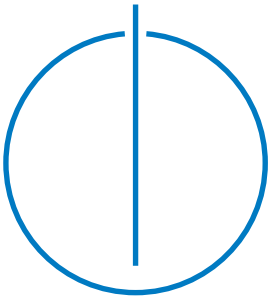
Department of Informatics

Technical University Munich

Master's Thesis in Informatics

Discovering Clinical Pathways of an Adaptive Integrated Care Environment

Simon Bönisch



Department of Informatics

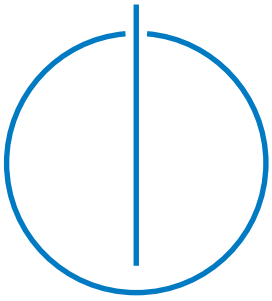
Technical University Munich

Master's Thesis in Informatics

Discovering Clinical Pathways of an Adaptive Integrated Care Environment

Erkennen Klinischer Behandlungspfade einer Adaptiven Integrierten Pflegeumgebung

Author:	Simon Bönisch
Supervisor:	Prof. Dr. Florian Matthes
Advisor:	M.Sc. Felix Michel
Submission Date:	June 15th, 2019



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

June 15th, 2019 Simon Bönisch

Acknowledgments

I would like to thank my advisor, Felix Michel, for his support and guidance during this work and for the many circumstances he took on himself to make it possible. I would also like to thank him and my supervisor, Prof. Florian Matthes, for giving me the chance to work on this topic and allow me to discover my affinity for the area of process mining, and for their support throughout.

I would like to extend my gratitude to all my fellow students, colleagues, flatmates, friends and family who accompanied me and supported me during this time. A special thank you is due to my parents who supported me financially and morally through all these years, to my dear friend Jonas La Roche who endured many nerve-racking evenings filled with lengthy conversations and to my friend Alina Götz who never hesitated to encourage me and spent countless days with me in the library. This would not be here without you.

Abstract

The overall demographic trend of an increasingly aging population leads to more complex chronic patients resulting in rising healthcare costs. Integrated care systems can reduce these costs and increase patients' quality of life by replacing part of the necessary hospital visits with home hospitalization and automated monitoring. However, classical hospital information systems are not flexible enough to support these types of uncertain, human-driven treatment processes, also called clinical pathways. Therefore, adaptive case management systems have been developed. They are highly configurable, offer additional runtime flexibility and are currently being advanced in multiple large research projects. One such project is called Personalised Connected Care for Complex Chronic Patients or CONNECARE and is currently being tested at three sites across Europe in two clinical case studies each.

The goal of this work is to evaluate the usage patterns observed during the course of the CONNECARE project in order to assert the degree of executional flexibility that was employed while treating the patients. To achieve this, the web access logs of the underlying Smart Adaptive Case Management (SACM) component, CONNECARE's case execution engine, will be analyzed using process mining techniques. Afterward, the generated visual process maps for the different case studies will be compared and evaluated regarding their contained execution flexibility. The expected high complexity of the maps is mitigated by means of manual data clustering, the provision of models at different abstraction levels, the use of the fuzzy mining algorithm and the employment of interactive visualization techniques.

Results show that while professionals in both studies and at all sites made use of the provided flexibility measures, the degree of flexibility that was used depends on the exact site. However, the degree of employed flexibility correlates with the overall system activity during each study. Additional features for collaboration and communication that are provided by the SACM component were used at a varying degree, also depending on the site. One site made extensive use of the user and role modeling capabilities but all sites showed an accumulation of performed work for individual users. At the site with the highest system activity, clinicians were shown to react to alarms based on automated measurement threshold violations by contacting colleagues or the affected patient. As the case studies are still being executed at the time of writing this work, the evaluation was designed to be re-executable once the study period has passed.

Keywords: Integrated Care, Clinical Pathway, Adaptive Case Management, ACM, Process Mining, Process Discovery, REST, Healthcare, CONNECARE

Contents

Part I.

Introduction and Background Theory

1. Introduction

In this chapter, an overview of the research motivation is given, followed by some background information on concepts essential for understanding the research goals. Afterward, the concrete questions to investigate are stated and explained in detail before the thesis outline is presented.

1.1. Motivation

The overall demographic trend of an increasingly aging population leads to more complex chronic patients (CCPs) resulting in rising costs in many areas of the healthcare sector [115, 146, 174]. This, in turn, creates incentives to optimize the processes in the healthcare sector and to increase the delivered service quality while reducing resource usage and thereby minimizing costs. For this reason, Clinical Pathways (CPs) were developed as a way for hospitals to streamline their processes, optimize their resource utilization and reduce costs while also incorporating principles of evidence-based medicine to increase patient outcomes and satisfaction [108]. CPs are an abstract representation of an idealized treatment process that a patient with a certain condition should follow. In practice, it is used as a guideline for medical professionals and can be adapted to the individual situation [26]. While CPs provide significant advantages in the treatment of acute conditions, special challenges but also opportunities arise when trying to apply them for the treatment of CCPs [29]. For example, as CCPs have chronic conditions, they are often monitored by primary care physicians or unpaid relatives. Integrating with those carers opens up the possibility of preventive monitoring to avoid a patient's hospital visit in the first place, freeing resources and saving costs [53]. On the other hand, treating CCPs is often more complex, as the chance for the appearance of unforeseen conditions or comorbidities is higher than for usual patients, possibly requiring ad-hoc treatment modifications. This also poses special challenges to the supporting IT systems [91, 109].

A project with the aim of meeting those challenges is *Personalised Connected Care for Complex Chronic Patients (CONNECARE)*[1]. It integrates primary and clinical care with medical devices for home use and offers CCPs the chance to self-manage their condition, reducing the number of hospital visits and increasing their quality of life [167]. For medical professionals, the system also provides advantages like treatment guidance, interdisciplinary communication or automated measurement validations and alerts. It is

[1] see http://www.connecare.eu/

designed as a multi-tenant system that can be adapted to the individual deployment locations and treatments that it should support. The EU-funded project is being evaluated since July 2018 through two clinical case studies each performed in hospitals in three different countries [40, 167]. CONNECARE is powered by the Smart Adaptive Case Management (SACM), a special kind of process engine that can be used for executing knowledge-intensive processes (KiPs) like they are found in hospitals [104, 105]. SACM offers additional features like patient-professional communication, notifications and a role-based authorization system. It can be invoked through a RESTful API that features full request logging. The SACM needs to be configured by supplying a case model which is a CP that has been transformed so the system can understand and interpret it.

As KiPs are inherently complex, modeling them in order to create a CP is difficult as well [111]. This makes it necessary to evaluate the developed CPs in practice to ensure they actually provide improvements before employing them on a larger scale. However, this is a hard task in itself due to the nature of the KiP that was modeled. Often, the only way to verify the developed models and the supporting IT system is to use it in a productive setting like e.g. a case study.

Doing so is the goal of this thesis which is also shown in figure 1.1. Based on the API logs that were generated so far during the execution of the developed case models, process discovery techniques are applied in order to find a process model that represents the actually performed activities. Process discovery is a Process Mining (PM) task that tries to construct a process model based on an event log thus capturing the behavior seen in the log [153, 154]. The process models are then compared to the modeled CPs to gain an insight if the execution time flexibilities that were built into the case model were actually used in practice. Apart from that, the process models will be used to evaluate if the additional communication and collaboration functionalities that the SACM provides significantly affected case execution. Finally, an organizational analysis will be performed to check the overall usage of the user and role system as well as to verify the quality of the modeled user and role assignments.

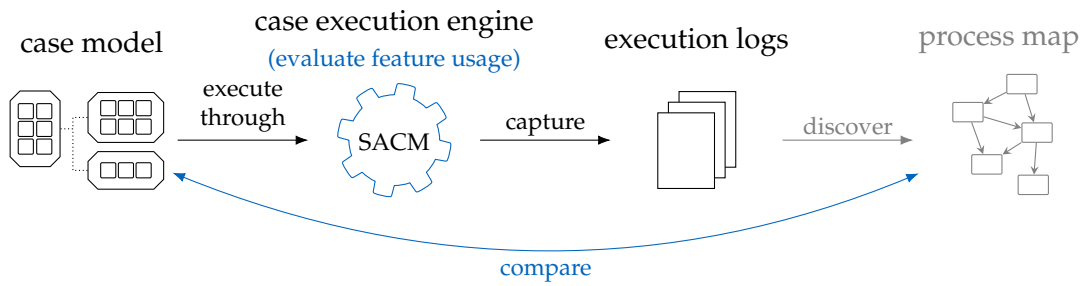


Figure 1.1.: Using process discovery to evaluate a model-based case engine. Evaluation goals are colored in blue, process steps that are necessary to enable them are colored in gray. Based on [154].

1.2. Fundamentals

In order to fully grasp the motivation of this research, the following section will define and explain some underlying concepts in more depth.

1.2.1. Knowledge-intensive Processes

KiPs can be defined as "processes whose conduct and execution are heavily dependent on users ('knowledge workers') performing various interconnected knowledge intensive decision making tasks" [147, p. 151].

KiPs are quite common and found in many businesses, e.g. for managing campaigns or providing technical support, but some business sectors like healthcare, research or emergency management feature especially many of them [29, 147]. Often, KiPs feature process elements that are executed optionally or with dynamic order, making the resulting process unstructured, unpredictable, emergent and non-repeatable. Besides, they are mostly executed collaboratively in a multi-user environment and are constraint-driven, meaning which action to take next may be defined by a set of explicit or implicit rules that the enacting user needs to comply with. These constraints can result in parts of the KiP being structured, depending on the exact character of the rules and constraints that apply.

Concluding, one can say that the degree of structure inversely correlates with the knowledge intensiveness that a certain process bears. The more rules that apply, the higher the possibility of a structured process outcome but the lower the degree of freedom the enacting user has when performing process decisions. This relation can also be seen in figure 1.2 which shows the four color-coded, structural categories according to Di Ciccio et al. [29, 105].

In the context of healthcare, most clinical processes like diagnosis and treatment can be classified as KiPs and bear only little structure. In contrast, organizational processes like patient admission or scheduling of appointments are highly structured, overall rendering the field of healthcare a hybrid regarding the degree of structure [28].

1.2.2. Clinical Pathways

The fact that evidence-based medicine is the primary paradigm in current healthcare practice lead to an increase in quality of care in a reproducible way [69, 133]. CPs can be seen as an instrument of clinical quality management and continuous quality improvement according to the principles of evidence-based medicine [119]. By iteratively improving them, CPs can act as a means of achieving the two goals of increasing the quality of care while minimizing costs through process optimization.

While the term Clinical Pathway and the main concepts behind it are used internationally, literature reviews showed that "there is no single, widely accepted definition of a clinical pathway" [26, p. 553]. Different terms are sometimes used to refer to the same concept (e.g. critical pathway, care pathway, care map) [26]. For this thesis, the definition

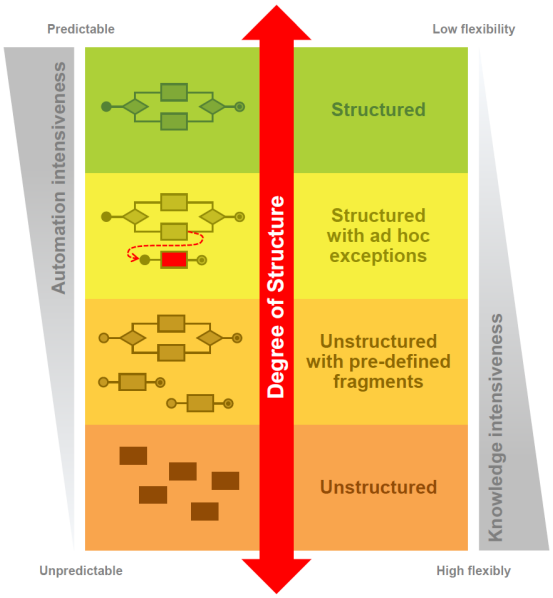


Figure 1.2.: The spectrum of structural degrees of a process according to Di Ciccio et al. [29, 105]

by *Mosby's Dictionary of Medicine, Nursing & Health Professions* is used. It defines CPs quite broadly as "a description of practices, usually in the form of an algorithm, likely to result in favorable outcomes for patients with a particular diagnosis that uses prospectively defined resources to minimize cost" [108, p. 382].

Put simply, CPs are an abstract, generic model of the treatment of patients with a certain condition. They are developed by medical professionals and based on scientific publications and clinical studies but also professional experience [28, 45]. The development of a CP is hard as it needs to model real human behavior in an everyday healthcare setting consisting of a large amount of KiPs with little structure. Therefore, CPs need to find an abstraction that represents reality while still providing relevant process and cost optimizations. The possibility for variations and overall flexibility is then purposefully built into the CP to allow for deviations from otherwise commonly followed paths, if necessary [111]. This generic CP is then adapted to the individual situation by the enacting knowledge worker when put into practice.

As CPs are one of the major instruments for managing the overall quality of healthcare, the developed CP's quality is of high relevance. This, in turn, is often hard to evaluate, due to the highly dynamic nature of the KiPs that are modeled by CPs, in many cases making the real world execution and an ex-post analysis the only way of assessing quality. IT applications have been shown to be able to improve the process alignment and thereby the CP conformance when they are integrated in the clinical routine [91].

1.2.3. Hospital Information Systems and Adaptive Case Management

A Hospital Information System (HIS) is "the socio-technical subsystem of a hospital, which comprises all information processing as well as the associated human or technical actors in their respective information processing roles" [62, p. 30]. This means that a HIS consists of many components and subsystems, as every information processing facility that the hospital incorporates is part of the HIS. Many of these facilities are expert systems, often with very specific purposes for individual departments (e.g. web interface to an x-ray scanner accessible to radiology specialists). But there are also integrative subsystems that span multiple or all departments and provide more holistic functionalities like central electronic health record management or applications for ordering medicine.

As HISs are very complex systems with large differences in implementation between individual hospitals, it is hard to find a classification that is generally applicable to all parts of a HIS. When only considering the information processing tools of a typical HIS, however, Mans et al. came up with such a classification, giving a good impression of the types of technical subsystems to expect. The classification primarily considers properties relevant to PM, but in the context of this work, this is a benefit rather than a limitation.

Administrative systems are responsible for tracking and billing which services were delivered to each patient. These could be surgeries, examinations, treatments, and the like which are often entered manually into the system.

Clinical support systems are used in departments with such a particular set of needs that they require a separate, highly specialized IT system, e.g. in an intensive care unit.

Healthcare logistics systems are concerned with the general hospital logistics like staff management, scheduling of appointments or ordering of medicine and supplies.

Medical devices are physical devices and their human-machine interfaces that are used by medical professionals like x-ray or MRI scanners. They often capture low-level data regarding their operational parameters.

A special kind of subsystem that can be found in an increasing number of hospitals but does not exactly fit this categorization are case management systems. They can be classified as process-aware information systems (PAISs). A PAIS can be defined as "a software system that manages and executes operational processes involving people, applications, and/or information sources on the basis of process models" [157]. Case management systems can further be divided into two categories called production case management (PCM) and adaptive case management (ACM) systems [61, 109, 142]. The former is used for executing a pre-defined process that has been put in place by software engineers or case modelers and cannot be modified at runtime. ACM on the other hand offers the advantage of providing users like knowledge workers the ability to modify the process at

execution time, creating a very flexible system that can be used for supporting the execution of KiPs. What systems of both categories have in common is a so-called case folder where all information relevant to a specific case is stored and from where all available actions are initiated. The case folder also contains a history of actions and changes that were taken during the course of the case.

1.2.4. Case Management Model and Notation

CPs traditionally come in the form of documents (written guidelines, task lists, questionnaires) aimed to be understood by humans but not necessarily by machines. In recent years, ACM systems are emerging as supporting tools for KiPs and ACM applications are offered by an increasing number of commercial providers [85]. These, however, use proprietary formats for expressing CPs, making the tools largely incompatible with each other, amongst other drawbacks. This is why the Case Management Model and Notation (CMMN) was developed in the year 2014, updated in 2016 and is now maintained as an open specification by the *Object Management Group* [2] [116, 117].

CMMN is a metamodel and a notation for representing, graphically displaying and interchanging case models for use in case management tools like ACM systems. Conceptually, it is quite similar to the Business Process Model and Notation (BPMN) but it is used for more dynamic processes with a higher degree of human decision making like KiPs [101]. This also implies that it provides more flexibility and adaptability than the classical BPMN does, like the possibility to skip certain activities or to execute them in a dynamic order. CMMN models can also aid in the process of decision making by providing next step suggestions or automated state transitions based on pre-defined events. Overall, this results in CMMN models primarily describing *what* can or cannot be done in a process while BPMN models define *how* to actually do it [100].

An example of a CMMN diagram can be seen in figure 1.3. The main elements of a typical CMMN diagram are explained in the following.

Case is an abstract representation of an optimized pathway a patient with a certain condition is likely to follow. It mostly consists of multiple stages and often has a case manager assigned. *Graphical representation: rectangle (outermost)*

Stage corresponds to an episode in a patients case. It groups together multiple tasks and represents a subprocess in the overall CP. It can optionally have entry or exit conditions and can be skipped or repeated if wanted. *Graphical representation: box with clipped off corners (octagon)*

Task is a unit of work which is atomic in the context of the CMMN model. It provides means for automatic activation when certain events occur as well as for manual activations by human users. It can be defined to allow repetitions or be required for the

[2] also responsible for BPMN, UML and others

completion of a stage or the activation of another task. Tasks have a special subclass *HumanTask* for tasks intended to be done by a human caseworker. Every task can have pre-defined user roles that constrain who can be responsible for its execution. *Graphical representation: rectangle with rounded corners*

Sentry is an optional watchdog linked to a certain task or stage waiting for pre-defined events or conditions to occur. It controls activation and completion of tasks and stages. *Graphical representation: diamond shape on the dependent with dashed lines to dependencies*

Role acts as a restriction for limiting which users can perform a task. *Does not have a graphical representation*

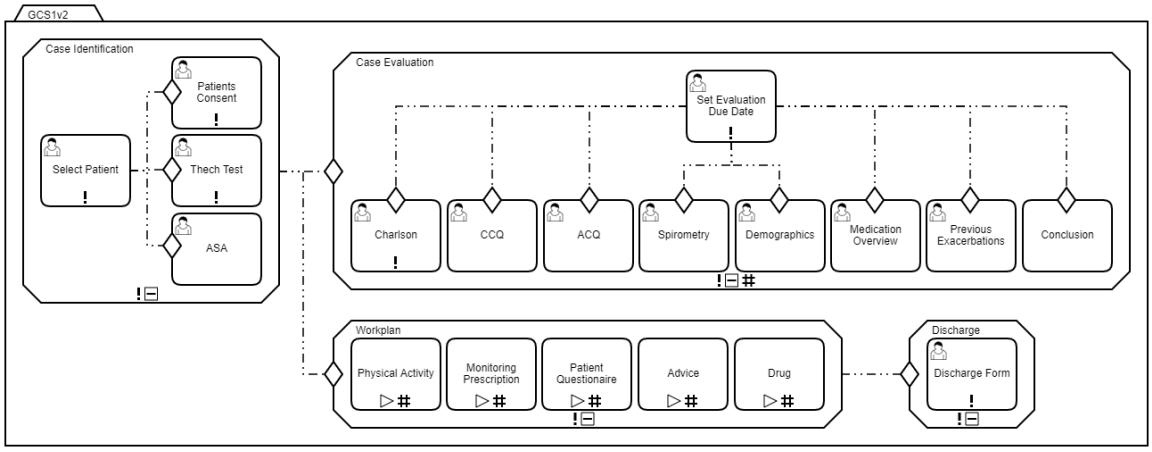


Figure 1.3.: CMMN diagram showing the second version of the CP for Groningen's first CONNECARE case study

1.2.5. The CONNECARE Project

As mentioned in section 1.1, there is an overall trend towards an increasingly aging population [115, 146]. This leads to an increasing amount of CCPs, as the number of chronic diseases increases with age [174]. CCPs can be defined as patients with two or more chronic diseases, comorbidities or frail (due to social, economic and/or clinical factors), who are usually elderly and consume a very high level of health resources [167]. They are a major cost factor in the healthcare sector as they require a large chunk of the overall health resources while their share in the total population is rather small [168]. In Catalonia for example, CCPs were responsible for 35.8% of costs in the year 2014 while they only made up 5% of the overall population [16]. Therefore, there is a strong incentive to unify and optimize healthcare processes for CCPs to increase cost-effectiveness and reduce overall

spending. Additional measures like home hospitalization can bring the costs down even further while also increasing the patient's quality of life [65].

Clinical Pathways for Complex Chronic Patients

One way to achieve this is to make use of CPs to guide clinical professionals when treating chronic diseases, as has been shown for patients with acute conditions [119]. But to leverage the full potential of CPs, some additional requirements have to be considered, as chronic conditions vary from acute ones. While most acute complaints are hard to prevent, this is not the case for chronic problems, as the nature of the condition is normally known in advance and steps can be taken to prevent rehospitalization [65]. This increases the quality of life for CCPs as they can spend more time in their familiar surroundings and need to spend less time in the hospital. However, these improvements induce extra system requirements as most preventive measures are coordinated by primary care or by unpaid carers (e.g. relatives or social workers), effectively enforcing the integration of primary, secondary (i.e. clinical) and unpaid care. Medical professionals also profit from such integrated systems, as they simplify interdisciplinary exchange by providing a platform for cross-organizational communication and data transfer. They can also provide professionals as well as patients with event notifications and incorporate components for automated data checks based on pre-defined rules.

Integrated Care for Complex Chronic Patients

The integration of these systems is however not a trivial task and provides many challenges on its own, which has been shown in multiple studies trying to achieve such an integration [15, 53, 91]. Further improvements can be achieved when medical devices for home use are integrated as well, enabling patients to do simple, routine measurements by themselves instead of having to visit the hospital [167]. Imagine for example an elderly person with hypertension who regularly needs to visit a hospital or primary physician for taking a blood pressure measurement. The integration of a personal blood pressure measurement device with the HIS can completely eliminate the need for routine hospital visits which can greatly increase the patient's quality of life. This functionality can even be extended into a system for patient self-management, enabling the preponing of the hospital discharge after acute episodes.

The EU-funded project CONNECARE aims to achieve that [40]. It offers an integrated care environment for managing complex clinical cases and features an interface for medical professionals of primary and secondary care as well as a self-management system that can be used by patients, relatives or personal carers [4]. The system also connects medical devices for home use and provides the patients with notifications and recommendations based on the gathered data. Additionally, dedicated communication channels are provided that can be used for interchanging patient-professional and professional-professional messages, simplifying interdisciplinary collaboration. CONNECARE was

developed as a multi-tenant system with the intention of productive use and is therefore designed to be adaptable to individual deployment sites but also to scale well.

Since June 2018, it is used in everyday clinical life as part of two case studies executed at three locations across Europe[3] which are described in more detail in section 1.2.7. The first study is concerned with the prevention of unplanned hospital-related events involving frail complex patients with high risk for hospitalization. The second study aims at preventive, patient-centered intervention in CCPs undergoing major elective surgery. On top of that, the system is being fully integrated into an existing, *SAP*-based HIS at a fourth site, but will most likely not be actively used for managing CCPs there, as the primary goal of that integration is a large-scale deployment that can be extended to provide services for an entire country's population.

System Architecture

CONNECARE's system architecture is divided into three major subsystems that communicate via a common message broker and can be seen in figure 1.4. The message broker relies on a JSON-based API for communication, enforcing this as the standard way for exchanging messages in the backend. The individual subsystems will be explained in detail in the following.

SACM The SACM contains the central case management component. It houses the main business logic expressed through a case model, is the primary data storage and offers the web interface for medical professionals and case modelers. Access for the latter is necessary as the SACM's backend is – in its capacity as an ACM system – the process engine of the CONNECARE environment. Obviously, the case modelers transforming the CPs into machine-readable form and adapting them to the individual needs of the deployment locations need access in order to do their job. The SACM is described in more detail in section 1.2.6.

SMS The Self Management System (SMS) is the patients' access to the overall system. It does not feature a web UI but is rather used through an app for smartphones or tablets. Apart from patient notification and communication functionalities, the integration of medical devices is accomplished through this subsystem. This is achieved by integrating the medical devices with the smartphone application using whatever means of communication both devices support (e.g. Bluetooth®). The app then transmits the data to the SMS backend which in turn passes it on to the message broker so that it finally reaches the SACM where it can be persisted.

[3] see https://clinicaltrials.gov/ct2/show/NCT02956395, https://clinicaltrials.gov/ct2/show/NCT02976064, https://clinicaltrials.gov/ct2/show/NCT03327233 and https://clinicaltrials.gov/ct2/show/NCT03327246

UIM The User Identity Management (UIM) is responsible for managing users, roles and active sessions. This is a critical task, as CONNECARE contains personal health information which needs to be properly secured in order to suffice European data protection regulations. Authentication tokens can be obtained from the UIM, which are necessary for accessing any other services that the system provides. It contains the user information for all tenants and features a role system for restricting the rights of groups or specific users down to the individual case level. All user and role information is mirrored and synchronized from the central SACM system for performance reasons.

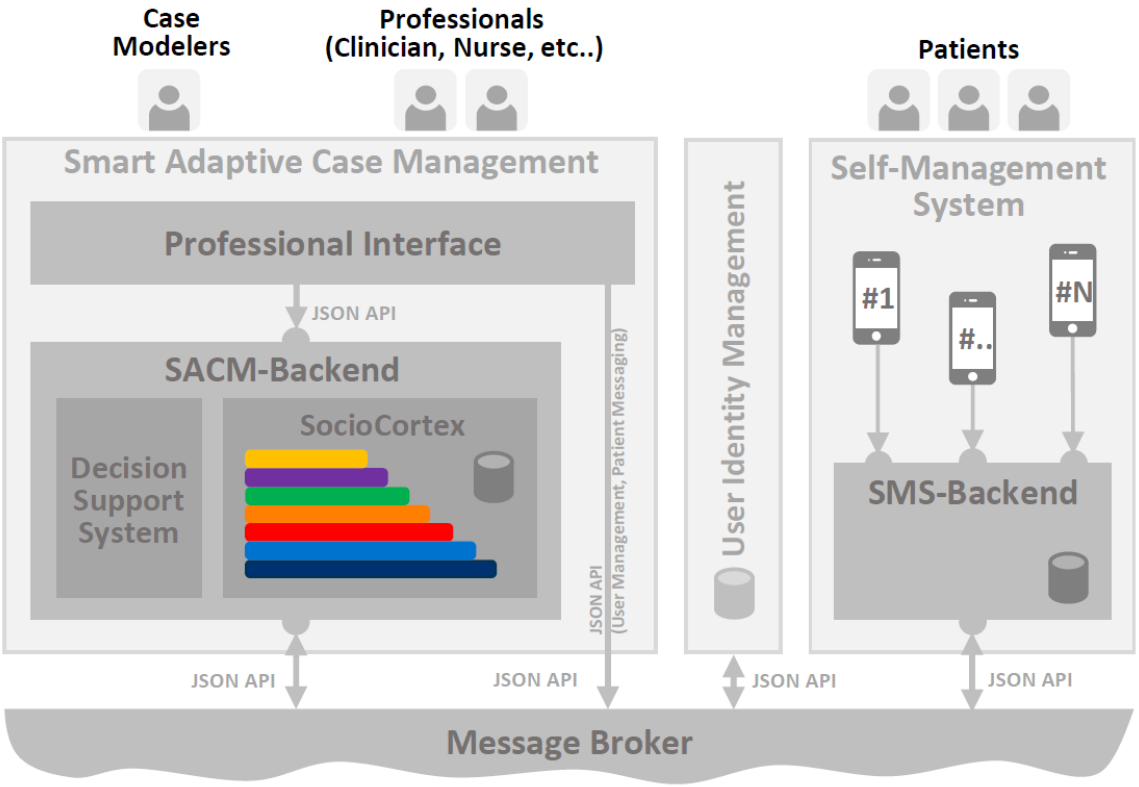


Figure 1.4.: Conceptual Architecture of the CONNECARE system [104]

1.2.6. Smart Adaptive Case Management

The SACM is CONNECARE's most central subsystem, provides the ACM functionality and is the primary persistent storage [104, 105]. It consists of two components, the Professional Interface that acts as a web frontend and the SACM-Backend. Communication

between both components is done directly using a RESTful, JSON-based API. For providing some functionality like patient messages, the frontend also communicates directly with the message broker. Full API logging is supported and metadata on all incoming requests is persisted to a separate logging database. The backend of the SACM is basically a process engine with some extended functionality. As a second component, it will feature a decision support system in the final release that provides clinicians with statistical probabilities and recommendations on how to proceed with a case. Such a system already exists in a prototypical form, but it is not yet integrated into the SACM in any form. Basic case modeling support is backed by SocioCortex, a generic full stack modeling engine that has been extended to fit the medical use case and the integrated care scenario. A detailed description of SocioCortex was given by Hernandez-Mendez et al. [66]. Extended functionalities include case-based communication for involved users, patient notifications, automated validation and alerting for professionals, and an extended user and role management.

Due to its underlying generic case engine, the SACM backend is highly flexible with regard to the case models that it can support during execution. All functionalities of the system are configured through case models, making the modeling step a crucial part of the development and a major factor in determining the quality of the operational support when executing CPs. The models are expressed in an XML-based syntax that can – except for some extended functionality – be transformed into CMMN, enabling simplified model visualization and modeling by externals who are unfamiliar with the project or its exact case model syntax. The extensions to CMMN include amongst others a differentiation between serial and parallel repeatable tasks, the introduction of case notes as a means of keeping completely unstructured information and the addition of extra task types like *DualTask* to properly describe tasks that are partly done by a professional and partly by the patient using an integrated system.

The described case modeling engine provides the SACM with a very high degree of adaptability. This is a central requirement, as the multi-tenancy of CONNECARE requires it to be adapted and deployed to different locations and patient treatments. As overall organizational processes and CPs vary from site to site and from treatment to treatment, adaptability has to be provided at design-time, in addition to the runtime adaptability that is required for supporting KiPs.

1.2.7. Performed Case Studies

As the case studies that are carried out in the course of the CONNECARE project are currently still being executed, only little information has been published yet regarding the performed treatments. In particular, no precise medical description of the exact hospital procedures exists. For this reason, the following section will refer to published short descriptions about CONNECARE's intended use, to example cases taken from the project's website and to the labels of the tasks modeled in the CMMNs to still give the reader an impression of the medical cases that are supported by the system [22, 40, 74, 75, 76, 167].

However, the section's goal is not a complete and fully correct description of the case studies from a medical point of view but rather to give a conceptual impression of the executed processes. Furthermore, medical abbreviations like ASA or COPD will not be spelled out, as this work is primarily targeted at information scientists without a deeper medical understanding who would not benefit from the complete terms anyways. Interested readers can refer to the sources provided with each abbreviation for further information or check the glossary at the end of this work to find the spelled out medical terms.

All case studies follow a similar overall structure consisting of the five stages *Case Identification*, *Case Evaluation*, *Workplan Definition*, *Workplan Execution* and *Discharge* that was developed during earlier research projects [15]. It can be seen in figure 1.5, colored according to the structural degrees by Di Ciccio et al. from figure 1.2. This top-level conceptual model also contains a specific, backward-facing path for unexpected events, enabling clinicians to react to unpredicted situations by re-defining the workplan stage. In the context of CONNECARE, the definition of the workplan can be done continuously and at runtime, reducing the number of stages that need to be modeled to four. This can be seen in the CMMN diagrams included in chapter A of the appendix which are also colored according to their structural degree. The only exception to this structure is the second case study in Lleida, where the Workplan stage is split into *Workplan before Hospitalization* and *Workplan after Hospitalization*.

Case Identification Stage collects patient information that is required to start the case [15]. Across all sites, it contains mandatory tasks for selecting the patient and the supervising medical professionals of a new case, checking the technological background regarding internet and smartphone usage, as well as for documenting the patient's consent to taking part in the study. Apart from that, one or two additional indicators can optionally be collected like the ASA index in Groningen for grading the overall physical status or the Global Deterioration Scale in Lleida for assessing some forms of dementia [125, 132].

Case Evaluation Stage includes more specific assessments of the patient to determine the eligibility through CONNECARE and to capture the information required in the next step, the workplan definition, which is performed at runtime and is therefore not modeled. The evaluation stage has the highest amount of available tasks that can be used for assessing different aspects of the patient's health. The stage also shows the most differences from the first to the second case study but also from one site to another and will, therefore, be described in more depth after the general case model structure has been presented. Still, it contains some tasks that are shared across all case models like the Charlson Comorbidity Index that was developed and refined over decades and is a central instrument when determining the severity of a patient's conditions and how aggressively to treat them [18, 19, 124].

Combined Workplan Stage is the most flexible stage but, at the same time, its modeling varies only very little across all sites and both studies. This can be seen by the fact

that almost all tasks are optional, need to be activated manually and can be instantiated multiple times in parallel. Additionally, all case models feature the same five tasks, at least at later points in their version history: *Physical Activity* for personal activity monitoring using fitness trackers, *Monitoring Prescription* for regularly checking a value like blood pressure or weight, *Patient Questionnaire* for informally gathering personal opinions[4], *Advice* for giving the patient guidance by a professional, and *Drug* for the intake of prescribed medicine. As mentioned, the second study performed at Lleida constitutes a small exception to this as it splits up the stage and the just mentioned tasks into two stages. Apart from that, both Lleida studies offer extra tasks for social carers and primary nurses.

Discharge Stage is quite simple and mostly just documents the date that a patient was discharged from hospital. Lleida also uses this stage to capture SF12 and EQ5D values, which are scales used to determine a person's physical and mental health and overall quality of life, respectively [41, 170].

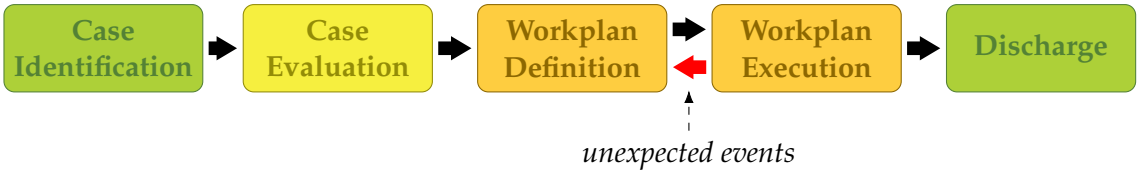


Figure 1.5.: Conceptual model of the performed case studies, colored according to structural degrees from figure 1.2. Developed by Cano et al. [15]. For CONNECARE, both workplan stages have been combined into a single stage.

First Case Study

The first case study aims to improve the monitoring and care for CCPs that went through a not further specified acute episode after they have been discharged from hospital, in order to reduce the readmission rate during the first 30 days at home [75]. Besides the common goal for all three sites, there are implementation differences, especially when comparing the amount and identity of tasks of the evaluation stage across the different sites. While Groningen focuses heavily on COPD, both Lleida and Tel-Aviv take a more holistic approach, therefore requiring a higher number of possible activities. COPD is a general term for chronic, progressive and irreversible lung diseases that are characterized by reduced lung capacity like e.g. asthma [108, p. 365]. COPD patients have been shown to have an increased risk for developing comorbidities and to cause an over-proportional amount of healthcare expenditure, so they can be characterized as CCPs [143, 168]. Groningen's evaluation stage is focused on the condition, as it features the tasks *CCQ*, *ACQ* and *Spirometry*

[4] e.g. "How do you feel today?"

which can be used for diagnosing COPD [73, 107, 164]. Even the task *Previous Exacerbations* is likely to be related to COPD exacerbations, which are a common problem during treatment of such diseases [17]. As the amount of tasks in Groningen's evaluation is not that high anyway, this leaves only some generic tasks like *Demographics* or *Medication Overview* and shows the focus on COPD-related conditions.

Lleida and Tel-Aviv, on the other hand, do not model these tasks, but rather evaluate more generic aspects of the patient's health. To name some examples, both sites feature ways for measuring the basic and instrumental activities of daily living, like eating and dressing (basic) or shopping and managing money (instrumental) [108, p. 29/183/934], through tools like the *Barthel* index [94]. Apart from that, the *HADS* can be employed to check for depression, the patient's overall quality of life can be assessed using the *EQ5D* and an indicator for physical and mental patient health can be calculated using the *SF12* task [41, 170, 177]. Even tasks for estimating a patient's risk of fall exist.

But there are also some differences between Lleida and Tel-Aviv, though they are not nearly as big as the ones between Groningen and the other sites. Lleida performs a *GMA Evaluation*, which classifies a patient as a member of one of six morbidity groups based on the local population [106]. This is somewhat similar to Groningen's *Demography* evaluation and is not performed in Tel-Aviv. Instead, the *MUST* test as well as the *Sweet 16* test can be performed for measuring nutritional and cognitive health aspects [34, 48].

Second Case Study

The second study tries to prevent perioperative complications[5] for CCPs undergoing major elective surgery by providing improved post-discharge monitoring and by promoting physical activity and prehabilitation [76]. The latter is a form of physical exercise that aims to prevent injuries before they actually occur and can be employed prior to surgeries for reducing the chance of injuries and complications [103]. For this purpose, the available tasks are more generic for all three sites, while, in comparison, Groningen still stands out due to its low amount of tasks. However, part of this effect can be explained by Groningen having a mainly different set of tasks compared to the first case study while the other sites kept most available tasks and only added some new ones.

At the same time, the differences regarding the captured medical indicators are not so elaborate anymore with Groningen also capturing nutritional data through the *NRS* and *MNASF* tasks, similar to Tel-Aviv's *MUST* task [34, 55, 82]. Apart from that, all three sites support evaluating the activities of daily living and determining a possible depression [94, 177]. Groningen also determines the *GFI*, an indicator that can be used to get an impression of a patient's frailty and therefore loosely relates to the tasks for measuring the risk of fall that exist at the other sites [138].

Again, there are also some fully site-specific tasks. For Groningen, this is e.g. the *Site of Surgery* task that is done to display the planned site of surgery in the case overview,

[5] i.e. complications before, during or after an operation

though this task is actually part of the earlier identification stage. The evaluation in Tel-Aviv primarily differs from the first case study in the presence of four additional tasks that are all modeled to be executed by a physiotherapist and are a result of the focus on physical checks and prehabilitation. In Lleida, the most apparent change is the split of the workplan stage into a pre- and a post-hospitalization phase, to further facilitate activities preparing the patient for the upcoming surgery. Apart from that, some additional tasks are available, e.g. for capturing neuropathic pain using the *S-LANSS* score or for determining the severity of arthritis on the *WOMAC* index [2, 7]. A task for capturing the *ASA* index value analogous to Groningen is also available.

1.3. Research Questions

Due to their highly dynamic and complex nature, KiPs are hard to understand and generalize, making the design of a CP a non-trivial task. To promote the dissemination of CPs, guidelines like [111] have been created to assist medical professionals during the creation process. Following the principles of evidence-based medicine and because CPs directly affect the treatment procedure and thereby the patient's health, they have to be assessed before they can be put into productive use, to ensure that they provide meaningful advantages over the status quo [111, 133]. Because evaluating a developed CP on paper is next to impossible, most guidelines recommend to execute it in practice through a small pilot study before integrating it in the ordinary course of patient treatment. By doing so, possible issues can be identified and by measuring treatment costs and patient satisfaction, an actual improvement factor can be determined that can then be used for comparison with other developed CPs and the like. Afterward, the CP can be refined and, depending on the changes, be put into clinical practice or re-evaluated through another pilot study.

But already the process of translating a paper-based CP into machine-readable form has its challenges that need to be considered [102]. Supporting software systems with advanced functionality exacerbate the situation, as those features could (and are maybe even designed to) impact the execution of the CP. The technical correctness and the fulfillment of all functional requirements can be assessed using a set of automated software tests, but this is not possible for conceptual validity, execution efficiency or ease of use. To gain insights into these areas, the system has to be put into productive use, e.g. by performing a pilot study like it is done when introducing a paper-based CP. During the course of the study, data on the system's runtime behavior needs to be collected so it can later be analyzed for evaluation purposes. This is exactly what is currently being done for the CONNECARE project and, more specifically, its case engine SACM. Case studies are performed at three hospital sites for two different treatments each, resulting in a total of six evaluation scenarios. The gathered data comprises a complete technical log of all requests to the engine's RESTful API that is created by the engine itself and is persisted to a separate logging database.

1.3.1. Process Discovery as Enabler for System Evaluation

Process discovery, a PM technique, will afterward be applied to generate process models (also known as process maps) for comparing the modeled CPs with the actual treatment process that was executed using the CONNECARE system. PM is a research field that emerged in recent years and is concerned with the analysis of processes based on event data [144, 153]. It combines techniques known from data mining with concepts of business process analysis and consists of the three major tasks *process discovery*, *conformance checking* and *process enhancement*.

Process discovery automatically constructs process models based on observed events, thus capturing the behavior seen in the event log [154]. It features three prevalent perspectives: the *control-flow*, the *performance* and the *organizational* perspective. The control-flow perspective only analyses the ordering of events, is the base perspective and connects all the others. It shows which activities followed each other, how often they occurred, and gives a general impression of how the process under analysis looks like. The performance perspective tries to give insights into how efficiently the process was executed and can be used e.g. during process enhancement to identify bottlenecks. The organizational perspectives focuses on resource usage and does not analyze *what* was done but rather *by whom* or *where* it was done [136]. This is done to unveil social networks and other organizational structures like room utilization.

For conformance checking, the modeled behavior (e.g. the case model/CP) and the observed behavior (i.e. event log/process map) are automatically compared in order to evaluate the existing model or as a starting point for future improvements [153]. Process enhancement, in turn, is concerned with aligning event log and process models, so that the log can be replayed on the model, enabling further insights into e.g. the process performance and the main bottlenecks. Additional information like event duration needs to be available in the log for being able to perform this step.

In the first step, the goal of this thesis is to employ process discovery techniques for generating process maps of the case studies performed at the different hospital sites in the context of the integrated care environment of CONNECARE. The process models will be generated based on API access logs created by the SACM backend system and can then be compared with the underlying case models. Afterward, the findings identified during the comparison will be used to answer three research questions regarding the evaluation of the SACM system, the modeled CPs and the CONNECARE integrated care environment. Due to the case studies still being performed until the end of 2019, all steps leading up to the process discovery will be automated, so that the evaluation can be repeated with minimal effort as soon as the studies are completed. As already mentioned, the overall evaluation procedure, the process discovery step and the research questions can be seen in figure 1.1. In the following, each question will be explained in detail and some reasoning will be given why it is relevant and worth answering.

1.3.2. RQ1: How is the model-provided flexibility employed during the execution of cases?

As described in section 1.2.1, KiPs are shaped by knowledge workers and their human decisions resulting in highly varying processes. This is necessary as KiPs are often executed in a context with many unknowns where unpredictable and unforeseen events can occur, possibly requiring some human action to account for the new situation. For this reason, a CP is only a guideline for the treatment of a certain disease and needs to be adapted and extended ad-hoc by medical experts when it is executed. Supporting IT systems need to allow a similarly high degree of flexibility to properly support the execution of KiPs. On the other hand, to still provide meaningful process optimizations, such a system also needs to apply some kind of structuring to the executed processes.

Therefore, during the modeling phase of the CONNECARE project, elements supporting adaptability and flexibility were purposefully built into some areas of the case models while others were kept more strictly structured. Figure 1.5 showed the different stages that a typical case executed during the case studies consists of and also visualized their structural degrees that served as a basis for the implementation of the model-provided process flexibilities. The flexibility is achieved through the use of case elements with certain properties. Stages and tasks can be executed optionally and possibly be repeated in serial or in parallel. Apart from that, many of them are activated automatically when certain conditions apply, but they can also be modeled to require manual activation by a human user. Sentries exist for enforcing certain inter-task or inter-stage dependencies and can be used in combination with the required property of tasks and stages to limit the possible flexibility of the overall case.

In theory, this results in highly dynamic process executions as knowledge workers can modify and adapt the executed process to their liking, based on the unpredictable real-world events that occur. Unfortunately, it is very hard to validate this hypothesis in theory, as the executed KiPs are very complex to simulate because they integrated multiple departments and therefore the major part of a whole hospital needs to be simulated which is a non-trivial task [51]. Furthermore, unexpected events cannot be simulated in the full variety that they have in real cases. Therefore, the system has to be used in a real-world setting to generate valid data based on truly unforeseen events. The data can then be used to discover process models of the real-world case executions and compare the flexibility and variability of those execution maps with the abstract and generic case models that were created based on the hospitals' CPs.

1.3.3. RQ2: How do communication and notification features affect case executions?

Beyond the fundamental IT-supported guidance through a case, the SACM offers additional functionalities designed to enhance the user experience of medical professionals and patients or simplify everyday life. To be precise, the SACM offers alerts, notifications,

case notes and communication features. Alerts notify a medical professional when an automated measurement was made but violated some threshold or other restriction that was previously defined at design- or at runtime. Notifications remind patients when they need to take a prescribed medication or encourage them to start or keep up with certain tasks like e.g. physical activity. The messaging features allow patients to directly contact an informed medical professional to resolve emerging issues or to get advice in unforeseen situations and vice-versa. Messages can also be sent between the professionals without access for patients, enabling case-based, interdisciplinary exchange and collaboration. The notes feature offers a completely unstructured place for gathering case-based information and collaboratively creating knowledge similar to what can be done with a wiki page [171]. The visual appearance of the notes section of a case can be pre-defined in the case model using HTML and CSS, making it a flexible tool for keeping knowledge in a freely structurable way that is unknown ex-ante.

Some of the features, like the team messaging functionality, have been requested by the professionals of certain sites in the early stages of the case studies while others were already implemented in the initial version. However, it is currently unknown whether all features have been used in the same way across all sites and case studies. At the same time, occurring alerts or messages can be viewed as unforeseen events that potentially require ad-hoc process changes. The intention of the second research question is therefore to generate further insights into how these additional SACM features are used by evaluating how they affect the execution of cases based on the process maps generated from the captured log data.

1.3.4. RQ3: How are collaboration and organization features reflected in case executions logs?

To support all possible deployment scenarios and conform with EU data protection regulations, especially for person- and health-related data, SACM implements a configurable user and role system. It will be explained in detail in the following paragraph and is also shown in figure 1.6.

Cases consist of stages which in turn are made up of tasks. Each task references a certain role that is responsible for its execution. Roles are defined at the case level and restrict which groups a user has to be part of for assuming the role. The assignment of users to groups is done globally and users can be members of multiple groups. When the case is executed, the case manager has to select a patient as well as a professional for each role before work on the case can be started. To give an example, a case could define the role *Radiologist* which is modeled to be responsible for all tasks related to radiology that can be performed during the case. This does neither mean that the user actually has to perform those tasks, nor that the tasks will be performed at all, but the user will be listed as the responsible task assignee in the user interface. For restricting who can be chosen as responsible radiologist, the case model could e.g. require membership in one of the two groups *Imaging Radiology* and *Therapeutic Radiology*. These represent the corresponding

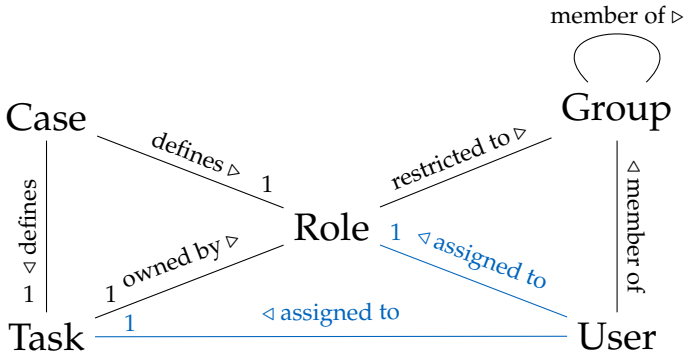


Figure 1.6.: Conceptual data model showing the role-based task responsibility system. Relations colored in blue are usually defined at runtime.

departments in the hospital and, as all department members actually have a membership in one of the two groups, any of them can be assigned to perform the *Radiologist* role.

Nonetheless, this does not necessarily affect how tasks and cases are executed, as any user with write access to a certain case can modify, complete and terminate all available tasks. Write access is gained automatically as soon as a user is assigned to a case role or through manual assignment. In the context of this work, the person actually completing or terminating a certain task will be called the *task processor*, in contrast to the *task assignee* who is responsible for it.

Due to this runtime flexibility, it is unclear if the modeled roles and the users performing them are actually used in practice and, if so, whether they pose significant advantages or are simply ignored. It is also possible that some kind of organizational structure is employed when the tasks are executed which the system could support if configured correctly through the case model. Therefore, the goal of this research question is to evaluate this aspect of the modeled cases, determine how the user and role system is used across the different sites and identify possible process improvements like the model over- or underfitting the real-world case execution logs.

1.4. Thesis Outline

The remaining thesis is structured as follows: The upcoming chapter 2 will first give an overview of similar works and their results in section 2.1, followed by a more detailed description of more closely related research efforts in section 2.2. Then, in section 2.3, the specific challenges that arise when performing PM in the healthcare sector are explained in depth.

The next part of the thesis, part II, contains the conceptual approach as well as a description of the practical implementation. First, section 3.1 shows ways how to deal with the challenges identified earlier while section 3.2.1 presents the consolidated methodol-

ogy by Erdogan and Tarhan that is used in this thesis. Afterward, section 3.2.2 explains what methodological changes and extensions were necessary in order to realize all project goals. Then, in chapter 4, the most relevant implementation details are given for each step of the chosen methodology. Quite at the end, in section 4.4, current PM tools are compared and instructions for the configuration of *Disco* are given, the tool that was used in this thesis. The chapter is concluded in section 4.5 by notes on the clinical deployment that was performed in the context of this thesis.

The final part evaluates the generated process models and the calculated indicators and interprets the findings for each site. The sites are then compared with each other to find possible similarities or differences. This is done for all three research questions, starting with the model-provided flexibility in section 5.1, the effects of communication features in section 5.2, and the usage of the user and role system in section 5.3. The first two questions are analyzed using control-flow perspective process models while the latter one relies on the organizational perspective. Having answered the research questions, a conclusion is drawn in section 6.1 followed by the limitations of this thesis in section 6.2 and an outlook on future work regarding the topic in section 6.3.

2. Literature

2.1. Process Mining in the Healthcare Sector

2.1.1. Overview of the Research Area

PM emerged as a new research field bridging the gap between Data Mining and Business Process Management over the last two decades [153]. It is gaining increasing attention and has sparked the creation of over 25 commercial tools facilitating the employment of PM in the industry [145, 155, 160]. Regarding the application areas of PM, a strong focus on economic sectors like finance and insurance, public administration, healthcare and manufacturing becomes apparent [25, 145]. Some of these industries are characterized by strongly structured processes like closing a new contract in an insurance company, while others are shaped by highly dynamic KiPs as they are e.g. found in the healthcare sector [29]. Structured processes can benefit from advanced PM techniques like conformance checking and process enhancement [144]. KiPs on the other hand often lack a formal representation of the processes which is required for performing those advanced techniques [29]. Nonetheless, the field still provides significant advantages as it enables visually modeling the dynamic processes based on the actual real-world execution events [56, 150]. For this reason, PM has been increasingly applied to sectors that heavily rely on KiPs with the field of healthcare gaining special attention for various reasons [6, 35].

In the six year period between January 2013 and January 2019, ten literature reviews on PM in the healthcare sector have been published [6, 35, 42, 50, 84, 96, 120, 126, 127, 175]. Three of those were systematic/systematized literature reviews [6, 35, 50] with the rest being traditional ones. The majority (n=6) of reviews is concerned with healthcare in general, while some only consider certain areas of the sector, specifically oncology [84], the support through CPs [175], the integration with primary care [126] and the analysis of care for elderly frail patients [42]. The latter only reviewed the small number of eight studies, but as the analysis area is so highly specialized while also being relevant for this work, the review still provides significant value added to be considered in the summary.

2.1.2. Summary of Insights from Literature Review Studies

The following paragraph will give an overview of the most important findings relevant to this work across eleven different analysis dimensions mentioned in the review studies. Data will primarily be sourced from the systematic reviews [6, 35] conducted in 2018 due to their high comprehensiveness. Additionally, instead of the systematized review pre-

sented in [50], data from [127] will be used, as it is the primary source of information for large parts of the formerly mentioned publication. All given percentages have been averaged across the available sources and mathematically rounded to integers for simplified comprehension. Percentages not adding up to 100% for certain analysis dimensions are due to studies employing techniques from multiple categories.

Date of Publication According to the years of publication evaluated in [35] and [50], it becomes apparent that the research area of PM in general but also the scientific application of PM in the healthcare sector is still gaining attention. About 70% (n=1503) of all reviewed publications (n=2176) have been published in the five year period between 2010 and 2015 [50]. The five years before that account for another around 20% of the overall publications, leaving only about 5% for the time before 2005. When focusing on work concerning the healthcare sector, an even more obvious trend can be seen, with roughly 80% of publications being released between 2011 and 2016, only 20% being published in the five years before and only an insignificant share with a publication date earlier than 2006. A quite similar conclusion is reached in [35] where around 70% of PM studies in the healthcare sector were published between 2012 and 2017, 25% being released the five years before and 5% of studies that have been published prior to 2007.

Place of Publication Most studies were performed in Europe (54%), followed by Asia (18%), North America (17%) and Australia, the latter being on par with Latin America (each 5%) [35, 126]. No studies from Africa have been reported across all reviews. Results from [127] show a similarly strong position of Europe but were left out of the calculation of average percentages due to the lack of precise numbers for other continents. Similar conclusions are drawn by [42] which has also been omitted from calculations due to the low count of reviewed studies (n=8). Regarding individual countries, the special position of the Netherlands must be noted, being by far the country with the most scientific studies and contributions [35, 126, 127].

Types of PM This is the review dimension with the highest support across all reviews, being analyzed in seven of ten papers [6, 35, 50, 84, 96, 120, 175]. All publications conclude that process discovery is the type most often applied with an average of 73% of studies making use of it and followed by conformance checking (25%) and process enhancement (6%, sometimes also called *evaluate and improve*). This is no surprise, especially for the area of healthcare, as the less used types require a formal model which often does not exist for KiPs but can be generated using process discovery, making it kind of a preliminary task before other types can be applied [35, 89]. Process variant analysis is performed by 30% of studies as reported by [35], however, this is not an established type of PM (as defined in [153]) and therefore only evaluated once.

Perspectives of PM A quite similar picture emerges when considering the use of the different perspectives of PM [6, 50, 120, 127]. The control-flow perspective is the most

used one (61%), followed by the performance (26%) and the organizational perspective (10%). The dominance of the control-flow perspective is again quite natural, as it is central to many PM activities and integrates all other perspectives by serving as a common basis [152, 153]. Apart from that, it shows the sequence of activities and can be used to understand the flow of a certain process, as it is often the goal of PM projects in the healthcare sector or when applying PM to KiP in general [29].

Process Discovery Algorithm Three reviews investigate the algorithm used for the process discovery step in greater detail [6, 35, 127]. Due to their robustness and ability to deal with noisy, incomplete event logs like they are often found in the healthcare sector [11, 67, 99], the three mining algorithms heuristic miner (25%), fuzzy miner (17%) and alpha miner (7%) are employed most often, followed by the genetic miner (4%) and hidden Markov models (3%). Studies proposing new discovery algorithms were measured by [35] and make up 27% of overall studies. The reviews also capture the share of papers that apply trace clustering to be at 6%, though formally this is not a process discovery algorithm but one used for automated data preprocessing to reduce the complexity of the generated process models by grouping similar event log traces and executing the discovery step for each homogeneous dataset [95, 137].

Software Tools Regarding the software tools that are used for generating the process models, the open-source solution ProM or one of its derivatives like RapidProM is predominantly used in 50% of published papers [6, 84, 127]. ProM's success is partly due to its open, extensible and plugin-based architecture which allows for easy extension and adaption while already providing a great variety of implemented tools and algorithms [97, 165]. Second is the commercial tool Disco which is only employed in 7% of all cases. It uses a proprietary enhanced version of the fuzzy miner algorithm and offers free licenses for academic persons and scientific projects[57]. The remaining 43% of papers rely on different, mostly commercial tools that each make up only insignificant shares of the overall tool usage or implement the tools themselves [6, 84, 127]. A survey done in 2013 reported similar popularity of ProM and Disco, though it was only exploratory and non-representative as it only interviewed 119 people from 26 countries and was biased towards the Netherlands [21].

Methodology The methodology that was applied for performing the PM studies was reviewed by [127] and showed that with a share of 88%, the majority of publications did not use a methodology at all. Self-developed methodologies were employed in 9% of cases, leaving 4% to the process diagnostics method (PDM) and 3% to the L* lifecycle model. PDM and L* were both developed specifically for mining processes [13, 152]. In the context of care for the frail elderly, all reviewed studies made use of a methodology they developed on their own [42].

Data Sources When analyzing where data was extracted from, Batista and Solanas [6] report 14 different data sources identified across only 55 studies that were reviewed,

showing the large spectrum of data types available in typical HISs [67, 99]. The source most often used was recorded logs about activities performed during patient treatments (44%), but CP data (25%), data from patient diagnostics (16%) and electronic health records (15%) were used as well [6]. In 40% of the reviewed studies, other data was used additionally. Summing up all percentages returns 140%, clearly showing that many studies use more than one data source. Of the mentioned sources, only electronic health records are used in publications on care for the frail elderly, making up 25% and leaving 75% to other sources [42].

Data Preprocessing Employed preprocessing techniques are described in 54% of all papers as found by [120]. The intended purpose of the preprocessing is roughly equally distributed between the four categories standardization & semantics (28%), aggregation & generalization (25%), privacy protection (25%) and noise filtering (22%) [6].

Site Comparison Only one review [120] investigated the number of studies performing a comparison between multiple hospital sites and found only one study (4%) doing so. At least in 2015, the time of that review, the topic is clearly underexplored compared to other appliances of PM in healthcare.

Implementation Strategy Regarding the implementation strategy that was used for realizing the PM project, Rojas et al. [127] report the majority of studies (89%) to perform a direct implementation consisting of manual data extraction and manual model generation. A fully automated approach is chosen by 9% of studies, resulting in the use of an integrated process modeling suite for automating all steps of the procedure, from data extraction to model generation and even conformance checks. The remaining papers (2%) chose a semi-automated approach, automating the generation of the event log through data extraction and preprocessing, but performing the actual model generation step manually.

2.2. Related Work

In further consideration of higher specialized primary literature regarding the goal of this thesis and related topics, three separate research areas can be identified. The following sections will name those areas, refer to some of their most influential papers and point to some recent publications.

2.2.1. Mining for Clinical Pathways

As explained in section 1.2.2, many processes in the healthcare sector are KiPs (e.g. the treatment of patients) and are therefore less structured and more complex. Using PM to visualize and analyze them is therefore not as simple as with structured processes [152]. For this reason, it took ten further years until PM was first applied to discover medical

processes, compared to the first applications of PM in general that were done around 1998 [23, 95]. This first application showed that it was possible to model healthcare processes using PM for generating new insights that would otherwise require a lot of effort to create manually [95]. The study also already denoted "spaghetti-like" process models to be a major challenge when mining KiP and identified the need for further research to understand existing discovery algorithms and develop new ones better suited for dynamic processes. This was done later the same year by performing a systematic evaluation and comparison of the seven most-used algorithms for process discovery of CPs which showed that no approach was particularly well-suited for the application, as they were unable to properly deal with noise, missing or duplicated activities and high variation [87]. Fuzzy mining, an algorithm better suited for dealing with dynamic processes and heavily used nowadays, was already proposed a year before that [58] but was only applied to healthcare a year later in 2009 [56].

The first clinical applications of PM that were primarily done to gain insight into CPs and not to e.g. prove the applicability of PM in healthcare, were done in 2010. In [43], the technique was used in a hospital where the resulting models served as additional feedback for clinical experts evaluating the actual execution of certain CPs. The work of [121] additionally applied formal concept analysis and trace clustering before performing the discovery in order to improve the care process for breast cancer patients. The extra preprocessing was done to reduce the high process complexity due to high patient heterogeneity by grouping the patients according to their length of stay, reducing specialized concepts to more generic ones and thereby generating more but simpler process models. Formal concept analysis and clustering techniques are known from the area of data mining where they are used for deriving concept hierarchies and for condensing large, high-dimensional data sets to enable timely data processing [8, 49].

This clustering-based approach is formalized into an own methodology in [24], which focuses on process variants and infrequent behavior. Apart from that, the work showed that the objective results generated through PM can differ from the reality as perceived by medical personnel and developed the first fully automated software tool for PM to reduce the costs of model generation. The methodology of [172] further refined the application scenario by incorporating drilldown and drillup practices, consisting respectively of filtering on specific activities and aggregation during preprocessing, to improve tangible process insights. The data occurring in a typical HIS was analyzed in [96]. Afterward, an exhaustive healthcare reference model was created that spans most possible data types and, based on this artifact, updated potentials of PM in the healthcare sector were identified. Four PM analyses were performed in a Maastricht hospital, each comparing two groups of patients with the goal of gaining insights regarding the influence of the execution of a CP on its effectiveness [118]. The author also identified the use of additional statistics and the repeated discussion of preliminary results with process owners as crucial parts of the overall PM procedure. The work of [176] uses fuzzy mining as an enabler for generating abstract workflow models to perform discrete event simulation on, a technique predominantly used in clinical settings to simulate and evaluate "what-if" scenarios for

optimizing efficiency and reducing costs [51]. The approach was picked up in [72] that proposed an own methodology for performing PM in order to run scenario simulations.

In 2014, a literature review was performed evaluating the use of PM for gaining CPs insights that identified 37 relevant studies in the period of 2004-2013 [175]. Apart from the general findings that were already mentioned in section 2.1.2, it described some challenges unique to PM for CP and named some trends in recent research. While the complex nature of KiPs was still being treated as the main challenge, the list was extended by missing explanations for process variants as well as a lack of systematic thinking when it comes to improving the CPs. Main trends were further analysis of process variants, integrated process management and process customization.

In more recent studies, PM was applied for increasingly advanced evaluations like cross-organizational comparisons or social interaction analyses. The former mentioned was done in [120, 141] which compared patients with chest pain symptoms across four South Australian hospitals while paying respect to extra requirements like population comparability. The approach enabled improvements regarding patient health and treatment costs. The social interaction was evaluated as the primary goal in [3] where PM techniques were applied to find role interaction patterns based on event data from emergency room processes, one of the highly interdisciplinary areas of a modern hospital. The study provided tangible new insights into clinical collaboration and uncovered multiple opportunities for process improvement.

2.2.2. Using Process Mining for Comparative Analysis

Multiple studies have been conducted where PM has been employed for carrying out comparative analyses. This section will give an overview of research on the topic which can basically be separated into visual and non-visual comparative analysis. The latter comprises techniques like delta analysis that were proposed and applied quite early in 2005 [81, 149]. Delta Analysis can be defined as "comparing the actual process, represented by a process model obtained through PM, with some predefined process representing the information system" [149]. Applications include e.g. the comparison of possible sequences of events as given by the model to actual sequences of events as given by the log in order to understand where and why humans deviated from the modeled process [81].

Conformance checking tries to quantify this approach by not comparing two models directly, but rather trying to replay an event log that is otherwise used for process discovery on a given descriptive/prescriptive process model [163]. Afterward, the fitness (i.e. how well the model describes the logs) and the appropriateness (i.e. how simple is the model) are measured thereby quantifying how well a certain model describes the real processes as represented by the event log. This can be achieved by e.g. converting the given model to a Petri net and then checking whether it accepts the log's events like it was done in [163].

In recent research, both delta analysis and conformance checking have gained increasing attention, especially in the area of Business Process Management [38]. There it can be used for quantifying how much a model that was developed in traditional ways using domain

knowledge and expert interviews, fits its real-world execution. For this purpose, new metrics were developed for delta analysis by converting the process graphs to a common vector model form, enabling mathematical comparisons [38]. Methods and algorithms from other areas are also transferred, like e.g. the Needleman-Wunsch algorithm that was initially developed for aligning similar regions in protein sequences and can be used for improving PM results by finding dominant behavior in event logs [37, 113]. Apart from that, improvements regarding the business dimension are also made, like the increased understandability of results through the generation of natural language to describe delta analysis findings [148].

The discipline complementary to non-visual means for comparing models is visual analytics. It combines the striking human capabilities for identifying patterns in unstructured data with automated analyses and specialized, interactive visualizations to improve the understanding of large and complex data sets [78]. In the context of PM, it can generate additional insights based on the generated models but also based on raw event log data [144]. This can e.g. be seen in [10] where the authors present a new type of visualization comparable to an interactive dotted chart analysis that can be used to find patterns in raw event log data and demonstrate its usefulness. Dotted chart analysis is similar to a Gantt chart and shows the events of an event log by plotting dots according to the event time [135]. But while the combination of visual analytics and PM offers great possibilities, it also poses some new challenges [54]. In 2016, a review of the most used discovery algorithms was done regarding their capabilities to support visual analytics [83]. The authors found that, with the exception of fuzzy mining, all widely spread algorithms have some major shortcomings regarding visual analysis requirements and future research is necessary to leverage the full potential of the technique.

The first applications of visual analytics to the healthcare sector could be observed in the year 2017 when e.g. a generic, interactive visual tool was developed to better support the analysis and optimization of CPs or other KiPs based on execution event logs [30]. Another research group addressed the problem from the clinical perspective, developing a domain-specific tool for visualizing event logs related to the CP of supposed sepsis patients [14]. The tool abstracts away any strictly PM-related information, resulting in an interactive, medical view on the executed pathway that can be used by domain experts for understanding common and exceptional behavior. One of the central improvements enabling visual analytics was developed some years before in [151]. The basic idea is to use techniques known from cartography to create abstract views of processes in order to cope with the high complexity of KiP, similar to how a road map aggregates and omits parts of the real world in order to increase understandability[1]

When studies are conducted for evaluating the changes between two processes (intra- or inter-organizational), typically a combination of visual and non-visual techniques is used. The main reason for this is that metrics can easily be captured and updated during preliminary phases of a PM project while the final results are usually better presented and

[1] think of road maps vs. satellite images as of process models vs. real processes

understood when they are visualized, even though this requires more effort. An example of this is the study of differences in the treatment of stroke patients between two hospitals that was also one of the first applications of PM in the healthcare sector [98]. Due to the early date of the study and the fact that the combination of PM and visual analytics was not yet very mature, it relied quite heavily on metrics and only visualized the resulting process models but no event log data.

This has changed in more recent publications. Two studies conducted in the years 2014 and 2015 show this. In [20], a comparison of the actual process of outpatient treatment with the prescribed standard treatment model is performed. In the second study, four South Australian hospitals are compared regarding their treatment of patients with chest pain [120, 141]. Both publications use similar visualizations. On the one hand, they use process graphs of the different version with their changing parts highlighted in color and place them in juxtaposition with each other so it is easy for the reader to recognize differences. On the other hand, they also visualize the metrics they captured, thereby simplifying the spotting of deviations. Similar observations can be made in non-healthcare related sectors, as e.g. the work of [86] shows. The authors' goal is to analyze the effects of the introduction of a modified information system which they achieve in a reproducible manner. One of their keys to success is the development of a visual tool that can be used for comparing the available activities and additional resource data (like e.g. human users) of the legacy system to those of the new system.

Finally, the 2016 work of [63] should be mentioned, as it introduces a new methodology with the sole purpose of comparing CPs between hospitals which shows the high research interest in the topic. The authors also released a follow-up publication, suggesting a common data model that could be used to facilitate cross-organizational process comparisons [64].

2.2.3. Service Mining and Software Analysis

Using PM for analyzing software and software-related processes has quite a long history. One of the first applications of PM in general was in 1998 and had the goal of discovering software development processes based on collected process event information [23]. Later, when software had gotten more complex and often more distributed, researchers used PM to analyze the software systems themselves, e.g. like it was done in 2008 for analyzing and optimizing the deployment of a distributed application to increase overall software quality [59].

A special form of software analysis with the use of PM is service mining which quite simply describes the application of PM to service-oriented architecture (SOA)-based web services [158, 159]. The goal of doing so could be e.g. to describe interactions between services but also to analyze an individual service's behavior or general service performance. Web services are especially well suited for analysis, as they typically record quite extensive trace logs that are used by service developers for maintenance purposes. When abstraction and aggregation techniques are applied to these trace logs, an event log can be generated

that can, in turn, be used for PM applications.

In the early years, service mining focused on SOA-based web services which were often realized using SOAP. SOAP is a protocol for exchanging XML-based messages between services [12]. It was widely used as it provided a standardized basis for developing distributed systems but in recent years has lost a bit in popularity due to the emergence of the REST concept and RESTful web services [60]. An example of this is [163] where events are extracted from inter-service SOAP messages and compared to manually created descriptive models using conformance checking to evaluate expected against observed behavior. Besides the development of new conformance checking techniques, the main challenge of the paper was the extraction of events from the XML-based messages. This is also a major challenge for many other works that evaluated SOA-based web services [79, 134, 161]. Regardless if they were evaluating a proprietary protocol by IBM ("Common Event Infrastructure" [161]), trace data of an SAP system ("SAP NetWeaver" [79]) or some proprietary request log data [134], all works had to pay attention on how to extract the meaningful event information from the XML-based documents describing the service-oriented systems. What the papers also have in common is their goal of either proving that PM is applicable to software and services or to check if the service behavior and architecture were implemented as specified. Additionally, in the most recent publication, not only the services but also the processes themselves are evaluated, resulting in an in-depth analysis with multiple abstraction levels [134].

In the years around 2014, an increasing number of publications were released trying to use the techniques of PM not only to evaluate the runtime behavior of individual services or components but to evaluate human-machine interaction. This results in a kind of bottom-up behavioral analysis which can be integrated into an agile development lifecycle. In [130], a concept for performing such an analysis is presented and its applicability is demonstrated for a large touristic system that is already in productive use and was developed in an agile way. The work integrated the analysis in the agile development process and discovered the great potential for optimizing software processes involving user interaction but also identified the need for more fundamental research in the area. The concept was applied another time in [131] to produce similar results but this time incorporated web access logs provided by a web server to understand the flow of events during a user session. A more generally applicable approach was presented in [122] where the authors also analyzed the behavior of users based on web access logs and therefore introduced a mapping from the technical URLs to the conceptual process events.

With the rise of Web 2.0 and the increasing popularity of RESTful web services, new opportunities arose for the application of PM. With the background of the generation of complete event logs, REST-based web service requests provide the advantage of being self-descriptive in that they transmit all information necessary to understand and process a request within the request itself, at least when properly implemented [47]. This simplifies the event log creation, as the web access log implementation can more easily be extended to contain all information relevant for PM, reducing the number of data sources that need to be integrated. This idea was proposed and demonstrated in 2014 when the

authors also identified a lack of research regarding PM for RESTful systems, highlighting the strong focus on service-oriented architectures as compared to resource-oriented ones [139]. A follow-up work refined the PM by proposing a new discovery algorithm that improves the problem of overly complicated process models by introducing a new discovery algorithm [140]. The algorithm visually clusters a model's activities into intra-service and inter-service ones, thereby better differentiating service behavior from service interaction. One of the latest works tries to simplify the application of PM to RESTful services even more by providing a tool that can mine technical interaction patterns ("RESTful conversations") based on generic web access logs that every RESTful service should be able to generate [71]. The tool provides interactive coloring, additional statistics and support for multiple ways of pattern specification to further enable the use of these techniques by non-PM experts.

2.3. Challenges of Process Mining in Healthcare

Information systems for ACM face many challenges like ad-hoc tasks, case definition changes or a strong focus on collaborative work [61]. Many of these challenges are due to the characteristics of the underlying KiPs that are supported by the system. Therefore, the majority of challenges to ACM also poses a challenge to PM in the healthcare sector as it tries to discover and analyze the same KiPs that ACM systems are trying to support. But some challenges are also special to PM like issues with data quality, manifesting e.g. in missing events or imprecise timestamps. In 2012, two papers have been published that focused on the application of PM in the healthcare sector and the challenges that lie therein [67, 77]. The authors of [77] followed an applied research approach by employing PM techniques for the discovery of treatment processes while observing and describing the challenges that were encountered during the course of the project. In contrast, Homayounfar followed a more formal approach, thoroughly reviewing the characteristics of HISs, CPs and PM before evaluating the existing work to identify the possible challenges [67]. In summary, the issues they identified can be classified as one of two major challenge categories. On the one hand, there is high process complexity, leading to the so-called "Spaghetti Effect" which describes models that are so overly complicated that they are very hard to understand for humans. On the other hand, there are data quality problems that can lead to incorrect process models that do not adequately represent the execution reality In the following subsections, for both categories, the ramifications for this work will be described in depth before an analysis of the different causes for each challenge will be done that explains the context, illustrated by some real-life examples.

2.3.1. The Spaghetti Effect

The first and biggest challenge to PM in healthcare is the high process complexity that will, if not accounted for, lead to overly complicated process models also called "Spaghetti

Models" [44]. They got their name because they contain an excessive amount of transitions between their activities, leading to large, chaotic models that look like a heap of spaghetti and are too difficult to comprehend [150]. The effect can very easily be seen visually and is illustrated in figure 2.1 that shows a process model of the case evaluation stage as executed during the first case study in Lleida. The depicted model is a directly-follows graph that was created using a fuzzy mining algorithm at the full level of detail and is based on 610 recorded process events describing 35 cases. A transition in the graph means that a certain activity was observed to have directly followed another activity. This, in turn, means that the overly complicated model is not a problem of the mining algorithm but actually reflects reality as it is directly caused by the complex and varying process that was executed.

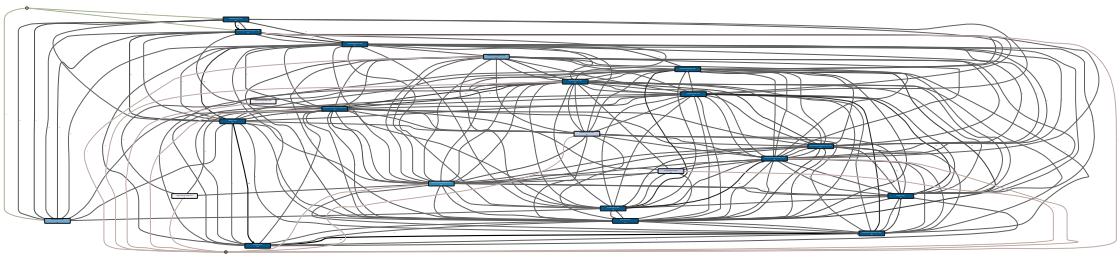


Figure 2.1.: The spaghetti effect when discovering models of unstructured processes

In fact, such high complexity is actually expected for KiPs like they are found in the healthcare sector [28]. However, as the issue primarily applies to unstructured procedures and some parts like the initial patient registration are well-structured and always executed in a similar way, the expected complexity is not equal for all stages of a typical treatment process. The degrees of structuredness range from *structured* over *structured with ad hoc exceptions* and *unstructured with pre-defined segments* to *unstructured*. This grading is also called "spectrum of process management", was explained in section 1.2.1 and can be seen in figure 1.2. Processes in the healthcare sector range over the full spectrum [28, 29]. But even though it is much easier to generate comprehensible models of structured processes, there is still a strong incentive to cope with the spaghetti effect as less structured processes tend to offer bigger opportunities for improvement [150].

To mitigate the extent of the spaghetti effect, the complexity of the underlying process has to be reduced. In order to achieve this, first, the root causes of process complexity in the healthcare sector need to be understood, of which there are four main ones [29, 67, 77]. The first reason is the high heterogeneity of processes due to changes in the execution context [28]. Apart from that, modifications in the environment (i.e. changes to the process definition) pose a big challenge as they lead to event logs containing historical data that may not fit the current process model [67]. Additionally, during the execution of KiPs, human deciders dynamically shape the performed process based on unpredictable input data. This leads to a high amount of process flexibility which makes processes even more complex [28, 77]. Another driver of process complexity is the strong focus on interdisci-

plinary collaboration that exists in the medical context due to individual cases typically requiring the expertise of professionals from multiple domains [67]. While this and two of the other reasons apply to KiPs in general, healthcare processes still take a special position as they combine the highest amount of challenges that can be encountered when applying PM [29]. In the following, the four mentioned main causes will be explained in detail.

Cause S1: Process Heterogeneity

In the healthcare sector, process complexity is even higher than for most other areas where KiPs are encountered [77]. Amongst other reasons, this is due to the high variability between process executions. This is caused by changes in the execution context caused mainly by high patient heterogeneity [67, 112]. This heterogeneity affects the therapeutic effect of certain drugs and treatments due to age, gender and genetic differences in general, but also due to lifestyle issues like physical fitness. Patients also have a medical history so e.g. commonly used pain medication could be less or even ineffective. Allergies and personal preferences or prejudices could exist making certain treatments impossible. A lack of proper health insurance could prevent high-cost treatments and can generally affect process execution. Also of relevance are the location of the hospital, the person in charge, the teams' domain knowledge, the current hospital occupancy, the number of people waiting for patient admission, etc. All of these factors can influence process execution. Even cultural aspects like how much overtime a professional can do or does on average can affect process executions. This makes it overall hard to find patterns, similar process variants or even a standard treatment guideline.

Cause S2: Process Definition Changes

Another reason for highly complex processes is modifications in the environment, resulting in changes to the process definitions themselves [11, 67]. This applies especially to processes that are supported by ACM systems which require a formal model of the processes being executed and the possible activities they consist of [29]. If the environment is modified, e.g. because a new treatment is discovered and introduced that was not available before, the formal model describing the possible activities needs to be updated. This is also the case if a simple typo was found or an activity is removed, e.g. because a treatment was discovered to be ineffective so it should not be executed anymore. If not taken into consideration, the formal model's updates can lead to execution logs that describe real-world behavior that seemingly violates the defined case model even though the observed behavior was model-conform at the time of execution.

As healthcare is a field with high research activity, the formal models and clinical guidelines need to be changed quite often, as new treatments are frequently becoming available [67]. But there are many other reasons leading to process definition changes. Treatment processes are being standardized and continuously improved across regions, countries and continents to enable comparability and improved means of quality assurance

or ensure certain service quality levels [114, 119]. New paradigms like evidence-based medicine or CPs emerge and influence how processes are structured [69, 114, 119]. Newly introduced or updated legal restrictions can enforce or forbid certain treatments. Examples include the obligation to help people seeking medical treatment ("duty to rescue"), regardless of their financial or social situation but also the ban of genetic modifications to humans that both exist in many countries [110]. Decisions of third-party stakeholders like insurance companies can also affect process definitions – intentionally or unintentionally – through financial incentives by paying, not-paying or over-paying for certain treatments. Apart from that, the high cost pressure can also affect treatments and render certain activities impossible due to lack of economic viability. The same applies vice-versa when the efficiency of treatments is improved. Even trends in society can affect processes in healthcare, as the cultural acceptance of smoking shows, which was once accepted in hospitals while in many countries it is now banned from all public institutions [9]. Lastly, the introduction of new supporting IT systems can also affect how processes are defined, e.g. due to lack of technical capabilities, newly added features and modified behavior in general.

The large amount and thematic diversity of possible occurrences that require process definition changes are characterizing for KiPs and especially for those in the healthcare domain.

Cause S3: Flexible Process Execution

As just mentioned, processes in the healthcare sector tend to change more frequently than in other industries. Additionally, often ad-hoc adaptions are necessary during process execution due to unexpected and unpredictable events that occur [11, 67]. For example, hidden diseases or comorbidities could exist that are only uncovered when the case is already running, making the actual process and its complexity evolve dynamically. This is also expected for KiPs as they describe a class of processes which are executed under uncertainty by heavily relying on human decision makers employing their domain knowledge to shape the executed process and dynamically react to unexpected events [28, 29]. This leads to a high variance regarding the executed activities, raising the overall process complexity.

But apart from those unstructured parts of the process, more well-structured activity sequences exist as well and a typical medical case features the full range of process structuredness [28, 29]. To illustrate this, imagine a patient with certain acute condition seeking help in a hospital. When the patient is identified the first time, the process is usually *structured* with the registration nurse giving each patient the exact same questionnaires regarding their personal data, current medications or contact persons. In rare cases, for example when there is a medical emergency, *ad hoc exceptions* might be necessary, but this is a rather rare occurrence. The process for the initial evaluation is partly structured, as some tests might or might not be necessary depending on the results of earlier tests. In other words, the process is *unstructured with pre-defined segments* that describe the dependencies that exist between the available test activities. Afterward, the diagnosis and workplan phases

follow which are highly dynamic and flexible as they might feature repetitions, changes or adaptions, depending on the course of the treatment and the occurrence of unexpected events. If necessary, the evaluation stage could be repeated intermediately in order to compare it to previous results. If the expected outcomes are observed, the well-structured discharge procedure is executed, consisting of feedback questionnaires and some legal forms.

As you can see, all types of structuredness can easily be combined in a single case, resulting in an overall treatment process that is very complex and hard to understand for humans. When PM is applied without accounting for this, spaghetti models are generated that do not provide much insight and bear little added value [28, 29].

Cause S4: Interdisciplinary Collaboration

The last reason for high process complexity in the healthcare sector is the high degree of interdisciplinary collaboration that is required in many treatment processes [67]. Often, multiple stakeholders with possibly diverging goals need to work together on the same, single patient. This is due to the knowledge-intensive nature of healthcare processes that require experts from multiple, different domains to collaboratively solve complex problems [29]. This can create conflicts of interest, e.g. when a physician is treating a certain disease using a drug that has side effects when combined with some anesthetic and the patient needs to undergo a surgery [67]. Now, either the anesthetist or the physician is forced to diverge from the traditional treatment behavior to prevent unwanted side effects. The issue becomes more significant when primary care is taken into consideration and clinical measures against acute conditions can conflict with the treatment of chronic diseases often done in primary care.

Apart from that, hierarchies and complex social networks exist in hospitals and medical institutions at large, featuring department heads, medical superintendents, specialized physicians like radiologists or surgeons but also case managers and nurses [29]. These hierarchies and networks influence process executions due to social effects like people starting tasks earlier that were assigned to them by their supervisor or the sickness of certain specialists requiring temporary changes to medical procedures. Therefore the handover of work within a certain case can easily get very complex. Again, this complexity is even more an issue when primary care is taken into consideration, as this causes the number of involved people to rise rapidly.

2.3.2. Data Quality Issues

Many original studies, as well as multiple literature reviews, have found that another big challenge for the application of PM techniques to healthcare processes is the quality of the event log that serves as input data to the discovery algorithms [11, 99]. Of course, being a discipline that is related to data mining, this is not a very surprising circumstance. Issues related to data quality can be further split up into problems with incorrect or insufficient

logging, the abundance and simultaneous redundancy of data and issues due to the data extraction from multiple, different systems [67]. In the upcoming paragraphs, these three areas will be defined and examples will be given for each.

Cause D1: Incorrect and Insufficient Logging

Earlier process discovery algorithms, like the α-algorithm or the region-based process miner, required exhaustive input data completely noise-free [87]. This meant that no events could be missing and all data properties of each event had to be provided. If not the case, they either failed to generate a model at all or came up with a seemingly correct model, that had logical inconsistencies and did not reflect the reality on a closer look. Newer algorithms like the fuzzy miner do not bear this problem anymore and can generate correct models from noisy or incomplete data, but the quality of the resulting model and the additional insights it grants are still vastly dependent on the quality of input data [11]. At the same time, many problems regarding data quality are due to documentation errors originating from data that was entered incorrectly by humans [67]. Therefore, many potential issues like missed events and missing or incorrect activity names are already solved by modern case management systems that have largely replaced manual user input with automated data collection and activity logging.

An issue that persists is the capturing of imprecise data. A typical example is wrong timestamp data that is being recorded due to slightly differently running clocks in individual sub-systems of a distributed HIS. If the imprecision is only in the range of hundreds of milliseconds, then the error might be hard to detect for humans. Even if it is detected, a human could decide that the issue is not worth fixing, due to the low relevance of a deviation of a couple of milliseconds compared to human processing speed. However, for a PM algorithm such imprecision can further increase the already high complexity of healthcare processes but confusing the sequence of activities that are otherwise ordered, making it even more difficult to understand the resulting model. Another example of imprecise data are timestamps that are captured too coarse for various reasons. Sometimes this is done in administrative systems where, due to space and cost restrictions or due to humans inputting the data, only day-based timestamps are recorded. This leads to a heavy loss of precision for performing PM but can also lead to loss of information, when changes that occur within one day are hidden as only the delta between two days is being recorded.

But even after ruling out these issues, there are still situations where data can be missing or wrong data is recorded, e.g. due to technical or usability issues with the HIS or due to human errors when creating the specifications and models that drive these systems. Apart from that, logging is not always as hard a requirement as it is when legal restrictions enforce it, e.g. for traceability reasons. Often, its primary use is for debugging the software system in case of issues in a productive environment. This can create incentives to implement a less reliable logging system that works in most cases and especially in those relevant to the developer but occasionally drops a log message to avoid impacting the overall system performance. The same applies when the software is changed and

maintained, resulting in the constant need to monitor data quality in order to keep it at an appropriate level.

Cause D2: Abundance and Redundancy of Data

Another issue often causing data quality problems is data abundance coupled with the redundancy of data [11, 67]. Most HISs are distributed systems consisting of multiple subsystems [62]. In order to keep performance high, prevent loss of data, provide high system availability and increase failure safety, copies of data are stored in multiple places and key information is distributed across many services. This is a crucial property of HISs to ensure the overall system can keep working even if some of its subsystems fail which enables clinicians to still access critical patient information in such a case. The main drawback of this solution is that changes to the redundant data need to be synchronized across all components storing the data. If this is not done properly and no versioning system for data exists, issues regarding data inconsistencies will arise. In turn, these can lead to outdated process models being generated from freshly extracted data [11, 67].

Apart from the data redundancy, the abundance of data in a typical HIS can lead to problems as well. As explained in section 1.2.3, HISs consist of many different types of subsystems [62]. Typically, all of these systems have some logging capabilities and produce a certain amount of data when they are used. The high count of individual subsystems and their different level of abstraction and granularity is challenging as data needs to be aggregated and filtered to get a single dataset with relevant information on the same level of abstraction. For example, a HIS may contain a clinical administration system that is used for manually documenting the general services that certain patients received. At the same time, a more specialized system supporting patient treatments in an intensive care unit could protocol the vital signs of patients as well as the exact medications they took in. Very low-level events are often captured as well, especially in large medical devices that are computer-controlled. For example, a robotic x-ray scanner featuring a movable arm that can be controlled by humans through a computer interface could log all the motor movements in addition to the user input for legal protection and traceability reasons. Clearly, the three given examples log data at vastly varying granularity and abstraction levels.

In order to enable PM, the available data needs to be integrated, aggregated and filtered [11, 67, 77]. This needs to be done in an efficient way so the HIS performance is kept at roughly the same level and the work performed in the hospital is not impacted by the procedure. At the same time, attention has to be paid to extract the correct version of the data and prevent the generation of outdated models.

Cause D3: Cross-System Data Extraction

The last issue with a possible effect on data quality are complications related to the extraction of data from multiple components and across system borders [67, 99]. This is neces-

sary because of the mosaic-like nature of many HISs resulting in a situation where not all data is present in every subsystem which in turn requires the extraction of data from different systems. Each system might use varying means of representing data conceptually, apart from different protocols and technologies being employed to store data. This heavily increases the complexity when extracting the data, as all employed concepts need to be unified and a common conceptual data model needs to be developed that can then used to perform PM and deviate meaningful insights. Creating the unified data model is a difficult process that requires experts in modeling as well as in domain knowledge which increases the costs to apply PM techniques. As the healthcare sector faces high cost pressure anyways, this further increases the inhibition threshold for introducing PM-based evaluation techniques.

The work of [96] tries to mitigate the problem by developing a reference model for healthcare data that they propose for broad use. The developed model is intended to be used both in clinical and primary care settings for data representation purposes but also as a common basis to facilitate PM efforts. It exhaustively lists all possible types of data that may be captured during the medical care process of a patient, regardless of the treatment duration. However, while it supports transforming the data after it was extracted, it does not directly influence the extraction itself which could still require the integration of different technologies and protocols.

Still, the impact of the extraction problem itself is expected to become less burdening in the coming years due to increased research and development efforts regarding a standardized way for exchanging information between HIS components but also between different HISs. This was the conclusion of a recent literature review that was conducted in 2017 and evaluated the knowledge representation in the common data model of the *Observational Medical Outcomes Partnership* [129]. The study also reviewed the *Health Level 7 Fast Healthcare Interoperability Resource* standard that can be used to exchange information between HISs. As soon as a critical share of hospitals has adopted the mentioned standards, evaluation efforts can focus in these technologies and protocols which will greatly simplify the data extraction problem by in most cases obsoleting the need to extract data from multiple systems.

Part II.

Concept and Implementation

3. Approach

This chapter will describe the approach that was taken for this work and the main concepts that were employed for this purpose. It will first give details on the challenges mentioned in section 2.3 with a focus on how they impacted this project and how they were mitigated. Afterward, the methodology that served as a foundation for the execution of this study and the adaptions that were necessary to account for special circumstances will be presented. Then, it will give an overview of alternative methodologies and briefly list their shortcomings regarding the scenario at hand.

3.1. Mitigating the Challenges

Of course, the challenges identified and described in section 2.3 need to be addressed to prevent negative influences on the model quality and the final PM analysis While some of the challenges are mitigated by the employment of the CONNECARE system itself, others are not and require dedicated countermeasures that need to be incorporated into the formal approach of this work. This is what the upcoming section is about. In the following, different means of mitigating the individual challenges will be presented. Apart from that, the general improvements achieved by using the CONNECARE system will be described and also put in context with the challenges identified earlier.

3.1.1. Preventing Data Quality Issues

A big factor influencing the PM procedure and the quality of the resulting models are issues with data quality. As explained in section 2.3.2, many of these issues are mitigated by the use of CONNECARE, specifically in its capacity of an ACM system but also in its role as an integrated care environment. Apart from that, the only way for clinical professionals to access the service is through its central JSON-based API. This provides further advantages regarding data quality, as the interface adheres to principles of REST, which in turn facilitate the use of PM techniques by enforcing e.g. self-descriptive messages and consistent, well-structured identification of available resources. Additionally, a log analysis system will be used for recording and preprocessing the log files in a continuous, near real-time way. The log system will also feature a web frontend that can be used for continuous log inspection and event data analysis to ensure a once reached quality level can be upheld across a longer timespan. The following sections will explain the mentioned aspects in more detail, in order of appearance.

Evaluating an Integrated Care Environment

CONNECARE is a modern and highly automated HIS subsystem that provides an integrated care environment. It automatically captures many data points regarding the activities that were performed by users, amongst others e.g. the name of the activity, the case it belongs to, by whom it was finished, who was responsible for the execution and the completion timestamp. Additionally, CONNECARE's backend, SACM, is an ACM system and shares most of the commonly found properties.

The most fundamental one is the centralized case management that all ACM systems feature. The main benefit of centralization of patient/case data into a single system is that all professionals working on a case have full and easy access to all relevant data, not only to the data that was captured by their department. This enables improved collaboration as it allows professionals to better coordinate their treatment efforts, avoids unnecessary patient checks and reduces the chance for unwanted side effects. From a PM perspective, the centralized approach is also quite promising because, in order to analyze treatment processes, data only needs to be extracted from one system. An implication of this is that only one, consistent data model has to be integrated and it is furthermore quite likely that this can be done using only one protocol and only a limited number of background technologies.

Apart from that, central data management also solves some of the issues with data redundancy, as the main source of truth is now the central ACM system that contains the data storage. This forces all other HIS subsystems to synchronize their case-relevant data to the ACM system. Even if some more specialized systems may choose to keep local copies of the data for performance purposes, they are still forced to push the data to the central ACM to ensure all professionals working on the case can access it. In turn, this enables users employing PM techniques to focus data preparation efforts on the preprocessing instead of having to integrate multiple data sources with possibly different protocols just to extract the event data.

As mentioned in the beginning, CONNECARE also provides an integrated care environment. This means that data of medical devices like e.g. fitness trackers is captured and fed into the central SACM backend component. By monitoring all boundary interfaces of the component, the complete data entering the system can be gathered and persisted for later PM analysis. This is especially simple in the case of CONNECARE because it only provides a single interface to the SACM which is its JSON-based RESTful API.

Mining RESTful Web Service Logs

When modern, HTTP-based web services are operated, usually, HTTP access logs are created and stored for some amount of time. This is done for various reasons, the most prominent ones being security concerns and performance analysis reasons. When the web service is connected to an application server upstream, logs can additionally often be used for debugging the application as they contain the data that triggered certain server action

and that is needed for reproducing issues that occurred. Typically, data captured about a request contains at least the source IP address, timestamp, processing time, HTTP method, requested URL, HTTP status code, user agent and some authentication information. This is at least true for the default configuration of the Apache and nginx web servers which together make up around 85% of all web servers used today [169].

The data is very easy to collect in a complete and noise-free fashion, as it can be provided in a standardized way by the web server and does not require a custom implementation like it is the case for an application server. Apart from that, it can also be enabled after the development of the application itself has finished, through the use of an additional web server that is configured as a reverse proxy and forwards requests to the application server. This way, centralized logging can be enabled through a simple configuration change of the reverse proxy which in turn reduces the requirements to the logging capabilities of the application. Fewer requirements generally simplify the development while leading to cost savings and at the same time ensuring complete, noise-free and high-performance logging that captures all interface interactions. If all interfaces of a system are web-based, this enables full monitoring of all system interactions at low or even no cost, as many operational deployments will already have access logging enabled by default.

As RESTful messages need to be self-descriptive, all information required for processing the request is explicitly included. PM also benefits from this circumstance, as the event already contains all data that is relevant to it. However, due to performance, data protection and storage restrictions, logging of the full request and response bodies is often disabled and only the minimal information mentioned above is captured. Using the data from the bodies, a complete reproduction of an observed interaction workflow would be possible. But even if logging of the full request and response bodies is disabled, thanks to the RESTful architecture, the captured access logs are still highly relevant and contain much valuable data. A large part of it is encoded in the requested URL, as the REST principles enforce URLs to contain a consistent, well-defined, hierarchical structure that can be used for addressing individual resources or for filtering for specific request logs. In conjunction with the HTTP method that specifies if the request was a read or a potential write operation, the obtained information can be used to apply efficient, multi-layered filtering for reducing the amount of data that needs to be persisted.

Other information like the identifiers of the entities that are modified are usually contained in the URL as well and can simply be extracted from there instead of performing a more expensive lookup. The user that performed a request can also often be determined implicitly by parsing the request's authentication information that is required in personalized systems for security reasons. Alternatively, the source IP address can be used which has some drawbacks like the inability to identify individual persons that used the same device to access the system. On the other hand, IP addresses allow for geographical analyses up to a certain degree of accuracy which is not easily possible otherwise. Alternative user identification techniques like e.g. fingerprinting based on the supplied user agent are also available and there is a high probability that this situation will continue as the online advertising industry needs these techniques for tracking users and sessions.

Preprocessing Events in Near Real-Time

In order to ensure that a certain level of data quality is sustained, the data needs to be checked constantly so that degradations are detected and can be mitigated. Apart from being tedious and time-consuming, repeated manual quality evaluations are an additional cost factor that should be avoided. Additionally, a domain expert is needed for the analysis, as the exact definition of quality depends on the current use case and the insights that should be generated through the PM application. Therefore, to keep costs manageable, the manual process needs to be as quick and easy as possible and the expert should be able to perform it at any point in time.

The usage of a modern log analysis application like graylog[1], loggly[2] or the elastic stack[3] facilitates this. It enables the creation of one or more dashboards that provide all information necessary to assess the overall data quality in a limited number of views while also offering possibilities to deep-dive into the data for investigating possible issues. Often, a frontend is provided for creating the dashboards, so this can be done by domain experts as it does not require software engineering skills.

However, in order to yield reliable results, data quality needs to be assessed after the preprocessing has been finished. Otherwise, issues that are eliminated or introduced by the preprocessing step might lead to invalid conclusions. This requires the step to happen in near real-time so that the expert's ability to assess data quality at any point in time is not limited. To achieve this, the preprocessing should also be done by the log analysis application which is designed for this purpose of gathering and manipulating a huge number of log entries like they are produced in many business and web applications.

Apart from that, filtering the logs is also quite performance demanding, as typically a high number of log entries has to be parsed and discarded before a request that is relevant to PM is found. This is due to the fact that many requests have no effect on the process to be analyzed like e.g. most GET requests that should be read-only according to REST principles. To give an impression, the quota of relevant to non-relevant requests for the CONNECARE system during the performed case studies was roughly 1:350. At the same time, on average 1 300 requests arrived per hour with a peak value of 36 000 hourly requests (or around 10 requests per second over the period of one hour). To enable the processing with such a high throughput, the system needs to be implemented in an efficient, high-performant way and provide good abilities for scaling horizontally and vertically. Most established log analysis applications, like the ones mentioned before, fulfill these requirements. Using those not only enables near real-time preprocessing, but the tools can also be employed for operational analysis, the purpose most of them were initially designed for.

[1] https://www.graylog.org/
[2] https://www.loggly.com/
[3] https://www.elastic.co/

3.1.2. Dealing with the Spaghetti Effect

If not accounted for, the challenge of high process complexity can lead to complicated models that are hard to understand for a human due to the spaghetti effect. It is caused by the complex, varying processes like they are found in the healthcare sector and is characterized by large models with many elements and even more transitions between them. Fernandez-Llatas et al. dedicate an own section to the problem where they describe means of reducing the spaghetti effect or at least mitigating its influence on process understandability [44]. The measures that will be employed during this work will be described in the following sections.

Fuzzy Mining

One of the biggest influences on the generated model's complexity is the mining algorithm used for process discovery. The reasons for this high complexity are two assumptions that do not hold for flexible, real-life processes like KiPs [58]. The first problem is that traditional mining algorithms assume that the event log is completely noise-free and all events are on the same level of abstraction. This is not true for many use cases, as most logs are not homogeneous but contain events that lack some kind of information or lack some kinds of events at all. Also, the event log usually contains entries with a different level of abstraction. The second assumption is that the logs exactly reflect a well-defined process. As explained in section 1.2.1, this is also not the case for KiPs. Based on these assumptions, traditional algorithms like the heuristic miner aim at discovering complete, precise models that correctly describe all executed processes and can ideally be used for executing the process using a workflow engine. But because of the high complexity of KiPs, this leads to equally complex process models that do correctly describe the observed behavior but provide little insights as they are too difficult to understand. Consequently, traditional discovery techniques are well-suited for descriptive but not for explorative modeling of flexible processes.

These findings were described by Günther and van der Aalst in the course of a paper and a follow-up dissertation that introduced the concept of *fuzzy mining* to improve the results of PM in flexible environments [56, 58]. The basic idea is to approach the issue of high model complexity with techniques known from cartography. The field faces quite similar challenges as it needs to simplify the highly complex and unstructured topologies of the real world into maps that can be understood by the average reader. To achieve this, four different techniques are available and commonly applied during the creation of a map. The figure 3.1 shows an example of a motorist road map that was created using all four concepts that will be described in the following.

Abstraction To improve the understandability of the map, the contained information needs to be reduced to an appropriate amount. To achieve this, insignificant information is intentionally left out. To give an example, in the case of a motorist map, this information could be bicycle ways or hiking paths.

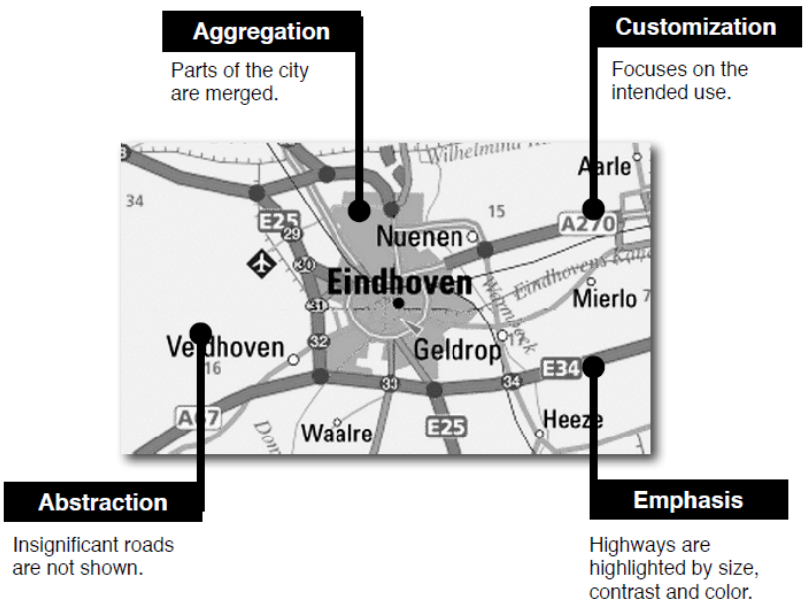


Figure 3.1.: Example of a road map, showing the concepts available for complexity reduction and how they are applied. Taken from [56]

Aggregation Often times, similar map elements close of each other are aggregated into one homogeneous cluster, reducing complexity. Examples include multiple houses being visualized as a single colored area or an airport that is displayed as a single icon instead of a collection of runways.

Customization As the real world is complex and topological features are of varying interest to different people, maps are often designed with one specific group of persons in mind. Depending on the intended use of the map, different elements can be included in or excluded from the map. For example, a motorist map will not include bicycle ways, while a map intended for bicyclists could even visualize hiking paths. This often correlates to the map's scale which to some degree determines the level of abstraction. A naval map, on the other hand, will focus on a completely different set of topological elements and abstract away most of the land-related features. All kinds of maps could include additional information like the identifier of a highway or the water depth at a certain point to make the map better suited for its intended purpose.

Emphasis Most maps use colors, enhanced contrast and increased sizes to highlight map elements that are of special importance. In the case of a motorist map, highways will most likely be larger and of a different color than they are in reality so that a user of the map is quicker in finding the relevant information.

In the context of PM, a location on a map corresponds to an activity in a process model, with the roads between them being comparable to connections in the model. The fuzzy mining algorithm uses the two metrics *significance* and *correlation* to decide on the degree of abstraction and what should be aggregated.

Significance measures the relative importance of an activity or a relation between two activities and is used for hiding graph elements that occur at a low frequency. This assumes that a reader is more interested in the frequent behavior a process model represents than in edge cases and exceptional events.

Correlation, on the other hand, describes how closely related two events following each other are. The metric is not concerned with the activities themselves but only evaluates the directly-follows relation between two events (i.e. activity instances). It is determined based on the similarity of activity names or on the overlap of additional data attributes. Closely related events are assumed to belong conceptually together or to represent a common sub-process which is why they can be aggregated into a single activity cluster. The resulting clusters can also be seen in figure 3.2 where they are represented by the green octahedra.

Based on these metrics, the fuzzy mining algorithm uses the following approach to generate understandable process models that avoid the spaghetti effect:

- *Highly significant* elements are preserved, as they are important for understanding the frequent process behavior
- *Lowly significant* elements with *high correlation* are aggregated into clusters representing sub-processes
- *Lowly significant* elements with *low correlation* are excluded from the final process model

In addition to this simplification, an emphasis is employed through a color grading that is used to indicate the relative frequency of how often a certain activity or the relation between two activities has occurred. Apart from that, the plugin developed in [58] that implements the fuzzy mining algorithm offers some interactivity for setting the significance thresholds that indicate whether a certain model element should be displayed or not. This allows for the creation of customized models and will be explained in more detail later in this section.

Mapping API Access Logs onto Process Events

Another way to mitigate the spaghetti effect is to reduce the complexity of the input to the discovery algorithm by splitting up or shrinking the event log. The former is typically called *clustering* and will be explained in a later subsection. The latter, reducing the event log size, can be achieved by mapping the low-level API access logs to more high-level process events. This is commonly known as *event mapping*. Apart from the provided complexity reduction, mapping techniques also offer semantic advantages when trying to

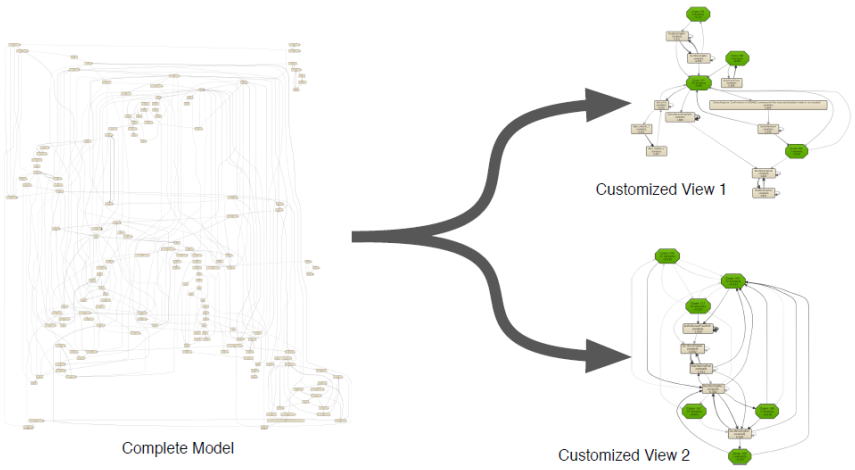


Figure 3.2.: Example of a spaghetti model and two customized fuzzy models that were generated from it. Taken from [56].

understand the generated process model as the level of abstraction where the case is modeled is typically higher than the level where events are recorded. Studies in recent years have shown the topicality of the approach and have proven that it is able to improve the understandability of generated process models [46, 92].

The authors of [46] developed an automatic, generic algorithm for finding a mapping from low-level to high-level and demonstrated its usefulness by applying it to a big event log of a car manufacturer's incident management system. Using their algorithm, they managed to create a high-level model that correctly represents all low-level behavior without requiring the input of any additional information. However, this is not necessary with the context of this work, as the API of the SACM system is RESTful and therefore well-structured in a hierarchical way which can be exploited to deviate the mappings needed for event log reduction. For this reason, an approach similar to that of [92] will be taken. The authors use a knowledge-based ontology that is created in advance using expert knowledge for mapping the events according to their activity name. Afterward, the mappings are applied to a medical dataset from the

field of stroke management and results suggest that performing the mapping leads to clearer process models that are easier to understand as unnecessary details are left out. But still, proper integration into existing PM framework and the formal proof needs to be done. Apart from that, the authors also developed an algorithm that can be used to apply the mapping, but it only maps ordered sequences of low-level events onto higher-level ones and takes extra steps to find the correct parent of ambiguous low-level events based on the execution context.

Both of these properties are unwanted and unnecessary for the evaluation of the CONNECARE case studies. In addition, the top-level structure of the studies is known, was

explained in section 1.2.7 and is illustrated in figure 1.5. It basically consists of the four stages *case identification*, *case evaluation*, *workplan* and *discharge*. Each stage again consists of a fixed amount of executable tasks that can only be executed in the context of the stage. Each task has a lifecycle as well which can be controlled using the endpoints that are available for this purpose on the API. They include actions like *activate*, *get*, *draft*, *complete*, *terminate* and *alert*. Additional endpoints exist (e.g. *enable*) but are rarely explicitly called by humans and will be excluded from this evaluation. The different events on each abstraction level and their relation are shown in figure 4.2.

To make use of this hierarchy, a static mapping will be employed that maps low-level API accesses to a higher conceptual level. During this process, read-only actions, as well as write actions not affecting the task's lifecycle, are discarded, while all events that do affect the lifecycle are aggregated into one consolidated event that contains all relevant information extracted from the different low-level events. This reduces all events related to an individual task instance to one high-level event, reducing the number of events per task from thousands to a single one. Being in the focus of the second research question, alerts are an exception to this logic and passed on to higher levels without modification to enable the evaluation of their influence on the process execution.

Providing Views at Different Levels of Abstraction

To further break down complexity, the consolidated event log can be exported at three different levels of abstraction to allow for more or less detailed views on the same event log. In the order of degree of abstraction ranging from high to low the available levels are *system view*, *case view* and *stage view*. The abstraction level, the contained elements and their use during the evaluation will be explained in the following.

System View The system view is the most abstract view of the recorded events. It does not differentiate them at all and treats events from both case studies as well as from all treatment sites as if they were equal. Tasks are not identified by the actual activity that was performed but rather by the stage they belong to. Different tasks within a stage are not differentiated. This results in an abstract view of tasks as *some activity performed in a certain stage*. All case definitions are also treated equal and no respect is paid to version changes or differences in the fundamental case model design (e.g. the split workplan stage in Lleida's second case study is ignored). The view included all available event types and is used to analyze the overall feature utilization and the structuredness of the system usage as a whole. It results in one process model when visualized.

Case View For each site, the case view shows the behavior recorded for a certain case model version and all compatible versions. It is similar to the system view in that it includes all event types and does not differentiate between tasks within each stage. However, it represents the stages according to the actual case definitions, meaning Lleida's second case studies has one more stage than the other executions. It can be

used for evaluating feature usage and its impact on case execution but also for comparing how the system was employed at different sites and if the two case studies were executed differently. When the visualization is generated with high precision (i.e. little to no graphical elements are excluded), it can be used to judge the flexibility applied during case executions. However, as the tasks of a stage are treated equal, it can only be used to evaluate the overall case flexibility, i.e. the flexibility *between* but not within stages. Nonetheless, this is a useful feature as such flexibility is expected to occur according to the fundamental study design described in section 1.2.7 and depicted in figure 1.5. The view results in six process models when visualized, one for each site and case study combination, plus an extra one due to non-backward-compatible changes.

Stage View The most detailed view on the consolidated event log is provided by the stage view which only shows activities *within* a stage. It is restricted to the event types of tasks and alerts as other events cannot be related to an individual stage (i.e. notes and messages are a property of the case instance, not of the stage instance). The stage view provides insights into the flexibility within a certain stage and can be used to evaluate if stages have really been executed as dynamic as they have been modeled. Therefore, it needs to respect changes that are breaking the stage compatibility from one case model version to the next. It can be used for evaluating the first research question but not the second one, as not all features are included in the view. For answering the third research question, this most detailed level of abstraction is used as well, but it is viewed from the organizational perspective. When visualized, the view produces one process model for each combination of stage, case study and site, accounting for 24 models. With the addition of stage versions with non-backward-compatible changes, roughly 35 process models are produced in total.

The different views and their relations are also depicted in figure 4.2, however, the illustration includes the task view which is not available because task-related events are aggregated during preprocessing. Additionally, it lacks the system view which simply ignores the location where the study took place and enforces the four stages from the fundamental study design introduced in section 1.2.7.

Clustering Data

Data clustering can be quite broadly defined as the general task of grouping objects similar to each other [8]. It is the main task of exploratory data mining but has also been employed in many other fields as it is a general approach for discovering knowledge in big data sets. Many algorithms like k-means or DBSCAN exist for clustering data based on certain properties [39, 93]. Clustering has also successfully been applied to PM where it was shown to reduce the complexity of the resulting models [27, 95, 137]. This is mainly due to the fact that it splits up the initial dataset into smaller ones that are more homogeneous which results in more but less complex process models. More specifically, it was applied

for discovering different CPs based on large datasets of observed behavior. The use of a clustering algorithm allowed the classification of the data into patient groups without the need to first understand the similarities and differences amongst the patients. Afterward, a process model could be generated for each dataset, resulting in understandable models.

However, no algorithmic approach with general applicability will be taken in this work, due to the comparably low number of study participants (n=210). Instead, a two-stage classification of cases into fixed, pre-defined clusters will be done. The first classification stage assigns the cases into clusters based on the site where the treatment was performed, resulting in three clusters, each corresponding to an individual site. This is done to prevent additional complexity being induced by social and cultural differences from one country to the other. As a positive side effect, the clusters can be compared to each other to get insights about whether the sites use the system differently.

The second classification is actually somewhat dynamic, as it changes based on the abstraction level of the process model to generate. It considers the exact case definition version and assigns cases into clusters with versions compatible with each other. This is a necessity to ensure that execution traces are always valid representations of the modeled CPs, even if the model changed in a non-backward compatible way. The compatibility analysis is required because otherwise too many clusters would be created leading to very small datasets not suited for PM. However, which case models are compatible depends on whether the system view, the case view or the stage view is chosen. This is because changes affect varying parts of the model and could depending on the current view be irrelevant at all. A simple example of this are changes that only correct typos in clinical questionnaires and do not affect the execution of the process at all but still lead to an incremented version number. If the change instead reduces the available tasks inside a stage but the interdependencies between stages do not change, then the new version is not backward compatible when in the stage view, as it models the activities inside the stage. As less are available, some older execution traces could be invalid in the updated context. However, when in the system view, the new version is still compatible with the old one, as this abstraction level ignores the activities within the stages and only focuses on inter-stage actions. An exhaustive listing of all case model version and their compatibility based on the desired view can be found in section C.3.

Using Interactive Visualizations

As mentioned earlier this section, customized visualizations provide many advantages over their generic equivalent, especially to domain experts. However, depending on the nature of the evaluated KiPs and the questions that should be answered, many different customizations may be required [57]. On top of that, answering certain types of questions in a dynamic and highly complex environment may lead to new questions that also require new, differently specialized visualizations. The domain experts involved in the evaluation of the customized models need to request the new or updated visualizations from a PM expert who then needs to generate them. This leads to increased costs and the repetitive

nature of the procedure intensifies the problem.

To mitigate the issue, *interactive visualizations* can be used [30, 44, 57]. They enable the PM expert to generate each model only once and decide for an initial, customized view that is then handed over to the domain expert. When new questions emerge that require different customizations, the domain expert can simply modify the existing customized model in order to fit the new requirements. This reduces the need for a PM expert in later stages of the process evaluation. At the same time, it allows for advanced customizations as more domain knowledge is available when generating the customized visualizations.

While the exact features that make up the interactivity vary based on the implementation, a common set of customizable parameters can be found in most software that supports visualizing complex processes. For models generated using a fuzzy miner, usually these types of interaction are available:

- Customization of the level of abstraction (i.e. significance threshold)
- Customization of the level of zoom
- Ability to incorporate additional metrics into the model (e.g. performance indicators)
- Display of a reduced amount of information with the ability to view all available data when hovering over an element with the mouse
- Filtering for or excluding cases with certain data properties enables drilldown and drillup techniques
- Visualizations of the raw event data allow further investigations of the event sequences leading to certain model behavior

An additional degree of interactivity is provided by animation features that some PM tools offer. These typically show the flow of each case through the discovered process model based on the recorded timestamps that specify when an event occurred during case execution. For this, the absolute timestamps are used, shifting the focus from the duration between two events to the absolute timing of the whole process. This can be used to visually analyze its behavior across the full duration of the event log and greatly simplifies spotting peak times when more process activity occurred than usual.

Focusing on Delta Analysis

Even after applying all proposed mitigations, when performing PM based on KiPs, the generated models will always bear some degree of complexity. This is explained by the nature of a KiP which is inherently complex and can therefore not be expressed as a perfectly structured model that is trivial to understand. Luckily, the research questions do not focus solely on analyzing, understanding and visualizing the underlying CPs. Rather, they aim at evaluating the impact of the case engine that was used supportively for executing the modeled CPs. The evaluation part of this work is therefore primarily interested in the comparison of the discovered process models with the underlying case models. In

other words, the delta between the case definition and the process execution should be evaluated.

This allows us to refine the process models up to a certain point and to then switch over to identifying differences between the discovered models and the case definitions. Finally, the differences can be evaluated to generate new insights not only about the processes themselves but also about the engine executing them. When simply taking the inverse of the delta, process similarities can be analyzed. Using the approach, complexities that still exist in the models are not removed but can simply be ignored as maximizing the understandability of the whole process is not the primary goal. Of course, a lot of attention still needs to go into the complexity reduction. Otherwise, the differences are so vast that no evaluation can realistically be performed.

The delta can also be generated for comparing the different sites of the study with each other, giving insights into e.g. cultural or economic differences. This has already been performed in [120] where a cross-organizational comparison was done to generate insights about the execution of a CP for patients with chest pain symptoms in four public hospitals across Australia. When applied in the healthcare sector from an organizational perspective, analyzing the delta between models can reveal differences in social interactions and hierarchical structures. This was shown in [3] which discovered role interaction models for different diagnoses and also evaluated differences and similarities in social behavior. Comparing a descriptive organizational process model to a prescriptive organizational case model was also already done and shown to generate valuable results in a study that evaluated a role-based access control system which was configured through user and role models [90]. This exemplifies that delta analysis of discovered process models can be used to compare intended with observed social behavior, as it is required for the third research question of this work.

3.2. Proposed Methodology

3.2.1. Consolidated Methodology for Process Mining in Healthcare

This work uses the consolidated methodology for process mining in healthcare proposed by Erdogan and Tarhan in [36]. The authors did a systematic literature review to identify existing methodologies which found four generic methodologies and eight papers employing healthcare specific approaches [35]. Based on these methodologies, they collected PM features (like the definition of goals, process discovery, conformance checking or trace clustering) and evaluated which methodology supported which feature [36]. As they found that no single methodology was generic enough to support all the available PM features, they decided to develop a consolidated methodology that can be used for analyzing arbitrary processes in the healthcare sector based on pre-defined goals and questions. The proposed approach was evaluated in a case study that successfully managed to analyze and improve the surgery process of a Turkish university hospital while also lowering

costs. The methodology consists of seven subsequent stages and can be repeated in an iterative way. It is also shown in figure 3.3.

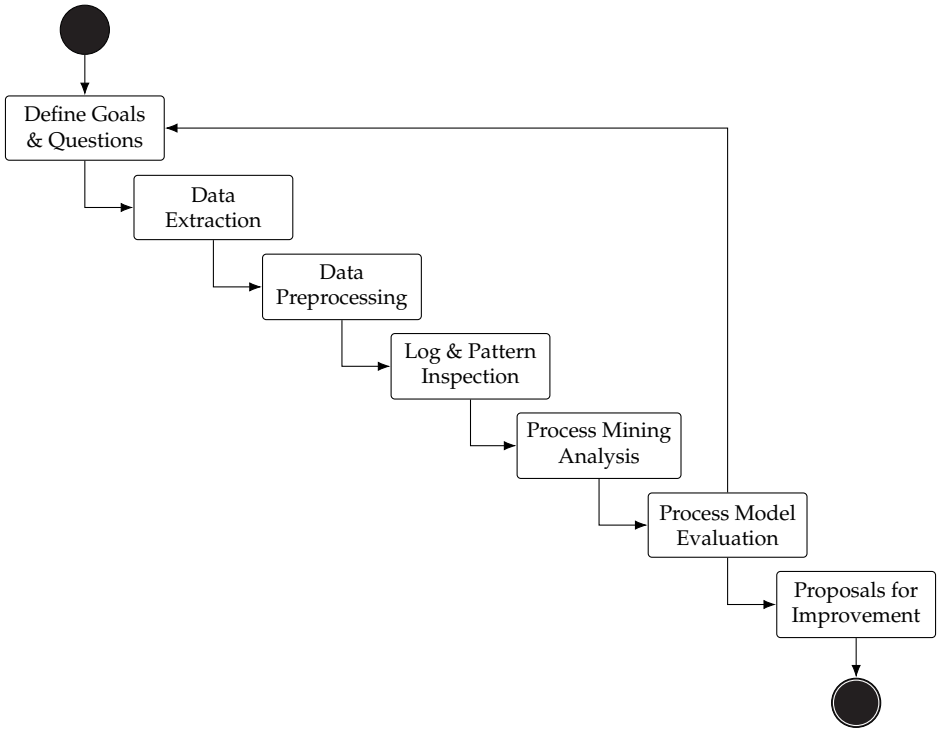


Figure 3.3.: Consolidated methodology for mining healthcare processes as proposed by Erdogan and Tarhan in [36]

(1) Define Goals & Questions The first stage aims to define the scope of the project as well as project goals and questions based on these goals. The scope specifies what processes will be analyzed for which patients and from when to when the evaluation will take place. The goals of the project may be things like optimizing one of the hospital's key performance indicators (KPIs), reducing the execution time of a certain surgery process or evaluating the process influence of a supportive tool that was newly introduced. As the methodology follows a procedure similar to that of the goal question metric approach known from software engineering, the defined goals are translated into a set of concrete performance-driven questions [5]. The questions are in turn mapped onto certain indicators which can be measured using one or more PM features. By doing this, quantitative answers can be measured for each of the posed questions, enabling an objective judgment if the desired goals were reached. Output: Goal-Question-Feature-Indicator (GQFI) table

(2) Data Extraction In this stage, all relevant data is extracted from the HIS. Being mostly

distributed systems, HISs tend to distribute their data across multiple sources, complicating the extraction procedure. The extracted data may be noisy and incomplete as long as every data entry contains information on the performed activity, the patient that was affected and the timestamp when it was performed. Apart from all event related data, pre-existing process models are collected for later conformance checking, together with other artifacts like CPs, the data models, patient feedback or the KPIs defined in the GQFI table.
Output: Extracted dataset

(3) Data Preprocessing As described in section 3.1, multiple measures can and have to be taken to prevent data quality issues and a consequential impact on overall PM quality. Doing so is the purpose of this stage, which applies formatting, filtering and abstraction techniques in order to transform the noisy and potentially unstructured dataset into a well-formed event log. Additionally, clustering techniques can be employed to split the dataset into multiple, smaller but more homogeneous event logs.
Output: (Clustered) event log

(4) Log & Pattern Inspection The stage for log and pattern inspection is executed to evaluate the GQFI indicators, collect and create a statistical summary and gain a first impression of the event log. The summary consists of generally available data like the number of cases and events, the duration of events and their absolute and relative frequencies. Most PM tools support this stage out-of-the-box and provide an adequate summary by default.
Output: Evaluated indicators; log and pattern summary

(5) Process Mining Analysis Now that a proper event log has been created, different types of PM analysis can be performed. Most projects include the application of a process discovery algorithm, but there are many PM features available that can be employed for different reasons, like the decision points analysis that aims to extend an existing model with data related attributes like age or gender. The paper by Erdogan and Tarhan exhaustively lists the PM techniques broadly available today and what they are generally trying to achieve [36].
Output: Discovered process models; most-followed paths, outliers and bottlenecks

(6) Evaluate Results This is the stage where all indicators that were captured in previous stages and the artifacts generated during the PM analysis are evaluated. Performance problems and inefficiencies are identified and the most complicated or time-consuming cases are evaluated in depth to uncover potential risks. The initial goals and questions are answered but new ones may arise during the execution of this stage.
Output: New process insights; probably new goals and questions

(7) Proposals for Process Improvement The stage for process improvement proposals aims to increase the overall process quality by eliminating rework loops or bottle-

necks, preventing unnecessary process elements, and understand complicated or time-consuming activities. The knowledge generated up to this point is gathered, reviewed and can then be used to first derive opportunities and lastly proposals for process improvements.
Output: Proposals for process improvement

3.2.2. Conceptual Methodology Adaptions

In a recent paper, van der Aalst described his vision of using PM as technology for analyzing event-based data similar to how spreadsheets are used for dealing with numbers [156]. Spreadsheets have been around since over 40 years, have wide-spread appliances in most business areas and have been shown to be able to provide added value in an "almost endless" amount of situations [156]. The author therefore argues that PM is similar to spreadsheets in that both technologies provide generic operations to deal with arbitrary input data while also offering means for customizing the generated visualizations to the use case at hand. He also identifies the technology of PM to bridge the gap between data science and business process management.

To support this vision and enable domain experts not familiar with PM techniques to gain insights into the executed processes, a high degree of automation and little to no required configuration was desired to generate process models in the context of this work. Apart from that, case studies are still running after the end of this evaluation project, offering opportunities for creating additional scientific value if the analysis can easily be repeated after the studies have been finished. RapidProM is a tool that provides such features with a visual scripting environment that can be used to define automated processing and analysis workflows without requiring programming skills [97]. The software relies on ProM's plugin ecosystem to provide implementations for the available PM algorithms. However, these plugins have often been developed during scientific studies, are tailored towards PM experts and require some non-trivial configuration effort to produce meaningful results (e.g. users need to understand the concepts of significance and correlation in order to run the fuzzy miner).

Therefore, a different approach is used and Disco is employed for an interactive discovery of process models using a user interface that is intuitive to non-PM-experts [57]. The drawback to this is that it requires a manual import of the dataset which in turn requires manual configuration. In comparison to the ProM plugins however, the necessary configuration is limited to specifying the mappings of the individual columns in the event log to import. As the mappings are static, this does not require advanced knowledge but is still tedious. To mitigate the issue, Disco's Airlift component could be used, which is a server that can be used to serve up-to-date, pre-configured datasets to authenticated Disco user, eliminating any manual import or configuration effort. In this work, however, as the Airlift component is not available in the academically licensed version of Disco, the feature will not be implemented and a manual import & mapping will be used. Currently, no other tools are available supporting a fully integrated PM approach for non-PM-experts

without requiring any commercial licenses.

To still enable the repetition of the analysis procedure at low costs, all process steps prior to the PM analysis need to be automated. Additionally, to prevent disrupting the domain experts' daily work and to further encourage the application of PM, performing the PM analysis should be possible at any point in time. Therefore, a log analysis application is used to continuously extract and preprocess log data in near real-time. The resulting event log is then persisted to an event log storage from where it can be retrieved to perform online log and pattern inspection using the log analysis application's user interface for data exploration. Afterward, the automatically created event log data can be imported into Disco to discover the process models and make use of the interactive visualization features for creating specialized process visualizations. As mentioned, this manual import could also be fully automated using an Airlift server if the feature was available under an academic license.

To account for the continuous approach of this work, the original methodology by Erdogan and Tarhan [36] had to be slightly modified. The adapted version of the methodology is shown in figure 3.4. It contains a new element, the event log storage, which holds an always up-to-date version of all events that occurred so far and have already been preprocessed. Furthermore, all stages have been automated and are executed continuously, except for the definition of goals & questions, the evaluation of the generated models and the proposal of process improvements. This is indicated in the figure by a self-looping transition. The automation of the PM analysis stage that generates the models is reflected in the adapted methodology but was not implemented due to licensing reasons. Apart from that, the two-part nature of the approach is illustrated in the updated figure, with stages on the left side leading towards generating the event log while later stages make use of the event log in order to generate process insights.

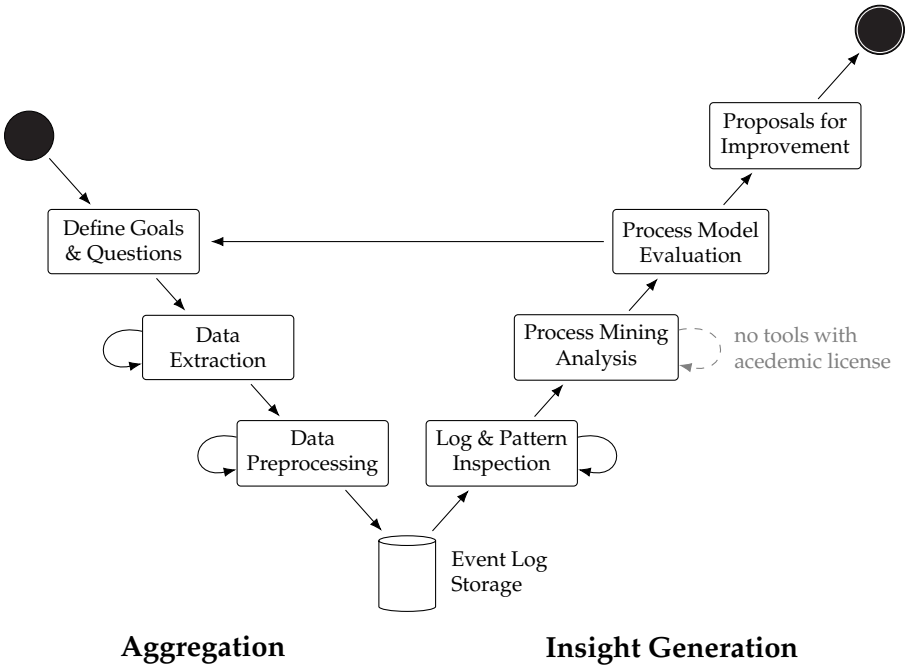


Figure 3.4.: Consolidated PM methodology adapted to the CONNECARE use case. Transitions in gray have not been realized due to lack of academically licensed tools.

4. Implementation

4.1. Goal-Question-Feature-Indicator Table

The first methodology step when following the approach proposed in the previous section is the creation of a GQFI table which is similar to a goal-question-metric table with an additional layer for the type of PM analysis that is used to generate a certain indicator [5, 36]. To create the table, first, the scope of the project, its goal, and the questions that need to be answered for reaching the goal are stated. Then, the appropriate PM analysis types are chosen and indicators are defined that are suitable for answering specific aspects of the questions in order to come up with an objective answer.

For this work, the usage of the CONNECARE integrated care system during two case studies in the period of July 2018 through March 2019 is evaluated. The studies themselves run for a total of one year, so the analysis will likely be re-executed when they have been finished, after July 2019. One study evaluates the provision of integrated care services like home hospitalization to CCPs with COPD and a recent acute episode, to avoid hospital readmission and reduce the cost of care. The other study assesses enhanced pre- and post-operational care to improve the quality of life for CCPs undergoing elective surgery while also saving costs. The EU-funded studies are conducted at three hospital sites, namely Lleida in Spain, Groningen in The Netherlands and Tel-Aviv in Israel. In the time period under consideration, around 300 patients have participated in the studies. The exact numbers for each site and study are shown in table 4.1.

Case Study	Groningen	Tel-Aviv	Lleida
CS1 (COPD)	25	51	35
CS2 (Surgery)	35	29	35
Both	60	80	70

Table 4.1.: Number of participants in each study for all three sites

The goal of this work is to evaluate the CONNECARE information system for integrated care and, more specifically, how it and its engine SACM was used to support the complex clinical processes found in the healthcare sector. However, the project was not started due to the general need to evaluate the system usage, but from three concrete research questions that were described in detail in section 1.3. The first question is concerned with the SACM's runtime flexibility and evaluates how it was employed during the execution of

the studies. The second question evaluates how its additional features for communication and notification affected the case executions. Finally, the third question checks how the collaboration and organization features were used in clinical practice.

As stated, the questions were not derived from the overall goal but the process was rather the other way around, deriving the goal from the individual questions as a kind of conceptual project summary. The three types of PM that will be applied to come up with answers are (1) log and pattern inspection, (2) process discovery and (3) process exploration. The perspectives will mainly be the control-flow and the organizational perspective, but some metrics from the performance perspective will be incorporated as well. The indicators used for evaluating the questions were recorded in a hierarchical GQFI table that can be seen in table 4.2. Their selection was based on the exhaustive list relating PM features and possible indicators included in [36]. Additionally, discussions with stakeholders of the SACM system and domain experts in the executed case studies were held in an iterative way to ensure the relevance and validity of the selected indicators.

4.2. Data Extraction and Preprocessing

To enable the continuous, near real-time processing of access logs and keep the event log always up-to-date, a log analysis tool has to be employed to realize the methodology steps for data extraction, preprocessing and log & pattern inspection. For this purpose, the *elastic stack*, formerly known as *ELK stack*, was chosen. The elastic stack describes the combination of the three applications *Elasticsearch*, *Logstash* and *Kibana*. Elasticsearch is a modern search engine that is based on the *Apache Lucene* search library and offers its functionality via a JSON-based REST API [31]. It is amongst the most popular search engines and often used for full-text search (e.g. in Wikipedia) but it can also be used for application performance monitoring and log analysis [68, 80].

The elastic stack is released under an open-source license and consists of 3 major components, namely Elasticsearch, Logstash and Kibana. As mentioned, Elasticsearch is the actual search engine that can be used to index documents and execute queries on the generated indices in a performant way. Logstash is a very flexible data preprocessor and log shipper that can be used for importing documents into Elasticsearch. The third component, Kibana, is a web application that can be used to explore the raw data in Elasticsearch with the help of a strong and flexible filtering system that enables an easy-to-employ drill-down process. Apart from that, it offers functionality to build custom visualizations based on that data. The visualizations can then be aggregated into dashboards which provide a comprehensive overview of the imported data. All components are developed as standalone projects but follow a common vision and a unified release cycle. Each component can be scaled up both vertically and horizontally, providing the whole stack with good overall scalability, as deployments can be tailored to the individual use cases. This keeps cost low but performance and availability high.

In the context of this project, Elasticsearch is used for storing the raw API logs as well as

Goal:	**Purpose**	**To Understand**
	Issue	The use of characteristic system features during
	Object	two different case studies at three sites each
	Viewpoint	from a case modeler's or system analyst's viewpoint
Research Question 1		
How is the model-provided flexibility employed during the execution of cases?		
PM Feature		**Indicator**
Log and Pattern Inspection		Number of cases/events/activities
		Number of manually activated tasks
		Share of manually activated tasks
Process Discovery		Number of overall paths
		Mean number of paths per activity
		Number of bidirectional paths
Process Exploration		Number of process variants
		Maximum share of cases per variant
Performance Analysis		Median/mean case duration
		Standard deviation of case duration
Research Question 2		
How do communication and notification features affect case executions?		
PM Feature		**Indicator**
Log and Pattern Inspection		Number of notes update events
		Number of alert events
		Number of message events (with professionals/patients)
		Share of feature-related events
Process Discovery		Mean number of paths per alert activity
		Mean number of paths per message activity
		Mean number of paths per other activity
Research Question 3		
How are collaboration and organization features reflected in case executions?		
PM Feature		**Indicator**
Log and Pattern Inspection		Number/Share of tasks not done by their assignee
		Number of task assignees/task processors
		Number of roles
Organizational Analysis		Maximum share of work for one person
		Mean number of collaborators per person
		Maximum number of collaborators per person

Table 4.2.: GQFI table for the CONNECARE case studies

the consolidated event logs. Apart from that, it powers Kibana, which is used for log and pattern inspection. To enable this, custom visualizations and dashboards were created to provide the user with a quick, domain-related overview. Finally, in order to get the data into Elasticsearch, Logstash is used to extract the data, preprocess it, derive the event log and persist both the raw access logs but also the consolidated event log to different indices in Elasticsearch. Due to the lack of a fully integrated PM tool under a suitable license, Logstash is also used to export the consolidated event log from Elasticsearch into .csv files that can then be imported in the PM tool.

4.2.1. Logstash and Logstash Pipelines

Logstash uses one or more pipelines to extract, transform and ship data. Each pipeline follows the internal structure of *input* → *filter* → *output*. The three stages are implemented using a plugin ecosystem that offers a vast amount of possible plugins for each stage. Custom plugins can also be developed if necessary. The input and output plugins show Logstash's good integration capabilities, with plugins for integrating with Elasticsearch, local files, JDBC-based databases, WebSockets and TCP or UDP sockets but also for proprietary solutions like Amazon's CloudWatch API, Google's BigQuery or loggly, a commercial log analysis service mentioned earlier. The filters are realized using the same plugin-based architecture and offer a wide range of functionality. The plugins enable string manipulation, looking up additional data, dealing with data types, encodings and time formats, but also throttling pipelines or dropping, cloning and aggregating events, to name only some examples. A full list of available input, filter and output plugins can be found in [32].

The overall processing structure for this project consists of multiple subsequent pipelines that perform different conceptual tasks and exchange data via *Pipeline-to-Pipeline Communication*. An overview of the different pipelines and their interaction is given in figure 4.1. By following this structure, the size of each pipeline is kept small which simplifies the change and version management and enables atomic redeployments, which is a great advantage during the initial pipeline development. Apart from that, splitting the processing up into multiple pipelines and passing data from one to the other explicitly enables the control of the flow of data. This, in turn, enables data to pass some pipelines but skip other ones while also allowing for the persistence of data at different stages in the process. Additionally, each pipeline is split up into multiple YAML-files so that every file only contains one transformation step which also helps to keep the complexity of the overall data processing code manageable. When the configuration is read, files belonging to a pipeline are joined together, making the additional structure transparent to Logstash.

The logic implemented in Logstash is divided into two parts because there are two different databases acting as primary data sources for data extraction. One of them is part of the SMS and contains the patient messages, while the other one is the logging database where API accesses to the SACM backend are written to. The data is extracted from there and preprocessed. During the preprocessing, the IDs of objects that are referenced but not included are extracted into a common form. The data is then passed to the next pipeline

that looks up the missing objects from the secondary source of information which is the SACM backend database. The data gathered during the lookups is then postprocessed, depending on the type of initial input data. Afterward, the data is persisted to Elasticsearch in its current form, so that the result of the lookups and the processing can be checked for data quality purposes. Apart from that, this enables the typical application performance monitoring use cases of the elastic stack which require the complete web access logs to be present in Elasticsearch. At the same time, the persisted data is also passed downstream, to the transformation pipelines. These apply some heavy filtering and aggregating, followed by the emittance of emulated events for lifecycle developments that are evident based on the recorded information in the SACM database but are not included in the API access logs for various reasons (e.g. due to predefined, automated actions performed by the engine). The transformation pipelines reduce the number of individual data items being processed by Logstash by a factor of roughly 500. After the data went through these pipelines, the final, consolidated event log is created and can be persisted to Elasticsearch. From there it can either be inspected to mitigate data quality issues once again or it can be exported to comma separated value (CSV)-files for performing PM analysis.

In the upcoming subsections, first, the primary data extraction and preprocessing will be described and technical details on the employed technologies will be given. Afterward, the additional object lookups are explained and details on the necessary postprocessing are given. Lastly, the transformation pipeline is described and details in applied filters, aggregations and the emittance of emulated events are presented.

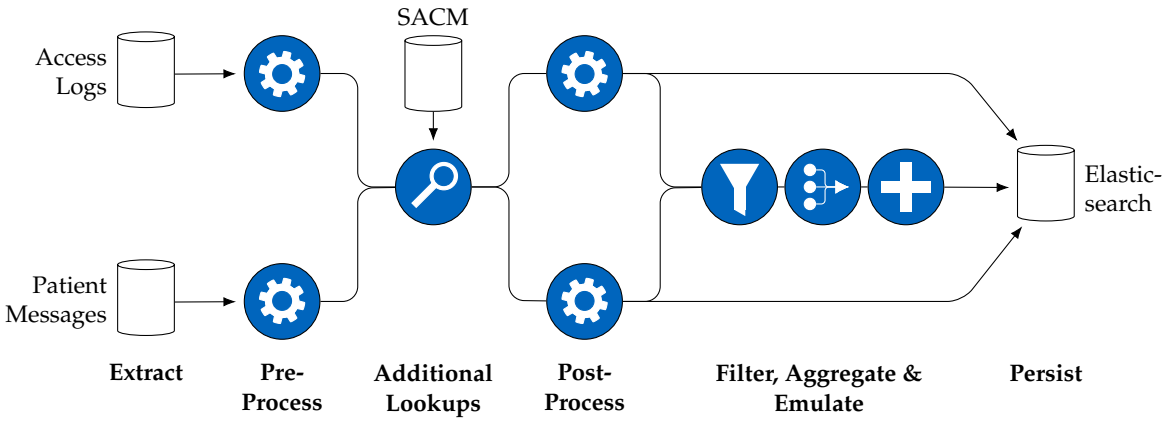


Figure 4.1.: Overview of Logstash pipeline components and their interaction. Arrows indicate data persistence operations.

4.2.2. REST Log Extraction and Preprocessing

The database containing the largest part of the process-relevant data is the API access log database. For the evaluation period from the 1st of July 2018 to the 30th of April 2019, the

database contains roughly 13 million access logs. The only other data source that needed to be integrated for data extraction was the database of the SMS which contains the messages between patient and professionals. This is due to the patient messaging feature not being part of the generic SACM core, which is why it was modeled as a dependency to an external service. However, as in the context of the CONNECARE project, the functionality is implemented by the SMS which is part of the common infrastructure environment, the database can directly be accessed from the elastic stack and the available data can be extracted to gain additional insights. Nonetheless, this data source is of secondary importance as it only contains the around 1 500 patient messages which is quite little data compared to the around 13 million access logs. Apart from that, the flexibility, the social interactions and even the largest part of the feature usage could still be analyzed if the patient messages were unavailable.

When starting this project, the API access logs were stored in a MongoDB[1] using a custom logging implementation based on *winston.js*[2]. The reason for using a custom implementation rather than relying on one provided by the upstream reverse proxy is that extra data from the execution context can be added this way without creating significant additional system load. For example, information on the user performing the request is looked up based on the authentication information provided with the request. Thanks to the custom logger implementation, this user data can be logged and does not have to be looked up again. In the case of logging through the reverse proxy, this information would not yet be available and would have to be looked up again when the PM event log is created, resulting in increased resource usage.

Unluckily, being a database that uses its own Java driver rather than relying on the JDBC standard, MongoDB is not supported by Logstash except when using some unusual workarounds. Therefore, and to align the database technology across the CONNECARE project, a decision was made to migrate the contents of the MongoDB to a MySQL[3] database. However, it was decided to stick with the existing custom implementation and not switch to logging through the reverse proxy, so that the mentioned advantage of reduced system load was kept. To achieve this, the implementation was first adapted to create the log entries in both databases. After ensuring that all information was persisted correctly using the new implementation and no other issues arose, the existing content of the MongoDB was migrated to the new MySQL-based system. This was implemented using a custom RESTful API endpoint that was created to facilitate a migration in multiple smaller batches which ensured that the system stayed accessible during the time. Afterward, the old database was switched off and the now obsolete endpoint was removed again. All other databases were based on MySQL anyhow and did not have to be migrated.

The input plugin used for extracting the data is the *logstash-input-jdbc*[4] plugin which

[1] https://www.mongodb.com/
[2] https://github.com/winstonjs/winston/
[3] https://www.mysql.com/
[4] https://github.com/logstash-plugins/logstash-input-jdbc

is based on the JDBC standard and run once a minute to import new data. It is configured to persist the ID (or modification timestamp if unavailable) of the record that was last imported, so only changes since the last run are picked up. A full import of the 13 million log entries takes about 6 hours under full system load. The one-minute interval is a compromise between the timeliness of the processing and the additional load on the database due to the increased amount of queries. The only way to further improve the import promptness is to feed the logs directly into Logstash and persist them only in Elasticsearch, eliminating the intermediate MySQL database. This could be done in the future to reduce the data storage and memory requirements of the overall system. However, as data was already captured before this PM project was started, the introduced elastic stack is not yet an integral part of the CONNECARE environment and the promptness of system was sufficient, the existing database was kept in place. As a positive side effect, the database served as a backup for doing full re-imports in case issues were discovered in the Logstash implementation. Additionally, decoupling the SACM backend and the process analysis stack enabled the latter one to be shut down if negative effects on the main system's performance were identified, which was already in productive use at the time and had some constraints regarding the available system memory.

The amount of preprocessing applied to the data after the extraction but before the additional lookups is actually quite small. Primarily, based on the provided request URL and the derived URL pattern, IDs contained in the URLs can be extracted. This is necessary to find out which object (e.g. case or task instance) a certain activity refers to. Based on the provided URL pattern, additionally, the corresponding API event is determined, which describes the conceptual activity that is performed when the request is executed. All other transformations applied in the preprocessing pipeline are done to reduce the noise in the data and ensure data consistency, e.g. regarding field names. For achieving this, some data like internal properties created by MongoDB or large fields with currently unused data (e.g. request bodies in case of erroneous requests) are explicitly removed. Other properties are simply renamed to ensure consistent naming in the context of this project.

4.2.3. Additional Lookups and Postprocessing

As mentioned, some objects are only referenced by their ID and need to be looked up in the SACM main database. Examples include entities referenced in the requested URL but also references in the returned objects. The lookups need to be done for various reasons, e.g. to determine to which task a received alert belongs to, which definition version a certain case is an instance of, or simply what the name of a referenced task is. Two special cases exist, where the number of alerts for a specific task ID and the number of messages for a certain case ID are counted using SQL queries. This is done so some additional data is available during early log and pattern inspections when the consolidated event log, which could be used to determine the same statistics, has not yet been created. An exhaustive list of all entities that are looked up and the reason why the lookup is necessary can be found in table 4.3.

Table	Property to Lookup	Data to Lookup
Task, Stage	Task ID	Task Metadata (Name, Assignee, etc.)
Case	Case ID	Case Metadata (Patient, Team, etc.)
Task Definition	Task Definition ID	Task Definition Version
Stage Definition	Stage Definition ID	Stage Definition Version
Case Definition	Case Definition ID	Case Definition Version
Group Membership	Request User ID	Groups a User belongs to
Group Membership	Task Assignee ID	Groups a User belongs to
Group Membership	Message Author ID	Groups a User belongs to
Alert	Alert ID	Reference to Task
Alert	Task ID	Number of Alerts for certain Task
Message	Message ID	Reference to Case
Message	Case ID	Number of Messages for certain Case

Table 4.3.: All data that is looked up, the property that it is looked up by and the table where the data originates from

Once the lookups are done, some postprocessing transformations are performed in a two-stage process. First, common transformations are applied that affect all data records, regardless if they originated from the access log or the patient message database. Afterward, different record-specific transformations are applied, depending on the primary data source. The common postprocessing consists of three basic operations, which are time duration calculations, the flagging of requests and the mapping of certain request properties to a form with improved human readability. Time duration calculations are automatically applied when a related start and end timestamp are included in the data. The resulting duration is used in the dashboards for log and pattern inspection but is not relevant to the PM analysis as that only takes end timestamps into consideration for conceptual reasons. Flagging of requests is applied to rule out irrelevant (test) requests, either based on known, hardcoded test user IDs or because the request was received outside of the fixed study period. Mapping is at this stage only applied to workspace IDs, which are translated into human-readable site names using a static mapping.

The transformations only applied to the access logs are the flagging of finished tasks based on the availability of a completion/termination timestamp and the parsing of the user agent string using an external library. Additionally, to further increase the information available during log and pattern inspection, the request's source IP is looked up in a GeoIP database[5] to enable geographical usage information to be displayed. The only transformation applied to the patient messages is the flagging of messages that contain an attachment based on the availability of an attachment reference ID, simplifying later

[5]GeoLite2 open source geolocation database, see https://www.maxmind.com/en/open-source-data-and-api-for-ip-geolocation

analyses of this information.

4.2.4. Transforming API Logs into Event Logs

From a PM point of view, all pipeline logic so far was concerned with data extraction as it primarily increased the amount of available information through additional lookups. Therefore, what comes next is the main logic that preprocesses the access logs and patient messages into an event log that is consumable by PM algorithms. This pipeline is the most complex one and basically consist of seven phases: (1) filtered input, (2) data minimization, (3) readability improvements, (4) clustering, (5) activity emulation, (6) sequence order fixing and the (7) aggregating output. Once all data was processed by the pipeline, the initial dataset of about 13 million entries is reduced to only around 25 000 records in the consolidated event log. In the following, the pipeline will be explained in depth according to the structure given by the seven subsequent stages.

(1) Filtered Input The input to the pipeline is like all non-database-related inputs based on pipeline-to-pipeline communication. To keep the amount of data being processed by this computation-intensive pipeline low, filtering is applied as early as possible. However, for technical reasons, filtering is not possible on the input side of the pipeline-to-pipeline communication plugin and has to be done on the output side, which results in slight code duplication that is acceptable in order to get the performance advantages. Therefore, in the output stage of both preceding postprocessing pipelines, only API access logs conforming to the following criteria are passed downstream:

- was created by a real user for a real case
- was created during the study period
- was issued from one of the study sites Groningen, Lleida or Tel-Aviv
- has a successful HTTP status code in the range of 200 to 299
- has a case instance associated
- was not issued by the UIM (which only requests details about users and is not relevant to PM analysis)
- refers to an API event that is relevant for the PM analysis

An alternative input is also available but is disabled by default as it is only used for development purposes. It offers the possibility to input the lookup-enhanced logs directly from Elasticsearch instead of passing them through preceding pipelines again. This turned out to be useful during the development, as it enabled the re-execution of the later pipelines based on the data that was already imported in Elasticsearch, eliminating the need to perform the lookups again and reducing this overall import time.

(2) Data Minimization The first phase after the input is the minimization of the data that is passed down the pipeline in order to reduce the memory footprint and the storage requirements. Previous pipelines also removed some unused data but have only explicitly excluded certain properties. However, Elasticsearch supports a dynamic, flexible mapping of properties so that new, previously unknown fields can be added and are picked up and persisted without any errors or warnings. This is helpful in the case of raw application logs as it enables the extension of the logged data without having to change the Elasticsearch configuration as well, in case e.g. new debugging requirements arise. In a PM event log, by contrast, it is desired to only include process-relevant data for keeping data quality high and noise low. New fields should only be added to the event log if the PM expert makes an explicit choice to do so. Therefore, this pipeline does not exclude unwanted fields but rather include certain process-relevant fields using a whitelist. All other properties are dropped. Apart from that, nested fields containing objects are flattened, so that the resulting event log entry is of single depth which simplifies the export. Lastly, some fields are renamed to increase consistency with the PM domain (e.g. all occurrences of the technical term "request" are removed as it is not related to PM).

(3) Readability Improvements In the next phase, the technical labels that are used internally by the SACM system are converted into formatted representations that are easier to read for humans. This is done to increase the understandability of the resulting process models. The conversion is done using Logstash's *translate filter plugin* which features a simple string replacement mechanism based on a static mapping that is kept in CSV-files called *dictionaries*. Dictionaries exist for humanizing task, stage and case names but also for mapping group IDs to the conceptual names of the groups. The assignment of a generic user role for each professional is realized in the same way. Apart from that, the previously determined API event is translated into a readable activity type and name. Additionally, IDs of users (patients and professionals) are translated into human readable random strings (HRRSs). An HRRS is a randomly generated string that is pronounceable by humans and facilitates the communication about data. The concept is similar to *proquints* but with a simpler implementation based on pseudo-random number generators and a simple alternation between vowels and constants [173]. The advantages, however, remain the same.

(4) Clustering As described in section 3.1.2, clustering will be applied based on the compatible case or stage versions. Which versions are considered compatible with each other depends on the view that is chosen for the PM step. To enable switching between the different views more easily, for each event log entry, the compatibility for all available view levels is determined. This is implemented using the same CSV-dictionary-based string matching approach known from the previous phase. Multiple dictionaries exist that match all of the possible case and stage names to a compatibility group used for the analysis. The groups are persisted to different fields

of the object, depending on the view that they are valid for. A full reference with all compatible analysis groups for every view can be found in section C.3 of the appendix.

(5) Activity Emulation As described earlier, the SACM logs request bodies only if errors are encountered, for debugging purposes, and does not log response bodies at all. This becomes an issue for POST requests that are used to create new objects (e.g. task or case instances) in the system, as they are identified by their ID which is not available before the request has been completely processed. The only way to get the ID of the created object is parsing the server response, which is not logged and therefore not available. Furthermore, based on the case model, the SACM supports automatic transitions if certain prerequisites are given which can trigger the implicit creation of objects. Afterward, users may interact with these objects in a process-relevant way making them suitable for inclusion in the event log. However, no request has ever been received for the creation of the object, which is why this step of the object's lifecycle is not included in the access log as well. To still be able to depict the complete lifecycle in the generated process models, the information needs to be extracted from the SACM database, where timestamps about all important lifecycle events are recorded. Using this creation timestamp information, an emulated access log can be emitted, enabling the inclusion of all lifecycle events without having to modify the remaining pipeline. This emulation is done for all object types that can be instantiated (case, stage, task, alert, message).

(6) Sequence Order Fixing As just explained for phase 5, some objects can implicitly be created by the engine. Apart from the problem of missing access logs, some other issues are created by this behavior. One of them is timing problems that occur because the engine creates entities faster than the precision of the logging system and the persisted timestamps allow. This leads to multiple entities being created with the same timestamp. In the event log, the ordering of these objects is not well defined and depends on the implementation of the export. This, in turn, can lead to invalid sequences in the event log, e.g. when the first stage of a certain case is activated before the referenced case instance was even created. To counter this effect, the timestamps of objects that are dependent on each other are slightly modified by adding or subtracting one or two milliseconds. For example, creation timestamps of case objects are always reduced by one millisecond while creation timestamps of tasks are increased by one millisecond. This ensures that – even if the timestamps were initially equal – an event for creating the case always occurs before an event for activating the first stage or a contained task. The change does not affect the correctness of the process models, as these are generated and used to gain insights into the human behavior and the issue only occurs in fully automated situations where a human only triggers the process and is then not involved anymore. Moreover, humans typically do not act in terms of milliseconds.

(7) Aggregating Output Last but not least, internal fields that were used for holding temporary transformation results are removed so that the remaining data can finally be persisted to Elasticsearch. This final step is also the aggregation step that creates one consolidated event based on information from multiple API access logs. The aggregation is realized implicitly using Elasticsearch's update mechanism. To enable it, Elasticsearch's default behavior had to be changed to make use of a custom, explicitly assigned document ID instead of using an automatically generated one, based on the document content (document being Elasticsearch's generic name for one consolidated event in the event log). The document ID that is used equals the ID of the object in the SACM database and is therefore static and does not change. This allows persisting the same document multiple times during its lifecycle, while Elasticsearch cares about determining changes from one version the other and only persists new or updated fields. This effectively aggregates all information about a certain object that is created during its lifetime.

4.2.5. Exporting the Event Log

The consolidated event log that has been created by Logstash and persisted to Elasticsearch now needs to be shipped to the PM tool to enable the process discovery. Unfortunately, no PM tool supports direct Elasticsearch integration but most tools rely on file-based data input. There are also some tools available that support a fully integrated approach from data extraction to model generation but none of those is available under an academic license. Therefore, and to prevent vendor lock-ins in the early stages of the project, this work exports the event log to text-based files. When combined with a small amount of documentation, the files can later be used to import the data into the PM tool, even by an untrained user. This will be done to enable the repetition of the analysis and evaluation step after the case studies have been finished.

Multiple formats are available for representing the data in a text-based file with the most prominent ones being mining eXtensible markup language (MXML), eXtensible event stream (XES) and CSV. MXML is the oldest of the three formats and was published in 2005 [166]. It was designed as a format for representing an event log based on the properties included in the first metamodel for PM that was presented in the same paper. It is based on XML and quickly became the de facto standard for event log representation as it was required for inputting data into the heavily used ProM toolkit.

However, it had some limitations, primarily due to its rigidness and lack of extensibility. As PM started to be employed in new industry sectors with new types of data, a more flexible format had to be found that allows for the custom extension of the data by new and unpredictable data types while still offering PM algorithms a way of exchanging data across system borders. This resulted in the development of the XES file format which was accepted and published as IEEE standard 1849–2016 in late 2016 [1]. The format is designed as a future proof way to represent event logs and streams to improve interoper-

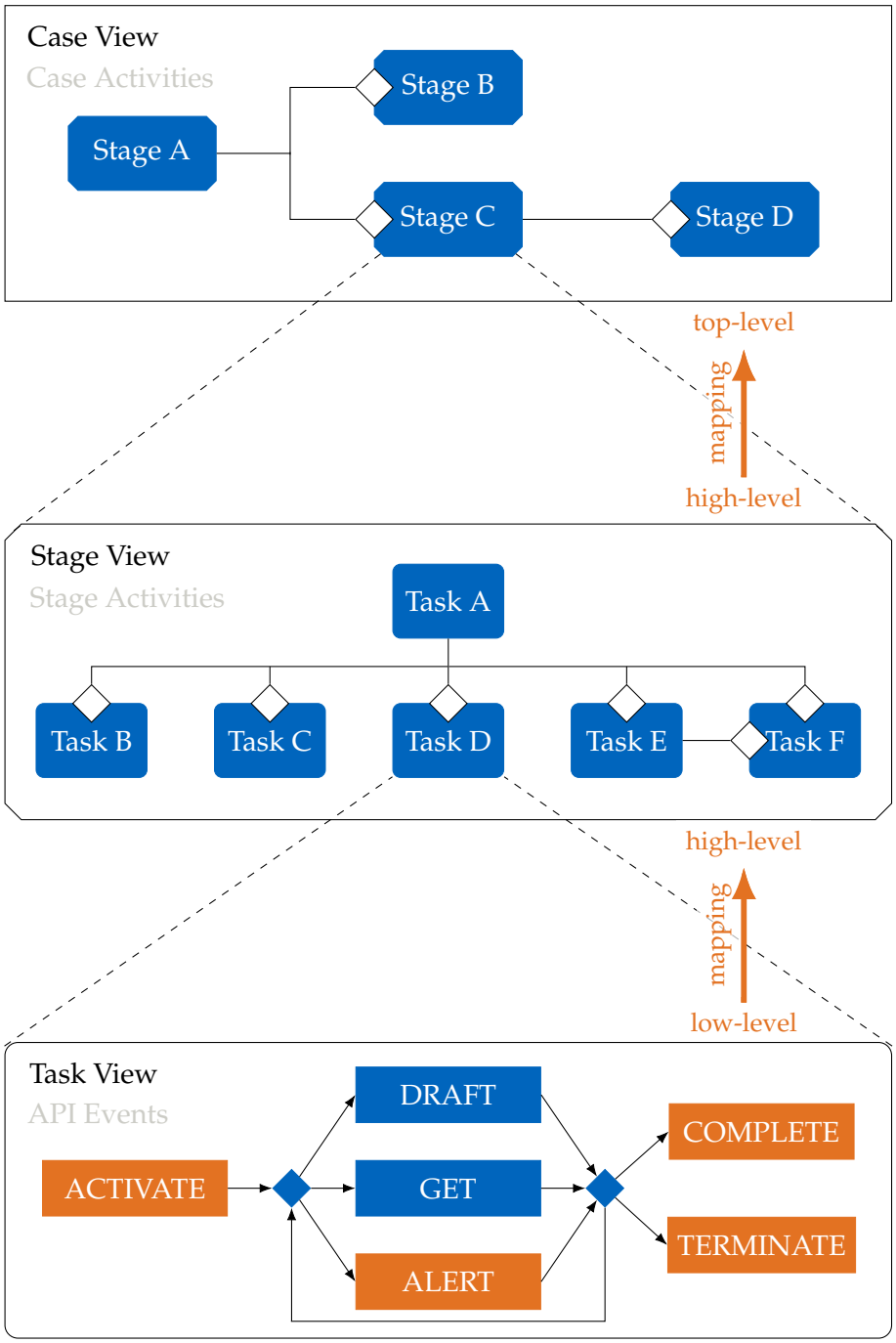


Figure 4.2.: Overview of process events at three levels of abstraction. Orange color indicates the mapping from API events to stage and case activities.

ability in the PM domain. However, not being native PM tools, neither Elasticsearch nor Logstash support exporting data to the XES (or the MXML) file format.

Therefore, the project exports all its event log data to CSV. CSV files are very easy to generate programmatically as they rely on a simple table-like structure. Historically, CSV is still supported by many tools because it was one of the first formats that were used in PM. Additionally, Disco, the tool that was chosen for the PM analysis and that will be explained in depth in section 4.4, has very strong CSV import capabilities [57]. In fact, it was created based on *Nitro*, a tool that was very popular before the XES standard was developed and that had the sole purpose of converting event logs from CSV to ProM's MXML. Logstash provides a plugin for exporting arbitrary data to CSV files [32].

The major drawback of using CSV is the lack of structure in the file, specifically the lack of representing the content of each column in a way that the PM tool can correctly parse it without needing an additional configuration. The required configuration needs to be provided in the form of a mapping. The mapping needs to include at least the columns containing the *case instance*, the *activity* and the *completion timestamp* and can optionally include additional properties. In the context of this project, the mappings are actually quite simple and an example for the case view can be seen in table 4.4. The full list of mappings is included in the appendix and shown in section C.2.

Disco Type	**CSV Column**
Case ID	CaseID
Activity	Activity
	ActivtityType
Timestamp	Timestamp
Other attributes	ClusterGroup
	_DebugID

Table 4.4.: Mappings for the case view

For varying the visual complexity and the level of detail in the resulting models, the event log can be exported at different levels of abstractions, called *views*. This was explained in section 3.1.2. When exporting the event log for the evaluation period of this project, nine months, at the highest level of detail (stage view), then the size of the resulting CSV file is only around 2 MB. As this small size will clearly not lead to storage issues and to simplify the implementation of the export functionality, the decision was made to continuously output the event log at all three levels of abstraction at the same time. This allows PM operators to choose the desired view on the event log at import time, right before the process models are actually generated.

When an individual event is exported, different properties are included depending on the level of abstraction in the target file. Additionally, certain events can be excluded (like messages in the stage view) and transformations can be applied to some values, e.g. for

treating Lleida's *Workplan after Hospitalization* stage as a simple *Workplan* stage.

4.3. Log and Pattern Inspection

The methodology's stage for log and pattern inspection is performed to gain an early impression of the data and created a comprehensive summary to improve process understanding [36]. It is usually done offline, based on the final event log and right before the process discovery step is initiated. The log and pattern inspection procedure is supported by many PM tools. ProM, for example, offers multiple plugins for generating data summaries and visualizations like the dotted chart analysis which shows the distribution of all events over time to gain an overview of the executed cases [21]. However, as the project is implemented using the elastic stack, data is continuously imported and the event log is also continuously exported, making online log and pattern inspection on the live data feasible.

This is implemented using Kibana, the elastic stack's browser-based analytics and visualization platform [33]. Kibana does not feature any explicit PM capabilities so some metrics (e.g. the number of paths or the number of process variants) are hard or even impossible to evaluate at this stage. Additional PM-specific metrics could be calculated at import time using Logstash, which features a filter freely programmable in ruby, but the increased effort of re-implementing parts of the underlying PM algorithms was deemed too high for the purpose of this project. After all, most data that is used in generic event log summaries is readily available which is sufficient for determining roughly half of the metrics required by the GQFI table. The remaining indicators are either generated by Disco, the PM tool used for the project (see the upcoming section 4.4), or they need to be evaluated manually. Kibana also offers the advantage of letting users choose the Elasticsearch data source (also called index) manually. Combined with the three-fold persistence mechanism explained in section 4.2, this enables users to inspect the raw data that the event log is created from, besides enabling traditional application performance evaluations.

Regarding application features, Kibana mainly offers discovery, visualization and dashboard functionality. The discover view, shown in figure 4.3a, can be used for drilling down into the data. It provides a text-based overview of the raw data points and allows to view the persisted information for each data record in its entirety. The view also provides a filter and query system that is very powerful but nonetheless quick and easy to use for non-experts. The application also features a popularity metric for all document properties. When a property is used in a filter or in a search query, its popularity is incremented. After some time of use, this leads to useful filter and search property proposals which facilitate collaboration in the analysis phase and provides a starting point for new users. The visualize view can be seen in figure 4.3b and is used to create visualizations that can be shared with other users. In addition to the search and filter functionality, which is available globally, the view features strong aggregation capabilities that can be used to group the data into customized buckets and calculate metrics for those. Apart from sharing, the created

visualizations can be grouped into dashboards, which are collections of multiple visualizations operating on the same data source and providing a customizable overview of the analyzed data. Multiple custom dashboards have been created to facilitate the general application performance monitoring and the log and pattern inspection in general but also for this project. Screenshots of all implemented dashboards are available in chapter D of the appendix and references are given in the corresponding descriptions which will follow in the remainder of this section.

Application Performance Monitoring Not project specific. Works on the API access logs and is therefore not even PM specific. Resembles a generic application performance monitoring dashboard based on RESTful web access logs like it could be used for monitoring the performance of arbitrary REST applications. Shown in figure D.1.

Event Log Overview Not project specific. Works on the event log and is used to gain an overview of the event log. Contains metrics like case count, task count, message count, etc., and does not differentiate between sites or case model versions. Shown in figure D.2.

Case Inspection Not project specific. Works on the event log and is used to gain an overview of one specific case. Contains e.g. the overall case activity displayed in a dotted-chart-like visualization, the active tasks and their lifecycle, the number of advanced feature usages, etc. Shown in figure D.4.

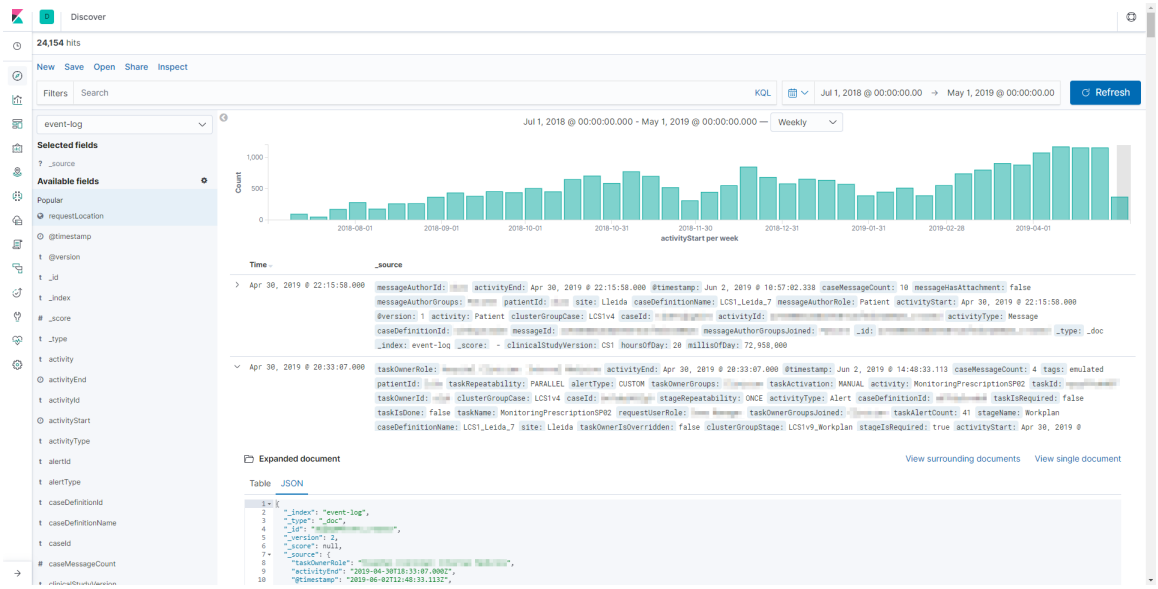
Site Comparison Project specific. Works on the event log and on the API access log an us used to gain a first impression of how the different sites use the system. Contains visualization showing e.g. the history of case creations but also at which time of day the system was used most actively. Does not differentiate between case model versions. Shown in figure D.3.

GQFI Evaluation Project specific. Works on the event log and contains all GQFI indicators that can be determined without the use of a PM algorithm. Shown in figure D.5.

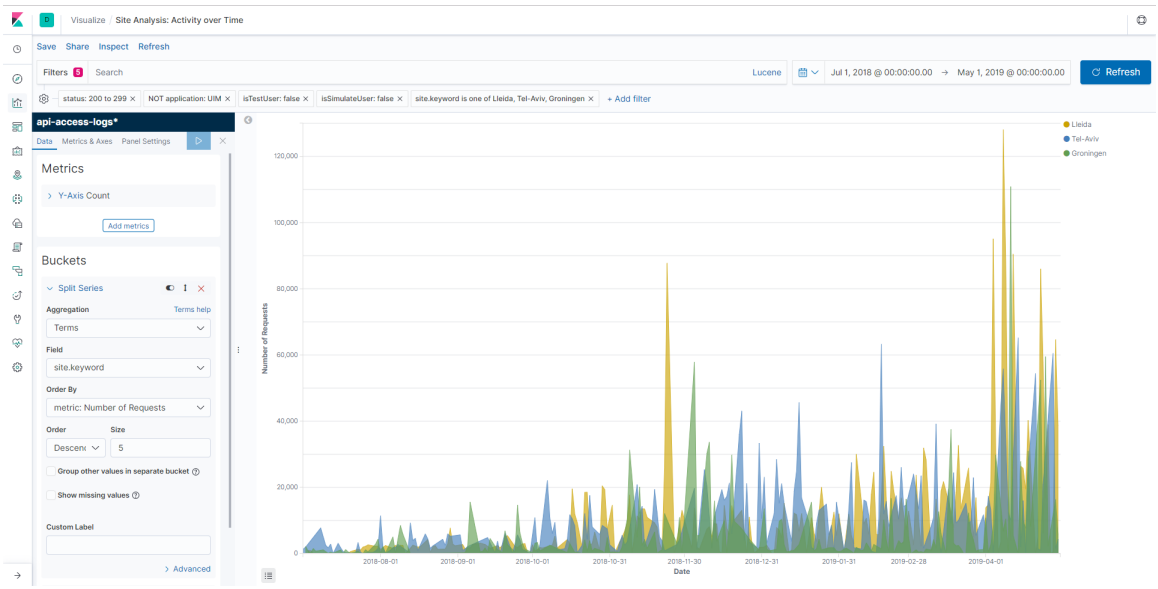
4.4. Process Mining Analysis

4.4.1. Available Tools

Different tools are available for assisting in the PM analysis step. To mitigate all identified challenges as described in section 3.1, the tool needs to offer support for comprehensive overview statistics, process discovery using the fuzzy miner algorithm and means of interactively visualizing the generated models. Apart from that, it needs to be able to import from CSV files. In general, two main categories of software solutions for performing PM exist: open-source and commercial software.



(a) *Discover* view for raw data inspection



(b) *Visualize* view for building visualizations of the aggregated data

Figure 4.3.: Different views of Kibana, the application used for online log and pattern inspection. Screenshots of the implemented dashboards can be found in chapter D of the appendix.

The most popular open source solution is the *ProM* toolkit. It features a plugin system that makes it very easy to extend, can be downloaded and used free of charge and is platform independent [165]. This lead to ProM becoming very popular and widely adopted amongst the PM research community [21]. Thanks to its plugin system, ProM offers the most PM-related functionality and contains the most up-to-date version of the available algorithms, as it is often used by researchers to develop new PM techniques. It is also well adaptable to the most PM use cases. However, this flexibility comes at the price of increased configuration effort. As plugins are often developed in the course of scientific publications, many plugins exist that require a large number of input parameters which in turn requires PM domain knowledge. In other words, the software needs to be properly configured by a PM expert in order to yield meaningful results. The issue is worsened by the fact that for producing process models, ProM requires the user to manually execute workflows that feature the subsequent application of multiple plugins. Each plugin transforms the data in a certain way and each requires an own conceptual understanding of the transformation and a correct configuration. Apart from that, development focus is generally not put on features like the intuitiveness and the usability of the user interface, but rather on the underlying algorithms that are usually the subject of scientific evaluation. Some solutions like *RapidProM* exist that try to improve these aspects and offer a visual workflow designer, enabling users to create the workflow once and then execute it automatically in subsequent runs. But the tools still rely on ProM's plugin ecosystem for performing the actual PM tasks, which require the same amount of configuration as if they were used conventionally through ProM.

As this project tries to support the vision of PM as a spreadsheet-like technology (see section 3.2.2), the usability and understandability of the employed techniques to users without expertise in PM were some of the focuses during implementation. Therefore, different commercial PM tools were evaluated regarding their suitability for this work. In general, they aim to make PM available to non-technical users like business analysts or CEOs. They usually do not require PM knowledge at an expert level, are designed to work with little configuration and follow the general aim of offering an increased ease-of-use compared to open-source solutions. Apart from that, they often offer additional means of support which are usually not free but can be used to quickly resolve emerging issues without requiring a dedicated PM expert to be hired.

However, many commercial tools can not be used in the context of this project as they are part of a larger, costly business analytics suite and can not be operated as a standalone solution. Another common reason for not being suited for the planned application are products being sold as software as a service model. These are hosted externally by the software vendor and can therefore not be part of a HIS, at least not physically. This leads to data protection issues as medical health data would have to be transferred across HIS borders.

In the end, the commercial tool Disco was chosen for performing the PM analysis. It provides the typical usability advantages of commercial PM tools but offers the option of an academic license that can be used to import up to 5 million events and use Disco's full

functionality for scientific research projects at no cost [57]. This lead to the tool being used for similar research projects that also dealt with KiPs-based PM [3, 128, 160]. The only missing component is the server for automated configuration and import of the event log, called Airlift, which can be replaced by a small amount of externally provided information that documents the required CSV mappings. One of the creators of Disco is Christian Günther, who is also a co-author of the original paper on the fuzzy miner and dedicated his dissertation to the same topic [56, 57, 58]. This is important, as the fuzzy miner algorithm is one of the main means of mitigating the challenge of high process complexity that can be expected for healthcare processes. For the Disco application, the algorithm has been developed further so that it is now able to determine the values for significance and correlation by itself, eliminating the need for an explicit configuration once the event log has been imported successfully [57]. Apart from the mentioned overview statistics that can be used for log and pattern inspection as described in section 4.3, it includes additional interactive visualization features. Examples are e.g. two simple abstractions sliders for refining the generated process models, zooming capabilities, the display of additional information on mouse hover and a very powerful filtering system that can be used for data drilldowns. The interactive visualization features will be explained in more detail in the upcoming section 4.4.2. All these properties, combined with the CSV-based import that does not require a PM expert if the mappings are provided (see section 4.2.5), makes Disco a suitable tool for the use in this project.

4.4.2. Practical Application

The following section will describe the practical application of the Disco PM tool for the analysis of the executed case studies. The process is split in two parts which will be described separately. First, an explanation will be given on how to import the event log into Disco and which additional settings to apply for generating useful process models. Then, some details will be given on Disco's capabilities for interactively viewing the generated models, in order to deal with the potentially high process complexity.

Generating the Process Models

Disco, being a tool that follows the map metaphor wherever possible, calls the visualizations of the discovered process models *process maps*. In order to generate such a process map in Disco, the event log needs to be imported, some filters applied, depending on the selected view, and the resulting visualization needs to be exported again. Importing the event log is done using CSV-files whose structure was previously described in section 4.2.5. All files have the common fields *CaseID*, *Activity*, *ActivityType* and *Timestamp*.

The case ID obviously refers to the individual patients' cases and is used to determine event sequences that belong together. The type of activity is appended to its name and the resulting field is used as activity by the process discovery algorithm. The two fields are not joined on export but only on import to allow higher granularity when interactively

filtering and refining the model (enables e.g. the exclusion of certain types of activities). The timestamp reflects the time when a certain task was finished, either by completing it successfully or by terminating it. The timestamp when a certain task was opened is also recorded but does not provide much extra value as tasks are often opened in bulk by the case execution engine. Therefore, the duration that they are open does not correspond to the time that work regarding the task was actually performed. All four fields are usually picked up automatically by Disco's importer interface. This means that the required mapping is applied by default and even the correct timestamp pattern is detected and used.

Depending on the view, four other fields exist in the exported event log that provide additional information but are usually ignored by the process discovery algorithm. They are however required for certain manual actions and their meaning will be described in the following. Afterward, information regarding their usage will be given. The first additional field, *ClusterGroup*, is added for clustering the event log while still only dealing with a single file. If instead one file was exported per cluster, this would result in roughly 30 files for the stage view, the lowest degree of abstraction. The *Processor* and *Assignee* fields are used to swap PM perspective from the control-flow to the organizational one. The performance perspective does not require additional information as it is calculated based on the given timestamps. The last field, *EventID* is specified so records from the event log can easily be retrieved in Kibana. This is done to enable advanced drill-downs and hypothesis evaluations based on different metrics and was also heavily used while developing and debugging the system. In case the mappings are not automatically picked up, they are also documented in a separate text file that is provided with each exported CSV-file.

To realize the clustering based on a single file, an additional filter is needed that excludes events based on a certain attribute, in this case, the clustering group. The label of the first cluster group is chosen and only events belonging to this group are kept, all others are excluded. An example of the filter configuration view can be seen in figure 4.4. Then, the process discovery is executed, resulting in a process map for this group only. Afterward, the process map configuration is duplicated and the value of the exclusion filter is set to the next group. A process map is generated and the procedure is repeated for every cluster group, resulting in one process map for each cluster. This may seem tedious at first but is much faster than importing and mapping around 30 individual files. Apart from the filters for event log clustering, additional filters are available but those are not required to be applied by default. Rather, they can be used optionally when a domain expert creates a customized view on a process map and considers additional filters to be semantically useful.

The export of the discovered process maps can be done by selecting all models and exporting them in bulk. This saves some additional effort. Unfortunately, however, no such bulk operation is available for importing multiple event logs or splitting a large one up according to the value of some column, making the described manual process for separating the cluster groups a hard requirement. Multiple formats are available for exporting the final, customized process visualizations, like e.g. PDF, PNG or JPG but also text-based formats like XML. Alternatively, the underlying, filtered event log can be exported to dif-

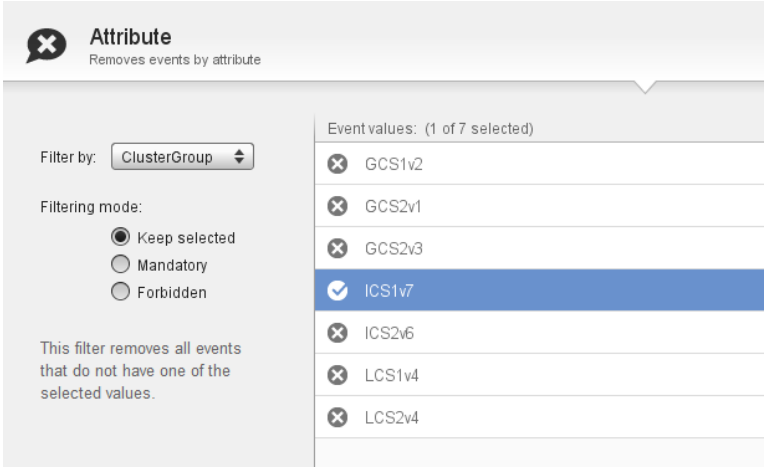


Figure 4.4.: Screenshot of Disco's filter configuration view

ferent formats so a different tool can be used to visualize the data. For this work, the PDF format is chosen, as it contains vector-based representations of the exported process models. This enables the continuous scaling of the models in the PDF version of this paper, which is useful for viewing especially complex models like they are included in this work (see e.g. figure 2.1 showing the spaghetti effect based on real data from the study). After the export is done, one additional step has to be performed in order to create the system view. Because Disco does not allow the customization of colors used in the process models, the exported PDF for the system view needs to be postprocessed and the colorization of the activities needs to be done manually, ideally using a vector graphics editor to keep the image's scalability.

All process maps that are created using Disco are automatically added to the current project. The individual maps and the overall project can be given a name. Apart from exporting the models into individual files, Disco also allows exporting the whole project at once. This persists all process maps including all filters and abstraction settings so they can be restored at a later point. The format they are stored in is proprietary and encoded in binary which can lead to a vendor lock-in in the long term. However, this is the only way to share the interactivity settings that are unique to Disco and the possibilities for collaboration that are enabled by the feature outweigh the disadvantages and risks.

Interactively Viewing the Models

As just mentioned, Disco offers some unique features for interactively viewing the discovered process maps. Most of them are included in Disco's main view, the *Map View*, which shows the visual representation of the event log using the given customization settings. Most settings can be easily adjusted in the same screen and the displayed process

map is updated immediately to enable a WYSIWYG[6]-style model creation. The view and noteworthy details are depicted in figure 4.5. The individual elements providing viewing interactivity are explained in detail in the following.

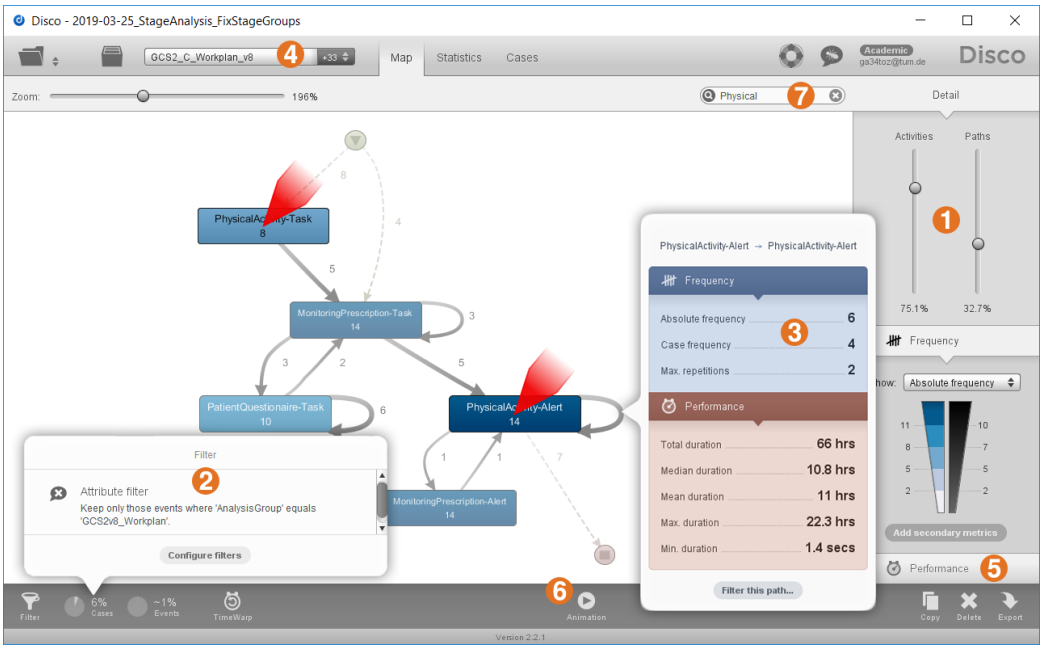


Figure 4.5.: Screenshot of Disco's Main View

1 Abstraction Sliders The most prominent feature of Disco's interactivity system is the abstraction sliders in the upper right corner. They do not reduce the significance or correlation threshold, but rather refer to the number of activities and paths (transitions between the activities) that is displayed at a time. Less frequent activities or paths are hidden when a slider's value is reduced. In combination with the zoom slider, the controls can be used to increase the understandability of complex process maps by refining the level of abstraction depending on the structuredness of the modeled process. The scales used by the sliders are kept in percent, making their value easy to understand by non-PM-experts.

2 Filtering System The next very important feature is the interactive filtering system which can be accessed in the lower left corner. The available filters are very powerful and configured on a separate screen that can be accessed using the funnel icon and that also includes some filter suggestions based on the characteristics of the data in the event log. Next to the icon, an overview is shown of how much percent of

[6] "What You See Is What You Get"

the total number of cases and events are left after the filters have been applied. Upon click, the currently configured filters are displayed in a shorthand form. All filters can be configured to include or exclude matching elements and users can decide if the filter should apply to whole cases or to individual events contained in different cases (e.g. exclude certain events or include all cases containing such events). Creating filters can either be done from the separate filter configuration screen or by selecting an activity or a path in the map view and then creating a filter based on the selected element. Six different types of filters are available: (1) timeframe-based filtering, (2) filtering of process variants, (3) filtering based on performance metrics, (4) endpoint-based filtering (e.g. for excluding incomplete cases), (5) filtering based on certain attributes and (6) filtering of specific event sequences. Collections of filters can be exported to so-called *recipes* that can then be used in other process models or other Disco projects and can even be shared with other users.

3 **Dynamic Display of Additional Information** A lot of statistical information on activities and paths can be generated from an event log, e.g. the number of cases that feature a certain activity or the number of times that a path (i.e. a sequence of two events) was observed in a single case. Depending on the analysis context, different information may be relevant or not relevant at all. To prevent overloading the user with information and obfuscating the process map, the software hides most of this information by default and only the absolute element frequencies are displayed. When a map element is clicked on, the available additional data is displayed in a pop-up menu that also allows creating a filter for the element.

4 **Quick Context Switches** Two different ways are available for switching the current analysis context and both are included in the top of the application. The three tabs labeled *Map*, *Statistics* and *Cases* can be used for switching between the process map and less graphical views on the event log based on some calculated metrics. The statistics view is used for log and pattern inspection and shows exhaustive overviews information like the number of cases and events but also more detailed indicators like the frequency distribution of individual activities and additional data attributes. The distributions are included as graphs. The cases view is the most detailed view and offers a look into the raw event data. It shows the actual execution traces for a single case and tries to place the case in the overall context. Apart from that, it displays similar cases together in a group, thereby showing the results of the variant analysis that is performed during import. The aforementioned interactive filtering features are available in all three views, not only in the process map screen. The dropdown menu left of the tabs can be used to quickly switch between different process maps. This is useful as the copy functionality allows to create deep-copies of a process map, enabling the quick creation of multiple views on the same process, which can then be shared with colleagues or be compared to each other using the dropdown menu.

5 Multiple PM Perspectives When an event log is loaded into Disco using the default recommended mapping, the application generates a process model in the control-flow perspective. However, as this requires timestamp information, Disco can additionally calculate the metrics for the performance perspective. The lower right section can be used for quickly switching between both perspectives. For both perspectives, the exact metrics to display can be chosen based on the current use case. Options include the total, maximum, minimum, average or median duration for the performance perspective and the total frequency, the frequency per case or the maximum number of repetitions for the control-flow perspective. Being able to quickly switch between both perspectives is especially useful, as the frequency of activities and paths is highly relevant for the understanding of average values and a meaningful bottleneck analysis that leads to influential improvements. This is also facilitated by the possibility to display an additional metric from the performance perspective in the control-flow perspective and vice-versa. An automatically adjusting color scale is available that shows the configured frequency and performance metrics in a graphical way.

6 Animations Using the button on the bottom middle of the screen, animations can be created that graphically replay the event log on the generated process map. Activities light up briefly and slowly fade out afterward when they were performed and case indicators travel across paths from one activity to the next activity. This can be used to visualize the time-based sequence of activities better than a static image could. It is also a nice way to evaluate the degree of concurrency the cases were performed at and provides insights into things like peak system usage time. The animation speed can be adjusted and the final animation can be exported to a `.avi` file.

7 Search in Process Map A search bar is available in the upper right corner that can be used for finding activities by their name. Results are highlighted by zooming in on them, graying out irrelevant parts of the process map and pointing big red arrows on the matched activities. This is helpful for domain experts to navigate and understand large, complex maps but can also be used for speeding up the filtering and customization feature.

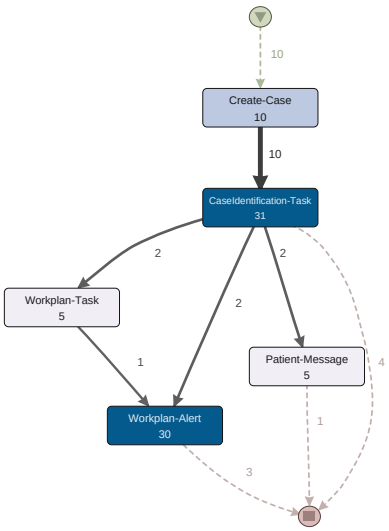
4.4.3. Notation of Generated Models

The visualizations Disco produces are called *directly-follows graphs* because they are graphs where each transition from one node to the other represents a directly-follows relation as also known by e.g. the α-algorithm [88, 89]. The syntax is quite simple as the graph consists of nodes connected by edges and both elements are annotated with one or two metrics. However, the semantics vary depending on the input data, the current PM perspective and the chosen metric. Animations, if viewed correctly, can further increase the understanding of the process. This section will explain the graphical notation and the semantic meaning

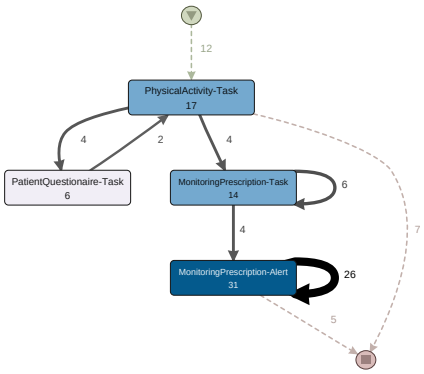
of the different types of models generated in this work. First, the traditional notation of a directly-follows graph will be described and afterward, different perspectives and notational extensions will be discussed.

Reading a Directly-Follows Graph

The default models produced by the case and the stage view are exported from Disco without any modifications and contain regular directly-follows graphs for visualizing control-flows. They show the sequence of activities as they are usually executed. Process map examples for the case and the stage view are depicted in figure 4.6. As mentioned before, directly-follows graphs consist of *nodes* representing the activities and *transitions* between them representing directly-follows relations [88, 89]. Apart from that, they include two special types of nodes, the *start* and the *end node*. Each element is annotated how many times it was observed in the event log. When abstractions are applied to the graph, as usually happens in the context of KiPs, these numbers do not add up anymore, i.e. activities occur more often as there exist in- and/or outgoing transitions. Therefore, a directly-follows graph cannot be used to tell process-conformant from non-process-conformant behavior but only shows the sequence of events as it was usually executed. The individual elements will be described in the following, giving details on their semantic meaning when employed in either the case or the stage view.



(a) Case view model



(b) Stage view model

Figure 4.6.: Simplified examples of process maps in case and stage view

Activity (Stage View) In the stage view, an activity signalizes that some explicit task was performed for a certain case. This could be the filling of a questionnaire, the execution of an image-producing procedure or a prescription of a certain amount of medication or physical activity. Another possible type of event that can be contained in a stage model are alerts from the workplan stage. Alerts represent exceptional situations where a certain pre-defined threshold was under or overrun. The threshold is defined for the individual workplan tasks (like prescription or physical activity) and incoming measurements are checked against it by the SACM system which also emits the alerts at midnight if necessary. When a professional takes note of the alert and marks it as seen, this activity will be included in the event log and the final process map. The number below the name of the activity signalizes how often it was observed across all cases. This frequency is also visualized using a color scale: the stronger the color of the node, the more often it occurred in relation to all other activities.

Activity (Case View) In the case view, however, tasks from multiple stages are included. To keep complexity low, the view abstracts what the tasks represent in detail and only treats each task as some not further specified activity that took place in a certain stage. Alerts are treated the same way and only associated with their stage, not with their exact task. Apart from these two types of activity, the case view also contains activities for received messages from patients or other professionals or for when the unstructured notes of a case were updated. The case creation and case termination activities are also included, showing when a certain case was created or terminated in the SACM system. Terminating a case prevents any further modifications to the tasks and stages of the case, so the ratio of case termination to creation indicates the share of cases that are still running.

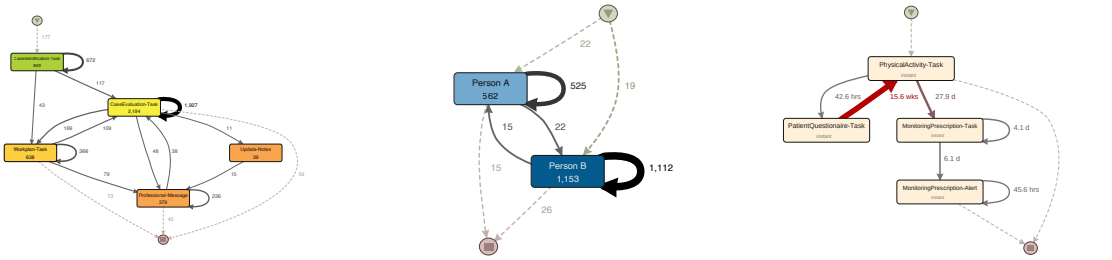
Transition A transition from one activity to another means that the two elements were observed to be directly followed by each other with the tip of the arrow pointing to the activity that came second. The number next to the transition indicates how often the directly-follows relation between the two activities was observed. It is also indicated visually by varying the thickness and blackness of the transition arrow. As mentioned, when abstractions are applied and some transitions are hidden, the numbers do not add up anymore which is not an issue if it is known by the reader. Additionally, the ratio of the transition count to the count of the activities still bears relevant information, even if some other transitions are not displayed. Self-transitions reveal activity repetitions as they only occur when the same activity was observed multiple times subsequently within a single case. In contrast, cycles in the graph, often called *rework loops*, only suggest that the illustrated activity sequence could have been repeated but cannot be used as a proof, especially not when dealing with KiPs. This is because every individual transition could have been observed in a different case but the whole of the cycle has never been observed in a single case at all, meaning

the rework loop is included in the map but was never actually executed. The high flexibility of case executions in the context of KiPs makes this much more likely than when dealing with more structured processes.

Start & End Nodes The green and red nodes at the top and the bottom of the process maps are start and end nodes, respectively. They are artificial events added by Disco, not contained in the event log and signalize the start and end of the cases. The transitions between the start node and the observed activities indicate which ones were the starting activities that were first observed across all cases. The same principle applies to the artificial end node. The numbers next to the transitions again show how often the transition was observed, i.e. how often a case was started/ended with a certain activity.

Understanding Additional Data Perspectives

Three additional perspectives on the data are available: the performance and the organizational PM perspective and the customized system view. How to read them and differences to the already known process maps will be explained in the following paragraphs. Simplified examples for all three are shown in figure 4.7.



(a) System view model

(b) Organizational perspective

(c) Performance perspective

Figure 4.7.: Simplified examples of process maps in system view, in organizational, and in performance perspective

The **system view** can be read similar to the case view as it also shows a process map from the control-flow perspective, but it does so at a slightly higher level of abstraction and without any data clustering. To increase its usefulness, especially when compared to the two other views, additional information was added by coloring each activity according to its structuredness. This visualizes the structuredness of different parts of the overall treatment process and shows their interrelationship. The chosen colors correspond to the ones known from the spectrum of structural degrees according to Di Ciccio et al. which is depicted in figure 1.2. Every type of activity can be assigned to such a structural degree.

For activities indicating work performed in a certain stage, the structural degree depends on that of the stage. The structuredness of all modeled stages was described in section 1.2.7 and can also be seen in figure 1.5. Alerts represent unexpected measurements, messages from patients or other professional indicate insecurities regarding the process execution and notes are intended for persisting unstructured information that turned out to be important but was not included in the system model. Therefore, these types of activities are exceptions to the usual process execution and are colored as unstructured process activities in dark orange. On the other hand, the case creation and termination activities are actually technical procedures in the SACM and are therefore highly structured and colored in green.

When using Disco, every process model can be viewed from the **performance perspective**. Activating it switches all metrics from frequency-based to performance-based and changes the color scheme from blue and black to ocher and red. Different metrics like minimum, average or median are available for the user's choice but they are not really useful without considering the observation frequency they are based on, which can optionally be included and is then given in brackets as the second number. In the performance perspective, the primary number next to a transition indicates the time it took for the second activity to occur after the first activity had occurred. Activities also have a duration, however, for Disco to be able to calculate that, timestamps for start and end of the activity need to be given, which is not the case for this project as the time when the actual work on a task was started is not valid in most cases. Therefore, all activities across all process maps included in this work will have the duration *instant*. Self-transitions indicate how long it took for subsequent repetitions to occur.

The last perspective is the **organizational perspective**. It is very similar to the default stage view and depends on the same dataset. It does, however, not show what work was done in the process but rather who performed the work. The stronger a self-transition is in the organizational perspective, the more the person is working alone. However, a transition from one person to the other means that the work on a case was handed over and the second person is now performing it [162]. If such a transition also exists the other way around, this suggests that the two persons work together, e.g. by delegating tasks, exchanging messages or by swapping cases. The start and end activities show who started the cases and who last performed some work on them. Like usual for Disco's control-flow perspective, the frequencies of activities and transitions are indicated by a number but also by coloring and emphasizing.

4.5. Deployment for Continuous Data Collection

4.5.1. Artifacts to Deploy

For deployment and operation, the whole elastic stack is packaged into a single Docker[7] container. While this will have to be changed to enable proper scaling in the long run, it also allows for the simplified integration into the existing environment which is already deployed using Docker Compose[8]. Using only one additional container centralizes all necessary configuration and thereby keeps deployment complexity low. The compiled container is pushed to Docker Hub[9], a public online container repository that can be used for easily distributing containers without access restrictions. Documentation on how to integrate the container into an existing `docker-compose.yaml` file is also provided on Docker Hub, further centralizing and simplifying the deployment. Most other artifacts of the CONNECARE project are also distributed via this mechanism. This is safe to do because the container itself contains no information relevant to data privacy protection purposes.

Database connection strings, credentials and other configuration data can be passed to the container via environment variables. The variables primarily include the configuration data required to access the three databases as well as the username and password that is required to access Kibana's web UI. Apart from that, they also contain means for disabling Logstash's export or preventing the application from starting altogether. This allows for read-only deployments which is useful during development and debugging but can also be used to reduce the system load in a productive setting (e.g. after the case studies have finished). The feature was also used during the database migration from MongoDB to MySQL. An example of a `docker-compose.yaml` file can be seen in listing E.1. The listing also shows the available environment variables.

The container is based on the public `sebp/elk` image[10] with some minor adaptions to its startup logic and cluster configuration as well as a large amount of customized Logstash pipeline scripts [123]. The image is available as open source release at Github[11] and some fixes with general applicability were contributed to the project[12]. The image is up-to-date with the latest version 7.0.1 of the elastic stack that is also used in the context of this work. Additionally, it provides a means for selectively starting individual applications. This enables the use of one single, common image to realize a distributed elastic stack where every application is running isolated in its own container. However, as no usage of the feature was planned in terms of this work, this was not tested and is expected to require some extra development effort, especially regarding cluster orchestration configuration.

[7] https://docs.docker.com/
[8] https://docs.docker.com/compose/
[9] https://hub.docker.com/r/connecare/sacm.analytics/
[10] https://hub.docker.com/r/sebp/elk/
[11] https://github.com/spujadas/elk-docker
[12] https://github.com/spujadas/elk-docker/pull/275

One drawback of using the open-source elastic stack for preprocessing the event log is its lack of a proper user authentication system. Using a single, Docker-based deployment, however, mitigates this issue as it enables the use of a reverse proxy for enforcing authentication before the request even reaches the elastic stack itself. Therefore, the image was extended by an nginx instance that acts as a reverse proxy in front of Kibana's web UI. It is configured to require HTTP Basic Authentication but more sophisticated security mechanisms like e.g. *ModSecurity*, a web application firewall, are also available. The credentials that nginx expects are configured every time the container starts to enable changing them via the environment variables and applying the changes using a simple container restart without having to recompile the image. Nginx currently only accepts connections on HTTP's port 80, as the whole CONNECARE environment uses HTTP for internal communication and the external HTTPs connectivity is only made possible through an additional proxy that acts as a gateway and cares about translating between both protocols. The only exposed port of the final container is the one where nginx is listening on. If the correct credentials are given, requests are passed on to the Kibana instance, otherwise, they are rejected.

4.5.2. Performing the Deployment

The deployment was performed quite early in the process to enable access to the real data stored in the productive environment. This was the easiest way to mitigate data privacy protection issues and avoid transmitting large amounts of raw data sets. Other reasons were the strict resource constraints that existed in the productive environment like the limitation to a maximum memory usage of 4GB which is also the minimum amount of memory required to start the combined elastic stack container. To ensure the stack works properly without negatively impacting the overall system performance and stability, the decision was made to deploy a prototype as early as possible. A complete mirror of the productive environment is available for testing purposes. It was used for evaluating that the deployment worked and testing out various settings before moving on to the productive application instance. Both deployments did not encounter any problems, except for the necessity to increase the `vm.max_map_count` on the productive Docker host to enable the start of Elasticsearch. This is required as Elasticsearch internally uses a `mmapfs` memory-mapped file system for storing its indices and easily exceeds the limit to virtual memory address space typical operating systems set [31]. Why the issue only occurred in the productive environment is unknown. The initial import of the at the time around 3.5 million API access logs on the productive system took roughly 90 minutes.

One special activity which had to be done during the first deployment was the migration of the MongoDB database to a MySQL one, which had to be done as MongoDB does not support the JDBC interface. Additional reasons are given in section 4.2.2. For the migration, most other applications were stopped to free up system resources and let the migration process itself happen as fast as possible. The migration was performed using a RESTful API endpoint that was created for this sole purpose and offered the possibil-

ity to specify the values for the insert batch size and the number of concurrent database connections to use. The endpoint also supported specifying values for the SQL LIMIT and OFFSET parameters, which allowed the migration to happen in chunks of 1 million records and prevented the process from completely starting over in case an error was encountered. Before the actual migration was started, some smaller ones were executed in the test environment in order to find ideal values. Afterward, the process was triggered in the productive environment using an insert batch size of 250 records and 10 concurrent database connections. Logs were imported in chunks of 1 million. The API endpoint also featured a simple timestamp-based security mechanism that included a timestamp in every request that had to be incremented by one for the next request to be valid. This was done to prevent the accidental triggering of the migration process and, besides its quite simple design and implementation, turned out to be quite useful during the database migration.

When the historical data was imported into Logstash for the first time, some data quality issues in older log records became apparent that were due to e.g. past logging implementation errors that were already fixed more recent versions of the application. To name an example, the source IP field of requests temporarily included two IPs (the proxies and the actual source IP) which lead to issues when trying to import those roughly 350 records into Logstash. The issues were corrected by executing individually created SQL scripts and thereby permanently correcting the log records in question. In total, around 11 000 records out of the 3.5 million records dataset or 0.3% were affected by such data quality issues that had to be manually corrected once.

Part III.

Results and Conclusion

5. Evaluation

As explained in section 1.2.7, the performed case studies and the underlying case models are derived from the work of Cano et al. where an abstract model is proposed for supporting integrated care services [15]. Multiple stages exist with varying degree of structuredness as defined by Di Ciccio et al. (see also figure 1.2) [29]. The developed case models can mostly be expressed in CMMN. All case models used in the executed studies are included in the appendix, starting at chapter A. The stages of the models are colored according to their degree of structure.

This chapter will provide answers to the three initial research questions. For this purpose, first, the case models will be analyzed to find elements and properties that affect aspects of research interests (e.g. case model elements enabling flexibility like repeatable tasks). Afterward, for each site and case study, the findings from the log and pattern inspection step will be presented and the first hypotheses will be developed. These will then be checked and refined using PM and the generated models. The last two steps are repeated for every site and study. Finally, a conclusion is drawn that tries to answer the posed questions in a concise way. This approach sets the focus on the difference between the theoretical case model and the actual case execution activities observed in the log. It is repeated for every research question.

5.1. Usage of Model-Provided Flexibility

5.1.1. Flexibility in Case Definitions and During Case Execution

To produce meaningful insights and focus the evaluation efforts on relevant parts, the case models need to be checked for elements that allow for flexibility. Different CMMN elements can be employed to enable extra adaptability or introduce additional structure. Tasks are aggregated into stages that represent different logical phases that the case undergoes. Sentries can be used to express dependencies between tasks, requiring one task to be completed before the next one can be started. They can also be used to express dependencies between stages, requiring all mandatory tasks of the stage to be done before the follow-up stage can be activated. As just mentioned, tasks can be required or optional, with the latter one still allowing a stage to be completed, even if some (optional) tasks were not yet completed. The same applies for stages which can also be required or optional. Undone but required tasks prevent users from successfully completing a case. The more structured stages like the case identification stage bear a high amount of sentries

while the less structured stages typically don't. Tasks and stages can also offer the possibility of being repeated, either serially after the previous instance has been completed or in parallel where multiple instances can be started at the same time. Both variants require manual activation by the users, at least according to the case models developed for the two CONNECARE case studies.

The inter-stage sentries are employed in a similar way across all case models. With only one exception, the case identification stage is the first stage to be executed and is a prerequisite for all other stages. This is not true for Lleida's first case study, where the final discharge stage can be activated at any point in time. The evaluation and workplan stages can be done in parallel in all case studies. When taking the historic case model version into consideration, Groningen is an exception to this observation. For both Groningen's case studies, the first versions of the case model feature a strictly linear process that requires the evaluation stage to be done before the workplan can be created. This was however changed very soon and the other sites which started modeling some time after Groningen apparently oriented at this change and modeled the two stages as parallel ones, to begin with. Depending on the exact model version, the discharge stage can either be started in parallel with the other stages, after the workplan stage has been completed or at any point in time.

5.1.2. Log and Pattern Inspection Findings

To get an impression of the employed execution flexibility, let us first have a look at the numbers known from the log and pattern inspection before taking a closer look at the process models. The numbers are contained in the final GQFI table that is fully evaluated and can be found in the appendix, in section C.1. For the flexibility analysis, both the stage view and the case view have been considered.

Groningen

In Groningen's **first case study**, tasks from only three of four stages were activated and the structured discharge stage was never executed. The discharge stage is modeled as required on a case level, so no case was actually completed and all opened cases are still running.

The identification and evaluation stages are not flexible but rather structured which shows in the lack of manually activated tasks and in the low quota of optional tasks (0% and 33% respectively). The identification stage was completed successfully for all 25 cases which is expected as it is required for any further action. Even though the order of three tasks could have been swapped according to the case model, all cases were executed the same way and therefore posses the same task order. Only one process execution variant was observed for all 25 cases.

The evaluation stage was only executed for a single case and only one activity was performed. A look in the case model shows that all tasks in the evaluation stage require setting an evaluation date before starting the task, so this was the one activity that was

performed and no actual work was done in a medical sense. Most COPD-related tasks are contained in the evaluation stage, as both Groningen's case studies share most tasks that are contained in the other stages. This shows a low usage of the domain-specific tasks, at least in the evaluation stage.

Out of the 25 initial cases, only 20 have some activity in the workplan stage. Both workplan definitions are exactly equal and contain five tasks, however, only two activities were performed, limiting the maximum amount of different tasks that were performed to two, provided that no alerts have been received. A detailed pattern inspection shows that this is not the case and only tasks and alerts for *PhysicalActivity* were received. Still, in spite of the limited amount of activities, 5 different variants exist for the 20 tasks. The most prominent variant is supported by 75% of cases and consists of only one task and no alerts. In turn, this means that 75% of cases consist of only one task in the workplan stage which explains the median stage duration of zero. The duration of a task is calculated based on the end times of two subsequent events, resulting in zero duration if only one task exists.

When examining the data in a more abstract way, from the case view, it becomes apparent that the quota of manually activated tasks is the lowest across all sites and both case studies. This suggests an under-usage of the more flexible stages which is supported by the fact that this is the only study that has fewer variants than case studies, meaning some case executions are non-unique. The study generally differs from the other studies with regards to the overall activity. It has the lowest median case duration with around 14 days which is half as long as the other median case durations. Apart from that, it showed the lowest amount of completed tasks, falling behind after all other cases by more than a factor of 10 (n=123). However, the case studies are still running and one can expect the study to catch up with those at the other sites.

When considering the identification and evaluation stages of the **second case study**, a very similar picture emerges. All 35 cases executed the identification stage in exactly the same way, with the exception of a single case that employed a different task order. The identification stage is quite short, with a median duration of around 75 seconds, which can be explained by looking at the contained tasks: all of them are quite simple, don't take much time and only require the user to enter some simple data or choose between a low amount of options. Like in the first case study, the evaluation stage was only executed for one case with a single task, resulting in questionable validity. Nonetheless, the workplan stage is quite active and shows a high degree of flexibility with 100% manual activations, 100% optional tasks and on average around 3.5 possible outgoing paths (i.e. different follow-up activities) per activity. What shows the employed flexibility even stronger is that each stage execution has its own process variant, meaning no executed workplan is like the other. This also leads to multiple bidirectional relations between two activities (n=5). That means there is no strong precedence which of the both comes first which is again an indicator of process flexibility. Like in the first study, the discharge stage was not yet executed.

From the case view, the second study shows a very high level of manual activations of 86% which suggests an under-usage of the structured and partly structured stages. When

keeping in mind that these more automated stages make up a large part of the case model and the available tasks, a discrepancy between modeled expectation and practical observation becomes apparent. Still, as in the first study, the CONNECARE project is still running and the numbers can change as more patients are admissioned towards the end of the project. For the 35 cases that were recorded to follow the second study, none is like the other and 35 different process variants exist. This shows that the provided flexibility is actually employed. The median case duration of more than 50 days also seems more reasonable and more like a productive use than that of the first case study. However, the duration must be taken with a grain of salt, as all cases are still running and the number only describes the time between the creation of a case and the last observed activity.

Tel-Aviv

With around 3 minutes, the **first case study** like it was executed in Tel-Aviv had a higher median duration for the initial identification stage compared to Groningen with 30 seconds. This is likely due to more complex tasks that have to be executed in Tel-Aviv like *Lace* or *InclusionCriteria*. While the earlier versions of the stage allowed for a fair amount of flexibility with only 2 sentries for 5 possible activities (see figure A.7), the most up-to-date version of the case model contains 4 sentries arranged in two levels for the same five activities of the identification stage (see figure A.8). This enforces additional structure in the most recent versions of the stage which can also be seen in the log and pattern inspection indicators. The versions prior to 8 contain a higher number of paths between activities (n=9 vs. n=4), a higher number of bidirectional paths (n=1 vs. n=0) and the 34 cases distribute on 5 process variants. The versions equal or greater than 8 have been executed 17 times but all executions have been done the same way and no process variants exist. This nicely shows how the CMMN elements for enforcing structure mentioned in section 5.1.1 actually do so and how leaving them out creates executional flexibility and complexity. However, when compared to Groningen's first study, all versions from Tel-Aviv still bear a higher complexity and flexibility even though the amount of sentries is higher.

The evaluation stage was executed for 47 out of 51 cases. It does not allow for any adaptability as it contains no manually activated tasks and all tasks are required for the completion of the stage but it still allows for a high amount of flexibility regarding the order of the tasks. It has a high amount of paths (n=101), a high amount of paths per activity (n=5.6) and is, with a median duration of around 46 hours, usually executed across multiple days. For the 47 executions, 34 different process variants exist and the one with the highest support is followed by 20% of the cases. This shows that most cases are unique and only some follow a common path, unlike the executions of the identification stage where most cases follow the same path but some exceptions occur. In the evaluation stage, deviating from the common variant is already the norm. Upon further investigation of the process variant with the highest support, it is discovered that the order in which tasks were executed is equal to the order of tasks in the web interface. This suggests that clinicians follow the order proposed by the system except they have a reason not to do so.

The workplan stage is fully adaptable as it consists only of optional and repeatable tasks that have to be activated manually. Besides its low amount of tasks (n=6), it has a similar amount of paths per activity as well as a likewise high amount of bidirectional paths (n=10) and therefore bears a comparable amount of flexibility. One could even argue that while the overall stage complexity is lower due to its reduced amount of tasks, the contained flexibility is actually higher as the 42 workplan executions can be grouped in 41 process variants. The variant containing two cases consists of only two initial tasks, no other activities and have been started recently, therefore the alignment of the cases is likely due to the low complexity that exists at the beginning of treatments. The stage lasts for a median duration of 57 days which makes up the largest part of the overall median case duration of 68 days for Tel-Aviv's first study. This means patients spend the main part of their clinical journey in flexible stages but each case starts with more structured processes and stages. Such behavior is expected and the CONNECARE project tries to exploit this by enabling patients to perform the more flexible workplan stages from home and save a big part of their stay at the hospital, increasing their quality of life.

Like in Groningen, the discharge stage has never been executed which is not surprising as the case studies are still running and the subjects are CCPs that have multiple chronic conditions and cannot be treated short-term. When viewing the overall case, it shows a value of 60% manual activations and optional tasks. As expected after the high number of variants for the workplan process, the number of variants of the overall case is maximized at 51 variants for 51 cases, showing a high amount of flexibility. This can also be seen in the high standard deviation regarding the overall case duration (around 42 days deviation from average 63 days duration). On the case level, the study has a higher amount of activities than Groningen's first study which indicates a more wholesome use of the CONNECARE system and the SACM's features.

In the **second case study**, the identification stage was executed very similarly to that of the first case study but slightly less flexible. This shows in e.g. the lower amount of paths per activity (n_1=1.5, n_2=0.75 vs. n_1=1.8, n_2=0.8). However, the effect is not very significant as the stage's flexibility is the lowest across all stages anyways. Apart from that, the low number of activities contained in the stage allows for less flexibility (and complexity in general) as well.

The evaluation stage differs from the one from the first study in that it contains a higher number of activities and a higher number of paths but a lower number of paths per activity. This means that potential complexity is higher in case of the second study while actually observed flexibility is slightly lower. This is countered by the fact that there are more execution variants in the stage of the second study (n=26 for 29 cases) which generally indicates flexibility. On median and average, the stage is also significantly longer than its counterpart and has a higher standard deviation which also indicates flexibility. These contradictory facts regarding flexibility could be explained by the low amount of executions, especially when compared to the first case study (n=29 vs. n=47). Overall, the stage seems more complex and similarly flexible to the one from the first study but this is subject to change when more patients are admissioned which will also increase the reliability of

the findings.

The workplan stage has the same amount of activities like the one from the first case study, slightly fewer paths overall but more paths per activity, indicating a more flexible usage. Again, the effect could be slightly reduced when more patients are admissioned and a comparable number of patients is treated in both studies. Almost every workplan is unique and creates an own process execution variant. The duration of the stage is also quite similar across studies, only the standard deviation is a bit higher in the second case, which could also be mitigated by the admission of new patients. Overall, the similarity between the two stages in both cases is actually expected, as when considering the case models, no differences regarding the stage can be found, i.e. the stage definitions are basically equal.

From the case view, way fewer cases were executed in the context of the second study than in the context of the first one (n=29 vs. n=51). If that is taken into consideration, no definitely significant differences between the two case studies can be found. This can also be seen in the case models which are exactly equal in large parts. The only exception is the contents of the evaluation stage which change according to the planned system usage. However, the changes are of domain-driven nature and primarily affect the identity and the contents of individual tasks but not such much the overall stage and case execution.

Lleida

The **first case study** in Lleida features an identification stage that is quite similar to its equivalent from Tel-Aviv. With the stage being very structured and the stage models of both sites being quite similar, this is expected behavior that also reflects in the numbers captured for the GQFI table. There are no significant differences between both sites with regards to the number of cases (n=35), the activity count (n=5), the average number of paths per activity (n=1.6) or the number of bidirectional paths (n=1). For 35 cases, there exist only 3 process execution variants, with the largest one spanning 77% of cases and following the task order displayed in the web frontend. This lets the stage seem a little less flexible when compared to the one from Tel-Aviv, which is surprising, as Lleida is the only site that features optional tasks in the identification stage, which are also actually executed in some cases (n=41). The median stage duration is slightly higher for Lleida (n=5.4 minutes) but the standard deviation is less (n=19.7 minutes). This suggests a more homogeneous patient group than in other studies but could also be related to other aspects, e.g. of cultural nature. Unfortunately, this hypothesis cannot further be investigated and confirmed in terms of this work due to data protection regulations.

Analogous to the finding from the identification stage, Lleida is the only site that also employs optional and even manually activated tasks in the evaluation stage. For the first case study, the share for both types is around 18%. The activity count is also quite high with 22 different activities that were observed during the executions. The same is true for the indicators for the overall number of paths (n=185), the number of bidirectional paths (n=57) and the average number of paths per activity (n=8.41). The mentioned val-

ues are actually global maxima across all sites and studies, making the evaluation stage of Lleida's first case study the most complex stage as well as the one with the most flexible usage. However, the adaptability is greater in all workplan stages, based on the low amount of 18% manual task activations compared to 100% manual activations in most workplan stages.

As in the already considered case studies, the workplan stage features a manual task activation quota of 100% which also applies to the share of repeatable and optional tasks. With 34 executions, all but one case performed activities in this stage. It features the highest number of activities across all workplan models (n=12) which results in a similarly high amount of overall paths (n=57) and the highest number of bidirectional tasks (n=15). Nevertheless, the average number of paths per activity is also high but lower than those of the Tel-Aviv workplans (n=4.75). With a median duration of 47 days, the stage duration is quite similar to the duration of the stage at other sites. All in all, the stage offers the highest potential for complexity and flexibility, though only some flexibility aspects are actually employed. Still, the stage is amongst the most flexible ones which also shows in the lack of common process variants.

Lleida is the only site that actually ever executed activities from the discharge stage. Roughly a third of all cases have some completed discharge tasks (n=13). The stage consists of only three activities which were completed in a highly flexible way, as 5 out of the 6 possible paths are included which leads to an average number of paths per activity of 1.67, a quite high indicator given the structured nature of the stage according to the case definition. That the stage is structured can be seen at the comparably high number of process variants (n=5 for 13 cases) and at the existence of two bidirectional paths out of a maximum of three.

When viewing the case study as a whole, the similarity to the Tel-Aviv studies becomes apparent once again. Lleida's median case duration is quite a bit smaller, taking only 49.3 days instead of roughly 65 for the Tel-Aviv studies. Apart from that, the numbers are very similar with a slight increase in flexibility observable in Lleida's system usage. It shows e.g. in the average number of paths per activity (n=4.8 instead of n=4.4), the number of bidirectional paths (n=16 instead of n=10) or the higher quota of manually activated tasks (n=61% instead of around 44%). The standard deviation of the average case duration is around 45 days for all three case studies.

Amongst all executed case studies, the identification stage of the **second case study** in Lleida has the highest degree of execution flexibility. It features the highest number of possible activities (n=6), executed paths (n=15), paths per activity (n=2.5), bidirectional paths (n=5) and process variants (n=7 for 35 cases) across all identification stages. The median duration of the stage is high but comparable to the other case studies (n=4.7 minutes). However, the mean duration (n=9.7 hours) and the standard deviation (n=55.6 hours) are exceptionally high which shows that the system was used in a more flexible way than in other case studies. The variant with the highest support applies to 57% of all cases, which shows that while there are many exceptions and high flexibility, there is still a common path that can be followed for simple cases. As already observed earlier, this biggest vari-

ant follows the order displayed in the frontend.

The evaluation stage, when compared to the first case study in Lleida, is less complex and less flexible used but still has higher flexibility indicators than the other sites. Like in the first case study, Lleida also offers the possibility to activate some additional evaluation tasks manually. Professionals made use of this possibility, though at a lower rate of 6%. There are multiple versions of the stage, with earlier ones offering one more task than later ones (n=24 and n=23) but not all tasks were ever executed (n=23 and n=21 respectively). The later versions also have lower execution flexibility in general, even though more cases were executed using this version (n=19 instead of n=15) and both stages feature one process variant for each case execution. The duration of the second version is lower as well (n=24 days vs. n=66 days) and the same goes for the standard deviation (n=26 days vs. n=53 days). Combined with the fact that there are hardly any differences between median and average duration, the data suggests that the later versions simply had a more homogeneous patient base.

The biggest specialty of Lleida's second case study is the workplan stage which is split into the two parts *workplan before hospitalization* and *workplan after hospitalization*. The workplan before hospitalization stage is used for prehabilitation as described in section 1.2.7. Similar to the workplan after hospitalization stage, which is employed for rehabilitation and similar to the generic workplan stages at the other sites, it features tasks like *Monitoring Prescription* that can be used to monitor e.g. the physical activity of a patient prior to surgery while being still at home. According to the case model, both workplan stages are unstructured stages with pre-defined fragments just like the generic workplan stages in the other case study executions. The workplan after hospitalization stage behaves very similarly to the workplan stages from other case studies. However, the flexibility of the execution of the workplan before hospitalization stage is less than that of the one after hospitalization. This can be seen as for the former stage, only half of the executed tasks were activated manually (n=54%), the rest was activated by default by the SACM system. In comparison to the other stage's close to 100% manual activation quota, this is quite low. The difference also shows in other indicators. While the later stage has only unique workplan executions, the earlier stage was executed in a similar way for multiple patients, i.e. there exist fewer process variants than stage executions for the stage that comes before hospitalization (n=23 for 32 cases). The stage duration is also significantly lower, with the workplan before hospitalization stage being more than half as short as the latter stages. Apart from that, the average number of paths per activity is lower than for the workplan after hospitalization stage (n=2.57 vs. n=3.78 or n=4.77) which also applies to the stage duration (17 days vs. 30+ days). Nonetheless, the standard deviation is higher which may be explained by the fact that the before hospitalization stage is started – as the name says – before the patient arrives at the hospital. This requires the stage to be active for a longer duration, especially as there is no direct benefit from completing the stage, as it does not serve as a sentry to another component.

The discharge stage was also executed in Lleida's second study. Out of the 35 overall patients, 13 cases feature tasks from the discharge stage. This means around 35% of pa-

tients that have been admissioned, have already been discharged again. The other 65% are still undergoing treatment or monitoring. In comparison to the other execution of the discharge stage in Lleida's first case study, the flexibility regarding the execution is even a bit higher, with the maximum number of 6 paths between 3 activities, making every path a bidirectional one. This results in 5 different process variants for the 13 cases, leaving room for one additional variant until all combinational possibilities have been observed. This leaves the stage at a very high degree of execution flexibility. However, the case model defines the stage as a structured stage which shows in the low number of activities allowing for only a low amount of complexity. One can see this when comparing the discharge stage execution with all other stage executions, where it falls into line as more flexible than the identification stages but less than the other, less structured ones.

From the case view, Lleida's second study execution is the most flexible and thereby most complex one. It shows one process variant for every patient case, making every execution unique. Apart from that, it possesses a manual task activation quota of 55%, the highest number of activities on the case level (n=12), the highest number of paths (n=73), the highest number of bidirectional paths (n=26) and the highest average number of paths per activity (n=6.08). The study bears the longest median and average duration (n=98.7 days) but also features the largest standard deviation from that value (n=65.9), again highlighting the employed flexibility. The same amount of 35 cases were executed for both the first and the second case study in Lleida.

5.1.3. Control-Flow Process Map Analysis

Moving on to the analysis of the generated process maps, the following section will try to evaluate the hypotheses that were developed in the previous section using visual analysis. The text will refer to individual maps but a complete overview of all generated visualizations can be found in the appendix in chapter B. The section will follow the same site-based structure as the previous one.

Groningen

The stage view process maps for Groningen's first case study are included in the appendix in section B.3.1. They are quite trivial to understand visually, which is expected after the log and pattern inspection phase due to the low amount of activities per stage and the overall lowest activity amongst the case studies so far.

The case identification stage shown in figure B.14 is a simple, strictly linear process. All activities and paths occur equally often and the stage was executed and completed for all 25 cases. For the evaluation stage, the GQFI indicators showed that it was only executed once and with one activity. Therefore the visualization is rather the depiction of a single activity than an actual map. Nonetheless, it shows the name of the task and confirms the hypothesis that the executed task was just for setting the due date and did not feature any medical tests or treatments.

The workplan stage's process map is also quite simple. It consists of only two activities, with a strong self-transition on the *PhysicalActivity-Alert*. This shows that alerts are often acknowledged in bulk, possibly because clinicians check the cases they're responsible for in fixed intervals. The fact that 16 of the 20 cases have a transition from the *PhysicalActivity-Task* to the end note confirms the presumption in the earlier section that the main part of cases never received an alert, only 4 cases actually did. Also, the map does not contain any outgoing, non-looping paths from the *PhysicalActivity-Alert*, meaning that no additional workplan task was ever done as a reaction on an alert.

The second study's process maps can be found in section B.3.2. The identification stage was executed very similarly to the one from the first study. Two variants of the stage definition exist, with the case models prior to version 4 being exact copies of the stage definition during case study 1. However, only one execution exists featuring such an early case model. All other cases were executed using later versions that features an additional *SiteOfSurgery-Task*. Except for one case which swaps the order of the last two tasks, all cases follow the same path even though the could deviate from it. The execution of the evaluation stage is just like the one from the other study at Groningen and consists only of the one task for setting the evaluation date.

The workplan stage is split into three different versions, each containing more available tasks and thereby enabling higher flexibility. The first version was executed only for one case. Still, it shows 3 task events and a bidirectional path between the task and the alert activity. This indicates some kind of reaction to the alert but that is hard to tell without interviewing the clinicians. The intermediate versions of the case model featured a workplan stage that enabled 4 different tasks. The process maps show that only 3 of the 4 tasks were actually ever completed and for 2 activities, alerts occurred at a later point in time. The later versions of the workplan stage allowed for an additional, fifth task but that was never executed and both process maps seem compatible. When considering the definitions, however, the incompatibility becomes apparent. Both versions look quite similar with the slight difference of the *PhysicalActivity-Alert* being way more prominent in the earlier definition which is probably due to the cases being around for a longer time and more alerts having the possibility to occur at any point in time. The median case duration from the log and pattern inspection confirms this (n=68.8 days for earlier, n=26 days for the later stage model). The same applies for the overall complexity, the number of paths and the number of bidirectional paths which is slightly higher in the earlier workplan stage definition versions, possibly due to historical reasons and the cases running for a longer amount of time. Apart from that, no real structure can be found in the stage executions, not even when turning down Disco's path abstraction slider all the way to zero, which results in a star-like structure without any linear process fragments.

Tel-Aviv

All process maps for the two case studies performed in Tel-Aviv are depicted in the appendix, in section B.3.3 and section B.3.4. For the case identification stage of the first

study, two non-compatible versions exist. The earlier stage models, up to version 7, allowed some degree of flexibility but requiring the execution of all tasks with the only restriction being the *InclusionCriteria* task to be completed prior to the *ThechTest* or the *Patient's Consent* task. This also shows in the respective process map which clearly contains a common path that is usually followed but also shows some amount of deviation and modified task order. The map also includes a bidirectional path and 4 cases that started it did not finish it but got stuck after the *SelectPatient* task. The next iterations, from version 8 onwards, prevent part of this flexibility by introducing two new sentries and thereby creating a layered stage with more restrictions regarding the task order. This can also be seen in the process maps, which depict an extremely linear process without any exceptions or special cases that was completed for every case that started it. The visualization of these later versions of the stage results in a graph very similar to the one from Groningen's first case study.

The evaluation stage was executed for 47 cases, lacking exactly the 4 just mentioned ones to reach the overall case count known from log and pattern inspection. The model is highly complex and suffers under the spaghetti effect described in section 2.3.1 when no additional Disco abstractions are applied. But even though the full, unfiltered model is too hard to understand, one can still see a dominant path, even if no paths are abstracted away. When the abstraction of paths is maximized, a very clear picture emerges that shows a linear process execution that most cases follow. Only two backward facing paths exist, all other paths are strictly directed into the same direction. The order of the activities follows the order of the tasks in the web frontend. Some paths have lower transition numbers than others which means they mark places where some exceptions occur, i.e. clinicians can and do deviate from the commonly followed frontend order if desired.

According to the log and pattern inspection phase, the workplan execution is less complex but more flexible than the execution of the evaluation stage. The reduced complexity is due to the lower activity count with 6 instead of 18 activities that enables a maximum of only 36 instead of 324 paths. However, while the stage also has a slightly lower count of paths per activity, the overall occurrence of the paths is more evenly distributed. In other words, this means that there is no common path in the model that can be seen visually and that indicates a default path through the stage from which clinicians only deviate for a reason. Instead, the stage resembles a highly individualized workplan that is heavily adapted to fit the individual patient's needs. This can also be seen visually as all paths have roughly the same thickness and no dominant path exists. Increasing the abstraction level does not result in a linear process but rather in a circular process. This could be related to the iterative combined workplan stage of the conceptual model of the case studies depicted in figure 1.5. The combined workplan stage consists of a workplan definition and a workplan execution stage but does not define a hard border and allows transitioning back and forth between both phases, e.g. when unexpected events like alerts occur. However, without interviewing a domain expert, this remains speculation.

The identification stage of the second case study in Tel-Aviv shows the same characteristics and development as the one from the first study. Case model versions up to version

6 enable some flexibility which is removed by the use of layered sentries in version 7 and onwards. This results in a strictly linear process flow where all cases that started the stage complete it successfully. Yet, the difference to the earlier versions is not that pronounced as only one case deviated from the default task order that is displayed in the web application.

The next stage, the case evaluation, makes an impression that is very close to the one just described in terms of the first case study. It is slightly more complex and slightly less flexible. The complexity shows in the increased amount of activities and paths, resulting in a larger process map. The reduced flexibility, on the other hand, becomes apparent when Disco's path abstractions are applied. While the first study showed quite a linear process with two paths facing back to the start at the highest degree of abstraction, the second study produces an execution model that is even more linear and does not feature any backward facing paths spanning more than 2 activities.

Finally, the workplan stage of the second case study also produces a process map similar to the workplan stage of the first study. This is not surprising, as the case models for the stage are actually equal across both case studies in Tel-Aviv, at least regarding the definition of the workplan stage. This visually confirms the impression gained during the log and pattern inspection, based on the measured indicators. Increasing the path abstraction level to the maximum results in a map that – similar to the first case study – has a circular structure, highlighting the non-linear nature of the treatment stage.

Lleida

The execution of Lleida's first case study lead to four process maps for the four different stages which can be found in section B.3.5. Like at the other sites, the case identification stage is the most structured and least complex one. It possesses a main path through the whole map that 27 of 35 cases (77%) follow. This is one of the results of the log and pattern inspection but can also be clearly seen in the generated process visualization. Some exceptional cases do not fully follow the main path and swap the order of the tasks. All of them, however, choose the *GlobalDeterioriationScale-Task* as their second activity. The first activity is always the *SelectPatient-Task* which is enforced through the use of sentries. One bidirectional path gives an additional hint towards the flexibility of the stage that, in contrast to developments at the other sites, was not removed by later changes but still persists at the time of writing this work. Another possibility for adaption is provided through the case model by making one of the provided tasks an optional one that could have been skipped by the professionals. Together with Groningen's first case study, this is the only identification stage that features optional tasks. Interestingly, for both sites, no task from the identification stage was ever skipped and professionals always performed all tasks before completing the stage. For Lleida, this leads to all activities being performed equally often. All cases that started the stage successfully completed it.

The evaluation stage of Lleida's first case study is the most complex stage definition and its execution event log produces the largest and most complex process map. Like the other evaluation stages, it contains a prevailing path that many cases follow. Nevertheless, its

complexity is so high and the spaghetti effect is so expressed that it is hard to find the main path and understand its order of activities without applying abstractions. But when doing so, in contrast to the other sites, no obvious, linear process emerges, even at the full level of path abstraction. The resulting model contains structured parts, but it also features less structured ones. Only when additionally applying an activity abstraction, the process becomes linear. The reason for this is the optional tasks that the stage provides. In the structured and more linear areas, all tasks are required and have to be executed, leading to an execution sequence that orients at the web frontend like it is known from other sites. However, the optional tasks have to be activated manually and actually only performed if the professionals think it is appropriate. This leads to an activation quota of around 10% to 50%, depending on the exact task, and to the lack of any higher structure in the resulting process map. Thanks to the difference in occurrence frequency, the distinction between a required task and an optional task can also be made visually, by inspecting the color of a certain activity. If its color is very saturated, the task was executed in every and it is required, otherwise, it's optional.

According to the log and pattern inspection, the workplan stage has the highest complexity amongst all workplan stages but has lower flexibility than that of the Tel-Aviv case studies. This can clearly be seen in the process map that is the most convoluted amongst the studies but still way less complex than the evaluation stages, due to the reduced amount of activities. It does not feature a dominant main path but the *PhysicalActivity-Alert* activity is clearly the most observed one. When full path abstraction is applied, the two-divided nature of the stage is revealed, as was already observed for Tel-Aviv's workplan stages. In Lleida, the effect is even more expressed: while most tasks are grouped on the right, the left side of the map consists mainly of alerts. Two paths connect both sides, one in each direction. This shows that in the practical execution, professionals tend to perform the check and acknowledgment of alerts and the adaption of the running workplan stage as two mostly separate activities. Exceptions exist with two tasks being on the left side, but they occur very rarely.

As determined in the log and pattern inspection, Lleida is the only site that ever executed tasks from the discharge stage. The stage is modeled as a structured stage but executed in a quite flexible way. The process map shows 2 bidirectional paths out of a maximum of 3 and 5 paths in total out of a maximum of 5. This shows that the stage makes use of a large part of its possible flexibility. However, thanks to its low activity count, the stage can still be referred to as structured, as its definition enables so little complexity and only the order of the 3 tasks can be varied. A total of 13 cases shows tasks from this stage but none of the activities was executed 13 times. As all tasks are modeled as required in the stage definition, this means that no case actually ever fully completed the stage.

The last and by definition most complex study is the second case study of Lleida. The generated process maps for this stage can be found in the appendix in section B.3.6. Amongst all identification stage executions, the second study in Lleida employed the most complexity and made use of the most flexibility according to log and pattern inspection. This can also be confirmed when evaluating the process map. The stage starts quite struc-

tured, with the *SelectPatient-Task* always being the first activity, which is enforced by the use of the stage's only sentry. In all cases, following that, the *TechnologicalTest-Task* is completed. Afterward, the stage is used very flexible regarding the task order but still has a main path that can clearly be seen visually and that is followed by the main part of cases. Nevertheless, in this later part, 5 bidirectional paths exist showing that many tasks are often swapped in order. However, all activities occur equally often (n=35), so all cases that started the stage completed it and the only difference between the stage executions are differences regarding the task order.

The case evaluation stage produces two process maps due to a breaking change with a removed but previously required task that existed in the case model's version 4 and before. Both models are very complex and show a very expressed spaghetti effect. According to the log and pattern inspection phase, the stage was used very flexible but bears slightly lower flexibility than the evaluation stage observed in the first case study in Lleida. Visually with full path abstraction applied and based only on the process maps, this does not actually seem the case, though two very similar maps to the first case study are created. For the first case study, when abstractions are applied, one can identify multiple paths that have very high support, with 80% of cases (n=28 of 35) showing an event trace where one task follows the other. While other parts of the map are less structured and have worse support, there are still combinations of tasks that are usually completed after each other. This is no more the case during the second study where the highest support is 67% for the earlier versions (n=10 of 15) and 58% for later versions (n=11 of 19). Usually, the support is even lower, with the paths occurring between 2 and 6 times on average. This shows that the stage was executed very flexible and the process contains only little structure. There is one difference between the two versions: in the earlier version, most activities occur roughly equally often (n=14), so most cases that started the stage already completed it. This is different for cases that follow the more recently developed case model, where a larger part of cases (n=6) has started but not yet finished the case. Visually, this manifests itself in the lower half of the process map for later versions having a lighter color than the upper half.

The process map for the workplan before hospitalization stage seems to fit the hypothesis from the log and pattern inspection in that it is similar to the already known workplan stages but bears less flexibility. This can be seen when the abstraction is increased, as this produces a process map that is split in two like previously observed, although the semantic interpretation is not as clear as in the previous studies. However, there are two other signs for low employment of flexibility. One the one hand, the manually activated *MonitoringPrescription-Task* activity has been executed 26 times, that is only one time less than the occurrence of the *BloodPressureControl-Task* activity that is required. On the other hand, some optional tasks only occurred once. When the activity abstraction is increased until these activities are hidden, a very simple model consisting of only 5 activities is created. This shows that while the map is complex without abstractions, it reveals structure when abstractions are applied.

The workplan after hospitalization stage is once more split into two process maps due

to a breaking change from version 7 to 8 where again a required task was removed. As discovered in the log and pattern inspection, the complexity of the stage is high but the flexibility falls below that of the Tel-Aviv case studies. This is similar to the workplan of Lleida's first case study. Comparing the two versions, one can see that the earlier stage definition produces a bigger and more complex process map. They possess the same amount of activities but the versions up to number 7 show more simple and bidirectional paths. The occurrence per activity is also slightly increased compared to that of versions 8 and above. Additionally, the alert count is heavily increased by a factor of more than 4 while the increase in tasks is only up to a factor of 2. Possible reasons for this are the reduced amount of cases that passed through the stage following the later definitions (n=10 vs. n=17) but also the fact that the earlier definitions contain longer-running cases that had more time to create exceptional situations that produce an alert. When path abstraction is applied, both versions yield similar results. The *PhysicalActivity-Alert* activity is in the center of the process map with the other elements being arranged around it in a star-like structure. Like observed for the previous workplan stages, alerts are clustered together, suggesting that they are acknowledged and checked in bulk. After applying full path abstractions, an unusually high amount of paths remain and the model still has many circular substructures.

The discharge stage is similar to the one from the first case study, but, even though it is more complex and was used in a more flexible way, its execution bears more structure than the previously discharge stage. This may seem counter-intuitive at first but is easy to understand when Disco's path abstraction is used. The stage from the first study has quite equally distributed paths and abstracting away some of them still results in a kind of circular model. The second stage, on the other hand, shows the maximum number of paths and bidirectional paths but increasing the abstraction results in a more linear process due to more cases following a single path. However, as only 13 cases show tasks from the stage and two activities were only executed 7 times, only 7 cases have yet completed the discharge stage. Therefore, the observation could also relativize when the stage is executed by more cases.

5.1.4. Summary of Findings

In conclusion, one can say that the model-provided flexibility during case execution was employed in a way that is very similar to the one expected by Cano et al. in the study that served as basis for the development of the CONNECARE case studies [15]. In other words, the observed during the execution of the stages corresponds to the expected degrees of structuredness, as depicted in figure 1.5. The identification stage was executed in a very structured way while the evaluation stage showed a structured execution with ad-hoc exceptions which result in a part of the cases following the same task order displayed in the web interface. Additionally to this flexibility, the workplan stage, while bearing an overall smaller complexity, was also used very adaptively, consisting mainly of repeatable tasks that needed to be activated manually. This resulted in unique stage execution se-

quences for all patients. The final discharge stage showed a relatively flexible usage but was still used in a structured way when compared to the other stages. Structured execution was achieved either but low possible stage complexity (e.g. low number of tasks in Discharge) or by using sentries which were also shown to increase execution structuredness when introduced at later points in time. Overall, the differences between case study 1 and case study 2 at each site were noticeably smaller than the differences between the sites, even when comparing different case studies. Apart from that, an overall trend could be identified that showed higher model complexity and higher flexibility, the higher the general system activity was for a certain site, even for structured stages. The highest flexibility and adaptability was observed in Lleida, followed by Tel-Aviv and then Groningen.

5.2. Effects of Communication and Notification Functionality

5.2.1. SACM Features for Communication and Notification

Apart from being an adaptive case execution engine, the SACM system offers additional features which are not part of the case study design itself as it was described in section 1.2.7. Their availability cannot be configured through the case model but is a system property and therefore enabled for all studies. The features are task alerts that signalize unexpected conditions during automated measurements, case notes that can be used for storing additional, unstructured information and messages that can be used for communicating with the patient or other professionals. The appearance of the case notes user interface can be structured and styled using the case model to allow for flexible usage. All of the features are only expected to be used in unforeseen situations, e.g. when a patient is unsure about some instructions, when a certain test returns ambiguous results that need to be discussed with a colleague or when new information not expected by the model but relevant to the case needs to be documented. Therefore, according to the spectrum of structural degrees depicted in figure 1.2, these process activities can be characterized as unstructured.

In the following section, first, the usage of these features and their occurrence in the overall case flow will be evaluated on a system-wide level across all sites and case studies. This will be done using the colored system view process map. Afterward, the results of the GeoIP analysis will be presented to see how the overall usage distributes amongst the individual sites. Finally, the indicators from the GQFI table and the case models for the different studies will be analyzed in order to gain insight into behavioral differences between the case study executions.

5.2.2. System-Wide Evaluation

The system view is the most abstract process map that is generated in this work. It is intended for gaining an overview of the executed case studies and the overall feature usage and can be found in the appendix in section B.1 where it is depicted in figure B.1. Without

any abstractions, it bears a very high complexity due to the diverse nature of the patient cases across all sites and studies. Examples for this are e.g. the path from *Create-Case* to *Terminate-Case* where a case is created and closed again without even selecting a patient, probably due to a human error. Other examples include cases that are created and immediately receive a professional message, but all of these cases are quite rare. To reduce the spaghetti effect and create a more understandable model, the path abstraction needs to be increased to get rid of rarely occurring activity relations. A nice, comprehensive overview that neither over nor under optimizes is reached at an abstraction level of 28% remaining paths. The resulting model is shown in figure 5.1 and will be analyzed in the following paragraph.

From a distance, one can clearly see that the top-level structure resembles the structure from figure 1.5 that served as the basis for the development of the models for these studies. It starts very structured, becomes structured with ad hoc exceptions and then unstructured with predefined fragments. Afterward, exceptions occur and lead to unstructured activities before finally returning to the structured discharge procedure. The performed *CaseIdentification-Task* and *CaseEvaluation-Task* activities show strong self-loops that account for a large number of their total occurrence. This can be explained by professionals spending a longer amount of time in these stages and performing much of the required tasks subsequently. In contrast, the *Workplan-Task*, *Patient-Message* and *Professional-Message* activities also show self-loops but less expressed ones. This is due to a higher frequency of other activities being performed in between the mentioned ones which is natural for message-based conversations but also for the workplans which are usually created once and then only adapted at a later point if special events made it necessary. The self-loop on the *Workplan-Alert* activity is equally strong as those of the identification and evaluation activities. However, it is more likely caused by the clinicians acknowledging the alerts in bulk because they e.g. check the cases in fixed intervals. This was also found during the evaluation of the first research question and is additionally backed by the intra-stage analysis which shows similarly string self-loops for alerts.

When considering the position of the activities related to additional SACM features, it becomes apparent that the case notes are commonly updated after a case evaluation task has been performed and that afterward oftentimes a message to the colleagues is sent. Temporarily turning the abstraction level back up to unfiltered reveals that notes are also often updated after a professional message was received. This makes sense as professionals need to somehow document the results of e.g. interdisciplinary discussions following ambiguous test results which are usually captured during the evaluation stage. Apart from that, when the notes page is updated, no notification is displayed in the web interface. The messages following the *UpdateNotes* activity are therefore quite likely sent in order to inform all colleagues about new case developments that have also be documented in the case notes. The path from the *Professional-Message* to the *Update-Notes* activity was hidden in the original abstraction level because it occurred only one time less than the path from *CaseEvaluation-Task* to *Update-Notes* which highlights the importance of an easy-to-use interactive visualization tool.

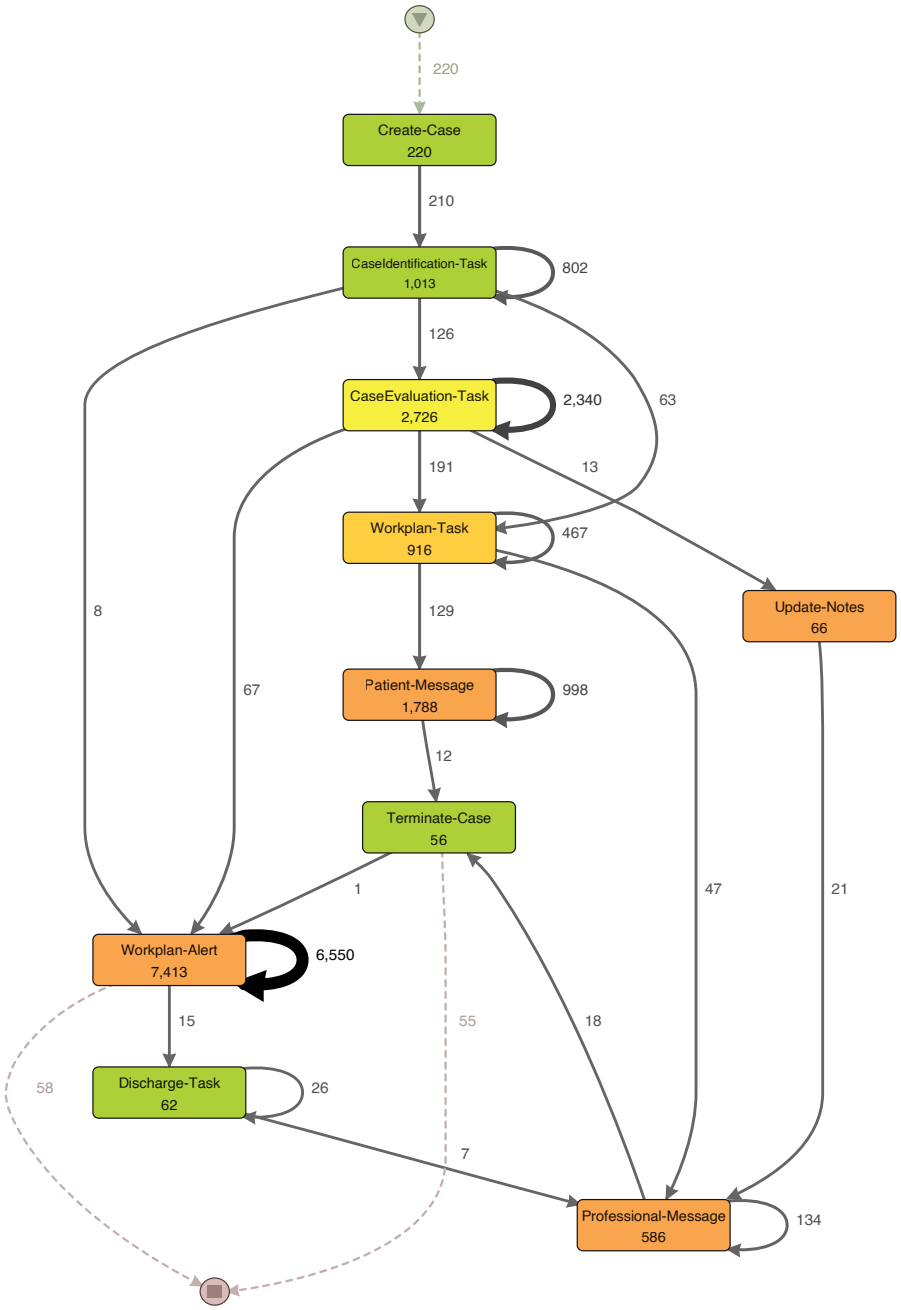
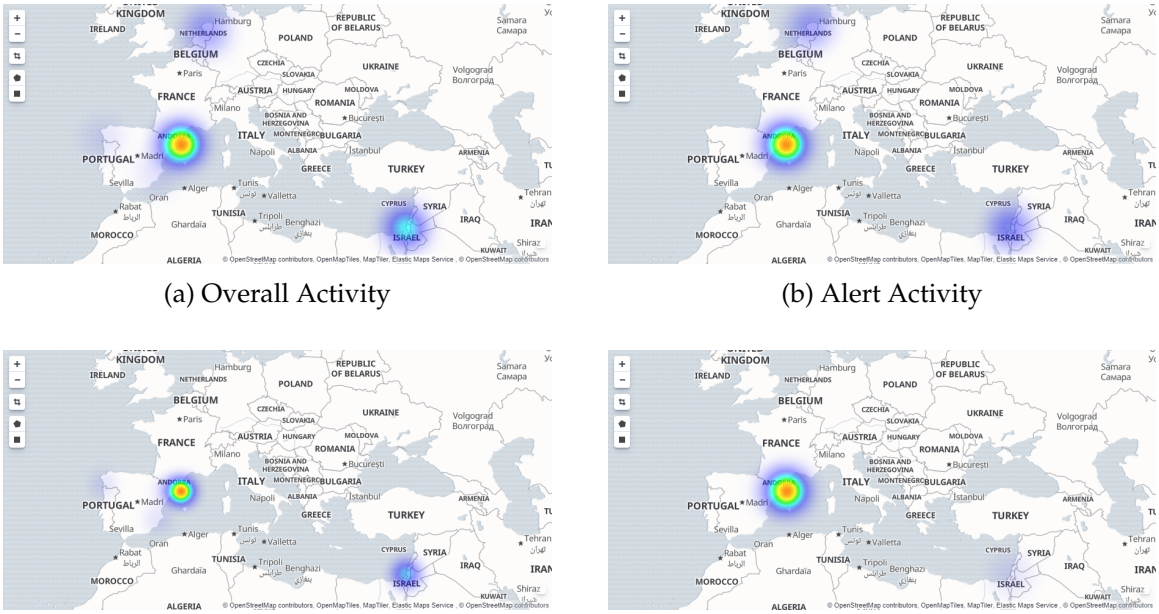


Figure 5.1.: System view at 28% path abstraction level

Most patient messages are received during the workplan stage, after the initial creation of the individualized workplan. This stands to reason as patients perform their workplan from home and are encouraged to use the messaging functionality to communicate with the clinicians in case any uncertainties regarding the execution or the overall state of health arise. The same order applies for professional messages which could be explained by insecurities regarding certain workplan tasks but also by the necessity to inform collaborating colleagues about workplan modifications. The *Workplan-Alert* activity is most often observed to follow a *CaseEvaluation-Task*. This seems counter-intuitive as the reader might expect the alerts to occur after the *Workplan-Task* activities that defined their threshold. However this is not the case, as alerts tend to not occur immediately after they were defined, otherwise, this would hint towards an issue regarding the developed workplan and its alerting thresholds or the patient's recovery motivation. Instead, alerts usually occur sometime after they were initially defined. As the case evaluation stage can be repeated and contains tests that can be done to assess the patient's health, it is not implausible that activities from this stage precede workplan alerts.

To see how the feature usages distribute onto the individual sites even before generating the process maps, the site comparison dashboard in Kibana can be used to gain an early impression. Its main information, the maps visualizing a GeoIP analysis, is shown in figure 5.2 and the full dashboard can be seen in the appendix, depicted by figure D.3.



(a) Overall Activity

(b) Alert Activity

(c) Message Activity

(d) Notes Activity

Figure 5.2.: Heatmaps showing activity for various features by GeoIP location
(Maps © OpenStreetMap contributors, released under CC BY-SA 2.0, see https://www.openstreetmap.org/copyright)

The GeoIP analysis uses the source IP of the underlying request and looks it up in a

GeoIP database that maps IP addresses to geolocations. While not being entirely accurate down to the meter level, it is still useful to find out the country where a request originated from and thereby determine the site that performed a specific activity. By filtering this data for the desired activity types, an overview can be gained of which feature is used how heavily in each site. The data is visualized using heatmaps which, when viewed interactively, can also be used to determine the activity counts. For this work, however, the heatmaps will only be used to visually determine a rank based on the indicated colors as the numbers have already been captured for the GQFI table.

The GeoIP analysis shows that alerts occur at all sites, with the most activity at Lleida in Spain, followed by both Tel-Aviv in Israel and Groningen in the Netherlands, which visually can't really be separated and seem equally active. The notes feature is primarily used in Lleida and has some very rare usages in Tel-Aviv which occur so little that their indication is very hard to spot without using the interactive viewer. The professional message feature is also used the most in Lleida and is again followed by Tel-Aviv, this time with a clearly visible indicator. The patient messages are handled by the SMS which does not feature request logging. Therefore, this location-based analysis cannot be done for this feature and the values from the GQFI table have to be used to gain an impression about its usage.

5.2.3. Site-Based Evaluation

The evaluated GQFI table for the second research question is shown in table 5.1 and will be explained in depth in terms of this section. One interesting finding that can be made without interpreting the numbers in detail is that the sites show roughly similar indicator values for both studies and the main differences are between the sites. The only exception to this rule is the difference in the alert count between the two case studies at Groningen which also affect some other indicators like the relative share of feature-related events. Therefore, the table will be combined with the generated process maps in the case view that show the predominant occurrence of each activity in the case flow. By doing so, each site can be compared to the other sites but also to the system view map which enables a thorough answer to the second research question. For the evaluation, all case models have been configured to the highest degree of path abstraction, so only the most common cases are considered.

Groningen

Amongst the three execution sites, Groningen used the advanced features of the SACM the least. Both studies show no use of the notes and the professional functionalities whatsoever. This is not surprising as Groningen until now featured only very little collaboration amongst users, which will be explained in depth in the upcoming section 5.3 that analyzes the employment of the user and role system. The patient messaging functionality is primarily used in the first case study but the message counts still lack behind studies

	Groningen		Tel-Aviv		Lleida	
Indicator	**CS1**	**CS2**	**CS1**	**CS2**	**CS1**	**CS2**
Number of notes update events	0	0	2	2	31	29
Number of alert events	28	1 192	1 181	649	2 353	1 764
Number of message events (with patients)	57	6	342	201	641	523
Number of message events (with professionals)	0	0	106	97	230	106
Share of feature-related events	37%	79%	55%	48%	76%	69%
Mean number of paths per alert activity	3	2	6	6	7	7
Mean number of paths per message activity	2	3	6.5	6	7.5	8
Mean number of paths per other activity	2	2.25	3.2	3	4.4	5.14

Table 5.1.: Evaluated GQFI table for the second research question

at other locations by a factor of close to 4. As mentioned, the first case study has a very small alert count with the next bigger count being more than 20 times as large. The second study shows an average use of the alert functionality but, instead, even completely lacks any other execution traces of additionally provided features.

When considering the case view process maps that are depicted in figure B.4 and figure B.5 of the appendix, both case studies show the *Patient-Message* activity at the same position that it has in the system view map, namely after the *Workplan-Task* activity. The *Workplan-Alert* activity is located directly following the *Workplan-Task* or the *Patient-Message* and is thereby at a different location than in the system view map. When keeping in mind, however, that both case studies featured only one activity execution of the case evaluation stage, the chance for this activity being the predominant predecessor to the alert activity is quite minimal. The other two features were not used and therefore no related activities are present in the process maps.

Tel-Aviv

Similar to the findings during the evaluation of the first research question, Tel-Aviv has a higher usage of additional, SACM-provided features than Groningen but falls behind the study executions in Lleida. It employs all available features except for the notes functionality which is only used 4 times in a total of 70 cases and is therefore quite underused. A very similar amount of around 100 professional messages is exchanged in both case studies. This is a little strange, as the second case study involved 9 professionals while the first one only involved 3, with one of them performing hardly any work (see the upcoming section 5.3 for further information). Therefore, 2 persons would have to exchange roughly as many messages as 9 persons which is not very likely. An interview with a local professional was conducted for this reason which showed that the notes functionality, not implemented in the initial SACM design but added later on, was not known to the clini-

cians. To compensate for the supposed lack of the feature, they came up with the idea of repurposing the professional messaging feature and used it for the internal documentation of the treatments each patient had received. This explains the equal professional message counts as well as the close-to-zero amount of case notes. Communication amongst the team was done via a preexisting third-party smartphone messaging application that was already in use while the case studies were still in the design phase.

The other features were not affected by this change and were used as they were intended. Case study 1 has an alert count that is similar to that of Groningen's second study (n=1 181). Close to 350 patient messages were exchanged, resulting in an average count of around 10 messages per patient. Roughly speaking, both values are only half as large for case study 2. Referring to the GQFI indicators for the first research question shown in table C.2, this correlates with the overall case and activity count which means both sites actually used the features in a similar way. This fact also shows in the relative share of the feature-related events compared to the overall events which is roughly equal for both studies at around 50%. In Tel-Aviv, being a site that actually makes use of the additional features, the dynamic nature of alerts, notes, and messages can also be seen. More specifically, it shows in the average number of paths an alert or a message activity possesses in comparison with the average number of paths that can be found on non-feature-related activities. Both studies have similar values of around 6 paths per feature-related activity versus only around 3 paths per non-feature-related one. This highlights the dynamic nature of the unstructured activities that can occur at basically any point during a case and are therefore better connected with all other activities.

The case view process map for case study 1 in Tel-Aviv shows a picture very similar to the system view. The activities for *Update-Notes*, *Professional-Message* and *Patient-Message* are at the same position they hold in the system view, following the *CaseEvaluation-Task* and the *Workplan-Task*, respectively. The only exception is the *Workplan-Alert* activity that follows the *Professional-Message* instead of the *CaseEvaluation-Task* activity. When temporarily turning down the abstraction level, the missing path from the evaluation stage to the alert becomes visible. However, it only occurs 7 times and is therefore not considered dominant when compared to the alternative path that occurred 12 times. Given the overall low path occurrence count of around 10 (out of roughly 3 000 activities overall), this can easily happen which again proves the importance of an interactive viewing mechanism for process maps to dynamically change the abstraction level during the evaluation.

For case study 2 the just observed issue exists as well, for the *Professional-Message* and the *Update-Notes* activities. Both are displayed at a different place than in the system view but reveal additional paths connecting them to the expected activities when the abstraction is reduced. The *Workplan-Alert* activity that had this issue in the previously described case study is now at the expected position. Seeming a bit strange, at full abstraction, the *Patient-Message* activity is barely connected to other activities at all and does not have any incoming paths. When reducing the abstraction factor the issue persists and is only mitigated when the abstraction is completely disabled. The reason for this strange behavior is the strikingly high dynamic nature that this activity bears, as patient messages are exter-

nal events that can occur at any point in time and, in contrast to the professional messages, the patient messaging functionality is actually used as it was intended. In this case, their occurrence is so well distributed, that most incoming or outgoing paths have a similar occurrence frequency which is why they are all abstracted away as soon as the degree of abstraction is raised.

Lleida

Both case studies in Lleida generated the most system activity with regards to the overall event count as shown in the evaluation of the previous research question (see table C.3). But this is also true in terms of the number of updates to the case notes, the alert count, as well as the number of patient and professional messages. This shows that the site makes full use of all features provided by the CONNECARE project and its SACM component. Therefore, the relative share of the SACM-feature-related events is also the highest with the only exception being case study 2 in Groningen that was presumed to have under-used the more structured activities due to the over-proportionally high number of alert events according to the preceding section 5.1. With a value ranging between 7.5 and 8, the average number of paths per message activity is extremely high for both case studies, supposedly due to the involved persons using the features in their intended way for communication which is an inherently unpredictable procedure, especially in KiPs. Therefore it is highly questionable if the generated process maps will contain a significantly dominant path connecting them with other activities or if they really occur at any point in time, yielding a result similar to the *Patient-Message* activity of the just evaluated case study 2 in Tel-Aviv. On the other hand, the average number of paths for non-feature-related activities is also the highest amongst all case studies, so the overall more dynamical system usage could also be an explanation for the extreme values.

Surprisingly, for the case view process map of the first study, a strong bidirectional relation between both message-related activities and the *Workplan-Alert* activity can be found. For patient messages, the quota of message activities following an alert is more than a third while for professional messages more than half directly follow a workplan alert. For both message activities, the occurrence frequency of the reversed paths closely matches that of the path coming from the alert. This suggests that clinicians working on a case contact their colleagues and/or the patient if an unexpected condition occurs. In fact, this suggestion was proven to be true using Disco's interactive log and pattern inspection features that allow isolating cases based on certain paths. Afterward, their event traces were viewed and it was confirmed that the bidirectional relations are supported by individual cases and not only occur when the process is viewed across all of them. Therefore, the bidirectional relation is not only a sign of flexibility but an actual rework loop. Broadly speaking, the traces show that one or multiple alerts occur, the acknowledging clinician contacts a colleague, the patient or both using the provided communication features, some messages are exchanged, some days pass, the alert occurs again and the procedure is repeated. Sometimes, the workplan is adapted in between the repetitions. This reflects ex-

actly the model developed by Cano et al. that serves as a basis for the case studies and is shown in figure 1.5 in the appendix [15]. The remaining features show the already known behavior: notes are mostly updated after tasks from the case evaluation stage have been performed and the workplan alerts often but not only occur at the same place. The latter finding is initially hidden and must be made visible by temporarily reducing the degree of abstraction.

For the patient messages of the second case study in Lleida, the same observation as for case study 1 can be made. However, the case model splits up the workplan stage into one that is performed before and one the is performed after the hospitalization. Rework loops between patient messages and alerts only exist for the *WorkplanAfterHospitalization-Task* activity. Nevertheless, this finding has limited validity as the *WorkplanBeforeHospitalization* activity occurs very rarely (n=37 versus n=1 727). When considering professional messages, this observation is not true at all and the activity is placed somewhere in the process, without significantly dominant relations to any other activity. As in all previous studies, the *Update-Notes* activity occurs mostly after tasks from the case evaluation have been performed.

5.2.4. Summary of Findings

Regarding the question how SACM's communication and notification features affected the case executions, some noticeable impacts were identified. Again, the observed usage structure closely resembles the one expected by Cano et al., shown in figure 1.5 [15]. All usages of the features under evaluation can be seen as unstructured events that occur at later points during the case executions, usually during the workplan or case evaluation stages. Rework loops were observed that consist of receiving an alert, messaging a colleague or the patient and optionally adapting the workplan to execute. Apart from that, it was shown that alerts are acknowledged in bulk, suggesting that clinicians check their cases in fixed intervals. Case notes were mostly updated before or after messages to colleagues were sent or received. Patient messages were often received during the execution of the workplan stage, which is intended to be performed in self-management at the patient's home. When comparing the sites, similar observations to those of the first research questions can be made with differences between the individual sites were larger than those between the two case studies. Again, Lleida showed the highest usage numbers for the communication and notification features, followed by Tel-Aviv and finally Groningen. These overall activity statistics can also be seen in the GeoIP analysis heatmaps depicted in figure 5.2. The notes were primarily used in Lleida and the professionals at Tel-Aviv did not even know about the feature as was discovered during an interview with a local domain expert. To mitigate the lack of the feature, they employed the unused professional messaging feature as a tool for documenting performed treatments and to ensure all clinicians working on a case were notified about the update.

5.3. Usage of User and Role Management System

5.3.1. User and Role Management Concept

As already explained in section 1.3.4, the SACM features an extensive user and role management system. Each case definition contains roles for the case instance that are restricted to certain user groups. Each user can be a member of one or more groups and groups themselves can also be group members. This enables the modeling of hierarchical structures. The system is also depicted in figure 1.6. But roles only specify who is responsible for a task, not who can and should actually execute it. Any user with write access to a certain case can update, modify and complete arbitrary tasks.

The upcoming section will evaluate if the role model created by the individual sites was used as it was intended or if the execution reality reveals a different collaboration pattern, based on the generated organizational process maps and the indicators from the GQFI table. The organizational process maps do not visualize, which tasks were performed but rather who performed a task. This person is called *task processor* in the next sections. Additionally, a process map will be created for visualizing who was responsible for a task that was performed according to the case model. This person is called *task assignee*. For each case study, this work will compare both versions to find similarities and differences, after getting a first impression using the numbers gathered during log and pattern inspection that are shown in table 5.2.

	Groningen		Tel-Aviv		Lleida	
Indicator	**CS1**	**CS2**	**CS1**	**CS2**	**CS1**	**CS2**
Number of tasks not done by their assignee	0	110	1 003	551	634	715
Share of tasks not done by their assignee	0%	37%	80%	57%	63%	68%
Number of task assignees	1	2	5	8	10	29
Number of task processors	1	4	3	9	9	6
Number of roles	1	2	2	3	5	4
Maximum share of work for one person	100%	83%	53%	37%	95%	98%
Mean number of collaborators per person	0	1.5	0.67	2.67	1.78	1.67
Maximum number of collaborators per person	1	3	1	6	8	5

Table 5.2.: Evaluated GQFI table for the third research question

5.3.2. Organizational Process Model Analysis

Groningen

The process maps for Groningen's organizational view can be found in the appendix in section B.4.1. The first case study is quite trivial to evaluate as it did not really make

use of the user and role system: all activities were performed by a single user having a single role. This seems to have happened on purpose, as the share of tasks not done by their assignee is 0% which means that the observed behavior matches the modeled one perfectly. The process maps that visualize these observations are exactly equal and trivial because they only contain a single process activity. Interviews with the responsible case study supervisor showed that a common workstation exists that professionals make use of to access the system through a single user account, no matter what actual person is operating the device. Of course, this inhibits the traceability features of the SACM but from a technical perspective, this usage is not an issue.

With only 37% of tasks not done by their intended assignee, Groningen's second case study features the best-fitting user and role model with more than a single modeled user across all studies. The fact is relativized because the model features a very central person who performed 83% of the work. Apart from that, the average number of collaborators per person is mediocre when compared to the other sites and only two responsible professionals are intended by the case model. The process map for the task assignees shows, that the two involved users should work by themselves and not collaborate at all. In practice, the distribution of work is quite similar to how it was modeled but some tasks initially assigned to one of both professionals are performed by two student research assistants. An interview with a local domain expert was performed to gain more insights about the social structure and confirmed that the students were hired only recently to help with the execution of the case study, simplify the admission process and improve overall patient care. However, the collaboration between the involved persons is not that big and working in autonomy is still the way to go. This can be seen in the low occurrence count for paths connecting the different persons. In addition, all collaboration is centered around the main case manager.

Tel-Aviv

In general, the studies at Tel-Aviv make more intensive use of the user and role system, featuring a higher count of involved users and roles. The generated process maps for both studies are included in section B.4.2 of the appendix. According to the log and pattern inspection, the first case study has the highest amount of tasks not done by their assignee, both in absolute (n=1003) and in relative numbers (n=80%). Only relying on these indicators, that means the study features the worst fitting user and role model amongst all case studies. Instead of the 5 intended users, only 3 users actually ever performed work using the system. There is no real collaboration amongst users, which can be seen at the low number of collaborators per person (n=0.67). The work is distributed quite good, however, as the maximum share of work per person of 53% is not actually that much for three users. Considering the process map showing the actual task processors, it becomes apparent that the distribution of work is evenly split between two case managers. The third user and second role, the operative research supervisor, performs only an insignificant amount of work. The task assignee process map shows a similar picture, with the three users that are

no case managers being responsible for only around 3% of the overall tasks. This means that for reaching a quota of 80% of tasks not performed by their assignee, the two case managers both need to heavily perform tasks that are assigned to their colleague. At the same time, the collaboration is not very expressed, ascertainable at the low occurrence count of one bidirectional path that exists. This means that the users often autonomously perform large chunks of work in a case that is assigned to a different user. The exact reason for this is unknown but could be related to issues regarding a missing password reset functionality that existed at the site when the case study was started and that led to users sharing their accounts due to forgotten passwords. However, the modeled user and role system, apart from not being very complex, fits the actual use quite well, except for the swapped responsibilities which do not show up in the log and pattern inspection indicators.

In contrast, the second case study in Tel-Aviv bears the highest amount of collaboration with 2.67 collaborators (bidirectional paths) per user and a well-distributed workload that leads to a maximum share of work per person of only 37%, the lowest value amongst all studies. It is based on a moderately complex user and role model that ranks itself between the already described models and the upcoming ones from Lleida. The number of task processors is one bigger than the number of task assignees and 3 roles were involved in executing the study. When taking a look at the task assignee process maps, one can see that it features the new role *Physiotherapist* which is played by 5 different users, each with quite a small share in the total workload. Apart from that, the map only contains one other role, the case manager, which is assigned to the two users known from the previous study and an additional one. The remaining workload is distributed amongst these users, with the new case manager being assigned the least amount of work. The task processor map draws a quite similar picture that has slight differences regarding the distribution of work. All physiotherapist performed more work than they were responsible for, but due to their overall low workload, this difference carries little weight. A more important change can be observed regarding the other users: both case managers known from case study 1 perform less work than what was modeled, while the new case manager performs more work than intended by the model. Considering the high workload for the case managers regarding the first study, this seems like an evident reality. The third role that is involved in the case study is the operative research supervisor who is also already known from the previously described study but again only performed an insignificant amount of work.

Lleida

The most complex user and role models are featured in Lleida. Their visualizations are shown in section B.4.3 of the appendix. The first case study in Lleida is only slightly more complex than the second one in Tel-Aviv as it features 2 additional task assignees (n=10). The number of roles also increases from 3 to 5, yielding the highest value amongst all studies and thereby making it the model with the highest degree of interdisciplinary collaboration. When consulting the process map showing the model, one can see two main users, both clinicians for internal medicine. At the top of the map is a case manager who

is only responsible for the third most amount of tasks but who works the most collaboratively, which can be seen at the high number of incoming and outgoing paths. According to the model, the user acts as a kind of distributor of work, continuously performing some work on the cases but delegating the main work to more specialized clinicians. However, the numbers from the GQFI table also show that this distributed style of working is not actually reflected in the execution logs, due to the incredibly high share of work for a single person of 95%. This becomes very apparent when looking at the process map for the task assignees. It features 9 persons of which 5 performed less than 8 tasks and 3 other persons who performed less than 80 tasks. The remaining 3 204 tasks have all been completed by the central case manager. Apart from that, no collaboration is performed directly between the specialized professionals, rather work is always handed over via the case manager. Still, the collaboration during the first study in Lleida is highly interdisciplinary with clinicians for internal medicine and nurses but also physicians and nurses from primary care being involved in the case executions. Therefore, while the modeled responsibilities seem a bit over the top and the observed usage does not make much use of the collaboration features, the collaboration that happens is highly interdisciplinary.

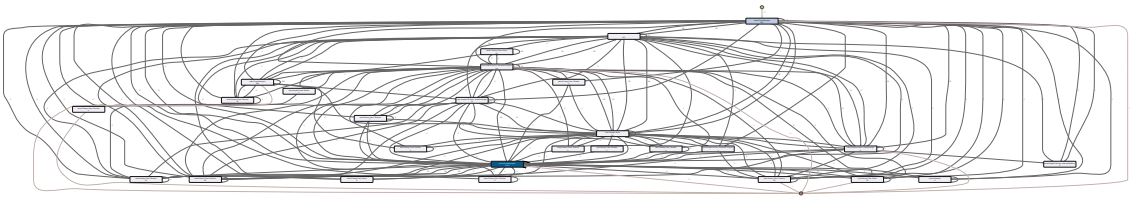


Figure 5.3.: Task assignees of Lleida's case study 2

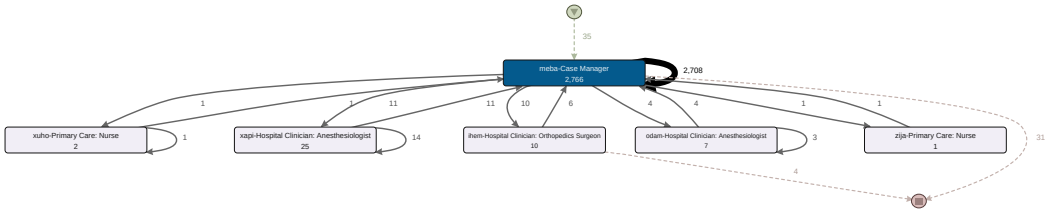


Figure 5.4.: Task processors of Lleida's case study 2

The second case study in Lleida has very similar characteristics but is even more extreme regarding its modeling and its execution. The underlying user and role system involves 29 different users while the number of observed task processors is only 6, lacking a total of 23 users. At the same time, the observed centralization on a single case manager is also more expressed, with a share of 98% of work for one person. As expected with 29 different activities, the process map that is generated for the modeled task assignees is very

large, complex and has a heavily expressed spaghetti effect. Therefore, this is the only organizational process map that requires path abstractions to be applied in order to make it understandable. But even at full path abstraction, the model is still quite complicated. A structure that is visible on first sight is the central role of the case manager that is known from case study 1 and that is also reflected in the observed system usage. Another clear finding is the prominence of one of the physiotherapists who is modeled to be responsible for the largest part of all tasks, close to 5 times as many as the central case manager. However, this is only true for the theoretical model but not for the practical observation, where not a single user with the role of a physiotherapist does even show up. An interview that was held with some of the professionals at Lleida confirmed that the individualized workplan is always developed by specialized physiotherapists. The only explanation for this observation is that the physiotherapists develop the workplan using a third-party system or pen and paper. Afterward, the developed workplan is passed on to the case manager who enters it into the SACM system.

5.3.3. Summary of Findings

The user and role management system showed the largest differences between intended (i.e. modeled) and actually observed behavior. The main finding from the analysis of the practical case executions is that every site and every case study has one or two dedicated case managers. These are heavily dominant regarding their share of performed work. They are responsible for managing the cases, distribute certain tasks to professionals with higher specialization and generally act like a hub for the handover of work on a case. Users also working on the case usually cooperate exclusively with the case manager and only very rarely directly with their colleagues. Like it was with previous findings regarding activity, the number of professionals working on the studies is the highest for Lleida, again followed by Tel-Aviv and then Groningen. For all studies except the first one at Groningen (which only modeled a single user), the modeled user and role assignments were more complex than the observed case execution reality. The differences between the theoretical model and the practical observation even tend to increase, the more system activity a certain site produced. Apart from that, the share of work of the case managers is exceptionally high. For Lleida, an interview with local domain experts confirmed that patients' workplans are always developed by a physiotherapist. However, during the practical execution, no physiotherapists were involved in any case at all, showing that an alternative, additional communication channel needs to exist (like e.g. pen and paper). The physiotherapists use this channel for sharing the workplan information with most likely the case manager, who then performs the corresponding tasks in the SACM system. This shows that so far during the case studies, the task processor from the system's point of view is not necessarily the person who really performed the actual work.

6. Final Remarks

6.1. Conclusion

The main question of this work is if the flexibility that is provided by an ACM system and, more specifically, by the SACM is actually employed in the clinical routine. The results of the evaluation show that this is clearly the case, with the process maps that were generated based on the observational data closely resembling the high-level study design that served as a basis for the developed case models. Apart from that, the process maps, as well as the indicators from the GQFI table, showed that the observed flexibility actually increased with increasing system activity. The flexibility was also higher for case studies that made full use of the additionally provided features for collaboration and communication and for the sites that had a large number of professionals directly using the system. The latter was often not the case as the sites tended to designate individual users to be responsible for the system interaction which was also confirmed using expert interviews. This suggests that a fully productive deployment without the background of a research project could yield even higher flexibility measurements.

These findings were achieved by combining PM techniques known from process discovery, process exploration, process comparison, performance analysis, and organizational analysis. The event log that served as an input to the PM step was generated based on web access logs of a RESTful API which were continuously extracted, preprocessed, aggregated and persisted using the elastic stack. The application was deployed to a productive environment by integrating it into a preexisting Docker-based infrastructure. To deal with the high complexity of KiPs like the targeted CPs, manual clustering techniques were employed and the resulting data was exported at multiple levels of abstraction. Furthermore, interactive visualization techniques were used to improve the analysis quality and enable further insights. The work was implemented in a way that facilitates the repetition of the evaluation.

6.2. Limitations

The main limitation of this work is that the system usage is evaluated while the case studies are still being performed. This leads to issues regarding the amount of participating patients which varies from site to site. Therefore, it is hard to draw definite conclusions regarding the usage of the provided execution flexibility as low patient numbers may distort the analysis results for certain sites. The same fact also causes issues regarding the fuzzy

mining algorithm which was designed to work with much higher event counts. While this work only analyzes 25 000 events, most PM tools like the employed Disco are capable of handling millions of events. Additionally, the fuzzy mining algorithm was developed specifically for the purpose of coping with large amounts of event data of varying quality and can be expected to yield more meaningful results when bigger event logs are analyzed and the abstraction features are used in their intended way.

Other limitations of the work are of more technical nature. The fact that the studies are still running was accounted for by developing a continuous event log creation mechanism that also enables log and pattern inspection at arbitrary points in time using a web-based interface. However, the event log itself is created as a simple CSV-file that is simply placed on the file system of the machine running the elastic stack. This is not ideal as it requires users that want to recreate the process maps to have access to the deployment infrastructure which is usually not the case. Enabling it requires additional configuration effort to ensure a smooth and reliable operation of all other applications that are executed in the environment.

Apart from that, the manual effort for performing the evaluation is still quite high. This expresses in the requirement to manually import the event log into Disco while also having to configure the column mappings for the CSV-files that change depending on the desired abstraction level. Furthermore, the GQFI table that is used for quickly judging the main execution characteristics regarding the research questions contains indicators from multiple tools that manually need to be aggregated. These circumstances impede the easy repeatability of the system evaluation.

6.3. Future Work

Possibilities for future work comprise the facilitation of the repetition of the analysis process on the one hand and the scientific deepening of the analysis itself on the other. The first can be reached by improving the accessibility of the current approach, e.g. by reconfiguring the already built-in nginx web server to provide the generated event log files using static URLs. Additionally, the CSV-based files could be exported as XES files which would render the import configuration-free at the cost of additional implementation effort, as it would require a new Logstash output plugin. Nevertheless, the plugin could be contributed to the Logstash community, making the approach implemented in this work available to a wider public.

An alternative would be to automatically generate the process maps at fixed intervals using ProM's command line interface and making the results accessible via the web browser. However, this would imply that the generated models are static pictures, effectively losing the benefits of interactive visualization which turned out to be very useful during the final evaluation phase.

Other possible improvements include the improved analysis of the system usage. For preprocessing the web access logs, instead of using a custom implementation, a more

generic algorithm could be used. An example of this is the algorithm for the grammar-based task analysis of web logs which was developed in 2004 and is capable of detecting business tasks performed by users based on a series of web requests. Unfortunately, it is also patented and currently assigned to Twitter Inc. [70].

Apart from that, research efforts could focus more on the deviations from the commonly followed CPs or on the changes that happened from version to version like it was done in [52]. Extending the capturing of data to include proper user session tracking could also be used to gain further insights into how the clinicians make use of the system. One could even think of implementing mouse tracking and generate frontend heatmaps showing where users put their attention and generating advanced knowledge of how professionals operate the system and its extended features.

Appendix

A. Case Models

A.1. Groningen

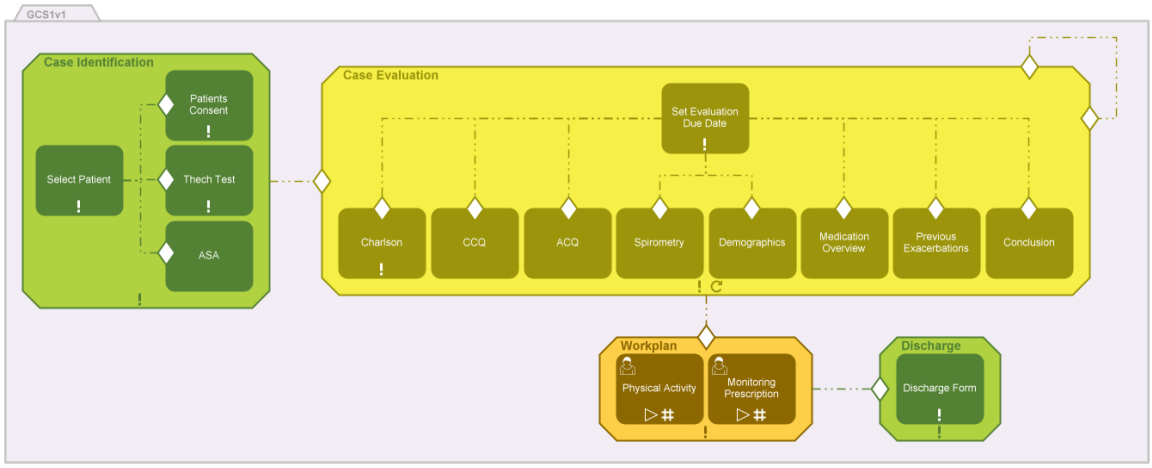


Figure A.1.: Case Study 1 Version 1

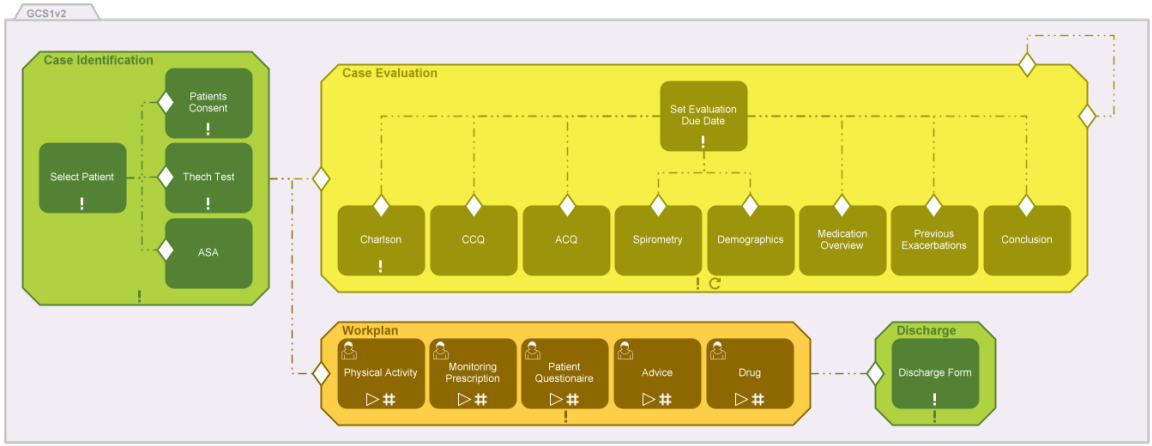


Figure A.2.: Case Study 1 Version 2

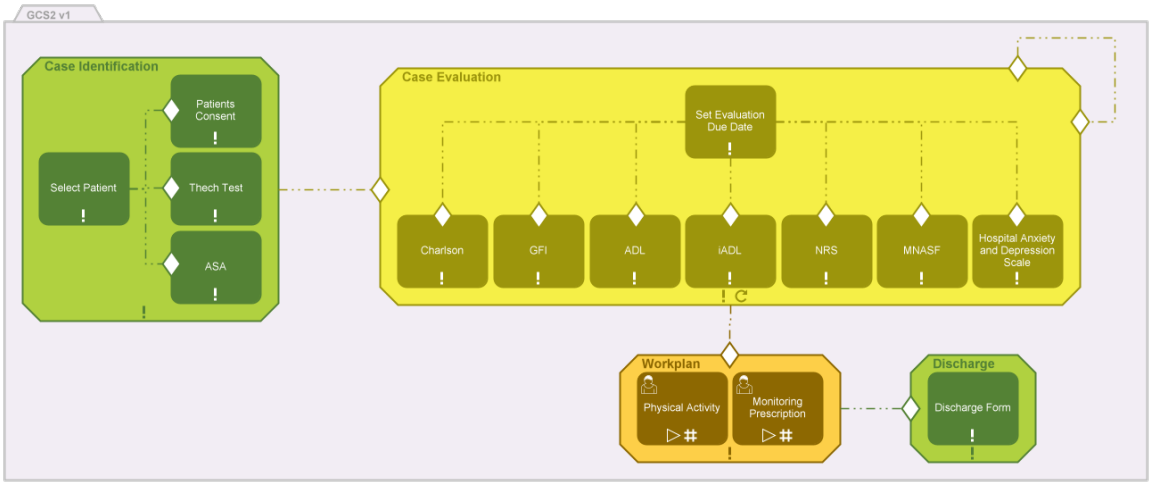


Figure A.3.: Case Study 2 Version 1-2

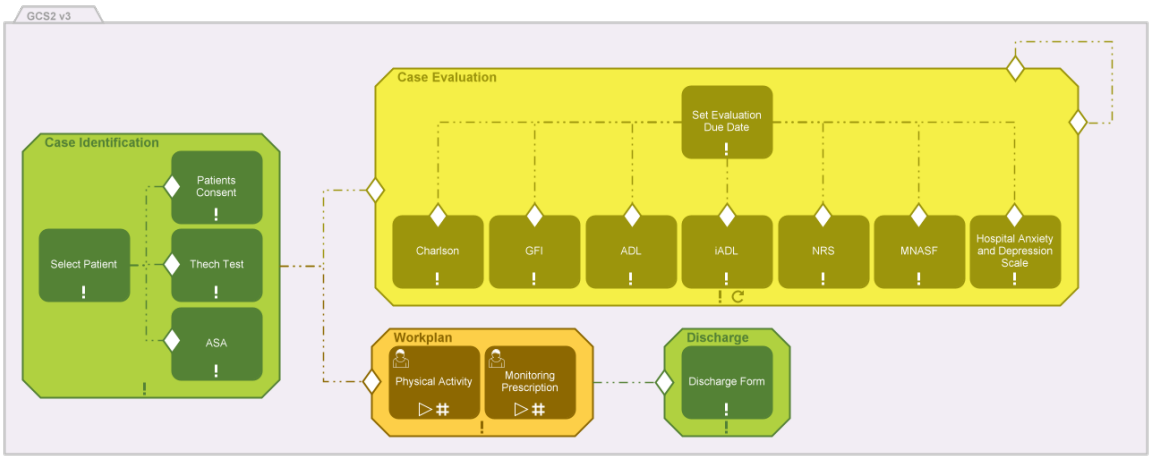


Figure A.4.: Case Study 2 Version 3

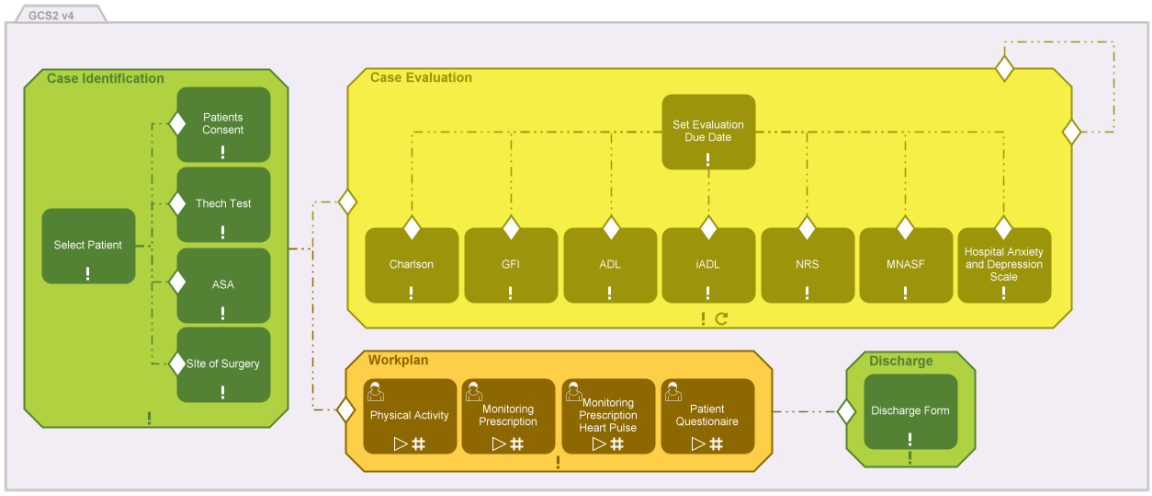


Figure A.5.: Case Study 2 Version 4-7

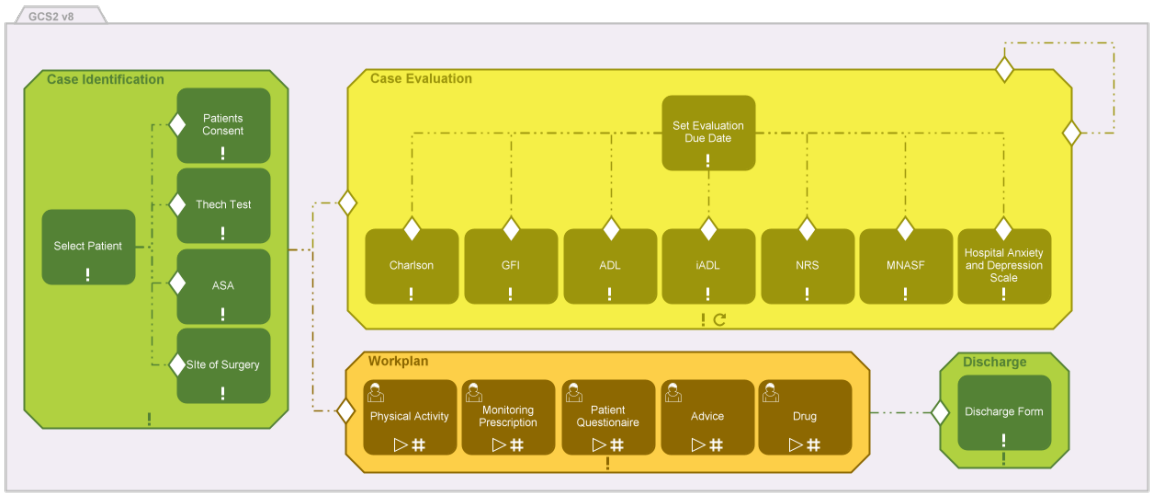


Figure A.6.: Case Study 2 Version 8

A.2. Tel-Aviv

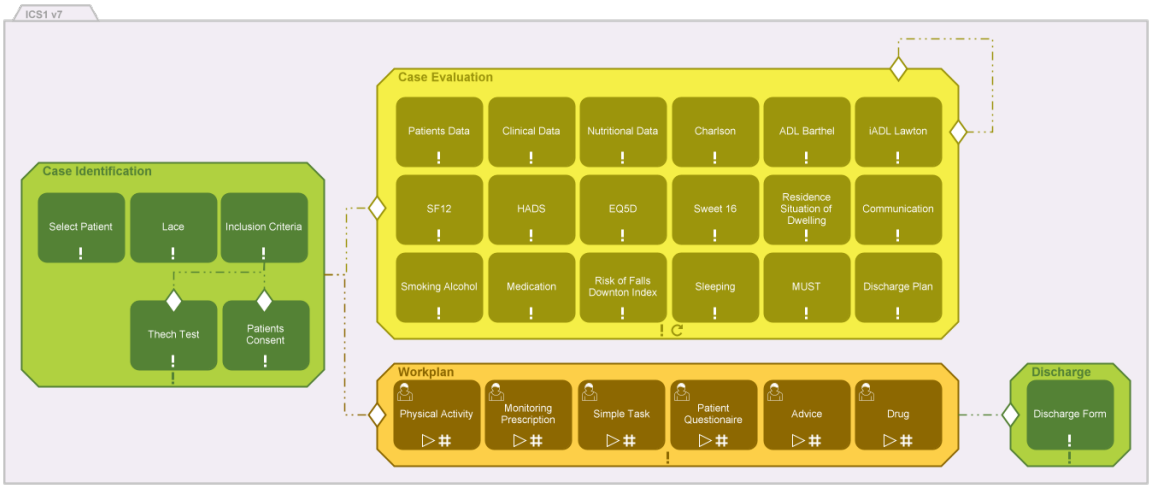


Figure A.7.: Case Study 1 Version 1-7

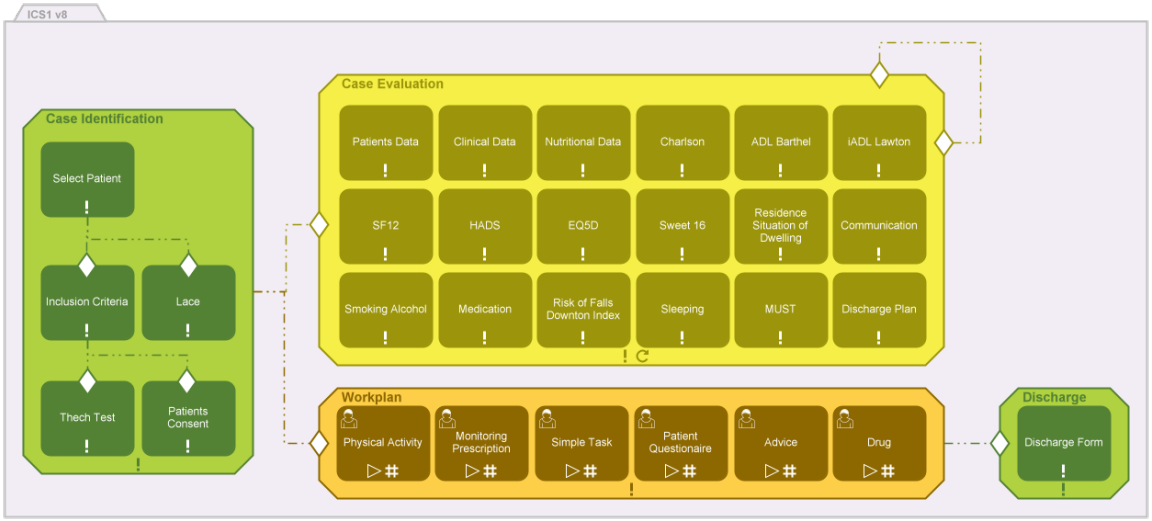


Figure A.8.: Case Study 1 Version 8+

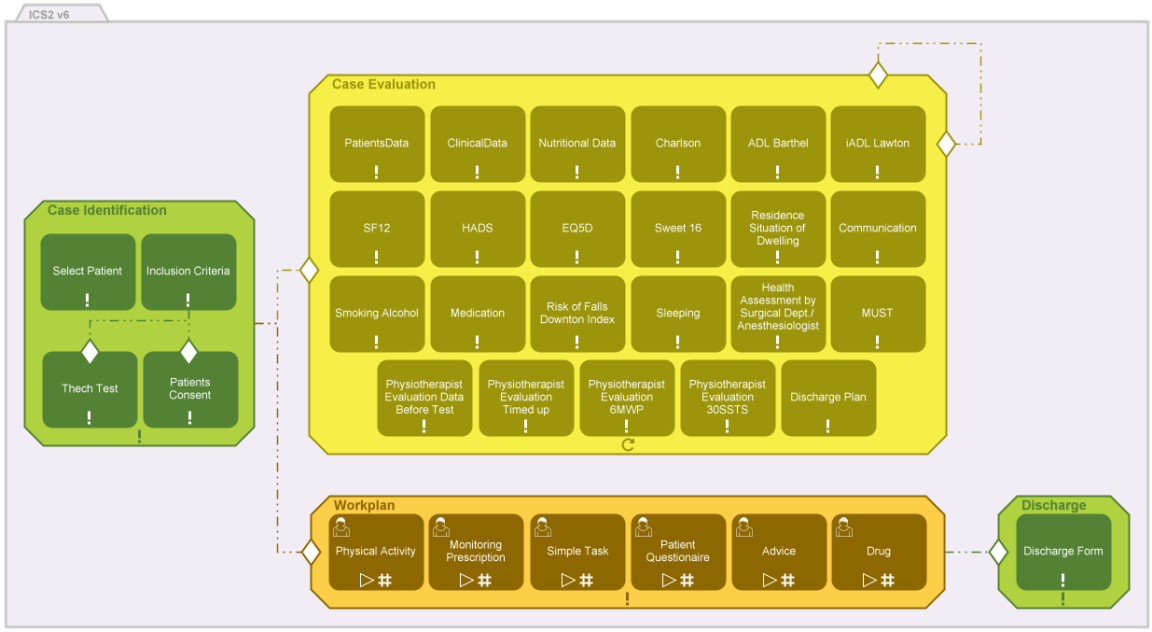


Figure A.9.: Case Study 2 Version 1-6

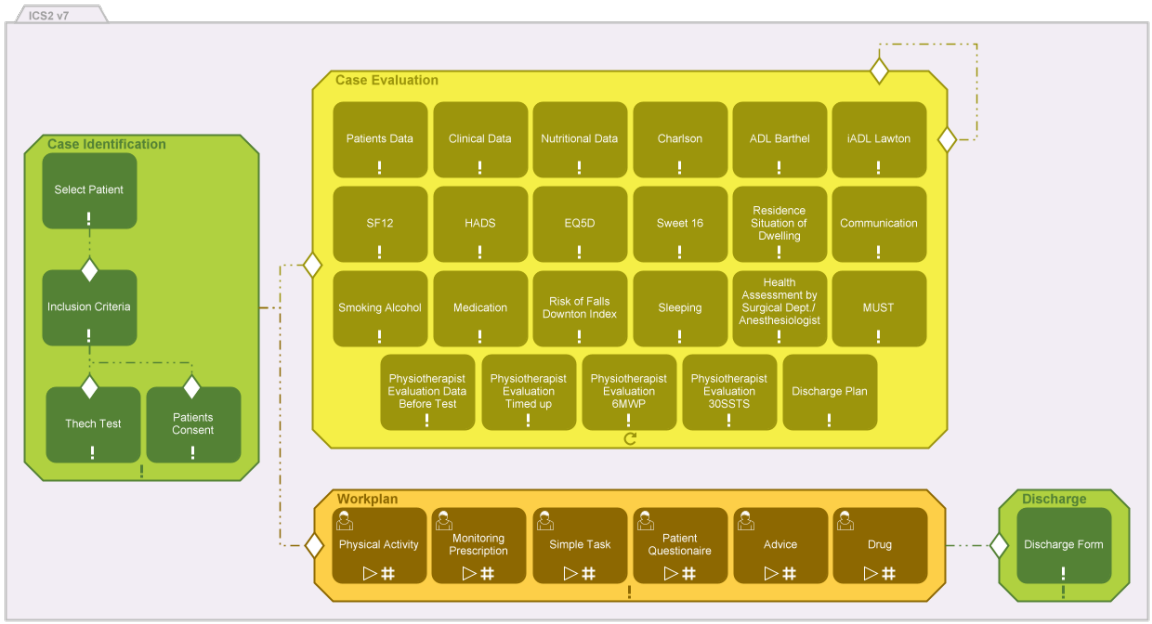


Figure A.10.: Case Study 2 Version 7

A.3. Lleida

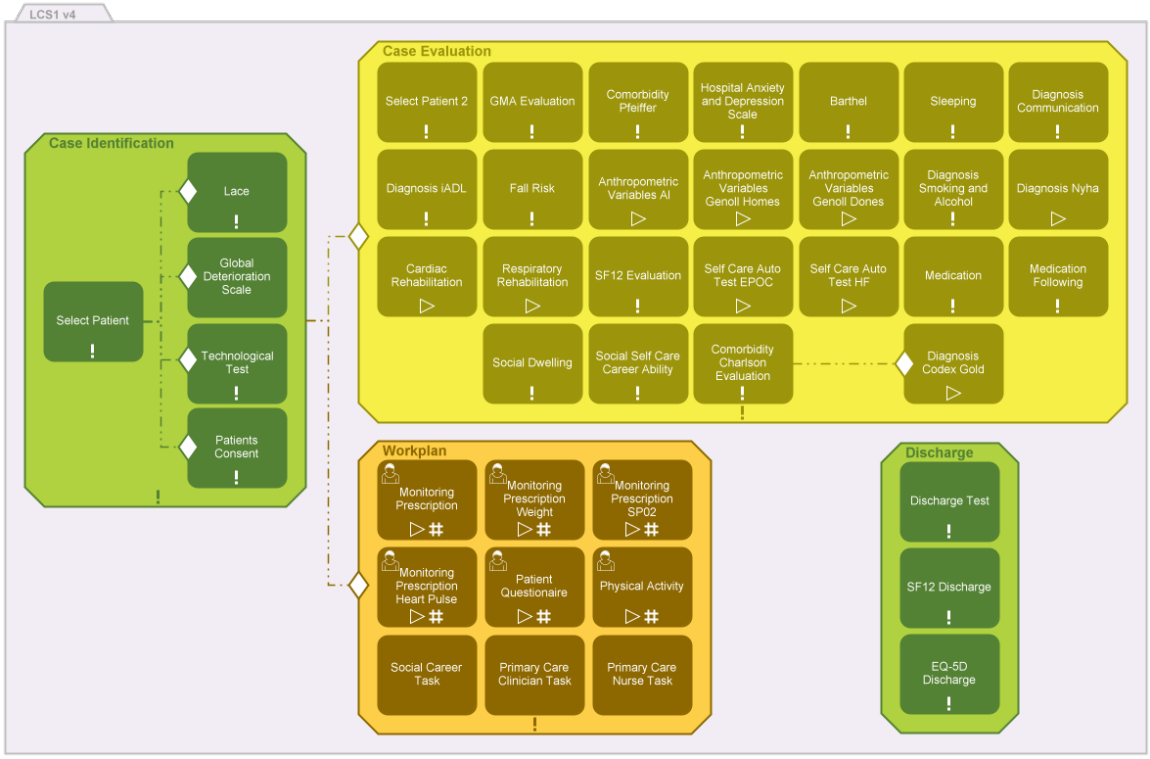


Figure A.11.: Case Study 1 Version 1-4

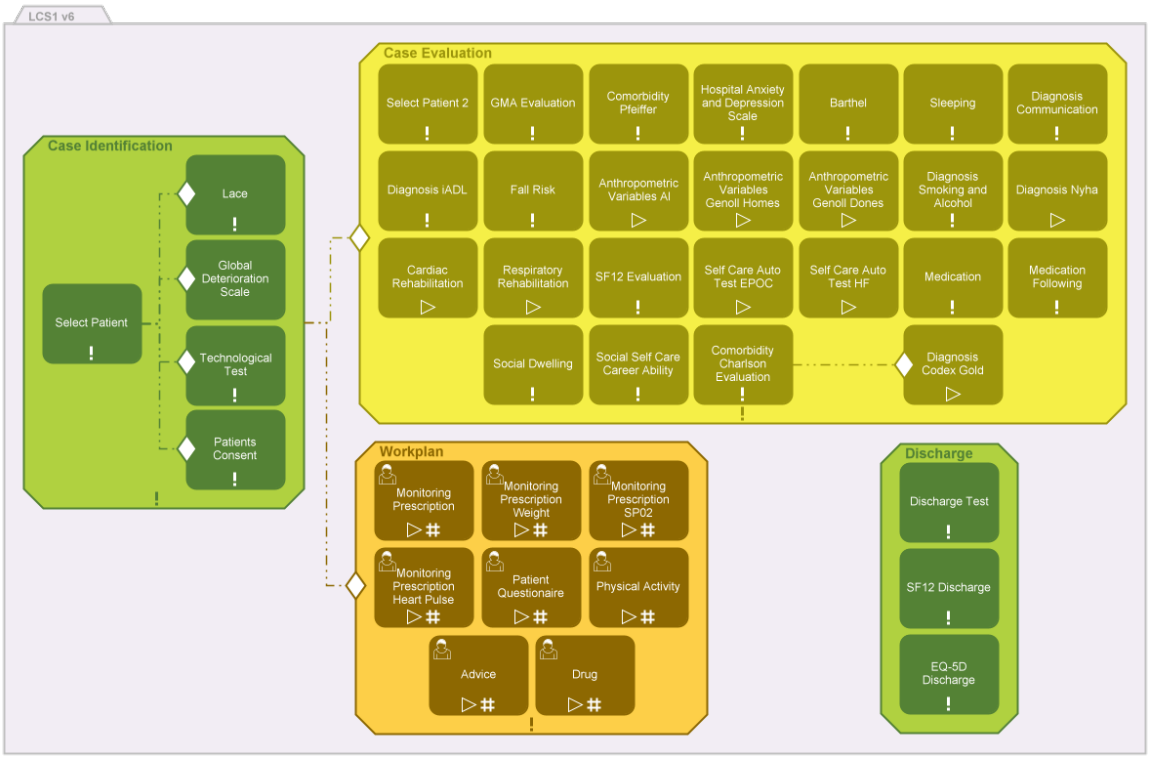


Figure A.12.: Case Study 1 Version 5-6

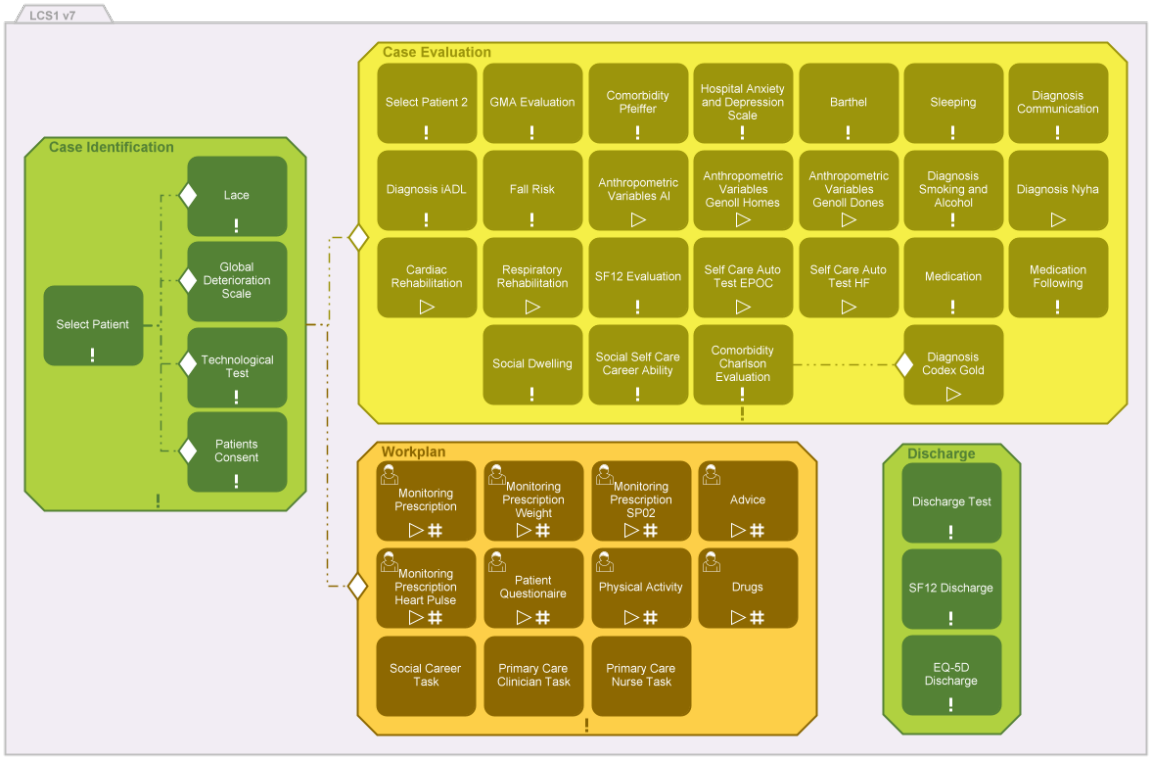


Figure A.13.: Case Study 1 Version 7

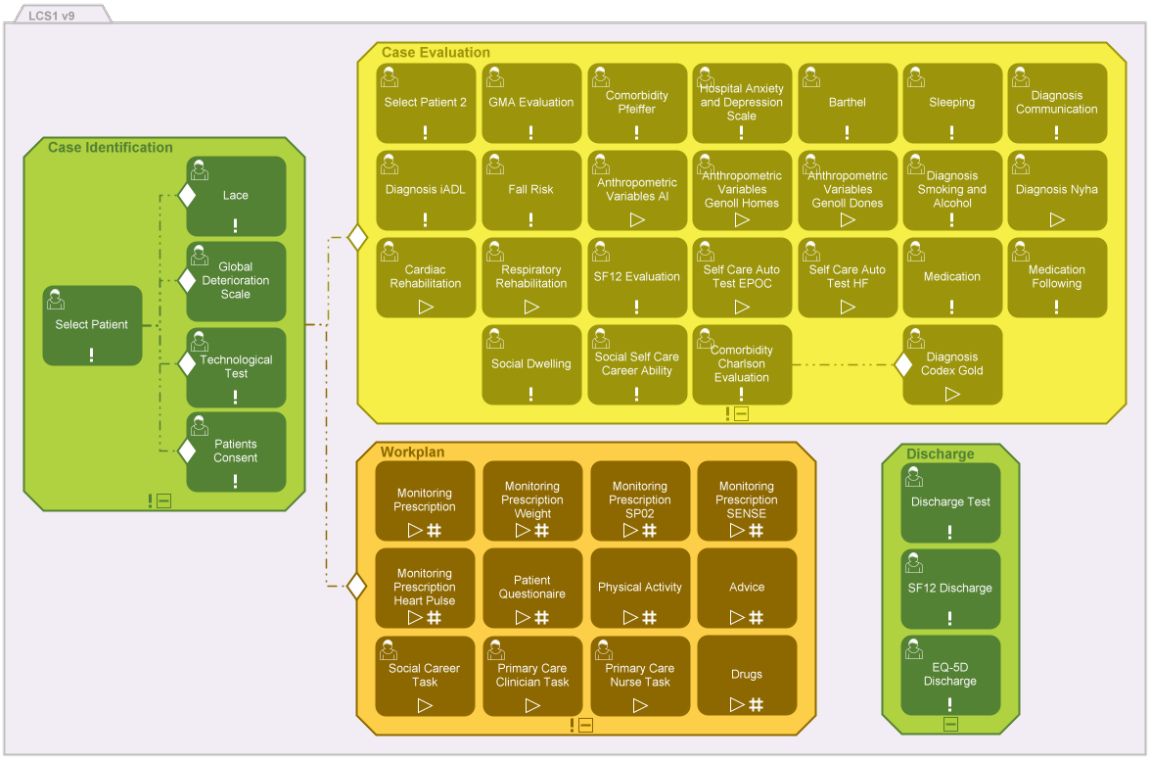


Figure A.14.: Case Study 1 Version 9

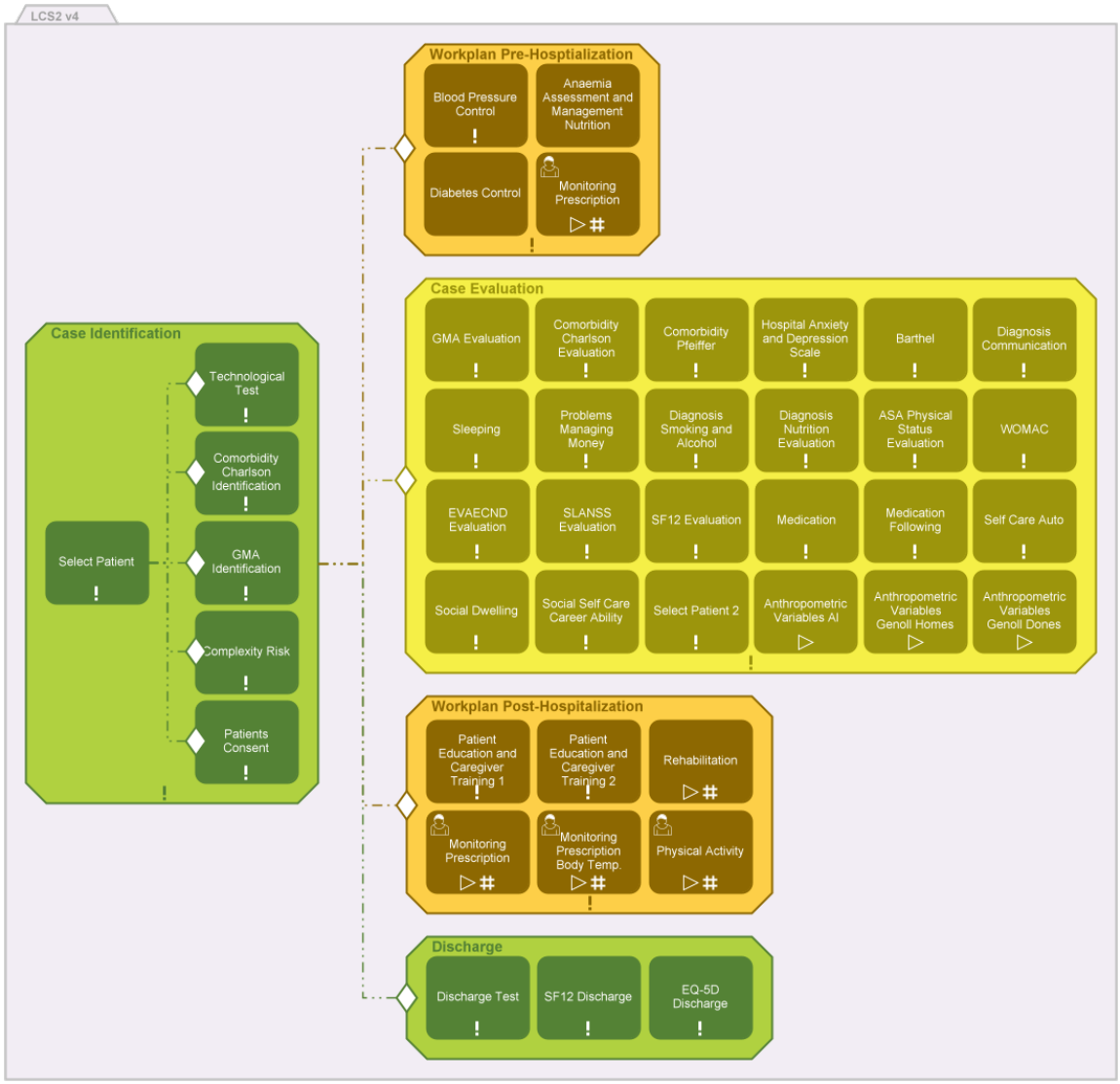


Figure A.15.: Case Study 2 Version 1-4

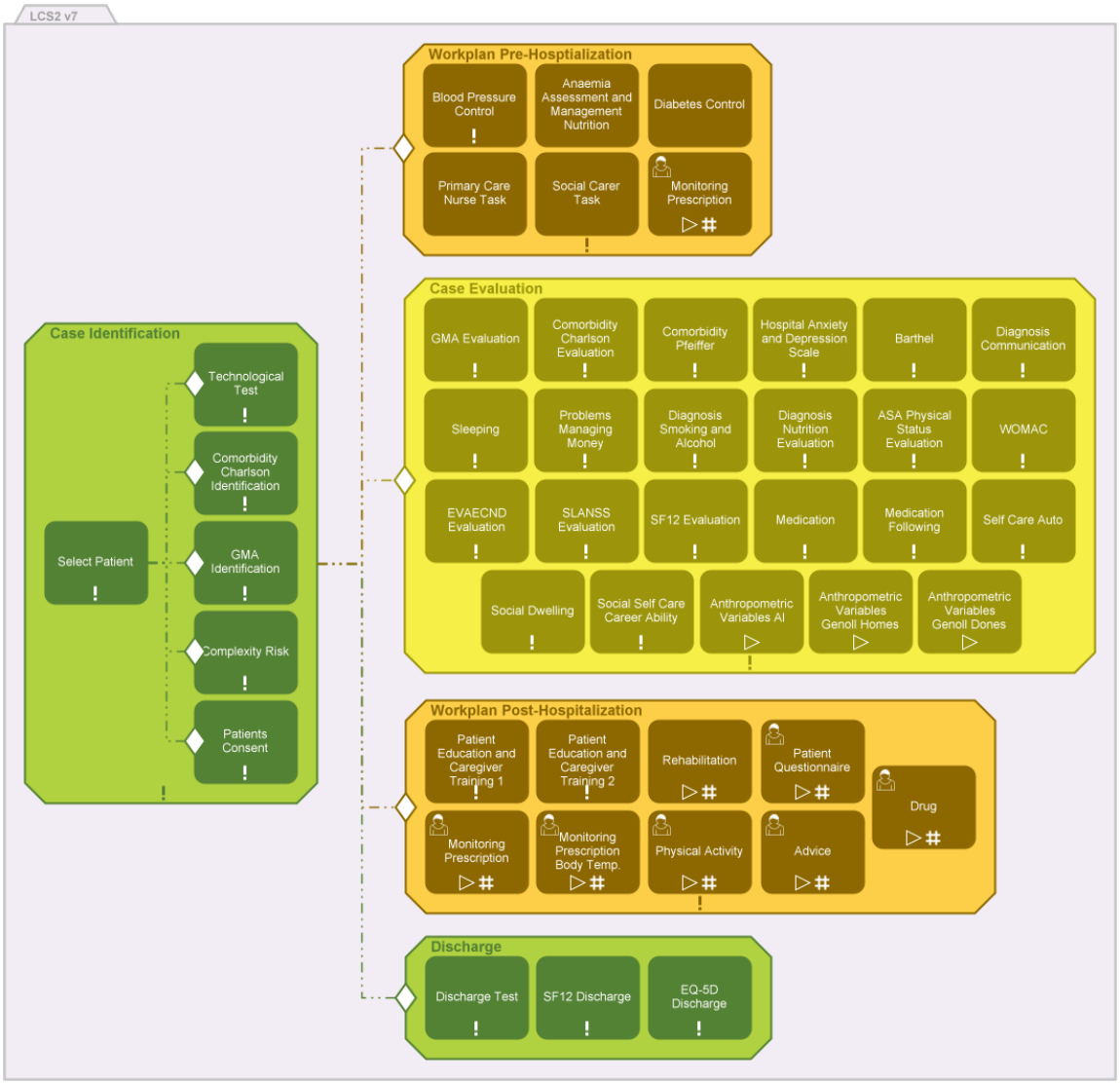


Figure A.16.: Case Study 2 Version 5-7

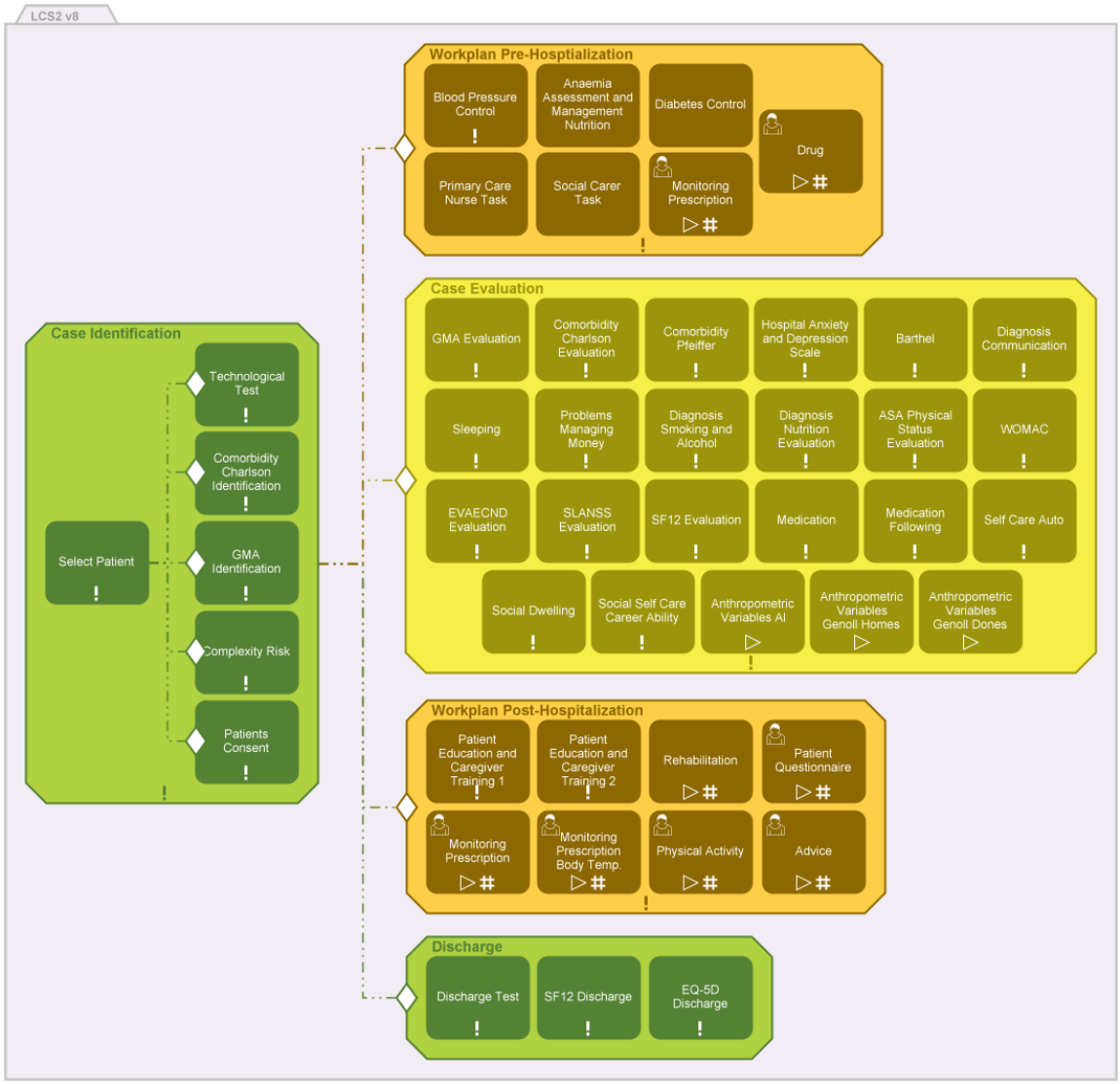


Figure A.17.: Case Study 2 Version 8

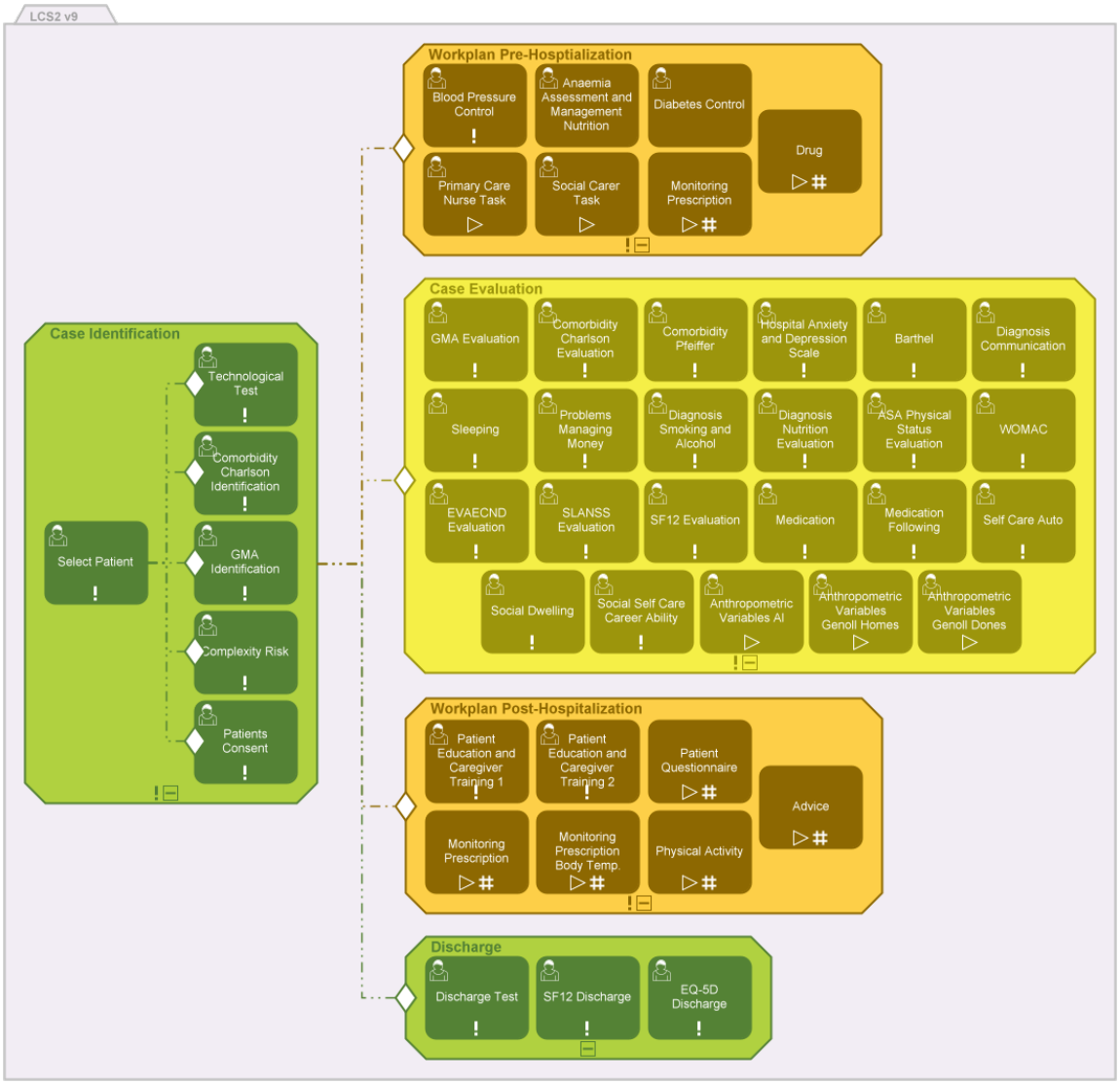


Figure A.18.: Case Study 2 Version 9

B. Process Maps

B.1. System View

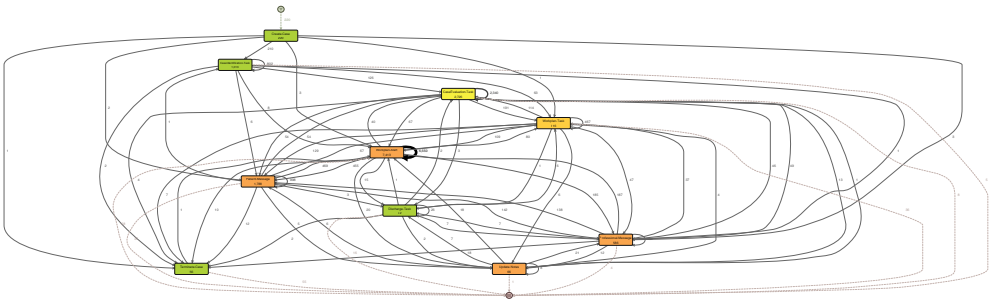


Figure B.1.: System view without any abstractions

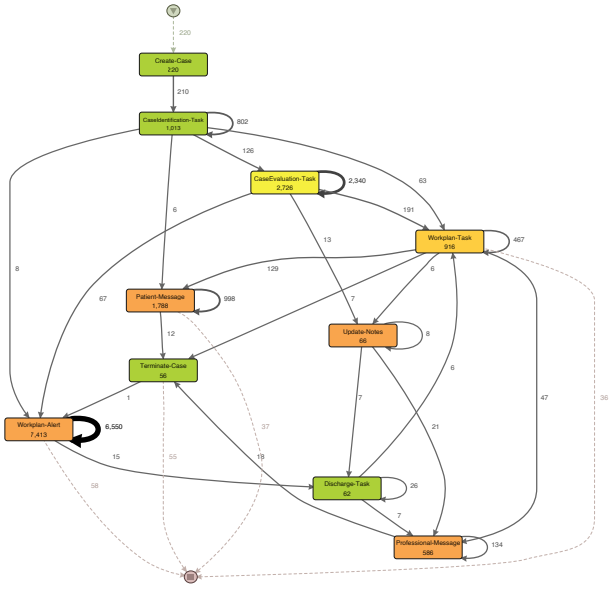


Figure B.2.: System view at 45% path abstraction level

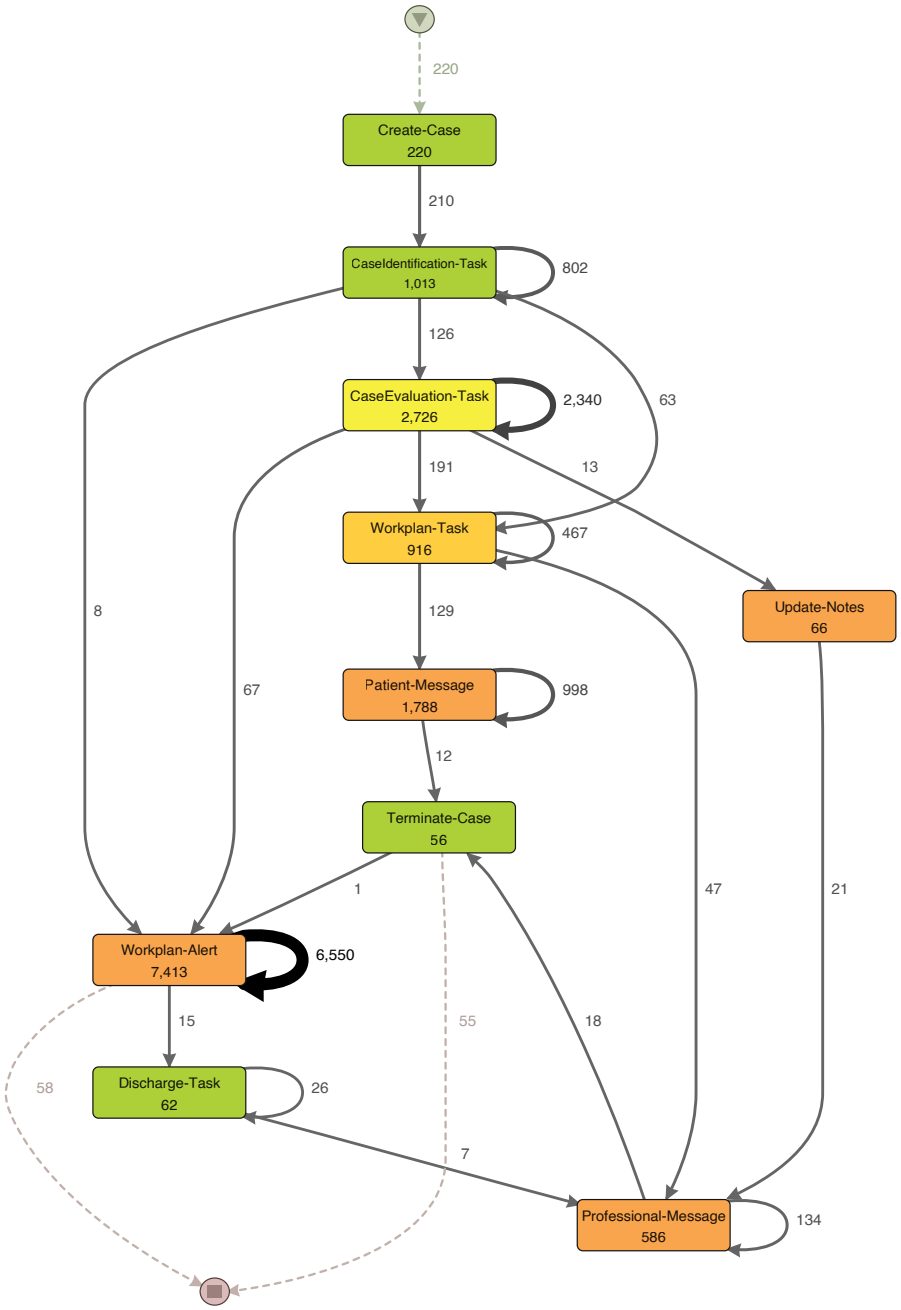


Figure B.3.: System view at 28% path abstraction level

B.2. Case View

B.2.1. Groningen

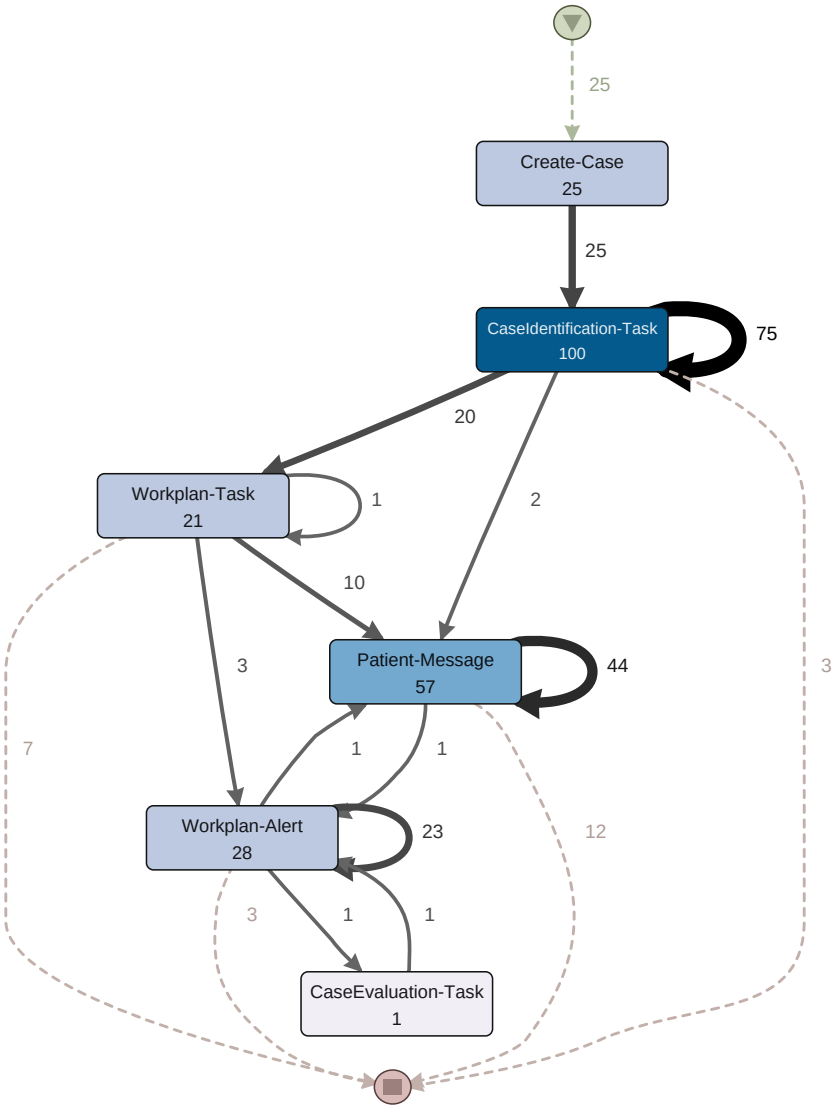


Figure B.4.: Case study 1 at Groningen without abstractions

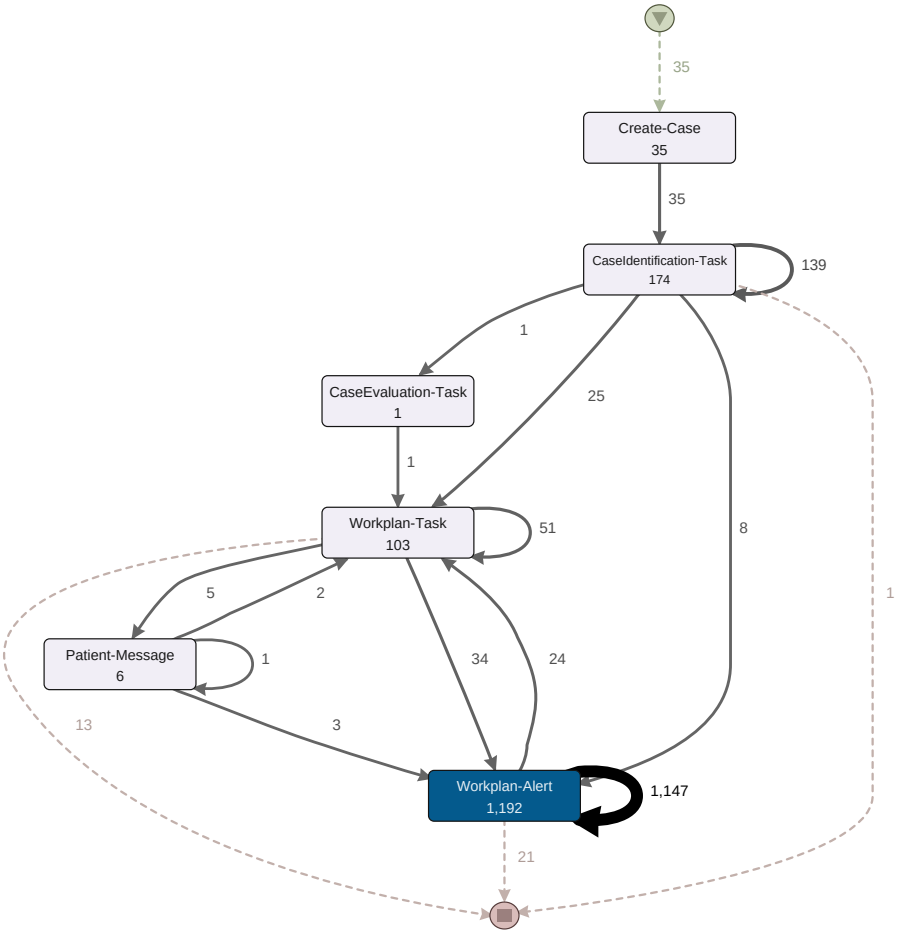
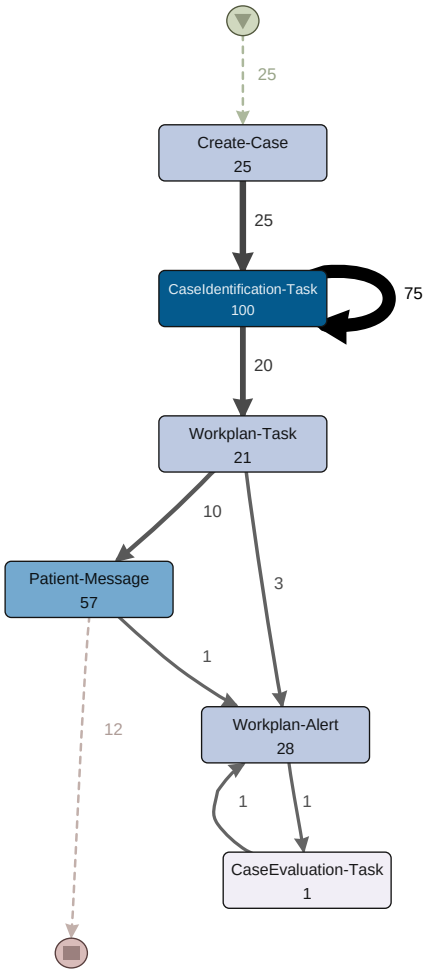
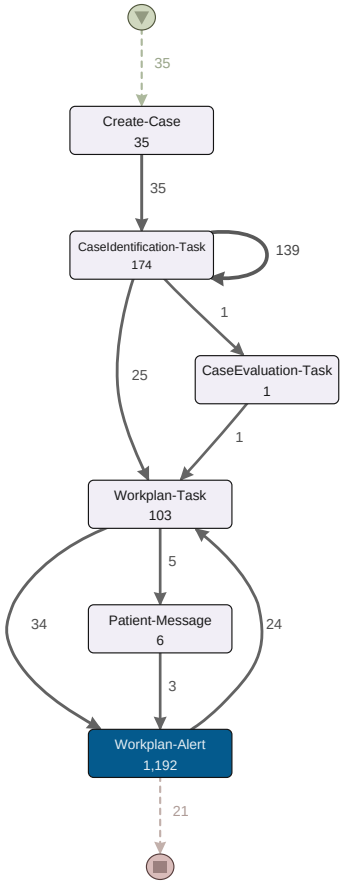


Figure B.5.: Case study 2 at Groningen without abstractions



(a) Case study 1



(b) Case study 2

Figure B.6.: Case studies at Groningen at full path abstraction level

B.2.2. Tel-Aviv

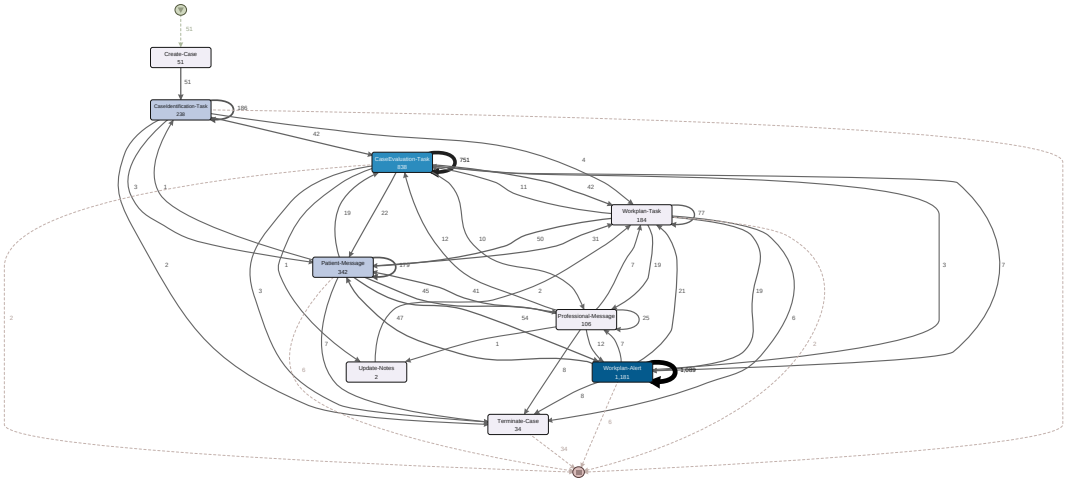


Figure B.7.: Case study 1 at Tel-Aviv without abstractions

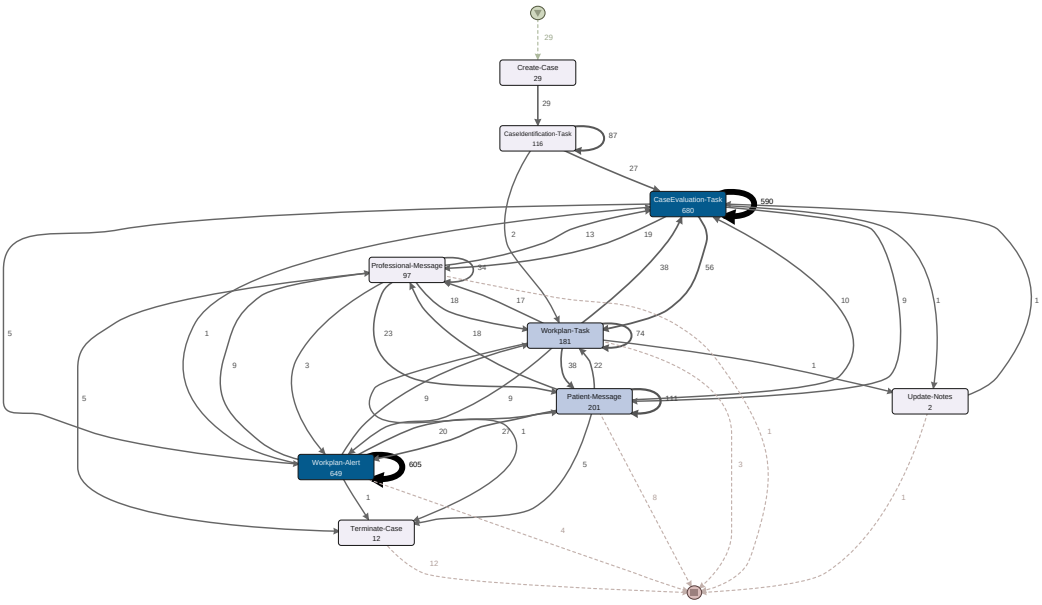
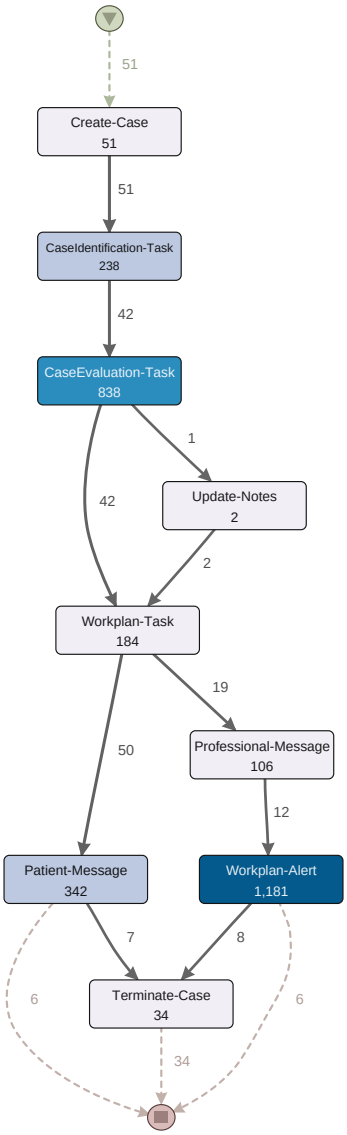
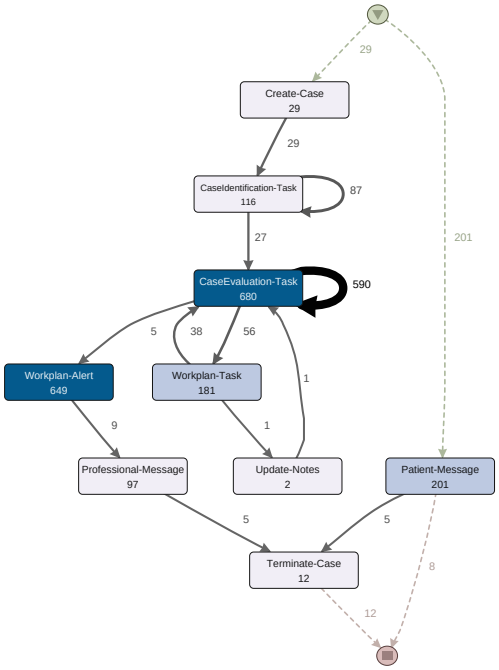


Figure B.8.: Case study 2 at Tel-Aviv without abstractions



(a) Case study 1



(b) Case study 2

Figure B.9.: Case studies at Tel-Aviv at full path abstraction level

B.2.3. Lleida

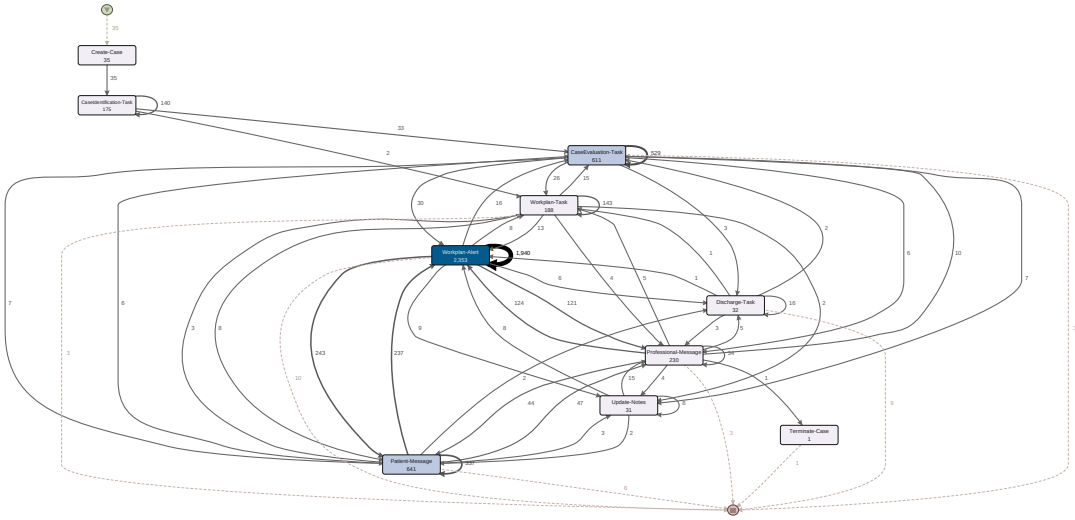


Figure B.10.: Case study 1 at Lleida without abstractions

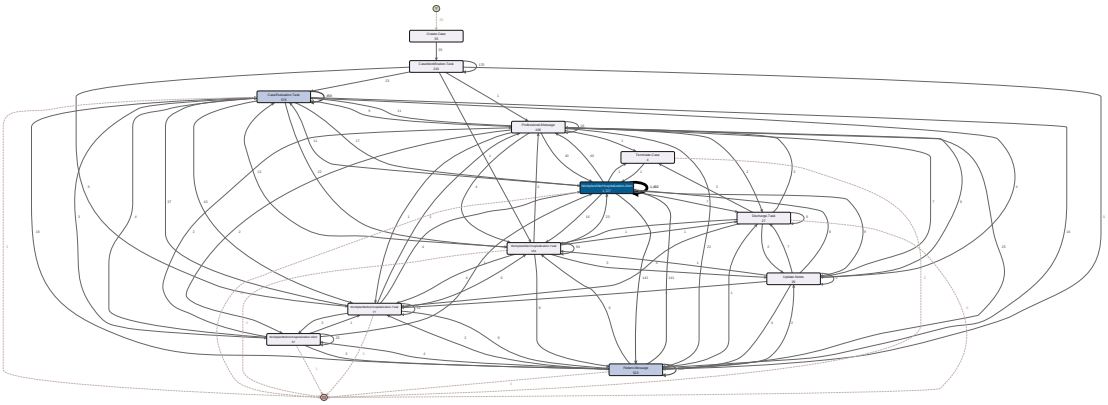


Figure B.11.: Case study 2 at Lleida without abstractions

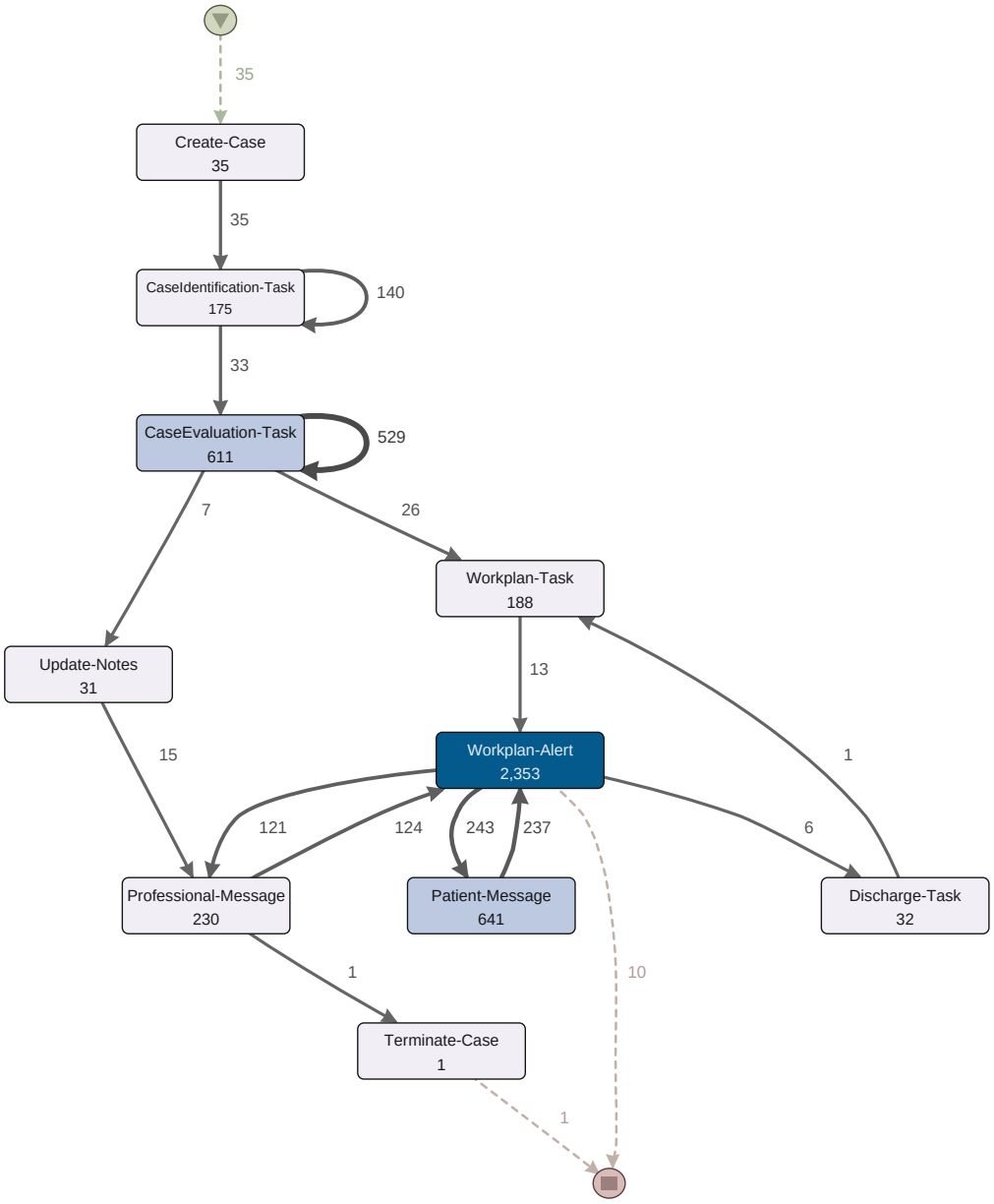


Figure B.12.: Case study 1 at Lleida at full path abstraction level

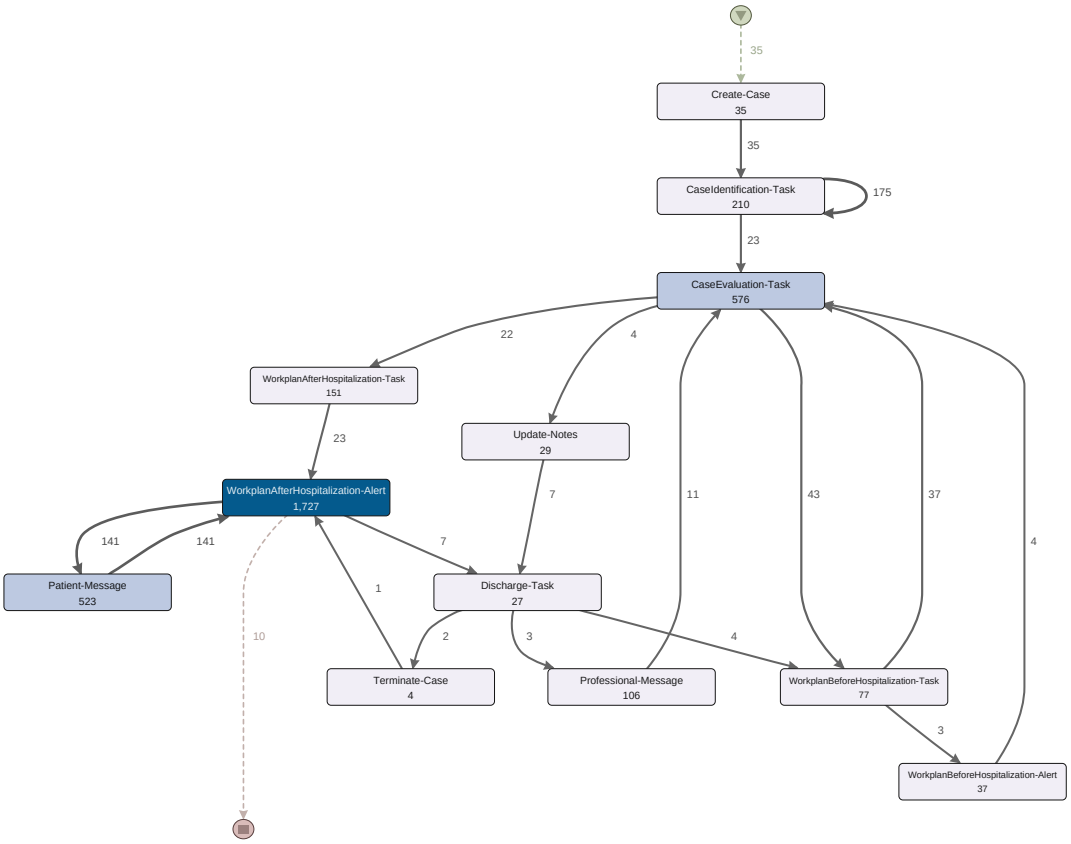


Figure B.13.: Case study 2 at Lleida at full path abstraction level

B.3. Stage View

If not specified otherwise, maps do not have any abstractions applied.

B.3.1. Groningen Case Study 1

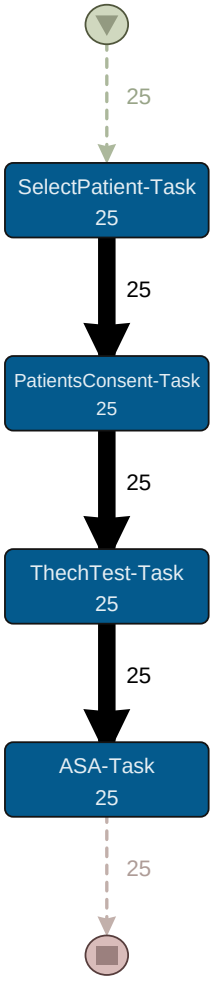


Figure B.14.: Case Identification

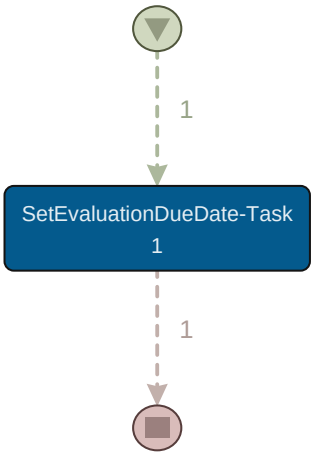


Figure B.15.: Case Evaluation

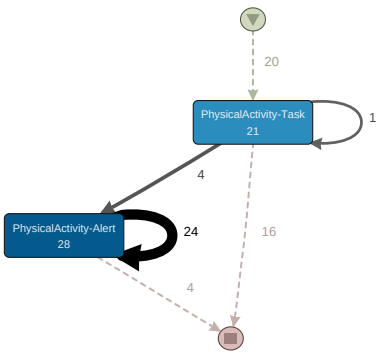


Figure B.16.: Workplan

B.3.2. Groningen Case Study 2

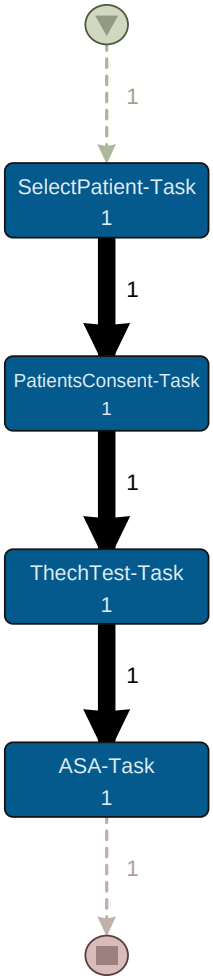


Figure B.17.: Case Identification versions 1-3

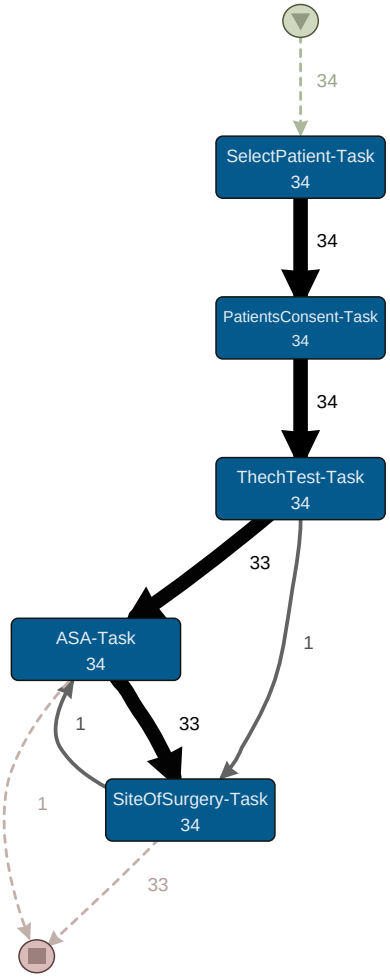


Figure B.18.: Case Identification versions 4+

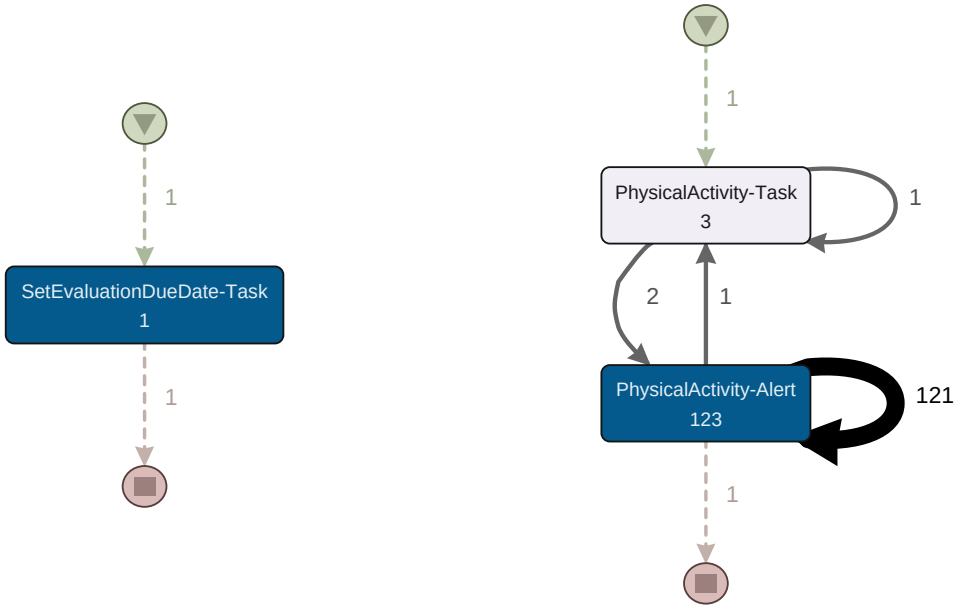


Figure B.19.: Case Evaluation

Figure B.20.: Workplan versions 1-3

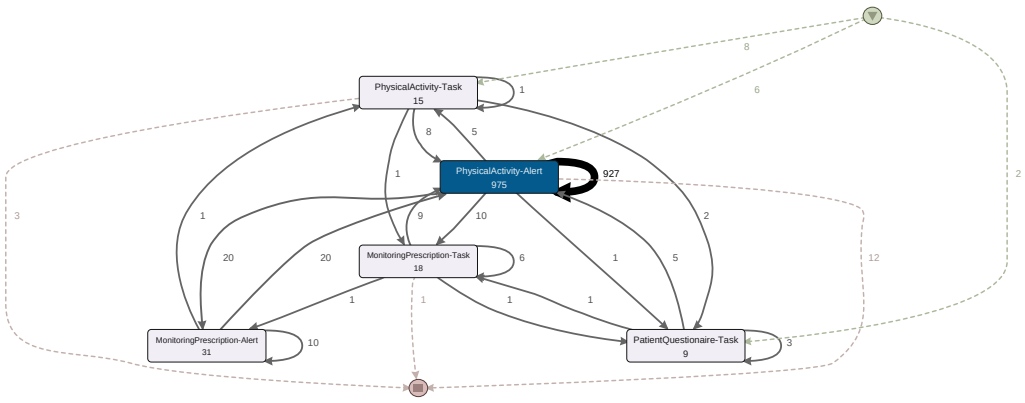


Figure B.21.: Workplan versions 4-7

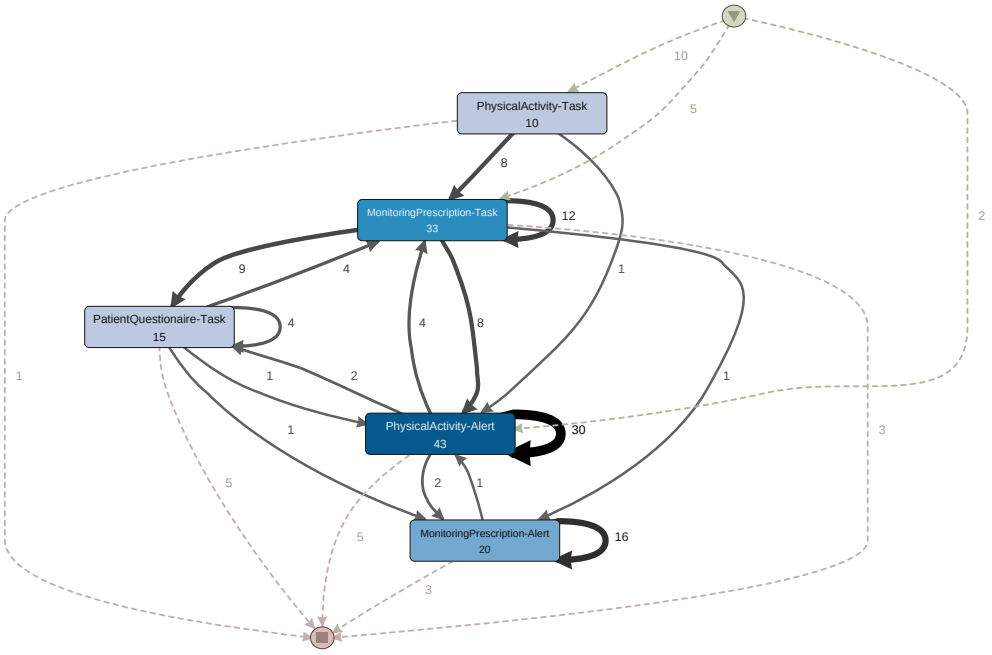


Figure B.22.: Workplan versions 8+

B.3.3. Tel-Aviv Case Study 1

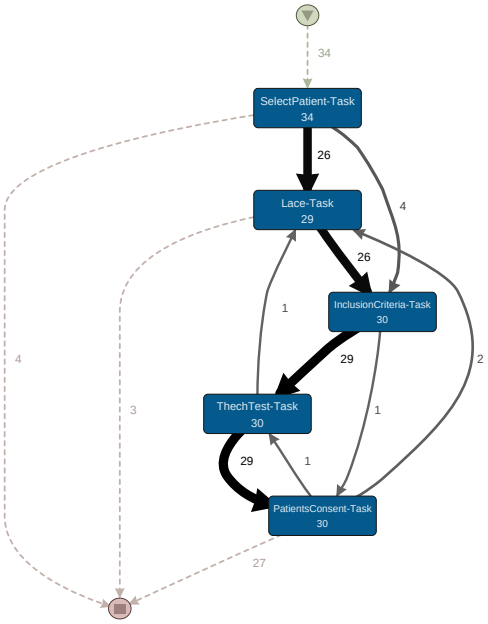


Figure B.23.: Case Identification versionss 1-7

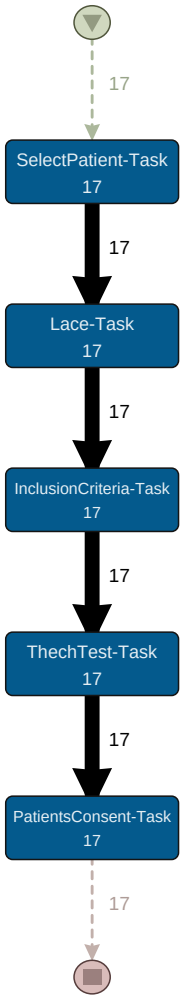


Figure B.24.: Case Identification versions 8+

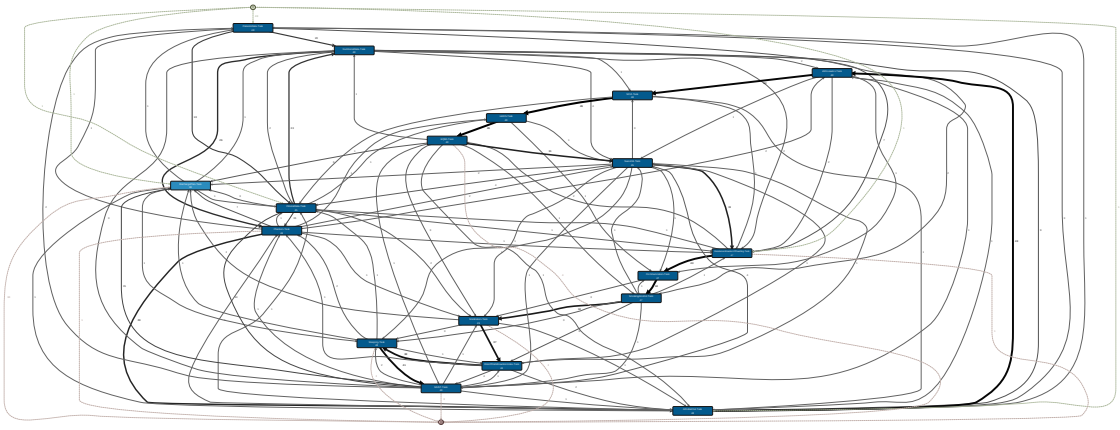


Figure B.25.: Case Evaluation

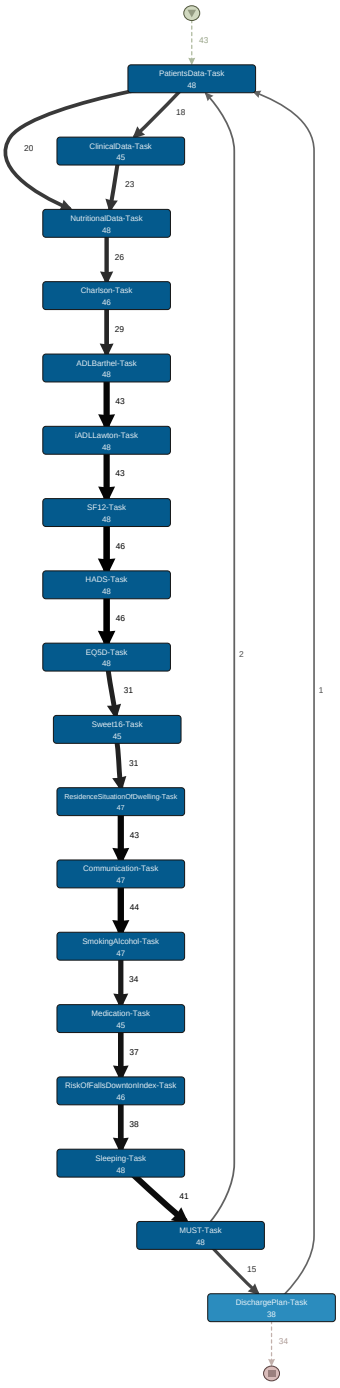


Figure B.26.: Case Evaluation at full path abstraction

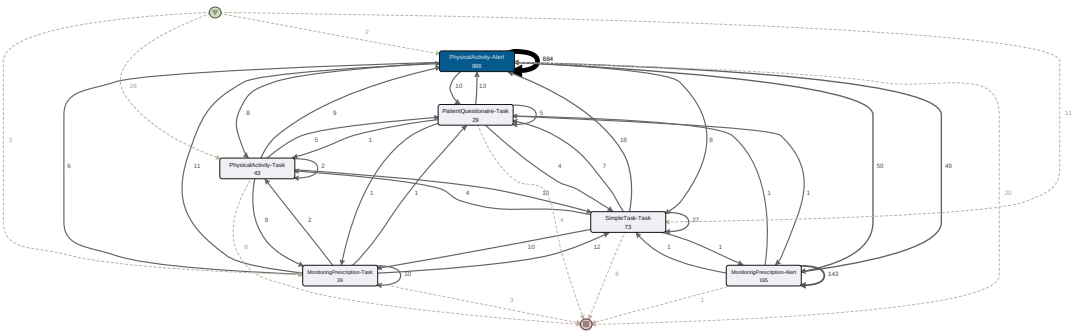


Figure B.27.: Workplan

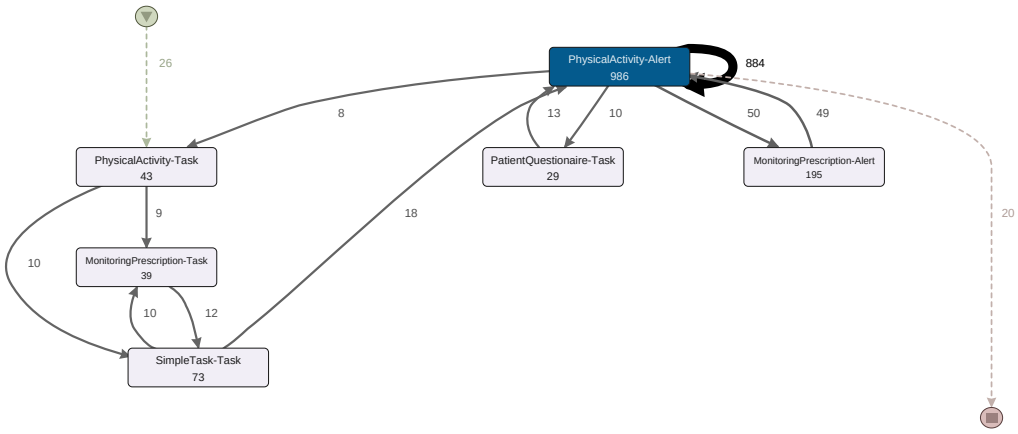


Figure B.28.: Workplan at full path abstraction

B.3.4. Tel-Aviv Case Study 2

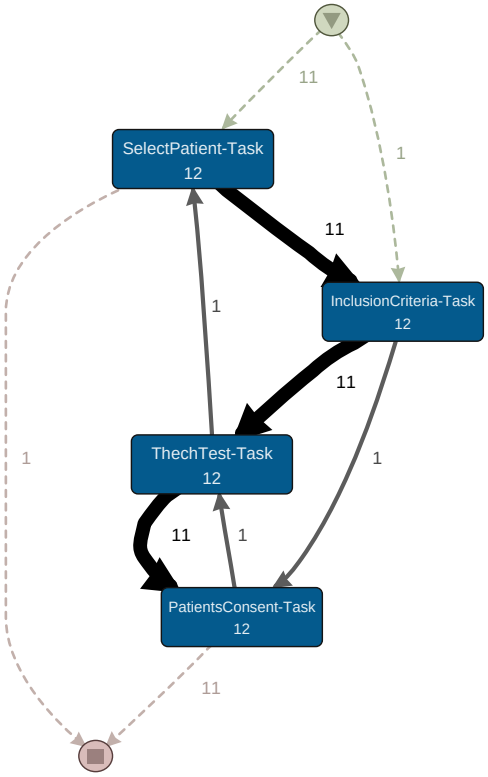


Figure B.29.: Case Identification versions 1-6

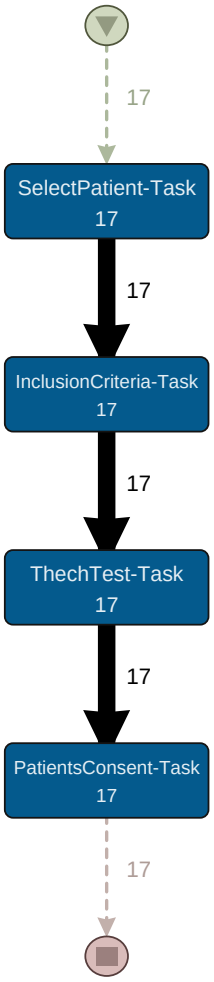


Figure B.30.: Case Identification versions 7+

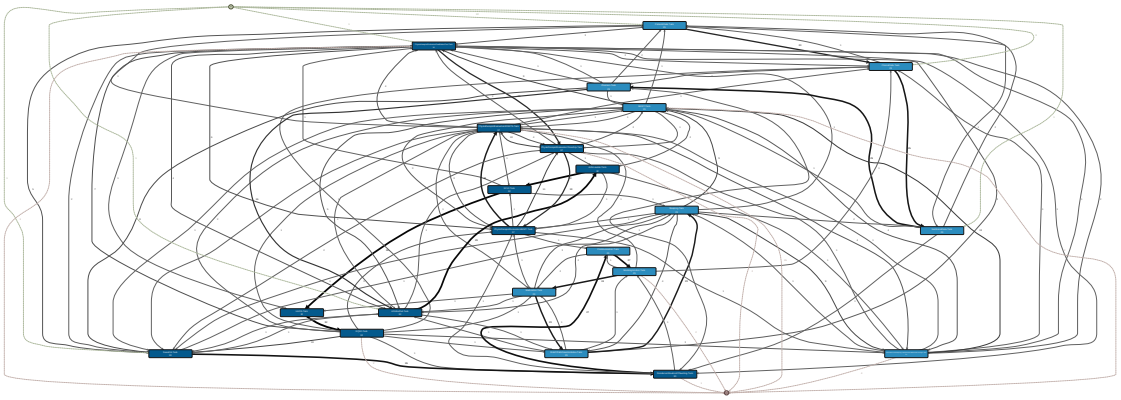


Figure B.31.: Case Evaluation

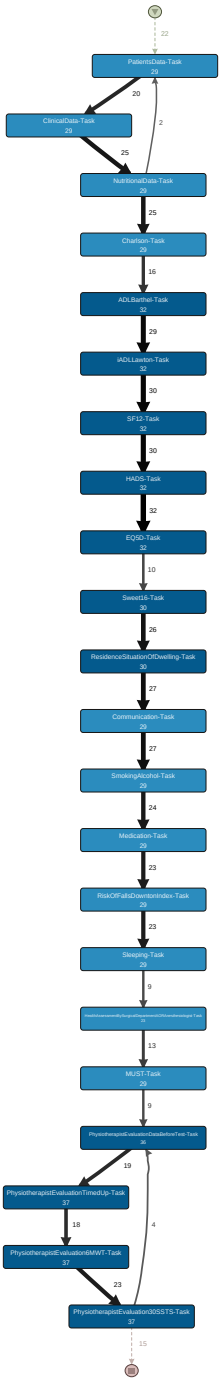


Figure B.32.: Case Evaluation at full path abstraction

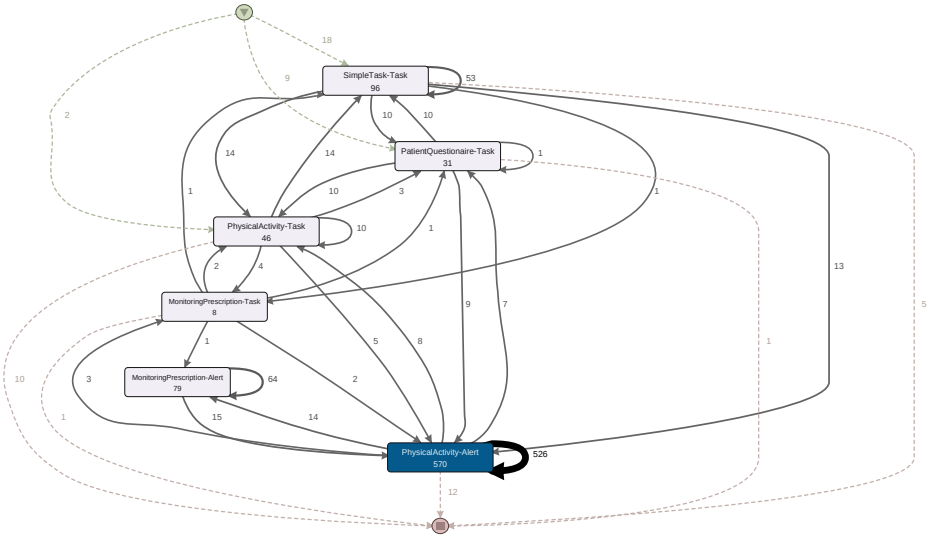


Figure B.33.: Workplan

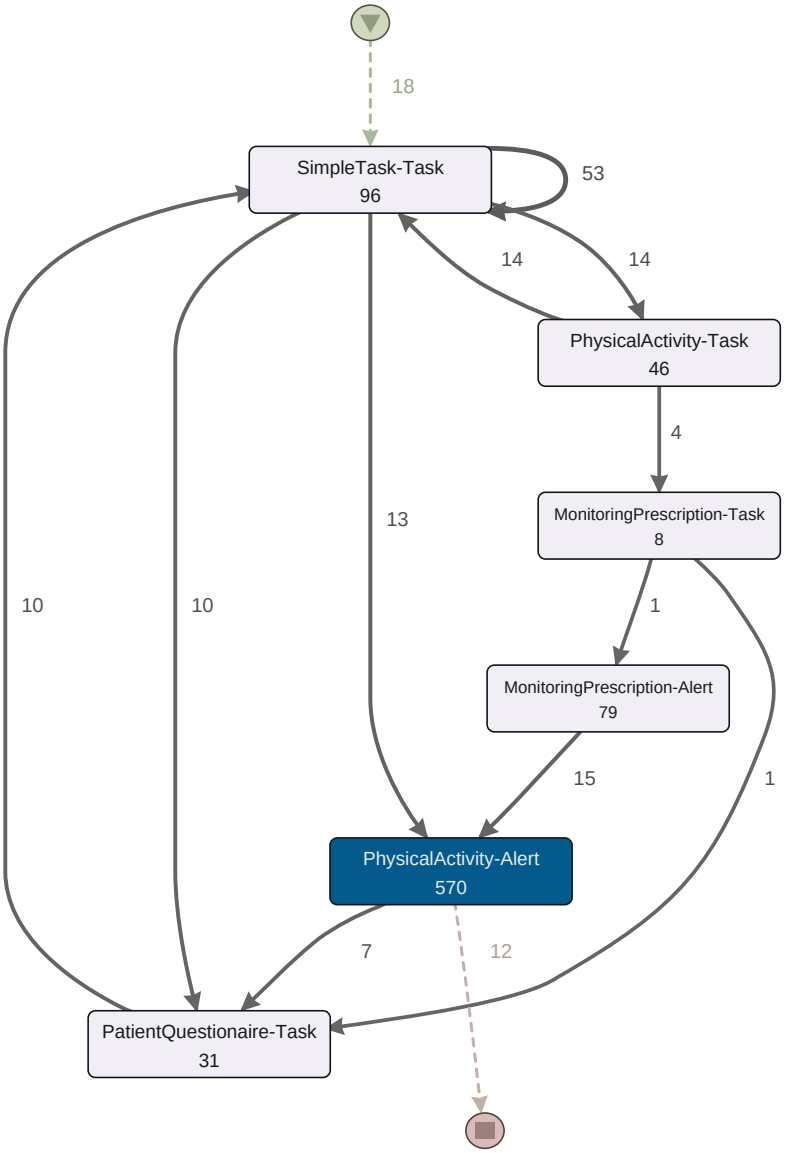


Figure B.34.: Workplan at full path abstraction

B.3.5. Lleida Case Study 1

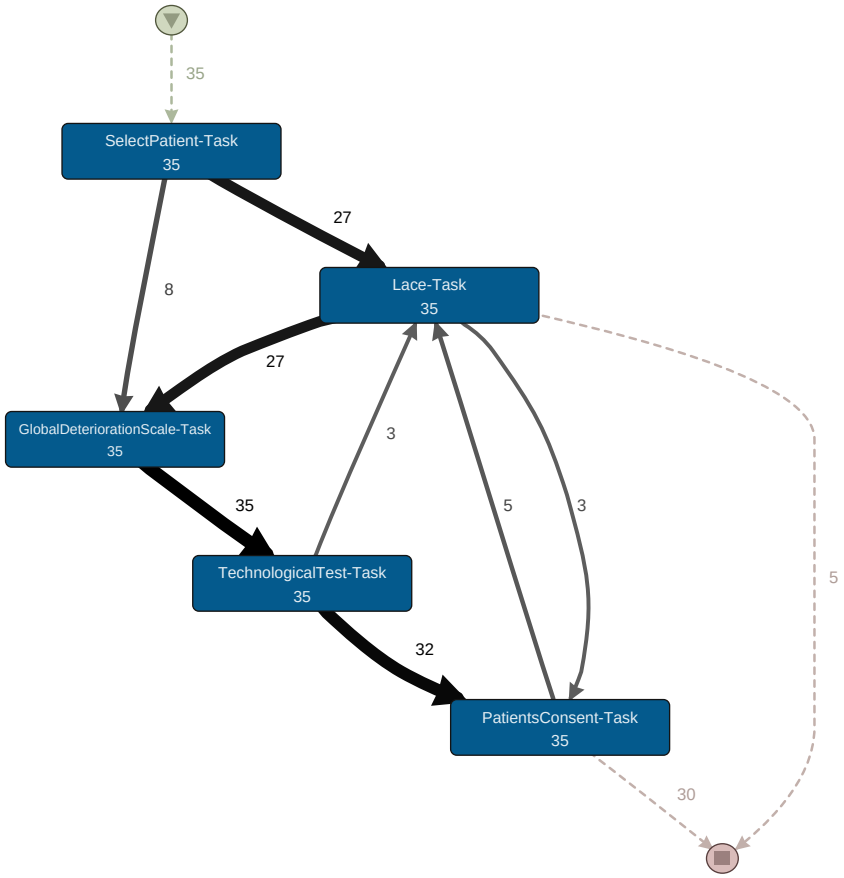


Figure B.35.: Case Identification

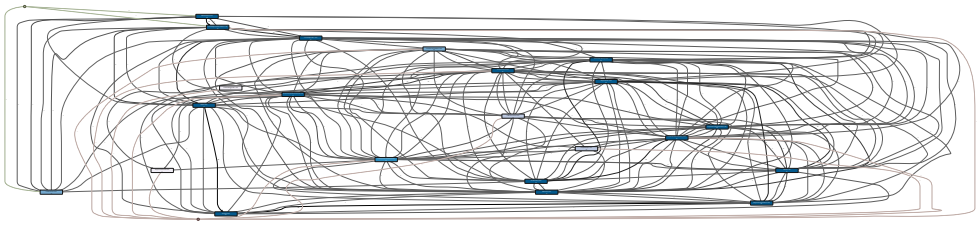


Figure B.36.: Case Evaluation

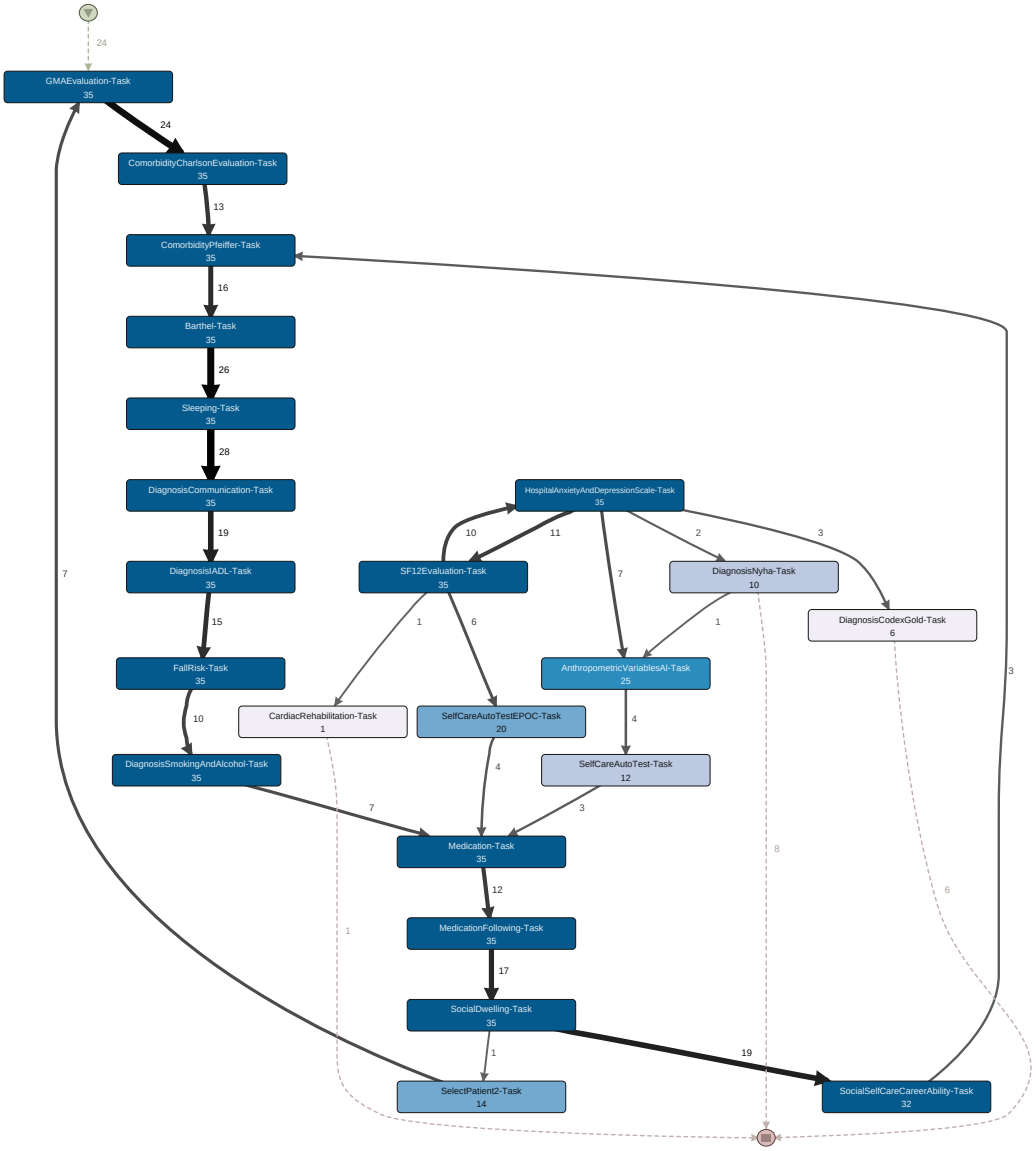


Figure B.37.: Case Evaluation at full path abstraction

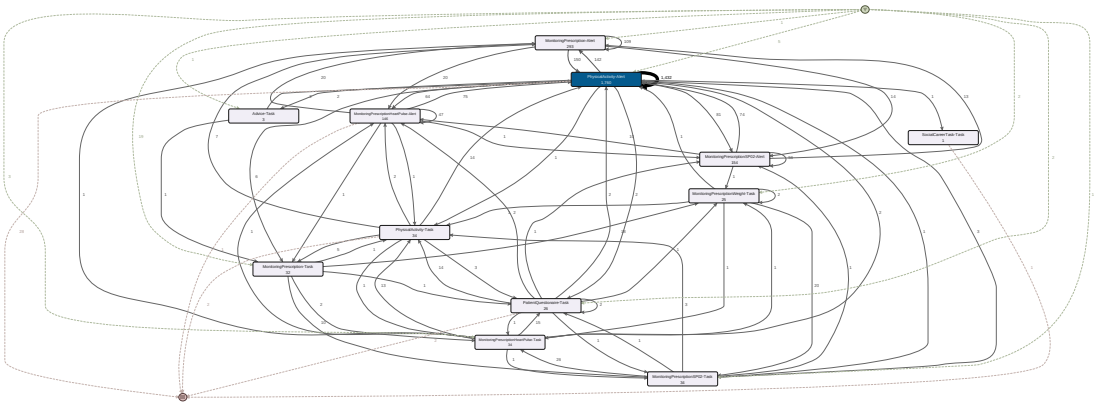


Figure B.38.: Workplan

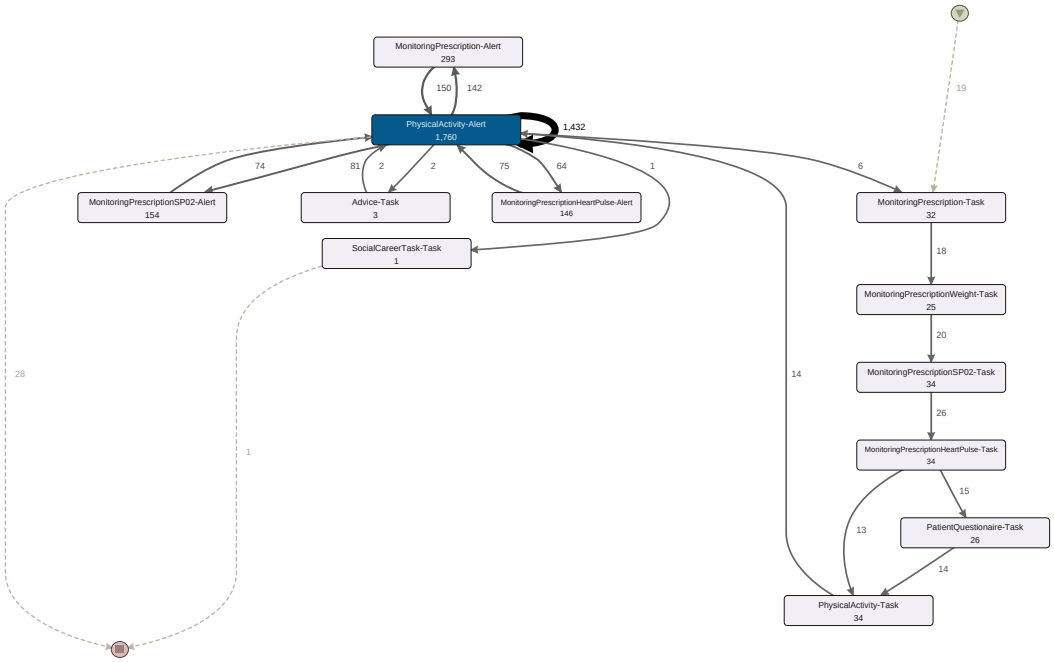


Figure B.39.: Workplan at full path abstraction

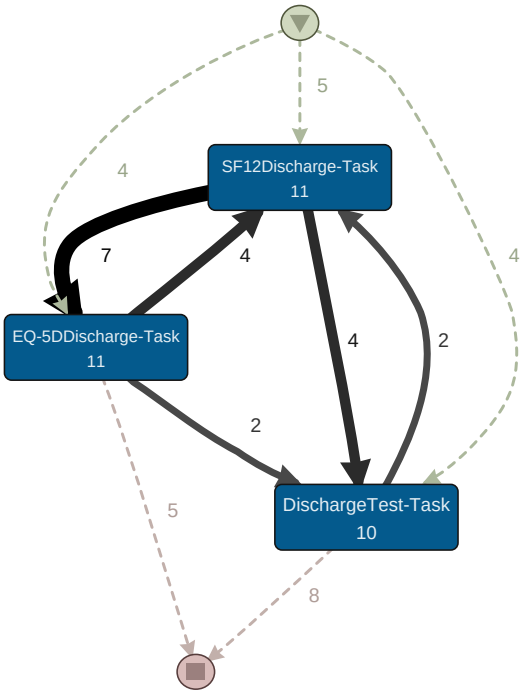


Figure B.40.: Discharge

B.3.6. Lleida Case Study 2

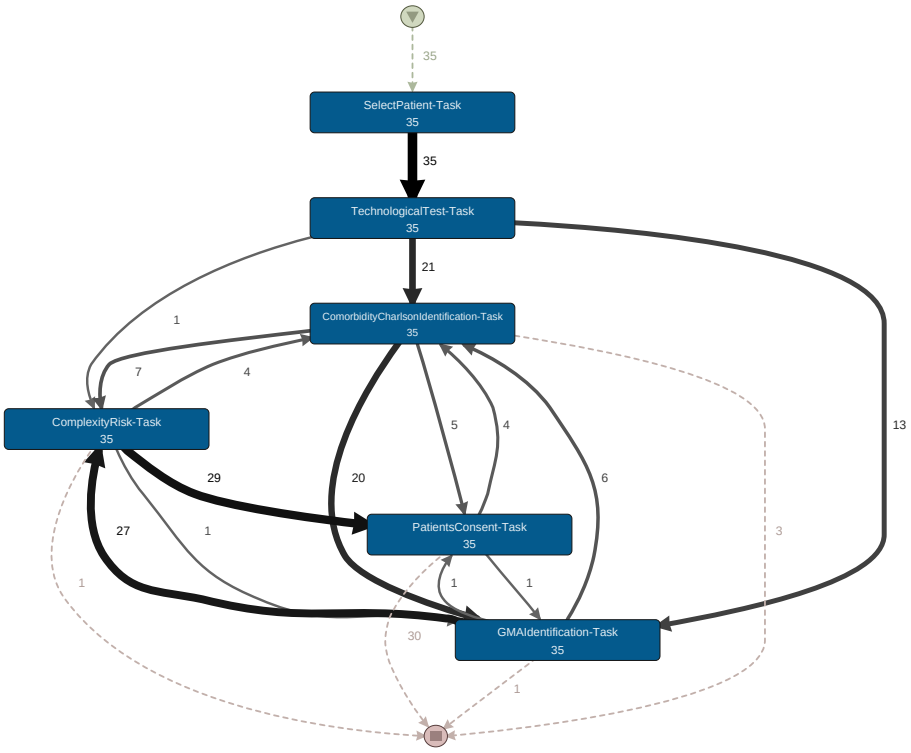


Figure B.41.: Case Identification

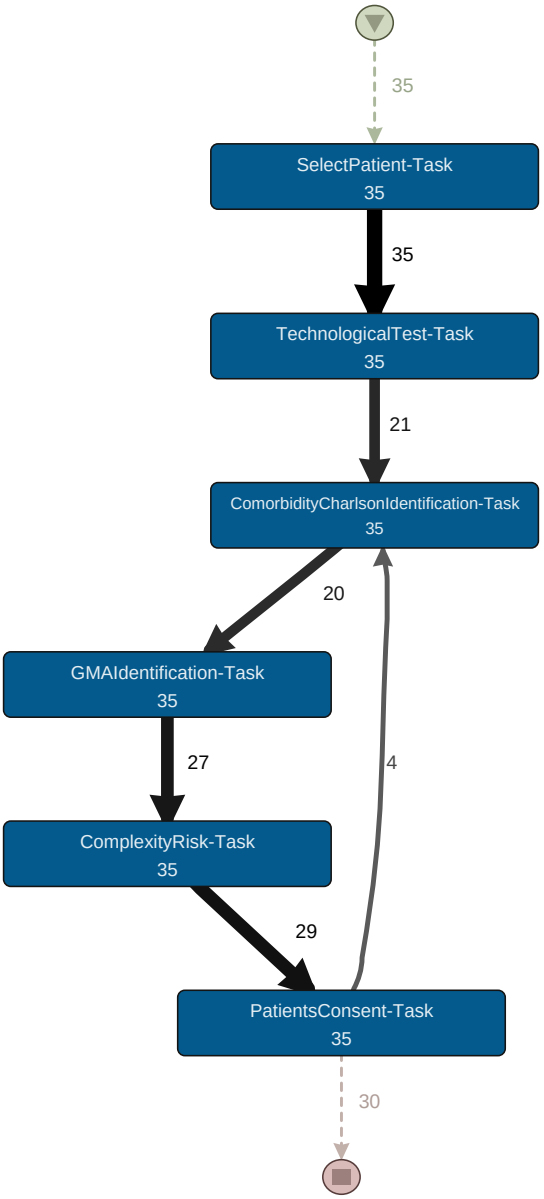


Figure B.42.: Case Identification at full path abstraction

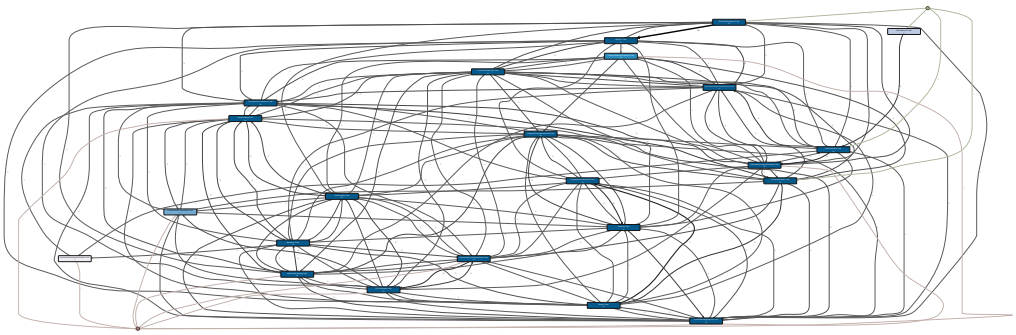


Figure B.43.: Case Evaluation versions 1-8

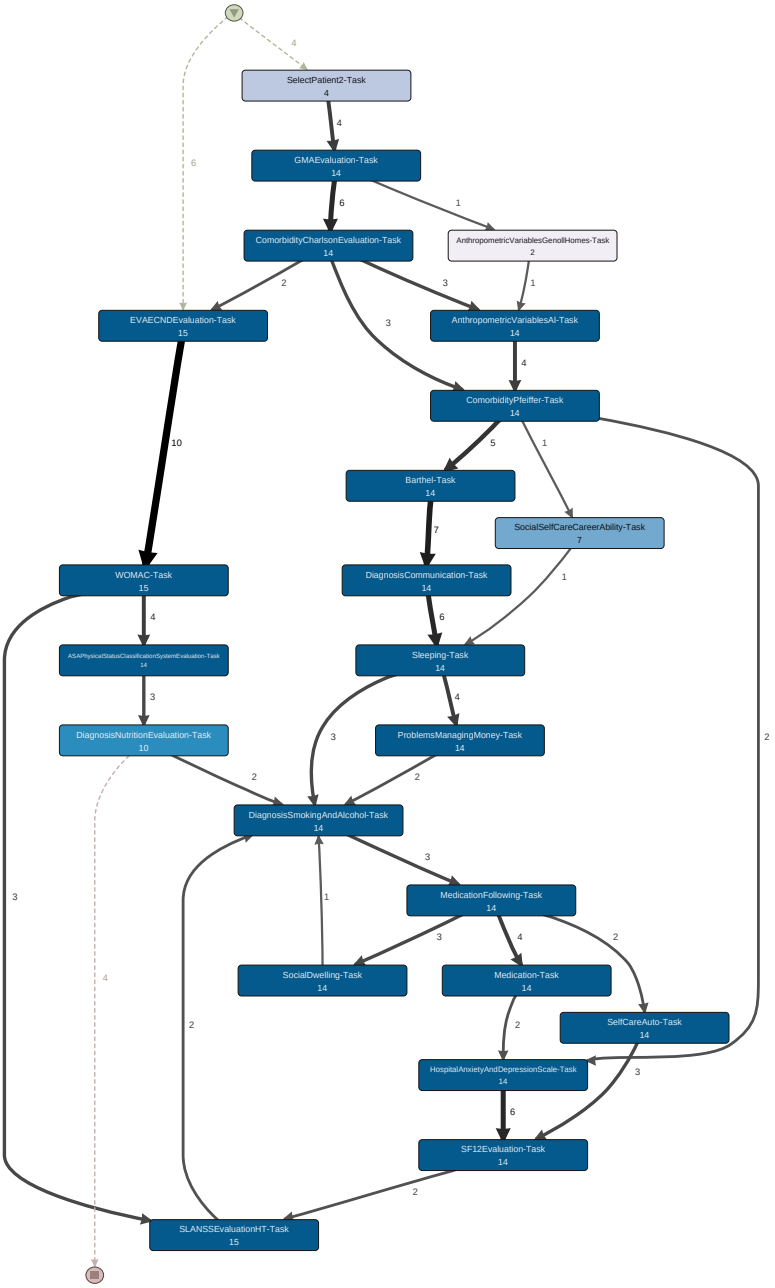


Figure B.44.: Case Evaluation versions 1-8 at full path abstraction

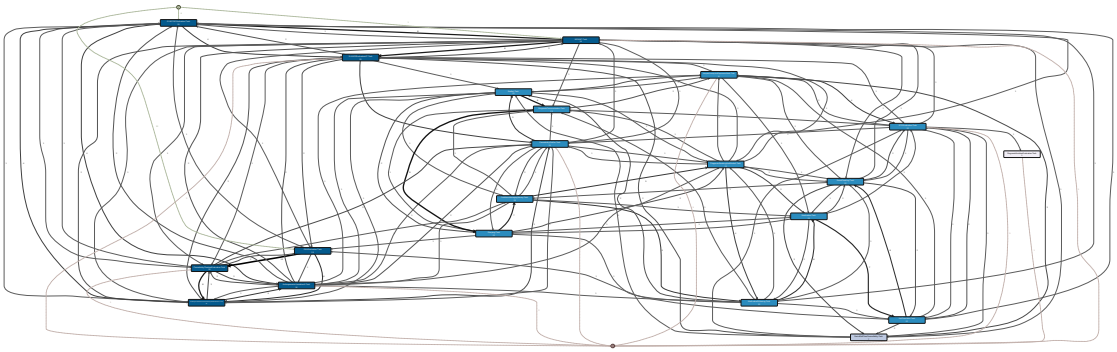


Figure B.45.: Case Evaluation versions 9+

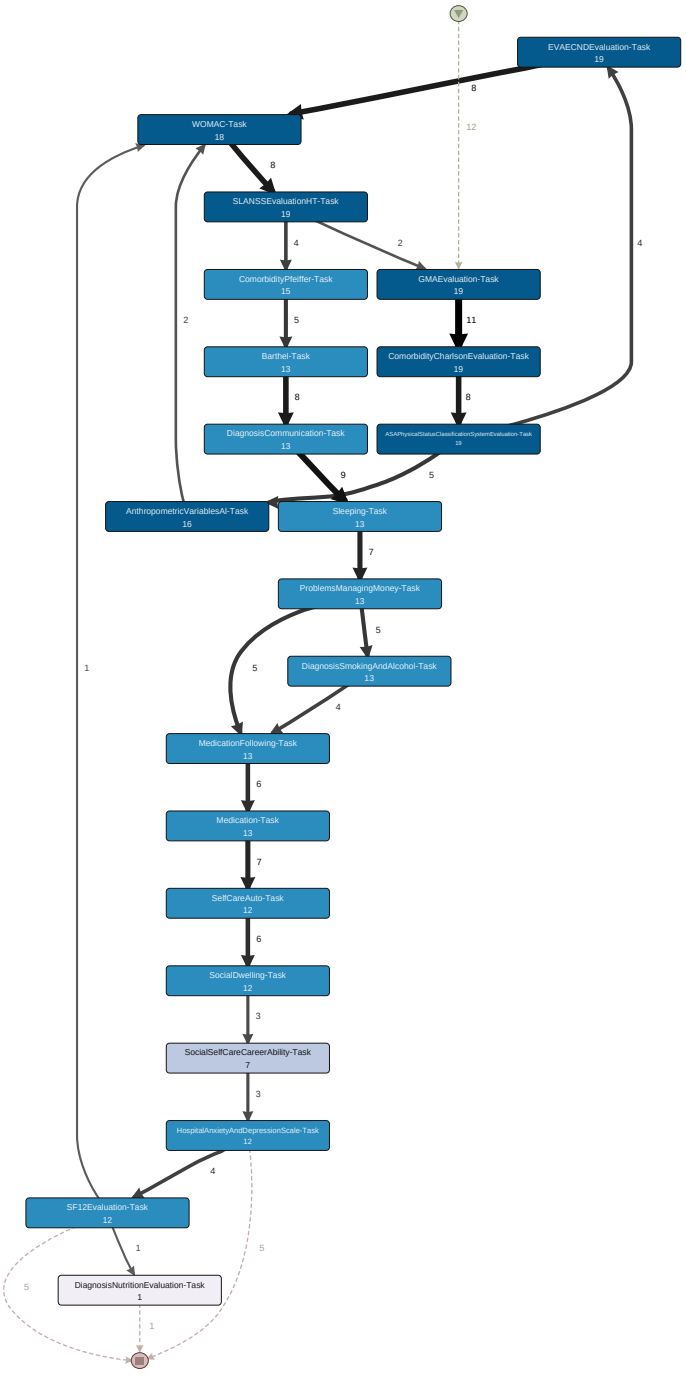


Figure B.46.: Case Evaluation versions 9+ at full path abstraction

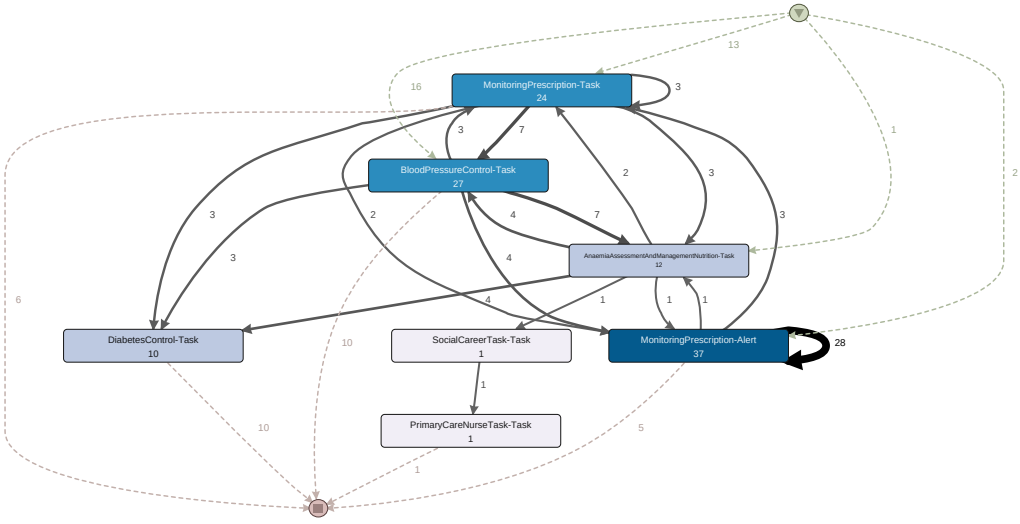


Figure B.47.: Workplan before Hospitalization

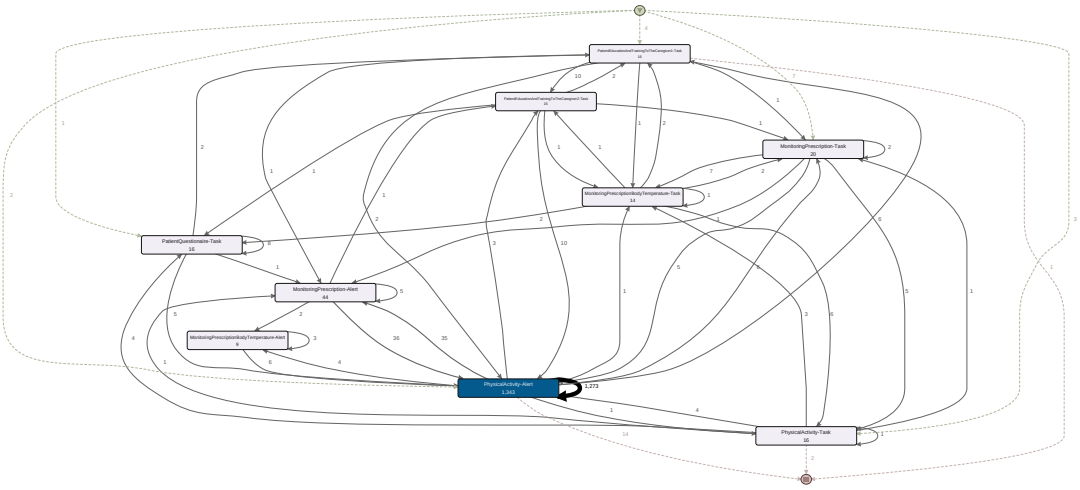


Figure B.48.: Workplan after Hospitalization versions 1-7

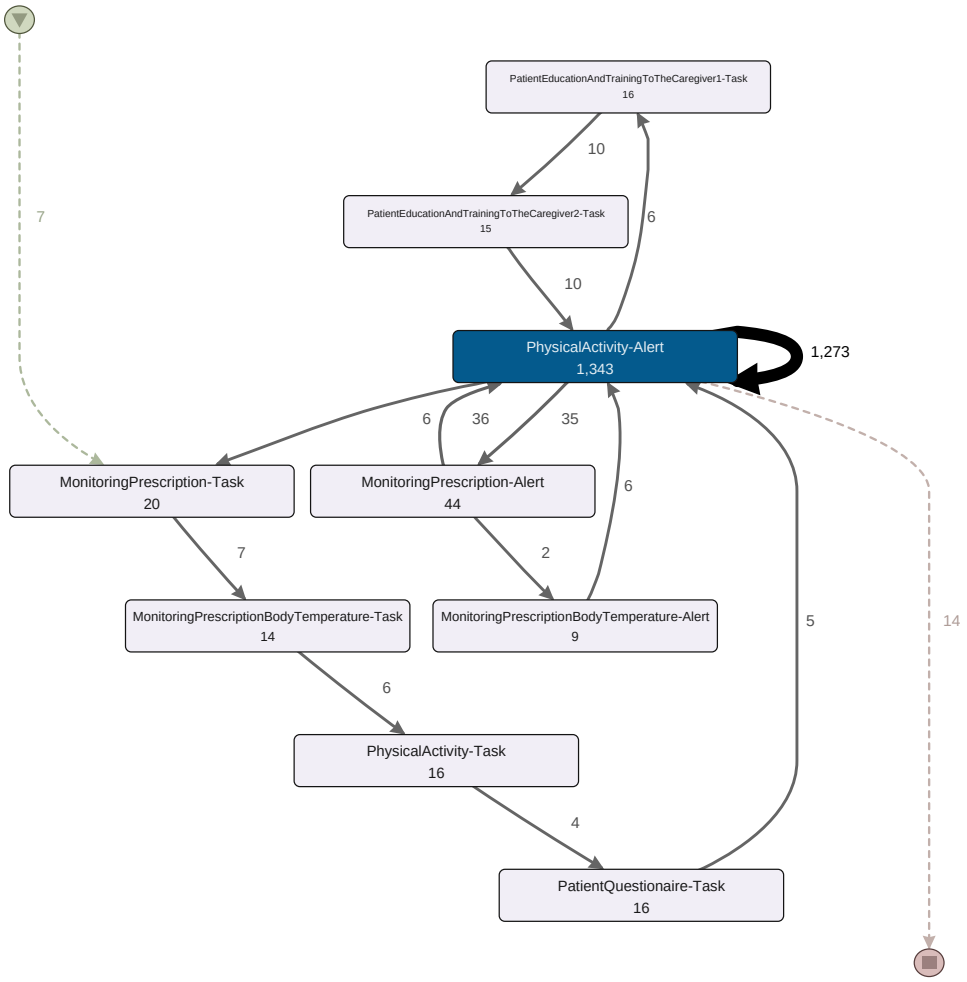


Figure B.49.: Workplan after Hospitalization versions 1-7 at full path abstraction

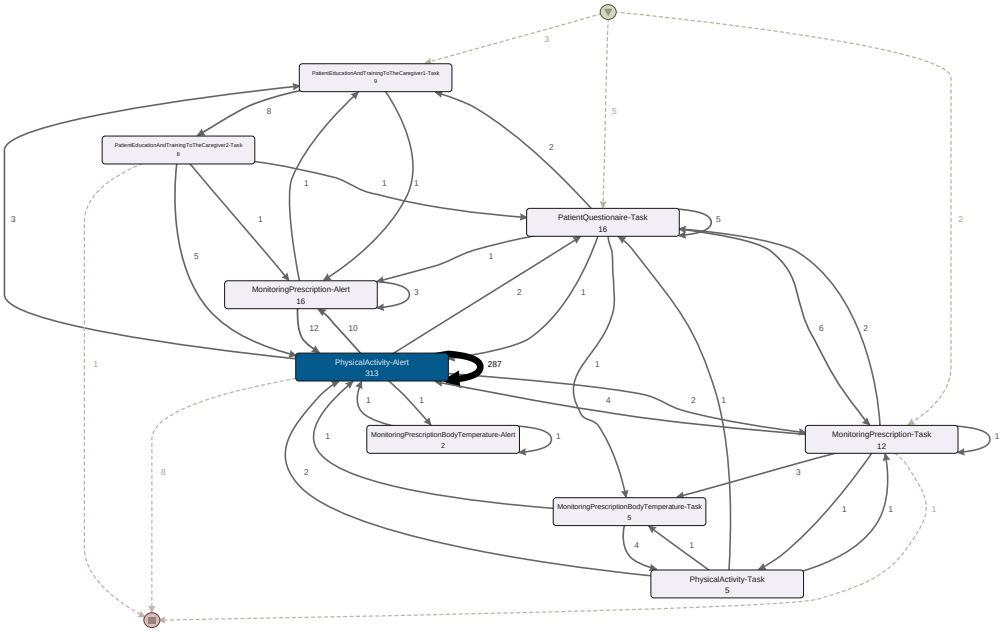


Figure B.50.: Workplan after Hospitalization versions 8+

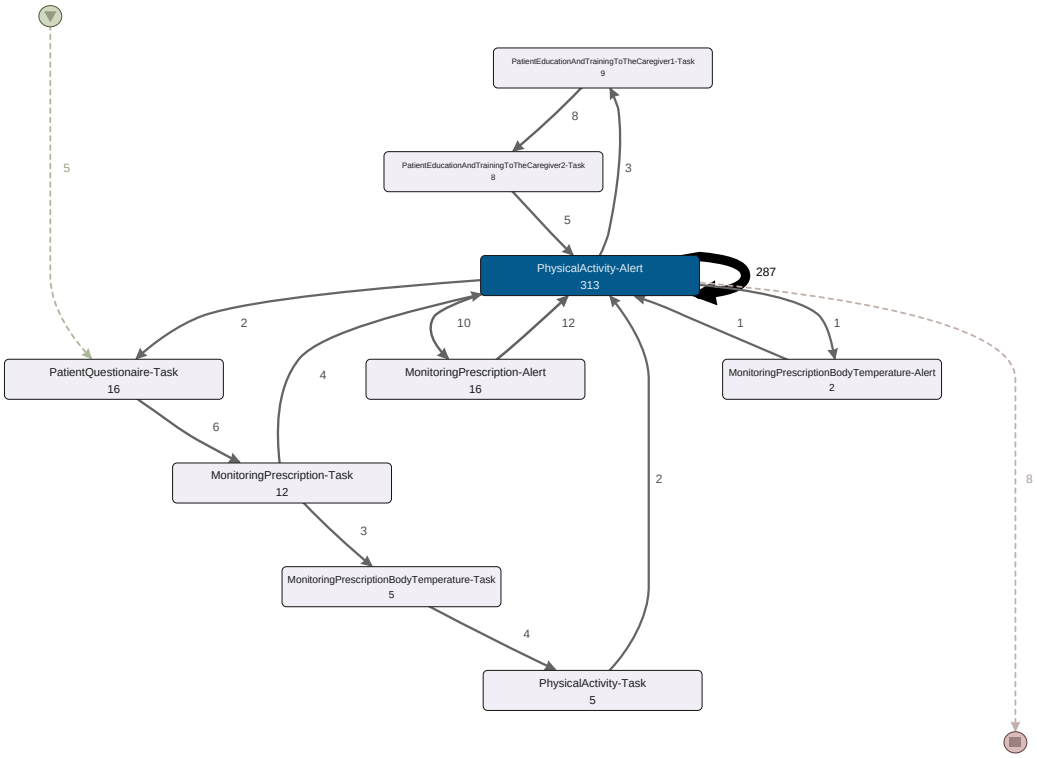


Figure B.51.: Workplan after Hospitalization versions 8+ at full path abstraction

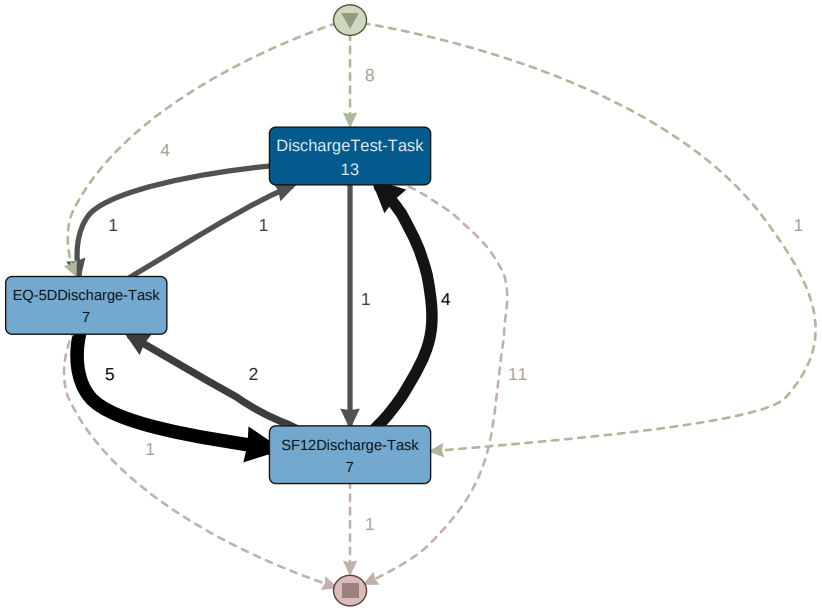


Figure B.52.: Discharge

B.4. Organizational View

B.4.1. Groningen

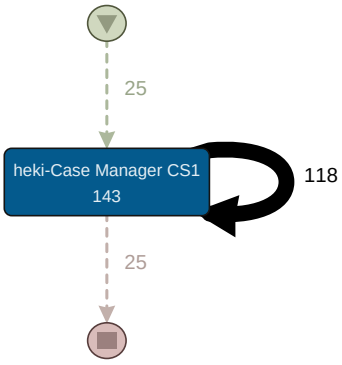


Figure B.53.: Task assignees of Groningen's case study 1

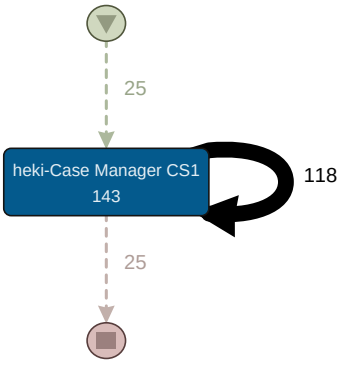


Figure B.54.: Task processors of Groningen's case study 1

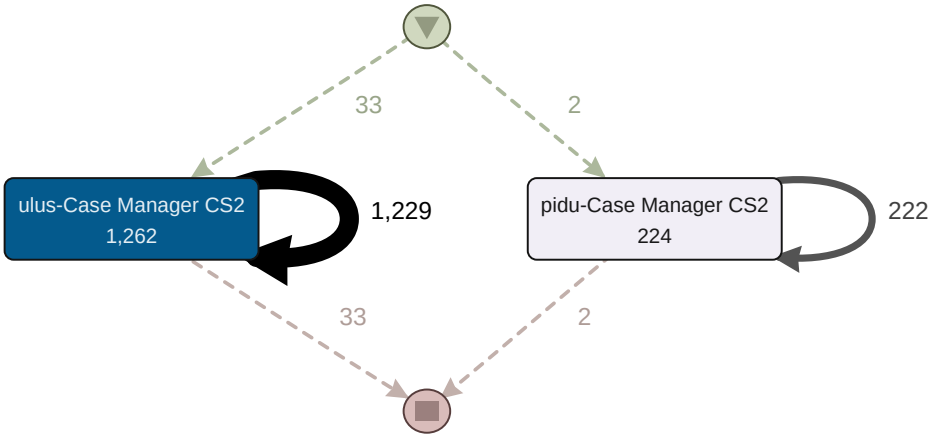


Figure B.55.: Task assignees of Groningen's case study 2

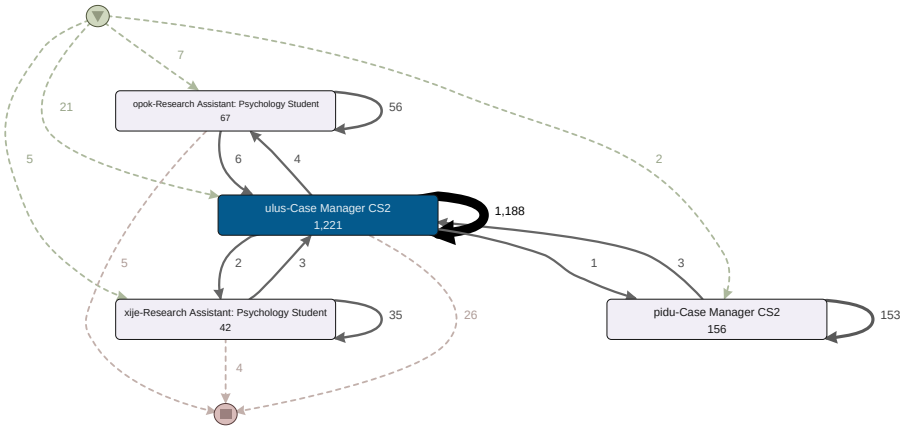


Figure B.56.: Task processors of Groningen's case study 2

B.4.2. Tel-Aviv

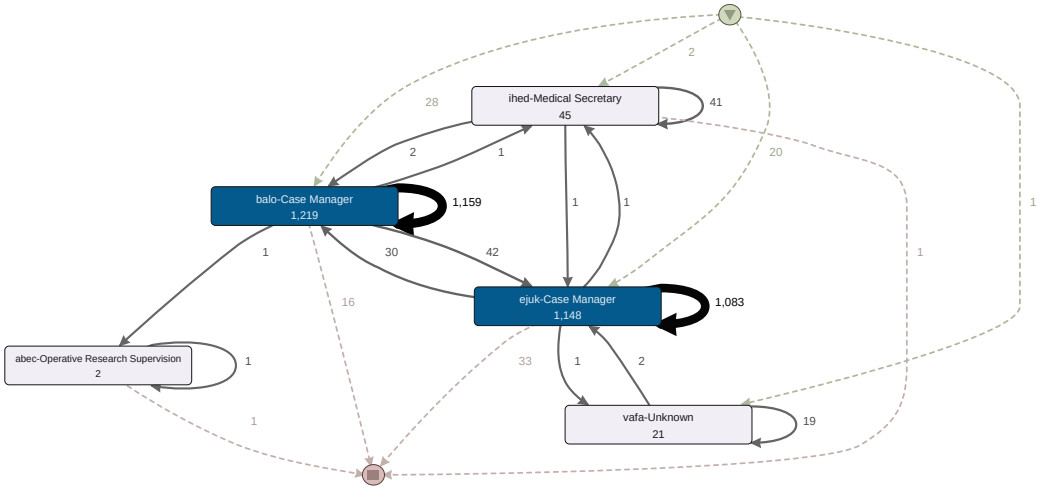


Figure B.57.: Task assignees of Tel-Aviv's case study 1

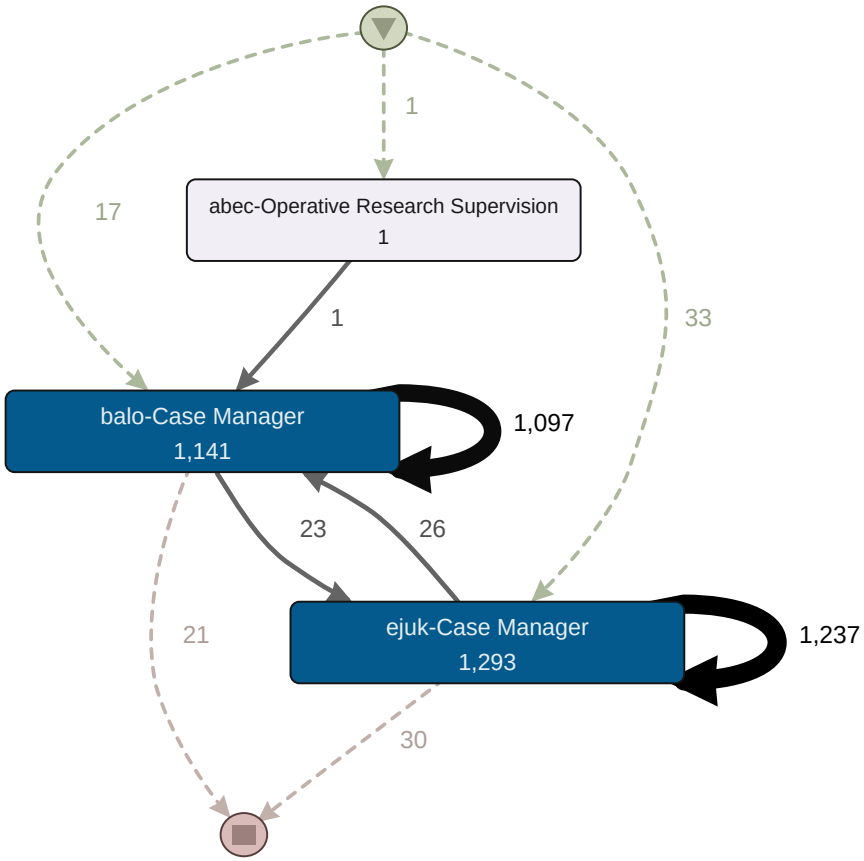


Figure B.58.: Task processors of Tel-Aviv's case study 1

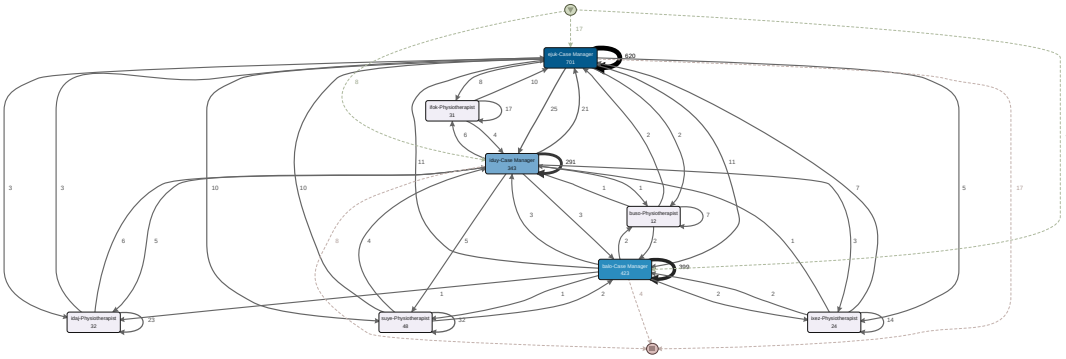


Figure B.59.: Task assignees of Tel-Aviv's case study 2

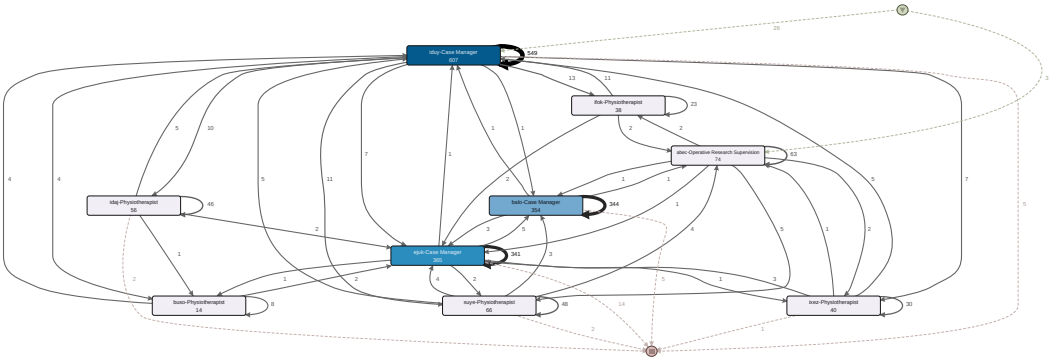


Figure B.60.: Task processors of Tel-Aviv's case study 2

B.4.3. Lleida

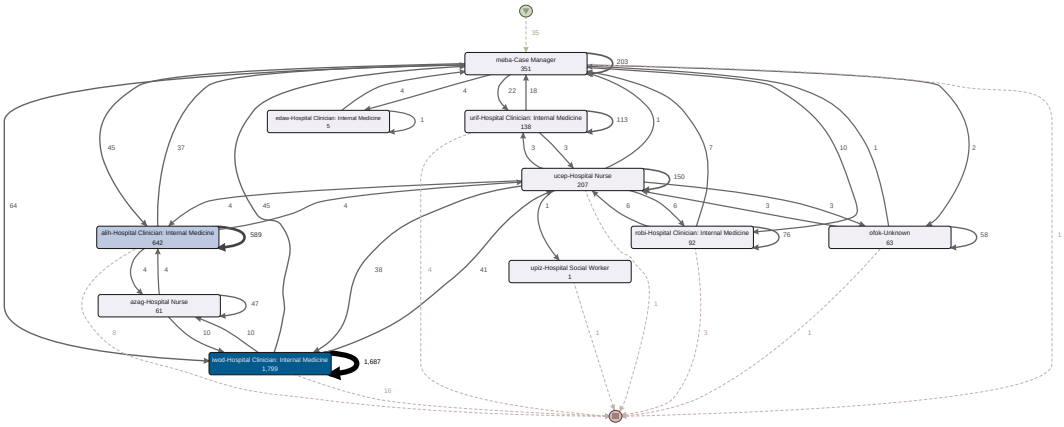


Figure B.61.: Task assignees of Lleida's case study 1

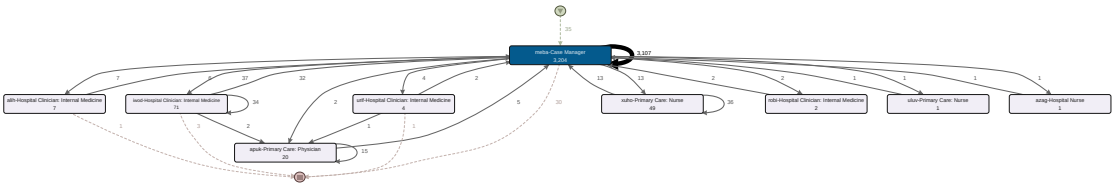


Figure B.62.: Task processors of Lleida's case study 1

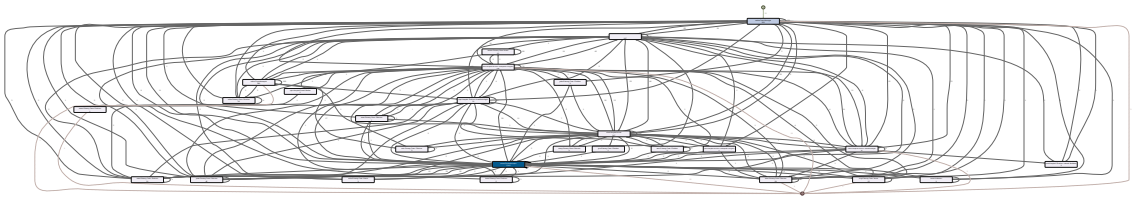


Figure B.63.: Task assignees of Lleida's case study 2

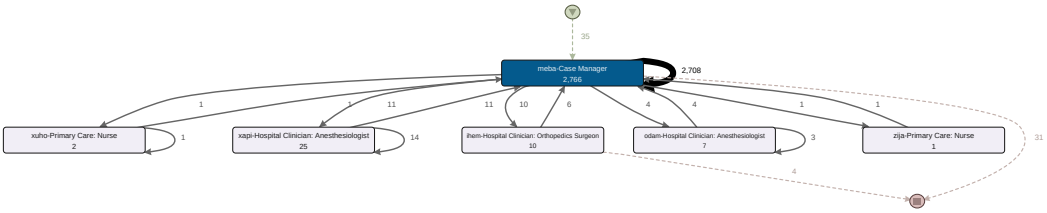


Figure B.64.: Task processors of Lleida's case study 2

C. Tables

C.1. GQFI Table Evaluation

C.1.1. Research Question 1

How is the model-provided flexibility employed during the execution of cases?

	Case Study 1				Case Study 2						
Indicator	***Identification***	***Evaluation***	***Workplan***	***Case Level***	***Identification v1***	***Identification v4***	***Evaluation***	***Workplan v1***	***Workplan v4***	***Workplan v8***	***Case Level***
Number of cases	25	1	20	25	1	34	1	1	16	17	35
Number of events	100	1	49	232	4	170	1	126	1 048	121	1 511
Number of activities	4	1	2	6	4	5	1	2	5	5	6
Number of manually activated tasks	0	0	49	49	0	0	0	126	1 048	121	1 295
Share of manually activated tasks	0%	0%	100%	21%	0%	0%	0%	100%	100%	100%	86%
Number of overall paths	3	0	3	13	3	6	0	4	19	16	14
Mean number of paths per activity	0.75	0	1.5	2.17	0.75	1.2	0	2	3.8	3.2	2.33
Number of bidirectional paths	0	0	0	2	0	1	0	1	5	4	2
Number of process variants	1	1	5	16	1	2	1	1	16	17	35
Maximum share of cases per variant	100%	100%	75%	24%	100%	97%	100%	100%	6%	6%	3%
Median case duration	32 s	0	0	14.2 d	107 s	75 s	0	170.8 d	68.8 d	26 d	51.1 d
Mean case duration	46.7 h	0	33.7 h	20.7 d	107 s	3.6 m	0	170.8 d	77 d	29.4 d	55.4 d
Standard deviation of case duration	6.3 d	0	3 d	21.9 d	0	12.8 m	0	0	28.1 d	21 d	37.6 d

Table C.1.: GQFI table for the first research question for Groningen's case studies

	Case Study 1					Case Study 2				
Indicator	***Identification v7***	***Identification v8***	***Evaluation***	***Workplan***	***Case Level***	***Identification v6***	***Identification v7***	***Evaluation***	***Workplan***	***Case Level***
Number of cases	34	17	47	42	51	12	17	29	29	29
Number of events	153	85	838	1 365	2 976	48	68	680	830	1 967
Number of activities	5	5	18	6	9	4	4	22	6	9
Number of manually activated tasks	0	0	0	1 365	1 365	0	0	0	830	830
Share of manually activated tasks	0%	0%	0%	100%	46%	0%	0%	0%	100%	42%
Number of overall paths	9	4	101	32	40	6	3	114	26	35
Mean number of paths per activity	1.8	0.8	5.6	5.33	4.4	1.5	0.75	5.18	6.5	3.9
Number of bidirectional paths	1	0	14	10	8	1	0	19	9	10
Number of process variants	5	1	34	41	51	2	1	26	28	29
Maximum share of cases per variant	76%	100%	21%	5%	2%	92%	100%	10%	7%	3%
Median case duration	2.8 m	3.7 m	46.4 h	57 d	68.1 d	3.4 m	2.8 m	5.9 d	48.8 d	63.9 d
Mean case duration	44.5 m	2.3 h	7.7 d	48.5 d	62.9 d	10.4 m	3.9 m	14.7 d	55 d	74.4 d
Standard deviation of case duration	3.76 h	5.83 h	17.7 d	34.1 d	41.6 d	13.6 m	3.2 m	24.6 d	45.1 d	49 d

Table C.2.: GQFI table for the first research question for Tel-Aviv's case studies

	Case Study 1					Case Study 2							
Indicator	***Identification***	***Evaluation***	***Workplan***	***Discharge***	***Case Level***	***Identification***	***Evaluation v4***	***Evaluation v9***	***Workplan bef. Hosp.***	***Workplan after Hosp. v7***	***Workplan after Hosp. v8***	***Discharge***	***Case Level***
Number of cases	35	35	34	13	35	35	15	19	32	17	10	13	35
Number of events	175	611	2 541	32	4 297	210	289	287	114	1 484	394	27	3 502
Number of activities	5	22	12	3	10	6	23	21	7	9	9	3	12
Number of manually activated tasks	0	74	2 541	0	2 615	0	16	16	61	1 462	369	0	1 924
Share of manually activated tasks	0%	12%	100%	0%	61%	0%	6%	6%	54%	99%	94%	0%	55%
Number of overall paths	8	185	57	5	48	15	156	125	18	43	34	6	73
Mean number of paths per activity	1.6	8.41	4.75	1.67	4.8	2.5	6.78	5.95	2.57	4.77	3.78	2	6.08
Number of bidirectional paths	1	57	15	2	16	5	37	30	5	10	7	3	26
Number of process variants	3	35	34	5	35	7	15	19	23	17	10	5	35
Maximum share of cases per variant	77%	3%	3%	31%	3%	57%	7%	5%	19%	6%	10%	46%	3%
Median case duration	5.4 m	70 h	46.5 d	3.1 d	49.3 d	4.7 m	65.9 d	23.9 d	16.6 d	93.1 d	29.8 d	22.9 h	98.7 d
Mean case duration	12.5 m	13.3 d	55.4 d	4.2 d	64.4 d	9.7 h	75.9 d	24.4 d	41.1 d	89.1 d	32.2 d	11.4 d	98.7 d
Standard deviation of case duration	19.7 m	29.8 d	34.4 d	5.6 d	45.2 d	55.6 h	53.4 d	25.9 d	60.9 d	27.1 d	15.9 d	22.2 d	65.9 d

Table C.3.: GQFI table for the first research question for Lleida's case studies

C.1.2. Research Question 2

How do communication and notification features affect case executions?

	Groningen		Tel-Aviv		Lleida	
Indicator	**CS1**	**CS2**	**CS1**	**CS2**	**CS1**	**CS2**
Number of notes update events	0	0	2	2	31	29
Number of alert events	28	1 192	1 181	649	2 353	1 764
Number of message events (with patients)	57	6	342	201	641	523
Number of message events (with professionals)	0	0	106	97	230	106
Share of feature-related events	37%	79%	55%	48%	76%	69%
Mean number of paths per alert activity	3	2	6	6	7	7
Mean number of paths per message activity	2	3	6.5	6	7.5	8
Mean number of paths per other activity	2	2.25	3.2	3	4.4	5.14

Table C.4.: GQFI table for the second research question

C.1.3. Research Question 3

How are collaboration and organization features reflected in case executions?

	Groningen		Tel-Aviv		Lleida	
Indicator	**CS1**	**CS2**	**CS1**	**CS2**	**CS1**	**CS2**
Number of tasks not done by their assignee	0	110	1 003	551	634	715
Share of tasks not done by their assignee	0%	37%	80%	57%	63%	68%
Number of task assignees	1	2	5	8	10	29
Number of task processors	1	4	3	9	9	6
Number of roles	1	2	2	3	5	4
Maximum share of work for one person	100%	83%	53%	37%	95%	98%
Mean number of collaborators per person	0	1.5	0.67	2.67	1.78	1.67
Maximum number of collaborators per person	1	3	1	6	8	5

Table C.5.: GQFI table for the third research question

C.2. Disco Mappings

Disco Type	CSV Column
Case ID	CaseID
Activity	Activity
	ActivtityType
Timestamp	Timestamp
Other attributes	_DebugID

Table C.6.: Mappings for the system view

Disco Type	CSV Column
Case ID	CaseID
Activity	Activity
	ActivtityType
Timestamp	Timestamp
Other attributes	ClusterGroup
	_DebugID

Table C.7.: Mappings for the case view

Disco Type	CSV Column
Case ID	CaseID
Activity	Activity
	ActivtityType
Timestamp	Timestamp
Other attributes	ClusterGroup
	_DebugID
Unmapped	*AssigneeID*
	AssigneeRole
	ProcessorID
	ProcessorRole

Table C.8.: Mappings for the stage view

Disco Type	CSV Column
Case ID	CaseID
Activity	AssigneeID
	AssigneeRole
Timestamp	Timestamp
Other attributes	ClusterGroup
	_DebugID
Unmapped	*Activity*
	ActivtityType
	ProcessorID
	ProcessorRole

Table C.9.: Mappings for the organizational view (task assignee)

Disco Type	CSV Column
Case ID	CaseID
Activity	ProcessorID
	ProcessorRole
Timestamp	Timestamp
Other attributes	ClusterGroup
	_DebugID
Unmapped	*Activity*
	ActivtityType
	AssigneeID
	AssigneeRole

Table C.10.: Mappings for the organizational view (task processor)

C.3. Cluster Group Mappings

C.3.1. Case View

Cluster	Case Model Version
GCS1v1	GCS1_Groningen_1
GCS1v2	GCS1_Groningen_2
GCS2v1	GCS2_Groningen_1
	GCS2_Groningen_2
GCS2v3	GCS2_Groningen_3
	GCS2_Groningen_4
	GCS2_Groningen_6
	GCS2_Groningen_7
	GCS2_Groningen_8

Table C.11.: Cluster mappings for Groningen's case view

Cluster	Case Model Version
ICS1v7	ICS1_Assuta_1
	ICS1_Assuta_5
	ICS1_Assuta_6
	ICS1_Assuta_7
	ICS1_Assuta_8
	ICS1_Assuta_9
ICS2v6	ICS2_Assuta_1
	ICS2_Assuta_2
	ICS2_Assuta_3
	ICS2_Assuta_4
	ICS2_Assuta_5
	ICS2_Assuta_6
	ICS2_Assuta_7
	ICS2_Assuta_8

Table C.12.: Cluster mappings for Tel-Aviv's case view

Cluster	Case Model Version
LCS1v4	LCS1_Leida_2
	LCS1_Leida_3
	LCS1_Leida_4
	LCS1_Leida_5
	LCS1_Leida_6
	LCS1_Leida_7
	LCS1_Leida_9
LCS2v4	LCS2_Leida_4
	LCS2_Leida_5
	LCS2_Leida_6
	LCS2_Leida_7
	LCS2_Leida_8
	LCS2_Leida_9

Table C.13.: Cluster mappings for Lleida's case view

C.3.2. Stage View

Cluster	Case Model Version
GCS1v1	GCS1_Groningen_1
	GCS1_Groningen_2
GCS2v1	GCS2_Groningen_1
	GCS2_Groningen_2
	GCS2_Groningen_3
GCS2v4	GCS2_Groningen_4
	GCS2_Groningen_6
	GCS2_Groningen_7
	GCS2_Groningen_8

(a) Case Identification

Cluster	Case Model Version
GCS1v1	GCS1_Groningen_1
	GCS1_Groningen_2
GCS2v1	GCS2_Groningen_1
	GCS2_Groningen_2
	GCS2_Groningen_3
	GCS2_Groningen_4
	GCS2_Groningen_6
	GCS2_Groningen_7
	GCS2_Groningen_8

(b) Case Evaluation

Cluster	Case Model Version
GCS1v1	GCS1_Groningen_1
GCS1v2	GCS1_Groningen_2
GCS2v1	GCS2_Groningen_1
	GCS2_Groningen_2
GCS2v3	GCS2_Groningen_3
	GCS2_Groningen_4
	GCS2_Groningen_6
	GCS2_Groningen_7
GCS2v8	GCS2_Groningen_8

(c) Workplan & Discharge

Table C.14.: Cluster mappings for Groningen's stage view

Cluster	Case Model Version
ICS1v7	ICS1_TelAviv_1
	ICS1_TelAviv_5
	ICS1_TelAviv_6
	ICS1_TelAviv_7
ICS1v8	ICS1_TelAviv_8
	ICS1_TelAviv_9
ICS2v6	ICS2_TelAviv_1
	ICS2_TelAviv_2
	ICS2_TelAviv_3
	ICS2_TelAviv_4
	ICS2_TelAviv_5
	ICS2_TelAviv_6
ICS2v7	ICS2_TelAviv_7
	ICS2_TelAviv_8

(a) Case Identification

Cluster	Case Model Version
ICS1v7	ICS1_TelAviv_1
	ICS1_TelAviv_5
	ICS1_TelAviv_6
	ICS1_TelAviv_7
	ICS1_TelAviv_8
	ICS1_TelAviv_9
ICS2v6	ICS2_TelAviv_1
	ICS2_TelAviv_2
	ICS2_TelAviv_3
	ICS2_TelAviv_4
	ICS2_TelAviv_5
	ICS2_TelAviv_6
	ICS2_TelAviv_7
	ICS2_TelAviv_8

(b) Other stages

Table C.15.: Cluster mappings for Tel-Aviv's stage view

Cluster	Case Model Version
LCS1v9	LCS1_Lleida_2
	LCS1_Lleida_3
	LCS1_Lleida_4
	LCS1_Lleida_5
	LCS1_Lleida_6
	LCS1_Lleida_7
	LCS1_Lleida_9

Table C.16.: Cluster mappings for Lleida's stage view of the first study

Cluster	Case Model Version
LCS2v4	LCS2_Lleida_4
	LCS2_Lleida_5
	LCS2_Lleida_6
LCS2v9	LCS2_Lleida_7
	LCS2_Lleida_8
	LCS2_Lleida_9

(a) Case Evaluation

Cluster	Case Model Version
LCS2v4	LCS2_Lleida_4
	LCS2_Lleida_5
	LCS2_Lleida_6
	LCS2_Lleida_7
LCS2v8	LCS2_Lleida_8
LCS2v9	LCS2_Lleida_9

(b) Workplan after Hospitalization

Cluster	Case Model Version
LCS2v9	LCS2_Lleida_4
	LCS2_Lleida_5
	LCS2_Lleida_6
	LCS2_Lleida_7
	LCS2_Lleida_8
	LCS2_Lleida_9

(c) Other stages

Table C.17.: Cluster mappings for Lleida's stage view of the second study

D. Kibana Dashboards

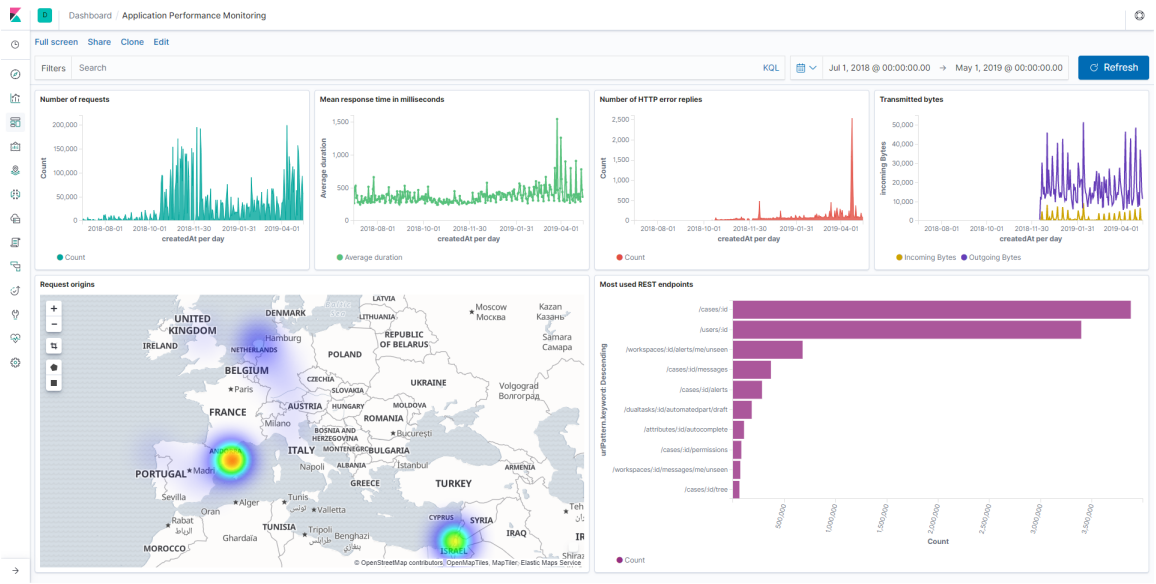


Figure D.1.: Application Performance Monitoring Dashboard

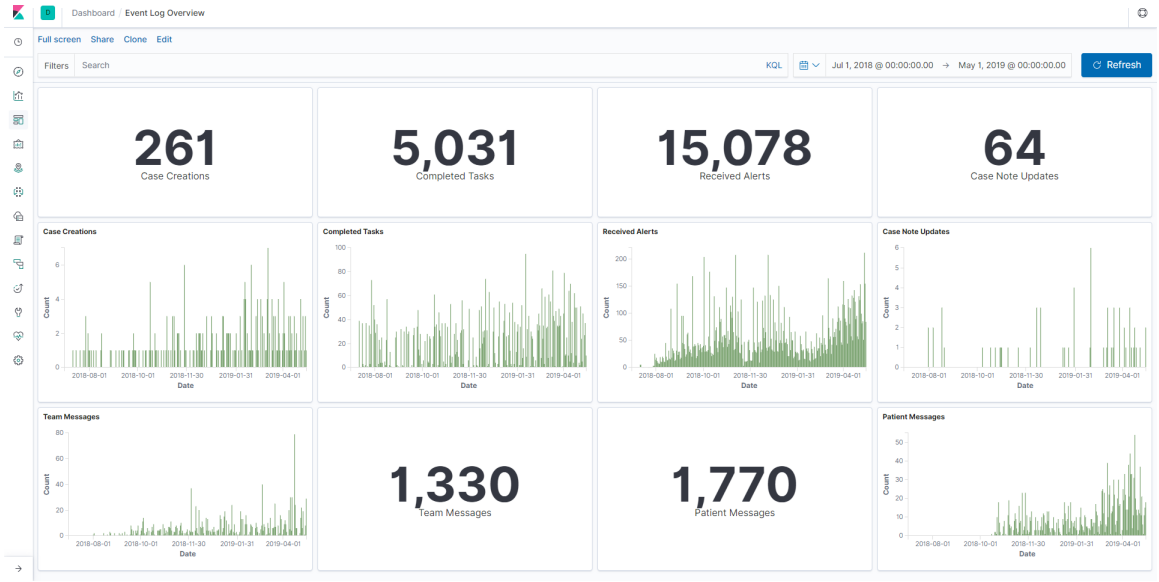


Figure D.2.: Event Log Overview Dashboard

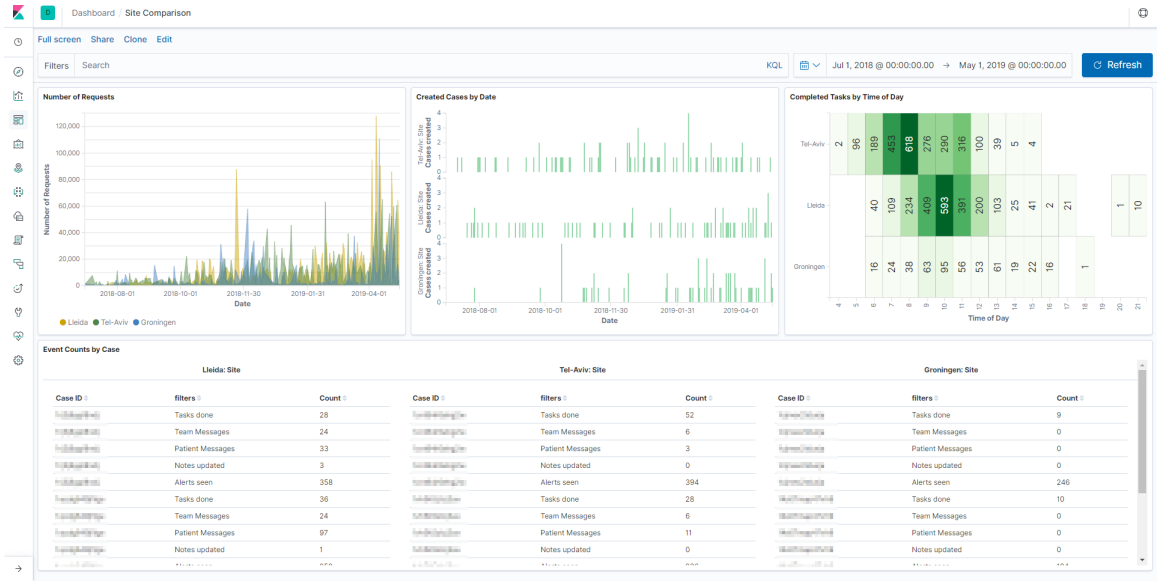


Figure D.3.: Site Comparison Dashboard

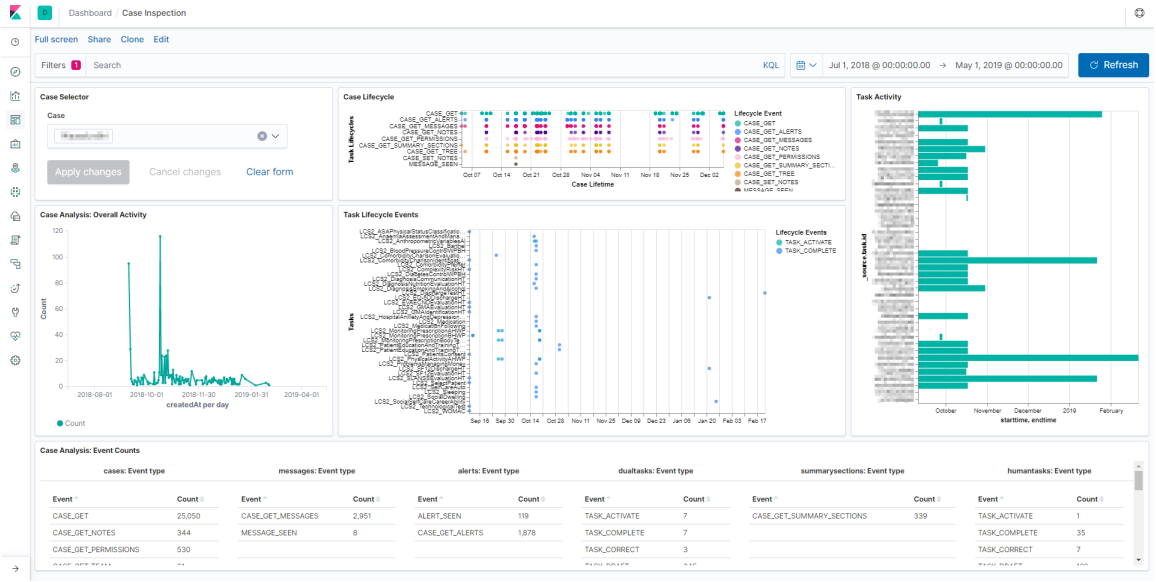


Figure D.4.: Case Inspection Dashboard

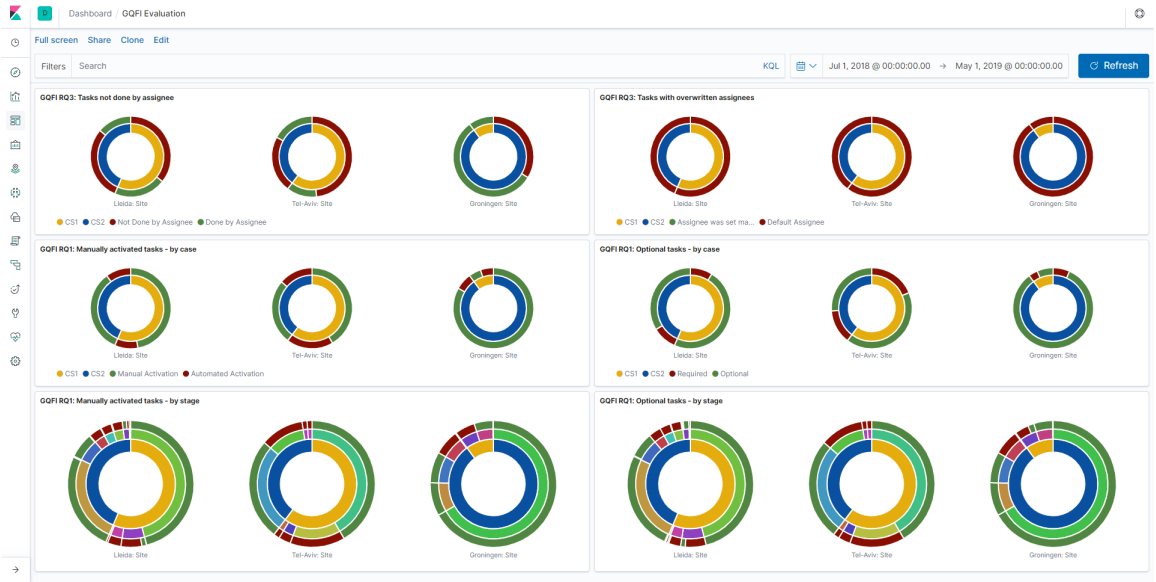


Figure D.5.: GQFI Evaluation Dashboard

E. Listings

```yaml
version: '3'
services:
  sacm.analytics:
    image: connecare/sacm.analytics:develop
    container_name: sacm.analytics
    ports:
      # nginx reverse authentication proxy for kibana web interface
      # ! should be the only exposed port in production use !
      - "5601:5601"
    environment:
      # connection details for API logs database
      - SACM_ANALYTICS_JDBC_LOGS=jdbc:mysql://<host>:<port>/<db-name>
      - SACM_ANALYTICS_JDBC_LOGS_USER=<username>
      - SACM_ANALYTICS_JDBC_LOGS_PASSWORD=<password>
      # connection details for SACM database
      - SACM_ANALYTICS_JDBC_DATA=jdbc:mysql://<host>:<port>/<db-name>
      - SACM_ANALYTICS_JDBC_DATA_USER=<username>
      - SACM_ANALYTICS_JDBC_DATA_PASSWORD=<password>
      # connection details for PatientMessages database
      - SACM_ANALYTICS_JDBC_MESSAGES=jdbc:mysql://<host>:<port>/<db-name>
      - SACM_ANALYTICS_JDBC_MESSAGES_USER=<username>
      - SACM_ANALYTICS_JDBC_MESSAGES_PASSWORD=<password>
      # username/password for HTTP basic authentication provided by nginx
      - SACM_ANALYTICS_KIBANA_USER=<username>
      - SACM_ANALYTICS_KIBANA_PASSWORD=<password>
      # base URL if Kibana can not be reached at the domain root
      - SACM_ANALYTICS_KIBANA_BASE_PATH=/sacm/analytics
      # prevent Logstash from starting for read-only deployments
      - LOGSTASH_START=1
      # prevent Logstash from exporting the event log during initial import
      - LOGSTASH_EXPORT=1
    volumes:
      # Data directories, to prevent data loss when re-deploying
      - home/data/elasticsearch:/var/lib/elasticsearch
      - home/data/logstash:/opt/logstash/data
      - home/data/kibana:/opt/kibana/data
      # Logging directories
      - home/logs/elasticsearch:/var/log/elasticsearch
      - home/logs/logstash:/var/log/logstash
      - home/logs/kibana:/var/log/kibana
      - home/logs/nginx:/var/log/nginx
    networks:
      - "xcarebackend"
networks:
  xcarebackend:
    external:
      name: connecare-network
```

Source Code E.1.: Example of a docker-compose.yaml file

Glossary

ACM adaptive case management. 7, 8, 11, 12, 32, 34, 43, 44, 125

ACQ Asthma Control Questionnaire. 15

API Application Programming Interface. 4, 11, 13, 17, 18, 43, 44, 49–51, 62, 64–67, 69, 70, 72, 73, 76, 90, 91

ASA American Society of Anesthesiologists. 14, 17

BPMN Business Process Model and Notation. 8

CCP complex chronic patient. 3, 9–11, 15, 16, 61, 99

CCQ Clinical COPD Questionnaire. 15

CMMN Case Management Model and Notation. 8, 13, 14, 95, 98

CONNECARE Personalised Connected Care for Complex Chronic Patients. 3, 4, 9–15, 17–19, 43, 44, 46, 50, 60, 61, 63, 66, 67, 89, 90, 96, 98, 99, 109, 117

COPD chronic obstructive pulmonary disease. 14–16, 61, 97

CP Clinical Pathway. 3–6, 8–11, 13, 17–19, 23, 26–30, 32, 35, 53–55, 57, 125, 127

CSS Cascaded Style Sheet. 20

CSV comma separated value. 65, 70, 72, 74, 76, 79, 80, 126

EQ5D EuroQol five dimension scale. 15, 16

GFI Groningen Frailty Indicator. 16

GMA Adjusted Morbidity Groups. 16

GQFI Goal-Question-Feature-Indicator. 56, 57, 61–63, 75, 76, 96, 100, 103, 110, 114–116, 119, 122, 125, 126, 205

HADS Hospital Anxiety and Depression Scale. 16

HIS Hospital Information System. 7, 10, 11, 26, 27, 32, 37–39, 44, 56, 57, 78

HRRS human readable random string. 70

HTML Hypertext Markup Language. 20

HTTP Hypertext Transfer Protocol. 44, 45, 69, 90

IEEE Institute of Electrical and Electronics Engineers. 72

JDBC Java Database Connector. 64, 66, 67, 90

JSON JavaScript Object Notation. 11, 13, 43, 44, 62

KiP knowledge-intensive process. 4–6, 8, 13, 17, 19, 23–29, 32–35, 47, 53, 54, 79, 85–87, 117, 125

KPI key performance indicator. 56, 57

MNASF Mini Nutritional Assessment Short Form. 16

MUST Malnutrition Universal Screening Tool. 16

MXML mining eXtensible markup language. 72, 74

NRS Nutritional Risk Screening. 16

PAIS process-aware information system. 7

PCM production case management. 7

PDM process diagnostics method. 25

PM Process Mining. 4, 7, 18, 21–32, 34, 36–39, 43–47, 49, 50, 52–62, 64–70, 72, 74–76, 78–80, 82, 84, 87, 95, 125, 126

REST Representational State Transfer. 4, 17, 31, 32, 43–46, 50, 62, 66, 90, 125

SACM Smart Adaptive Case Management. 4, 11–13, 17–20, 44, 50, 61, 62, 64–67, 70–72, 86, 88, 99, 102, 110, 111, 114, 115, 117–120, 123, 125

SF12 Short Form Health Survey with 12 items. 15, 16

S-LANSS self-report Leeds Assessment of Neuropathic Symptoms and Signs. 17

SMS Self Management System. 11, 64, 66, 114

SOA service-oriented architecture. 30, 31

SOAP Simple Object Access Protocol. 31

UIM User Identity Management. 12, 69

UML Unified Modeling Language. 8

WOMAC Western Ontario and McMaster Universities Osteoarthritis Index. 17

XES eXtensible event stream. 72, 74, 126

XML Extensible Markup Language. 13, 31, 72, 80

Bibliography

[1] Giovanni Acampora, Autilia Vitiello, Bruno Di Stefano, Wil van der Aalst, Christian Gunther, and Eric Verbeek. IEEE 1849: The XES Standard: The Second IEEE Standard Sponsored by IEEE Computational Intelligence Society [Society Briefs]. *IEEE Computational Intelligence Magazine*, 12(2):4–8, may 2017. doi: 10.1109/mci.2017.2670420.

[2] Ilana Ackerman. Western Ontario and McMaster Universities Osteoarthritis Index (WOMAC). *The Australian journal of physiotherapy*, 55:213, 2009. ISSN 0004-9514.

[3] Camilo Alvarez, Eric Rojas, Michael Arias, Jorge Munoz-Gama, Marcos Sepúlveda, Valeria Herskovic, and Daniel Capurro. Discovering role interaction models in the Emergency Room using Process Mining. *Journal of Biomedical Informatics*, 78:60–77, feb 2018. doi: 10.1016/j.jbi.2017.12.015.

[4] Anael Barberan-Garcia, Elena Gimeno-Santos, Isabel Blanco, Isaac Cano, Graciela Martínez-Pallí, Felip Burgos, Felip Miralles, Miquel Coca, Serafín Murillo, María Sanz, Alexander Steblin, Marta Ubré, Jaume Benavent, Josep Vidal, Marta Sitges, and Josep Roca. Protocol for regional implementation of collaborative self-management services to promote physical activity. *BMC Health Services Research*, 18(1), jul 2018. doi: 10.1186/s12913-018-3363-8.

[5] Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. The Goal Question Metric Approach. In *Encyclopedia of Software Engineering*. Wiley, 1994.

[6] Edgar Batista and Agusti Solanas. Process Mining in Healthcare: A Systematic Review. In *2018 9th International Conference on Information, Intelligence, Systems and Applications (IISA)*. IEEE, jul 2018. doi: 10.1109/iisa.2018.8633608.

[7] Michael I. Bennett, Blair H. Smith, Nicola Torrance, and Jean Potter. The S-LANSS score for identifying pain of predominantly neuropathic origin: Validation for use in clinical and postal research. *The Journal of Pain*, 6(3):149–158, mar 2005. doi: 10.1016/j.jpain.2004.11.007.

[8] P. Berkhin. A Survey of Clustering Data Mining Techniques. In *Grouping Multidimensional Data*, pages 25–71. Springer-Verlag, 2006. doi: 10.1007/3-540-28349-8_2.

[9] Roberto Bertollini, Sofia Ribeiro, Kristina Mauer-Stender, and Gauden Galea. Tobacco control in Europe: a policy review. *European Respiratory Review*, 25(140):151–157, may 2016. doi: 10.1183/16000617.0021-2016.

[10] Peter Bodesinsky, Bilal Alsallakh, Theresia Gschwandtner, and Silvia Miksch. Visual process mining: Event data exploration and analysis. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*. IEEE, oct 2014. doi: 10.1109/vast.2014.7042504.

[11] R. P. Jagadeesh Chandra Bose, Ronny S. Mans, and Wil M. P. van der Aalst. Wanna Improve Process Mining Results? In *2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*. IEEE, apr 2013. doi: 10.1109/cidm.2013.6597227.

[12] Don Box, David Ehnebuske, Gopal Kakivaya, Andrew Layman, Noah Mendelsohn, Henrik Frystyk Nielsen, Satish Thatte, and Dave Winer. Simple object access protocol (SOAP) 1.1, 2000.

[13] Melike Bozkaya, Joost Gabriels, and Jan Martijn van der Werf. Process Diagnostics: A Method Based on Process Mining. In *2009 International Conference on Information, Process, and Knowledge Management*. IEEE, feb 2009. doi: 10.1109/eknow.2009.29.

[14] Humberto S. Garcia Caballero, Alberto Corvo, Prabhakar M. Dixit, and Michel A. Westenberg. Visual analytics for evaluating clinical pathways. In *2017 IEEE Workshop on Visual Analytics in Healthcare (VAHC)*. IEEE, oct 2017. doi: 10.1109/vahc.2017.8387499.

[15] Isaac Cano, Albert Alonso, Carme Hernandez, Felip Burgos, Anael Barberan-Garcia, Jim Roldan, and Josep Roca. An adaptive case management system to support integrated care services: Lessons learned from the NEXES project. *Journal of Biomedical Informatics*, 55:11–22, jun 2015. doi: 10.1016/j.jbi.2015.02.011.

[16] Isaac Cano, Ivan Dueñas-Espín, Carme Hernandez, Jordi de Batlle, Jaume Benavent, Juan Carlos Contel, Erik Baltaxe, Joan Escarrabill, Juan Manuel Fernández, Judith Garcia-Aymerich, Miquel Àngel Mas, Felip Miralles, Montserrat Moharra, Jordi Piera, Tomas Salas, Sebastià Santaeugènia, Nestor Soler, Gerard Torres, Eloisa Vargiu, Emili Vela, and Josep Roca. Protocol for regional implementation of community-based collaborative management of complex chronic patients. *npj Primary Care Respiratory Medicine*, 27(1), jul 2017. doi: 10.1038/s41533-017-0043-9.

[17] B. R. Celli and P. J. Barnes. Exacerbations of chronic obstructive pulmonary disease. *European Respiratory Journal*, 29(6):1224–1238, mar 2007. doi: 10.1183/09031936.00109906.

[18] Mary Charlson, Ted P. Szatrowski, Janey Peterson, and Jeffrey Gold. Validation of a combined comorbidity index. *Journal of Clinical Epidemiology*, 47(11):1245–1251, nov 1994. doi: 10.1016/0895-4356(94)90129-5.

[19] Mary E. Charlson, Peter Pompei, Kathy L. Ales, and C. Ronald MacKenzie. A new method of classifying prognostic comorbidity in longitudinal studies: Development

and validation. *Journal of Chronic Diseases*, 40(5):373–383, jan 1987. doi: 10.1016/0021-9681(87)90171-8.

[20] Minsu Cho, Minseok Song, and Sooyoung Yoo. A Systematic Methodology for Outpatient Process Analysis Based on Process Mining. In *Lecture Notes in Business Information Processing*, pages 31–42. Springer International Publishing, 2014. doi: 10.1007/978-3-319-08222-6_3.

[21] Jan Claes and Geert Poels. Process Mining and the ProM Framework: An Exploratory Survey. In *BPM 2012: Business Process Management Workshops*, volume 132, pages 187–198, September 2012. doi: 10.1007/978-3-642-36285-9-19.

[22] Connecare Project. The Facts — Connecare, 2019. URL http://www.connecare.eu/about-connecare/facts/.

[23] Jonathan E. Cook and Alexander L. Wolf. Discovering models of software processes from event-based data. *ACM Transactions on Software Engineering and Methodology*, 7 (3):215–249, jul 1998. doi: 10.1145/287000.287001.

[24] Álvaro José da Silva Rebuge. Business Process Analysis in Healthcare Environments. Master's thesis, Universidade Técnica de Lisboa, 2012.

[25] Dusanka Dakic, Darko Stefanovic, Ilija Cosic, Teodora Lolic, and Milovan Medojevic. Business Process Mining Application: A Literature Review. In *Proceedings of the 29th International DAAAM Symposium 2018*, pages 866–875. DAAAM International Vienna, 2018. doi: 10.2507/29th.daaam.proceedings.125.

[26] Leentja de Bleser, Roeland Depreitere, Katrijn de Waele, Kris Vanhaecht, Joan Vlayen, and Walter Sermeus. Defining pathways. *Journal of Nursing Management*, 14(7):553–563, oct 2006. doi: 10.1111/j.1365-2934.2006.00702.x.

[27] Pavlos Delias, Michael Doumpos, Evangelos Grigoroudis, Panagiotis Manolitzas, and Nikolaos Matsatsinis. Supporting healthcare management decisions via robust clustering of event logs. *Knowledge-Based Systems*, 84:203–213, aug 2015. doi: 10.1016/j.knosys.2015.04.012.

[28] Claudio Di Ciccio, Andrea Marrella, and Alessandro Russo. Knowledge-intensive Processes: An Overview of Contemporary Approaches. In *CEUR Workshop Proceedings*, volume 861, pages 33–47, 06 2012.

[29] Claudio Di Ciccio, Andrea Marrella, and Alessandro Russo. Knowledge-Intensive Processes: Characteristics, Requirements and Analysis of Contemporary Approaches. *Journal on Data Semantics*, 4(1):29–57, apr 2014. doi: 10.1007/s13740-014-0038-4.

[30] P. M. Dixit, H. S. Garcia Caballero, A. Corvò, B. F. A. Hompes, J. C. A. M. Buijs, and W. M. P. van der Aalst. Enabling Interactive Process Analysis with Process Mining and Visual Analytics. In *Proceedings of the 10th International Joint Conference on Biomedical Engineering Systems and Technologies*. SCITEPRESS - Science and Technology Publications, 2017. doi: 10.5220/0006272605730584.

[31] Elasticsearch B.V. Elasticsearch Reference [6.5], February 2019. URL https://www.elastic.co/guide/en/elasticsearch/reference/6.5/index.html.

[32] Elasticsearch B.V. Logstash Reference [6.5], February 2019. URL https://www.elastic.co/guide/en/logstash/6.5/index.html.

[33] Elasticsearch B.V. Kibana User Guide [6.5], February 2019. URL https://www.elastic.co/guide/en/kibana/6.5/index.html.

[34] M. Elia. The 'MUST' report. Nutritional screening for adults: a multidisciplinary responsibility. Development and use of the 'Malnutrition Universal Screening Tool' (MUST) for adults., 2003.

[35] Tugba Gurgen Erdogan and Ayca Tarhan. Systematic Mapping of Process Mining Studies in Healthcare. *IEEE Access*, 6:24543–24567, 2018. doi: 10.1109/access.2018.2831244.

[36] Tugba Gurgen Erdogan and Ayca Tarhan. A Goal-Driven Evaluation Method Based on Process Mining for Healthcare Processes. *Applied Sciences*, 8(6):894, may 2018. doi: 10.3390/app8060894.

[37] Eren Esgin and Pınar Karagoz. Sequence Alignment Adaptation for Process Diagnostics and Delta Analysis. In *Lecture Notes in Computer Science*, pages 191–201. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-40846-5_20.

[38] Eren Esgin and Pinar Senkul. Delta Analysis: A Hybrid Quantitative Approach for Measuring Discrepancies between Business Process Models. In *Lecture Notes in Computer Science*, pages 296–304. Springer Berlin Heidelberg, 2011. doi: 10.1007/978-3-642-21219-2_38.

[39] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231. AAAI Press, 1996.

[40] European Comission. Periodic Reporting for period 1 - CONNECARE (Personalised Connected Care for Complex Chronic Patients), February 2018. URL https://cordis.europa.eu/project/rcn/202638/reporting/en.

[41] EuroQolGroup. EuroQol - a new facility for the measurement of health-related quality of life. *Health Policy*, 16(3):199–208, dec 1990. doi: 10.1016/0168-8510(90)90421-9.

[42] Nik Farid, Marc De Kamps, and Owen Johnson. Process Mining in Frail Elderly Care: A Literature Review. In *Proceedings of the 12th International Joint Conference on Biomedical Engineering Systems and Technologies*. SCITEPRESS - Science and Technology Publications, 2019. doi: 10.5220/0007392903320339.

[43] Hongying Fei. Discovering patient care process models from event logs. In *Conférence Internationale de Modélisation et Simulation (MOSIM 10), Special track on Healthcare Information systems and decision making*, 2010.

[44] Carlos Fernandez-Llatas, Antonio Martinez-Millana, Alvaro Martinez-Romero, Jose Miguel Benedi, and Vicente Traver. Diabetes care related process modelling using Process Mining techniques. Lessons learned in the application of Interactive Pattern Recognition: coping with the Spaghetti Effect. In *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 2127–2130, 08 2015. doi: 10.1109/EMBC.2015.7318809.

[45] C. Fernández-Llatas, T. Meneu, J. M. Benedí, and V. Traver. Activity-based Process Mining for Clinical Pathways Computer aided design. In *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*, pages 6178–6181, Aug 2010. doi: 10.1109/IEMBS.2010.5627760.

[46] Diogo R. Ferreira, Fernando Szimanski, and Célia G. Ralha. Improving process models by mining mappings of low-level events to high-level activities. *Journal of Intelligent Information Systems*, 43(2):379–407, jul 2014. doi: 10.1007/s10844-014-0327-2.

[47] Roy T. Fielding, Richard N. Taylor, Justin R. Erenkrantz, Michael M. Gorlick, Jim Whitehead, Rohit Khare, and Peyman Oreizy. Reflections on the REST architectural style and "principled design of the modern web architecture" (impact paper award). In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering - ESEC/FSE 2017*. ACM Press, 2017. doi: 10.1145/3106237.3121282.

[48] Tamara G. Fong, Richard N. Jones, James L. Rudolph, Frances M. Yang, Douglas Tommet, Daniel Habtemariam, Edward R. Marcantonio, Kenneth M. Langa, and Sharon K. Inouye. Development and Validation of a Brief Cognitive Assessment Tool. *Archives of Internal Medicine*, 171(5), mar 2011. doi: 10.1001/archinternmed.2010.423.

[49] Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis*. Springer Berlin Heidelberg, 1999. doi: 10.1007/978-3-642-59830-2.

[50] Mahdi Ghasemi and Daniel Amyot. Process mining in healthcare: a systematised literature review. *International Journal of Electronic Healthcare*, 9(1):60, 2016. doi: 10.1504/ijeh.2016.078745.

[51] M. M. Günal and M. Pidd. Discrete event simulation for performance modelling in health care: a review of the literature. *Journal of Simulation*, 4(1):42–51, mar 2010. doi: 10.1057/jos.2009.25.

[52] Christian W. Günther, Stefanie Rinderle Ma, Manfred Reichert, Wil M. P. Van Der Aalst, and Jan Recker. Using process mining to learn from process changes in evolutionary systems. *International Journal of Business Process Integration and Management*, 3(1):61, 2008. doi: 10.1504/ijbpim.2008.019348.

[53] R. W. Grant and B. Middleton. Improving primary care for patients with complex chronic diseases: Can health information technology play a role? *Canadian Medical Association Journal*, 181(1-2):17–18, jul 2009. doi: 10.1503/cmaj.091101.

[54] Theresia Gschwandtner. Visual Analytics Meets Process Mining: Challenges and Opportunities. In *Lecture Notes in Business Information Processing*, pages 142–154. Springer International Publishing, 2017. doi: 10.1007/978-3-319-53435-0_7.

[55] Yves Guigoz, Bruno Vellas, and Philip J. Garry. Assessing the Nutritional Status of the Elderly: The Mini Nutritional Assessment as Part of the Geriatric Evaluation. *Nutrition Reviews*, 54(1):59–65, apr 2009. doi: 10.1111/j.1753-4887.1996.tb03793.x.

[56] Christian W. Günther. Process mining in flexible environments, 2009.

[57] Christian W. Günther and Anne Rozinat. Disco: Discover Your Processess. In *BPM*, 2012.

[58] Christian W. Günther and Wil M. P. van der Aalst. Fuzzy Mining – Adaptive Process Simplification Based on Multi-perspective Metrics. In Gustavo Alonso, Peter Dadam, and Michael Rosemann, editors, *Business Process Management*, pages 328–343, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-75183-0.

[59] Christian W. Günther, Anne Rozinat, and Wil M. P. van der Aalst. Monitoring deployed application usage with process mining. In *BPM reports*. BPMcenter. org, 2008.

[60] Festim Halili and Erenis Ramadani. Web Services: A Comparison of SOAP and REST Services. *Modern Applied Science*, 12(3), February 2018. doi: 10.5539/mas.v12n3p175.

[61] Matheus Hauder, Simon Pigat, and Florian Matthes. Research Challenges in Adaptive Case Management: A Literature Review. In *2014 IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations*. IEEE, sep 2014. doi: 10.1109/edocw.2014.24.

[62] Reinhold Haux, Alfred Winter, Elske Ammenwerth, and Birgit Brigl. *Strategic Information Management in Hospitals*, chapter Basic Concepts, pages 25–42. Springer New York, 2004. doi: 10.1007/978-1-4757-4298-5.

[63] Emmanuel Helm and Josef Küng. Process Mining: Towards Comparability of Healthcare Processes. In *Information Technology in Bio- and Medical Informatics*, pages 249–252. Springer International Publishing, 2016. doi: 10.1007/978-3-319-43949-5_20.

[64] Emmanuel Helm, Barbara Franz, Andreas Schuler, Oliver Krauss, and Josef Küng. Towards Standards Based Health Data Extraction Facilitating Process Mining. In *Proceedings of the International Workshop on Innovative Simulation for Health Care*, 2017.

[65] Carme Hernández, Jesus Aibar, Nuria Seijas, Imma Puig, Albert Alonso, Judith Garcia-Aymerich, and Josep Roca. Implementation of Home Hospitalization and Early Discharge as an Integrated Care Service: A Ten Years Pragmatic Assessment. *International Journal of Integrated Care*, 18(0), may 2018. doi: 10.5334/ijic.3431.

[66] Adrian Hernandez-Mendez, Felix Michel, and Florian Matthes. A Practice-Proven Reference Architecture for Model-Based Collaborative Information Systems. *Enterprise Modelling and Information Systems Architectures*, 13:262–273, 2018. doi: 10.18417/emisa.si.hcm.20.

[67] P. Homayounfar. Process mining challenges in hospital information systems. In *2012 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 1135–1140, Sep. 2012.

[68] Chad Horohoe. Wikimedia moving to Elasticsearch – The Wikimedia Blog, January 2014. URL https://blog.wikimedia.org/2014/01/06/wikimedia-moving-to-elasticsearch/.

[69] Bui The Hung, Nguyen Phuoc Long, Le Phi Hung, Nguyen Thien Luan, Nguyen Hoang Anh, Tran Diem Nghi, Mai Van Hieu, Nguyen Thi Huyen Trang, Herizo Fabien Rafidinarivo, Nguyen Ky Anh, David Hawkes, Nguyen Tien Huy, and Kenji Hirayama. Research Trends in Evidence-Based Medicine: A Joinpoint Regression Analysis of More than 50 Years of Publication Data. *PLOS ONE*, 10(4), apr 2015. doi: 10.1371/journal.pone.0121054.

[70] IBM. Grammar-based task analysis of web logs, 2010. URL https://patents.google.com/patent/US7694311B2/en?oq=US7694311.

[71] Ana Ivanchikj, Ilija Gjorgjiev, and Cesare Pautasso. RESTalk Miner: Mining RESTful Conversations, Pattern Discovery and Matching. In *Handbook of Experimental Pharmacology*, pages 470–475. Springer Berlin Heidelberg, 2019. doi: 10.1007/978-3-030-17642-6_46.

[72] Owen Johnson, Thamer Ba Dhafari, Angelina Kurniati, Frank Fox, and Eric Rojas. The ClearPath Method for Care Pathway Process Mining and Simulation. In *BPM 2018: Business Process Management Workshops*, volume 342, 09 2018.

[73] E. F. Juniper, P. M. O'Byrne, G. H. Guyatt, P. J. Ferrie, and D. R. King. Development and validation of a questionnaire to measure asthma control. *The European respiratory journal*, 14:902–907, October 1999. ISSN 0903-1936.

[74] Rachelle Kaye, Khaled Abu-Hosien, Felip Miralles, Eloisa Vargiu, and Bella Azaria. From Connected Care to Integrated Care: A Work In Progress (Presentation Slides), May 2017. URL http://www.connecare.eu/wp-content/uploads/2017/07/ICIC17.pdf. Accessed: 2019-03-27. (Archived by WebCite® at http://www.webcitation.org/77BiNShjG).

[75] Rachelle Kaye, Khaled Abu-Hossien, Felip Miralles, Eloisa Vargiu, and Bella Azaria. From Connected Care to Integrated Care – A work in Progress. *International Journal of Integrated Care*, 17(5), oct 2017. doi: 10.5334/ijic.3768.

[76] Rachelle Kaye, Reut Ron, Bella Azaria, Michal Bar-Ilan, Erela Rotlevi, and Tanya Iacubovsky. Integrated Care for High Risk Surgical Patients – Before, During and After. *International Journal of Integrated Care*, 18(s2), oct 2018. doi: 10.5334/ijic.s2241.

[77] Uzay Kaymak, Ronny Mans, Tim van de Steeg, and Meghan Dierks. On process mining in health care. In *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, oct 2012. doi: 10.1109/icsmc.2012.6378009.

[78] Daniel Keim, Gennady Andrienko, Jean-Daniel Fekete, Carsten Görg, Jörn Kohlhammer, and Guy Melançon. Visual Analytics: Definition, Process, and Challenges. In *Lecture Notes in Computer Science*, pages 154–175. Springer Berlin Heidelberg, 2008. doi: 10.1007/978-3-540-70956-5_7.

[79] Ateeq Khan, Azeem Lodhi, Veit Köppen, Gamal Kassem, and Gunter Saake. Applying Process Mining in SOA Environments. In *Service-Oriented Computing – ICSOC 2007*, pages 293–302. Springer Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-16132-2_28.

[80] Patrick Kleindienst. Building a real-world logging infrastructure with Logstash, Elasticsearch and Kibana, 2016.

[81] Nikolaus Kleiner. Delta analysis with workflow logs: aligning business process prescriptions and their reality. *Requirements Engineering*, 10(3):212–222, August 2005. doi: 10.1007/s00766-005-0004-7.

[82] J. Kondrup. Nutritional risk screening (NRS 2002): a new method based on an analysis of controlled clinical trials. *Clinical Nutrition*, 22(3):321–336, jun 2003. doi: 10.1016/s0261-5614(02)00214-5.

[83] Simone Kriglstein, Margit Pohl, Stefanie Rinderle-Ma, and Magdalena Stallinger. Visual Analytics in Process Mining: Classification of Process Mining Techniques. In *EuroVis Workshop on Visual Analytics*, June 2016.

[84] Angelina Prima Kurniati, Owen Johnson, David Hogg, and Geoff Hall. Process mining in oncology: A literature review. In *6th International Conference on Information Communication and Management (ICICM)*. IEEE, oct 2016. doi: 10.1109/infocoman.2016.7784260.

[85] 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*. ACM Press, 2015. doi: 10.1145/2723839.2723843.

[86] Hyukjin Kwon and Dongsoo Kim. Comparative evaluation of quantitative effects of information systems implementation. In *ICIC Express Letters*, volume 7 of *1*, pages 43–49. ICIC International, January 2016.

[87] Martin Lang, Thomas Bürkle, Susanne Laumann, and Hans-Ulrich Prokosch. Process Mining for Clinical Workflows: Challenges and Current Limitations. *Studies in health technology and informatics*, 136:229–34, 2008.

[88] Sander J. J. Leemans, Dirk Fahland, and Wil M. P. van der Aalst. Discovering Block-Structured Process Models from Event Logs - A Constructive Approach. In *Application and Theory of Petri Nets and Concurrency*, pages 311–329. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-38697-8_17.

[89] Sander J. J. Leemans, Dirk Fahland, and Wil M. P. van der Aalst. Scalable process discovery and conformance checking. *Software & Systems Modeling*, 17(2):599–631, jul 2016. doi: 10.1007/s10270-016-0545-x.

[90] Maria Leitner. Delta Analysis of Role-Based Access Control Models. In *Computer Aided Systems Theory - EUROCAST 2013*, pages 507–514. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-53856-8_64.

[91] R. Lenz, R. Blaser, M. Beyer, O. Heger, C. Biber, M. Bäumlein, and M. Schnabel. IT support for clinical pathways—Lessons learned. *International Journal of Medical Informatics*, 76:397–402, dec 2007. doi: 10.1016/j.ijmedinf.2007.04.012.

[92] G. Leonardi, M. Striani, S. Quaglini, A. Cavallini, and S. Montani. Towards Semantic Process Mining Through Knowledge-Based Trace Abstraction. In *JIMD Reports*, pages 45–64. Springer Berlin Heidelberg, 2019. doi: 10.1007/978-3-030-11638-5_3.

[93] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Symposium on Math, Statistics, and Probability*. University of California Press, 1967.

[94] F. I. Mahoney and D. W. Barthel. Functional Evaluation: The Barthel Index. *Maryland state medical journal*, 14:61–65, February 1965. ISSN 0025-4363.

[95] R. S. Mans, M. H. Schonenberg, M. Song, W. M. P. van der Aalst, and P. J. M. Bakker. Application of Process Mining in Healthcare – A Case Study in a Dutch Hospital. In *Biomedical Engineering Systems and Technologies*, pages 425–438. Springer Berlin Heidelberg, 2008. doi: 10.1007/978-3-540-92219-3_32.

[96] R. S. Mans, W. M. P. Aalst, van der, and R. J. B. Vanwersch. *Process mining in healthcare: opportunities beyond the ordinary*. BPM reports. BPMcenter.org, January 2013.

[97] R. S. Mans, W. M. P. Aalst, van der, and H. M. W. Verbeek. Supporting process mining workflows with RapidProM. In L. Limonad and B. Weber, editors, *BPM Demo Sessions 2014 (co-located with BPM 2014, Eindhoven, The Netherlands, September 20, 2014)*, CEUR Workshop Proceedings, pages 56–60. CEUR-WS.org, 2014.

[98] Ronny Mans, Helen Schonenberg, Giorgio Leonardi, Silvia Panzarasa, Anna Cavallini, Silvana Quaglini, and Wil van der Aalst. Process mining techniques: an application to stroke care. *Studies in health technology and informatics*, 136:573–578, 2008. ISSN 0926-9630.

[99] Ronny S. Mans, Wil M. P. van der Aalst, Rob J. B. Vanwersch, and Arnold J. Moleman. Process Mining in Healthcare: Data Challenges When Answering Frequently Posed Questions. In Richard Lenz, Silvia Miksch, Mor Peleg, Manfred Reichert, David Riaño, and Annette ten Teije, editors, *Process Support and Knowledge Representation in Health Care*, pages 140–153, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-36438-9.

[100] Mike A. Marin. Introduction to the case management model and notation (CMMN). *arXiv preprint arXiv:1608.05011*, 2016.

[101] Mike A. Marin, Matheus Hauder, and Florian Matthes. Case Management: An Evaluation of Existing Approaches for Knowledge-Intensive Processes. In *Business Process Management Workshops*, pages 5–16. Springer International Publishing, 2016. doi: 10.1007/978-3-319-42887-1_1.

[102] Christian Mauro, Tobias Happle, Ali Sunyaev, Helmut Krcmar, and Jan Marco Leimeister. From Medical Processes to Workflows - Modeling of Clinical Pathways with the Unified Modeling Language. In *HEALTHINF 2010 - 3rd International Conference on Health Informatics, Proceedings*, pages 388–391, 01 2010.

[103] Daniel I. McIsaac, Chelsey Saunders, Emily Hladkowicz, Gregory L. Bryson, Alan J. Forster, Sylvain Gagne, Allen Huang, Manoj Lalu, Luke T. Lavallee, Husein Moloo, Julie Nantel, Barbara Power, Celena Scheede-Bergdahl, Monica Taljaard, Carl van Walraven, and Colin J. L. McCartney. PREHAB study: a protocol for a prospective randomised clinical trial of exercise therapy for people living with frailty having cancer surgery. *BMJ Open*, 8(6), jun 2018. doi: 10.1136/bmjopen-2018-022057.

[104] Felix Michel and Florian Matthes. A Holistic Model-Based Adaptive Case Management Approach for Healthcare. In *2018 IEEE 22nd International Enterprise Distributed Object Computing Workshop (EDOCW)*. IEEE, oct 2018. doi: 10.1109/edocw.2018.00014.

[105] Felix Michel, Adrian Hernandez-Mendez, and Florian Matthes. An Overview of Tools for an Integrated and Adaptive Healthcare Approach. In *2018 IEEE 22nd International Enterprise Distributed Object Computing Workshop (EDOCW)*. IEEE, oct 2018. doi: 10.1109/edocw.2018.00015.

[106] David Monterde, Emili Vela, and Montse Clèries. Los grupos de morbilidad ajustados: nuevo agrupador de morbilidad poblacional de utilidad en el ámbito de la atención primaria. *Atención Primaria*, 48(10):674–682, dec 2016. doi: 10.1016/j.aprim.2016.06.003.

[107] V. C. Moore. Spirometry: step by step. *Breathe*, 8(3):232–240, mar 2012. doi: 10.1183/20734735.0021711.

[108] Mosby. *Mosby's Dictionary of Medicine, Nursing & Health Professions*. Elsevier - Health Sciences Division, 2016. ISBN 0323222056. URL https://www.ebook.de/de/product/25867674/mosby_mosby_s_dictionary_of_medicine_nursing_health_professions.html.

[109] Hamid R. Motahari-Nezhad and Keith D. Swenson. Adaptive Case Management: Overview and Research Challenges. In *2013 IEEE 15th Conference on Business Informatics*. IEEE, jul 2013. doi: 10.1109/cbi.2013.44.

[110] Thérèse Murphy. *New Technologies and Human Rights*. Oxford University Press, 2009.

[111] National Council for the Professional Development of Nursing and Midwifery. Improving the patient journey: understanding integrated care pathways. Technical report, National Council for the Professional Development of Nursing and Midwifery (NCPDMW), 2006.

[112] Nature Publishing Group. Embracing patient heterogeneity. *Nature Medicine*, 20(7):689–689, July 2014. doi: 10.1038/nm.3632.

[113] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, mar 1970. doi: 10.1016/0022-2836(70)90057-4.

[114] C. R. Nicolay, S. Purkayastha, A. Greenhalgh, J. Benn, S. Chaturvedi, N. Phillips, and A. Darzi. Systematic review of the application of quality improvement methodologies from the manufacturing industry to surgical healthcare. *British Journal of Surgery*, 99(3):324–335, November 2011. doi: 10.1002/bjs.7803.

[115] Ellen Nolte, Cécile Knai, and Martin Mckee. *Managing Chronic Conditions: Experience in Eight Countries*, volume European Observatory Studies Series. World Health Organization, 2008. ISBN 928904294X. URL https://www.ebook.de/de/product/8311175/managing_chronic_conditions_experience_in_eight_countries.html.

[116] Object Management Group. Case Management Model and Notation Specification Version 1.0, May 2014. URL https://www.omg.org/spec/CMMN/1.0/.

[117] Object Management Group. Case Management Model and Notation Specification Version 1.1, December 2016. URL https://www.omg.org/spec/CMMN/1.1/.

[118] Michael Overduin. Exploration of the link between the execution of a clinical process and its effectiveness using process mining techniques. mathesis, Eindhoven University of Technology, September 2013.

[119] M. Panella. Reducing clinical variations with clinical pathways: do pathways work? *International Journal for Quality in Health Care*, 15(6):509–521, dec 2003. doi: 10.1093/intqhc/mzg057.

[120] Andrew Partington, Moe Wynn, Suriadi Suriadi, Chun Ouyang, and Jonathan Karnon. Process Mining for Clinical Processes – A Comparative Analysis of Four Australian Hospitals. *ACM Transactions on Management Information Systems*, 5(4):1–18, jan 2015. doi: 10.1145/2629446.

[121] Jonas Poelmans, Guido Dedene, Gerda Verheyden, Herman Van der Mussele, Stijn Viaene, and Edward Peters. Combining Business Process and Data Discovery Techniques for Analyzing and Improving Integrated Care Pathways. In *Advances in Data Mining. Applications and Theoretical Aspects*, pages 505–517. Springer Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-14400-4_39.

[122] Nicolas Poggi, Vinod Muthusamy, David Carrera, and Rania Khalaf. Business Process Mining from E-Commerce Web Logs. In *Lecture Notes in Computer Science*, pages 65–80. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-40176-3_7.

[123] Sébastien Pujadas. Elasticsearch, Logstash, Kibana (ELK) Docker image documentation, February 2019. URL https://elk-docker.readthedocs.io/.

[124] H. Quan, B. Li, C. M. Couris, K. Fushimi, P. Graham, P. Hider, J.-M. Januel, and V. Sundararajan. Updating and Validating the Charlson Comorbidity Index and Score for Risk Adjustment in Hospital Discharge Abstracts Using Data From 6 Countries. *American Journal of Epidemiology*, 173(6):676–682, feb 2011. doi: 10.1093/aje/kwq433.

[125] Berry Reisberg, Steven H. Ferris, Mony J. de Leon, and Thomas Crook. The Global Deterioration Scale for assessment of primary degenerative dementia. *American Journal of Psychiatry*, 139(9):1136–1139, sep 1982. doi: 10.1176/ajp.139.9.1136.

[126] Williams Richard, Rojas Eric, Peek Niels, and A. Johnson Owen. Process Mining in Primary Care: A Literature Review. *Studies in Health Technology and Informatics*, 247 (Building Continents of Knowledge in Oceans of Data: The Future of Co-Created eHealth):376–380, 2018. ISSN 0926-9630. doi: 10.3233/978-1-61499-852-5-376.

[127] Eric Rojas, Jorge Munoz-Gama, Marcos Sepúlveda, and Daniel Capurro. Process mining in healthcare: A literature review. *Journal of Biomedical Informatics*, 61:224–236, jun 2016. doi: 10.1016/j.jbi.2016.04.007.

[128] Eric Rojas, Marcos Sepúlveda, Jorge Munoz-Gama, Daniel Capurro, Vicente Traver, and Carlos Fernandez-Llatas. Question-Driven Methodology for Analyzing Emergency Room Processes Using Process Mining. *Applied Sciences*, 7(3):302, mar 2017. doi: 10.3390/app7030302.

[129] S. T. Rosenbloom, R. J. Carroll, J. L. Warner, M. E. Matheny, and J. C. Denny. Representing Knowledge Consistently Across Health Systems. *Yearbook of Medical Informatics*, 26(01):139–147, 2017. doi: 10.15265/iy-2017-018.

[130] Vladimir Rubin, Irina Lomazova, and Wil M. P. van der Aalst. Agile development with software process mining. In *Proceedings of the 2014 International Conference on Software and System Process - ICSSP 2014*. ACM Press, 2014. doi: 10.1145/2600821.2600842.

[131] Vladimir A. Rubin, Alexey A. Mitsyuk, Irina A. Lomazova, and Wil M. P. van der Aalst. Process mining can be applied to software too! In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '14*. ACM Press, 2014. doi: 10.1145/2652524.2652583.

[132] Meyer Saklad. Grading of patients for surgical procedures. *Anesthesiology: The Journal of the American Society of Anesthesiologists*, 2(3):281–284, 1941.

[133] Desmond J. Sheridan and Desmond G. Julian. Achievements and Limitations of Evidence-Based Medicine. *Journal of the American College of Cardiology*, 68(2):204–213, jul 2016. doi: 10.1016/j.jacc.2016.03.600.

[134] S. A. Shershakov and V. A. Rubin. System Runs Analysis with Process Mining. *Modeling and Analysis of Information Systems*, 22(6):818–833, January 2016. doi: 10.18255/1818-1015-2015-6-818-833.

[135] Minseok Song and Wil M. P. van der Aalst. Supporting process mining by showing events at a glance. In *Proceedings of the 17th Annual Workshop on Information Technologies and Systems (WITS)*, pages 139–145, 2007.

[136] Minseok Song and Wil M. P. van der Aalst. Towards comprehensive support for organizational mining. *Decision Support Systems*, 46(1):300–317, dec 2008. doi: 10.1016/j.dss.2008.07.002.

[137] Minseok Song, Christian W. Günther, and Wil M. P. van der Aalst. Trace Clustering in Process Mining. In *Business Process Management Workshops*, pages 109–120. Springer Berlin Heidelberg, 2009. doi: 10.1007/978-3-642-00328-8_11.

[138] N. Steverink. Measuring frailty: developing and testing the GFI (Groningen Frailty Indicator). *The Gerontologist*, 41:236, 2001.

[139] Andrzej Stroiński, Dariusz Dwornikowski, and Jerzy Brzeziński. Resource Mining: Applying Process Mining to Resource-Oriented Systems. In *Business Information Systems*, pages 217–228. Springer International Publishing, 2014. doi: 10.1007/978-3-319-06695-0_19.

[140] Andrzej Stroinski, Dariusz Dwornikowski, and Jerzy Brzezinski. RESTful Web Service Mining: Simple Algorithm Supporting Resource-Oriented Systems. In *IEEE International Conference on Web Services*. IEEE, jun 2014. doi: 10.1109/icws.2014.106.

[141] Suriadi Suriadi, Ronny S. Mans, Moe T. Wynn, Andrew Partington, and Jonathan Karnon. Measuring Patient Flow Variations: A Cross-Organisational Process Mining Approach. In *Lecture Notes in Business Information Processing*, pages 43–58. Springer International Publishing, 2014. doi: 10.1007/978-3-319-08222-6_4.

[142] Keith D. Swenson. *Mastering the Unpredictable: How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done*. Meghan-Kiffer Press, 2010. ISBN 0929652126. URL https://www.ebook.de/de/product/13724164/keith_d_swenson_mastering_the_unpredictable_how_adaptive_case_management_will_revolutionize_the_way_that_knowledge_workers_get_things_done.html.

[143] Ákos Tényi, Emili Vela, Isaac Cano, Montserrat Cleries, David Monterde, David Gomez-Cabrero, and Josep Roca. Risk and temporal order of disease diagnosis of comorbidities in patients with COPD: a population health perspective. *BMJ Open Respiratory Research*, 5(1), jun 2018. doi: 10.1136/bmjresp-2018-000302.

[144] TFPM – IEEE Task Force on Process Mining. Process Mining Manifesto. In *Business Process Management Workshops*, pages 169–194. Springer Berlin Heidelberg, 2012. doi: 10.1007/978-3-642-28108-2_19.

[145] Malte Thiede, Daniel Fuerstenau, and Ana Paula Bezerra Barquet. How is process mining technology used by organizations? A systematic literature review of empirical studies. *Business Process Management Journal*, 24(4):900–922, jul 2018. doi: 10.1108/bpmj-06-2017-0148.

[146] UN. *World Population Ageing 2015*, volume Statistical Papers - United Nations (Ser. A), Population and Vital Statistics Report. United Nations, oct 2017. doi: 10.18356/88fa44e7-en. New York.

[147] R. Vaculin, R. Hull, T. Heath, C. Cochran, A. Nigam, and P. Sukaviriya. Declarative business artifact centric modeling of decision and knowledge intensive business processes. In *2011 IEEE 15th International Enterprise Distributed Object Computing Conference*, pages 151–160, Aug 2011. doi: 10.1109/EDOC.2011.36.

[148] Nick R. T. P. van Beest, Marlon Dumas, Luciano García-Bañuelos, and Marcello La Rosa. Log Delta Analysis: Interpretable Differencing of Business Process Event Logs. In *Lecture Notes in Computer Science*, pages 386–405. Springer International Publishing, 2015. doi: 10.1007/978-3-319-23063-4_26.

[149] W. M. P. van der Aalst. Business Alignment: Using Process Mining As a Tool for Delta Analysis and Conformance Testing. *Requirements Engineering*, 10(3):198–211, aug 2005. doi: 10.1007/s00766-005-0001-x.

[150] W. M. P. van der Aalst and C. W. Gunther. Finding Structure in Unstructured Processes: The Case for Process Mining. In *Seventh International Conference on Application of Concurrency to System Design (ACSD 2007)*. IEEE, July 2007. doi: 10.1109/acsd.2007.50.

[151] W. M. P. van der Aalst, M. de Leoni, and A. H. M. ter Hofstede. *Process mining and visual analytics: breathing life into business process models*. BPM reports. BPMcenter.org, 2011.

[152] Wil van der Aalst. Process mining: discovering and improving Spaghetti and Lasagna processes. In *IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*. IEEE, apr 2011. doi: 10.1109/cidm.2011.6129461.

[153] Wil van der Aalst. Process Mining. *ACM Transactions on Management Information Systems*, 3(2):1–17, jul 2012. doi: 10.1145/2229156.2229157.

[154] Wil van der Aalst. *Process Mining: Data Science in Action*, chapter Part III - From Event Logs to Process Models, pages 123–240. Springer Berlin Heidelberg, 2016. doi: 10.1007/978-3-662-49851-4.

[155] Wil van der Aalst. *Process Mining: Data Science in Action*, chapter Chapter 11 - Process Mining Software, pages 325–352. Springer Berlin Heidelberg, 2016. doi: 10.1007/978-3-662-49851-4.

[156] Wil van der Aalst. Spreadsheets for business process management. *Business Process Management Journal*, 24(1):105–127, feb 2018. doi: 10.1108/bpmj-10-2016-0190.

[157] Wil M. P. van der Aalst. *Transactions on Petri Nets and Other Models of Concurrency II*, chapter Process-Aware Information Systems: Lessons to Be Learned from Process Mining, pages 1–26. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-642-00899-3. doi: 10.1007/978-3-642-00899-3_1. URL https://doi.org/10.1007/978-3-642-00899-3_1.

[158] Wil M. P. van der Aalst. Challenges in Service Mining: Record, Check, Discover. In *Lecture Notes in Computer Science*, pages 1–4. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-39200-9_1.

[159] Wil M. P. van der Aalst. Service Mining: Using Process Mining to Discover, Check, and Improve Service Behavior. *IEEE Transactions on Services Computing*, 6(4):525–535, oct 2013. doi: 10.1109/tsc.2012.25.

[160] Wil M. P. van der Aalst. Process discovery from event data: Relating models and logs through abstractions. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(3):1244, feb 2018. doi: 10.1002/widm.1244.

[161] Wil M. P. van der Aalst and H. M. W. Verbeek. Process Mining in Web Services: The WebSphere Case. *IEEE Data Eng. Bull.*, 31:45–48, 2008.

[162] Wil M. P. van der Aalst, Hajo A. Reijers, and Minseok Song. Discovering Social Networks from Event Logs. *Computer Supported Cooperative Work (CSCW)*, 14(6):549–593, oct 2005. doi: 10.1007/s10606-005-9005-9.

[163] Wil M. P. van der Aalst, Marlon Dumas, Chun Ouyang, Anne Rozinat, and Eric Verbeek. Conformance checking of service behavior. *ACM Transactions on Internet Technology*, 8(3):1–30, may 2008. doi: 10.1145/1361186.1361189.

[164] Thys van der Molen, Brigitte W. M. Willemse, Siebrig Schokker, Nick H. T. ten Hacken, Dirkje S. Postma, and Elizabeth F. Juniper. Development, validity and responsiveness of the Clinical COPD Questionnaire. *Health and Quality of Life Outcomes*, 1(1):13, 2003. doi: 10.1186/1477-7525-1-13.

[165] B. F. van Dongen, A. K. A. de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters, and W. M. P. van der Aalst. The ProM Framework: A New Era in Process Mining Tool Support. In *Applications and Theory of Petri Nets 2005*, pages 444–454. Springer Berlin Heidelberg, 2005. doi: 10.1007/11494744_25.

[166] Boudewijn F. van Dongen and Wil M. P. van der Aalst. A Meta Model for Process Mining Data. In *In Proceedings of the CAiSE WORKSHOPS*, pages 309–320, 2005.

[167] Eloisa Vargiu, Juan Manuel Fernàndez, Felip Miralles, Isaac Cano, Elena Gimeno-Santos, Carme Hernandez, Gerard Torres, Jordi Colomina, Jordi de Batlle, Rachelle Kaye, Bella Azaria, Shauli Nakar, M. H. Lahr, Esther Metting, Margot Jager, Hille

Meetsma, Stefano Mariani, Marco Mamei, Franco Zambonelli, Felix Michel, Florian Matthes, Jo Goulden, John Eaglesham, and Charles Lowe. Integrated Care for Complex Chronic Patients. *International Journal of Integrated Care*, 17(5):302, October 2017. doi: 10.5334/ijic.3619.

[168] Emili Vela, Ákos Tényi, Isaac Cano, David Monterde, Montserrat Cleries, Anna Garcia-Altes, Carme Hernandez, Joan Escarrabill, and Josep Roca. Population-based analysis of patients with COPD in Catalonia: a cohort study with implications for clinical management. *BMJ Open*, 8(3), mar 2018. doi: 10.1136/bmjopen-2017-017283.

[169] W3Techs. Usage Statistics and Market Share of Web Servers, April 2019, April 2019. URL https://w3techs.com/technologies/overview/web_server/all.

[170] John Ware, M. Kosinski, Diane Turner-Bowker, and B. Gandek. How to score SF-12 items. *SF-12 v2: How to Score Version 2 of the SF-12 Health Survey*, pages 29–38, 01 2002.

[171] Kate Watson and Chelsea Harper. Supporting Knowledge Creation – Using Wikis for Group Collaboration. In *Educause Australasia Conference Proceedings*, 2007.

[172] Jochen De Weerdt, Filip Caron, Jan Vanthienen, and Bart Baesens. Getting a Grasp on Clinical Pathway Data: An Approach Based on Process Mining. In *Lecture Notes in Computer Science*, pages 22–35. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-36778-6_3.

[173] Daniel Shawcross Wilkerson. A Proposal for Proquints: Identifiers that are Readable, Spellable, and Pronounceable. *CoRR*, abs/0901.4016, 2009.

[174] Julia K. Wolff, Sonja Nowossadeck, and Svenja M. Spuling. *Altern im Wandel: Zwei Jahrzehnte Deutscher Alterssurvey (DEAS)*, chapter Altern nachfolgende Kohorten gesünder? Selbstberichtete Erkrankungen und funktionale Gesundheit im Kohortenvergleich, pages 125–138. Springer Fachmedien Wiesbaden, Wiesbaden, 2017. ISBN 978-3-658-12502-8. doi: 10.1007/978-3-658-12502-8_8.

[175] Wei Yang and Qiang Su. Process Mining for Clinical Pathway: Literature Review and Future Directions. In *2014 11th International Conference on Service Systems and Service Management (ICSSSM)*. IEEE, jun 2014. doi: 10.1109/icsssm.2014.6943412.

[176] Zhichao Zhou, Yong Wang, and Lin Li. Process mining based modeling and analysis of workflows in clinical care - A case study in a chicago outpatient clinic. In *Proceedings of the 11th IEEE International Conference on Networking, Sensing and Control*. IEEE, apr 2014. doi: 10.1109/icnsc.2014.6819692.

[177] A. S. Zigmond and R. P. Snaith. The Hospital Anxiety and Depression Scale. *Acta Psychiatrica Scandinavica*, 67(6):361–370, jun 1983. doi: 10.1111/j.1600-0447.1983.tb09716.x.