FAKULTÄT FÜR INFORMATIK

DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Information Systems

Guideline for combining and differentiating between CMMN, DMN and BPMN: An indicator-based use case study

Dominik Gerbershagen

FAKULTÄT FÜR INFORMATIK

DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Information Systems

Guideline for combining and differentiating between CMMN, DMN and BPMN: An indicator-based use case study

Richtlinie zur Kombination und Differenzierung zwischen CMMN, DMN und BPMN: Eine indikatorbasierte Falluntersuchung

Author:	Dominik Gerbershagen
Supervisor:	Prof. Dr. rer. pol. Hans-Arno Jacobsen
Advisor:	Martin Jergler, M.Sc.

Date: November 15, 2016

Ich versichere, dass ich diese Bachelor-Thesis selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

I confirm that this bachelor's thesis is my own work and I have documented all sources and material used.

Munich, November 15, 2016 Dominik Gerbershagen

Abstract

Acknowledgements

Acronyms

CMMN Case Management and Notation Model

DMN Decision Model and Notation

OMG Object Management Group

BPMN Business Process Model and Notation

BPM Business Process Management

ARIS Architecture of Integrated Information Systems

BPR Business Process Redesign

IBM International Business Machines Corporation

BPMI Business Process Management Initiative

BPEL Business Process Execution Language

UML Unified Modeling Language

DRD Decision Requirement Diagram

CM Case Management

ACM Adaptive Case Management

DRG Decision Requirement Graph

FEEL Friendly Enough Expression Language

SQL Structured Query Language

PMML Predictive Model Markup Language

URI Uniform Resource Identifier

XPDL XML Process Definition Language

WSBPEL Web Services Business Process Execution Language

XML Extensible Markup Language

REST Representational State Transfer

Contents

List of Figures

List of Tables

Part I.

Introduction

1. Introduction

1.1. Motivation

1.2. Problem Statement

The Business Model and Notation (BPMN) modeling language and technique has become a quasi-standard for modeling business processes, logical steps in software-systems or align companies along the process chains. BPMN was standardized in 2005 by the Objective Management Group (OMG), but had a long list of predecessors including the *Event-driven Process Chain* (EPC), the *Swimlane Visualization*, *Business Process Re-engineering* to mention only a few. Despite the inheritance of these languages, BPMN has some major downsizes which will be discussed. BPMN suits best when it comes to processes that incorporate the *Value Chain Model* by Michael E. Porter [2]. Every company needs these strict processes to optimize the value chain, for the value creation and to separate the hierarchies between employees and departments.
By 2016 these processes have been largely automated. For this automation, the BPMN syntax suited very well due to the strict processes that could be automated more or less easily. The next challenge is to support processes that cannot be simply automated: decisions and case management. At this point, *Case Management Model and Notation* (CMMN) and *Decision Model and Notation* (DMN) come into play. Both are were highly anticipated by people working in modeling departments. This thesis will investigate the benefits of separating the decision logic and case management into these new standards as well as combining them into a macro model. Additionally, every language has its own indicators and way of modeling. The weaknesses and strengthens of BPMN, CMMN and DMN will be presented in detail in order form indicators for when to use which language best. Both modeling languages are relatively new to the market as they were standardized in 2014 (CMMN) and 2015 (DMN). The goal of this thesis is to provide a guideline helping modelers to differentiate between the languages, to help combining each language and to provide information about the historical background

of these new standards. Another key aspect in this thesis is the use case study to prove the indicators by real process models taken from the eKulturPortal GmbH. An evaluation will show how the indicators applied on a model work and how they can work for the reader's own models.

Summing up the preceding paragraph leads to three research questions:

- Investigation of the new Decision Model and Notation specification published by the Object Management Group, extracting the downsides and advantages especially concerning the vast modeling of gateways in BPMN. Is DMN the solution to simplify decision-modeling?
- Investigation of the Case Management Model and Notation specification by the Object Management Group, particularly how case modeling can be applied in a model-driven software development project.
- How do CMMN, DMN and BPMN work together in a model-driven software project? Is there a valid possibility to combine all three specifications in one model? Is it possible to improve the process and information flow, readability and eventually implementation of the model by the developer?

1.3. Approach

To start the analysis of each modeling language, a brief background information about the demand for standardization by people working in modeling departments or researchers will be provided. On this account a thorough literature research suits best, presenting the results in the Background chapter.

In chapter 2 we will examine the specifications provided by the OMG and have a look at each modeling language. What are the strengthens and weaknesses, what has been standardized and when do the standards suit best. These questions will be answered for each CMMN and DMN. BPMN will not be analyzed in detail, but is a key element to do the comparison between the standards. The goal is to carve out the scope, requirements which were met and indicators of each language. In the following chapter, the ability to combine each standard with BPMN will be presented. At first, the modularity of each standard will be determined. Is each language able to stand for its own or does it need to be implemented in a larger model to work properly? Afterwards, the results are taken in order to compare the combination aspect of BPMN, DMN and CMMN. The questions are directed on how to combine each language, whether a model with all three languages is possible or if there are any downsides that prohibit combining the languages. Additionally, the connectors of each model will be carved out.

With these findings obtained we will do the use case study in order to demonstrate the robustness of the indicators. Process models from the eKulturPortal GmbH will be analyzed with regard to the indicators and remodeled to show the working concept of each standard. A demonstration of the connectors, the scope of each language and the requirements which were met will be presented in this chapter.

Closing up this thesis a chapter of evaluation and discussion is succeeding and the conclusion with a summary of the findings and proposals for future work.

1.4. Contributions

1.5. Organization

Part II.

Related Work

2. Related Work

2.1. BPMN

– todo

Geplant: Am Ende der Ausarbeitung

2.2. CMMN

Case Management and knowledge work are not brand new inventions that have been created in the past few years. "Peter F. Drucker made the first reference to knowledge work in (...) 1959 (...)" [3]. A current "overview and research challenges" provide [3] who explain the difference between business process management and adaptive case management. They briefly sum up the state of the art in case management technology and the next generation solutions. Mentioning technology and tools for Case Management, CMMN and Adaptive Case Management, there are many articles dealing with these topics. [4] describe how adaptive case management can be implemented in businesses and integrated in Enterprise Resource Planing systems (ERP). Additionally they approach a new architecture which decouples decision logic, knowledge work and process flows. All this leads to a better handling of information and an optimization of business modeling. Another practical example provide [5] explaining the company's approach towards an implementation of the CMMN paradigm. This includes the ability to change requirements or orders during run-time, which is one of the major aspects in their system. To achieve this goal, they first set up a meta model of their order-based system and enhanced it afterwards. These practical examples are important in order to evaluate the compatibility with the CMMN specification and other modeling languages, specifically BPMN. They also provide a good overview of how to combine modeling techniques and how they are realized as a system in companies. A more theoretical approach to case management and CMMN particularly provide [6] and [7]. They both do research on transforming CMMN into different

languages. [6] do model-to-model transformation from CMMN to DDML (DEVS-driven Modeling Language) which is used to formalize CMMN and analyze it afterwards. [7] have a similar approach, but a different goal. Due to weaknesses of CMMN, the language cannot be used to create a platform for both agile and route processes. They describe agile processes as the ones "(...) of which the exact flow cannot be determined completely a priori" [7], which is a fundamental characteristic of knowledge work and the reason why case management is so important for many industries. Coping with CMMN's downsides they build their platform on a "rule-based cross-perspective and model intermediate language on textual basis, (...) called *Declarative Process Intermediate Language (DPIL)*" [7].
A useful source for evaluating CMMN as a standardization for adaptive case management is [8]. The subtitle *Examining the applicability of a new OMG standard for adaptive case management* is a good foundation to see how OMG met the expectations from the industry and researchers. This paper sets up requirements deriving from different sources described in detail in section two [8]. At the end of their paper, they evaluate how good the requirements were fulfilled by the CMMN standardization and provide feedback for future improvements.

2.3. DMN

The Decision Model and Notation standardization was meant to improve the *separation of concerns* [9] which is the decoupling of decision logic and the control-flow. Biard et al. investigate how the new standard DMN can be used for decoupling BPMN and the decisions modeled as gateways. Decision-modeling is not typically included in control-flow oriented modeling languages. BPMN has not the power to model vast decision-trees due to the gateway restrictions. [10] even calls it a "(...) [misuse] for modeling decision logic". They found an autonomous way of separating the concerns. After averaging more than 900 models from different industries they introduced a "(...) semi-automatic approach to identify decision logic in business processes (...)" [10]. This semi-automatic approach incorporates the 900+ models they used to identify patterns in decision modeling. Formalization is not one of the key issues in this thesis, but the translation of BPMN to DMN or the link between them definitely is. Evaluating the compatibility of DMN with different modeling languages has been the objective of [11]. They approached a combined solution for knowledge-intensive work modeling and extracting the decision logic, what lead them to a new language called *Declare-R-DMN*. Although the Declare language is not part of this thesis, the combination of it with DMN is useful to evaluate the compatibility with BPMN and CMMN.

Part III.

Background

3. Background

3.1. Origins of BPMN

The evolution of business processes and process thinking dates back to the 1980s when Michael Porter developed the *Value Chain* making a first proposal on how to align companies along the business processes [2]. This major step initiated consecutive research and improvements in the field of business process engineering and re-engineering. The latter has been a very popular technique for companies to strip down their legacy processes, optimize them and implement the new ones. A very prominent example is Ford who copied the Mazda's concept of a central database replacing an old fashioned paper stream [12].
Inspired by Ford and Mazda, the *Business Process Redesign* (BPR) was emerging in until the late 1990s. At that time, companies tried to redesign in a radical manner the way processes, people and data keeping works. Dumas et al. define four reasons for the fall of BPR, whereof over-radicalism might be the most influential one [12].

Despite the fall of BPR, the management of business processes became a prominent part of business optimization. Consequential different methodologies like the *BPM lifecycle* or the *Architecture of Integrated Information Systems* (ARIS) were born along with new modeling languages. "ARIS [also] provides a modeling language known as event-driven process chains (EPCs)" [13]. By today, ARIS still is a very popular, thus a well known, methodology for enterprise modeling and EPCs a widely used modeling language for event-driven processes, as the name states.
ARIS was published 1994 by August Wilhelm Scheer, a researcher for Business Information Systems at the Saarland University. Around ten years later, Stephen A. White and the *Business Process Management Initiative*, consisting of roughly 35 individuals and companies total, published the first version of the new *Business Process Modeling and Notation* (BPMN) concept [14]. In 2006, the OMG adopted BPMN as well as the BPMI and published an overhauled version, BPMN v. 2.0 in 2011. This version was highly anticipated and had to meet high expectations. The goal was

to create a language with the ability to interchange modeling tools. New parts were added such as the *Choreography diagram* for modeling data exchange between partners and the *Conversation diagram* showing the relationship between several partners [14].

3.2. Origins of CMMN

Case Management and Notation Model (CMMN) was standardized by the Object Management Group (OMG) in 2014 which represented a peak in terms of notation for the case management research. However, research began more than ten years ago when Wil Van der Aalst first published his paper about case handling offering a new approach for flexible processes [17]. He summarized different opinions from several authors into the statement that contemporary workflow-systems were not able to handle flexible processes. To be precise, he identified four problems:

- work needs to be divided into atomic steps, so called tasks
- the routing of activities was used to distribute the tasks and authorize workers the workers to do them
- *context tunneling*: workers only focused on the process flow but not on the context surrounding the activities
- the individual's focus was more on what *should* be done instead of what *cane* be done

He portrayed a *blind surgeon* as a metaphor who incorporates the four problems in his daily work, for instance refusing a blood sample when it was necessary but not part of the problem in case of what could be done and what should be done [17]. Consequently he solved the problems inventing an adapted model for cases, which consists of objects that can be found in the OMG'S specification from 2014 as well. In 3.1 a comparison of the key objects of Case Handling and CMMN is provided. The Case Handling's key objects are:

- activities or tasks which are the atomic unit of work
- Actors with roles who execute, skip or redo the tasks
- Case representing the product the actor is producing
- data objects for information and values

Table 3.1.: Comparison CMMN and Case Handling Model

	van der Aalst	CMMN Specification
Product	Case	Case
Atomic Unit of work	Activity / Task	Task
Information Handler	Data Object	Case File Items
Information Representation	Forms	
Executing Person	Actors and Roles	Human tasks with asignees
Connectors	not specified	specified
Triggers	not specified	Event Listeners

- forms that present data objects from different point of views

Van der Aalst proved his concept practically in a joint venture with dutch Construction company. They applied the Case Handling model and built a prototype with FLOWer resulting in a rudimentary worklfow management system. They requested workers to use the system during their work and evaluated the prototype in the end. Van der Aalst calculated a Return on Investment (ROI) of 0.7 to 1.4 years in the year 2002 [17]. With today's workflow management systems, the ROI would presumably be much higher.

3.3. Origins of DMN

The Decision Model and Notation language describes processes in a different way as Business Process Model and Notation (BPMN) does. They differ in their way of how to achieve the actual goal of the process. BPMN, for instance, describes an imperative way. Goedertiert et al. define imperative as "[...] [focusing] on providing a precise definition of the control-flow of the business process in a graph-based process modeling language" [18]. In order to model decisions, BPMN has a rich amount of gateways to direct the control-flow into the desired tasks depending on the decisions being made. Additionally BPMN provides a Business Rule Task sending data to and receiving it from a business rule engine [19]. However, this business rule task " [...] cannot be decomposed any further - and involves no user interaction" [20]. Thus the modeler is not able to define roles for decision, define any requirements for these and is not able to decompose them into fine-grained steps. This might be unnecessary for simple decisions, but in larger and more sophisticated diagrams, complex decisions cause huge branches in the control-flow making the diagram confusing the reader. [21] formulated *Seven Process Modeling Guidelines* according to an empirical study. They built an EPC model with various branches connected by *AND*, *OR* and *XOR* connections. Additionally they violated naming conventions and used multiple end points. One of the resulting guidelines (G1) states "Use as few elements in the model as possible" [21], which is contradicting the imperative way of modeling decisions in BPMN. In this situation, declarative modeling approaches are more appropriate. According to Alan Fish, declarative languages "[...] focus on what should be done in order to achieve business goals, without prescribing how an end state should be reached" [20]. The CMMN specification is a declarative modeling approach as well, whereas EPCs and BPMN aren't. Fish also described a way to refine the decision requirements. He invented the Decision Requirement Diagram (DRD) to show " [...] the structure of the required decision-making as a network of decisions and subdecisions with their supporting areas of business knowledge and data" [20]. The main use for this DRD is to make knowledge involved in decision-making explicit instead of implicit. In the Decision Model and Notation (DMN) Specification by the Object Management Group, a DRD can also be found. This DRD includes the elements as Fish invented for his DRD: Decisions, Input Data (Fish: Data area), Knowledge Source (Fish: Knowledge Area). However, the OMG's DRD is a much more refined modeling approach since it incorporates Fish's DRD and is under constant development. Summarizing the findings, the DMN technique is derived from Fish's findings on how to automate decisions and the declarative modeling approaches which supplement BPMN with tools to model processes without strict control-flows.

3.4. eKultur GmbH

This thesis not only investigates the OMG's specifications, but also intendeds to serve as a practical guide and example for analysts, modelers and further interested readers.
The eKultur GmbH was founded in 2012 with the objective of digitizing the branch of touring theater companies. As a theater company's spin-off, eKlutur serves both as a domain expert and a software developing company. Supported by the German Federal Ministry for Economic Affairs and Energy, the project has a dual role

as a company with the aim of creating a product on the one hand, as well as creating a reference model for different branches on the other hand. This thesis' results contribute to the applied research done by this project's participants and helps modeling the processes implemented later in a middleware system.

bisschen besser ausformulieren an dieser Stelle

Part IV.

Indicators

4. Differentiating between BPMN, CMMN and DMN

Starting with process management for some companies means to start from the very bottom. They build up their infrastructure, workflows and eventually models on a greenfield. The following sections provide a brief overview of process discovery techniques (manually as well as automated) and serves as an introduction for further reading. It starts with data sources and ends with process maps. Building on process maps and the associated documentation, indicators for the differentiation between BPMN, CMMN and DMN are introduced. Terms and categories will be explained as well as the derivation of each set of indicators along with an explanation of the associated standard.

Diagramm mit Legende versehen

Einleitung ins Kapitel erscheint mir noch etwas holprig

4.1. Discovering the processes

Processes are resources. Similar to iron or copper, processes lie deep in the bedrock of a company and need to be discovered. This metaphor is not even far-fetched, since the correct terms for gathering information is *process mining* or *process discovery*. In this chapter, the aggregation of information before the actual process modeling will be discussed, specifically two approaches: A manual one (often referred to as process discovery) and an automated one (often referred to as process mining). Both techniques will be discussed briefly and put in the context of the eKulturPortal venture.

4.1.1. The manual way

According to the BPM Lifecycle [12], gathering information begins with identifying the process. This first step poses the question that needs to be answered and leads to the accumulation of all the relevant processes surrounding this questions [12]. Ideally, the accumulation shows the posed question in a map surrounded by all the relevant processes. The way this map is created is up to the involved personnel. It can very from a mind-map to a more detailed flowchart or even something like a BPMN or EPC diagram. In the end, this map is useful for the following step: the process discovery.

Laury Verner identified three "key players" [22] in the process discovery phase:

- "The Sponsor", who decides the scope and the goal that needs to be achieved and is also called the process owner.
- "The Subject Matter Expert" (SME) who is also referred to as the *domain expert* since he is the person working with the processes.
- "The Analyst", often referred to as Business Analyst, who is the expert for modeling and designing the processes.

These three roles are commonly assigned to various people in large ventures. However, in this can also be small team according to the size of the departments. To begin the process discovery, the process owner sets the goal such as optimizing the a department's workflow or integrating automation into manual processes. He is the one who is responsible for the whole project and has to monitor the progress. The SME or domain expert is the employee working in the process. SMEs are the subject of every process discovery technique due to their role as knowledge source. A process analyst has to extract the knowledge they implicitly keep and to visualize it as explicit processes. The one created are most commonly not the ones which are presented as results. However, it is important for the progress of such a project to visualize the gathered knowledge.

The following process discovery technique are summarized by Marlon Dumas et al. in [12] in a theoretic way, which are emphasized by [22] and [23]. Dumas categorizes three different approaches in terms of process discovery: The "evidence-based discovery", the "interview-based discovery" and the "workshop-based discovery" [12]. Each approach deals with different input sources.
Beginning with he **evidence-based discovery,** it deals with documents, models, observations and event logs[1]. The process analyst may choose different approaches in the evidence-based category: He may either examine the documents and aggregate the information the can be found there; he may also choose to observe the employee by playing an active participant in the process or just passively watching the employee work; or he may work with the event logs according to [12]

[1]Dumas et al. includes event logs in his manual approach for the reason of integrity

which will be explained in the automatic section in more detail. Each decision bears several advantages and challenges that need to be kept in mind:

Advantages:

- Choosing the document analysis, an analyst can identify workflows in the company and the cooperation of different departments. He gets a big picture of how the company works and can locate the as-is process better [12].
- Documentation provides an unbiased view of the processes, if it is not written by a single person.
- Observation is good to get a detailed understanding of how a process works. The analyst observes the SME by initiating either an active instance of a process (e.g. ordering a product as a customer) or by playing a passive role and just observing how the employee works [12].
- Observation additionally shows the boundaries, milestones and interfaces to different processes.

Challenges:

- Documentation is not written process-ready, so the analyst needs aggregate the information to fit to the process.
- The work described in documentation might not fit the granularity level of the process [12]. Often the steps are documented too detailed or, contrary, the documents only describe the big picture but leave out essential interfaces to different processes.
- The biggest problem of documentation is the date of creation and to keep it up to date: Most documentation is written once and does not fit the as-is processes. Current adaptations to the processes might not be captured.
- Observation might lead to wrong results because employees work different when they are observed. They might want to show the best practice example and leave out occurring failures.
- Observing an employee might only show one perspective of the process but not different ones such as the customer perspective. In this case, the analyst switch the passive and active roles to get a thorough overview.

The **interview-based strategy** works in a different way: The analyst sets up an interview with a domain expert. In this case, there is no specific input data involved except from the knowledge the domain expert has. Dumas et al. provide two paths the interviewer can guide the SME through the interrogation: backwards (from the product to the start) or forwards (start to product). Each direction has advantages: backwards show the input data the process has to wait for and forwards clarifies the path of the process and how the employee works. [12] Interview-based discovery has the following advantages and challenges:

Advantages:

- Processes can be explained in detail by the persons directly involved. This results in a current view of the processes with advantages and downside directly from the workers. Later, the process design benefits from this additional information.
- Different routes show the process from various angles. "This is particularly helpful for understanding which decisions are taken at a given state" [12].
- By interviewing various employees, a centralized documentation of the knowledge can be created that might not have been existing before.

Challenges:

- Interviews need need a good plan: each process activity needs to be clarified, the expected outcome of the process needs to be defined, all the required input data from interface processes has to be figured out and all the subsequent processes need to be mentioned.
- The analyst might need various iterations. Each interview leads to a script that needs to be transformed into a model, which then has to be verified by the interviewee again.
- Structured vs. free-from interviews: In the free-from interview, SMEs are completely "uncensored" [22] and free to mention everything they associate with the process. In structured interviews, workers answers to "pre-defined questions" [22]. The former method captures a larger picture of the process, the latter one provides a better understanding and more details for the analyst.
- Similar to the evidence-based method, the interviewee might keep common failures secret that need to be figured out by the analyst.

A third method is the **workshop-based discovery**. Unlike in interviews, workshops include a group of SMEs, several analysts and one or more process owners. In these situations it is eminent to have a *facilitator* who is responsible for the "verbal contributions" [12]. A good practice is to create the models directly in the workshops. The resulting illustrations are not the desired output, but it is possible to create a process map or brief processes that can be revised in further iterations. Dumas calls the responsible person the *tool operator* [12].

Advantages:

- A workshop combines different views on a process at once and offers the possibility to create a rich and detailed model through several iterations [12].
- Models can be created simultaneously.
- Involved analysts have the same level of knowledge, thus have an advantage in the design stadium.

Challenges:

- Workshops need several iterations and are very similar to meetings. Each meeting needs a good preparation, a minute keeper and someone taking notes. In this case, the facilitator and the tool operator need to be well prepared and interact ad hoc with the participants.
- The attendance of process owners may influence the employees, consequently they might not talk about failures or downsides of steps.
- A level of detail needs to be defined before the meeting starts in order to differentiate between unnecessary tasks like "putting the document on a fax machine" [12] and the ones essential for the model.
- Workshops are very time consuming and need to be planned in advance in order to unite all the roles in the workshop.

Apart from the three described methodologies, their challenges and advantages, there are a few meta-questions that need to be answered according to [22]

- A centralized vs. a distributed approach: The analyst need to decide whether workshops or interviews suit best in the specific situation.
- Top-down vs. bottom-up: Is it better to start from the top (e.g. the output of each process) and proceed until the smallest tasks are reached, or is it better

to start from the bottom (the smallest tasks) and aggregate the model from the bottom up?

- Free from vs. structured: As mentioned in the interview-based strategy, a workshop or interview can be held in a free from way or in a structured one.

Verner also offers a recommendation that we also use in the eKulturPortal context: [22]:

1. Start with a top-down view of the processes. Cluster the processes in a map for navigation purposes and get an overview of how the company works. This can be done by interviews and documentation analysis.
2. Use bottom-up structured interviews to clarify the details and use the created map to validate them. The resulting outcome is an as-is process model that needs to be verified in step three.
3. Create integrated models of the processes and validate them with stakeholders such as process owners in a workshop. These models are a good base for discussions and lead to structured outcomes.

Each scenario is individual and needs its own mix of approaches. However, the recommendation from Laura Verner worked for the eKulturPortal venture in a highly unstructured context where knowledge is implicit and a mix of documentation analysis, interviews and workshops were necessary. Although adapting the mixture, the actual structure of the recommendation was maintained. In the eKulturPortal context, there hasn't been any as-is documentation but SMEs that were observed and interviewed in a free-form way. Additionally, workshops were nearly impossible to hold due to the missing facilitator and tool operator. The involved domain experts had dual roles combining an SME and a process owner which made it complicated to differentiate between process knowledge and goals that need to be achieved. The outcome was a combined approach of observation and interviewing simultaneously.

4.1.2. The automated way

Small companies or such without any supporting IT infrastructure might rely to the manual way of process discovery. However, there is an option for automation, the so called *automated process discovery* (ADP) [12]. This method relies on the event logs of a workflow management system. ADP is a sub-category of the

Process mining research field, which will be discussed briefly in this section. The term process mining originates from the very similar technique *Data mining*. Both approaches use a set of input data to form it into a desired output structure like a spreadsheet, trees or rules, cluster etc. [24]. Process mining combines three different strategies: the process *discovery*, *conformance* and *enhancement*. The first phase is corresponding to the preceding section: the system uses event logs to create models that have been unknown or implicit before. This phase works without any pre-defined input data apart from the logs. Conformance takes an existing process and the corresponding event logs to compare them. The goal of this comparison is to check whether the model matches the executed process and vice versa [24]. In the enhancement phase, the process generated from the event logs *enhances* the pre-defined models. The desired result is a model that matches the real executed processes by either repairing the models or extend them with another perspective [24]. Depending on the infrastructure the executing company has, its to it's discretion with which phase to start. For the purpose of this thesis, we start with the discovery phase and assume there are no a priori models given. The very first requirement for process mining is the conformance of event logs. Scanning data storage for event logs often results in various outputs. They might be scattered across departments, stored in various formats and - most importantly - not prepared for process mining. Solving this problem needs a three-way paradigm: *Extract, transform and load* [24]. At first, the data needs to be extracted from the data storage and aggregate in a central warehouse for the process mining. Next, the data needs to be transformed to meet the requirements. This means, the event logs need to have at least an identification for the corresponding process and an activity column [24]. With these two columns, the events occurring in the log can be ordered and the model can be generated with accordingly. Without either ordering or activities, there is no possibility to create a model. Third, the transformed tableau is loaded into the target system where process mining takes place. The more data is included in the logs, the more useful the output is for the later conformance and enhancement phase. Helpful additions are costs of paths and timestamps. With these two extra columns it is possible to analyze the critical path with e.g. the *Critical Path Method* (CPM) [25]. Besides, timestamps solve another problem of event logs: Imagine two instances of the same activity start at different times. Which one is the first that stops and which one the last? [24] illustrates this example for event logs with an insufficient amount of attributes.

After the event logs are transformed and loaded into the target system, the analysts need to decide how to cluster the data. A well-known example is the α-

algorithm invented and described by [24]. This algorithm takes an event log and creates a *WF-net*, which are "[...] a natural subclass of Petri nets tailored toward the modeling and analysis of operational processes" [24]. Apart form this theoretic approach, tools like ProM[2] offer plugins with algorithms that can be selected according to the desired input and output. An example is the Mutli-phase miner that generates and EPC from the causal dependencies in the event log [26]. This makes it easy for practical uses of the process mining approach and leads to fast results compared to the manual method.

Each of the mentioned phases (discovery, conformance, enhancement) are optional and it is to the analyst's discretion what needs to be executed. However, the discovery does not improve the as-is state but only displays it. It is recommended to do each phase of the process mining to get the desired results and to check, if the implemented processes achieve the intention designed in the models.

The process mining advantages are:

- Accurate results describing the real-time status of the process.
- Fast method that can be tracked in real-time.
- Real-time monitoring of processes in a map-like structure.

And the according challenges according to [27] and [12]

- Aggregating the data in a format the process mining tool can handle and the output requires.
- Understanding the resulting models, as they can be very complex and hard to read for the analysts.
- Asking the right questions: Taken into account that companies use ERP-systems with sometimes more than 10,000 data points, it is necessary to define the right question and extract the right data.
- Dealing with bias and noise resulting from vast amounts of data.

Both, the process mining and manual process discovery strategies provide methodologies that lead to the desired as-is and to-be models. It is up to the skills of the analysts, to the IT infrastructure of the company and the objective set for the whole project to choose the one.

[2]for further information see `http://www.promtools.org/doku.php`, 2018-09-12

4.2. Introducing the indicators

In the preceding section, two major approaches on how to aggregate input data were explained: The manual process discovery as a set of tools to get information from SMEs or process owners, the automated process discovery (or process mining) as a methodology to scan event logs of workflow management systems and generate models according to algorithms. Both techniques result in process maps and / or documentation, e.g. spreadsheets, textual descriptions and interview transcripts. The next step is choosing the correct modeling language to either do an as-is model or start with the design model. Most commonly, an as-is model is made first as a foundation for further analysis, followed by the to-be model.
In this section, the conduction of indicators will be introduced and explained. Indicators in general are tools that take an input and present the user a value or a choice that should be made. The subsequent indicators are intended to help the analyst to choose the appropriate modeling language for the identified processes. Of course, there are many modeling techniques on the market, which is why the scope of this thesis is limited on three modeling techniques by the OMG: BPMN, CMMN and DMN.
Table 4.1 provides the categorization and the corresponding definition of each category. Each standard is examined in the following section and the key aspects are summarized in the categories. The intention behind the indicator table is, that business analysts take their results from the process discovery phase and check whether their process maps, documentation and description of work, objectives etc. match the key aspects of a standard. In Chapter 6, a DMN decision logic is presented with the intention to automate the decision, which approach to use. There are matches that indicate a combined approach and matches that recommend one single language. However, there is also a certain threshold that indicates further analysis of the given information.
The first and foremost important step to use the decision logic in the named chapter is the correct understand of the indicators and the modeling languages. This section is intended to get familiar with the indicators and the categories, followed by practical explanations of each specification.

The categories in Table 4.1 still need to be refined to provide identification of a suitable modeling language. Immediately after process discovery, documentation and process maps exist, sometimes even deprecated process models complete the setup for design phase. Process mapping can be done in various ways and often is used to illustrate and cluster work packages or, ideally, tasks.

Indicators	Definition
Documentation style	During process identification and process discovery, the gathered information is usually denoted in a document. Each modelling language corresponds to a more or less specific style of documentation.
Preceding process map	Process discovery should result in macro models showing how processes cooperate or how single processes roughly look like. These can be used to identify the correct modelling language.
Characteristics of work	Describes the art of work that is captured by the process. Some jobs require agility, whereas other are executed in a routine way.
Characteristics of process	Characteristics of (existing) processes. Are they rigid or flexible, situation-dependent or predefined?
Characteristics of decisions	Describes the decisions along the control flow.
Control flow	A control flow acts as a guidance for the process workers. Do they have to adhere to a predefined sequence of tasks or are they free to decide what to execute?
Intervention at run-time	Is it necessary to adapt the process during run-time or adaption during design-time sufficient?
Objective	The objective of modelling and implementing a process.
Type of process	Classification of a process. The modelled process might resemble a Business process, a Decision or a Case.
Typical application	Example use cases that help to classify the identified process.

Table 4.1.: The indicators' categories and their definition.

Category	Term	Defintion
Documentation style	Directives	Documentation describes exactly what has to be done in which sequence.
	Descriptions of best practices and recommendation	Description of what is best for which situation.
	Spreadsheets	Data that needs to be processed to generate an output.
Preceding process map	Flowchart	A flowchart provides a strict sequencing of tasks with decisions.
	Cluster	Post-it style of tasks that need to be executed with our with sequencing.
	Decision Trees	Decisions, sub-decisions and responsibilities aggregated in a
Characteristics of work	routine	Each iteration of the process must be executed in the same way.
	predictable	The output of a sequence of tasks is known a priori.
	automatable	An execution of tasks can be done without human workers.
	agile	Workers execute tasks and sequence of tasks depending on the situation.
	emerging	A set of tasks can arise during process execution.
	partly automatable	Some parts of the process can be executed by machines, some need human execution.
	decision-intensive	Tasks that incorporate mostly decisions.
Characteristics of process	rigid	The process execution has no room for situation-dependent interaction.
	predefined	Normal task sequencing and exception handling is completely defined before execution.
	workflow-centric	The focus of the process is executing a certain worklfow.
	adaptive	The process can be adapted to changes emerging from situations.
	partly predefined	The execution of some parts can be completely predefined, some are situation-dependent.
	knowledge-centric	The process worker and his knowledge is in the center of the process.
	data-centric	Data processing is in the center of the process.
Characteristics of decisions	Simple (either / or)	Either, or questions.
	Stateful	Decisions require statuses and milestones to be done.
	Complex	Data needs to be processed for decision-making.
Control flow	Definite (required)	Strictly defined sequence of tasks that must adhered to.
	Indefinite (optional)	A lose sequence exists, but mostly workers decide the execution plan.
	Dependencies (required)	To execute a step, sub-steps need to be executed before.
Objective	Automated workflow execution	A workflow should mostly be executed by workflow-systems.
	Support manual work	Human workers should be supported by partly automated processes.
	Automated data processing	Data should be aggregated to an output.

Table 4.2.: Definition of terms forming the indicators.

In the following sections, each standard is going to be introduced in short followed by the corresponding table of indicators.

4.3. BPMN and routine processes

Creating business process diagrams nowadays almost seems to be intuitive. This results from a broad background of routine process modeling incorporating the rise and fall of BPR, ARIS and its EPCs and many more routine process modeling languages. Altogether, they formed the current understanding of business process management and modeling. Since BPMN is a broadly adopted modeling language and a standard that shaped the industry, users might be mislead when to use BPMN correctly.

Business Process Model and Notation incorporates the term *Business Process*, which has been defined and redefined many times by today. For this thesis, we use the OMG's definition of business processes:

> Business Processes: business plans include Courses of Action - what the enterprise has to do to achieve its Ends - transformed into Business Processes that ecnompass activities, sequencing, dependencies, interactions, triggering by business events, etc. [28]

Business processes are ways to achieve goals, to align activities along them and to define interactions, decisions and events. This definition basically incorporates all the core elements of BPMN: Activities, paths, swimlanes (for responsibilities), decisions and events. To recap the notation of BPMN, a small example is shown in figure 4.1. In this model, the application process for a student program is shown, which consists of tasks, for instance *Receive the registration* or *Check deadline*. Tasks can be precised as *Manual tasks*, *Send / Receive Tasks* and have assignees, as shown in *Check deadline*. In current Workflowmanagementsystems, this workflow model can be fully automated and the assignee is presented its tasks when they are instantiated, in example when an application form was received. BPMN models have a starting and an end point which can also be events. In our example, the starting point is a *Message Event*, that indicates the process has started due to a received message.

Another major concern in every control flow oriented modeling language are decisions. BPMN models usually consist of several decision points, depict as diamond-shaped *Gateways*, either empty or with a special symbol inside. Each symbol

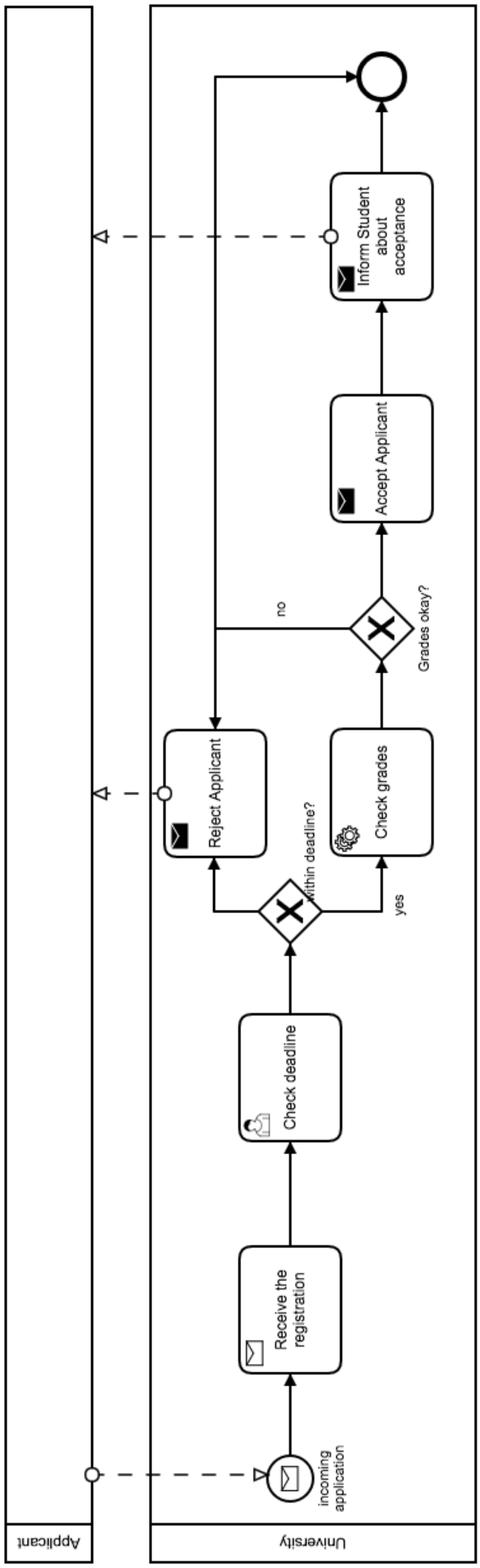

Figure 4.1.: A small BPMN example: the application process for study programs.

stands either for *AND*, *OR*, *XOR* or *COMPLEX*. They direct the control flow depending on the decision logic prescribed. In our example, a *XOR* Gateway guide the control flow into either the direction of acceptance or the opposite one, depending on what date the applicant's form was submitted. Additionally, the communication between the workflow's two participants is notated as so called *Pools*, which can be divided into *Swimlanes*. Each pool is a communication partner and a swimlane indicates different responsibilities, such as departments. If a pool is not filled with BPMN elements, the communication partner is out of scope of the modeler and the processes cannot be modeled. These pools are *Black boxes*.

As we see in the example, the process is structured and completely predefined, thus has no room for creative decisions or even planning at run-time. The only way the process can differ is human intervention or changing the process at design-time. The former is equal to not executing the process in the intended way and the latter is bad for processes in volatile environments. BPMN is suited for a limited variety of processes, foremost business processes and processes and derivatives. The OMG's specification excludes specifically the following [19]:

- Definition of organizational models and resources,
- Modeling of functional breakdowns,
- Data and information models,
- Modeling of strategy,
- Business rules models

Furthermore, the OMG's specification not only provides exclusions such as the ones mentioned above, but it helps to indicate the correct usage of BPMN. Derived from the background chapter 3 and the example above, the following indicators (table 4.3) serve as a guidance for modelers to identify when to use BPMN correctly.

Routine processes occur regularly and their flow is "known a priori" according to [7]. Consequently it is not intended to change the process flow during run-time, as routine are well-structured and predefined.

In our example, the decisions are simple yes or no questions. One of them, *Check grades*, is a business rule task containing a small table of when a grade is acceptable or must be rejected. Overall, the decisions do not require modeling any requirements or personnel apart from the assignee involved in the decision making process. Furthermore, the decisions are always stuck with rules or events that happened before. The process itself is control-flow-oriented and the control flow

Indicators	BPMN
Documentation style	Directives
Preceding process map	Flowchart
Characteristics of work	Routine, predictable, automatable
Characteristics of process	Rigid, predefined, workflow-centric
Characteristics of decisions	Simple (either / or)
Control flow	Definite control flow, required
Intervention at run-time	No
Objective	Automated workflow execution
Type of process	Business process
Typical application	Billing, Accounting, Assembly-line work

Table 4.3.: BPMN indicators.

is required and definite since every candidate needs to be processed in a fair and equal manner. In the end, the process could be automated completely with the appropriate workflow management system and would not need any assignee to check the deadline or the grades.

Overall, BPMN processes are intended to be predictable and to capture routine work. As a matter of fact, a process model's object is either optimizing or automating a process. In terms of BPMN, the automation of routine work is the main objective although it is not always possible to fully automate business processes.

The given example lacks the documentation and the preceding process map. Both are key artifacts to determine the modeling language, because each contains meta information about the process. Since BPMN processes are often predefined and orchestrated, meaning someone in the company fully designs the process without intended deviation, the documentation resembles directives that are also fully specified and provide a guideline how to correctly execute a task. Directives provide a precise description and strict sequence of tasks along with exception handling and interfaces to different processes. BPMN's core elements are tasks, control flow, events and gateways that are intended to capture the directives and make them executable for machines and human interaction.

A typical application is a billing process or assembly-line work that, in a big picture, result in a certain output creating a value for the company. A billing process can involve several departments and persons of responsibilities. However, every iteration of the process is intended to provide a similar, predictable output and a

fully repeatable execution.
It is to say that BPMN and DMN resemble a lot when it comes to indicators. The goal of DMN was to create a modeling technique that incorporates both, decision logic and the declarative illustration of dependencies as well as authorities for a decision (see DMN section). The goal was also to create a data handling modeling technique with the ability to process complex rules and data points. Apparently, these decisions are also routine ones and predictable up to a certain point. The main difference between DMN and BPMN is that DMN is not able to capture workflows and BPMN is not able to capture data processing. As a matter of fact, a combined approach of both modeling techniques might in almost every case suitable due to the complementary aspect of DMN [1].

4.4. CMMN and knowledge work

Routine work occurs in every company, every industry and even in our leisure time. However, not every job is comprised entirely of routine work.
If a person is sick and needs to see the doctor, often the doctor's job consists of routine parts and ad hoc parts. Each patient suffers in a different way, has a private or statutory health insurance, is able to understand the doctor's language or not. In each case, the doctor or the medical personnel needs to decide depending on the situation to identify the patient's disease and medicate him accordingly. Overall, this job requires more knowledge and experience than a strict workflow. Applying the BPMN indicators makes the necessity of a different modeling technique obvious as the indicators do not meet the requirements of this example (adaptable instead of a definite workflow, knowledge-centric process instead of worklflow-centric).
Standardized by the OMG in 2014 [29], the modeling language is intended to meet the requirements of knowledge intensive professions and support workers with models and their execution in workflow management systems. The above described example diverges from the business processes explained in the BPMN section earlier. Instead of focusing on what *has* to be done in which order, the medical personnel should adhere to what *can* be done, as Van der Aalst et al. described in [30]. This is an essential part of the *Case Management* (CM) and *Adaptive Case Management* (ACM) approach. Case management starts with a case such as a legal case, a patient's disease or "some other focal point around which actions are taken to achieve an objective" [29]. As a matter of fact, a case can also be a project that has to be managed or an event that needs to be organized.

Each case contains a set of tasks and files necessary to achieve the goal. These sets are ”minimally predefined” [29] resulting in two phases of working with cases: the design phase and the running phase [29]. During the design phase, tasks and information are aggregated in a manner that is very similar to business process designs. As a result, the work is available in a more structured way as it has been before, yet with the possibility to perform the work completely situation-dependent, if necessary. Case management requires the run-time flexibility such as performing tasks depending on the situation, changing the order of tasks or even collaborate with different people involved as intended. At a first glance, this seems counter-intuitive to the common understanding of modeling processes, which is presumably modeling business processes or similar strictly structured ones. To illustrate the paradigm of case management and introduce the CMMN notation, figure 4.4 provides an example. The subsequent explanation is derived from [29] and [31].

The first element that immediately catches the eye in this example is a large folder depicting the `Case Plan Model`, a root element containing all further elements involved with a name tag in the upper right corner. Next, there is a big structure called a stage and illustrated as a rectangle. Similar to BPMN, stages form a group of elements into a sub process or - to put it in the CMMN context - in a sub case. Stages can be connected via sentries and connectors that are the lines between tasks and the diamond shaped exit or entry criteria. Both, exit and entry criteria trigger when events occur. There are three rules to make sentries trigger:

- `on [event] if [condition]`
- `on [ event]`
- `if [condition]`

According to the CMMN specification, this provides the opportunity to receive events and check whether they have effect on the following process steps [29].
In figure 4.4, a timing event is called *Deadline missed*. This event is connected to an exit criterion, denoted as a black diamond on the border of the Case Plan Model. When the event is triggered, the sentry checks `on [deadline missed] if [date today > deadline date]` and triggers the exit criterion terminating the case due to the missed deadline. A different event type is the *User Event* listening to user interactions that trigger the exit criterion. Besides these two event types, each task, stage and *Case File Item* can trigger events on their own which are received by sentries. *Case File Items* are containers for pieces of information, for instance the patient’s record or in this example the papers, which need to be read

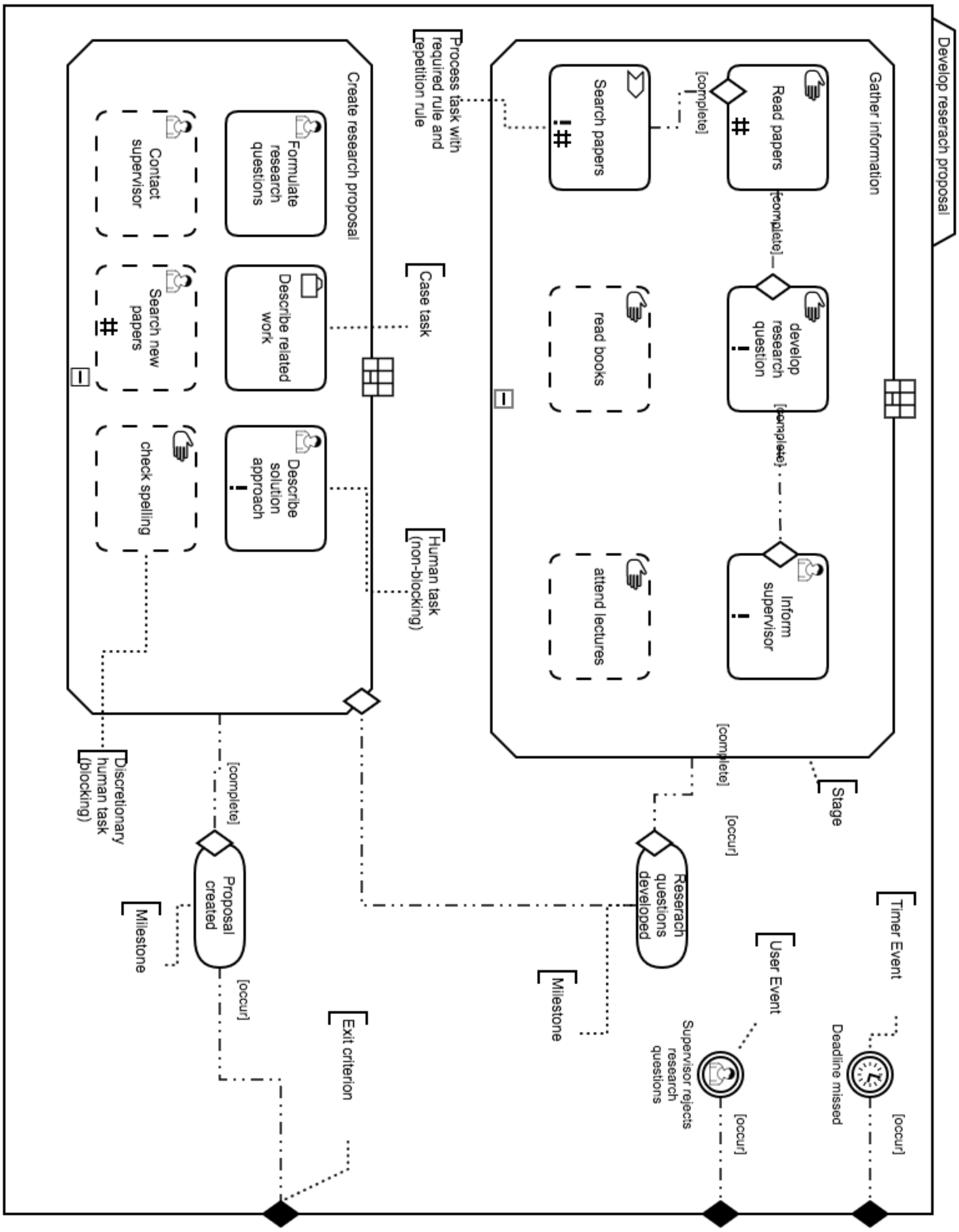

Figure 4.2.: CMMN example: Creating a research proposal.

(not contained in the diagram). They are denoted as .
Another structural element is the *Milestone*, an oval shaped form representing an "achievable target" [29] and providing the ability to measure the progress of a case. Each milestone can have multiple entry criteria or none to show that a milestone is reached or not. In general, "no work is directly associated with a milestone" [29] but rather a "completion of a set of tasks" or "the availability of key deliverables" [29].
Figure 4.4 contains two example milestones: *Research questions developed* and *Proposal created*. Both milestones are connected to either a stage or an exit criterion indicating a dependency. The first milestone needs to be achieved to create the research proposal, the latter one determines the successful termination of the case.
As mentioned earlier, CM requires run-time flexibility and by now, there has nothing been explained but tasks, events and milestones which are per se inflexible. To guarantee run-time planning, tasks and stages have a discretionary version. Discretionary means, a user may decide during run-time if this task or stage will be executed in a following iteration. This mechanism makes it possible to model knowledge and experience combined with routine steps. In our example, the gathered papers might not be enough to provide good citations and emphasize the solution approach. Here, only the case worker can identify if he needs to do more research to find correct citations, as this is a decision requiring knowledge and experience in writing research papers. It is to his or her discretion if the task will be executed or not.
Supplementary to *Discretionary Items*, *Planning Tables* indicate the ability to plan Discretionary Items and apply control mechanisms called *Applicability Rules*. Case workers are presented *Planning Items* if the corresponding Applicability Rule evaluates to true. Overall, this allows the modeler and the case workers to plan at design-time (creating the case model) and at run-time (planning the time discretionary items). The last visible part of the CMMN notation, that has not been explained, is the type of tasks available. In the example case, there are four types of tasks (each type can be indicated in the upper right corner of the task):

- *Human Tasks* either blocking (indicated with a "User" in the upper right corner) or non-blocking with a hand in the right corner
- *Process Task* with a "Chevron" symbol
- *Case Task* with a folder symbol

Both Human Tasks indicate that an assignee with the appropriate role needs to

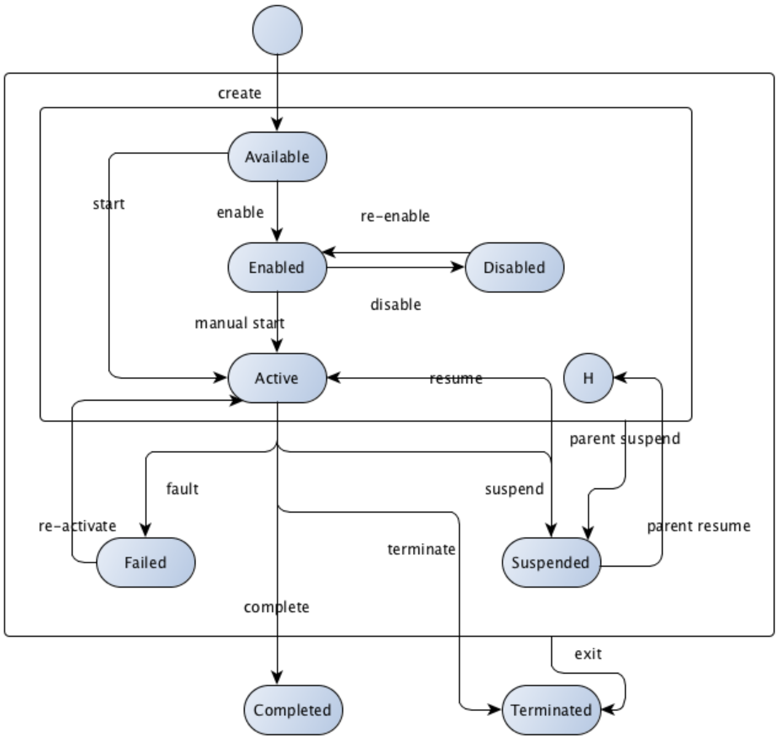

Figure 4.3.: Stage and Task Lifecycle adopted from Figure 7.3 in the CMMN specification [29].

fulfill the modeled work here. *Non-blocking* signifies this step can be done while other tasks of the model run simultaneously in contrast to the blocking one, which does not allow other tasks to be performed at the same time. A *Process Task* calls a business process modeled, for example, in BPMN and a *Case Task* calls a different case handing over input parameters and demanding an output.
In figure 4.4, *Read Papers* is a non-blocking human task, *read books* a non-blocking discretionary human task, *Formulate research questions* a blocking human task, *Describe related work* a case task and *Search papers* a business process task. Apart from the mentioned types of tasks, there a additional execution semantics as seen in the *Search papers* task. An exclamation mark signifies this task as to perform, a sharp that this task has several iterations and a play symbol indicates the *Manual Activation Rule* (not in the example). This rule is connected to entry criteria and transitions from the status `enabled` to *active* if the criterion's condition is fulfilled. CMMN, in contrast to BPMN, offers a complete lifecycle with corresponding states.

Figure 4.3 is a replica of Figure 7.3 in [29] and shows the different states and tran-

Indicators	CMMN
Documentation style	Descriptions of best practices and recommendations
Preceding process map	Cluster
Characteristics of work	Agile, emerging, not automatable
Characteristics of process	Adaptive, partly predefined, knowledge-centric
Characteristics of decisions	Stateful (transition form one to another state)
Control flow	Indefinite control flow, optional
Intervention at run-time	Yes
Objective	Support manual work
Type of process	Case
Typical application	Reviews, Medical attendance, Managing and organising

Table 4.4.: CMMN indicators.

sitions for stages and tasks. These stages, compared to BPMN, allow the modeler and the worker to measure and plan work in a different way than business processes are handled. One of the most commonly mentioned downsides of BPMN is the lack of states and transitions according to [32]. In the example (Fig. 4.4), states are not included since they are not visible in the notation and only occur during run-time. In addition to Fig. 4.3, CMMN offers similar states for sentries and the whole case plan model and can be found in the execution semantics chapter in [29].

Table 4.4 summarizes the notation's characteristics and classifies them as indicators. In the earlier examples we saw that CMMN is suitable for agile and partly predefined work. The worker's knowledge is crucial during execution phase, whereas workflows and sequences are less important. Compared to BPMN, CMMN captures the flexibility of processes that require situation-dependent decisions, agile work whose tasks emerge during run-time and cannot, or at least to a small extent, predicted and thus predefined during design-time. Furthermore, decisions are stateful due to rules and sentries and function as monitoring tools instead of sequencing work. This leads to a flexible control flow, only describing the dependencies of task or milestones that need to be achieved in order to fulfill the whole case.
CMMN documentation and process maps might resemble best practices or descriptions of work, as the content of cases often diverges. However, process discovery should lead to some extent of a description, what can be done to achieve the case and what are the key elements that need to be modeled. Following this, pro-

cess mapping would rather result in an aggregation of tasks that belong together instead of a flowchart indicating steps and sequences.
Overall, BPMN and CMMN differentiate in many points and capture completely different settings. Whereas BPMN has the objective to automate processes, CMMN is intended to support workers by automating certain steps or simplifying the monitoring of case. This makes it a declarative modeling technique according to [35].

4.5. DMN and decision making

In section 4.3, decision-making in BPMN was shown with the aid of a small example, whose excerpt is shown in figure 4.4. Two gateways guide the control flow according to the decisions made. Each gateway compares a value with a given limit and verifies it. If the value is correct, the control flow passes on to the next gateway or, in case the value is out of bounds, rejects the student's application. TThese are simple *either or* questions.
Figure 4.5 demonstrates a more sophisticated version of the same process. To pass the application process, one of the four requirements needs to be fulfilled. Each requirement is implemented as a gateway that checks if the applicant either fulfills it or not. All of the gateways are `OR` gateways signalizing a lose decision making compared to `XOR` gateways. At a first glance, figure 4.5 looks more complicated than figure 4.4. Although the actual decision-making process is kept simple, the BPMN notation with its gateways makes it look confusing and, at a much larger scale, for the reader impossible to comprehend the process as a whole. At this point, DMN becomes relevant.

The DMN notation comprises two parts: the Decision Requirement Graph with one ore more Decision Requirement Diagrams and the decision tables. Figure 4.6 illustrates the corresponding Decision Requirement Graph (DRG) for the decision-making seen in Fig. 4.5.
DMN's notation for DRDs contains only four distinct shapes and three different types of connectors, which can be seen in Fig. 4.6:

- *Decision*: Decisions are depicted as rectangles and incorporate the decision logic which is mapped in decision tables. In the example there are two decisions, *Deadline check* and *Acceptance rate according to Transcript of Record* and both have a table symbol in the upper right corner indicating the decision logic.

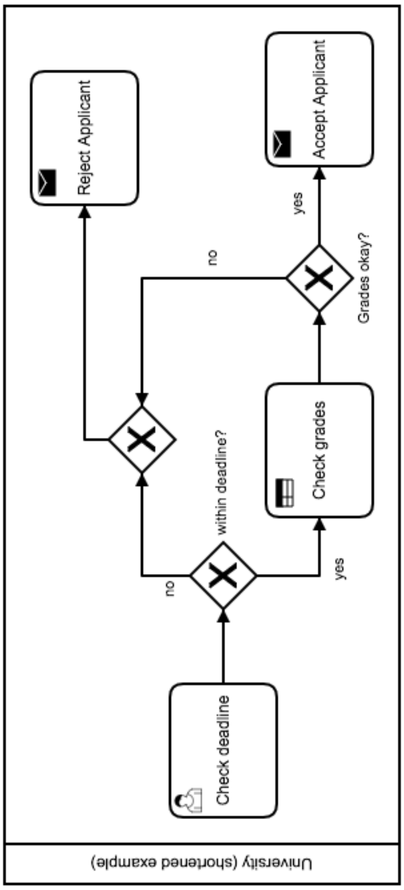

Figure 4.4.: Shortened example from section 4.3 with only two decisions.

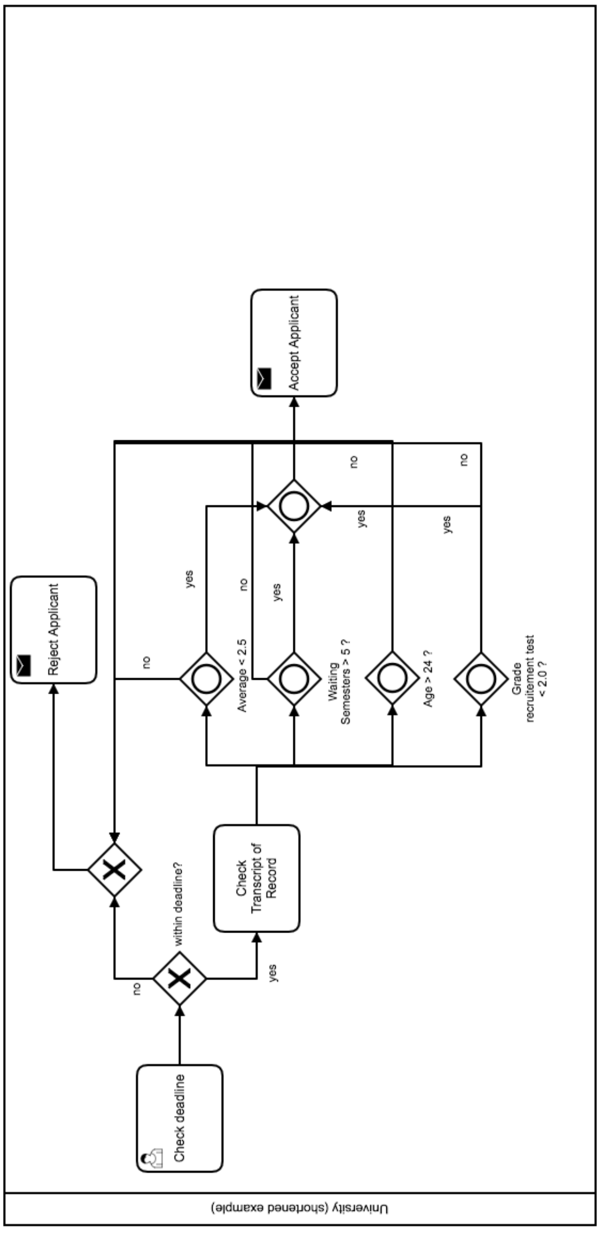

Figure 4.5.: Shortened example from section 4.3 with additional decisions.

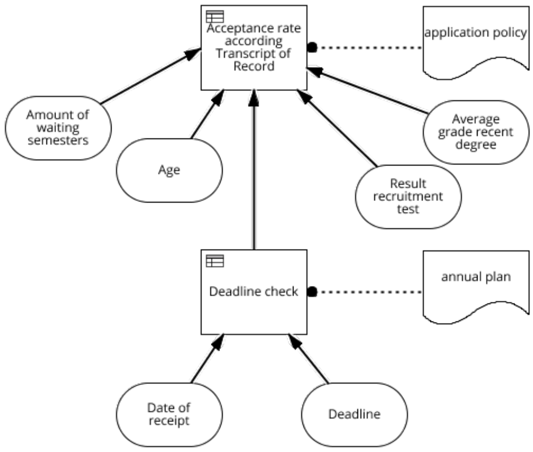

Figure 4.6.: A DMN Decision Requirement Graph incorporating the decision-making of the BPMN process in 4.5.

- *Input Data*: Each decision needs inputs and *input data* is the fundamental and least complex form possible. This element stores information of any type from simple types like dates and integers to lists and functions expressed via Friendly Enough Expression Language (FEEL). An *Input Data* element may only be connected directly to decisions and knowledge sources.

- *Knowledge Source*: DMN makes it explicit where knowledge sources are involved in the decision-making. Typically knowledge sources are documents, best practices but also people or bodies of legislation [1]. "Knowledge sources are the authorities for a decision [...]" [33], which may be connected with input data since managers, for instance, might need background information not directly involved in the decision logic. In the example, the application policy and annual plan of the university is used as *knowledge source*.

- *Business Knowledge Model*: In contrast to Knowledge Sources, Business Knowledge Models are non-authoritative and serve as container for analytic models, decision logic or business rules [1]. It is depicted as a rectangle with cut-off corners in the upper left and lower right. As they only influence decisions directly, the specification allows consequently no connection but with decisions or different business knowledge models. Due to the shortness of the example, Business Knowledge models are not contained in Figure 4.6.

As stated above, the notation comprises only four elements making it a very compact modeling language in terms of the DRG. In addition, there are three different connectors indicating either an *information requirement* (connections from decisions to decisions or input data to decisions), an *authority requirement* (connections from knowledge models to decisions, business knowledge models or different knowledge sources) or *knowledge requirement* (connections from business knowledge models to decisions or different business knowledge models).

The second part of the DMN specification allows the modeler to create decision tables containing the decision logic. An example table [3] is provided in Figure 4.6 implementing the decision logic according to the BPMN example in Figure 4.5. Each table is comprised of several input columns and at least one output column. An input can be a string, integers, floats or basically any type of numbers or enumerations. In this example table, there are five input columns and one output column with annotations such as `Deadline missed`. Each row signifies a rule in the following syntax: `If [INPUT A] AND [INPUT B] then [OUTPUT C]` [1]. The inputs are logically connected by the boolean `AND`-operator resulting in as many outputs as desired, if a rule matches. For every input in the decision table, there is an `input data` element modeled in the related DRG. This also applies for decisions and sub-decisions.
Comparing the table with the BPMN example demonstrates a downside of the decision table: the "`AND` only" syntax. However, modeling `OR` connections is possible with a workaround. Besides the table's data and rules, business analysts can also define *hit policies*. These policies are necessary if some of the rules overlap. In the example table (Fig. 4.6), all of the rules are intended to overlap making a hit policy necessary. There are two major categories of policies: `single hit` and `multiple hit`. The former one allows only to trigger one rule, the latter one allows several rules to trigger. A brief overview with an explanation can be found in table 4.5.

To achieve an `OR` connection of the rules, the hit policy is set to `Multiple Hit: collect (sum)` and the measuring unit of the output is set to percentage. Consequently, the acceptance rate is measured in percentage. The applicant is able to achieve zero or more than 100 percent according to the `collect: sum` rule. Another problem of this method is the inflexibility of the outputs. In this example, an output value of `true` or `false` would have been the best way to model the decision, but neither the decision table nor FEEL allow the aggregation of boolean values.

DMN Tabelle probieren auf Englisch umzustelle

To individualize the decision logic or create more advanced rules, the DMN speci-

[3] Each modeling tool implementing the DMN specification is allowed to vary the orientation of decision tables and the concrete look, such as colors or shades.

C+	Eingabewerte					Ergebnisse	Anmerkungen
	Result recruitment... Zahl	Age Zahl	Amount of waiting ... Zahl	Average grade rec... Zahl	Deadline check Wahrheitswert	Acceptance rate a... Prozentwert (%)	Consequence Informational
1	< 2	-	-	-	-	100%	
2	≤ 2	-	-	-	-	0%	
3	-	> 24	-	-	-	100%	
4	-	≤ 24	-	-	-	0%	
5	-	-	> 5	-	-	100%	
6	-	-	≤ 5	-	-	0%	
7	-	-	-	< 2.4	-	100%	
8	-	-	-	≥ 2.4	-	0%	
9	-	-	-	-	= Ja	0%	Within deadline
10	-	-	-	-	= Nein	-500%	Deadline missed

Figure 4.7.: An example DMN decision table with the decision logic accord to the BPMN example in Fig. 4.5.

Policy	Variations
Single hit	Unique (no overlap allowed)
	Any (overlap allowed, outputs must be equal)
	Priority (returns matching rule with highest output priority)
	First (first hit by rule order)
Multiple hit	Output order (returns all hits in decreasing output priority order)
	Rule order (returns all hits in rule order)
	Collect (aggregates all hits with specified operators: sum, min, max, count)

Table 4.5.: DMN hit policies cited from the DMN specification [1].

fication aggregated several functions and types into an expression language called Friendly Enough Expression Language. It is similar to Structured Query Language (SQL) and Predictive Model Markup Language (PMML) making it easier to understand for non-developers. The expression used in the `deadline check` table is also a predefined function with two inputs (deadline date and date of receipt) and a range of numbers implemented implemented as two rules: `DayDiff(Date of receipt, Deadline)`.
Last of all to mention is the sub-decision `Deadline check`. The decision logic is not implemented in this table, but is referenced by a another decision table (see Fig. 4.6 for illustration). A simple check written in FEEL counts the difference between the application's date of receipt and the given deadline. If the result is less then zero, the deadline is missed and the output of `Deadline check` is `false`. Otherwise, if the result equals zero or more, it evaluates to `true`.

In review, the examples illustrated the multiple uses of the DMN language. Decision tables are useful for automating data intensive decisions and replacing vast BPMN gateway trees. In the same way it is possible to model decisions that require rather knowledge, best practices or analyses than a large amount of data by skipping the decision logic part and only define the dependency graph. On the contrary, it is not possible to any sequence with the DMN notation but the dependency of decisions or the amount of iterations. Although the specification emphasizes the independence of the the notation [1], the DRG on its own might have few useful cases pf application where a standalone DRG is the most convenient solution.

Indicators	DMN
Documentation style	Spreadsheets
Preceding process map	Decision trees
Characteristics of work	Decision-intensive, predictable, automatable
Characteristics of process	Predefined, adaptive, data-centric
Characteristics of decisions	Complex (data processing)
Control flow	Dependencies, required
Intervention at run-time	Yes
Objective	Automated data processing
Type of process	Decisions
Typical application	Calculation of discount rates, salary, choosing assignees

Table 4.6.: Indicators for the identification of DMN use cases.

Table 4.6 summarizes the knowledge acquired by analyzing the DMN specification [1], Debevoise and Taylor [33] and [34], as well as creating the provided examples in this section. It was shown that DMN is an applicable solution for processes that focus on decisions or require decisions, thus are complex to illustrate in BPMN. The most common applications are calculations of salaries, provided in [1] or making a choice by several influencing factors, for instance choosing the right vehicle for hazardous material [33]. Particularly the breakdown into two parts, the Decision Requirement Graph and the decision tables, makes it simple for the modeler to decide between decisions that only need data, therefore use the decision table, model the dependencies of decisions and the decision-makers, ergo utilize the DRG, or combine both.

Table 4.6 also shows a similarity to BPMN: Both modeling approaches aim at the automation of their domains. BPMN's objective is to automate workflows, whereas DMN provides the ability to automate decisions. Both notations capture routine work that is mostly predefined. In addition to that, DMN provides the ability to adapt values of the decision logic during run-time. How this is exactly achieved depends on the implementation by the corresponding software tool. Camunda, for example, provides a Representational State Transfer (REST) API that allows the users to create or delete decisions during run-time.

The main difference between DMN and BPMN, however, is what they are able to capture. DMN is a notation that only is able to process data and outputs a decision that can be handled by other modeling languages, if desired. BPMN is also able

to handle decisions, but not to process data in the way DMN does. This is emphasized by the control flow of the DMN language: decisions can enhanced with dependencies and knowledge sources. Overall, DMN is a notation that only serves for decisions and does not suit for any different use case.
Data that needs to be processed to find a decision is often stored in spreadsheets instead of free text documents. This is a strong indicator for DMN, since the spreadsheets can be copied in DMN's decision logic. Furthermore, decisions that have sub-decisions or dependencies might be clustered in a decision tree during process discovery. The DRD is able to incorporate these decision trees and make them executable.
In Chapter 5, the combination of DMN and BPMN is described along with use cases where DMN suits as a replacement for gateways and a functions as a data processor or event-handler. This chapter describes the complementary character of the DMN notation as well.

4.6. Summary

In this chapter, each specification was presented with examples and their characteristics. By analyzing the OMG's standards and further research publications, these characteristics could be transformed into indicators helping modelers to identify when to use the corresponding technique. Additionally, the examples explained the fundamental principles of the notations and how to use them correctly. The main focus, however, was developing the indicators. For further information about the notations and more sophisticated examples or instructions, please refer to the corresponding OMG specification or the provided sources in the references chapter. Additionally, a recommendation of how to combine manual process discovery techniques was presented along with an explanation of each approach. Besides the manual discovery, the automated one was explained and a recommendation for a corresponding tool was stated. Concluding the indicator sections, BPMN is useful for predefined processes, mainly mapping routine work without the possibility for run-time intervention; CMMN is suitable for people working in volatile environments with routine parts but mainly work depending on their personal experience and knowledge; and DMN is a valuable way to encapsulate complex, data-intensive decision-making in a separate, compact notation that allows runtime adjusting. For all of the mentioned notations, there are overlaps that make it possible to combine them and creating a whole model with separate notations for each domains. This will be discussed more detailed in the following chapter.

Todo list

Part V.

Combination

5. Combining the different standards

In the previous chapter, indicators were presented to distinguish between the standards and to identify when to use the corresponding one for the given use case. Some, however, require a combined use of the presented models to achieve run time flexibility in BPMN models or to represent routine parts in a flexible CMMN model. An explanation, when to combine models and how to combine them is presented in this chapter.

5.1. When to combine

The indicators in Chapter 4 provide a set of characteristics to differentiate between the three named specifications: DMN, CMMN and BPMN. This is useful for rather small processes with exactly defined purposes and without any exception handling or interfaces to external processes. However, processes might become larger and thus harder to understand. In addition, process models can become biased by human error and cause performance issues, apart from the inability to model certain scenarios on their own. Besides the performance task, small processes are often part of a larger process landscape the need to be integrated somehow. Consequently, there need to be interfaces for inter-procedural communication and modeling tools that support inter-procedural modeling.

When to combine different modeling approaches is a question, that needs to be answered individually for every use case. The indicators in Chapter 4 provide a set characteristics supporting the decision, when to use the corresponding modeling approach correctly. At this point, the so called *Separation of concerns* [34] comes into play, meaning each modeling language has a domain, where it suits best. DMN is best for decisions, BPMN for routine and CMMN for unpredictable work. Separating the concerns means, in transferred sense, to divide the discovered information into parts and domains that can be modeled with the correct modeling language.
Process discovery aims, as explained in a preceding chapter, to identify individual processes and their tasks. These processes form small, definite units, similar to

decoupling in software engineering. Each process should represent a logical unit that can be inter- and exchange in a larger process landscape. Forming such a landscape consequently leads to a high degree of decoupling and functional cohesion. As a matter of fact, this is the realization of *separating the concerns*.
To achieve a high degree of decoupling and a functional cohesion, the process discovery first needs to define small processes encapsulating one logical unit. Second, the indicators should be applied to identify the correct modeling language. Finally, the composition of a larger process model can be done by integration the small units with interface tasks. This last step is discussed in the subsequent sections.

5.2. Combining CMMN with BPMN

To illustrate the benefits of a BPMN and CMMN combination, we construct a real world scenario of a bank that with a customer who wants to open a new account. One of the very first steps is to verify the customer's identity. Of course, this process can be done in a branch office where an employee checks the customer's identity card. However, some banks rely on online procedures with photos and codes to fasten up this procedure. Apparently, the greatest part of this procedure should be repeatable and follow a strict routine to validate the identity. Consequently, the main process should be modeled in BPMN. However, each process is error-prone leading to exceptions. In our example, call center agents are supposed to handle these exceptions. At a first glance, call center agents work partly routine and partly depending on the circumstances. A look at the indicators identifies CMMN as the right modeling language for this purpose. Finally, these exceptions can be modeled in CMMN and linked to the BPMN model.
This short example shows a possible combination of two models. The online process is designed and modeled in BPMN due to the vast amount of routine steps. The agent's call is more of a knowledge task he has to do. It comprises routine parts, but discretionary items such as answering questions or fixing problems posed by the customer. Consequently, a combination of CMMN and BPMN suits best for this use case.

There are two ways of combination: integrating CMMN in a BPMN model and the opposite way.
Figure 5.1 illustrates a simple Case Plan Model containing two tasks: `A simple process task` and `Another simple process task`. Both "[...] can be used in the Case to call a Business Process" [29]. As the CMMN specification stated, this is a *call activity* with the intention to direct the process into another (sub-)

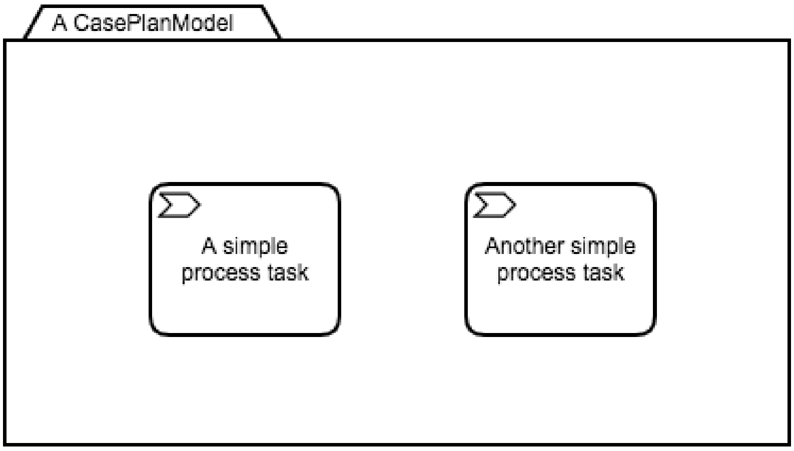

Figure 5.1.: A CMMN process task to call a BPMN process.

process. A brief look at the meta model proves this hypothesis by the following attributes of the CMMN process task: `implementationType : URI`[1], `inputs : ProcessParameter` and `outputs : ProcessParameter` [29]. The `implementation` attribute indicates the link to other processes, namely BPMN, XML Process Definition Language (XPDL) version 2.x, Web Services Business Process Execution Language (WSBPEL) 2.0 and WSBPEL 1.0. The called processes are handed over zero or more input parameters and return the results as output parameters, subsequently processed by the main process.
How this link between both models is visually presented depends on the tool the analyst uses. Camunda, for example, uses attributes to link the processes. Signavio links the models directly, so the user only has to click the process symbol in the upper right corner of a process task to switch the models. Though, it is not possible to explicit model BPMN in the CMMN notation, which holds for all notations: BPMN, CMMN and DMN. It is possible to link them, but no to create BPMN models with e.g. the notation set of CMMN. As CMMN is a fairly new specification, the way how combined models can be presented to the user might vary until the tools support the standard completely.

BPMN does not provide any designated interface that links to Case Management or CMMN in particular+, but a workaround is possible similar to the link CMMN to BPMN.
In Fig. 5.2 a very small business process is shown, modeled in BPMN. It com-

[1]Read `<attribute> : <type>`

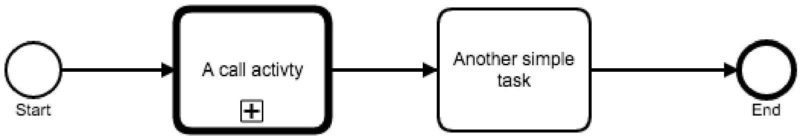

Figure 5.2.: A BPMN model with a call activity and an unspecified task.

prises a `call activity` and a very simple `task`. Here, the call activity is able to integrate a CMMN model. According to the BPMN specification, "a call activity identifies a point in the process, where a global process or a global task is used" [19]. Again, the meta model proves the ability to link different processes. A Call activity has an attribute `calledElement: CallableElement` that instantiates an object of the super class `CallableElement`. This super class provides an `ioSpecification` and an `ioBinding` attribute as well as a specification of the used interface. Eventually, the `ioBinding` allows to define the inputs and outputs as well as one or more operations on the data sets. The `ioSpecification` is used to specify the inputs and outputs of the activity, whereas the `ioBinding` becomes necessary when the Call Activity is linked to a service instead of a process [19]. Consequently, a globally defined[2] CMMN case (which can also referred to as a process) can be called here. As the call activity is not a native CMMN-link, the `CallActivityType` needs to be set to CMMN. Additionally, each call activity has to deliver an output value that is generated during the execution [19]. When invoking a process it is necessary to keep in mind that a call activity is able to override the attributes of a called process. Consequently, the behavior of the called process might change [19].

In BPMN, a token traverses the model. Each gateway either directs the token to another direction or even multiplies it (AND gateway). At each endpoint of the process model, only one token arrives and ends the instance of the process. By inserting a call activity in BPMN, the token is blocked until the called process ends and escalates the result to the calling process. Consequently, the called process could call another one and this one could also call a process leading to a huge capacity usage and consequently to an abortion of the instance. To avoid this situation, BPMN provides the `Termination Event` that, when reached by a token, terminates the process immediately and discards the other tokens.

[2]Globally defined means the called object is not a sub-object that can only be instantiated with the main object. Globally defined objects can be called and instatiated.

5.3. Combining DMN with BPMN

The "DMN notation is designed to be usable alongside the standard BPMN business process notation" states the first paragraph in the DMN specification [1]. Consequently, DMN serves as a complementary language to BPMN and helps to "[...] create lighter, more focused processes by moving decision details into DMN" [33]. This leads to a clear structure without a vast amount of gateways and faster processing. Generally speaking, the standalone version of DMN is useful for modeling decisions and their requirements with the DRG. The benefits of DMN can be used to a full extent, however, by combining it with another modeling language, for instance BPMN.
BPMN has one designated interface for DMN decision logic and several more or less well-suited ones. Table 5.1 shows BPMN's interface tasks explained by the DMN specification. It stands out that the Business Rule Task is the designated interface, since the BPMN specification defines it as a "[...] mechanism for the Process to provide input to a Business Rules Engine and to get the output of calculations that the Business Rules Engine might provide" [19]. As a matter of fact, this is the main purpose of the DMN notation.
Figure 5.3 illustrates the combination of a BPMN model and the DMN decision logic and how it is implemented by the signavio modeler. The Business Rule Task has an attribute that links to the DMN model and calls the decision with the input parameters. According to the decision logic, the parameter is matched with the correct rule leading to the desired output. In this example, the salary of an employee along with additional privileges needs to be calculated. Obviously, this seems to be a predefined and fully repeatable task combined with a data-centric decision. Although it is possible to model both characteristics in BPMN, it is better to separate them according to their concerns. This leads to simpler models and better performance.
The example company employs only three different roles: executives, knowledge workers and routine workers. Each position is paid accordingly and gets an additional privilege. The executive, for instance, gets a salary of 3000 EUR. coupled with a company vehicle. In BPMN, the process sends the parameter to the decision engine and returns the right salary, which is paid after the calculation. The payment procedure is denoted as a script task incorporating the automated payment. According to table 5.1, a script task is used for automated decision execution, which can be verified by the example.

Debevoise and Taylor identified common patterns for the combination of DMN and BPMN in [33]:

- *Task Sequencing (inclusive)*: A DMN decision can guide the control flow into one or more subsequent, parallel executed tasks connected by an `AND`-gateway (also called *inclusive* gateways).

- *Participant Assignment*: Responsibilities are usually illustrated by pools and swim lanes. The DMN decision logic can identify the correct swim lane and guide the control flow with an `XOR`-gateway there.

- *Effect or Gateway Sequencing (exclusive)*: The same as *Task sequencing* but with `XOR`-gateways.

- *Data Information*: DMN is data-centric notation and suits well for data validation and data warehousing. Meta information such as age, origin and accessibility can be managed by DMN tables and stored in the correct database or presented the right person.

- *Detection of Events*: So called *detection decisions* [33] react to internal or external events. External events are defined as a weather change, security events, or can be issued by customers as well as contractual partners. Internal ones arise from process activities and related steps. An event might eventually make a process adapt to changing circumstances in order to create the same value. Detecting decisions represent the switch that regulate the process accordingly and make it react the occurring events.

So far, only the integration of DMN into a BPMN process model was described. Apparently, integration works seamlessly and complements the BPMN notation. Next, integrating a BPMN model into DMN will be examined.
Recapturing the examination of the DMN notation in Chapter 4 recalls the four distinct elements:

- the Decision encapsulating the decision logic

- the Business Knowledge Model which is basically a decision that can be reused and parameterized [33]

- the Knowledge Source representing the authoritative foundation

- and the Input Data elements that are necessary to make the decision.

The only possible interface to integrate another modeling language in DMN could be the Knowledge Source element, as the Business Knowledge Model also encapsulates decision logic. According to the DMN meta model, Knowledge Source elements inherit from `DRGElement` (similar to an interface class in java) and from `NamedElement` "[...] from which it inherits [the attributes `name`, optional `id`, `description` and `label`" [1]. Additionally, the element has three distinct attributes: `type` of type string, `owner` of type `OrganisationalUnit` and the `locationURI` of type `URI`. The `type` attribute is intended to identify the "kind of authoritative source" such as "policies, regulations or analytic insights" [1]. The Uniform Resource Identifier (URI) specifies the concrete location where the authoritative source can be found.
In BPMN and CMMN, the interface tasks have designated attributes to call or reference processes. In CMMN, a process task has the attribute `ProcessRef` of type `Process` specifically to link different processes [29]. The BPMN Business Rule Task provides an attribute called `implementation` that states the technology used to perform the task [19]. The DMN meta model, however, lacks these interface tasks and, in particular, the necessary attribute to reference a different process. The Knowledge Source element's `URI` could link to a process, but it still lacks the `implementation` attribute that specifies the process type. The `URI`'s intention is only to provide a specific description where the authoritative documentation can be found.

In conclusion, integrating different models in DMN does not work yet. The current version at the time this thesis was developed was DMN 1.0 from May 2016. As seen in BPMN, it is possible to extend the set of elements of a notation and to make interfaces available. Comprising only four elements, the DMN notation is more a complementing instead of a standalone model as shown in this section. Separating the decision logic in DMN and including them in a BPMN model makes the models easier to model, better to understand and eventually perform faster including run-time adjustment of the decision logic.

5.4. Combining CMMN with DMN

In the previous section, we examined the compatibility with DMN and BPMN and came to the conclusion, that BPMN is indeed able to integrate different notations. The DMN notation, on the other hand, has no ability to integrate other models, but to provide a reference by an Uniform Resource Identifier (URI). The URI helps to identify and locate the documents or processes referenced by the Knowledge

	Inputs	Outputs	Annotations
U	Position {routine worker, knowledge worker, executive}	Salary Currency (€)	Additional privileges Informational
1	= [routine worker]	€ 1,800	voucher for a massage
2	= [knowledge wo...	€ 2,500	access to a library of his choice
3	= [executive]	€ 3,000	company vehicle

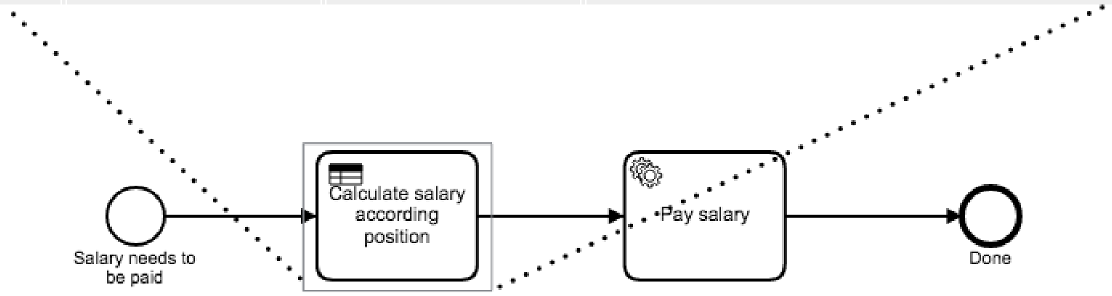

Figure 5.3.: The DMN decision logic implemented as a BPMN business rule (illustrating example).

Source element. Thus, this is not an integration but a documentation where to find the right document to acquire knowledge about the decision-making.
In the preceding section about the integration of CMMN and BPMN we found out, that the integration of Business Processes, particularly BPMN processes, is possible in CMMN. Business processes, in particular BPMN or WSBPEL, are imperative modeling languages. They focus on two concepts according to [35]: "The continuous changes of the process' objects" and the "[...] distinct actions, events, and changes of the process and how these can possibly succeed each other." A DMN decision can behave similar, but additionally has declarative parts. This leads to the question, whether it, is on the one hand, possible to integrate DMN in CMMN and, on the other hand, if it is generally useful.

The compatibility is the first and foremost question that needs to be answered. CMMN offers two ways to link external models with the process: Case Tasks and Process Tasks. A Case Task can only establish a link to another Case. A process task can link to business processes. DMN is per se not defined as a business process modeling language. It serves as complementary modeling language to map decisions and the according requirements. In general, this works more in a declarative way than in a imperative. Thus, the CMMN standard does not intend to link DMN with CMMN.
This holds for the current version of the standard. At the time this thesis was in development, the latest version available was CMMN v1.0. However, the OMG published a beta version of the upcoming CMMN specification version 1.1 [36]. In this version, a dedicated interface to DMN models is added (see Figure 5.4). A `Decision Task` with a `decisionRef : Decision` attribute referencing the DMN model establishes the missing link. The attribute `implementationType : URI` specifically states the compatibility with the DMN notation. Similar to the Business Rule Task in BPMN, there are input and output parameters of type `DecisionParameter` and another parameter, `externalRef : QName`, that defines the exact decision to be used [36]. In general, the OMG intends to support automated and data-centric decision making in CMMN. At this point in time, it has not been possible yet.

Figure 5.4 shows a simple case with a discretionary decision task modeled. The scenario deals with a student who has to write his bachelor's thesis (see Human Task) and is monitored by his supervisor. The decision task incorporates the decision logic and links to a DMN table. When the case worker, presumably the

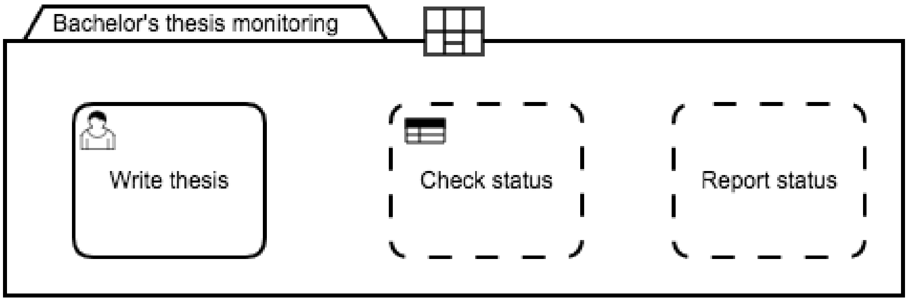

Figure 5.4.: CMMN case with a decision task linking to a DMN decision table.

student, needs to report the thesis' status to his supervisor, he checks the decision table that aggregates the information and outputs the status. In this example, the worker simply enters the amount of written pages and a comment, for instance *on track* or *far behind*, is presented. The student sends this result afterwards to his supervisor.
This brief example use case demonstrated an automated decision making in an environment where the focus lies on human behavior and human reacting. As DMN serves not only as a `decision-making` notation, but also as a possibility to process data and generate an output in a fast and flexible way, it helps knowledge workers to react to data-intensive tasks and to plan following discretionary steps.

Decision Role	Activity Type Role
Apart from looping or multiple instantiating tasks, loops have no decision role. They can help to iterate through decisional steps.	
A service task executes decisison in an automated way. The corresponding decision model, however, does not guarantee the automated execution.	
The assignee has to execute the decision manually.	
The Business Rule Task is defined as a placeholder according to [19] for decisional services. This is the predefined inteface for DMN models.	
Decisions can be encoded by process scrip languages.	

Table 5.1.: BPMN elements and their DMN compatibility, adopted from Table 70 in [1].

Part VI.

Use Case Study

6. Case Case Study

Introdction

Todo list

6.1. Methodology

6.2. CMMN

6.3. DMN

6.4. Combined approach

Appendix

Bibliography

[1] The Object Management Group. *Decision Model and Notiation*. Object Management Group, version 1.1 edition, May 2016.

[2] Micheal Porter Michael E. Porter. *Competitive Advantage: Creating and Sustaining Superior Performance*. FREE PR, 1998.

[3] H. R. Motahari-Nezhad and K. D. Swenson. Adaptive Case Management: Overview and Research Challenges. In *2013 IEEE 15th Conference on Business Informatics*, pages 264–269, July 2013.

[4] Łukasz Osuszek and Stanisław Stanek. Knowledge Management and Decision Support in Adaptive Case Management Platforms. In *Proceedings of the 2015 Federated Conference on Computer Science and Information Systems*. Polish Information Processing Society PTI, October 2015.

[5] Vadim Kuzin and Galina Kuzina. *On the Move to Meaningful Internet Systems: OTM 2013 Workshops: Confederated International Workshops: OTM Academy, OTM Industry Case Studies Program, ACM, EI2N, ISDE, META4eS, ORM, SeDeS, SINCOM, SMS, and SOMOCO 2013, Graz, Austria, September 9 - 13, 2013, Proceedings*, chapter CMMN Implementation in Executable Model of Business Process at Order-Based Manufacturing Enterprise, pages 112–123. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[6] Shaowei Wang and Mamadou Kaba Traoré. A System-theoretic Approach to Case Management. In *Proceedings of the 2014 SpringSim Poster Session*, Posters '14, pages 31–32, San Diego, CA, USA, 2014. Society for Computer Simulation International.

[7] Michael Zeising, Stefan Schönig, and Stefan Jablonski. Towards a Common Platform for the Support of Routine and Agile Business Processes. In *Proceedings of the 10th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pages 94–103. Institute for Computer

Sciences, Social Informatics and Telecommunications Engineering (ICST), October 2014.

[8] Matthias Kurz, Werner Schmidt, Albert Fleischmann, and Matthias Lederer. Leveraging CMMN for ACM. In *Proceedings of the 7th International Conference on Subject-Oriented Business Process Management - S-BPM ONE '15*. Association for Computing Machinery (ACM), 2015.

[9] Thierry Biard, Alexandre Mauff, Michel Bigand, and Jean-Pierre Bourey. *Risks and Resilience of Collaborative Networks: 16th IFIP WG 5.5 Working Conference on Virtual Enterprises, PRO-VE 2015, Albi, France,, October 5-7, 2015, Proceedings*, chapter Separation of Decision Modeling from Business Process Modeling Using New "Decision Model and Notation" (DMN) for Automating Operational Decision-Making, pages 489–496. Springer International Publishing, Cham, 2015.

[10] Kimon Batoulis, Andreas Meyer, Ekaterina Bazhenova, Gero Decker, and Mathias Weske. Extracting Decision Logic from Process Models. In *Advanced Information Systems Engineering*, pages 349–366. Springer Science Business Media, 2015.

[11] Steven Mertens, Frederik Gailly, and Geert Poels. Enhancing Declarative Process Models with DMN Decision Logic. In *Enterprise, Business-Process and Information Systems Modeling*, pages 151–165. Springer Science Business Media, 2015.

[12] Marlon Dumas, Marcello La Rosa, Jan Mendling, and Hajo Reijers. *Fundamentals of Business Process Management*. Springer Science & Business Media, Berlin Heidelberg, 2013.

[13] Marc Lankhorst. *Enterprise Architecture at Work*. Springer Science Business Media, 2009.

[14] T. Allweyer. *BPMN 2.0: Introduction to the Standard for Business Process Modeling*. Books on Demand, 2010.

[15] V. Repa and O. Zelenzfik. Methodological limitations of modeling languages BPMN and ARIS. In *Proc. 15th Int Control Conf. (ICCC) Carpathian*, pages 507–512, May 2014.

[16] Jan C. Recker, Marta Indulska, Michael Rosemann, and Peter Green. How Good is BPMN Really? Insights from Theory and Practice. In Jan Ljungberg and Magnus Andersson, editors, *14th European Conference on Information Systems*, Goeteborg, Sweden, 2006.

[17] W.M.P. van der Aalst, M. Stoffele, and J.W.F. Wamelink. Case handling in construction. *Automation in Construction*, 12(3):303–320, may 2003.

[18] Stijn Goedertier, Jan Vanthienen, and Filip Caron. Declarative business process modelling: principles and modelling languages. *Enterprise Information Systems*, 9(2):161–185, sep 2013.

[19] The Object Management Group. Business Process Model and Notation. Technical report, The Object Management Group, 2013. Version 2.0.2.

[20] Alan N. Fish. *Knowledge Automation: How to Implement Decision Management in Business Processes*. JOHN WILEY & SONS INC, 2012.

[21] J. Mendling, H.A. Reijers, and W.M.P. van der Aalst. Seven process modeling guidelines (7PMG). *Information and Software Technology*, 52(2):127–136, feb 2010.

[22] Laury Verner. The challenge of process discovery. *BPM Trends, May*, 2004.

[23] Sandeep Jadhav. Business Process Discovery. *BP Trends (February)*, 2011.

[24] Wil van der Aalst. *Process Mining - Discovery, Conformance and Enhancement of Business Processes*. Springer Science & Business Media, Berlin Heidelberg, 2011. aufl. edition, 2011.

[25] Robert Klein & Armin Scholl Wolfgang Domschke, Andreas Drexl. *EinfÃ¼hrung in Operations Research*. Springer Berlin Heidelberg, 2015.

[26] Marco Weimer. Integration adaptiver prozess-management-technologie und process mining, 2006.

[27] Wil Van Der Aalst, Arya Adriansyah, Ana Karla Alves De Medeiros, Franco Arcieri, Thomas Baier, Tobias Blickle, Jagadeesh Chandra Bose, Peter van den Brand, Ronald Brandtjen, Joos Buijs, et al. Process mining manifesto. In *International Conference on Business Process Management*, pages 169–194. Springer, 2011.

[28] The Object Management Group. Business Motivation Model. Technical report, The Object Management Group, 2015.

[29] The Object Management Group. *Case Management Model and Notation version 1.0*. Object Management Group, version 1.0 edition, May 2014.

[30] Wil M.P. van der Aalst, Mathias Weske, and Dolf GrÃ¼nbauer. Case handling: a new paradigm for business process support. *Data & Knowledge Engineering*, 53(2):129–162, may 2005.

[31] Knut Hinkelmann. Case Management Model and Notation - CMMN, 2015.

[32] Jan Recker. Opportunities and constraints: the current struggle with BPMN. *Business Process Mgmt Journal*, 16(1):181–201, feb 2010.

[33] Tom Debevoise, James Taylor, Jim Sinur, and Rick Geneva. *The MicroGuide to Process and Decision Modeling in BPMN/DMN: Building More Effective Processes by Integrating Process Modeling with Decision Modeling*. CreateSpace Independent Publishing Platform, USA, 2014.

[34] Thierry Biard, Alexandre Le Mauff, Michel Bigand, and Jean-Pierre Bourey. Separation of Decision Modeling from Business Process Modeling Using New "Decision Model and Notation" (DMN) for Automating Operational Decision-Making. In *Risks and Resilience of Collaborative Networks*, pages 489–496. Springer Science Business Media, 2015.

[35] Dirk Fahland, Daniel Lübke, Jan Mendling, Hajo Reijers, Barbara Weber, Matthias Weidlich, and Stefan Zugal. Declarative versus imperative process modeling languages: The issue of understandability. In *Enterprise, Business-Process and Information Systems Modeling*, pages 353–366. Springer, 2009.

[36] The Object Management Group. *Case Management Model and Notation version 1.1 (beta)*. Object Management Group, version 1.1 (beta) edition, May 2015.