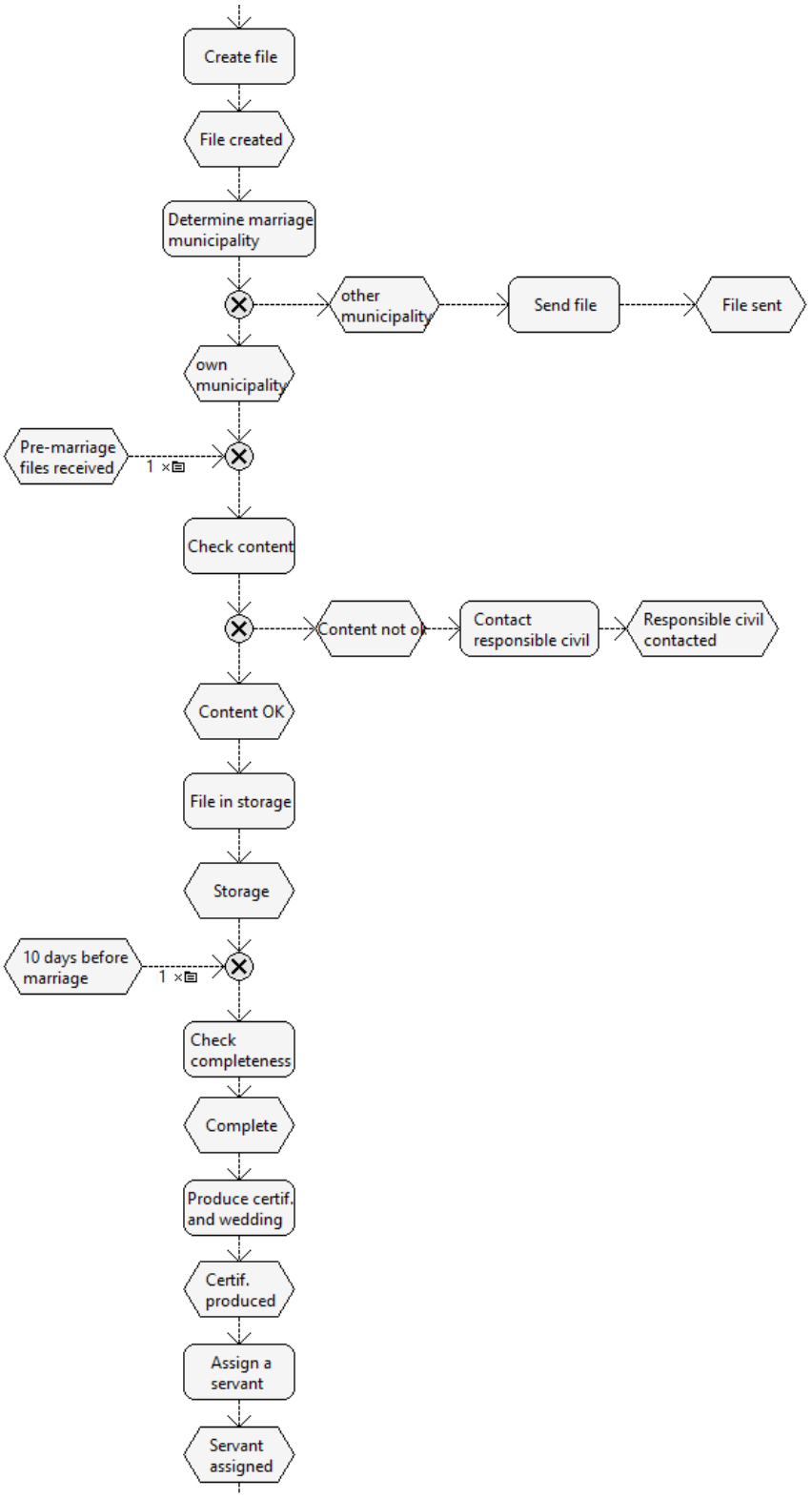


FIGURE E.14: Marriage - Variant 4

(A) Mariage- Variant 5 - Part 1/3



(B) Mariage- Variant 5 - Part 2/3



(c) Mariage- Variant 5 - Part 3/3

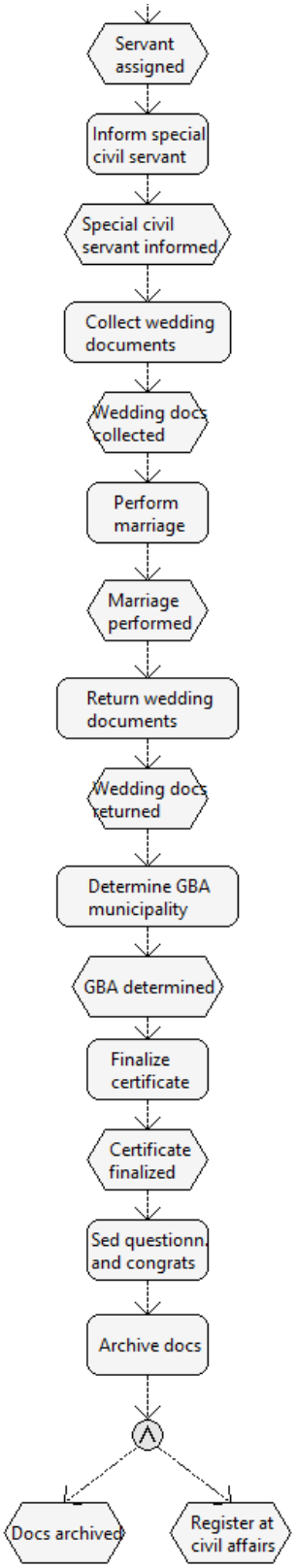
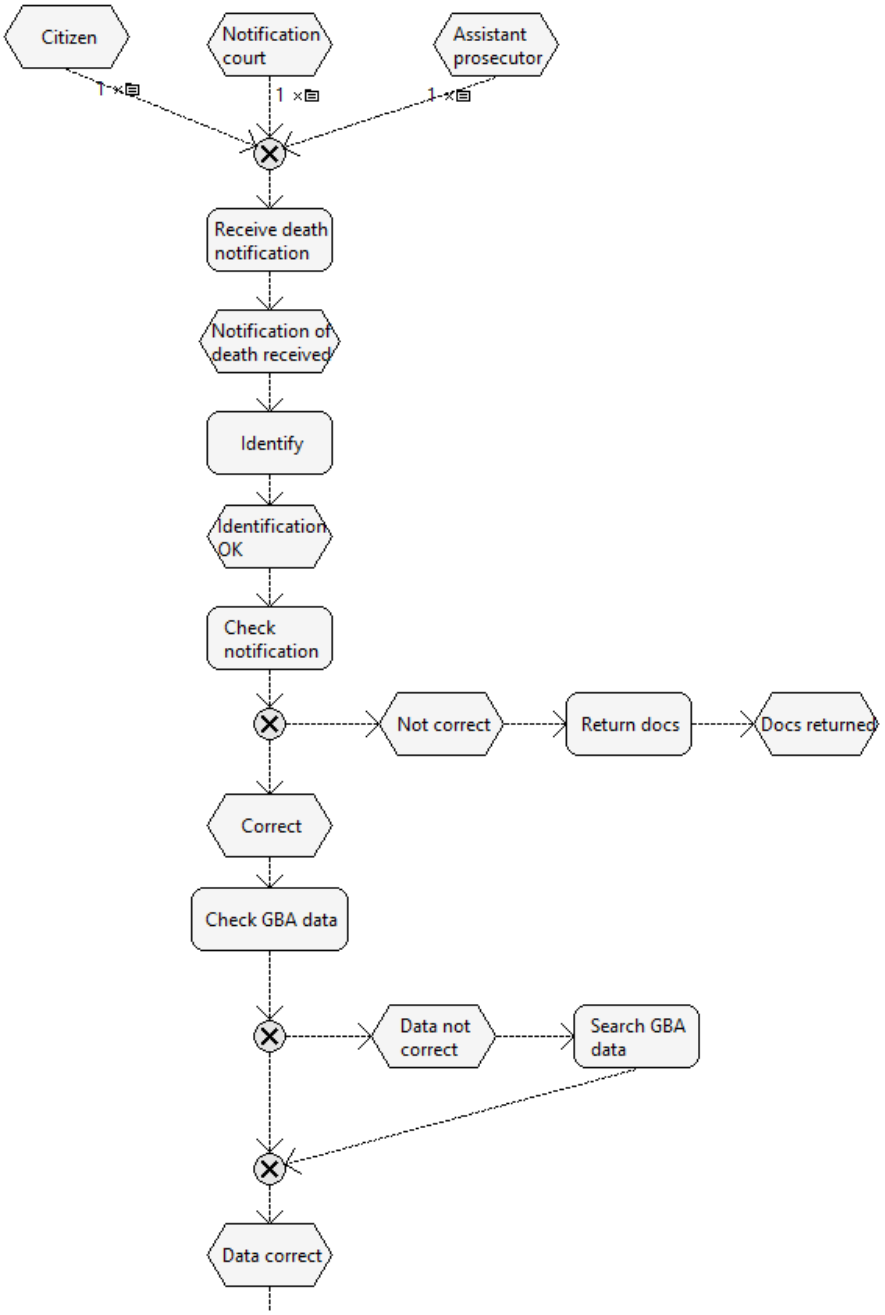


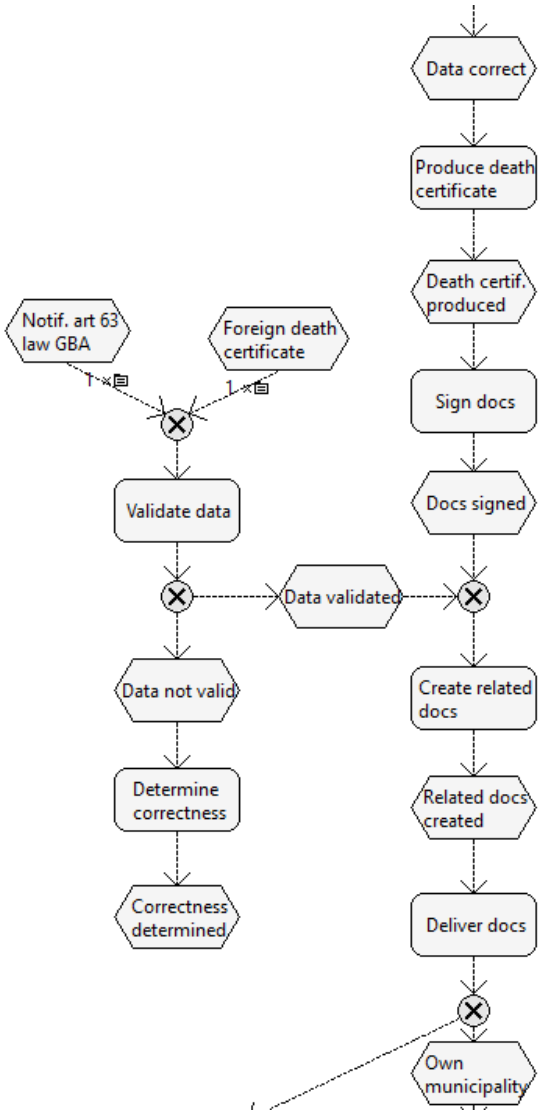
FIGURE E.15: Marriage - Variant 5

E.4 Issuing Death Certificate

(A) Issuing a Death Certificate - Variant 1 - Part 1/3



(B) Issuing a Death Certificate - Variant 1 - Part 2/3



(c) Issuing a Death Certificate - Variant 1 - Part 3/3

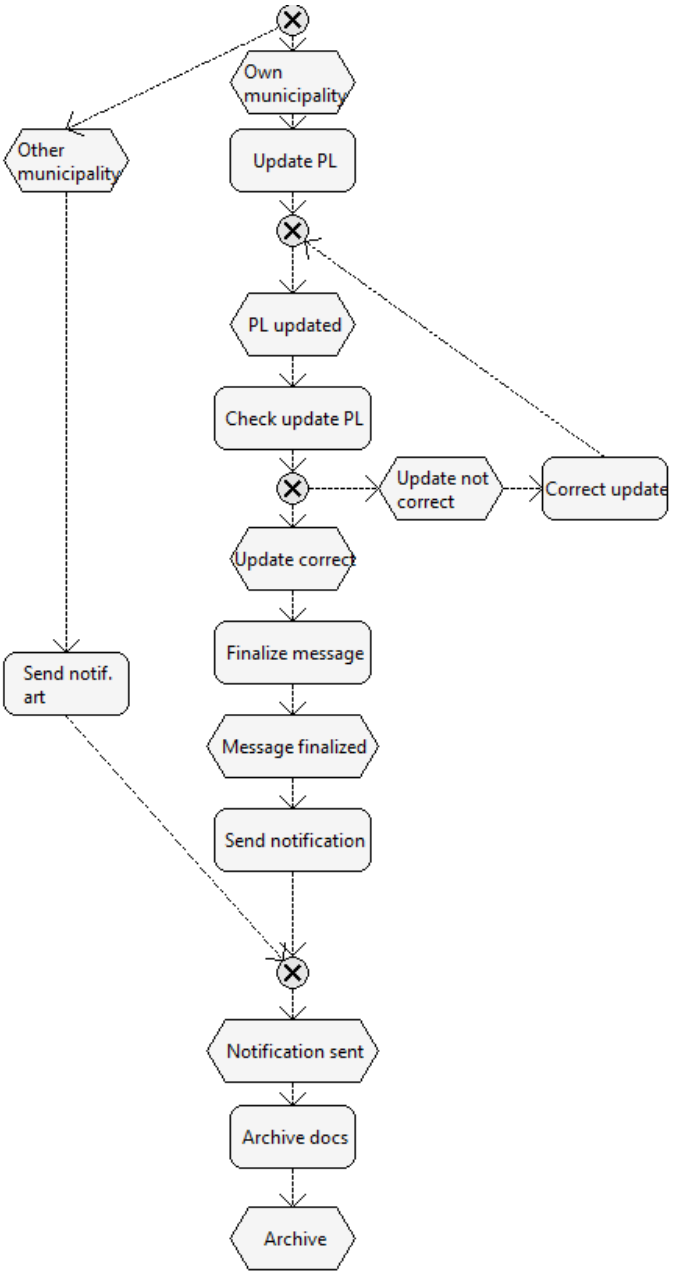
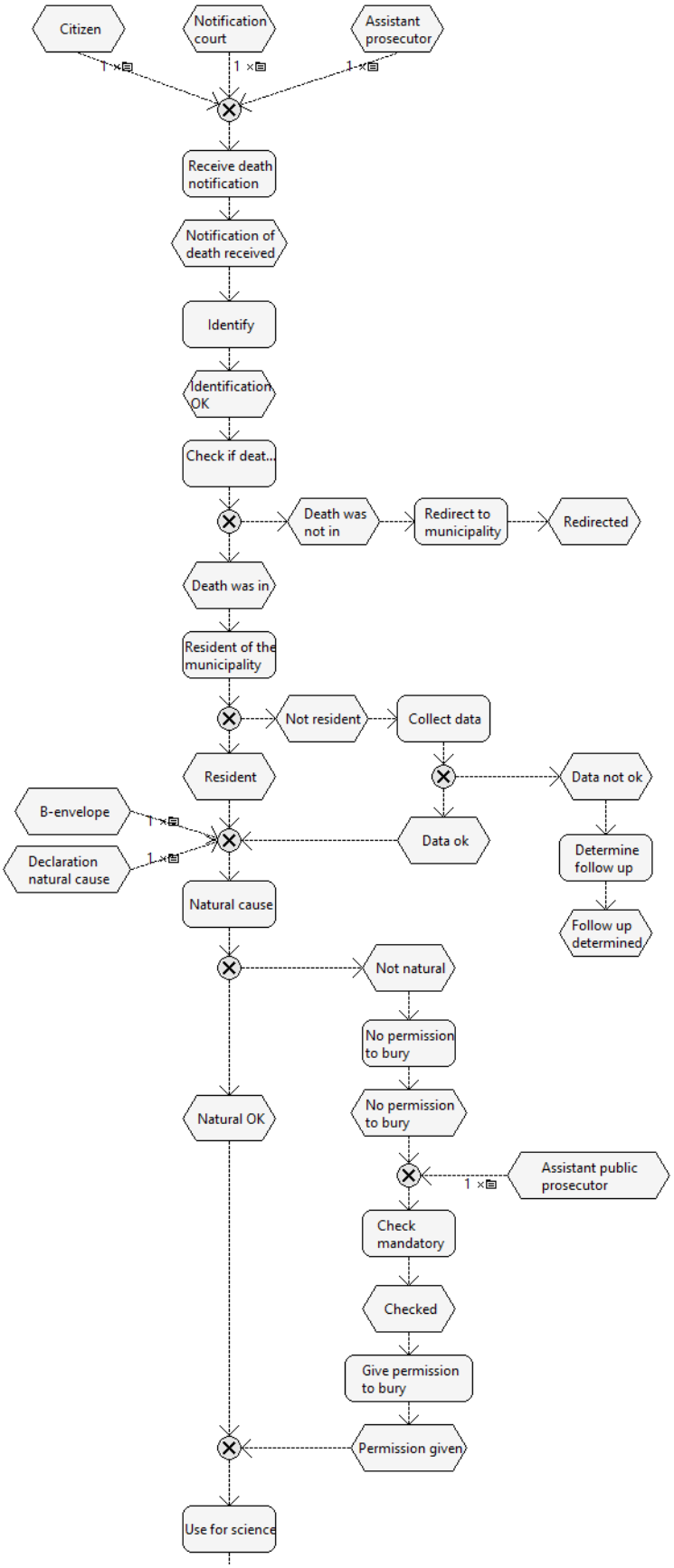
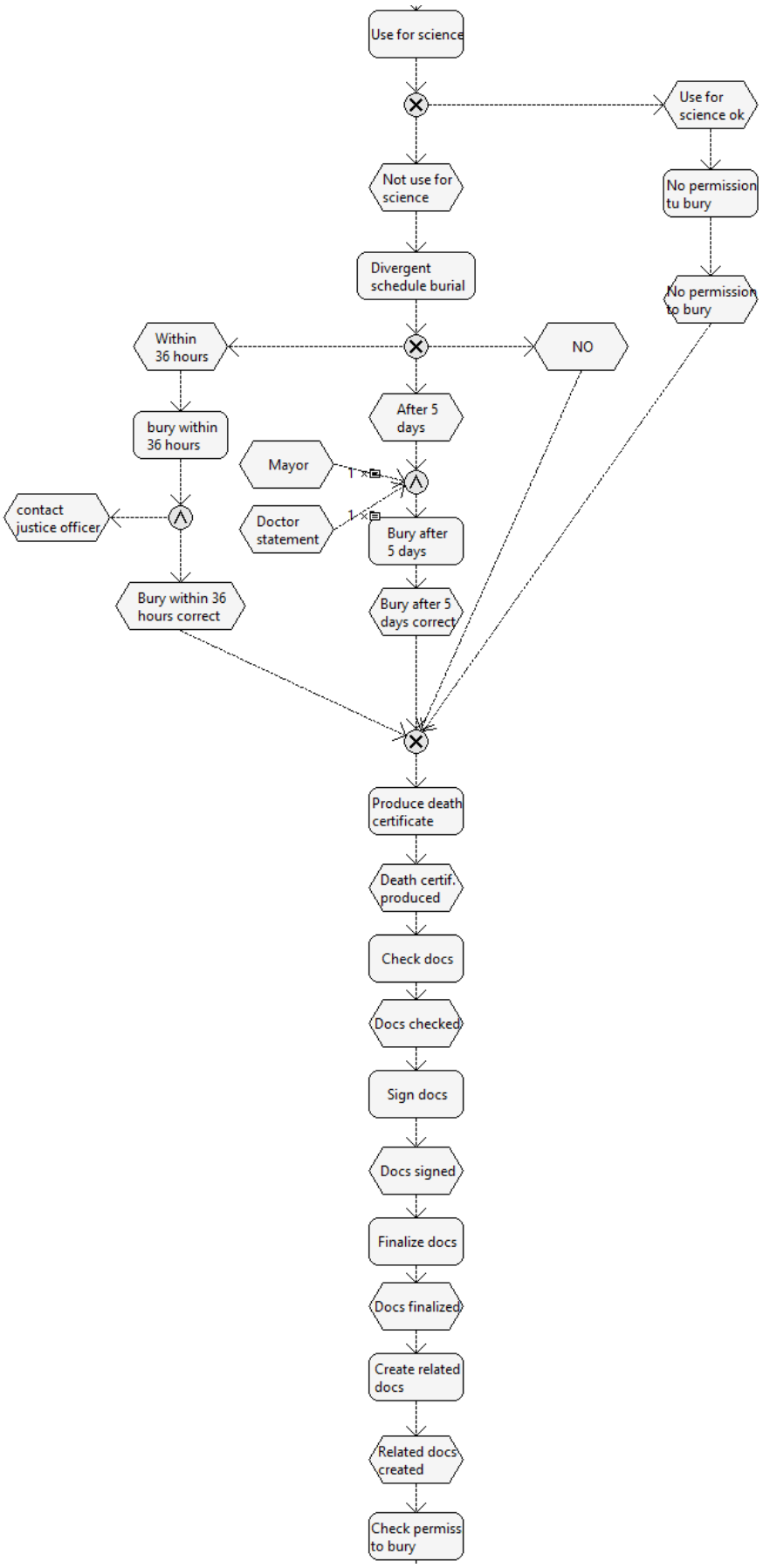


FIGURE E.16: Issuing a Death Certificate - Variant 1

(A) Issuing a Death Certificate - Variant 2 - Part 1/3



(B) Issuing a Death Certificate - Variant 2 - Part 2/3



(c) Issuing a Death Certificate - Variant 2 - Part 3/3

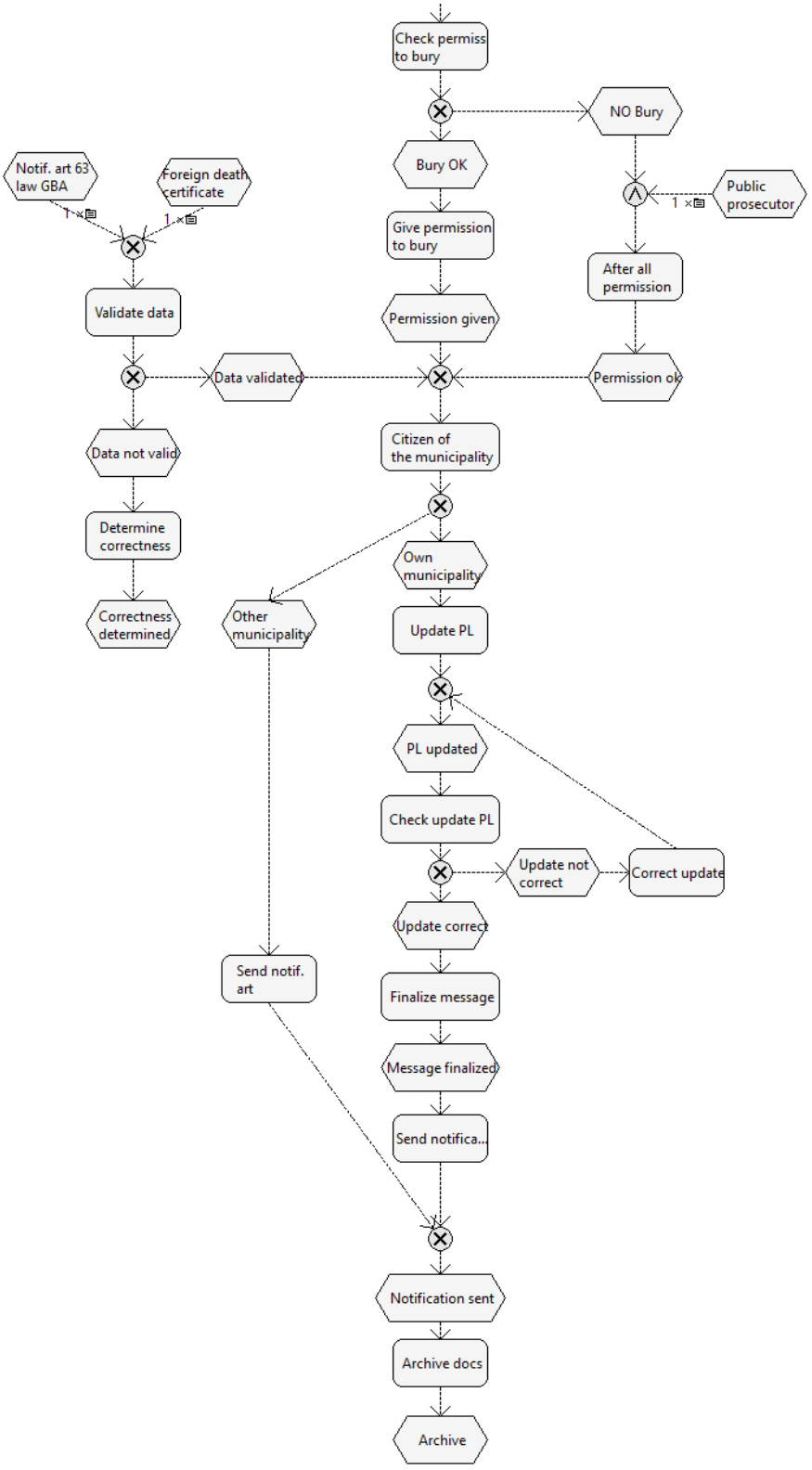
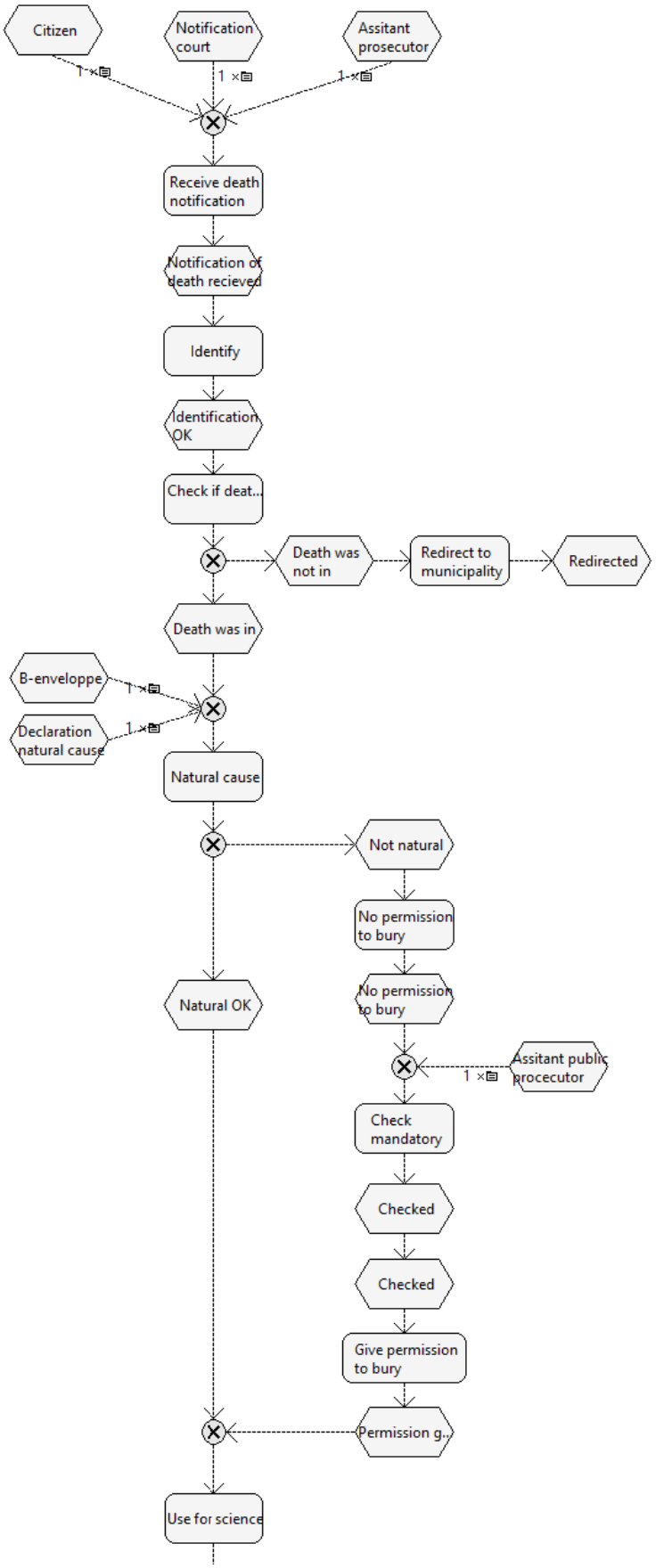
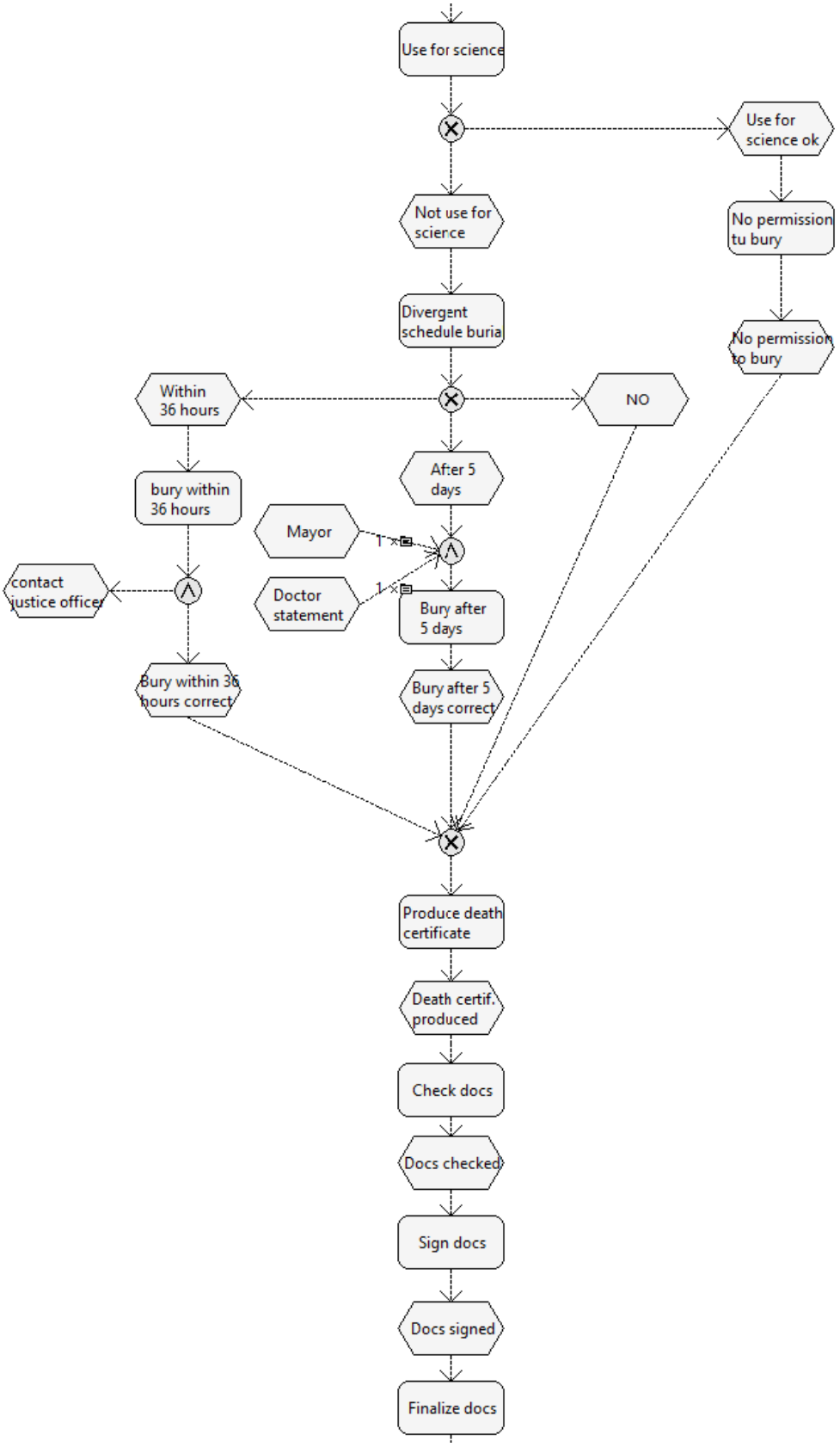


FIGURE E.17: Issuing a Death Certificate - Variant 2

(A) Issuing a Death Certificate - Variant 3 - Part 1/3



(B) Issuing a Death Certificate - Variant 3 - Part 2/3



(c) Issuing a Death Certificate - Variant 3 - Part 3/3

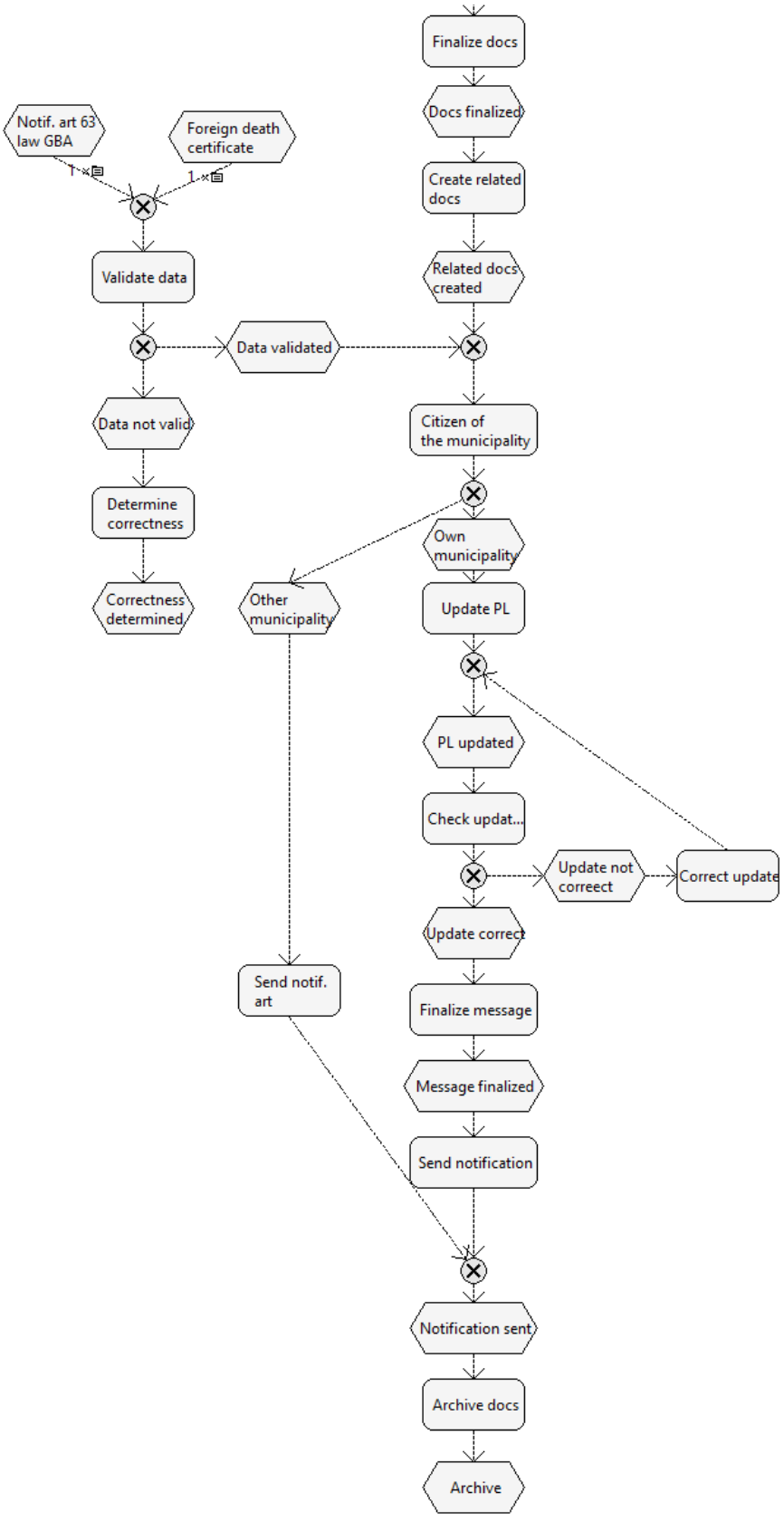
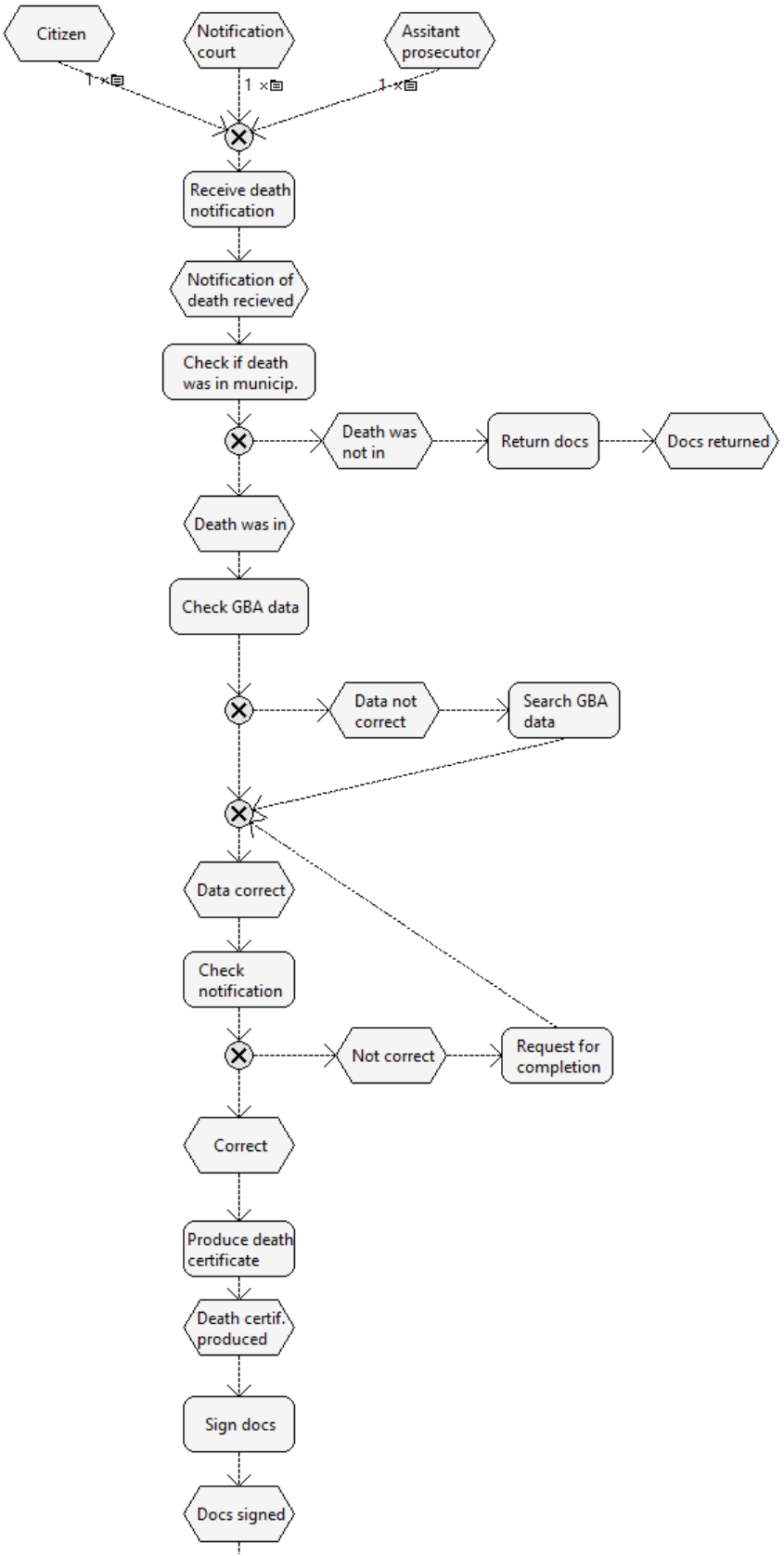


FIGURE E.18: Issuing a Death Certificate - Variant 3

(A) Issuing a Death Certificate - Variant 4 - Part 1/2



(B) Issuing a Death Certificate - Variant 4 - Part 2/2

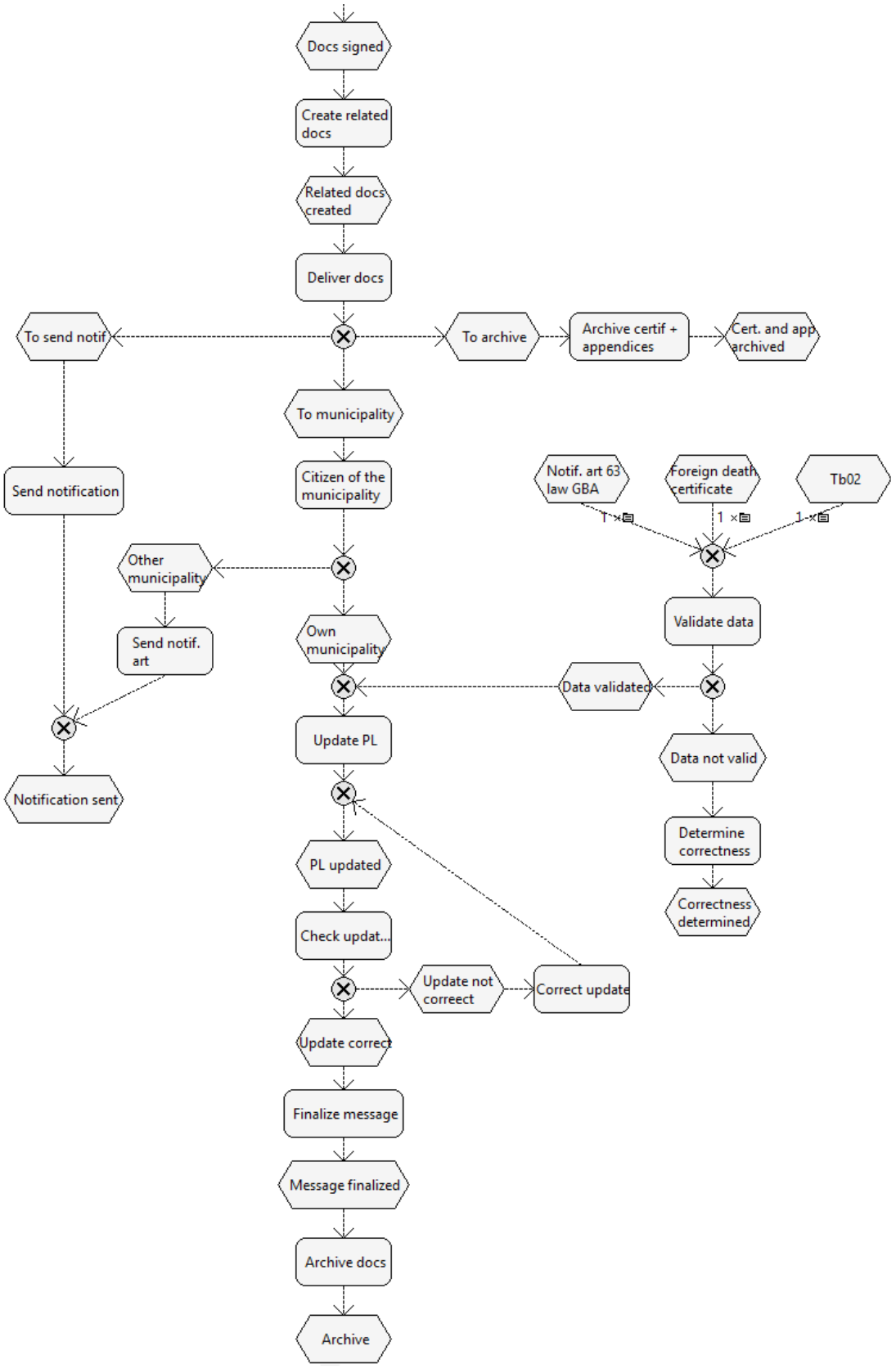
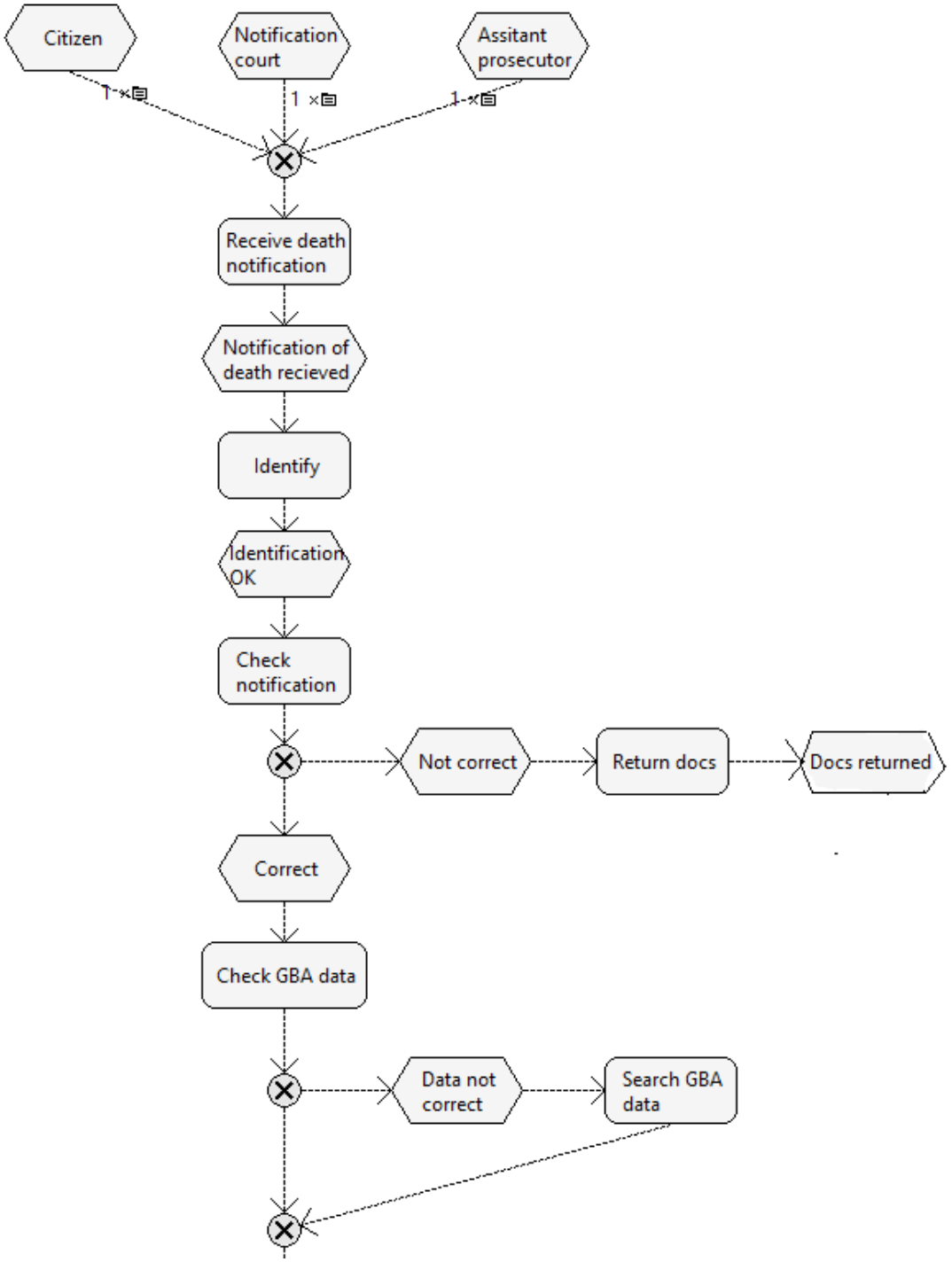


FIGURE E.19: Issuing a Death Certificate - Variant 4

(A) Issuing a Death Certificate - Variant 5 Part 1/2



(B) Issuing a Death Certificate - Variant 5 - Part 2/2

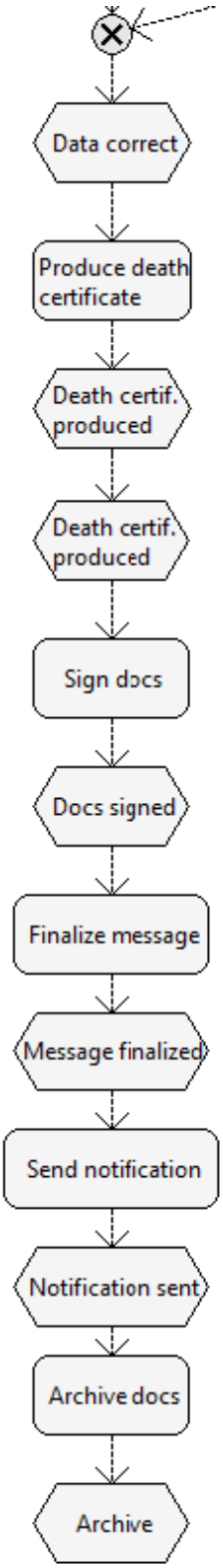


FIGURE E.20: Issuing a Death Certificate - Variant 5

Bibliography

[1] Serge Abiteboul, Omar Benjelloun, and Tova Milo. The active xml project: an overview. *VLDB J.*, 17(5):1019–1040, 2008. (Cited on page 55.)

[2] Asma Adala, Nabil Tabbane, and Sami Tabbane. A framework for automatic web service discovery based on semantics and NLP techniques. *Adv. in MM*, 2011: 238683:1–238683:7, 2011. doi: 10.1155/2011/238683. (Cited on pages 8 and 47.)

[3] Raphael Amit and Paul J. H. Schoemaker. Strategic assets and organizational rent. *Strategic Management Journal*, 14(1):33–46, 1993. ISSN 1097-0266. (Cited on pages 27 and 71.)

[4] Gonçalo Antunes, Marzieh Bakhshandeh, José Luis Borbinha, Joao Cardoso, Sharam Dadashnia, Chiara Di Francescomarino, Mauro Dragoni, Peter Fettke, Avigdor Gal, Chiara Ghidini, Philip Hake, Abderrahmane Khiat, Christopher Klinkmüller, Elena Kuss, Henrik Leopold, Peter Loos, Christian Meilicke, Tim Niesen, Catia Pesquita, Timo Péus, Andreas Schoknecht, Eitam Sheetrit, Andreas Sonntag, Heiner Stuckenschmidt, Tom Thaler, Ingo Weber, and Matthias Weidlich. The process model matching contest 2015. In Jens Kolb, Henrik Leopold, and Jan Mendling, editors, *Enterprise Modelling and Information Systems Architectures, Proceedings of the 6th Int. Workshop on Enterprise Modelling and Information Systems Architectures, EMISA 2015, Innsbruck, Austria, September 3-4, 2015.*, volume 248 of *LNI*, pages 127–155. GI, 2015. ISBN 978-3-88579-642-8. (Cited on page 165.)

[5] Gabriela Arévalo, Stéphane Ducasse, and Oscar Nierstrasz. Lessons learned in applying formal concept analysis to reverse engineering. In Bernhard Ganter and Robert Godin, editors, *Formal Concept Analysis, Third International Conference, ICFCA 2005, Lens, France, February 14-18, 2005, Proceedings*, volume 3403 of *Lecture Notes in Computer Science*, pages 95–112. Springer, 2005. ISBN 3-540-24525-1. doi: 10.1007/978-3-540-32262-7_7. (Cited on page 149.)

[6] Nour Assy, Walid Gaaloul, and Bruno Defude. Mining configurable process fragments for business process design. In Monica Chiarini Tremblay, Debra E. VanderMeer, Marcus A. Rothenberger, Ashish Gupta, and Victoria Y. Yoon, editors, *Advancing the Impact of Design Science: Moving from Theory to Practice - 9th International Conference, DESRIST 2014, Miami, FL, USA, May 22-24, 2014. Proceedings*, volume 8463 of *Lecture Notes in Computer Science*, pages 209–224. Springer, 2014. ISBN 978-3-319-06700-1. doi: 10.1007/978-3-319-06701-8_14. (Cited on page 157.)

[7] Nour Assy, Nguyen Ngoc Chan, and Walid Gaaloul. An automated approach for assisting the design of configurable process models. *IEEE T. Services Computing*, 8(6):874–888, 2015. doi: 10.1109/TSC.2015.2477815. (Cited on pages 9, 53 and 184.)

[8] Ahmed Awad and Sherif Sakr. On efficient processing of bpmn-q queries. *Computers in Industry*, 63(9):867–881, 2012. (Cited on page 56.)

[9] Mustapha Aznag, Mohamed Quafafou, and Zahi Jarir. Leveraging formal concept analysis with topic correlation for service clustering and discovery. In *2014 IEEE International Conference on Web Services, ICWS, 2014, Anchorage, AK, USA, June 27 - July 2, 2014*, pages 153–160. IEEE Computer Society, 2014. ISBN 978-1-4799-5054-6. doi: 10.1109/ICWS.2014.33. (Cited on pages 8, 48, 49, 51, 52 and 149.)

[10] Sana Baccar, Wassim Derguech, Edward Curry, and Mohamed Abid. Modeling and Querying Sensor Services Using Ontologies. In Witold Abramowicz, editor, *Business Information Systems - 18th International Conference, BIS 2015, Poznań, Poland, June 24-26, 2015, Proceedings*, volume 208 of *Lecture Notes in Business Information Processing*, pages 90–101. Springer, 2015. (Cited on page 193.)

[11] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet project. In *COLING-ACL '98: Proceedings of the Conference*, pages 86–90, Montreal, Canada, 1998. (Cited on pages 74 and 199.)

[12] Eran Balan, Tova Milo, and Tal Sterenzy. Bp-ex: A uniform query engine for business process execution traces. In *Proceedings of the 13th International Conference on Extending Database Technology*, EDBT '10, pages 713–716, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-945-9. doi: 10.1145/1739041.1739134. (Cited on pages 55 and 67.)

[13] Mateusz Baran, Krzysztof Kluza, Grzegorz J. Nalepa, and Antoni Ligeza. A hierarchical approach for configuring business processes. In Maria Ganzha, Leszek A.

Maciaszek, and Marcin Paprzycki, editors, *FedCSIS*, pages 915–921, 2013. (Cited on pages 62 and 67.)

[14] Dinesh Batra and Joseph G. Davis. Conceptual data modelling in database design: Similarities and differences between expert and novice designers. *International Journal of Man-Machine Studies*, 37(1):83–101, 1992. doi: 10.1016/0020-7373(92)90092-Y. (Cited on pages 89 and 96.)

[15] Jörg Becker, P Delfmann, and R Knackstedt. *Efficient Information Systems Design Through Reuse of Information Models*, chapter Adaptive Reference Modeling: Integrating Configurative and Generic Adaptation Techniques for Information Models, pages 27–58. Physica-Verlag HD, 2007. (Cited on page 60.)

[16] Catriel Beeri, Anat Eyal, Simon Kamenkovich, and Tova Milo. Querying business processes. In Umeshwar Dayal, Kyu-Young Whang, David B. Lomet, Gustavo Alonso, Guy M. Lohman, Martin L. Kersten, Sang Kyun Cha, and Young-Kuk Kim, editors, *VLDB*, pages 343–354. ACM, 2006. ISBN 1-59593-385-9. (Cited on pages 55 and 66.)

[17] Catriel Beeri, Anat Eyal, Simon Kamenkovich, and Tova Milo. Querying business processes with bp-ql. *Inf. Syst.*, 33(6):477–507, 2008. (Cited on pages 54 and 67.)

[18] Catriel Beeri, Anat Eyal, Tova Milo, and Alon Pilberg. Bp-mon: query-based monitoring of bpel business processes. *SIGMOD Record*, 37(1):21–24, 2008. (Cited on pages 55 and 67.)

[19] Radim Belohlávek and Vilém Vychodil. Background knowledge in formal concept analysis: constraints via closure operators. In *Proceedings of the 2010 ACM Symposium on Applied Computing (SAC)*, pages 1113–1114, Sierre, Switzerland, 22-26 March 2010. ACM. ISBN 978-1-60558-639-7. (Cited on page 132.)

[20] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284(5):34–43, May 2001. (Cited on page 16.)

[21] Sami Bhiri, Wassim Derguech, and Maciej Zaremba. Web service capability meta model. In Karl-Heinz Krempels and José Cordeiro, editors, *WEBIST 2012 - Proceedings of the 8th International Conference on Web Information Systems and Technologies, Porto, Portugal, 18 - 21 April, 2012*, pages 47–57. SciTePress, 2012. (Cited on page 193.)

[22] Sami Bhiri, Wassim Derguech, and Maciej Zaremba. Modelling Capabilities as Attribute-Featured Entities. In José Cordeiro and Karl-Heinz Krempels, editors, *Web Information Systems and Technologies - 8th International Conference, WEBIST 2012, Porto, Portugal, April 18-21, 2012, Revised Selected Papers*, volume

140 of *Lecture Notes in Business Information Processing*, pages 70–85. Springer, 2012. (Cited on page 193.)

[23] Walter Binder, Adina D. Mosincat, Samuel Spycher, Ion Constantinescu, and Boi Faltings. Multiversion concurrency control for the generalized search tree. *Concurrency and Computation: Practice and Experience*, 21(12):1547–1571, 2009. doi: 10.1002/cpe.1387. (Cited on page 13.)

[24] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked Data - The Story So Far. *IJSWIS*, 5(3), 2009. (Cited on page 81.)

[25] François Bodart, Arvind Patel, Marc Sim, and Ron Weber. Should optional properties be used in conceptual modelling? A theory and three empirical tests. *Information Systems Research*, 12(4):384–405, 2001. doi: 10.1287/isre.12.4.384.9702. (Cited on pages 17, 18, 89, 96 and 124.)

[26] David Bourgeois. *Information Systems for Business and Beyond*. Saylor Foundation, 2014. (Cited on page 28.)

[27] OMG BPMI. BPMN Specification: Business Process Modelling Notation. `http://www.bpmn.org/`, 2004. (Cited on page 106.)

[28] Robert Braun and Werner Esswein. Classification of reference models. In *Advances in Data Analysis*. Springer, 2007. (Cited on page 154.)

[29] Malte Brettel, Niklas Friederichsen, Michael Keller, and Marius Rosenberg. How virtualization, decentralization and network building change the manufacturing landscape: An industry 4.0 perspective. *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, 8(1):37 – 44, 2014. ISSN 1307-6892. (Cited on page 2.)

[30] M. Bunge. *Treatise on Basic Philosophy. Ontology I: The Furniture of the World.* Boston, Riedel, 1977. (Cited on pages 20, 91, 92, 94, 95 and 192.)

[31] Jorge Cardoso and Wil M. P. van der Aalst. Path mining and process mining for workflow management systems. In John Wang, editor, *Encyclopedia of Data Warehousing and Mining*, pages 1489–1496. IGI Global, 2009. ISBN 9781605660103. (Cited on page 53.)

[32] Jorge Cardoso, Amit P. Sheth, John A. Miller, Jonathan Arnold, and Krys Kochut. Quality of service for workflows and web service processes. *J. Web Sem.*, 1(3):281–308, 2004. doi: 10.1016/j.websem.2004.03.001. (Cited on pages 12, 104 and 127.)

[33] Nguyen Ngoc Chan, Karn Yongsiriwit, Walid Gaaloul, and Jan Mendling. Mining event logs to assist the development of executable process variants. In Matthias Jarke, John Mylopoulos, Christoph Quix, Colette Rolland, Yannis Manolopoulos, Haralambos Mouratidis, and Jennifer Horkoff, editors, *Advanced Information Systems Engineering - 26th International Conference, CAiSE 2014, Thessaloniki, Greece, June 16-20, 2014. Proceedings*, volume 8484 of *Lecture Notes in Computer Science*, pages 548–563. Springer, 2014. ISBN 978-3-319-07880-9. doi: 10.1007/978-3-319-07881-6. (Cited on page 157.)

[34] Hsinchun Chen, Roger H. L. Chiang, and Veda C. Storey. Business intelligence and analytics: From big data to big impact. *MIS Q.*, 36(4):1165–1188, December 2012. ISSN 0276-7783. (Cited on page 2.)

[35] Erik Christensen, Francisco Curbera, Greg Meredith, and Sanjiva Weerawarana. WSDL: Web Services Description Language. http://www.w3.org/TR/wsdl, 2001. Accessed: 25/05/2014. (Cited on pages 33, 36 and 72.)

[36] Philipp Cimiano, Andreas Hotho, Gerd Stumme, and Julien Tane. Conceptual knowledge processing with formal concept analysis and ontologies. In Peter W. Eklund, editor, *Concept Lattices, Second International Conference on Formal Concept Analysis, ICFCA 2004, Sydney, Australia, February 23-26, 2004, Proceedings*, volume 2961 of *Lecture Notes in Computer Science*, pages 189–207. Springer, 2004. ISBN 3-540-21043-1. doi: 10.1007/978-3-540-24651-0_18. (Cited on page 147.)

[37] Edward Curry, Souleiman Hasan, and Seán O'Riain. Enterprise energy management using a linked dataspace for energy intelligence. In *Sustainable Internet and ICT for Sustainability, SustainIT 2012, 4-5 October, 2012, Pisa, Italy, Sponsored by the IFIP TC6 WG 6.3 "Performance of Communication Systems"*, pages 1–6. IEEE, 2012. ISBN 978-3-901882-46-3. (Cited on pages 21 and 142.)

[38] Edward Curry, James O'Donnell, Edward Corry, Souleiman Hasan, Marcus M. Keane, and Seán O'Riain. Linking building data in the cloud: Integrating cross-domain building data using linked data. *Advanced Engineering Informatics*, 27(2): 206–219, 2013. doi: 10.1016/j.aei.2012.10.003. (Cited on page 143.)

[39] Richard Cyganiak. Buildings and rooms vocabulary. http://vocab.deri.ie/rooms, 2012. Accessed: 25/05/2014. (Cited on page 144.)

[40] Richard Cyganiak and Anja Jentzsch. The linking open data cloud diagram. http://lod-cloud.net/, 2014. Accessed: 16/04/2016. (Cited on page 199.)

[41] Maya Daneva. ERP requirements engineering practice: Lessons learned. *IEEE Software*, 21(2):26–33, 2004. doi: 10.1109/MS.2004.1270758. (Cited on page 14.)

[42] Florian Daniel, Jianmin Wang, and Barbara Weber, editors. *Business Process Management - 11th International Conference, BPM 2013, Beijing, China, August 26-30, 2013. Proceedings*, volume 8094 of *Lecture Notes in Computer Science*, 2013. Springer. ISBN 978-3-642-40175-6. doi: 10.1007/978-3-642-40176-3. (Cited on pages 292 and 308.)

[43] Gordon B. Davis. Strategies for information requirements determination. *IBM Systems Journal*, 21(1):4–30, 1982. doi: 10.1147/sj.211.0004. (Cited on page 91.)

[44] Rob Davis. *ARIS Design Platform: Advanced Process Modelling and Administration*. Springer Publishing Company, Incorporated, 1st edition, 2008. ISBN 184800110X, 9781848001107. (Cited on page 1.)

[45] Jos de Bruijn, Holger Lausen, Axel Polleres, and Dieter Fensel. The web service modeling language WSML: an overview. In York Sure and John Domingue, editors, *The Semantic Web: Research and Applications, 3rd European Semantic Web Conference, ESWC 2006, Budva, Montenegro, June 11-14, 2006, Proceedings*, volume 4011 of *Lecture Notes in Computer Science*, pages 590–604. Springer, 2006. ISBN 3-540-34544-2. doi: 10.1007/11762256_43. (Cited on page 31.)

[46] Gero Decker, Hagen Overdick, and Mathias Weske. Oryx - an open modeling platform for the bpm community. In Dumas et al. [66], pages 382–385. ISBN 978-3-540-85757-0. (Cited on pages 56 and 66.)

[47] Wassim Derguech. Towards a framework for business process models reuse. In Anne Persson, Boualem Benatallah, and Adnene Guabtni, editors, *Proceedings of the CAiSE Doctoral Consortium 2010*, volume 593, Hammamet, Tunisia, 2010. ceur-ws.org. (Not cited.)

[48] Wassim Derguech and Sami Bhiri. Reuse-Oriented Business Process Modelling Based on a Hierarchical Structure. In Michael zur Muehlen and Jianwen Su, editors, *Business Process Management Workshops*, volume 66 of *Lecture Notes in Business Information Processing*, pages 301–313. Springer, 2010. ISBN 978-3-642-20510-1. (Cited on pages xix, 62, 63 and 67.)

[49] Wassim Derguech and Sami Bhiri. Merging Business Process Variants. In Witold Abramowicz, editor, *Business Information Systems - 14th International Conference, BIS 2011, Poznan, Poland, June 15-17, 2011. Proceedings*, volume 87 of *Lecture Notes in Business Information Processing*, pages 86–97. Springer, 2011. (Cited on page 194.)

[50] Wassim Derguech and Sami Bhiri. An Automation Support for Creating Configurable Process Models. In Athman Bouguettaya, Manfred Hauswirth, and Ling

Liu, editors, *Web Information System Engineering - WISE 2011 - 12th International Conference, Sydney, Australia, October 13-14, 2011. Proceedings*, volume 6997 of *Lecture Notes in Computer Science*, pages 199–212. Springer, 2011. (Cited on pages 14, 152, 155 and 194.)

[51] Wassim Derguech and Sami Bhiri. Capability Modelling - Case of Logistics Capabilities. In Marcello La Rosa and Pnina Soffer, editors, *Business Process Management Workshops - BPM 2012 International Workshops, Tallinn, Estonia, September 3, 2012. Revised Papers*, volume 132 of *Lecture Notes in Business Information Processing*, pages 519–529. Springer, 2012. (Cited on page 193.)

[52] Wassim Derguech and Sami Bhiri. Modelling, interlinking and discovering capabilities. In *ACS International Conference on Computer Systems and Applications, AICCSA 2013, Ifrane, Morocco, May 27-30, 2013*, pages 1–8, 2013. (Cited on pages 77, 78 and 193.)

[53] Wassim Derguech and Sami Bhiri. Business Process Model Overview: Determining the Capability of a Process Model Using Ontologies. In Witold Abramowicz, editor, *Business Information Systems - 16th International Conference, BIS 2013, Poznań, Poland, June 19-21, 2013. Proceedings*, volume 157 of *Lecture Notes in Business Information Processing*, pages 62–74. Springer, 2013. (Cited on page 193.)

[54] Wassim Derguech, Gabriela Vulcu, and Sami Bhiri. An Indexing Structure for Maintaining Configurable Process Models. In Ilia Bider, Terry A. Halpin, John Krogstie, Selmin Nurcan, Erik Proper, Rainer Schmidt, and Roland Ukor, editors, *BMMDS/EMMSAD*, volume 50 of *Lecture Notes in Business Information Processing*, pages 157–168. Springer, 2010. ISBN 978-3-642-13050-2. (Cited on pages 62 and 67.)

[55] Wassim Derguech, Feng Gao, and Sami Bhiri. Configurable Process Models for Logistics Case Study for Customs Clearance Processes. In Florian Daniel, Kamel Barkaoui, and Schahram Dustdar, editors, *Business Process Management Workshops - BPM 2011 International Workshops, Clermont-Ferrand, France, August 29, 2011, Revised Selected Papers, Part II*, volume 100 of *Lecture Notes in Business Information Processing*, pages 119–130. Springer, 2011. (Cited on page 194.)

[56] Wassim Derguech, Souleiman Hasan, Sami Bhiri, and Edward Curry. Organizing Capabilities Using Formal Concept Analysis. In Sumitra Reddy and Mohamed Jmaiel, editors, *2013 Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, Hammamet, Tunisia, June 17-20, 2013*, pages 260–265. IEEE, 2013. (Cited on page 194.)

[57] Wassim Derguech, Sami Bhiri, Souleiman Hasan, and Edward Curry. Using Formal Concept Analysis for Organizing and Discovering Sensor Capabilities. *The Computer Journal*, 58(3):356–367, 2015. (Cited on page 194.)

[58] Claudia Diamantini, Laura Genga, Domenico Potena, and Wil M. P. van der Aalst. Towards process instances building for spaghetti processes. In Domenico Lembo, Riccardo Torlone, and Andrea Marrella, editors, *23rd Italian Symposium on Advanced Database Systems, SEBD 2015, Gaeta, Italy, June 14-17, 2015.*, pages 256–263. Curran Associates, Inc., 2015. ISBN 978-1-5108-1087-7. (Cited on page 4.)

[59] Remco M. Dijkman, Marlon Dumas, Boudewijn F. van Dongen, Reina Käärik, and Jan Mendling. Similarity of business process models: Metrics and evaluation. *Inf. Syst.*, 36(2):498–516, 2011. doi: 10.1016/j.is.2010.09.006. (Cited on page 165.)

[60] Remco M. Dijkman, Boudewijn F. van Dongen, Marlon Dumas, Luciano García-Bañuelos, Matthias Kunze, Henrik Leopold, Jan Mendling, Reina Uba, Matthias Weidlich, Mathias Weske, and Zhiqiang Yan. A short survey on process model similarity. In Janis A. Bubenko Jr., John Krogstie, Oscar Pastor, Barbara Pernici, Colette Rolland, and Arne Sølvberg, editors, *Seminal Contributions to Information Systems Engineering, 25 Years of CAiSE*, pages 421–427. Springer, 2013. ISBN 978-3-642-36925-4. doi: 10.1007/978-3-642-36926-1_34. (Cited on page 165.)

[61] Xin Dong, Alon Y. Halevy, Jayant Madhavan, Ema Nemes, and Jun Zhang. Similarity search for web services. In *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases*, pages 372–383, Toronto, Canada, 31 August - 3 September 2004. Morgan Kaufmann. ISBN 0-12-088469-0. (Cited on page 13.)

[62] E. Drever and Scottish Council for Research in Education. *Using semi-structured interviews in small-scale research: a teacher's guide*. SCRE publication. Scottish Council for Research in Education, 1995. ISBN 9781860030116. (Cited on pages 17, 89, 96 and 124.)

[63] Susan T Dumais, George W Furnas, Thomas K Landauer, Scott Deerwester, and Richard Harshman. Using latent semantic analysis to improve access to textual information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 281–285, 1988. (Cited on page 164.)

[64] Marlon Dumas, Wil M. van der Aalst, and Arthur H. ter Hofstede, editors. *Process-Aware Information Systems : Bridging People and Software through Process Technology*. Wiley-Interscience, Hoboken, NJ, 2005. ISBN 9780471663065. (Cited on page 103.)

[65] Marlon Dumas, Wil M. P. van der Aalst, and Arthur H. M. ter Hofstede. Introduction. In *Process-Aware Information Systems: Bridging People and Software Through Process Technology*. Wiley, 2005. ISBN 978-0-471-66306-5. (Cited on pages xix, 1 and 2.)

[66] Marlon Dumas, Manfred Reichert, and Ming-Chien Shan, editors. *Business Process Management, 6th International Conference, BPM 2008, Milan, Italy, September 2-4, 2008. Proceedings*, volume 5240 of *Lecture Notes in Computer Science*, 2008. Springer. ISBN 978-3-540-85757-0. (Cited on pages 284 and 293.)

[67] Marlon Dumas, Luciano García-Bañuelos, Marcello La Rosa, and Reina Uba. Fast detection of exact clones in business process model repositories. *Inf. Syst.*, 38(4): 619–633, 2013. (Cited on pages 57, 58, 66 and 153.)

[68] Shantanu Dutta, Om Narasimhan, and Surendra Rajiv. Conceptualizing and measuring capabilities: methodology and empirical application. *Strategic Management Journal*, 26(3):277–285, 2005. ISSN 1097-0266. (Cited on pages 27 and 71.)

[69] Johann Eder and Schahram Dustdar, editors. *Business Process Management Workshops, BPM 2006 International Workshops, BPD, BPI, ENEI, GPWW, DPM, semantics4ws, Vienna, Austria, September 4-7, 2006, Proceedings*, volume 4103 of *Lecture Notes in Computer Science*, 2006. Springer. ISBN 3-540-38444-8. (Cited on pages 293 and 297.)

[70] Marc Ehrig, Agnes Koschmider, and Andreas Oberweis. Measuring similarity between semantic business process models. In John F. Roddick and Annika Hinze, editors, *APCCM*, volume 67 of *CRPIT*, pages 71–80. Australian Computer Society, 2007. ISBN 1-920-68248-1. (Cited on page 58.)

[71] Daniel Elenius, Grit Denker, David B. Martin, Fred Gilham, John Khouri, Shahin Saadati, and Rukman Senanayake. The OWL-S editor - A development tool for semantic web services. In Asunción Gómez-Pérez and Jérôme Euzenat, editors, *The Semantic Web: Research and Applications, Second European Semantic Web Conference, ESWC 2005, Heraklion, Crete, Greece, May 29 - June 1, 2005, Proceedings*, volume 3532 of *Lecture Notes in Computer Science*, pages 78–92. Springer, 2005. ISBN 3-540-26124-9. doi: 10.1007/11431053_6. (Cited on pages 48, 51 and 149.)

[72] Rik Eshuis, Jochem Vonk, and Paul W. P. J. Grefen. Transactional process views. In Robert Meersman, Tharam S. Dillon, Pilar Herrero, Akhil Kumar, Manfred Reichert, Li Qing, Beng Chin Ooi, Ernesto Damiani, Douglas C. Schmidt, Jules White, Manfred Hauswirth, Pascal Hitzler, and Mukesh K. Mohania, editors, *On*

the Move to Meaningful Internet Systems: OTM 2011 - Confederated International Conferences: CoopIS, DOA-SVI, and ODBASE 2011, Hersonissos, Crete, Greece, October 17-21, 2011, Proceedings, Part I, volume 7044 of *Lecture Notes in Computer Science*, pages 119–136. Springer, 2011. ISBN 978-3-642-25108-5. doi: 10.1007/978-3-642-25109-2_9. (Cited on pages 7, 11, 39, 41, 47, 103, 104 and 127.)

[73] Peter H. Feiler and Watts S. Humphrey. Software process development and enactment: Concepts and definitions. In *ICSP*, pages 28–40, 1993. (Cited on page 154.)

[74] Simon Ferndriger, Abraham Bernstein, Jin Song Dong, Yuzhang Feng, Yuan-Fang Li, and Jane Hunter. Enhancing semantic web services with inheritance. In Sheth et al. [205], pages 162–177. ISBN 978-3-540-88563-4. doi: 10.1007/978-3-540-88564-1_11. (Cited on page 48.)

[75] Peter Fettke, Peter Loos, and Jörg Zwicker. Business process reference models: Survey and classification. In Christoph Bussler and Armin Haller, editors, *Business Process Management Workshops*, volume 3812, pages 469–483, 2005. ISBN 3-540-32595-6. (Cited on page 154.)

[76] Charles J. Fillmore. The case for case. In Emmon Bach and Robert T. Harms, editors, *Universals in Linguistic Theory*, pages 0–88. Holt, Rinehart and Winston, New York, 1968. (Cited on pages 35 and 37.)

[77] Christiane Floyd. A comparative evaluation of system development methods. In *Information Systems Design Methodologies: Improving the Practice*, pages 19–54, 1986. (Cited on pages 89 and 96.)

[78] André Freitas, João Gabriel Oliveira, Seán O'riain, João C P Da Silva, and Edward Curry. Querying linked data graphs using semantic relatedness: A vocabulary independent approach. *Data & Knowledge Engineering*, 88:126–141, 2013. (Cited on page 164.)

[79] E Gabrilovich and S Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the 20th international joint conference on Artifical intelligence*, pages 1606–1611, 2007. (Cited on page 164.)

[80] Bernhard Ganter and Rudolf Wille. *Formal concept analysis - mathematical foundations*. Springer, 1999. ISBN 978-3-540-62771-5. (Cited on pages 13 and 132.)

[81] Feng Gao and Sami Bhiri. Capability annotation of actions based on their textual descriptions. In Sumitra Reddy, editor, *2014 IEEE 23rd International WETICE Conference, WETICE 2014, Parma, Italy, 23-25 June, 2014*, pages 257–262. IEEE Computer Society, 2014. ISBN 978-1-4799-4249-7. doi: 10.1109/WETICE.2014.68. (Cited on pages 74 and 200.)

[82] Feng Gao and Wassim Derguech. Ubiquitous Service Capability Modeling and Similarity Based Searching. In *Web Information Systems Engineering - WISE 2011 and 2012 Workshops - Combined WISE 2011 and WISE 2012 Workshops, Sydney Australia, October 12-14, 2011 and Paphos, Cyprus, November 28-30, 2012, Revised Selected Papers*, pages 173–184, 2012. (Cited on page 193.)

[83] Feng Gao and Wassim Derguech. Ubiquitous Service Capability Modeling and Similarity Based Searching. In Armin Haller, Guangyan Huang, Zhisheng Huang, Hye-young Paik, and Quan Z. Sheng, editors, *Web Information Systems Engineering - WISE 2011 and 2012 Workshops - Combined WISE 2011 and WISE 2012 Workshops, Revised Selected Papers*, volume 7652 of *Lecture Notes in Computer Science*, pages 173–184, Sydney, Australia, 28-30 November 2012. Springer. ISBN 978-3-642-38332-8. (Cited on page 77.)

[84] Christian Gerth and Markus Luckey. Towards rich change management for business process models. *Softwaretechnik-Trends*, 32(4), 2012. (Cited on pages 186, 188 and 189.)

[85] Christian Gerth, Jochen Malte Küster, and Gregor Engels. Language-independent change management of process models. In Andy Schürr and Bran Selic, editors, *Model Driven Engineering Languages and Systems, 12th International Conference, MODELS 2009, Denver, CO, USA, October 4-9, 2009. Proceedings*, volume 5795 of *Lecture Notes in Computer Science*, pages 152–166. Springer, 2009. ISBN 978-3-642-04424-3. doi: 10.1007/978-3-642-04425-0_12. (Cited on page 186.)

[86] Christian Gerth, Markus Luckey, Jochen Malte Küster, and Gregor Engels. Detection of semantically equivalent fragments for business process model change management. In *2010 IEEE International Conference on Services Computing, SCC 2010, Miami, Florida, USA, July 5-10, 2010*, pages 57–64. IEEE Computer Society, 2010. ISBN 978-0-7695-4126-6. doi: 10.1109/SCC.2010.38. (Cited on pages 186, 188 and 189.)

[87] Robert Godin, Rokia Missaoui, and Hassan Alaoui. Incremental concept formation algorithms based on galois (concept) lattices. *Computational Intelligence*, 11:246–267, 1995. doi: 10.1111/j.1467-8640.1995.tb00031.x. (Cited on pages 149 and 201.)

[88] Karthik Gomadam, Ajith Ranabahu, and Amit Sheth. SA-REST: Semantic annotation of web resources. `http://www.w3.org/Submission/SA-REST/`, 2010. Accessed: 25/05/2014. (Cited on pages 7, 29, 30, 33, 38 and 98.)

[89] Florian Gottschalk. *Configurable process models*. PhD thesis, Technische Universiteit Eindhoven, Eindhoven, Netherlands, 2009. (Cited on pages 9, 19, 53, 64, 66, 67, 153, 157, 178 and 227.)

[90] Florian Gottschalk, Wil M. P. van der Aalst, and Monique H. Jansen-Vullers. Merging event-driven process chains. In Robert Meersman and Zahir Tari, editors, *On the Move to Meaningful Internet Systems: OTM 2008, OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008, Monterrey, Mexico, November 9-14, 2008, Proceedings, Part I*, volume 5331 of *Lecture Notes in Computer Science*, pages 418–426. Springer, 2008. ISBN 978-3-540-88870-3. doi: 10.1007/978-3-540-88871-0_28. (Cited on pages 9, 53, 153, 185, 186, 187, 188 and 189.)

[91] Florian Gottschalk, Wil M. P. van der Aalst, and Monique H. Jansen-Vullers. Mining reference process models and their configurations. In Robert Meersman, Zahir Tari, and Pilar Herrero, editors, *On the Move to Meaningful Internet Systems: OTM 2008 Workshops, OTM Confederated International Workshops and Posters, ADI, AWeSoMe, COMBEK, EI2N, IWSSA, MONET, OnToContent + QSI, ORM, PerSys, RDDS, SEMELS, and SWWS 2008, Monterrey, Mexico, November 9-14, 2008. Proceedings*, volume 5333 of *Lecture Notes in Computer Science*, pages 263–272. Springer, 2008. ISBN 978-3-540-88874-1. doi: 10.1007/978-3-540-88875-8_47. (Cited on pages 14, 152, 155 and 184.)

[92] Florian Gottschalk, Teun A. C. Wagemakers, Monique H. Jansen-Vullers, Wil M. P. van der Aalst, and Marcello La Rosa. Configurable process models: Experiences from a municipality case study. In Pascal van Eck, Jaap Gordijn, and Roel Wieringa, editors, *Advanced Information Systems Engineering, 21st International Conference, CAiSE 2009, Amsterdam, The Netherlands, June 8-12, 2009. Proceedings*, volume 5565 of *Lecture Notes in Computer Science*, pages 486–500. Springer, 2009. ISBN 978-3-642-02143-5. doi: 10.1007/978-3-642-02144-2_38. (Cited on pages 19, 89, 96, 178, 179, 180 and 227.)

[93] Daniel Gotzmann and Christian Lindig. Colibri-Java, 2007. URL `https://code.google.com/p/colibri-java/`. 2014-06-06. (Cited on pages 140 and 146.)

[94] GS1 US. United Nations Standard Products and Services Code (UNSPSC). `https://www.unspsc.org/`, 1998. Accessed: 25/05/2014. (Cited on pages 32, 38, 44, 50, 51 and 81.)

[95] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Comp. Syst.*, 29(7):1645–1660, 2013. doi: 10.1016/j.future.2013.01.010. (Cited on page 131.)

[96] MIT Process Handbook. Distribute via electronic store, 2001. URL `http://process.mit.edu/Activity.asp?ID=3446&Page=1`. (Cited on pages xx and 83.)

[97] F. Harmsen and M. Saeki. Comparison of four method engineering languages. In *Proceedings of the IFIP TC8, WG8.1/8.2 Working Conference on Method Engineering on Method Engineering : Principles of Method Construction and Tool Support: Principles of Method Construction and Tool Support*, pages 209–231, London, UK, UK, 1996. Chapman & Hall, Ltd. ISBN 0-412-79750-X. (Cited on page 90.)

[98] Souleiman Hasan, Edward Curry, Mauricio Banduk, and Seán O'Riain. Toward situation awareness for the semantic sensor web: Complex event processing with dynamic linked data enrichment. In Kerry Taylor, Arun Ayyagari, and David De Roure, editors, *Proceedings of the 4th International Workshop on Semantic Sensor Networks, SSN11, Bonn, Germany, October 23, 2011*, volume 839 of *CEUR Workshop Proceedings*, pages 69–81. CEUR-WS.org, 2011. (Cited on page 143.)

[99] Martin Hepp and Andreas Wechselberger. Ontonavierp: Ontology-supported navigation in ERP software documentation. In Sheth et al. [205], pages 764–776. ISBN 978-3-540-88563-4. doi: 10.1007/978-3-540-88564-1_49. (Cited on page 10.)

[100] Rudy Hirschheim and Heinz K. Klein. Four paradigms of information systems development. *Commun. ACM*, 32(10):1199–1216, 1989. doi: 10.1145/67933.67937. (Cited on page 90.)

[101] Information Administration, National Telecommunication. *Telecommunications : Glossary of Telecommunications Terms*. Lanham: Government Institutes, 1997. (Cited on page 78.)

[102] Paul Istoan. Defining composition operators for BPMN. In Thomas Gschwind, Flavio De Paoli, Volker Gruhn, and Matthias Book, editors, *Software Composition - 11th International Conference, SC 2012, Prague, Czech Republic, May 31 - June 1, 2012. Proceedings*, volume 7306 of *Lecture Notes in Computer Science*, pages 17–34. Springer, 2012. ISBN 978-3-642-30563-4. doi: 10.1007/978-3-642-30564-1_2. (Cited on pages 18 and 112.)

[103] Michael C. Jaeger, Gregor Rojec-Goldmann, and Gero Mühl. Qos aggregation for web service composition using workflow patterns. In *8th International Enterprise Distributed Object Computing Conference (EDOC 2004), 20-24 September 2004, Monterey, California, USA, Proceedings*, pages 149–159. IEEE Computer Society, 2004. ISBN 0-7695-2214-9. doi: 10.1109/EDOC.2004.10027. (Cited on page 116.)

[104] Lina Azleny Kamaruddin, Jun Shen, and Ghassan Beydoun. Evaluating usage of WSMO and OWL-S in semantic web services. In Aditya Ghose and Flavio Ferrarotti, editors, *Eighth Asia-Pacic Conference on Conceptual Modelling, APCCM*

2012, Melbourne, Australia, January 2012, volume 130 of *CRPIT*, pages 53–58. Australian Computer Society, 2012. ISBN 978-1-921770-11-1. (Cited on pages 32, 33, 38 and 48.)

[105] Paul Karaenke and Jörg Leukel. Towards ontology-based qos aggregation for composite web services. In Klaus-Peter Fähnrich and Bogdan Franczyk, editors, *Informatik 2010: Service Science - Neue Perspektiven für die Informatik, Beiträge der 40. Jahrestagung der Gesellschaft für Informatik e.V. (GI), Band 1, 27.09. - 1.10.2010, Leipzig*, volume 175 of *LNI*, pages 120–125. GI, 2010. ISBN 978-3-88579-269-7. (Cited on page 116.)

[106] Bartek Kiepuszewski, Arthur H. M. ter Hofstede, and Christoph Bussler. On structured workflow modelling. In Benkt Wangler and Lars Bergman, editors, *Advanced Information Systems Engineering, 12th International Conference CAiSE 2000, Stockholm, Sweden, June 5-9, 2000, Proceedings*, volume 1789 of *Lecture Notes in Computer Science*, pages 431–445. Springer, 2000. ISBN 3-540-67630-9. doi: 10.1007/3-540-45140-4_29. (Cited on page 112.)

[107] Christopher Klinkmüller, Ingo Weber, Jan Mendling, Henrik Leopold, and André Ludwig. Increasing recall of process model matching by improved activity label matching. In Daniel et al. [42], pages 211–218. ISBN 978-3-642-40175-6. doi: 10.1007/978-3-642-40176-3_17. (Cited on page 165.)

[108] Christopher Klinkmüller, Henrik Leopold, Ingo Weber, Jan Mendling, and André Ludwig. Listen to me: Improving process model matching through user feedback. In Shazia Wasim Sadiq, Pnina Soffer, and Hagen Völzer, editors, *Business Process Management - 12th International Conference, BPM 2014, Haifa, Israel, September 7-11, 2014. Proceedings*, volume 8659 of *Lecture Notes in Computer Science*, pages 84–100. Springer, 2014. ISBN 978-3-319-10171-2. doi: 10.1007/978-3-319-10172-9_6. (Cited on page 165.)

[109] Matthias Klusch, Benedikt Fries, and Katia P. Sycara. Automated semantic web service discovery with OWLS-MX. In *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006)*, pages 915–922, Hakodate, Japan, 8-12 May 2006. ACM, New York, NY, USA. ISBN 1-59593-303-4. (Cited on page 13.)

[110] Volkmar Koch, Simon Kuge, Reinhard Geissbauer, and Stefan Schrauf. *Industry 4.0: Opportunities and challenges of the industrial internet.* Strategy&, 2015. (Cited on page 2.)

[111] Jacek Kopecký, Tomas Vitvar, Carine Bournez, and Joel Farrell. SAWSDL: semantic annotations for WSDL and XML schema. *IEEE Internet Computing*, 11 (6):60–67, 2007. doi: 10.1109/MIC.2007.134. (Cited on pages 10, 11, 12 and 72.)

[112] Elena Kuss, Henrik Leopold, Han van der Aa, Heiner Stuckenschmidt, and Hajo A. Reijers. Probabilistic evaluation of process model matching techniques. In Isabelle Comyn-Wattiau, Katsumi Tanaka, Il-Yeol Song, Shuichiro Yamamoto, and Motoshi Saeki, editors, *Conceptual Modeling - 35th International Conference, ER 2016, Gifu, Japan, November 14-17, 2016, Proceedings*, volume 9974 of *Lecture Notes in Computer Science*, pages 279–292, 2016. ISBN 978-3-319-46396-4. doi: 10.1007/978-3-319-46397-1_22. (Cited on page 165.)

[113] Jochen Malte Küster, Jana Koehler, and Ksenia Ryndina. Improving business process models with reference models in business-driven development. In Eder and Dustdar [69]. ISBN 3-540-38444-8. (Cited on page 154.)

[114] Jochen Malte Küster, Christian Gerth, Alexander Förster, and Gregor Engels. Detecting and resolving process model differences in the absence of a change log. In Dumas et al. [66], pages 244–260. ISBN 978-3-540-85757-0. doi: 10.1007/978-3-540-85758-7_19. (Cited on pages 186, 188 and 189.)

[115] Jochen Malte Küster, Christian Gerth, Alexander Förster, and Gregor Engels. A tool for process merging in business-driven development. In Zohra Bellahsene, Carson Woo, Ela Hunt, Xavier Franch, and Remi Coletta, editors, *Proceedings of the Forum at the CAiSE'08 Conference, Montpellier, France, June 18-20, 2008*, volume 344 of *CEUR Workshop Proceedings*, pages 89–92. CEUR-WS.org, 2008. (Cited on pages 153, 186, 188 and 189.)

[116] Marcello La Rosa. *Managing Variability in Process-Aware Information Systems*. PhD thesis, Queensland University of Technology, Brisbane, Australia, April 2009. (Cited on pages xix, xx, 8, 14, 53, 64, 65, 66, 67, 152, 153, 154 and 155.)

[117] Marcello La Rosa and Marlon Dumas. Configurable process models: How to adopt standard practices in your how way? BPTrends Newsletter, November 2008. (Cited on pages xix and 64.)

[118] Marcello La Rosa, Johannes Lux, Stefan Seidel, Marlon Dumas, and Arthur H. M. ter Hofstede. Questionnaire-driven configuration of reference process models. In *Proceedings of the 19th International Conference on Advanced Information Systems Engineering, CAiSE*, pages 424–438, 2007. (Cited on pages 64 and 67.)

[119] Marcello La Rosa, Marlon Dumas, Arthur H. M. ter Hofstede, Jan Mendling, and Florian Gottschalk. Beyond control-flow: Extending business process configuration

to roles and objects. In Qing Li, Stefano Spaccapietra, Eric S. K. Yu, and Antoni Olivé, editors, *Conceptual Modeling - ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008. Proceedings*, volume 5231 of *Lecture Notes in Computer Science*, pages 199–215. Springer, 2008. ISBN 978-3-540-87876-6. doi: 10.1007/978-3-540-87877-3_16. (Cited on pages 8 and 53.)

[120] Marcello La Rosa, Wil M. P. van der Aalst, Marlon Dumas, and Arthur H. M. ter Hofstede. Questionnaire-based variability modeling for system configuration. *Software and System Modeling*, 8(2):251–274, 2009. doi: 10.1007/s10270-008-0090-3. (Cited on pages 8, 9 and 53.)

[121] Marcello La Rosa, Marlon Dumas, Reina Uba, and Remco M. Dijkman. Merging business process models. In Robert Meersman, Tharam S. Dillon, and Pilar Herrero, editors, *On the Move to Meaningful Internet Systems: OTM 2010 - Confederated International Conferences: CoopIS, IS, DOA and ODBASE, Hersonissos, Crete, Greece, October 25-29, 2010, Proceedings, Part I*, volume 6426 of *Lecture Notes in Computer Science*, pages 96–113. Springer, 2010. ISBN 978-3-642-16933-5. doi: 10.1007/978-3-642-16934-2_10. (Cited on pages 153, 158, 164, 167, 185, 187, 188 and 189.)

[122] Marcello La Rosa, Hajo A. Reijers, Wil M. P. van der Aalst, Remco M. Dijkman, Jan Mendling, Marlon Dumas, and Luciano García-Bañuelos. Apromore: An advanced process model repository. *Expert Syst. Appl.*, 38(6):7029–7040, 2011. (Cited on pages 57 and 67.)

[123] Marcello La Rosa, Marlon Dumas, Reina Uba, and Remco M. Dijkman. Business Process Model Merging: An Approach to Business Process Consolidation. *ACM Trans. Softw. Eng. Methodol.*, 22(2):11, 2013. (Cited on pages 5, 8, 14, 53, 152, 153, 158, 164, 167, 185, 187, 188 and 189.)

[124] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business & Information Systems Engineering*, 6(4):239–242, 2014. doi: 10.1007/s12599-014-0334-4. (Cited on pages 2 and 131.)

[125] Jon Lathem, Karthik Gomadam, and Amit P. Sheth. SA-REST and (s)mashups : Adding semantics to restful services. In *Proceedings of the First IEEE International Conference on Semantic Computing (ICSC 2007), September 17-19, 2007, Irvine, California, USA*, pages 469–476. IEEE Computer Society, 2007. ISBN 0-7695-2997-6. doi: 10.1109/ICSC.2007.94. (Cited on pages 10, 11, 12 and 72.)

[126] Timothy Lebo and Gregory Todd Williams. Converting governmental datasets into linked data. In *I-SEMANTICS*. ACM, 2010. ISBN 978-1-4503-0014-8. (Cited on page 81.)

[127] Jay Lee, Hung-An Kao, and Shanhu Yang. Service innovation and smart analytics for industry 4.0 and big data environment. *Procedia {CIRP}*, 16:3 – 8, 2014. ISSN 2212-8271. doi: http://dx.doi.org/10.1016/j.procir.2014.02.001. Product Services Systems and Value Creation. Proceedings of the 6th {CIRP} Conference on Industrial Product-Service Systems. (Cited on page 2.)

[128] Laurent Lefort, T Meyer, and Kerry Taylor. Review of semantic enablement techniques used in geospatial and semantic standards for legacy and opportunistic mashups. In *Australasian Ontology Workshop 2009 (AOW 2009)*, volume 112 of *CRPIT*, pages 17–26, 2009. (Cited on page 34.)

[129] Henrik Leopold, Jan Mendling, and Artem Polyvyanyy. Generating natural language texts from business process models. In Jolita Ralyté, Xavier Franch, Sjaak Brinkkemper, and Stanislaw Wrycza, editors, *Advanced Information Systems Engineering - 24th International Conference, CAiSE 2012, Gdansk, Poland, June 25-29, 2012. Proceedings*, volume 7328 of *Lecture Notes in Computer Science*, pages 64–79. Springer, 2012. ISBN 978-3-642-31094-2. doi: 10.1007/978-3-642-31095-9_5. (Cited on page 127.)

[130] Henrik Leopold, Mathias Niepert, Matthias Weidlich, Jan Mendling, Remco M. Dijkman, and Heiner Stuckenschmidt. Probabilistic optimization of semantic process model matching. In Alistair P. Barros, Avigdor Gal, and Ekkart Kindler, editors, *Business Process Management - 10th International Conference, BPM 2012, Tallinn, Estonia, September 3-6, 2012. Proceedings*, volume 7481 of *Lecture Notes in Computer Science*, pages 319–334. Springer, 2012. ISBN 978-3-642-32884-8. doi: 10.1007/978-3-642-32885-5_25. (Cited on page 165.)

[131] Henrik Leopold, Sergey Smirnov, and Jan Mendling. On the refactoring of activity labels in business process models. *Inf. Syst.*, 37(5):443–459, 2012. doi: 10.1016/j.is.2012.01.004. (Cited on pages 74 and 199.)

[132] Henrik Leopold, Jan Mendling, Hajo A. Reijers, and Marcello La Rosa. Simplifying process model abstraction: Techniques for generating model names. *Inf. Syst.*, 39:134–151, 2014. doi: 10.1016/j.is.2013.06.007. (Cited on pages xix, 43, 47 and 127.)

[133] Chen Li, Manfred Reichert, and Andreas Wombacher. Discovering reference models by mining process variants using a heuristic approach. In Umeshwar Dayal, Johann Eder, Jana Koehler, and Hajo A. Reijers, editors, *Business Process Management, 7th International Conference, BPM 2009, Ulm, Germany, September 8-10, 2009. Proceedings*, volume 5701 of *Lecture Notes in Computer Science*, pages 344–362. Springer, 2009. ISBN 978-3-642-03847-1. doi: 10.1007/978-3-642-03848-8_23. (Cited on page 154.)

[134] Christian Lindig. Fast concept analysis. In *Working with Conceptual Structures – Contributions to ICCS 2000*, pages 152–161. Shaker Verlag, 2000. (Cited on page 146.)

[135] Ling Liu and M. Tamer Özsu, editors. *Encyclopedia of Database Systems*. Springer US, 2009. ISBN 978-0-387-35544-3, 978-0-387-39940-9. (Cited on pages 299 and 306.)

[136] Ruopeng Lu and Shazia Sadiq. Managing process variants as an information resource. In *Proceedings of the 4th International Conference on Business Process Management*, BPM'06, pages 426–431, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-38901-6, 978-3-540-38901-9. doi: 10.1007/11841760_34. (Cited on pages 54 and 67.)

[137] Ruopeng Lu and Shazia Wasim Sadiq. On the discovery of preferred work practice through business process variants. In Christine Parent, Klaus-Dieter Schewe, Veda C. Storey, and Bernhard Thalheim, editors, *ER*, volume 4801 of *Lecture Notes in Computer Science*, pages 165–180. Springer, 2007. ISBN 978-3-540-75562-3. (Not cited.)

[138] Ruopeng Lu, Shazia Wasim Sadiq, and Guido Governatori. On managing business processes variants. *Data Knowl. Eng.*, 68(7):642–664, 2009. (Cited on pages 54 and 67.)

[139] Ruopeng Lu, Shazia Wasim Sadiq, Guido Governatori, and Xiaoping Yang. Defining adaptation constraints for business process variants. In Witold Abramowicz, editor, *BIS*, volume 21 of *Lecture Notes in Business Information Processing*, pages 145–156. Springer, 2009. ISBN 978-3-642-01189-4. (Cited on page 61.)

[140] Zhilei Ma, Branimir Wetzstein, Darko Anicic, Stijn Heymans, and Frank Leymann. Semantic business process repository. In Martin Hepp, Knut Hinkelmann, Dimitris Karagiannis, Rüdiger Klein, and Nenad Stojanovic, editors, *SBPM*, volume 251 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007. (Cited on pages 55, 58, 66 and 67.)

[141] Michele Mancioppi, Olha Danylevych, Dimka Karastoyanova, and Frank Leymann. Towards classification criteria for process fragmentation techniques. In Florian Daniel, Kamel Barkaoui, and Schahram Dustdar, editors, *Business Process Management Workshops (1)*, volume 99 of *Lecture Notes in Business Information Processing*, pages 1–12. Springer, 2011. ISBN 978-3-642-28107-5. (Cited on pages 59 and 68.)

[142] James Manyika, Michael Chui, Jacques Bughin, Richard Dobbs, Peter Bisson, and Alex Marrs. Disruptive technologies: Advances that will transform life, business, and the global economy. Technical report, McKinsey Global Institute, May 2013. (Cited on page 131.)

[143] Ivan Markovic, Alessandro Costa Pereira, and Nenad Stojanovic. A framework for querying in business process modelling. In Martin Bichler, Thomas Hess, Helmut Krcmar, Ulrike Lechner, Florian Matthes, Arnold Picot, Benjamin Speitkamp, and Petra Wolf, editors, *Multikonferenz Wirtschaftsinformatik*. GITO-Verlag, Berlin, 2008. ISBN 978-3-940019-34-9. (Cited on pages 56, 66 and 67.)

[144] David Martin, Massimo Paolucci, and Matthias Wagner. Bringing Semantic Annotations to Web Services: OWL-S from the SAWSDL Perspective. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 340–352, Busan, Korea, 11-15 November 2007. Springer Berlin Heidelberg. ISBN 978-3-540-76297-3. (Cited on pages 10, 11, 12, 34, 38 and 72.)

[145] David L. Martin, Mark H. Burstein, Drew V. McDermott, Sheila A. McIlraith, Massimo Paolucci, Katia P. Sycara, Deborah L. McGuinness, Evren Sirin, and Naveen Srinivasan. Bringing semantics to web services with OWL-S. *World Wide Web*, 3:243–277, 2007. doi: 10.1007/s11280-007-0033-x. (Cited on pages 12 and 13.)

[146] Massachusetts Institute of Technology. MIT process hanbook. `http://process.mit.edu/`, 2001. Accessed: 25/05/2014. (Cited on pages 44, 51, 81, 82 and 83.)

[147] Jan Mendling and Markus Nüttgens. EPC syntax validation with XML schema languages. In Markus Nüttgens and Frank J. Rump, editors, *EPK 2003 - Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, Proceedings des GI-Workshops und Arbeitskreistreffens (Bamberg, Oktober 2003)*, pages 19–30. GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, 2003. (Cited on pages 106, 154 and 169.)

[148] Jan Mendling and Markus Nüttgens. EPC markup language (EPML): an xml-based interchange format for event-driven process chains (EPC). *Inf. Syst. E-Business Management*, 4(3):245–263, 2006. doi: 10.1007/s10257-005-0026-1. (Cited on pages 106, 154 and 176.)

[149] Jan Mendling and Carlo Simon. Business process design by view integration. In Eder and Dustdar [69], pages 55–64. ISBN 3-540-38444-8. doi: 10.1007/11837862_7. (Cited on pages 74, 185 and 199.)

[150] Jan Mendling, H. M. W. Verbeek, Boudewijn F. van Dongen, Wil M. P. van der Aalst, and Gustaf Neumann. Detection and prediction of errors in epcs of the sap reference model. *Data Knowl. Eng.*, 64(1):312–329, 2008. (Cited on pages 57, 58 and 66.)

[151] Jan Mendling, Jan Recker, and Hajo A. Reijers. On the usage of labels and icons in business process modeling. *IJISMD*, 1(2):40–58, 2010. doi: 10.4018/jismd.2010040103. (Cited on page 10.)

[152] Jan Mendling, Hajo A. Reijers, and Jan Recker. Activity labeling in process modeling: Empirical insights and recommendations. *Inf. Syst.*, 35(4):467–482, 2010. doi: 10.1016/j.is.2009.03.009. (Cited on page 9.)

[153] Sonia Ben Mokhtar, Davy Preuveneers, Nikolaos Georgantas, Valérie Issarny, and Yolande Berbers. EASY: Efficient semAntic Service discoverY in pervasive computing environments with QoS and context support . *Journal of Systems and Software*, 81(5):785 – 808, 2008. (Cited on pages 8, 12, 13, 37, 47, 48, 50, 51, 52 and 149.)

[154] Daniel Laurence Moody, Patrick Heymans, and Raimundas Matulevicius. Visual syntax does matter: improving the cognitive effectiveness of the *i** visual notation. *Requir. Eng.*, 15(2):141–175, 2010. doi: 10.1007/s00766-010-0100-1. (Cited on pages 89 and 96.)

[155] John Mylopoulos. Information modeling in the time of the revolution. *Information Systems*, 23(3-4):127–155, May 1998. ISSN 03064379. doi: 10.1016/s0306-4379(98)00005-2. (Cited on page 79.)

[156] Cuntz Nicolas and Kindler Ekkart. EPC Tools, 2006. URL `http://www2.cs.uni-paderborn.de/cs/kindler/research/EPCTools/`. (Cited on page 176.)

[157] Nurzhan Nurseitov, Michael Paulson, Randall Reynolds, and Clemente Izurieta. Comparison of JSON and XML data interchange formats: A case study. In Dunren Che, editor, *Proceedings of the ISCA 22nd International Conference on Computer Applications in Industry and Engineering, CAINE 2009, November 4-6, 2009, Hilton San Francisco Fisherman's Wharf, San Francisco, California, USA*, pages 157–162. ISCA, 2009. ISBN 978-1-880843-73-4. (Cited on page 37.)

[158] Philipa Oaks. *Enabling Ad-hoc Interaction with Electronic Services*. PhD thesis, Faculty of Information Technology, Queensland University of Technology, Brisbane, Australia, July 2005. (Cited on page 95.)

[159] Phillipa Oaks, Arthur H. M. ter Hofstede, and David Edmond. Capabilities: Describing What Services Can Do. In Maria E. Orlowska, Sanjiva Weerawarana,

Mike P. Papazoglou, and Jian Yang, editors, *ICSOC*, volume 2910 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2003. ISBN 3-540-20681-7. (Cited on pages 7, 8, 10, 11, 16, 28, 29, 30, 35, 36, 37, 38, 47, 71, 72, 73, 74 and 96.)

[160] OASIS SOA Reference Model (SOA-RM) Technical Committee (TC). OASIS reference model for service oriented architecture 1.0. http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf, 2006. Accessed: 25/05/2014. (Cited on pages 28, 29, 71 and 72.)

[161] Office of Management and Budget. North American Industry Classification System (NAICS). http://www.census.gov/eos/www/naics/, 1997. Accessed: 25/05/2014. (Cited on pages 32, 38, 44, 50, 51, 81 and 82.)

[162] Dirk Ohst, Michael Welle, and Udo Kelter. Differences between versions of UML diagrams. In Jukka Paakki and Paola Inverardi, editors, *Proceedings of the 11th ACM SIGSOFT Symposium on Foundations of Software Engineering 2003 held jointly with 9th European Software Engineering Conference, ESEC/FSE 2003, Helsinki, Finland, September 1-5, 2003*, pages 227–236. ACM, 2003. doi: 10.1145/940071.940102. (Cited on page 185.)

[163] OMG - Object Management Group. OMG - Object Management Group. http://www.omg.org/, 1989. (Cited on page 106.)

[164] Seán O'Riain, Barry Coughlan, Paul Buitelaar, Thierry Declerck, Uli Krieger, and Susan Marie Thomas. Cross-lingual querying and comparison of linked financial and business data. In Philipp Cimiano, Miriam Fernández, Vanessa Lopez, Stefan Schlobach, and Johanna Völker, editors, *The Semantic Web: ESWC 2013 Satellite Events - ESWC 2013 Satellite Events, Montpellier, France, May 26-30, 2013, Revised Selected Papers*, volume 7955 of *Lecture Notes in Computer Science*, pages 242–247. Springer, 2013. ISBN 978-3-642-41241-7. doi: 10.1007/978-3-642-41242-4_32. (Cited on page 162.)

[165] Nathaniel Palmer. Activity. In Liu and Özsu [135], page 41. ISBN 978-0-387-35544-3, 978-0-387-39940-9. doi: 10.1007/978-0-387-39940-9_822. (Cited on page 28.)

[166] Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, and Katia P. Sycara. Semantic matching of web services capabilities. In Ian Horrocks and James A. Hendler, editors, *The Semantic Web - ISWC 2002, First International Semantic Web Conference, Sardinia, Italy, June 9-12, 2002, Proceedings*, volume 2342 of *Lecture Notes in Computer Science*, pages 333–347. Springer, 2002. ISBN 3-540-43760-6. doi: 10.1007/3-540-48005-6_26. (Cited on pages 8, 39 and 45.)

[167] Carla Marques Pereira, Artur Caetano, and Pedro Manuel Antunes Sousa. Using a controlled vocabulary to support business process design. In Joseph Barjis, Tillal Eldabi, and Ashish Gupta, editors, *Enterprise and Organizational Modeling and Simulation - 7th International Workshop, EOMAS 2011, held at CAiSE 2011, London, UK, June 20-21, 2011. Selected Papers*, volume 88 of *Lecture Notes in Business Information Processing*, pages 74–84. Springer, 2011. ISBN 978-3-642-24174-1. doi: 10.1007/978-3-642-24175-8_6. (Cited on page 164.)

[168] Maja Pesic, Helen Schonenberg, and Wil M. P. van der Aalst. Declare demo: A constraint-based workflow management system. In Ana Karla A. de Medeiros and Barbara Weber, editors, *BPM (Demos)*, volume 489 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009. (Cited on page 61.)

[169] Maja Pesic, Dragan Bosnacki, and Wil M. P. van der Aalst. Enacting declarative languages using ltl: Avoiding errors and improving performance. In Jaco van de Pol and Michael Weber, editors, *SPIN*, volume 6349 of *Lecture Notes in Computer Science*, pages 146–161. Springer, 2010. ISBN 978-3-642-16163-6. (Cited on page 61.)

[170] James Lyle Peterson. *Petri Net Theory and the Modeling of Systems*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1981. ISBN 0136619835. (Cited on pages 18, 20, 45, 112, 129 and 193.)

[171] Axel Polleres. From SPARQL to rules (and back). In Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy, editors, *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 787–796. ACM, 2007. ISBN 978-1-59593-654-7. doi: 10.1145/1242572.1242679. (Cited on page 196.)

[172] Artem Polyvyanyy, Sergey Smirnov, and Mathias Weske. Reducing complexity of large epcs. In Peter Loos, Markus Nüttgens, Klaus Turowski, and Dirk Werth, editors, *Modellierung betrieblicher Informationssysteme - Modellierung zwischen SOA und Compliance Management - 27.-28. November 2008 Saarbrücken, Germany*, volume 141 of *LNI*, pages 195–207. GI, 2008. (Cited on pages xix, 7, 11, 12, 39, 41, 42, 47, 104 and 127.)

[173] Artem Polyvyanyy, Luciano García-Bañuelos, and Marlon Dumas. Structuring acyclic process models. *Inf. Syst.*, 37(6):518–538, 2012. doi: 10.1016/j.is.2011.10.005. (Cited on page 128.)

[174] Artem Polyvyanyy, Luciano García-Bañuelos, Dirk Fahland, and Mathias Weske. Maximal structuring of acyclic process models. *Comput. J.*, 57(1):12–35, 2014. doi: 10.1093/comjnl/bxs126. (Cited on page 128.)

[175] Erhard Rahm and Philip A. Bernstein. A survey of approaches to automatic schema matching. *VLDB J.*, 10(4):334–350, 2001. doi: 10.1007/s007780100057. (Cited on pages 74, 185 and 199.)

[176] Maryam Razavian and Ramtin Khosravi. Modeling variability in business process models using uml. In *ITNG*, pages 82–87. IEEE Computer Society, 2008. (Cited on pages xix, 62, 63, 66 and 67.)

[177] Jan Recker. Conceptual model evaluation. towards more paradigmatic rigor. In J. Castro and E. Teniente, editors, *Proceedings of the CAiSE'05 Workshops*, volume 1 of *CEUR Workshop Proceedings*, pages 569–580. CEUR-WS.org, 2005. (Cited on pages xxiii, 17, 88, 89, 90, 91 and 95.)

[178] Jan Recker and Jan Mendling. The state of the art of business process management research as published in the BPM conference - recommendations for progressing the field. *Business & Information Systems Engineering*, 58(1):55–72, 2016. doi: 10.1007/s12599-015-0411-3. (Cited on page 27.)

[179] Manfred Reichert, Jens Kolb, Ralph Bobrik, and Thomas Bauer. Enabling personalized visualization of large business processes through parameterizable views. In Sascha Ossowski and Paola Lecca, editors, *Proceedings of the ACM Symposium on Applied Computing, SAC 2012, Riva, Trento, Italy, March 26-30, 2012*, pages 1653–1660. ACM, 2012. ISBN 978-1-4503-0857-1. doi: 10.1145/2245276.2232043. (Cited on pages xix, 7, 11, 12, 39, 40, 41, 47, 103, 104 and 127.)

[180] Stephen Rockwell and Akhilesh Bajaj. Applying cognitive theories to evaluate conceptual models in systems analysis. *JITR*, 3(1):55–72, 2010. doi: 10.4018/jitr.2010010105. (Cited on page 91.)

[181] Dumitru Roman, Uwe Keller, Holger Lausen, Jos de Bruijn, Rubén Lara, Michael Stollberg, Axel Polleres, Cristina Feier, Christoph Bussler, and Dieter Fensel. Web Service Modeling Ontology. *Applied Ontology*, 1(1):77–106, 2005. (Cited on page 31.)

[182] Dumitru Roman, Uwe Keller, Holger Lausen, Jos de Bruijn, Rubén Lara, Michael Stollberg, Axel Polleres, Cristina Feier, Cristoph Bussler, and Dieter Fensel. Web service modeling ontology. *Appl. Ontol.*, 1(1):77–106, January 2005. ISSN 1570-5838. (Cited on page 56.)

[183] Dumitru Roman, Jos de Bruijn, Adrian Mocan, Holger Lausen, John Domingue, Christoph Bussler, and Dieter Fensel. WWW: wsmo, wsml, and WSMX in a nutshell. In Riichiro Mizoguchi, Zhongzhi Shi, and Fausto Giunchiglia, editors,

The Semantic Web - ASWC 2006, First Asian Semantic Web Conference, Beijing, China, September 3-7, 2006, Proceedings, volume 4185 of *Lecture Notes in Computer Science*, pages 516–522. Springer, 2006. ISBN 3-540-38329-8. doi: 10.1007/11836025_49. (Cited on pages 10, 11, 12 and 72.)

[184] Michael Rosemann and Wil M. P. van der Aalst. A configurable reference modelling language. *Inf. Syst.*, 32(1):1–23, 2007. (Cited on pages 8, 14, 57, 63, 67, 152, 153, 154, 155 and 157.)

[185] Matti Rossi and Sjaak Brinkkemper. Complexity metrics for systems development methods and techniques. *Inf. Syst.*, 21(2):209–227, 1996. doi: 10.1016/0306-4379(96)00012-9. (Cited on page 91.)

[186] James Rumbaugh, Ivar Jacobson, and Grady Booch. *Unified Modeling Language Reference Manual, The (2nd Edition)*. Pearson Higher Education, 2004. ISBN 0321245628. (Cited on page 106.)

[187] Per Runeson and Martin Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, 2009. doi: 10.1007/s10664-008-9102-8. (Cited on pages 96, 97, 124, 125, 180 and 181.)

[188] Nick Russell, Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, and Petia Wohed. On the suitability of UML 2.0 activity diagrams for business process modelling. In Markus Stumptner, Sven Hartmann, and Yasushi Kiyoki, editors, *Conceptual Modelling 2006, Third Asia-Pacific Conference on Conceptual Modelling (APCCM 2005), Hobart, Tasmania, Australia, January 16-19 2006*, volume 53 of *CRPIT*, pages 95–104. Australian Computer Society, 2006. ISBN 1-920-68235-X. doi: 10.1145/1151855.1151866. (Cited on page 106.)

[189] Shazia W. Sadiq, Wasim Sadiq, and Maria E. Orlowska. Pockets of flexibility in workflow specification. In Hideko S. Kunii, Sushil Jajodia, and Arne Sølvberg, editors, *ER*, volume 2224 of *Lecture Notes in Computer Science*, pages 513–526. Springer, 2001. ISBN 3-540-42866-6. (Cited on pages 60 and 67.)

[190] Shazia Wasim Sadiq, Maria E. Orlowska, and Wasim Sadiq. Specification and validation of process constraints for flexible workflows. *Inf. Syst.*, 30(5):349–378, 2005. (Cited on pages 60 and 67.)

[191] Wasim Sadiq and Maria E. Orlowska. On correctness issues in conceptual modelling of workflows. In Robert D. Galliers, Ciaran Murphy, Sven A. Carlsson, Claudia Loebbecke, Hans Robert Hansen, and Ramon O'Callaghan, editors, *Proceedings of the Fifth European Conference on Information Systems, ECIS 1997,*

Cork, UK, 1997, pages 943–964. Cork Publishing Ltd, 1997. (Cited on pages 162 and 163.)

[192] Wasim Sadiq and Maria E. Orlowska. Analyzing process models using graph reduction techniques. *Information Systems*, 25(2):117 – 134, 2000. ISSN 0306-4379. doi: http://dx.doi.org/10.1016/S0306-4379(00)00012-0. The 11th International Conference on Advanced Information System Engineering. (Cited on page 54.)

[193] Ratnesh Sahay, Ronan Fox, Antoine Zimmermann, Axel Polleres, and Manfred Hauswirth. A methodological approach for ontologising and aligning health level seven (HL7) applications. In A Min Tjoa, Gerald Quirchmayr, Ilsun You, and Lida Xu, editors, *Availability, Reliability and Security for Business, Enterprise and Health Information Systems - IFIP WG 8.4/8.9 International Cross Domain Conference and Workshop, ARES 2011, Vienna, Austria, August 22-26, 2011. Proceedings*, volume 6908 of *Lecture Notes in Computer Science*, pages 102–117. Springer, 2011. ISBN 978-3-642-23299-2. doi: 10.1007/978-3-642-23300-5_9. (Cited on page 162.)

[194] Sherif Sakr and Ahmed Awad. A framework for querying graph-based business process models. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 1297–1300, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-799-8. doi: 10.1145/1772690.1772906. (Cited on pages 56, 66 and 67.)

[195] SAP. Automatically Approve Travel Requests, . URL http://help.sap.com/saphelp_rem10/helpdata/en/f5/4fe23cab43ba5be10000000a114084/content.htm. (Cited on pages 6 and 159.)

[196] SAP. Approve Travel Request, . URL http://help.sap.com/saphelp_rem10/helpdata/en/04/928b0b46f311d189470000e829fbbd/content.htm. (Cited on pages 6 and 159.)

[197] SAP. Workflow Scenarios in Travel Management, . URL http://help.sap.com/saphelp_rem10/helpdata/en/d5/202038541ec006e10000009b38f8cf/frameset.htm. (Cited on pages xix, xx, 6, 159 and 162.)

[198] SAWSDL Working Group. SA-WSDL: Semantic Annotations for WSDL. http://www.w3.org/2002/ws/sawsdl/, 2007. Accessed: 25/05/2014. (Cited on pages 7, 29, 30, 33, 38 and 98.)

[199] August-Wilhem Scheer and Kristof Schneider. ARIS Architecture of Integrated Information Systems. *Bernus Peter Mertins Kai Schmidt Gunter*, 2006. (Cited on page 72.)

[200] Guus Schreiber and Mike Dean. OWL web ontology language reference. W3C recommendation, W3C, February 2004. http://www.w3.org/TR/2004/REC-owl-ref-20040210/. (Cited on page 31.)

[201] David Schumm, Dimitrios Dentsas, Michael Hahn, Dimka Karastoyanova, Frank Leymann, and Mirko Sonntag. Web service composition reuse through shared process fragment libraries. In Marco Brambilla, Takehiro Tokuda, and Robert Tolksdorf, editors, *ICWE*, volume 7387 of *Lecture Notes in Computer Science*, pages 498–501. Springer, 2012. ISBN 978-3-642-31752-1. (Cited on pages 59 and 68.)

[202] Serhiy Yevtushenko and contributors. Conexp, 2013. URL `http://sourceforge.net/projects/conexp/`. (Cited on pages 134, 140 and 144.)

[203] John Sheridan and Jeni Tennison. Linking UK Government Data. In *LDOW 2010*, 2010. (Cited on page 81.)

[204] Amit P. Sheth, Karthik Gomadam, and Jon Lathem. SA-REST: semantically interoperable and easier-to-use services and mashups. *IEEE Internet Computing*, 11(6):91–94, 2007. doi: 10.1109/MIC.2007.133. (Cited on pages 33 and 34.)

[205] Amit P. Sheth, Steffen Staab, Mike Dean, Massimo Paolucci, Diana Maynard, Timothy W. Finin, and Krishnaprasad Thirunarayan, editors. *The Semantic Web - ISWC 2008, 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008. Proceedings*, volume 5318 of *Lecture Notes in Computer Science*, 2008. Springer. ISBN 978-3-540-88563-4. (Cited on pages 288 and 291.)

[206] Keng Siau and Matti Rossi. Evaluation techniques for systems analysis and design modelling methods - a review and comparative analysis. *Inf. Syst. J.*, 21(3):249–268, 2011. doi: 10.1111/j.1365-2575.2007.00255.x. (Cited on pages xxiii, 88, 89, 95, 96 and 99.)

[207] Adina Sirbu, Annapaola Marconi, Marco Pistore, Hanna Eberle, Frank Leymann, and Tobias Unger. Dynamic composition of pervasive process fragments. In *ICWS*, pages 73–80. IEEE Computer Society, 2011. ISBN 978-1-4577-0842-8. (Cited on page 59.)

[208] Sergey Smirnov, Remco M. Dijkman, Jan Mendling, and Mathias Weske. Meronymy-based aggregation of activities in business process models. In Jeffrey Parsons, Motoshi Saeki, Peretz Shoval, Carson C. Woo, and Yair Wand, editors, *Conceptual Modeling - ER 2010, 29th International Conference on Conceptual Modeling, Vancouver, BC, Canada, November 1-4, 2010. Proceedings*, volume 6412

of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2010. ISBN 978-3-642-16372-2. doi: 10.1007/978-3-642-16373-9_1. (Cited on pages xix, 7, 11, 12, 39, 44, 46, 47, 82, 104, 105, 111, 127 and 128.)

[209] Sergey Smirnov, Hajo A. Reijers, and Mathias Weske. A semantic approach for business process model abstraction. In Haralambos Mouratidis and Colette Rolland, editors, *Advanced Information Systems Engineering - 23rd International Conference, CAiSE 2011, London, UK, June 20-24, 2011. Proceedings*, volume 6741 of *Lecture Notes in Computer Science*, pages 497–511. Springer, 2011. ISBN 978-3-642-21639-8. doi: 10.1007/978-3-642-21640-4_37. (Cited on pages 7, 11, 12, 39, 42, 47, 104 and 127.)

[210] Sergey Smirnov, Hajo A. Reijers, Mathias Weske, and Thijs Nugteren. Business process model abstraction: a definition, catalog, and survey. *Distributed and Parallel Databases*, 30(1):63–99, 2012. (Cited on pages 39, 40 and 127.)

[211] Naveen Srinivasan, Massimo Paolucci, and Katia Sycara. Adding OWL-S to UDDI, Implementation and Throughput. In *First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004)*, July 2004. (Cited on pages 8, 48, 49, 50, 51, 52 and 149.)

[212] Naveen Srinivasan, Massimo Paolucci, and Katia P. Sycara. Semantic web service discovery in the OWL-S IDE. In *39th Hawaii International International Conference on Systems Science (HICSS-39 2006), CD-ROM / Abstracts Proceedings, 4-7 January 2006, Kauai, HI, USA*. IEEE Computer Society, 2006. ISBN 0-7695-2507-5. doi: 10.1109/HICSS.2006.431. (Cited on pages 12 and 13.)

[213] Alexander Streit, Binh Pham, and Ross Brown. Visualization support for managing large business process specifications. In Wil M. P. van der Aalst, Boualem Benatallah, Fabio Casati, and Francisco Curbera, editors, *Business Process Management*, volume 3649, pages 205–219, 2005. ISBN 3-540-28238-6. (Cited on page 129.)

[214] Fedor Strok and Alexey Neznanov. Comparing and analyzing the computational complexity of FCA algorithms. In Paula Kotzé, Alta van der Merwe, and Aurona Gerber, editors, *Proceedings of the 2010 Annual Conference of the South African Institute of Computer Scientists and Information Technologists, SAICSIT Conf. 2010, Bela Bela, South Africa, October 11-13, 2010*, ACM International Conference Proceeding Series, pages 417–420. ACM, 2010. ISBN 978-1-60558-950-3. doi: 10.1145/1899503.1899557. (Cited on page 49.)

[215] Vijayan Sugumaran and Veda C. Storey. A semantic-based approach to component retrieval. *DATA BASE*, 34(3):8–24, 2003. doi: 10.1145/937742.937745. (Cited on page 196.)

[216] Katia P. Sycara, Seth Widoff, Matthias Klusch, and Jianguo Lu. Larks: Dynamic matchmaking among heterogeneous software agents in cyberspace. *Autonomous Agents and Multi-Agent Systems*, 5(2):173–203, 2002. doi: 10.1023/A:1014897210525. (Cited on pages 7, 8, 29, 30, 47 and 96.)

[217] Katia P. Sycara, Massimo Paolucci, Julien Soudry, and Naveen Srinivasan. Dynamic discovery and coordination of agent-based semantic web services. *IEEE Internet Computing*, 8(3):66–73, 2004. doi: 10.1109/MIC.2004.1297276. (Cited on pages 12 and 13.)

[218] Reina Uba, Marlon Dumas, Luciano García-Bañuelos, and Marcello La Rosa. Clone detection in repositories of business process models. In Stefanie Rinderle-Ma, Farouk Toumani, and Karsten Wolf, editors, *BPM*, volume 6896 of *Lecture Notes in Computer Science*, pages 248–264. Springer, 2011. ISBN 978-3-642-23058-5. (Cited on pages 57, 58 and 66.)

[219] Wil M. P. van der Aalst. Formalization and verification of event-driven process chains. *Information & Software Technology*, 41(10):639–650, 1999. doi: 10.1016/S0950-5849(99)00016-6. (Cited on pages xx, 112, 113, 118, 120 and 121.)

[220] Wil M. P. van der Aalst. Process mining. In Liu and Özsu [135], pages 2171–2173. ISBN 978-0-387-35544-3, 978-0-387-39940-9. (Cited on page 53.)

[221] Wil M. P. van der Aalst, Marlon Dumas, Arthur H. M. ter Hofstede, Nick Russell, H. M. W. (Eric) Verbeek, and Petia Wohed. Life after bpel? In Mario Bravetti, Leïla Kloul, and Gianluigi Zavattaro, editors, *EPEW/WS-FM*, volume 3670 of *Lecture Notes in Computer Science*, pages 35–50. Springer, 2005. ISBN 3-540-28701-9. (Cited on page 55.)

[222] Wil M. P. van der Aalst, Maja Pesic, and Helen Schonenberg. Declarative workflows: Balancing between flexibility and support. *Computer Science - R&D*, 23 (2):99–113, 2009. (Cited on page 61.)

[223] W.M.P. van der Aalst. Putting high-level petri nets to work in industry. *Computers in Industry*, 25(1):45 – 54, 1994. ISSN 0166-3615. doi: http://dx.doi.org/10.1016/0166-3615(94)90031-0. (Cited on page 113.)

[224] W.M.P. van der Aalst. Process mining: discovering and improving spaghetti and lasagna processes. In *Computational Intelligence and Data Mining (CIDM), 2011*

IEEE Symposium on, April 2011. doi: 10.1109/CIDM.2011.6129461. (Cited on pages xix and 4.)

[225] Jussi Vanhatalo, Hagen Völzer, and Jana Koehler. The refined process structure tree. *Data Knowl. Eng.*, 68(9):793–818, 2009. doi: 10.1016/j.datak.2009.02.015. (Cited on pages 128 and 129.)

[226] Kunal Verma and Amit P. Sheth. Semantically annotating a web service. *IEEE Internet Computing*, 11(2):83–85, 2007. doi: 10.1109/MIC.2007.48. (Cited on pages 7, 29, 30, 33, 37, 38 and 98.)

[227] Werner Vogels. Web services are not distributed objects. *IEEE Internet Computing*, 7(6):59–66, 2003. doi: 10.1109/MIC.2003.1250585. (Cited on page 72.)

[228] Gabriela Vulcu, Sami Bhiri, Wassim Derguech, and María José Ibáñez. Semantically-enabled Business Process Models Discovery. *International Journal of Business Process Integration and Management*, 5:257–272, 2011. (Cited on pages 7, 8, 12, 39, 45, 46, 47, 56, 58, 66, 67, 104, 113, 127 and 128.)

[229] W3C Semantic Web Interest Group. Basic geo (wgs84 lat/long) vocabulary. `http://www.w3.org/2003/01/geo/`. Accessed: 25/05/2014. (Cited on page 144.)

[230] Priscilla Walmsley. *XQuery*. O'Reilly Media, Inc., 2007. ISBN 0596006349. (Cited on page 55.)

[231] Yair Wand, Veda C. Storey, and Ron Weber. An ontological analysis of the relationship construct in conceptual modeling. *ACM Trans. Database Syst.*, 24(4): 494–528, 1999. doi: 10.1145/331983.331989. (Cited on pages 17, 90, 91, 92, 93, 94 and 95.)

[232] Web-Ontology Working Group. OWL-S: Semantic markup for web services. `http://www.w3.org/Submission/OWL-S/`, 2004. Accessed: 25/05/2014. (Cited on pages 7, 8, 29, 30, 31, 33, 38, 47 and 98.)

[233] Barbara Weber, Jan Mendling, and Manfred Reichert. Flexibility in process-aware information systems (proflex) workshop report. In *15th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE 2006), 26-28 June 2006, Manchester, United Kingdom*, pages 269–270. IEEE Computer Society, 2006. ISBN 0-7695-2623-3. doi: 10.1109/WETICE.2006.40. (Cited on page 157.)

[234] Barbara Weber, Shazia Wasim Sadiq, and Manfred Reichert. Beyond rigidity - dynamic process lifecycle support. *Computer Science - R&D*, 23(2):47–65, 2009. (Cited on pages xix, 60 and 61.)

[235] Ingo Weber, Jörg Hoffmann, and Jan Mendling. Beyond soundness: on the verification of semantic business process models. *Distributed and Parallel Databases*, 27(3):271–343, 2010. doi: 10.1007/s10619-010-7060-9. (Cited on page 114.)

[236] Ron Weber et al. *Ontological foundations of information systems*. Coopers & Lybrand and the Accounting Association of Australia and New Zealand Melbourne, 1997. (Cited on page 94.)

[237] Matthias Weidlich, Tomer Sagi, Henrik Leopold, Avigdor Gal, and Jan Mendling. Predicting the quality of process model matching. In Daniel et al. [42], pages 203–210. ISBN 978-3-642-40175-6. doi: 10.1007/978-3-642-40176-3_16. URL http://dx.doi.org/10.1007/978-3-642-40176-3_16. (Cited on page 165.)

[238] Mathias Weske. *Business Process Management: Concepts, Languages, Architectures*. Springer, 2007. ISBN 978-3-540-73521-2. (Cited on pages 1, 4, 5, 28, 71 and 154.)

[239] Gerhard Wickler and Austin Tate. Capability representations for brokering: A survey. In *Available from: www.aiai.ed.ac.uk/project/oplan/cdl/cdl-ker.ps*, 1999. (Cited on pages 35, 36, 73 and 76.)

[240] Gerhard Jurgen Wickler. *Using Expressive and Flexible Action Representations to Reason about Capabilties for Intelligent Agent Cooperation*. PhD thesis, University of Edinburgh, Edinburgh, UK, April 1999. (Cited on pages 73 and 76.)

[241] Roel Wieringa. *Design Science Methodology for Information Systems and Software Engineering*. Springer, 2014. ISBN 978-3-662-43838-1. doi: 10.1007/978-3-662-43839-8. URL http://dx.doi.org/10.1007/978-3-662-43839-8. (Cited on pages 96, 124 and 180.)

[242] WSMO working group. WSMO: Web service modelling ontology. http://www.wsmo.org/, 2005. Accessed: 25/05/2014. (Cited on pages 7, 8, 29, 30, 31, 33, 38, 47 and 98.)

[243] Surya B. Yadav, Ralph R. Bravoco, Akemi T. Chatfield, and T. M. Rajkumar. Comparison of analysis techniques for information requirement determination. *Commun. ACM*, 31(9):1090–1097, 1988. doi: 10.1145/48529.48533. (Cited on page 90.)

[244] Wand Yair and Weber Ron. An ontological evaluation of system analysis and design methods. In Falkenberg E D and Lindgren P, editors, *Information Systems Concepts: An in-depth Analysis*. Elsevier Science Publishers, Amsterdam, The Netherlands, 1989. (Cited on pages 17, 90 and 91.)

[245] Maciej Zaremba, Tomas Vitvar, Sami Bhiri, Wassim Derguech, and Feng Gao. Service Offer Descriptions and Expressive Search Requests - Key Enablers of Late Service Binding. In Christian Huemer and Pasquale Lops, editors, *E-Commerce and Web Technologies - 13th International Conference, EC-Web 2012, Vienna, Austria, September 4-5, 2012. Proceedings*, volume 123 of *Lecture Notes in Business Information Processing*, pages 50–62. Springer, 2012. ISBN 978-3-642-32272-3. doi: 10.1007/978-3-642-32273-0. (Cited on page 36.)

[246] ZhangBing Zhou, Feng Gao, and Lei Shu. Service Protocol Replaceability Assessment in Mediated Service Interactions. In *Proceedings of IEEE International Conference on Communications, ICC 2011*, pages 1–5, Kyoto, Japan, 5-9 June 2011. IEEE. ISBN 978-1-61284-232-5. (Cited on page 13.)

[247] ZhangBing Zhou, Walid Gaaloul, Lei Shu, Samir Tata, and Sami Bhiri. Assessing the replaceability of service protocols in mediated service interactions. *Future Generation Computer Systems*, 29(1):287–299, 2013. (Cited on page 13.)