UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Laura Lilleste

Structuring Business Processes and Eliciting Requirements at the Estonian Literary Museum

Bachelor's Thesis (9 ECTS)

Supervisor:
Raimundas Matulevičius

Tartu 2016

Structuring Business Processes and Eliciting Requirements at the Estonian Literary Museum

Abstract:

Business processes are often difficult to structure in large organisations. Business processes are often described by graphical process models. It is generally accepted that complex business processes are easier to understand when they are represented as a collection of smaller and simpler process models. Process models are also an important source of information for requirements elicitation. In this thesis the case of the Estonian Literary Museum (LM) is considered. The LM has problems with structuring their business processes. For modelling the business processes with variation at the LM, a decomposition-driven method is applied. In collaboration with the representatives of the LM, business process models describing the current business processes are created. Process models are created using the BPMN standard. Then software system support for the business processes is described. Using the process models with the software system support description and a method for requirements elicitation from BPMN models, the requirements for the software system are elicited. Applying these methods in the case study at the LM, helps structure the business processes and elicit the software system requirements. Since the business processes models are created in cooperation with the representatives from the LM and based on their current processes, the resulting models and software system requirements describe their actual business processes.

Keywords: Structuring and Modelling Business Processes, Requirements Elicitation, BPMN.
CERCS: T120, Systems engineering, computer technology.

Äriprotsesside struktureerimine ja nõuete välja selgitamine Eesti Kirjandusmuuseumis

Lühikokkuvõte:

Suurtes organisatsioonides võib esineda probleeme äriprotsesside struktureerimisega. Sageli kasutatakse äriprotsesside kirjeldamiseks graafilisi mudeleid. Üldiselt nõustutakse, et keerulisi äriprotsesse on kergem mõista, kui neid kujutatakse kollektsiooni väikemate ja lihtsamate mudelitena. Protsesse kirjeldavad mudelid on samuti olulised informatsiooni allikad tarkvara süsteemi nõuete välja selgitamisel. Selles bakalaureuse töös käsitletakse Eesti Kirjandusmuuseumi praktilist juhtumit, kuna neil esineb probleeme äriprotsesside struktureerimisega. Variatsioonidega äriprotsesside modelleerimiseks Kirjandusmuuseumis kasutatakse dekompositsioonist ajendatud äriprotsesside modelleerimise meetodit. Koostöös Kirjandusmuuseumi esindajatega luuakse mudelid, mis kirjeldavad aktuaalseid äriprotsesse. Protsessi mudelid koostatakse kasutades BPMN normatiive. Järgmisena kirjeldatakse tarkvara süsteemi poolt äriprotsessidele pakutavat süsteemi tuge. Kasutades protsessi mudeleid koos tarkvara süsteemi toe kirjeldusega ning BPMN mudelitest nõuete tuvastamise meetodit selgitatakse välja nõuded tarkvara süsteemile. Nende meetodite rakendamine Kirjandusmuuseumi juhtumis aitab nende äriprotsesse struktureerida ning tarkvara süsteemi nõudeid välja selgitada. Kuna äriprotsesside mudelid luuakse koostöös Kirjandusmuuseumi esindajatega ning kuna nende mudelite aluseks on aktuaalsed äriprotsessid, siis tulemusena saadud mudelid ja nõudmised tarkvara süsteemile kirjeldavad Kirjandusmuuseumi tegelikke äriprotsesse.

Võtmesõnad: Nõuete tuvastamine, äriprotsesside modelleerimine, BPMN.
CERCS: T120, Süsteemitehnoloogia, arvutitehnoloogia.

Table of Contents

1. Introduction

Organisations often have problems with structuring business process models in an understandable way [1]. In order to create comprehensible process models, it is agreed that a complex business process should be captured as a collection of smaller and simpler models [1]. For dividing business processes, Milani [1] proposes a decomposition-driven method for modelling business processes with variation. Software systems are often described with graphical models [2]. Understanding the actual needs of an organisation helps create an effective software system [2]. Requirements elicitation is about discovering the needs of the users and learning the environment, and the primary source for requirements elicitation is the software system that is currently being used [2]. For eliciting requirements from models, Valvas [2] proposes a method for requirements elicitation from BMPN models.

This leads to the following research questions:

RQ1: What is the applicability of the method for modelling business processes with variation [1]? In other words, we will study if this method helps structuring business processes in an organisation.

RQ2: What is the applicability of the method for requirements elicitation from BPMN models [2]? In other words, we will analyse whether this method helps eliciting requirements from the BPMN models.

In order to answer these research questions, the case of the LM is considered. The different departments of the LM all have different business processes. Due to historical reasons there is currently no full documentation of the business processes and software system requirements at the LM and they have problems managing their business processes. In this thesis, the business processes of the LM are elicited using the method for modelling business processes with variation and the method for eliciting requirements from BPMN models.

In this thesis, one of the main business processes at the LM is analysed. Specifically the sub-process of analysing information and uploading analysed material is chosen for further investigation. Out of the four departments of the LM, three will be interviewed to create business process models for each. One department is then chosen for eliciting requirements for the software system. Out of the process of analysing information and uploading analysed material, the sub-process of digitising is selected for creating requirement specifications.

First the business process models are created using the method for modelling business processes with variation proposed by Milani in [1]. In order to create the process models, first the main processes of the LM are elicited. Then the main process of analysing is decomposed and the sub-process of analysing information and uploading analysed material is then decomposed into sub-processes for each department. For each department, the process is decomposed until such level of detail that no sub-processes remain in the process model. Then the model is presented using BPMN and the data objects are described. Once the models are presented, the requirements elicitation from the process of analysing information and uploading analysed material is performed for one of the departments. For the sub-process of digitising, requirements specifications are created using the method for requirements elicitation from BPMN models proposed by Valvas in [2].

The thesis is structured as follows: In Chapter 2 the background for the work is described. Overview of business process management, business process model and notation and also requirements engineering is given. Then the method for modelling business processes with variation and the method for requirement elicitation from BPMN models are introduced. In Chapter 3 the analysis method, the case description and overview of the case study results are explained. In Chapter 4 the case study at the LM is conducted and the business process models created. In Chapter 5 the requirements elicitation for EKLA is described. In Chapter 6 the conclusion of the work is presented.

2. Background

In the following chapter the background of this thesis is described. Introductions to business process management (BPM) and to business process model and notation (BMPN) are made. Then requirements engineering is described. Finally a method for modelling business processes with variation and also a method for requirements elicitation from BPMN models is introduced.

2.1 Business Process Management

Dumas, La Rosa, Mendling and Reijers [3] describe the Business Process Management (BPM) as a science of overseeing how work is carried out in an organisation in order to guarantee consistent outcome and discover possible advancements. They add that BPM does not focus on improving an individuals' way of working but instead on managing entire chains of events, activities and decisions that add value to both the customer and the organisation. In order to define a business process and BPM, some keywords are introduced first [3].

An *event* corresponds to things that happen atomically and have no duration but an event can trigger the execution of series of *activities* that each take time. If an *activity* is quite simple and considered a single unit of work, it is called a *task*. Typical processes also involve *decision points*, that affect the way a process is executed, different *actors* (such as humans, organisations and software systems), *physical objects* (such as equipment, materials) and *immaterial objects* (such as electronic documents, electronic records).

An executed process can lead to one or several *outcomes* that ideally are positive and deliver value to the actors involved in that process. Sometimes that value will not be achieved at all or it will be achieved partially and that corresponds to a *negative outcome*. The actors that consume the output play the role of the *customers* and the customers can be either internal or external to the organisation.

Having introduced these keywords, a definition can be used: a business process is a collection of inter-related events, activities and decision points that involve different actors and objects, and it collectively leads to an outcome that can be of value to customers. Business processes are the focal point of BPM since it is a collection of methods, techniques and tools to discover, analyse, redesign, execute and monitor these business processes. Generally, the purpose of executing a BPM initiative is to make sure that the business processes always lead to positive outcomes and also deliver maximum value. The BPM discipline is described in the BPM lifecycle model in Figure 1.

The initial phase in the BPM lifecycle is termed as *process identification* and the main tasks are identifying the processes that are relevant to the problem on hand, determining the scope of these processes and then identifying the relations between these processes. The first phase produces the process architecture, typically a collection of processes and the different relations between these processes. It is important to define the process performance measures that will be used for deciding what shape the process is in. The most recurrent classes of measure are cost-related, time-related and quality-related especially the error rate which is the percentage of times that an execution of the process ends in a negative outcome.

The second phase called *process discovery* (also called *as-is process modelling*) aims to understand the business processes in detail and produce one or several as-is process models that reflect the understanding that people in this organisation have concerning how the work is done. These documents are most commonly a combination of diagrammatic process models (flowcharts) and text. Diagrammatic process models usually consist of activity nodes that represent units of work, control nodes that capture the flow of execution between activities and possibly also event nodes which tell that some events may or must happen that require a reaction. Flowcharts can also be divided into swimlanes that indicate different organisational units.

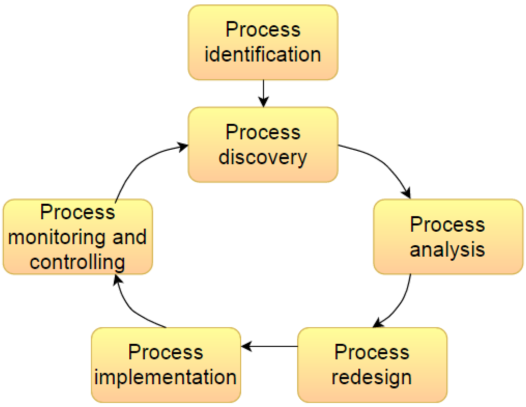


Figure 1. BPM lifecycle, adapted from [3]

Once the as-is process has been described in detail, the third phase called *process analysis* starts. In this phase the issues in the as-is process are identified, analysed and then documented. When the issues have been analysed and possibly quantified the *process redesign* (also called process improvement) phase follows. The goal of this phase is to identify and analyse potential remedies that would help to address the issues identified previously. Multiple possible options are considered for addressing a problem since making one change in the process to solve one issue could possibly cause other issues later. To address the issues identified in the as-is process, a redesigned version (to-be version) of the process is proposed during the phase of *process redesign*.

In order to execute the to-be process the necessary changes in the IT systems and in the ways of working need to be implemented. This phase is called the *process implementation* phase and it usually involves two aspects: organisational change management and process automation. Once the redesigned process has been implemented, it is likely that some adjustments are needed in order to meet the expected results. This final phase is called *process monitoring and controlling* and its goal is to collect data and examine it closely in order to determine how well the process is performing. The changes executed in process implementation phase may cause new issues to arise and hence creating a need to repeat the BPM cycle on a continuous basis.

In this thesis, the first three phases of the BPM lifecycle (process identification, process discovery and process analysis) will be executed. A popular standard nowadays for process modelling is called Business Process Model and Notation (BPMN), it was released by the Object Management Group in 2011. As BPMN is a widely used standard it is also used in this thesis for business process management.

2.2 Business Process Model and Notation

In BPMN documentation [4], the descriptions are as follows: a process is represented as a graph consisting of activities, events, gateways and sequence flow. The symbols which make up the process flowcharts are shown in Figure 2.

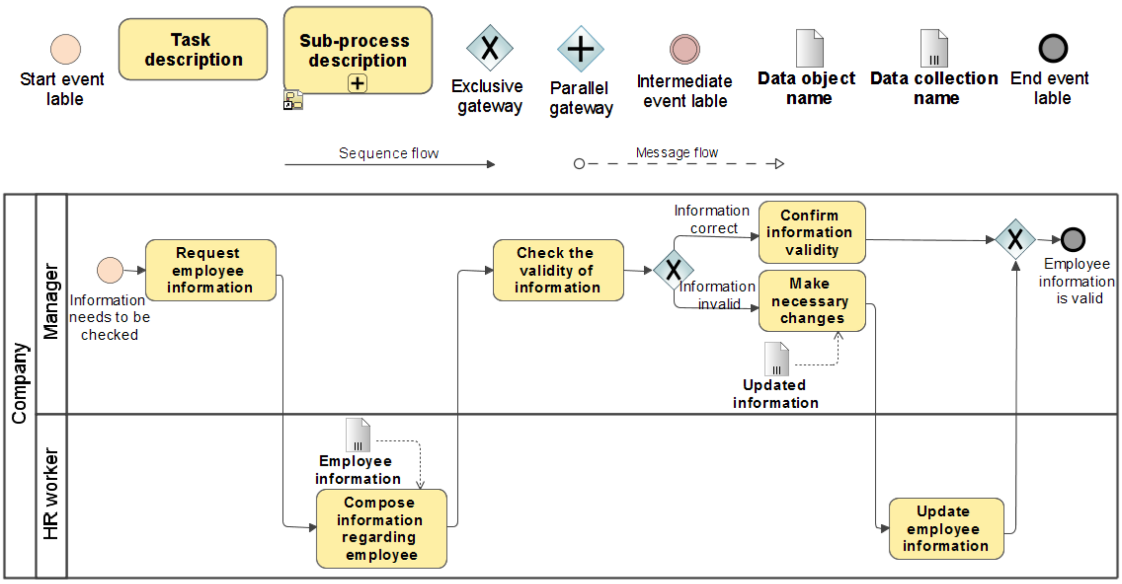


Figure 2. BPMN symbols, adapted from [3] [4]

The event that starts a process is a *start event* that has a descriptive label stating why the process is initiated. A start event is represented with a circle that has a thin border [3], it has no incoming sequence flow and it has a trigger. An activity that is considered a single unit of work is called a *task* and it is represented by a rectangle with rounded corners and has a task description. If an activity is considered more complex than a task it is represented as a *sub-process* rectangle with rounded corners. A sub-process has two views: expanded and collapsed view. Models also have many decision points (also called *gateways*) where the process can split into several alternative paths. The two types of gateways used in following models are exclusive gateway and parallel gateway. An *exclusive gateway* represents a decision point where only one outgoing path can be followed. A *parallel gateway* represents a point in the process where the sequence flow splits into several paths that run parallel. An *intermediate event* indicates that something happens that affects the flow but it does not start or terminate the process. It is represented with a circle with a double line. Data used for or created during a task is described as a *data object*. When a large amount of data is used or created it is represented with a *data collection*. A process ends with an *end event* that has no outgoing sequence flow. End events are represented by a circle that has a thick line [4] and it has a label describing in which state the process ended [3].

When a process involves different participants in a collaboration the flowchart is represented in a *pool*. A pool is divided into separated *swimlanes* to partition sets of activities from other activities (see Figure 2). Lanes extend either vertically or horizontally the full length of the pool. In the example pool labeled *Company* are two separate swimlanes labeled *HR worker* and *Manager*. The connecting object that shows the order of the activities in the model is *sequence flow* and each flow can only have one source and target. Sequence flow is represented with a solid line. Sequence flow can also be controlled and then it is subject to conditions or dependencies. *Message flow* is used to indicate the flow of messages between two participants. A connecting object that connects to data objects is a called an *association* and it is represented as a dotted line with an arrowhead. The full set of available elements is described in the BPMN documentation [4].

2.3 Requirements Engineering

This description of requirements engineering is offered by Kotonya and Sommerwille [5]. In order to reuse knowledge of solving a problem the process needs to be documented.

Processes are a fundamental part of everyday activities and usually the details are explained by describing the process. The level of detail that the processes are defined at varies largely, it depends on the complexity of the process, the presumed actor and the expected environment. Processes can also be design processes that require creativity, people interaction, engineering judgement, background knowledge and experience. There is a wide range of possible outputs of a design process that all satisfy the inputs given. A design process can not be automated nor can it be specified in high level of detail. Requirements engineering (RE) process is considered a design process with a set of inputs and outputs visible in Figure 3.

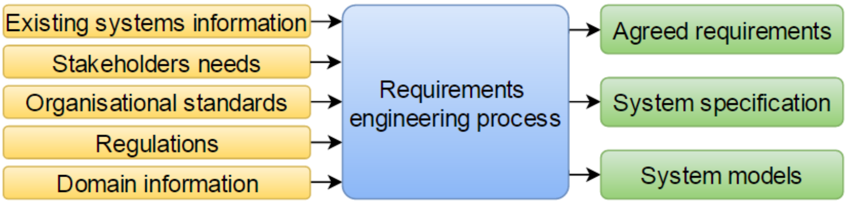


Figure 3. Requirements engineering process inputs and outputs, adapted from [5]

There are five inputs: *existing system information* (information concerning the functionality of software system to be replaced or other systems that interact with it), *stakeholders needs* (description of what they need from the software system), *organisational standards* (standards in the organisation concerning software system development practice, quality management, etc.), *regulations* (external regulations that apply to the system) and *domain information* (information concerning the application domain of the software system). Three outputs of the RE process are: *agreed requirements* (description of the software system requirements that stakeholders agree with and understand), *system specification* (detailed specification of the software system functionality) and *system models* (description of the software system from different perspectives using models such as data-flow models, object models, process models, etc).

In Figure 3, the RE process is presented as a 'black box' but in practice the processes are very variable due to a number of factors that contribute: technical maturity (the technology and method used for RE), disciplinary involvement (the types of engineering and managerial disciplines involved in RE), organisational culture (the culture of an organisation has important influence on all business processes including RE), application domain (different types of RE processes are needed for different types of application systems).

In order to describe a process in detail and create a complete understanding of the process several different types of models giving different process information are needed. The types of the models depend on the expected use of these models. Some of the different types of models: *Coarse-grain activity models* (describe the context of different activities in the process, usually the starting point for process description), *fine-grain activity models* (more detailed models of a specific process), *role-action models* (show the roles and actions of the different people involved in the process) and *entity-relation models* (show the inputs, outputs and intermediate results of the process and the relationships between them).

Since the RE process varies widely it is not advisable to suggest an ideal RE process and impose it on every organisation, but instead organisations should start with a generic RE process (see Figure 4) and then instantiate this into a detailed process suitable to their needs.

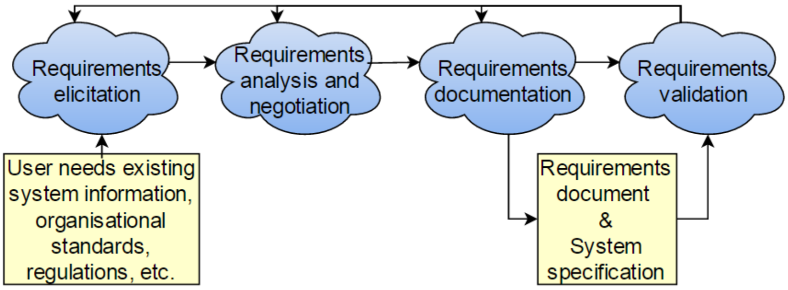


Figure 4. Coarse-grain activity model of RE process, adapted from [5]

In Figure 4 the activities have no distinct boundaries since in practice these activities are interleaved and there is a lot of iteration and feedback between the activities. The activities of the RE process are: *requirements elicitation* (the software system requirements are identified through discussions with the stakeholders, software system documents, domain knowledge and the market studies), *requirements analysis and negotiation* (in detail analysis of the requirements and negotiations with the stakeholders to decide on the set of agreed requirements for the software system), *requirements documentation* (the previously agreed on requirements are documented in a way that is understandable by all system stakeholders) and *requirements validation* (checking the consistency and completeness of the requirements in order to detect problems).

Requirements elicitation and analysis is a complex negotiation process that involves all system stakeholders. The requirements elicitation process consists of four main components: application domain understanding (knowledge of the general area that the system is applied in), problem understanding (details of the specific customer problem), business understanding (knowledge of how the software system contributes to the development of the organisation) and understanding the needs and constraints of system stakeholders (understanding which work processes the software system is intended to support and also the role of existing software systems in those processes).

Most of the knowledge gained during requirements elicitation comes from reading existing documents about the software system and talking to people involved with the system. The result is a large amount of information and there are three fundamental ways of structuring this knowledge: *partitioning* (systematising information into collections by the relationship where the knowledge is described in terms of its parts), *abstraction* (systematising knowledge according to general or specific relationships by relating specific instances to abstract structures) and *projection* (systematising knowledge from different viewpoints since different sources contribute different information about the software system).

2.4 A Method for Modelling Business Processes with Variation

In [1] [6] it is noted that large and complex process models are easier to understand once they are decomposed into smaller sub-processes. Architecture of a process is visible in Figure 5. The highest level of process decomposition is the *process map*, an abstract model demonstrating the processes of an organisation and defining their relationships. The processes on the highest level are either *core processes* (serving external customers) or *support processes* (serving internal customers). Both core and support processes have a *main process* (a process that does not belong to any larger process). A main process is decomposed into *sub-processes* that are processes on their own and can also be decomposed into sub-processes. Decomposition is final once a sub-process only consists of *activities*. Process architecture should consist of 4-5 levels and even more levels can be added if necessary.

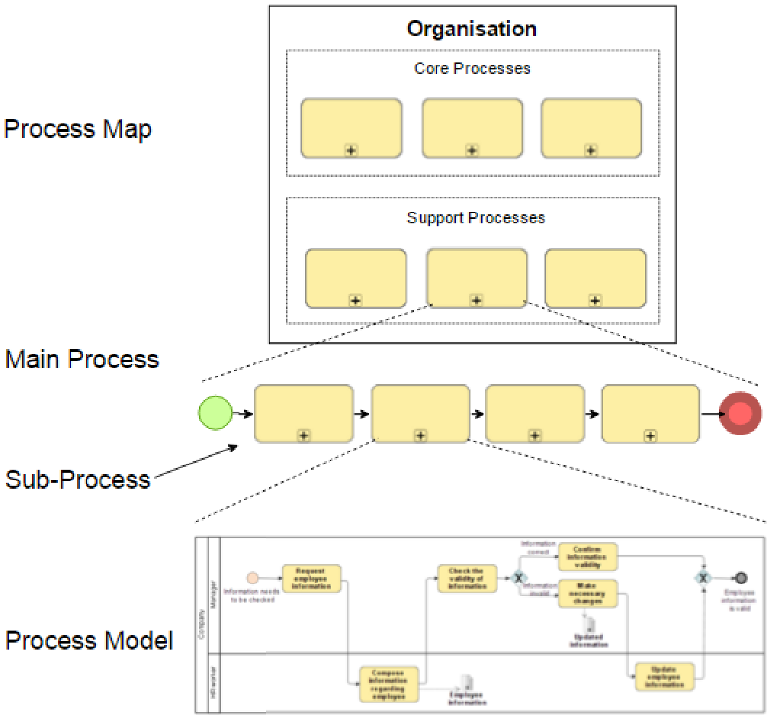


Figure 5. Process architecture illustration, adapted from [1]

For modelling families of process variants, it is proposed [1] [6] to use the decomposition-driven method that considers both the business reasons and the similarity of variants. During discovering process models or consolidating process models, collections of variants need to be managed and the decomposition-driven method can be used for both. The method suggests that deciding on whether to model different variants together or separately, should not be done at the level of the top-level process, instead it should be done between the process decomposition steps. Decisions to model sub-processes together or separately should be taken in an optimal point in the process hierarchy, which depending on the process might happen on a higher or a lower level. For each sub-process, the consolidated modelling approach should always be considered first until it is evident that the fragmented approach is more favored. The steps of decomposition-driven method are visible in Figure 6.

The goal of the first step of the decomposition-driven method is to *model the main process* in cooperation with the domain experts and business stakeholders and also to scope the process on hand. Major milestones are identified (usually around 5 sub-processes) by applying "breakpoint" decomposition heuristics by: identifying what initiates the main process, identifying in what conditions the process ends, determining the major steps needed to get from start to end, organising the sub-processes (so they are in the order that they are executed in).

The goal of the second step is to elicit and classify the different variants induced by *variation drivers*. To elicit the business drivers, the framework visible in Figure 7 is used. This framework describes an organisation as a system operating inside of another system that is bigger. The factors from the larger system affect the organisation and cause variability in its business processes.

To elicit the business drivers two rounds of questions (in relation to the framework) are asked. First round is about: the products/services the main process produces, who the customers are, where are the products distributed, how the products/services are produced, whether there are any circumstances (external) that require having a different process. In the second round of the questions, all the categories of drivers are further investigated and clarified. The output of the second step is a collection of possible variation drivers (the implicit branching points of the process).

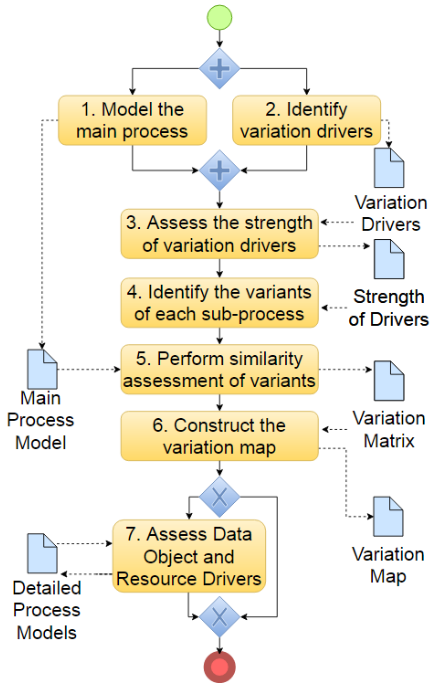


Figure 6. Decomposition-driven method steps, adapted from [1]

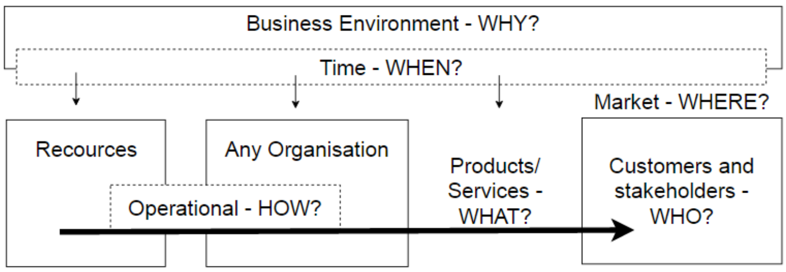


Figure 7. Variation drivers framework, adapted from [1]

Branching points are either defined as decision or as variation points. Variation points are identified first since the outgoing branches of a variation point are considered viable variants. To elicit viable variants, the first step is to identify all branching points in the process model. Then each branching point is assessed to classify it as a decision or variation point. Viable variant is considered to be an outgoing path from a variation point that has similar input and/or leads to similar output as the other outgoing paths from that variation point. To assess each decision point questions are asked about: the similarity of the starting events, the similarity of the outcomes, whether the variants are closely related. If a variation point is identified, the variation options are determined and then the variation driver is identified.

As the viable variants and their business drivers have been identified, the third step is *assessing the relative strength* (importance) *of the variation drivers*. The strength reflects the level of investments needed to merge the variants induced by the driver and also the level of management. Very strong drivers require the variants to be managed separately.

Strong variants can in principle be merged so they could be managed together but it would require significant investments and decisions from higher management layers. Somewhat strong drivers generate variants that differ only in minor details and can be managed together without any decisions from upper management but with decisions from low or mid-layer of management. Not strong drivers generate variants that can be merged or kept separate without it playing a significant role. To determine the strength of a driver, questions are asked about: whether merging the variants is possible, how much investment would merging the variants require, from what level of management would decision be required.

Identifying existing variants for each sub-process and for each business driver is done in step four, *identifying the variants of each sub-process*. For each sub-process, the existing variants per driver are added into the variation matrix (rows correspond to business drivers and columns correspond to sub-processes identified in step 1). One cell in the variation matrix lists the variants of a sub-process induced by the driver (listed in descending order of strength).

The fifth step is *performing similarity assessment of variants of each sub-process* (on a 4-point scale). The assessments used are: identical, very similar (no significant differences), similar (clear similarities), somewhat similar (some isolated parts of the process are similar) and not similar. Result of this step is an annotated variation matrix including the similarity assessment. Having the strength assessment of the drivers and the similarity between the variants, the next step is *constructing the variation map*. In this step, the decision to model the variants in a consolidated or in a fragmented manner is done using the decision matrix described in Figure 8.

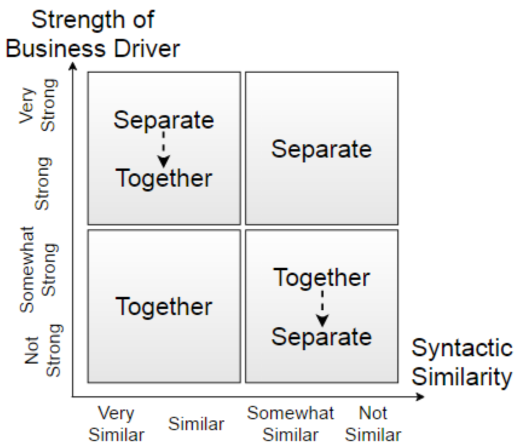


Figure 8. Decision matrix for modelling variants, adapted from [1]

The variants in the top left and lower right quadrant are modelled together or separately depending on the current process decomposition level (levels 1-3 are considered as high levels). On a higher level of process decomposition, if the variants are similar but the driver is very strong, then the driver prevails and the variants are modelled separately. On a lower level of process decomposition, if the business driver is weak but the variants are not very similar, then the syntactic driver prevails and the variants are modelled separately. The output of this step is a variation map that consists only of tasks and splits representing the separation between variants. In the variation map of the first level of process decomposition each sub-process variant is in turn decomposed into a lower-level process model and steps 2-4 are repeated on that level. After having completed the step, models need to be re-modelled (process model consolidation) or modelled from scratch (process discovery).

The last step is titled *data objects and resource driven variation*. The variability induced by data objects and resources is visible only in models on the lowest level of process decomposition (on that level the input and output objects and the performers are modelled). On that level it is decided whether to model activities in a consolidated or fragmented manner based on the strength of the data objects or resources and the similarity of the underlying procedures.

2.5 Requirements Elicitation from BPMN Models

The requirements elicitation method described by Valvas [2] will be used as a guide in the elicitation process. The results of requirements elicitation will be a set of requirement specifications. The template of requirements specifications is visible in Figure 9.

Component	Description
ID:	
Business Process (optional):	
Activity:	
Goal:	
Primary Actor:	
Trigger:	
Steps of activity (positive scenario):	Operational steps: Step 1: ... Step 2: ... ...
	Alternative paths: In case 1: ... In case 2: ... ...
Failure conditions and handling	

Figure 9. Requirement specification template, adapted from [2]

The method for requirements elicitation described by Valvas [2] proposes steps to be followed when discussing every activity during the meetings with the domain experts. This method also proposes a set of questions to be asked during every step of the requirements elicitation process. The process done with every activity can be divided into 7 main steps as visible in Figure 10.

The goal of *identifying relevancy* is to verifying if the activity requires some sort of software system support and as such has a need for specifying the functional requirements, and also verifying whether any external software systems are involved during that activity. If the activity is relevant, a unique ID is given to it. Thus the *ID* field in the requirements specification is filled in and also the name of the *activity* and the *business process* it belongs to.

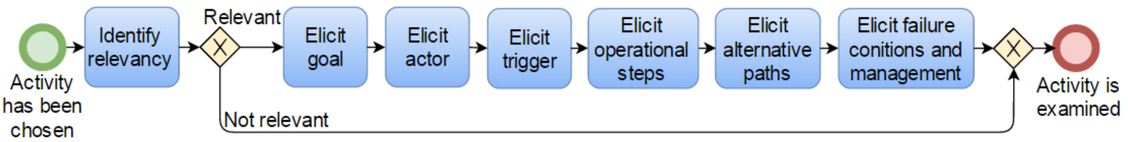


Figure 10. Steps of requirements elicitation method, adapted from [2]

During *goal eliciting*, the purpose is to describe what that activity is aiming to achieve in order to meet the interests of the stakeholders. Also the form and format of the goal is described. The *goal* section of the requirements specification is filled in.

During *actor eliciting*, the actor performing the activity is described. The actor can be human or a resource (non-human, for example machine, information system) but it might not be the one doing all the operational steps in this activity (a human might use a computer program). The *actor* field in the specification is filled in based on the information.

The goal of *eliciting the trigger* is to describe how the actor knows when to start the activity. There are three possible options: receiving a message (verbal message, email, horn sound, etc.); activity starts at a certain time (every 10 seconds, at 5 o'clock, etc.); the preceding activity has finished (only possible if the both activities are performed by the same actor and thus the actor knows when to start the next activity). The *trigger* field in the requirement specification is filled.

In order to reach the goal of the activity one or many operational steps must be completed. There might be different ways of reaching the goal but in this step: *eliciting operational steps*, the standard set of operational steps are described. There are three possible types of operational steps: actor interaction (interaction with some other actor e.g a person, the computer system, an external system), action verification (verifying if some conditions are met), internal action (software system changes some data e.g updates some fields in database, enters to log). The *operational steps* field in the requirement specification is filled in.

During the elicitation of operational steps, a standard set of operational steps was described but in addition to that set alternative paths could be taken to achieve the goal. These sets of other operational steps are elicited during *eliciting alternative paths*. The subsection *alternative paths* is filled with the information in the requirement specification.

In situations called failures in the method, it is not possible to execute all the steps needed to achieve the goal and the activity is interrupted. Such failures usually need some additional actions to be taken. During *eliciting failure conditions and failure management* step, these conditions that cause an activity to be interrupted or not executed and also the activities this failure causes, are elicited. All gathered information is specified in the *failure conditions and management* field in the requirement specification.

2.6 Summary

In this chapter the background for this thesis was presented. First business process management and BPMN were introduced. Then the requirements engineering process was explained. Finally the methods for modelling business processes with variation and requirements elicitation from BPMN models were described.

3. Analysis Method and Case Description

In the following chapter, a detailed description of the analysis method is given. Then follows the case description and the overview of the case study results.

3.1 Analysis Method

Analysis workflow can be divided into two parts: business process elicitation and requirements elicitation. The analysis workflow of this thesis can be seen in Figure 11. The business process modelling is conducted following the method proposed by Milani [1], described in Section 2.4. The first step of process elicitation is to create LM process map and elicit main processes (Section 4.1). Then the scope of the study is defined: analysing information and uploading analysed material (Section 4.2). Once the scope is defined, the variation drivers are elicited for the process of analysing information and uploading analysed material (Section 4.3). Then the relative strength of the drivers is assessed (Section 4.4) and the variants of each sub-process are identified (Section 4.5). Next the similarity of each variant is assessed (Section 4.6) and based on the results a variation map is constructed (Section 4.7). Then for each department, the business process modelling method (Section 2.4) is applied and the process models are elicited (Sections 4.8-4.11).

The second part of the analysis workflow is requirements elicitation. First, the scope is defined for specifying software system support: EKLA process of analysing information and uploading analysed material (Section 5.1). Then the software system support for EKLA is defined and the requirements elicited (Sections 5.2-5.4). Next the scope for requirements elicitation from BPMN models is defined as EKLA sub-process of digitising and the requirement specifications are created (Section 5.5). Finally the conclusion is presented (Chapter 6). Throughout the analysis process, the continuous elicitation and validation of the elicited business processes is done incorporation with the LM.

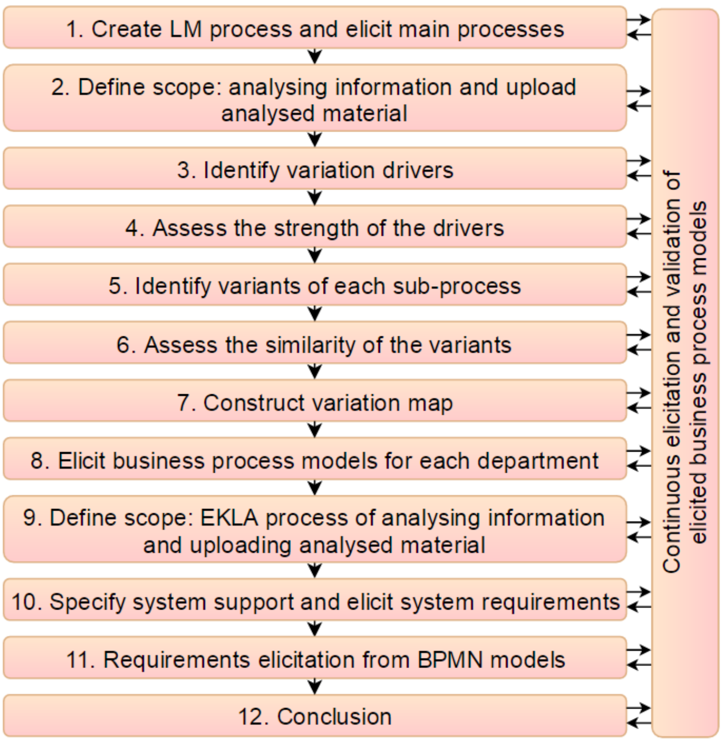


Figure 11. Analysis workflow

3.2 Case Description

The case of this thesis is described [7] [8] as follows: The Estonian Literary Museum (LM) is a state institution that operates in the area of government of the Ministry of Research and Education. It was created on the basis of the Archival Library (founded in 1909), the Estonian Bibliography Foundation (founded in 1921), the Estonian Folklore Archives (founded in 1927) and the Estonian Cultural History Archives (founded in 1929). The main activities include research in the fields of folklore, religions, literature, art and culture, cultural history, life writing, ethnomusicology and bibliography. LM participates in the respective research and development activities of the field like strategic and systematic collection, long-term preservation and scholarly study of cultural heritage. The LM also makes available both the results of the scholarly research and the source materials in publications as well as in digital environments and also as a public service.

LM is divided into four main departments: Archive Library (AR), Department of Folkloristics (FO), Estonian Folklore Archive (ERA) and Estonian Cultural History Archive (EKLA). The main purpose of *EKLA* is to collect, organise, preserve and make accessible culturally important manuscript source materials, photographs, artwork and audiovisual materials. The archive consists of 402 manuscript funds that mainly are individual collections but also some institution or organisation collections. *ERA* is Estonian central folklore archive and its main purpose is to capture as diversely as possible, maintain and also make accessible non-institutional intellectual culture phenomena. *AR* collects, preserves and makes accessible all types of publications: books, periodicals, geographical maps, sheet music, pamphlets. The main goal of *FO* is to introduce Estonian folklore and folkloristics both in Estonia and abroad. Main activities include research in various types and genre of folk poetry like metaphors, folktales and folk religion also analysing the expression and context of folklore in new media, the nature and impact of humour in culture and popular knowledge in astronomy, botany and medicine.

In the LM, there are several software systems used. The database management system that manages data in database (DB) called Kivike will also be referred to as Kivike during the rest of the thesis. The database management system that Manages data in a DB called Ellen will also be referred to as Ellen.

3.3 Overview of Case Study Results

The results of the case study are illustrated in Figure 12. The core processes of the LM are identified as: *Registering, Analysing* and *Scientific Research*. In the *analysing* process, two sub-processes were elicited: *get information uploaded during registering* and *analyse information and upload analysed material*. For the process of *analyse information and upload analysed material* business process models are created for EKLA, ERA and AR departments using the method for modelling business processes with variation (see Section 2.4). For EKLA department software support is defined for sub-processes: *Digitising, Insert into DB* and *Adding presentations*. For the process of *digitising* requirement specifications are created using the method for requirements elicitation from BPMN models (Section 2.5).

The main results of this analysis include:

- business process models with variation of AR, ERA and EKLA, presented in Sections 4.8-4.10;
- software system support for EKLA business processes presented in Sections 5.2-5.4;
- requirements specification for digitising process, presented in Section 5.5.

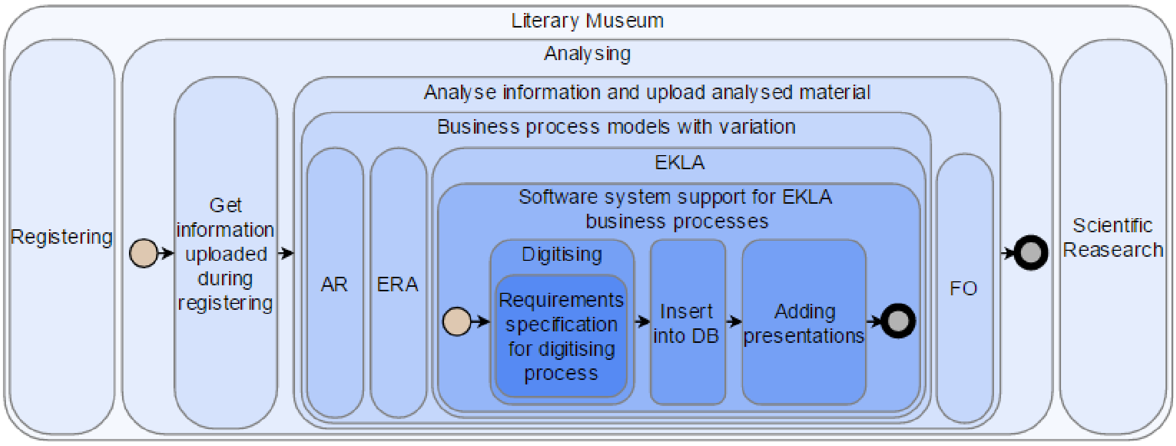


Figure 12. Results of the case study

3.4 Summary

In this chapter the analysis method used for the case study at the LM was described in detail. Then followed the case description introducing the LM and the four departments. Finally the overview of case study results was given.

4. Case Study at the Literary Museum

In the following chapter, the results of the case study at the LM are described. First the process map and the main process are elicited. Then the scope is defined and the variation drivers for the process are elicited. Next the relative strength of the variation drivers is assessed, the variants of each sub-process are elicited and then the similarity of the variants is assessed. Then the variation map is constructed. Finally the business process models are elicited for the departments of the LM.

4.1 Process Map and Main Process

The core processes of the LM are represented in a Process Map visible in Figure 13.

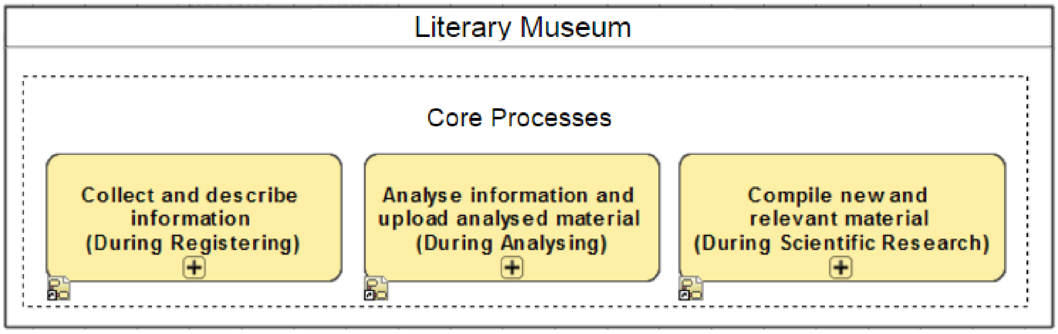


Figure 13. Literary Museum process map

The first core process is *collecting and describing information* and these actions are done during registering. The second core process is *analysing information and uploading analysed material* and the actions are performed during analysing. The third core process done during scientific research is *compiling new and relevant material*.

A more detailed description of registering, analysing and scientific research processes can be seen on Figure 14.

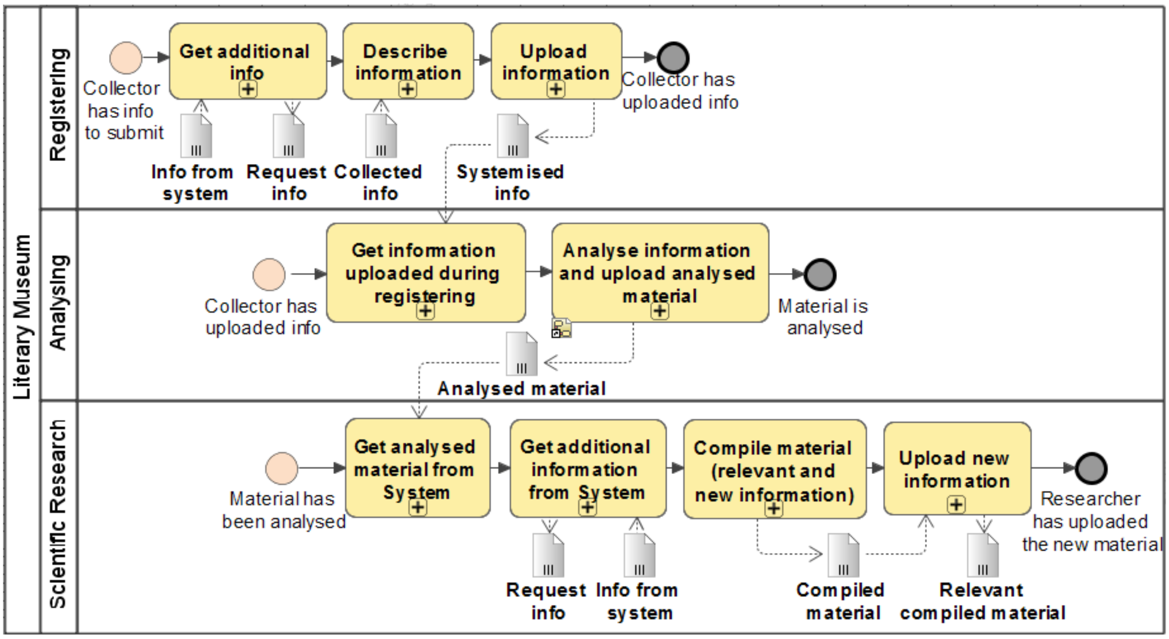


Figure 14. Registering, analysing and scientific research descriptions

Registering process starts when the collector has materials that need registering. The collector can request for additional information from the software system and use that information for describing the material. During the description of material, the collector inserts collected info and uploads the systemised information. The registering process ends once the systemised info has been uploaded.

Analysing process starts after the collector has uploaded the systemised info. The first step is to get the information that the collector uploaded during registering. That systemised info will be used in the following process where this information will be analysed and the analysed material will be uploaded. This process ends once the analysed material is uploaded.

Scientific research process starts once material has been analysed and uploaded. The first step is to get the analysed materials from the software system. Then additional information is requested from the software system. Once info is received from the software system, compiling new material begins. Then the relevant compiled material is uploaded. The process ends once the researcher has uploaded the new material.

4.2 Defining Scope

Out of the three main process flows *registering*, *analysing* and *scientific research*, the focus of this thesis will be on the *analysing* process. During analysing, two activities (both sub-processes) are done: getting information uploaded during registering and also analysing information and uploading analysed material. Out of these two activities, the focus in this thesis will be on the latter. The main process under study is visible in Figure 15.

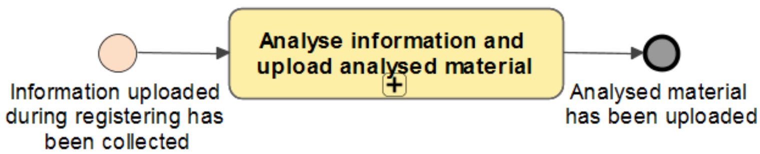


Figure 15. Main process: Analyse information and upload analysed material

4.3 Identification of Variation Drivers

To elicit the business drivers, two rounds of questions were asked from the representatives of each department of the LM during personal interviews. The aim of the first round is to identify the existence of drivers. During the interviews it quickly became apparent that the departments operate differently due to historical differences in the processes. The two main points elicited during this phase were: (1) The products produced by different departments are different due to the difference of their respective fields; (2) since the outputs are different, the process of producing varies form one department to another.

The second round of questions aims to identify and classify the viable variants produced by the drivers. The answers show that: (1) the variants all have identical starting points; (2) the variants produce very similar outcomes; (3) the variants are closely related to each other. These results imply that the variants of the variation point are viable.

4.4 Assessing the Relative Strength of Variation Drivers

The assessment bases on two aspects: (1) the level of investments needed to merge the variants produced by the driver and (2) the level of management needed to make the decision of such changes. The assessment of the strength of the variation driver was conducted during interviews with the representatives of the LM. Although the process of different departments varies due to historical reasons, the merge of the variants is possible. As the result, the variation driver that produces the variants for each departments is induced as "strong" since the merge would require significant investments and also the standardisation would require decision from upper-level management.

4.5 Identifying the Variants of Each Sub-process

For the variation driver identified before, the existing variants of the sub-process are identified in collaboration with the representatives from the LM. The composed variation matrix is visible in Figure 16.

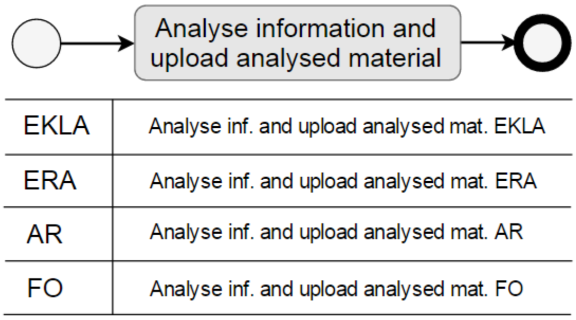


Figure 16. Analyse information and upload analysed material: Variation matrix

4.6 Similarity Assessment of Each Variant

The similarity for each variant subset is identified and graded on a 4-point scale: (1) very similar, (2) similar, (3) somewhat similar and (4) not similar (identical variants are marked accordingly as identical). The annotated variation matrix was developed in cooperation with the representatives from the LM and the matrix is visible in Figure 17.

	EKLA	ERA	AR	FO
EKLA	Identical	3	3	4
ERA	3	Identical	3	4
AR	3	3	Identical	4
FO	4	4	4	Identical

Figure 17. Analyse information and upload analysed material: Annotated variation matrix

4.7 Constricting the Variation Map

The strength of the business driver and the similarity assessments of variants are used to decide whether to model these variants in a conciliated or fragmented manner. To make that decision, the decision matrix in Figure 8 is used.

Since the strength of the driver was graded as "Strong" and the similarities of the variants were graded as either "(3) Somewhat similar" or "(4) Not similar", all of the variants will be modelled separately. The resulting variation map is visible in Figure 18.

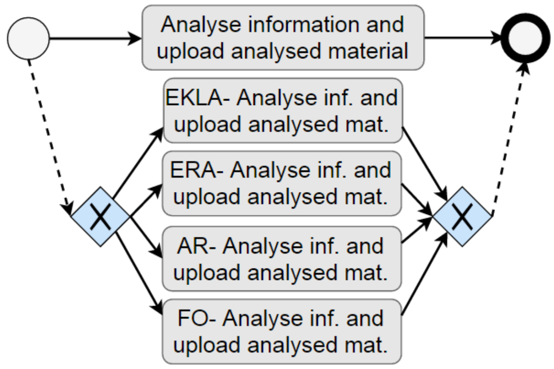


Figure 18. Analyse information and upload analysed material: Variation map

The sub-process of the analysing information and uploading analysed material process (see Figure 14) and the different variants for every department are visible in Figure 19. Each of the variants will be considered and decomposed in turn in the following sections.

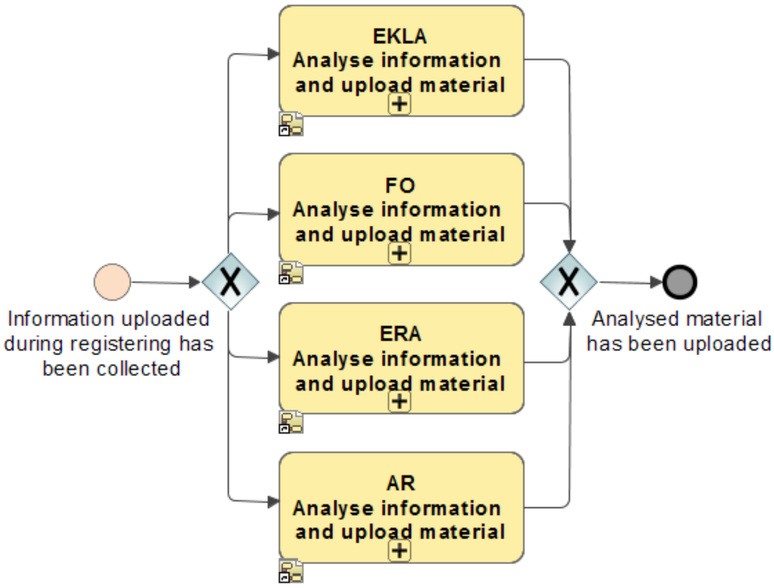


Figure 19. Analyse information and upload analysed material: Process model

4.8 EKLA: Analyse Information and Upload Analysed Material

The process of analysing information and uploading analysed material is decomposed into sub-processes in Figure 20.

Figure 20. EKLA analysing process decomposition

The variation driver was identified as: different products are produced differently. The different products are: manuscripts, photos, artwork and sound/video record. During the driver's assessment of strength, it became apparent that in order to manage these variants together, significant investments would be required, but it would not be impossible. The merging of these variants would also require decisions from upper layer of management.

Hence the strength was assessed as “Strong”. The identified variants of the sub-processes are visible in Figure 21.

Manuscripts	Sort man.	Manual	Digitising man.	Inserting man.	Adding man.
Photos	Sort ph.	Manual	Digitising ph.	Inserting ph.	Adding ph.
Artwork	Sort ar.	Manual	Skipped	Inserting ar.	No presentations
Sound/Video	Sort so.	Manual	Digitising so.	Inserting so.	Adding so.

Figure 21. EKLA variation matrix

During the similarity assessment of the variants of each sub-process, it became apparent that some activities (sorting, inserting into DB) are different for each variant. In several activities (digitising, adding presentations) the manuscript and photo variants are very similar, but other variants are different. Packing activity was the same for every variant. Using the strength of the driver and the assessments of similarity of the variants, variation map visible in Figure 22 was constructed.

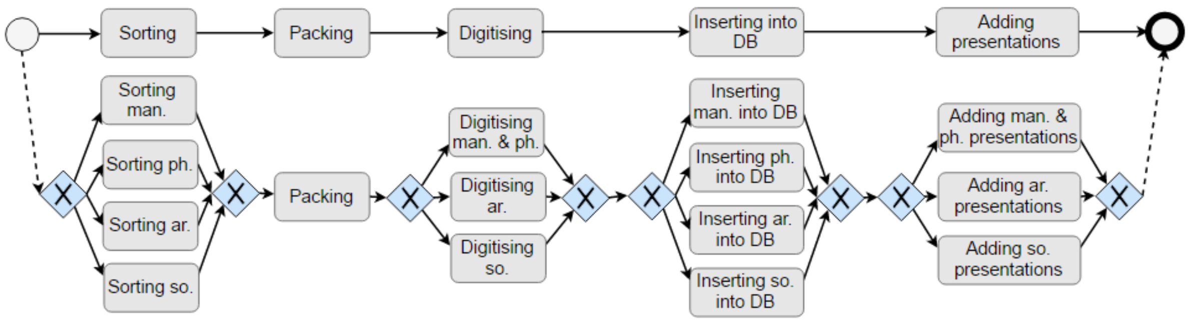


Figure 22. EKLA variation map

Based on the variation map, the final process model for EKLA is constructed and it is visible in Figure 23. The process starts once there is information that needs to be analysed. The first task during *sorting* is to divide the complete archive into three: manuscripts and photos, artwork, sound/video. In case of manuscripts and photos, the next task is to create a physical fund reference. Then the whole collection of manuscripts is sorted into series based on the type. Different manuscript types are: letters, manuscripts, documents, associated materials, about him/her, other, published materials. Then the manuscript series are sorted into subseries. In case of photos the collection is sorted according to the size (format) and then sorted into series based on type. The types of photos are: portraits, group photos, place of residence, family, pictures of manuscripts, funeral/grave, works, shows, artworks, other, negatives and glass negatives. Once the manuscripts and photos have been sorted into series, the materials are ordered alphabetically and chronologically. The collection of artwork is sorted into series based on type: graphics, paintings, sculptures, death masks. The sound/video collection is sorted into series based on the carrier: minidisc, cassette, tape, videotape.

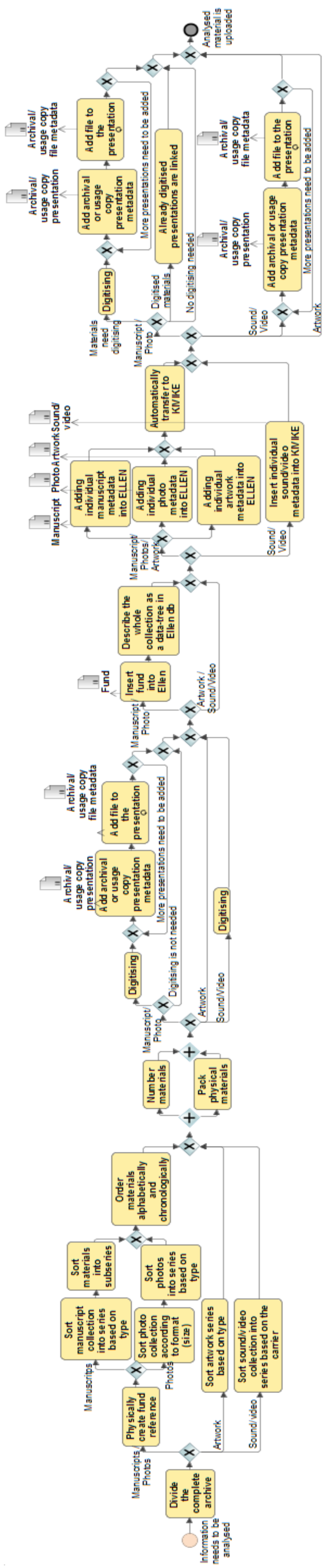


Figure 23. EKLA process model

After the sorting, the next step is *packing* the materials. During packing, the materials are numbered and physical materials are packed. During *digitising* and in case of manuscripts and photos the artifacts are either digitised or not. If the artifacts are decided to be digitised then after the digitising task, the presentations are added. Both archival and usage copy presentations are added and then files are added to both of the presentations. In case of artwork, there is no digitising performed. In case of sound/video digitising is always performed.

After digitising and during *inserting into DB* the first step is inserting the fund into DB. In case of manuscripts and photos, the fund is inserted into Ellen DB and then the whole collection is described as a data-tree. For artwork and sound/video no funds will be inserted. Then in case of manuscripts, photos and artwork the metadata of each individual artifact is inserted into Ellen and then the metadata is automatically transferred to Kivike. In case of sound/video the individual artifact metadata is inserted directly into Kivike.

The last step is *adding presentations*. In case of manuscripts and photos, there are three possible ways: (1) if the materials need digitising then after digitising both the archival and usage copy presentations are added and then files are added for both presentations, (2) if materials were already digitised earlier then the presentations created previously are linked to the artifacts, (3) materials do not need to be digitised. In case on sound/video, the previously digitised materials are added as archival and usage copy presentations and then files are added for both of the presentations. In case of artwork, no digitising and adding presentations is done. The process ends once analysed material has been uploaded.

The data collections in EKLA model (Figure 23) describe the data inserted into the software system. *Archival/usage copy presentation metadata* includes values: PID, type, subtype, archive, registering date, access, state, title, copyright owner. Data collection *archival/usage copy file metadata* is generated automatically by the software system. *Fund* includes values: fund type, fund number, fund title, list number, creators, boundary timelines, languages, size, fund overview, fund formation, storage location, keywords. For adding individual *manuscript* into database, needed values are: portfolio number, archival unit number, archival unit title, content keyword, technical keyword, condition keyword, language, date, size, notes, closed until date, archival unit PID and file PID. For adding individual *photo*, the values needed are: reference number, photo type, content title, content keywords, location, date, dimensions, registry number, notes, archival unit PID, file PID. When adding an individual *artwork* record into database, the necessary values are: reference number, title, content title, content keywords, date, dimensions, technique, material, registry number, technical keywords, condition, notes, closed until date, terms of use, archival unit PID, file PID. Values needed for adding a *sound/video* record into database: PID, type, subtype, archive, registering date, catalogue, access, state, institute, reference, title, copyright owner.

4.9 ERA: Analyse Information and Upload Analysed Material

The decomposition of analysing information and uploading analysed material process into sub-processes in ERA is visible in Figure 24. The identification of drivers showed that the way products are produced varies depending on the product. The different products are manuscripts, photos and sound/video records. Since the variants generated by the driver differ only at the level of minor details and whether these variants are managed together or not is irrelevant for the upper layer of management. The assessment of strength for that driver is “somewhat strong”.

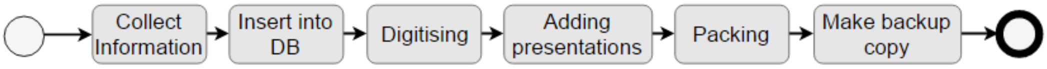
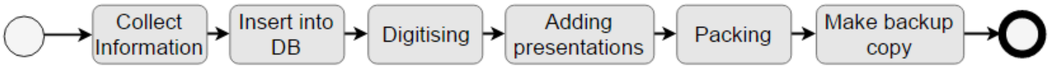


Figure 24. ERA analysing process decomposition

The identified variants for each sub-process are visible in Figure 25. It is apparent from the variation matrix, that most variants are very similar. The only variants that are not similar are the variants of the "Insert into DB" sub-process.



Manuscripts	Collecting inf.	Inserting man.	Digitising	Adding presentations	Manual	Make backup
Photos	Collecting inf.	Inserting ph.	Digitising	Adding presentations	Manual	Make backup
Sound/Video	Collecting inf.	Inserting so.	Digitising	Adding presentations	Manual	Make backup

Figure 25. ERA variation matrix

Using the strength assessment of the driver, the similarity assessments of the variants and the decision matrix in Figure 8 a variation map is constructed in Figure 26. The variants of "Insert into DB" are modelled separately because the syntactic driver prevails.

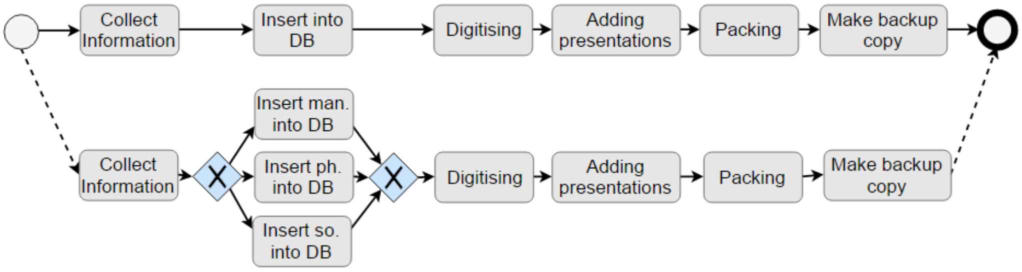


Figure 26. ERA variation map

Using the variation map, the final process model for analysing information and uploading analysed material process for ERA is constructed in Figure 27. The process starts when there is information that needs analysing. The first step is collecting information and during that step three activities are performed in parallel: requesting information from the internal software system, analysing and composing the necessary information and getting information from other people by phone calls, emails and conversations.

Next step is inserting into DB. Manuscripts, photos and sound/video records are all added separately. For each individual artifact the metadata is added into Kivike DB. Then for each artifact, the makers are linked. During digitising, the materials either do not need or in case of physical materials need to be digitised. After digitising, archival and usage copies are added to artifacts and then files are added for both presentations.

Next step is packaging and during that step either packaging is not needed or is needed (in case of physical materials). If needed, the materials are packaged, pages are numbered, reference and ACT numbers are added to the header. The last step is making backup copies during which copies of archive copies are imported into the internal server OHTO. Process ends once the analysed material has been uploaded.

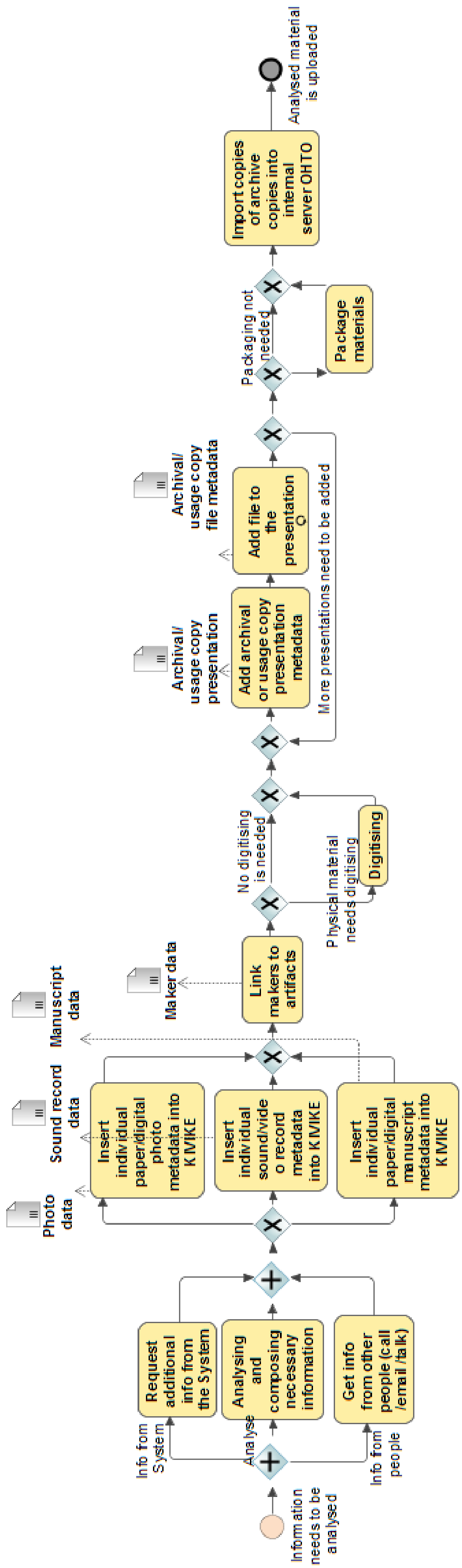


Figure 27. ERA process model

The data collections described in ERA model (see Figure 27) represent the values inserted into the software system. *Photo data* includes values: PID, additional info, access, state, archive, catalogue, project, institute, reference, entry date, year, access deadline, usage limitations, copyright owner, image brand, image size, image quality, paper brand, paper size, paper quality, archive original, colouration, digitaliser, camera, persons in the photo, parish, detailed location, content description, photographer, collector, makers. *Sound record data* includes values: PID, additional info, access, state, archive, catalogue, project, institute, reference, entry time, start time, limitation deadline, usage limitations, copyright owner, data carrier brand, record carrier brand, digitaliser, recorder used, parish, retailed location, collector, content description, performer, makers. *Manuscript data* collection represents values: PID, additional info, access, state, archive, catalogue, project, institute, reference, name, entry time, size, time period, date, limitation deadline, usage limitations, copyright owner, keywords, genre, notes, collector, performer, makers. *Maker data* includes values: PID, additional info, access, state, archive, catalogue, project, name, date of birth, date of passing, occupation, nationality, biography, notes, email, www, postal address, phone number, fax, correspondent, residence, part of a collective, place of birth. Values in *archival/usage copy presentation* are: PID, type, subtype, archive, registering date, access, state, title, copyright owner.

4.10 AR: Analyse Information and Upload Analysed Material

The decomposition of analysing information and uploading analysed material process (see Figure 19) for AR department is visible in Figure 28.

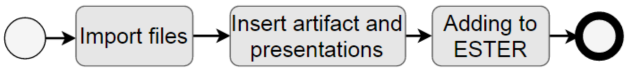
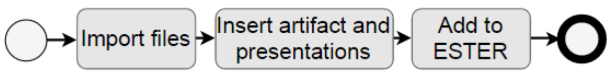


Figure 28. AR analysing process decomposition

During elicitation of the variation drivers it became apparent that there are two different operational ways to complete the process. The options differ in the order of activities: inserting artifact metadata and then adding presentations; inserting presentations first and then adding artifact metadata. From a business perspective the two variants created by the driver differ only at a level of minor details. From the perspective of upper management it is irrelevant whether these variants are managed together or separately. Thus the strength of the driver is rated as “Somewhat strong”.

The variants created by the operational driver are visible in Figure 29. It is visible from the variation matrix that other steps are very similar except for *Insert artifact and presentations*. During the similarity assessment of the variants the variants of *Import files* and *Add to ESTER* were assessed as “Very similar”. The variants of *Insert artifact and presentations* were graded as “Somewhat similar”.



Artifact first	Import files	Insert artifact first	Add to ESTER
Present. first	Import files	Insert presentations first	Add to ESTER

Figure 29. AR variation matrix

Using the strength of the driver, the similarity assessments of the variants and the decision matrix (see Figure 8) a variation map is created visible in Figure 30. The variants of *Insert* *artifact and presentations* are managed separately because the strength of the driver is “Somewhat strong” and the similarity of the variants is “Somewhat similar” and since this is on a lower level of process decomposition, the syntactic driver prevails.

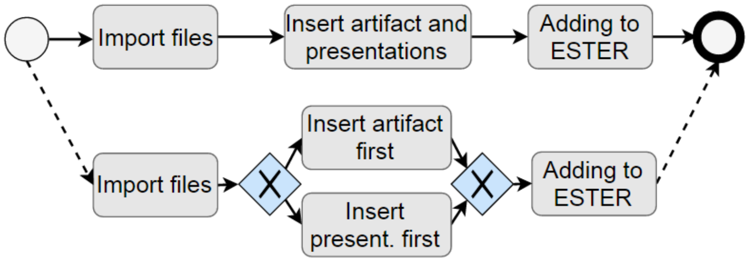


Figure 30. AR variation map

Using the variation map, a detailed process description is created for AR. The description of analysing information and uploading analysed materials is visible in Figure 31. The process of analysing information and uploading analysed material starts in AR when there is information that needs to be analysed. During importing files, the first activity is creating files using ABBY text recognition server. Three files are created: archival copy, OCP doc and OCR pdf file. Then the files are imported into Kivike DB.

Next phase is inserting artifact and presentations. There are two possible paths: inserting artifact first, inserting presentations first. In case the artifact is inserted first, the first activity is creating a new artifact in the DB. Then metadata of the artifact is imported from Ester using the bibliographic record number. Then all the imported fields are verified and corrected if needed. The next activity is adding archival/usage copy presentation metadata and then adding files to the presentation. Usually both archival and usage copy presentations are added to an artifact. In case the presentations are added first, the first step is selecting publication by the call number or the bibliographic record number from the AR inlet area. Then metadata is inserted for file import. If the usage copy was imported incorrectly, it needs to be added again by first adding usage copy presentation metadata and then adding files to the presentation. Once the usage copy is correctly imported the next activity is adding initial metadata to the artifact. Then artifact metadata is imported from Ester using the bibliographic record number. Once the metadata is imported, the fields are verified and corrected if needed.

After the artifacts and presentations have been imported, the artifacts are added to Ester. That is done by adding the PID code, the reference in Kivike and the bibliographic record code to the ISBN file.

Data collections in AR model (see Figure 31) represent values inserted into the software system. For *OCR doc file*, *ORC pdf file* and *archival copy* all values are generated automatically by the software system. Values included in *import data* are: import type, file or folder, document subtype, archive, presentation (archival copy and file type), presentation (usage copy and file type). *Usage copy metadata* includes values: PID, subtype, additional info, access, state, archive, project, call number, name, owner, year created, year collected, digitising machine, digitising software. Data collection *initial artifact metadata* includes values: catalogue, bibliographic record number.

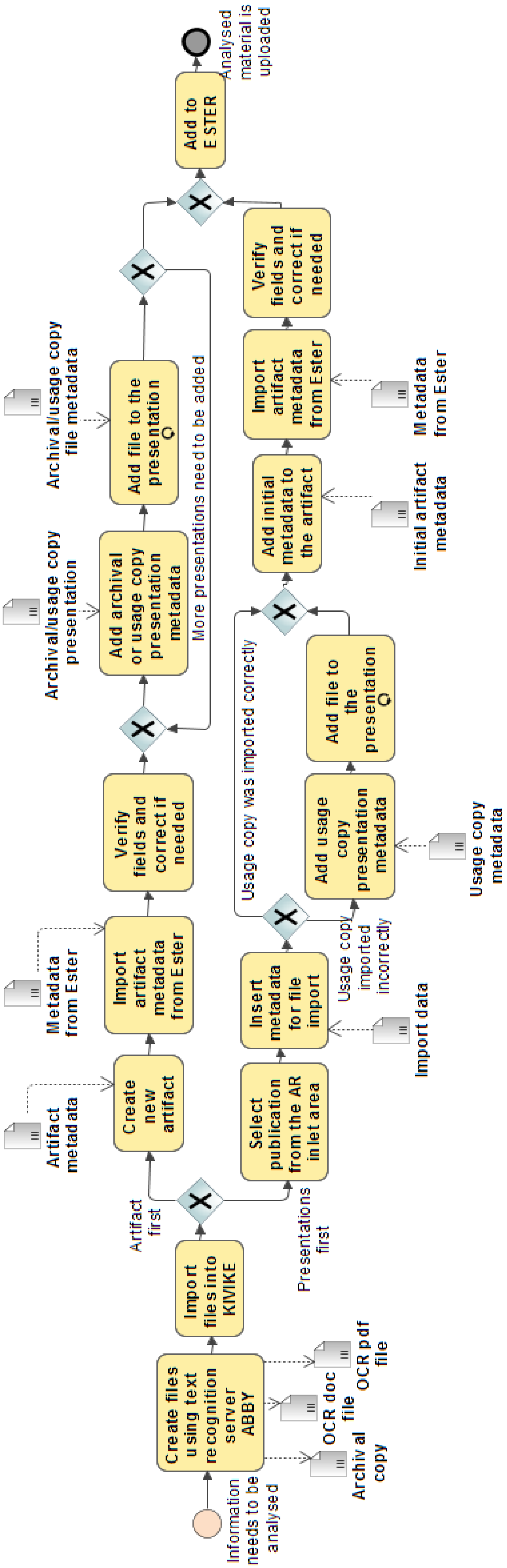


Figure 31. AR process model

Values included in data collection *metadata from Ester* are: institution, call number, original title, publication data, location, year, description, ESTER bibliographic record code, UDK code, owner, keywords, notes, illustrator, archival copy PID, usage copy PID. Values in *artifact metadata* collection are: subtype, archive, catalogue, call number. *Archival/usage copy presentation metadata* includes values: call number, subtype, access, archive, name, owner, year of creation, year of collecting. Values included in *archival/usage copy file metadata* are: access, archive, name, owner, year of creation, year of collecting.

4.11 FO: Analyse Information and Upload Analysed Material

During the interviews with the FO representatives it became apparent that the process of analysing information and uploading analysed material in FO department differs significantly from other departments' process. Thus the method of modelling (see Section 2.4) will not be applied and the FO process will not be in scope of this thesis. The initial description of the process is presented.

FO works with several different databases that differ based on the type of data inserted. The information analysis and material upload process (model visible in Figure 32) starts when information needs to be analysed. Depending on the type of the material, there are two different paths to follow. If the material is graffiti related, the first task is to upload the picture to the folklore server. Once the picture is uploaded, the next task is to add metadata. If the material was mythical folklore and spells, the first task is to add metadata describing the artifact. The following step is to enter the manuscript text. After finishing these tasks, the information analysis is finished and analysed material is uploaded.

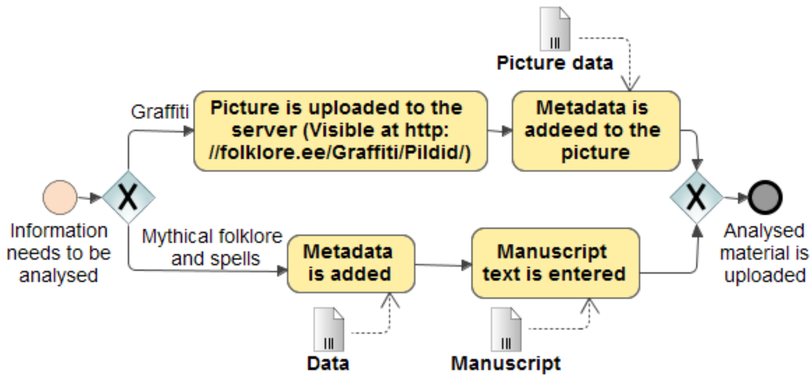


Figure 32. FO process model

The data collections represent values inserted into the software system. Data collection *picture metadata* includes values: graffiti text, typology, translation, location, date of collecting, collector (photographer), language, technique, keywords. Collection *data* includes values: PID, topography, collector, performer, collection year, keywords. *Manuscript* data collection includes values: content, inserter, date of inserting, controller, date of controlling, corrector, date of correcting.

4.12 Summary

In this chapter the results of the case study at the LM were described and the resulting business process models constructed. First the process map and the main processes were elicited and the scope was defined as: analysing information and uploading analysed material. Next the variation drivers were identified and the relative strength of the drivers assessed. Then the variants of each sub-process were identified and the similarities of variants assessed. Based on the results a variation map was constructed. Finally the business process models for each department of the LM were elicited.

5. Software System Support for EKLA

In this chapter, the scope for requirements engineering is specified. Next the software system support for the process is defined and requirements for the software system elicited. Finally the requirement specifications are created using the method of requirements elicitation from BPMN models (see Section 2.5).

5.1 Scope

The scope for the requirements elicitation is EKLA process of analysing information and uploading analysed material (see Figure 23). The requirements elicitation will be performed using only the non-physical activities that require some software system support (see Figure 33).

5.2 Digitising

The software system support for the sub-process of digitising (see Figure 20) is visible in Figure 34. The participants in this process are: EKLA personnel (user), personal computer (PC) and Kivike. The PC must be able to: create new files, send notifications to the user. Kivike must be able to: create new presentations, add metadata to presentations, add files to presentation, send notifications to user.

5.3 Insert Into DB

The software system support for EKLA process of insert into DB (see Figure 20) is visible in Figure 35. The participants of the process are: EKLA personnel (user), Ellen and Kivike. Ellen must be able to: create a new fund, add metadata to fund, add description to fund, create new artifact record, insert artifact metadata, collect data for transfer into Kivike, transfer collected data to Kivike, send notification to the user. Kivike must be able to: create new artifact record, insert artifact metadata, collect transferred data, insert new data to database, send notification to the user.

5.4 Add Presentations

The software system support for EKLA sub-process of adding presentations (see Figure 20) is visible in Figure 36. The participants in this process are: EKLA personnel (user), personal computer (PC) and Kivike. PC must be able to: create new files, send notifications to the user. Kivike must be able to: create new presentations, add metadata to presentation, add files to presentation, link presentation to artifact, send notifications to user.

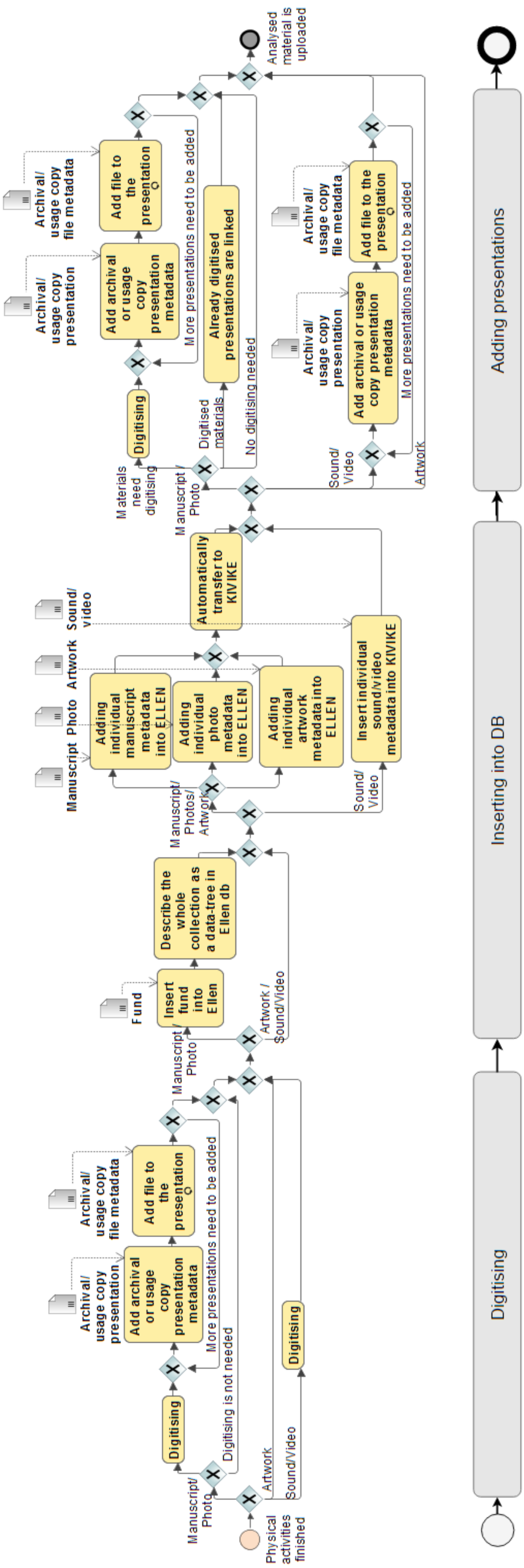


Figure 33. EKLA non-physical activities model

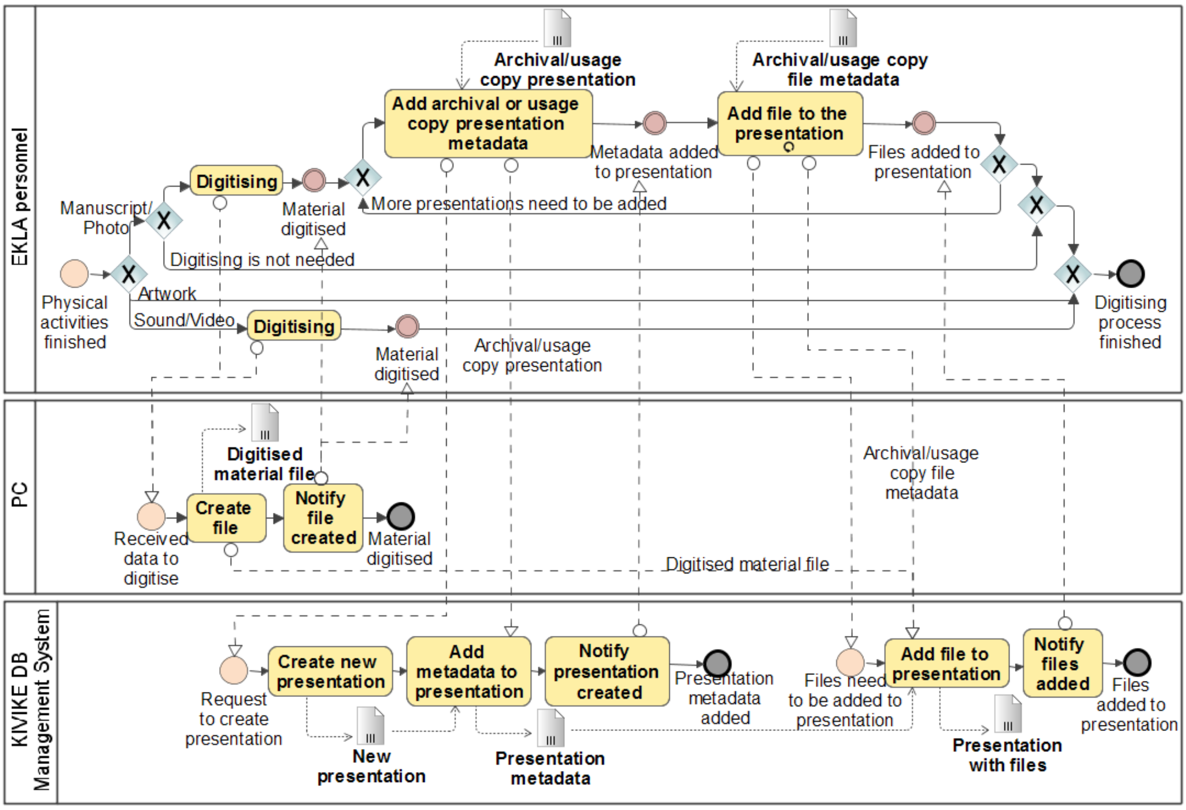


Figure 34. Software system support for digitising

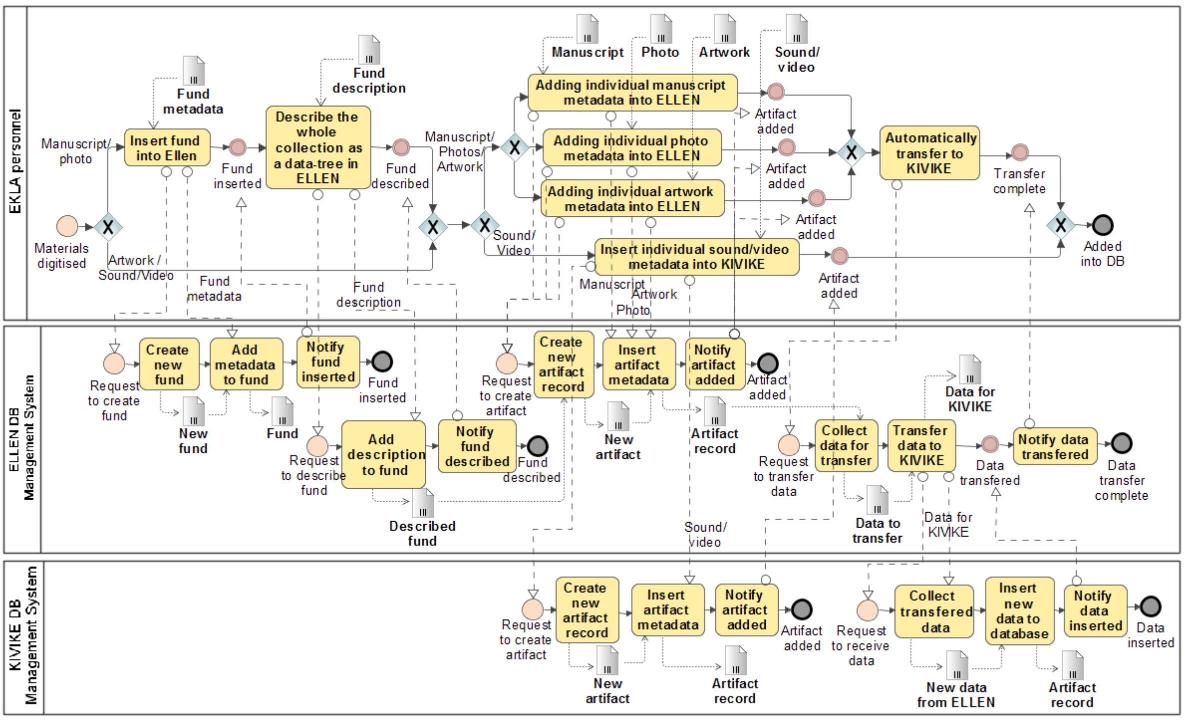


Figure 35. Software system support for insert into DB

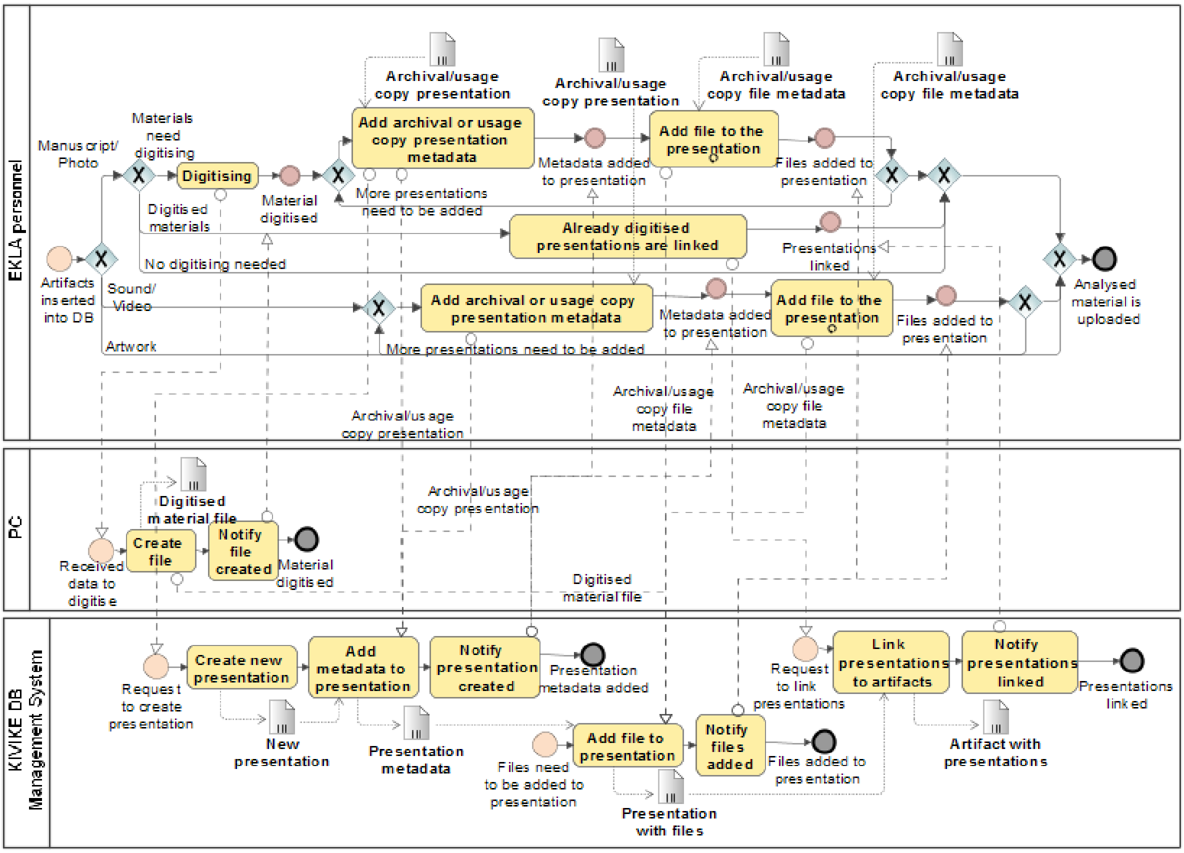


Figure 36. Software system support for adding presentations

5.5 Requirements Specifications for Digitising Process

For requirements elicitation, the *EKLA sub-process of digitising* (see Figure 20) from the process of analysing information and uploading analysed material will be used. The method for the elicitation of requirements proposed by Valvas [2] is described in Section 2.5. The steps (illustrated in Figure 10) will be applied on every activity performed during the digitising process in EKLA (see Figure 34). The result will be a collection of requirement specifications (template visible in Figure 9).

The first activity is *digitising* and it is the same for manuscripts, photos and sound/video. The activity requires software system support and thus it is relevant and given a unique ID number. Name of the activity and the process model are filled in the requirement specification. During goal elicitation it became apparent that the goal of digitising is to create digitised file. The *actor* performing the activity is EKLA personnel. The *actor* knows when to start the activity because the same actor is also responsible for the preceding activity. Digitising activity is *triggered* once the physical activities are finished. The *operational steps* start with EKLA personnel (user) requesting material to be digitised, once PC receives data for digitising it creates a digitised material file and then notifies user that file has been created. No alternative paths nor failure conditions and handling for digitising process were elicited. The resulting requirement specification is visible in Figure 37.

The next activity is *adding archival or usage copy presentation metadata* and it uses software system support. Thus the activity is *relevant*, it is given a unique ID number and then the business process name and the name of the activity are filled in the requirement specification. Eliciting *goal* indicated that the goal of the activity is to add metadata to presentation. The *actor* performing this activity is EKLA personnel (user). The actor is the same for the preceding activity and thus the activity is *triggered* once the preceding activity of digitising material has finished.

Component	Description
ID:	1
Business Process (optional):	Analysing information and uploading analysed material (ELKA)
Activity:	Digitising
Goal:	Create digitised file
Primary Actor:	EKLA personnel
Trigger:	The physical activities are finished
Steps of activity (positive scenario):	Operational steps: Step 1.1: EKLA personnel requests material to be digitised Step 1.2: PC creates digitised material file Step 1.3: PC notifies user that file was created
	Alternative paths: -
Failure conditions and handling:	-

Figure 37. Requirement specification: Digitising

The *operational steps* start with user sending the presentation metadata to Kivike, once Kivike receives the request, it creates a new presentation, then Kivike adds the metadata to the presentation and finally Kivike notifies the user that presentation metadata has been added. No alternative paths were elicited during the meetings. No failure conditions were also elicited. The resulting requirement specification is visible in Figure 38.

Component	Description
ID:	2
Business Process (optional):	Analysing information and uploading analysed material (ELKA)
Activity:	Add archival or usage copy presentation metadata
Goal:	Adding metadata to presentation
Primary Actor:	EKLA personnel
Trigger:	Material digitised
Steps of activity (positive scenario):	Operational steps: Step 2.1: EKLA personnel sends presentation metadata to Kivike DB Step 2.2: Kivike creates new presentation Step 2.3: Kivike adds metadata to the presentation Step 2.4: Kivike notifies user that metadata is added to presentation
	Alternative paths: -
Failure conditions and handling:	-

Figure 38. Requirement specification: Adding presentation metadata

The final activity is *adding file to the presentation* and since it uses software system support, it is considered *relevant* and given a unique ID number. The name of the activity and the business process fields are filled in the requirement specification. *Goal* eliciting revealed that the expected outcome is that files are added to the presentation. The *actor* performing this activity is EKLA personnel (user). The actor is the same as the actor of the preceding activity. The activity is *triggered* when metadata has been added to the presentation. The operation steps begin with user sending archival/usage copy file metadata to Kivike, then Kivike adds file to the presentation and finally Kivike notifies user that files are added. The resulting requirement specification is visible in Figure 39.

Component	Description
ID:	3
Business Process (optional):	Analysing information and uploading analysed material (ELKA)
Activity:	Add file to the presentation
Goal:	Files added to presentation
Primary Actor:	EKLA personnel
Trigger:	Metadata has been added to the presentation
Steps of activity (positive scenario):	Operational steps: Step 3.1: User sends file metadata to Kivike Step 3.2: Kivike adds file to the presentation Step 3.3: Kivike notifies user that file was added to presentation
	Alternative paths: -
Failure conditions and handling:	-

Figure 39. Requirement specification: Adding files to presentation

5.6 Summary

In this chapter, the software system support for EKLA process of analysing information and uploading analysed material was described. Then software system requirements were elicited. Finally requirement specifications were elicited from BPMN models following the method of requirements elicitation from BPMN models (see Section 2.5).

6. Conclusion

In this chapter, first the threats to the validity of the results are described. Then the lessons learnt during the application of the method for modelling business processes with variation and the method for requirements elicitation from BPMN are presented. Finally,the future work is described.

6.1 Threats to Validity

There are several aspects that influence the resulting models, elicited software system requirements and requirement specifications:

- analysis conductor's subjectivity and incomplete background knowledge of museum work and processes;
- representative's subjectivity;
- models of different departments differ in level of detail;
- variability of the same process in different departments causes errors;
- time limitation for concentrating on describing the business processes during the elicitation of process models;
- the lack of time during the application of the method for requirements elicitation from BPMN models also limited the communication with the museum personnel.

In order to mitigate the threats, preventive actions were taken.

- In order to reduce the analysis conductor's subjectivity, all knowledge of museum work was gained through communication with the museum representatives.
- In order to reduce the representative's subjectivity, two representatives were interviewed when possible.
- In order to prevent errors caused by differences between the departments, the representatives from each department were asked to focus on the individual department and not compare their processes to other department's processes.
- In order to unify the level of detail in the models, additional questions were asked from the representatives when necessary.
- The results of the case study will be presented at the LM in order to ensure the validity of the results.

6.2 Lessons Learnt

The case study with the LM showed that the method for modelling business processes with variation (see Section 2.4) helps to structure business processes in an organisation. While considering the first research question (RQ1) we have learnt the following lessons.

- Close collaboration should be established with the executors of the process. There should be iterative meetings, agreements regarding the process models.
- Different people tell the same story differently, with different level of abstraction. This method helps group the different levels of abstraction.
- Method guides elicitation of business processes to find missing levels of abstraction.
- Application of this method is time consuming. Due to time constraint, the method was not applied for the FO department.

The case study at the LM showed that the method for requirements elicitation from BPMN models helps elicit requirements from BPMN models. While considering the second research question (RQ2), the following lessons were learnt.

- Close collaboration with the process executors should be established. Applying the method requires iterative meetings with the representatives of the LM.
- People might not have complete understanding of how the software system works. To elicit requirements for the software system, experts with in-depth knowledge of the software need to be involved.

6.3 Future Work

The results of the case study are illustrated in Figure 12. Future work includes the elicitation of requirements specifications for other processes in EKLA. Also defining software system support for the process of *analysing information and uploading analysed material* for AR and ERA departments and then creating the requirement specifications. For FO department, first the business process models need to be created, then software system support defined and finally the requirement specifications created. For the process *get information uploaded during registering* process models need to be created, software system support defined and requirement specifications created based on the models.

For the other core processes *Registering* and *Scientific Research* the method for modelling business processes with variation (see Section 2.4) needs to be applied and the process models created. Then software system support for the processes needs to be defined. Finally requirement specifications need to be created using the method for requirements elicitation from BPMN models (see Section 2.5).

Once the requirement specifications have been created for all processes of the LM, all issues in the as-is processes need to be identified and documented. Then *process analysis* phase in the BPM lifecycle (see Figure 1) is completed and the following phases need to be executed for all processes. In the next phase *process redesign* potential remedies for the issues are analysed and a redesigned version of the process is proposed. Then during *process implementation* the necessary changes for the execution of the to-be process are implemented. Future adjustments in the new process are done during *process monitoring and controlling*. The implemented changes in the process may cause new issues to arise and thus create a need to repeat the BPM lifecycle.

References

[1] Milani, F. (2015). On Sub-Process, Process Variation and their Interplay: An Integrated Divide-and-Conquer Method for Modeling Business Processes with Variation. University of Tartu Press.

[2] Valvas, S. (2015). Requirements Elicitation from BPMN Models. Master’s thesis, University of Tartu, Tartu.

[3] Dumas, M., La Rosa, M., Mendling, J. & Reijers, H. A. (2013). Fundamentals of Business Process Management.

[4] OMG. (2011). Business Process Model and Notation (BPMN). Retrieved May 8, 2016, from http://www.omg.org/spec/BPMN/2.0/

[5] Kotonya, G. & Sommerville, I. (1998). Requirements Engineering- Processes and Techniques.

[6] Milani, F., Dumas, M., Ahmed, N. & Matulevičius, R. (2016). Modelling families of business process variants: A decomposition driven method. Information Systems, 56, 55-72.

[7] Estonian Literary Museum. Retrieved May 8, 2016, from http://www.kirmus.ee

[8] Eesti Kirjandusmuuseumi folkloristika osakond. Retrieved May 8, 2016, from http://www.folklore.ee/rl/fo/

Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Laura Lilleste**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:

    1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

    1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

of my thesis

Structuring Business Processes and Eliciting Requirements at the Estonian Literary Museum,

supervised by Raimundas Matulevičius,

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, **12.05.2016**