Ulm University | 89069 Ulm | Germany

Faculty of Engineering, Computer Science and Psychology
Institute of Databases and Information Systems

The Empirical Analysis of the Comprehensibility of Process Models created by Process Mining

Master's thesis at Ulm University

Submitted by:
Jana Bühler
jana.buehler@uni-ulm.de
871153

Reviewer:
Prof. Dr. Manfred Reichert
Prof. Dr. Rüdiger Pryss

Supervisor:
Michael Winter

2021

Version of September 9, 2021

Satz: PDF-LaTeX 2_ε

Abstract

Companies use process models to specify their operational processes. With the help of process models, the business processes in a company are analysed by process mining techniques to optimise them. The subdiscipline of process discovery identifies the actual state of business processes and enables them to be examined. Various tools and algorithms can be used, which lead to different process visualisations. The type of process visualisation has a major influence on the comprehensibility of process models.

The objective of this thesis is to investigate the comprehensibility of process models generated by process mining. For this purpose, an exploratory eye-tracking study is conducted with fifteen participants. The study examines process models from two scenarios - a vaccination process and an insurance process. The corresponding process models are created manually, and event logs are generated from them using self-created applications. These event logs are loaded into the process mining tools Celonis Snap, Disco, ProM, Apromore and PM4Py and process models are generated from them. A selection of the resulting process models is then tested for comprehensibility in the user study. The analysis of variance (ANOVA) shows no significant differences between the different generated process models. Finally, with the Pearson correlation's help, the participants' subjective ranking is highly significantly related to the level of acceptability and cognitive load. The correlation between the time spent looking at the process models and the number of correctly answered comprehension questions is interesting. From this correlation, it can be concluded that understanding process models requires a certain amount of time.

An astonishing result of the study is that the quality between manually created models and models generated by process mining is similarly high. Despite interesting results, further studies are needed, as the study is confronted with some limitations (particularly the number of participants). The results can be used as a basis for future studies to further explore this field of research.

Acknowledgement

At this point I would like to thank everyone who supported and motivated me in the preparation of this Master's thesis.

First and foremost, I would like to thank all the participants in my study, without whom this thesis could not have been written. I thank them for their willingness to provide all information needed and for their answers and contributions to my questionnaire.

Finally, I would like to thank my parents whose support made my study possible.

Contents

1 Introduction

1.1 Motivation

Organisations can use process models to specify their operational processes [2]. By having a graphical representation of the process models, the processes can be better understood by different stakeholders [28].

Information systems are used to support business processes and enable users to control them [4]. Business data is traditionally stored in many different systems, such as ERP systems, to name an important one [16]. These systems directly or indirectly record business activities and can be used to create event logs [4]. Based on these logs, process visualisations may be created with the help of *process mining algorithms* and *process analyses* may be carried out. *Process mining* is seen as an interface between the fields of *Business Process Management (BPM)* and *data mining* [4]. Process mining combines traditional process model analyses, and data-oriented analyses [4]. Traditional process analyses include procedures for the simulation and verification of process models. Data-oriented analyses use real data and employ procedures from data mining, and do not focus on processes [4].

Process mining makes the identification of the actual state of business processes easier and is used to *discover*, *monitor* and *improve* processes [4]. Based on event logs, discovery is about getting a picture of the as-is processes automated. Process mining tools offer different views of the generated process model. These views facilitate the analysis of the business process concerned. Monitoring helps to find out how well the real execution of the process matches the documented or specified process model. Improvement is about identifying and solving bottlenecks and other weaknesses to optimise the business process and thus achieving greater added value. Improving process models reduces the risk of problems [16]. "The need to improve business processes is a competitive advantage for companies and should therefore be supported as much as possible." [48]. Therefore, the quality of the created process visualisation is of great importance.

In this thesis, the focus is on process discovery and the generated process models. A process discovery algorithm is used that extracts knowledge from a given event log to represent it as a process visualisation [4]. Various tools and algorithms can be used and which lead to different process visualisations. One type of visualisation that has been around for a long time is Petri nets. They were developed in the 1960s by Carl Adam Petri to represent a model for the flow of information [37]. In [47] an adapted representation of process visualisations, so-called Causal nets, is presented. A popular way to visualise process models is the Business Process Modelling Notation (BPMN) [35]. With the help of BPMN, process models can conceptually represent how the process should run. BPMN can also be used to create so-called workflows that can be executed by a business process engine [13].

The type of process visualisation has a major influence on the comprehensibility of process models. The research on this has been going on for many years [42, 54, 55]. The focus is on studying process models created by different process modellers, and people should understand the process visualisations. There are already studies that the experience of the viewer influences the comprehensibility of the process model, so experts understand process models more effectively than novices [33, 56]. Currently, there is no research on how understandable the process models generated by process discovery are. Therefore, a user study is conducted using eye-tracking to examine the comprehensibility of these models.

1.2 Objective

In this work, the comprehensibility of process models created by process discovery is investigated. For this purpose, two scenarios are created. Event logs have to be generated first to apply process mining techniques. In the next step, different process mining tools with different algorithms are evaluated concerning the generation of process visualisations. An important goal is to determine which kind of process visualisation is best suited to understand the process scenario. Therefore, the comprehensibility and cognitive load are examined. A study is to be conducted with the help of an eye-tracker. To the best knowledge of the author, there has been no earlier study that addresses this topic. Therefore the findings will be a good starting point for further investigations.

1.3 Structure of Thesis

After presenting the motivation and objectives of the thesis, Chapter 2 presents essential topics that help understand the thesis. First, Section 2.1 explains the different types of process visualisations that are used in process mining. Various process scenarios are considered in the context of the study carried out. These are described in Section 2.2. Section 2.3 explains the basics of process mining. Important terms and central techniques of process mining are explained. Subsequently, important process mining algorithms are presented in Section 2.4. Section 2.5 describes five essential process mining tools that are used during the thesis. Section 2.6 deals with the topic of eye-tracking. Important principles and techniques are described. Finally, the eye-tracking set "Pupil Core", used to conduct the study, is presented. In Chapter 3 related works are discussed. On the one hand, the papers deal with comprehensibility in general. On the other hand, papers are referenced that have also conducted user studies with the help of eye-tracking in the area of process visualisations. Chapter 4 deals with the implementation of the user study. For this purpose, among other things, the structure with the materials of the study and its procedure is explained. In Chapter 5 the results of the analyses are described and tested for possible significance. The thesis concludes with Chapter 6, which gives a summary of the thesis and an outlook on further possible work.

2 Fundamentals

This Chapter deals with the basic terms and topics from the field of process modelling and process mining. Furthermore, an introduction to eye-tracking is given to understand how the comprehensibility of process diagrams can be practically examined and evaluated.

2.1 Process Visualisations

For more than 25 years now, the consideration of business processes has increasingly become the focus of companies. In [26] the foundations for business process reengineering are laid. In the meantime, many consulting firms have specialised in analysing, executing, monitoring, and optimising processes. An organisation oriented towards business processes reduces costs and helps to achieve its strategic goals more quickly [34]. Many different modelling languages exist to represent business processes. This Section describes essential process visualisations that also play an important role in process mining.

2.1.1 Petri Nets

Petri nets go back to the work of Carl Adam Petri in 1962, who in his dissertation considered, among other things, simultaneous models, which are known today as Petri nets [37]. A net usually consists of several *places* which can be marked by one or more *tokens*. There are also Petri nets that may only hold one token per place. These are not considered in this thesis. A token indicates the holding of a possible state. Between places there are *transitions*. Transitions enable the change of a place. Places and transitions are connected by arcs representing the flow. The consideration of sequence, simultaneity, and conflicts are sufficient to describe all basic situations in Petri nets [3].

As in [3] a Petri net is formally defined as follows.

Definition 1 (Petri net)
A Petri net is a triple (P, T, F):
- P is a finite set of places,
- T is a finite set of transitions ($P \cap T = \emptyset$),
- $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (flow relation)

The following Figure 2.1 shows a Petri net with a simplified vaccination process. The process contains three transitions (check-in, receive vaccination, and check-out). Some places in the example contain tokens. These move through the net during the processing. The state of the process can be read from the distribution of the tokens [3]. Three process instances are in the first state, on arrival at the vaccination centre. Two instances have already passed the check-in and are now waiting for the vaccination. Three instances have already gone through the complete process and have reached the end, leaving the vaccination centre.

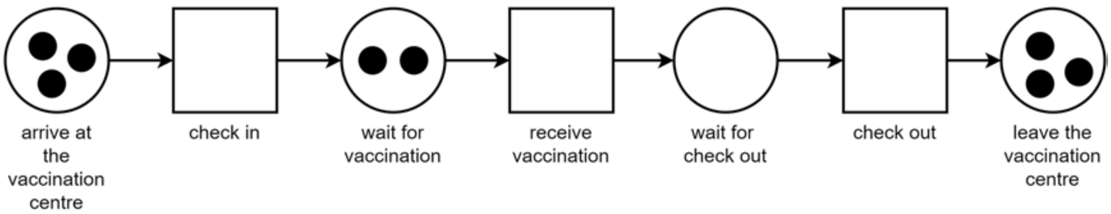

Figure 2.1: Simplified vaccination process as a Petri net

The use of Petri nets for workflow management has been considered for more than 20 years [3]. In addition, as Van der Aalst state in [47], there are many algorithms in the field of process mining that may produce Petri nets. He mentions among others variants of the α-algorithm [5, 17].

In terms of process mining, the places do not have much significance. The addition or removal of places may, however, introduce anomalies such as deadlocks or livelocks [47]. Central are the transitions that represent the different process steps.

There are different opinions on how good Petri nets are. In [3] some reasons are given why Petri nets are useful and shall be used for process analysis. Petri nets follow formal semantics and are graphical, enabling the modelling of workflows. Another advantage is the explicit representation of the status of the process. Petri nets can also be used for analyses to determine the correctness of workflow process definitions.

In recent years there has been a trend towards the use of Causal nets or Heuristic nets as they are considered more beneficial than Petri nets [47].

2.1.2 Causal Nets

Causal nets, for short C-nets, were first introduced by Van der Aalst in 2011 [47]. This kind of graph aims to customise and improve the representation of process models generated by process mining algorithms. According to Van der Aalst, C-nets shall be better suited than other process notation languages such as Petri nets or BPMN.

A C-net consists of nodes and directed edges, which represent causal dependencies. The nodes represent process activities. There is precisely one start and one end node. The difference between simple dependency graphs is that so-called *input* and *output bindings* are introduced to express the routing logic. Black dots represent bindings. A set of connected dots form a binding. If a binding is added after a node, it is called an output binding. If a binding is placed before a node, it is called an input binding.

As in [47] a C-net is formally defined as follows.

Definition 2 (C-net)
A Causal net (C-net) is a tuple $C = (A, a_i, a_o, D, I, O)$ *where:*
– A *is a finite set of activities;*
– $a_i \in A$ *is the start activity;*
– $a_o \in A$ *is the end activity;*
– $D \subseteq A \times A$ *is the dependency relation,*
– $AS = \{X \subseteq \mathcal{P}(A) \mid X = \{\varnothing\} \vee \varnothing \notin X\}$;
– $I \in A \rightarrow AS$ *defines the set of possible input bindings per activity; and*
– $O \in A \rightarrow AS$ *defines the set of possible output bindings per activity,*
such that
– $D = \{(a1, a2) \in A \times A \mid a_1 \in \bigcup_{as \in I(a_2)} as\}$;

– $D = \{(a1, a2) \in A \times A \mid a_2 \in \bigcup_{as \in I(a_1)} as\}$;
– $\{a_i\} = \{a \in A \mid I(a) = \{\varnothing\}\}$;
– $\{a_o\} = \{a \in A \mid O(a) = \{\varnothing\}\}$; *and*
– *all activities in the graph* (A, D) *are on a path from* a_i *to* a_o.

In the C-net, all activities and the start and end activities are defined first. Then the dependencies between the activities (see set D) can be defined. The possible input and output bindings are defined for each activity. In the refinement of the definition, it can be seen that for each dependency of activities a_1 and a_2, a_2 must have an input binding with a_1, and a_1 must have an output binding with a_2. In addition, the start activity has no input bindings, and the end activity has no output bindings.

The following Figure 2.2 shows a booking process. After the start of the process, there are three possible follow-up activities. However, the output bindings limit the number of possible versions after the start. For example, it is not possible to book only a hotel. A hotel can only be booked with a flight or booked with a flight and a car. The process must be started for a car to be booked, and a flight must have been booked before. At first, the notation looks pretty abstract. However, after understanding the principle of input and output bindings, one understands the possible process flows very well.

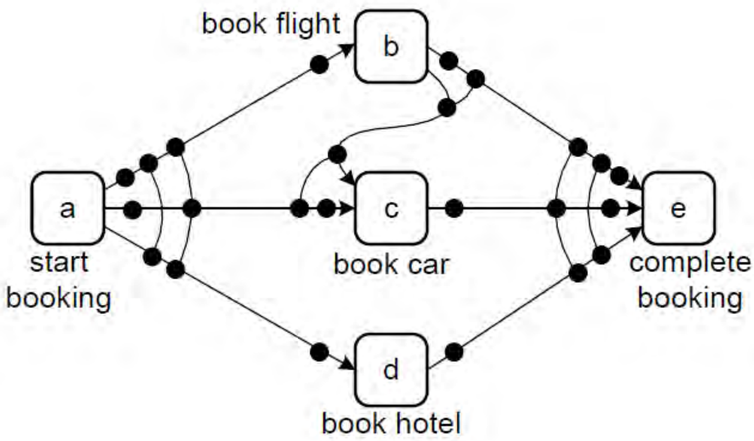

Figure 2.2: C-net for booking process [47]

As Van der Aalst explains in [47] "output bindings create obligations whereas input bindings remove obligations". If the node "complete booking" is taken, the booking can only be completed if a flight is booked. Another obligation is that a car is booked. In total, four obligations may be valid to successfully complete a booking (see input bindings of node "complete booking").

A C-net is sound if it does not contain any anomalies (e.g. deadlocks or livelocks) and forces the completion of the process. For this, it must be checked whether there is a valid sequence. All parts of the C-net must potentially be able to be activated by such a sequence.

In summary, the behaviour of process models is accurately described by valid, binding sequences, and C-nets do not follow the token-game principle as is the case in Petri net-based approaches. The advantage of C-nets is that XOR/OR/AND splits and joins traditionally used in other process model notations can be replaced by a more compact and richer semantics. In [47] Van der Aalst et al. show how C-nets can be mapped to Petri nets and vice versa.

2.1.3 Business Process Model and Notation

The Business Process Model and Notation (BPMN) was developed by Stephen A. White of IBM and first published by the Business Process Management Initiative in 2004 [22]. BPMN provides a standardised graphical process notation that can visualise not only conceptual models but also represent executable processes [13]. In the meantime, the OMG is responsible for the further development of BPMN, which attempts to combine readability, flexibility, and extensibility [35]. The currently valid version 2.0 was adopted by the OMG in 2011 [22].

The BPMN 2.0 offers a variety of modelling elements for the representation of processes. These can be roughly assigned to five categories: Flow Objects, Connection Objects, Artefacts, Participants, and Data [22]. Specific process steps, so-called activities, must always be carried out in a process, and certain events can occur. Sequence flows connect these flow objects. There can be conditions that divide the sequence flow in the process, and only activities that fulfil this condition are executed.

The following Figure 2.3 shows a BPMN process model for a simplified vaccination process. The process starts with the event at the confirmed date of vaccination.

Now the three activities *check in*, *wait* and *receive vaccination* are carried out sequentially. Now the XOR gateway decides whether side effects have occurred. If this is the case, an additional activity (receive first aid) is carried out. Afterwards, the sequence flow rejoins and the last two activities can be executed.

The process ends when the vaccination is completed.

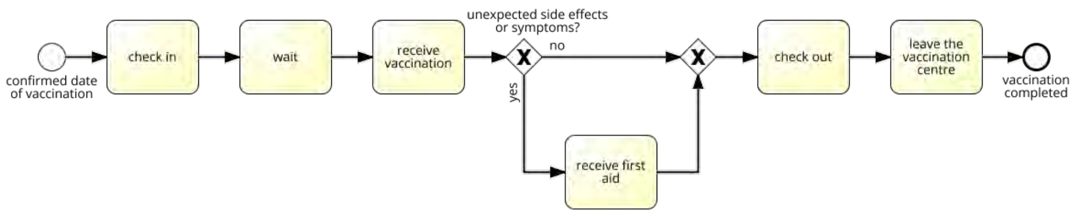

Figure 2.3: Simplified vaccination process as a BPMN model

BPMN offers many more possibilities to visualise processes in great detail [22]. However, BPMN diagrams in the context of process mining are mainly limited to the basic elements shown in the simplified example.

2.2 Process Scenarios

After explaining different forms of representation for processes, the process scenarios used for the user study are presented. The first process is the vaccination process. This scenario is chosen because it is essentially a sequential process and is easy to implement. The second process is an insurance process. This process contains advanced BPMN elements. Therefore, it is expected that investigating the comprehensibility of the process diagram created by process mining will reveal several problem areas.

2.2.1 Process of Vaccination

This process has been modelled after the recommendation from the German Federal Ministry of Health and the Robert Koch Institute for vaccination against SARS-CoV-2 in vaccination centres (as of December 2020) [12], seen in Figure 2.4.

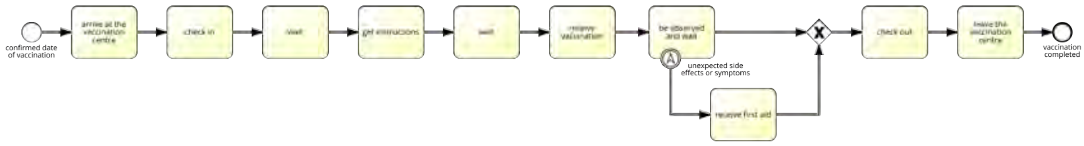

Figure 2.4: Manually created process model of vaccination, based on [12]

The procedure starts with the arrival at the vaccination centre at the confirmed appointment. Access control takes place here. The patient then comes to the check-in, where the registration data and vaccination eligibility are checked. After a short wait, the patient receives a medical explanation and can ask questions.

Then he has to wait again. Next, the patient is vaccinated. After the vaccination, the patient is observed for about 15-30 minutes. If unexpected side effects or symptoms occur during this time, first aid is given. Otherwise, the patient comes directly to the reception, where he receives an entry in his vaccination card and can then leave the vaccination centre. Leaving concludes the procedure.

Other conceivable exceptional cases are not considered in the process, because otherwise the model would become unnecessarily inflated. These include entering the centre without an appointment, dropping out at any possible time and missing necessary documents.

No detailed modelling and development of an executable process model are carried out to obtain the event log as quickly and simply as possible. Instead, the event log is generated directly. A Python script is written to obtain the event log, which is included in the Appendix A. A list of activities with time intervals, how long they can last is given as a text file. The script creates a CSV file from it. For each activity, a random timestamp is calculated within the time interval. A line in the CSV file contains a corresponding case ID, the activity name, and a start and end timestamp.

The event log contains 1000 process instances. Of these, the special case of first aid is included for 111 instances. Care is taken when creating the event log to ensure that the special case occurs often enough to be recognised as a different case but rarely enough that it is clear that it is a special case. Attention to the number of instances ensures that the problem of noise and incompleteness is addressed.

2.2.2 Process of Insurance

The process begins when a client's insurance application is received by the clerk (see in Figure 2.5). He checks the application and decides whether the application is valid and can be accepted or not. If not, the rejection is initiated, and the sub-process for rejecting the application is started. After the rejection of the application, the client is informed, and the process is finished. If the application is accepted, the system checks whether the customer is an existing customer or not. If not, the customer data are entered into the system by an additional step. After that, the contract terms are determined, and the insurance policy is issued. If the issue takes more than two days, the department head prioritises the application check, and the clerk issues the insurance policy. Then the insurance policy is sent to the client, and the application is completed.

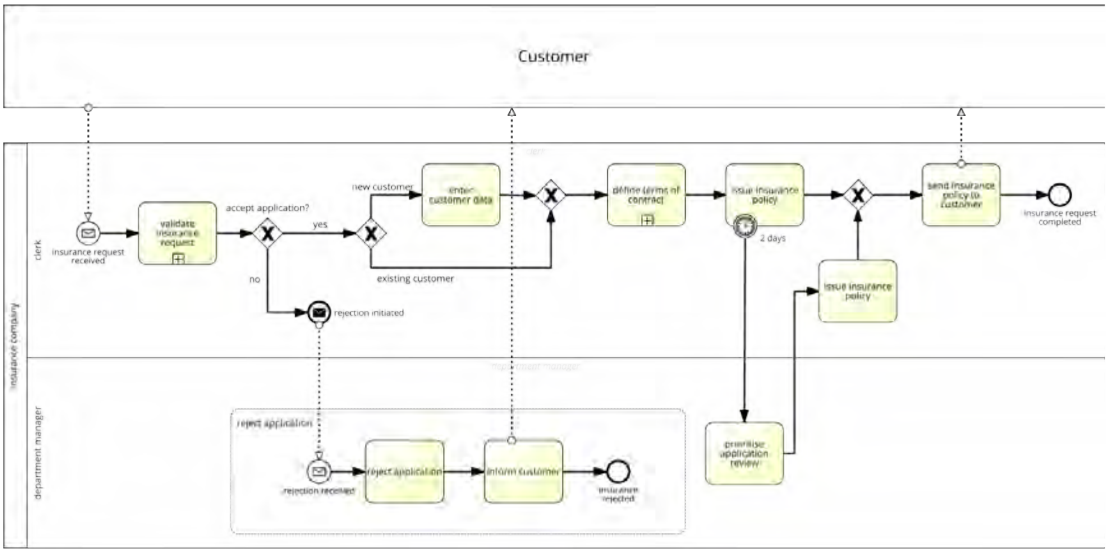

Figure 2.5: Manually created process model of insurance in BPMN

A corresponding business process is implemented in Camunda to create an event log for this process. For this purpose, an executable model is created, which can be seen in Figure 2.6. The executable BPMN process model is integrated into a Spring Boot application.

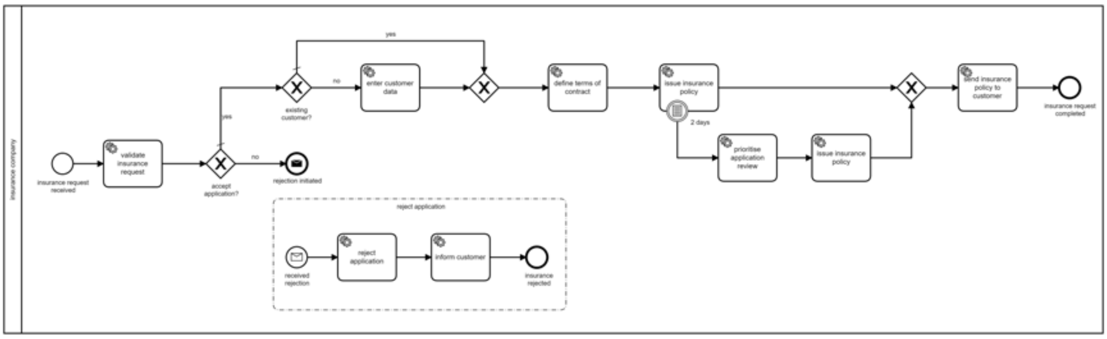

Figure 2.6: Implemented process model of insurance in Camunda

To be able to start the process, there is the class *WebAppMainProcessApplication* (see in Listing 2.1). The annotations *@EnableProcessApplication* and *@SpringBootApplication* declare the process application for the Camunda Spring Boot application, whereby the process instance can be started at this point. The main method starts the application.

```
@EnableProcessApplication
@SpringBootApplication
public class WebAppMainProcessApplication {
...
 public static void main(String... args) {
  SpringApplication.run(WebAppMainProcessApplication.
      ↪ class, args);
 }
...
}
```

Listing 2.1: Start of a spring boot application

Once the process is deployed, the method *processPostDeploy* of the *WebAppMainProcessApplication* class is called to launch 10 thousand process instances (see in Listing 2.2).

```
// set number of instances for event log
private int instances = 10000;
...
@EventListener
```

```
private void processPostDeploy(PostDeployEvent event) {
 // start process instances
 for(int i=0; i<instances; i++) {
  runtimeService.startProcessInstanceByKey("insurance_process");
 }
}
```

Listing 2.2: Run 10000 process instances

There is a JavaDelegate class that implements the specific business logic for each task in the process. However, there is no real need to implement the business logic for this insurance process. The focus of the implementation is on generating the event log. Therefore, each delegate implementation realises a random waiting time in a specific range. The delegate classes all look analogous to the class *EnterCustomerDataDelegator* in Listing 2.3.

```
public class EnterCustomerDataDelegater implements JavaDelegate {
// minimum and maximum for random waiting
private int min = 1;
private int max = 5;

@Override
public void execute(DelegateExecution delegateExecution) throws Exception {
 System.out.println("*** enter customer data ***");

 // wait random time
 int randomWait = (int) (max * Math.random() + min);
 TimeUnit.SECONDS.sleep(randomWait);

 System.out.println("*** customer data is entered ***");
}
```

Listing 2.3: Delegator class for task ”enter customer data”

Only for deciding which process path to take at the XOR gates, the implementation of the decision logic is required. Therefore, a random variable is initialised. The random variable is then checked at the appropriate point with the modulo operation, and the process instance is continued on the correct path. In the following Listing 2.4, the variable *randomAccept* is initialised. In 90% of the cases, the variable should be set to true, and only 10% of the requests should be rejected.

```
int randomAccept = (int) (Math.random()*(10-1)) + 1;
// set the variable acceptApplication to false in 1 tenth cases, otherwise true
if(randomAccept % 9 == 0) {
 delegateExecution.setVariable("acceptApplication", false);
} else {
 delegateExecution.setVariable("acceptApplication", true);
}
```

Listing 2.4: Decision handling for processing an insurance application

The same approach is used to decide whether the client already exists and where the application check should be prioritised. The client already exists in 50% of the cases. The application check must be prioritised in three-tenths of the cases.

In order to be able to generate the event log, the Spring Boot application is started, and 10 thousand instances are being executed. The current status and history of the running instances can be observed during the execution in the Camunda Cockpit. The history of the executed instances can be seen in Figure 2.7.

During the creation and realisation of the technical model, a problem occurred that is explained in more detail below.

When looking at the process model in Figure 2.5, it is noticeable that a boundary timer event is used in the process model to catch the case that the insurance request is not processed further. However, this event cannot be adopted for the executable model. A process instance that is being executed in a thread can only be interrupted from the outside. However, since the process is to be interrupted from the inside (depending on the variable's value), this is not feasible.

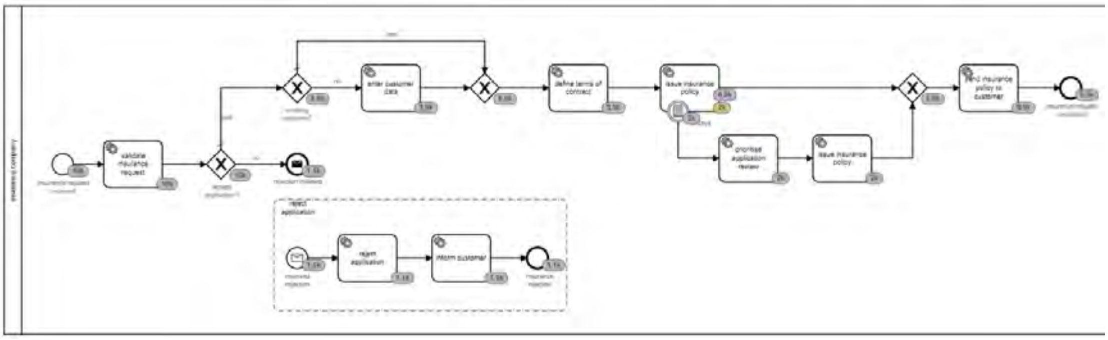

Figure 2.7: History view of Camunda Cockpit with all cases

Therefore, an intermediate conditional event is used as a workaround*.

It is also noticeable that the pools and lanes are not taken from the process model and that all activities are created as service tasks in the technical model. The service tasks are used because the goal is to get a big event log as quickly as possible. It should not be necessary to do something manually for every user task in every instance. This way, the process can be started once with many instances and run through directly.

A Python script is written that accesses the historical process data stored on the Camunda BPM platform to generate an event log. The platform's REST API allows direct access to the required information. In the header of the REST call, the content type is set to *application/json*, and the REST command is then sent to the engine via the endpoint *http://localhost:8080/engine-rest/*. The implementation of the Python method *make_rest_call*, with which the various REST calls are executed, is shown in Listing 2.5.

```
def make_rest_call(rest_method, params=None):
  url_endpoint = 'http://localhost:8080/engine-rest/'
  r_headers = {'Content-Type': 'application/json'}
  res = requests.get(url_endpoint + rest_method, params,
     ↪ headers=r_headers, timeout=3.0)
  if res.ok is False:
   print('make rest call failed')
  return res
```

Listing 2.5: Function used for REST calls

In the following Listing 2.6, the creation of the event log is explained in more detail. First, all process instances must be determined from the historical data via a process definition key. Using the method *get_process_instances*, all associated process IDs are determined via the REST command *history/process-instance* and saved in a list.

```
def get_process_instances(process_definition_key):
 payload = {'definitionKey': process_definition_key, 'finished': 'true'}
 res = make_rest_call('history/process-instance', params=payload)

 # get result as json
 response_json = res.json()

 # extract instances from json and save as list
 process_instances = []
 # get first element
 if response_json:
  elem = response_json.pop()
 else:
  print('instances response json is empty')
  return None

 # get next elements
 while elem['id']:
  process_instances.append(elem['id'])
  # if there are more elements, else break loop
  if response_json:
   elem = response_json.pop()
  else:
   break

 return process_instances
```

Listing 2.6: Function to get all process instance IDs

In the next step, a REST request for each process instance is executed in the *get_process_activities* method to obtain the executed process activities, including timestamps. The results are formatted accordingly and stored in the activity list. The code is shown in Listing 2.7.

```
def get_process_activities(instance_id):
  payload = {'processInstanceId': instance_id, 'sortBy':
      'startTime', 'sortOrder': 'desc'}
  res = make_rest_call('history/activity-instance',
      params=payload)

  # get result as json
  response_json = res.json()

  # extract activities from json and save as list
  activity_list = []
  # get first element
  if response_json:
    elem = response_json.pop()
  else:
    print('activities response json is empty')
    return None

  while elem['id']:
    # ensure that fields aren't empty
    # activity id
    activity_id = elem['id']
    if activity_id is None:
      activity_id = 'no_activity_id'

  # activity name
  activity_name = elem['activityName']
  if activity_name is None:
    activity_name = 'no_activity_name'
  else:
```

```
 # replace new line to empty space
 activity_name = activity_name.replace('\n', ' ')

# activity type
activity_type = elem['activityType']
if activity_type is None:
 activity_type = 'not_activity_type'

# start and end time
activity_start_time = elem['startTime']
if activity_start_time is None:
 activity_start_time = 'no_start_time'
 activity_end_time = elem['endTime']
if activity_end_time is None:
 activity_end_time = 'no_end_time'

# format the timestamps
start_time_split = activity_start_time.split(".")
activity_start_time = start_time_split[0]
activity_start_time = activity_start_time.replace("T", " ")

end_time_split = activity_end_time.split(".")
activity_end_time = end_time_split[0]
activity_end_time = activity_end_time.replace("T", " ")

# exclude exclusive gateway
if 'no_activity_type' not in activity_type and 'exclusiveGateway' not in activity_type and 'boundaryConditional' not in activity_type:
 activity_entry = instance_id + ';' + activity_id + ';' + activity_name + ';' + activity_type + ';' + activity_start_time + ';' + activity_end_time
 activity_list.append(activity_entry)

```

```
    # if there are more elements, else break loop
    if response_json:
        elem = response_json.pop()
    else:
        break

    return activity_list
```

Listing 2.7: Function to get all activities of a process instance

Finally, the activity lists had to be written to a log file. This is shown in Listing 2.8.

```
# open file and write header
f = open('event_log_insurance.csv', 'a')
f.write('case_id;activity_id;activity;activity_type;start_timestamp;end_timestamp\n')
for process_instance in instances:
    # get all activities of process instance
    activities = get_process_activities(process_instance)
    # write activities to file
    f.write(list_to_string(activities))
f.close()
```

Listing 2.8: Code to write the log file

Now that the event logs have been completed as preparation, the focus in the following is more on process mining.

2.3 Process Mining

Process mining is an approach to *discover*, *monitor* and *improve* processes [48]. In general, software systems execute the business processes. To be able to improve the processes, the actual process is identified with the help of discovery. For this purpose, the software system creates event logs that record all events. A process model can be generated from this event log with the help of process discovery algorithms. If a process model already exists, it can also be checked for conformity with the event log. In this way, it can be checked whether the process runs as it has been planned. The process model can also be extended with the help of the process model and the event log. The process can thus be improved. The three described techniques can be seen in Figure 2.8.

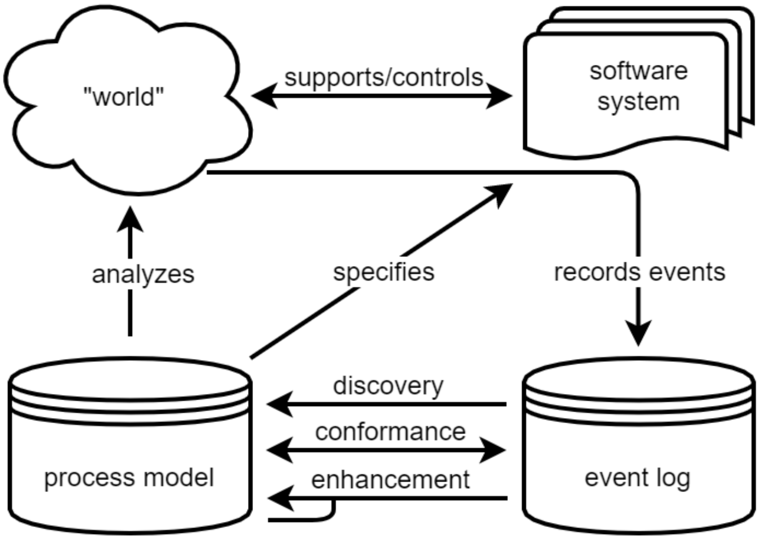

Figure 2.8: Areas of use of process mining, based on [48]

When applying process mining techniques, several aspects should be considered. The *Process Mining Manifesto* provides important guiding principles [48].

- GP1: Event Data Should Be Treated as First-Class Citizens
- GP2: Log Extraction Should Be Driven by Questions
- GP3: Concurrency, Choice and Other Basic Control-Flow Constructs Should be Supported
- GP4: Events Should Be Related to Model Elements
- GP5: Models Should Be Treated as Purposeful Abstractions of Reality
- GP6: Process Mining Should Be a Continuous Process

In order to generate meaningful process models, the quality of the event log is of great importance. Therefore, GP1 emphasises that the event log should be complete, and the events should satisfy well-defined semantics. GP2 emphasises that concrete questions are relevant for a meaningful analysis. Before a process mining technique can be applied, the questions to be answered by the technique should be selected. GP3 presents that different process modelling languages provide different modelling elements, and the process mining techniques should support these. GP4 points out that the starting point of the analysis is the relationship between events in the log and the elements in the model. Therefore, care must be taken to remove ambiguities to interpret the results correctly. GP5 illustrates that there is no perfect representation for process models but emphasises certain things for specific audiences. In this way, different perspectives and levels of interaction can be represented. Finally, GP6 suggests that process mining is not a one-off activity but that processes should be considered continuously.

The event log is crucial for generating the process models, as knowledge can be extracted using process mining techniques. It provides detailed information about the process history [48]. The log contains the execution sequence of a process instance [5]. In other words, the behaviour of the process. Each event in the event log refers to an activity (task) [5]. The event log contains all process instances (cases) with their events. The tasks take time [5]. Therefore, additional timestamps may be included in the event log. Here, both timestamps corresponding to the activity's start or the activity's end are possible.

In the context of process discovery, the terms *noise* and *incompleteness* are often used. Noise describes the problem of rare behaviour in an event log. Rare behaviour is not representative of the typical behaviour of the process.
Incompleteness describes the problem of incomplete event logs. An event log only contains the sequences that have already been executed. Thus, it may not contain all possible sequences of activities. Both problems affect the quality of process models.

2.4 Process Discovery Algorithms

Various algorithms can be used for Process Mining. In this Section, some of these are explained.

2.4.1 α Algorithm

One of the most known Process Discovery algorithms is the α *algorithm* [20]. The α algorithm identifies causal dependencies between activities. From this, a set can be created that is formulated into a workflow net. "The algorithm uses the fact that for many WF-nets, two tasks are connected iff their causality can be detected by inspecting the log." [5]

In [5] Van der Aalst et al. describes four log-based relations to analyze the causal relations. The < relation can be used to represent succession relationships. If activity B is directly after activity A in the log, this can be represented in this way: $A > B$. With → relation activities can be represented, where one activity is a successor of the other, but not vice versa. This means that there is a relation $A > B$, but $B \not> A$. Activities that do not have a successor relation to each other are represented with the # relation, which expressed in a > relation means that $A \not> B \wedge B \not> A$. Parallel activities are defined with the || relation. Here there exists both > relations, i.e. $A > B \wedge B > A$.

The different traces are first identified from the event log to identify the workflow net. These can be derived from the > relation. Then all initial transitions and all final transitions are found. Now a set is defined that contains all tuples of activities that fulfil the dependency conditions. If there is a causal dependency between two activities (transitions), then there must also be a place that connects them [5]. Therefore, from the set of causal dependencies, the set of places can be defined. Finally, the workflow net is formally described as a flow of transitions and places. The net can be generated from this description.

With the help of the α algorithm, a sound workflow net can be discovered based on a complete event log [5].

However, the algorithm also has some problems and limitations [5]. Among other things, each transition needs a unique name, and hidden tasks cannot be detected.

It cannot deal with short loops (loops of length one and two). The α *algorithm* can only be applied if the event log is based on an acyclic sound and structured WF-net. Due to this limitation, the selected scenarios cannot be examined with the α algorithm, as they contain such a loop.

Due to the problems, the α algorithm has been further developed. Among other things, the $Alpha$+ and $Alpha$++ algorithms have been developed [17]. The $Alpha$+ can now also handle short loops [20]. The $Alpha++$ is the most advanced development [17].

2.4.2 HeuristicsMiner

The *HeuristicsMiner* also uses the causal dependencies, like the α algorithm [50]. Based on this, the HeuristicsMiner considers the frequencies of the traces. Rare paths should not be included in the model [4].
First, the frequencies for all possible activity combinations are calculated and can be displayed in a matrix. The frequency between an activity A and an activity B is calculated by subtracting the number of times A follows B from the number of times B follows A and dividing by the sum of these two numbers.
Various threshold parameters can be used to modify the relevance of relationships [50]. In this way, the HeuristicsMiner can deal with noise. In order to be able to deal with short loops of lengths one and two, a frequency table is set up here. Loops of length 1 are loops in which an activity occurs several times. This means that the frequency must be calculated of how often activity A occurs after activity A. For loops of length two, two activities are repeated. This needs special treatment because the pair repeats.

The HeuristicsMiner can also recognise AND/XOR-split/join constructs, and no explicit modelling of non-observable tasks is done. For this purpose, a causal matrix is generated that represents the input and output expressions. In the last step, dependencies are mined depending on a decision in another part of the model.

The HeuristicsMiner uses a representation similar to Causal nets [4]. In conclusion, the HeuristicsMiner approach is robust due to the representational bias provided by Causal nets and the usage of frequencies [4].

2.4.3 Inductive Miner

The idea behind the *Inductive Miner* is based on the recursive splitting of the event log [4]. The approach is based on the use of process trees because they are sound by construction.

Four different types of cuts can be made for the split. *Exclusive-choice cuts*, *sequence cuts*, *parallel cuts*, and *redo-loop* cuts are considered. In order to perform an exclusive-choice cut, there must be no direct succession relationship between the activities of the different groups. There must be only one successor relationship between the subsets for each activity contained for a sequence cut. For the parallel cut, the subsets are considered. There must be a succession relationship in both directions. In addition, each subset must have a start and an end. In order to understand the redo-loop cut, it is helpful to imagine the process tree. The left subtree is called do-part, and the right subtree is called redo-part. If the redo part is executed, another do part must follow. After the do-part, whether the loop is executed again or continues to the end is checked. The following preconditions for the cut can be read from the formal description. If there is a link between an element from the do part and the redo part, this element must also end. Analogously, if an element from the redo part has a link to the do part, this element from the do part must also have a start. At the same time, all end elements must lead to the same element from the redo part. Furthermore, the elements from the redo part must not have any successor relationships to other subsets.

A directly-follows graph can be created for the event log. This graph is examined to see which cut can be performed. A cut divides the event log into smaller sub-logs, for which a direct-follows graph is created, and a cut is performed. The procedure is recursive, and cuts are made until the event log consists of only one activity.

The process tree can then be represented with the four operations $(\times, \rightarrow, \wedge, and \circlearrowleft)$. The advantage of this structure is that a sound process model is always produced and can easily be converted into other process visualisations such as Petri nets and BPMN models. There is a formal guarantee, which ensures fitness. The Inductive Miner can handle rare behaviour and large models. However, even the Inductive Miner has problems. If the process tree contains duplicates and silent activities, the algorithm may produce an underfitting model. Furthermore, it cannot handle loops of a fixed length.

2.4.4 Fuzzy Miner

In contrast to the process discovery algorithms considered so far, the fuzzy miner has a different approach. It also considers unstructured data ("spaghetti-like"), with which many algorithms have problems, and uses the concept of a roadmap [25]. For this, he does not try to map the behaviour found in the protocol to typical process design patterns but concentrates on a high-level mapping of the found behaviour [24]. First, all events discovered in the protocol are represented as activity nodes. Through unary significance, their importance can be expressed. For each precedence relation, a directed edge is added. Through various transformation methods, the model can now be successively simplified concerning certain aspects. First, conflicts (loops) are resolved, and the edges are treated. Edges can either be removed or clustered. Edges that do not correspond to the correct behaviour must be discarded. Clusters are also formed for activities. Rare behaviour that is below a certain threshold is clustered or abstracted.

2.4.5 Split Miner

The Split Miner is used to create a BPMN model, which requires five steps [8]. First, a directly-follows graph (DFG) is constructed. The DFG is not filtered directly but analysed to detect self-loops and short-loops. A self-loop, for example, is recognised by the fact that it has an arc to itself. The loops are removed from the DFG and only restored at the end to create the BPMN model. In the second step, concurrency relations are detected, and the corresponding edges are removed. The result after this step is a pruned DFG. In the next step, a special filter algorithm is applied. The filter algorithm guarantees a sound process model, a minimum number of edges and maximises fitness. In the last two steps, the split gateways and join gateways are added to obtain a BPMN process model from the DFG. A split gateway is recognised by the fact that it has one incoming edge and many outgoing edges. A join gateway has many incoming edges and only one outgoing edge. Finally, it can be emphasised that the BPMN model discovered by the split miner is sound and does not contain deadlocks. It has a complexity of $O(n+m^4)$. Compared to other discovery algorithms, it stands out as very powerful [7].

2.5 Process Mining Tools

The algorithms described in Section 2.4 are used in various process discovery tools. This Section presents various process mining tools that have been used in this thesis. Table 2.1 gives an overview of the process mining tools considered. The selection takes different criteria into accounts, such as supported formats for the event logs, algorithms, and different types of process visualisations.

	Celonis Snap	**Disco**	**ProM**	**Apromore**	**PM4Py**
Supported formats for event logs	CSV/XLSX, Google Sheets, XES	various, e.g. CSV, XES, XLS/XLSX	CSV, XES, MXML	various,e.g. CSV, XES, BPMN, MXML	CSV, XES
Process visualisations	process map	process map	various, e.g. Petri net	process map, BPMN	process tree, Petri net, BPMN
Algorithm	confidential	Fuzzy Miner	various, e.g. α algorithm	Split Miner	α, α+, HeuristicsMiner, Inductive Miner
Further analysis	yes	yes	yes	yes	yes
Access	commercial	commercial	open-source	open-source	open-source

Table 2.1: Overview of process mining tools used

2.5.1 Celonis Snap

Celonis Snap is a free cloud process mining platform from Celonis [14]. Not only is it intended to be the accessible version of the *Intelligent Business Cloud (IBC)*, but Celonis sees Snap as an entry-level tool for process mining. The Snap tool supports process discovery by uploading event logs with standard file formats such as CSV and XES and other file formats like Google Sheets. In addition, data can also be imported from ERP systems such as SAP. The generated process model is visualised as a process map. The algorithm used for discovery is confidential.* In addition to discovery, further analysis can be carried out with a focus on business results [14]. For example, customer satisfaction and operating costs can be monitored, and risk reduction is achieved. Conformance checking and process enhancement are also

*In response to an enquiry, the software vendor replied that it does not disclose the underlying algorithm.

supported. With conformance checking, Snap can automatically evaluate how well the process conforms to the ideal path. New friction points can be identified and corrected to improve the process.

In event logs uploaded using the standard CSV file format, only the standard columns *case id*, *activity* for the activity name and a timestamp *eventtime* can be used in the Snap Tool. Other columns in the event log can only be used for sorting means that the selected column is used instead of the column for the activity name. Since the event logs of the scenarios contain both a start timestamp and an end timestamp, a suitable column has to be selected here. Since the start event and the first activity have the same start timestamp, the choice of this timestamp harms the resulting process model. The end timestamp of the two differs. Therefore, the end timestamp is used.

Figure 2.9 shows an excerpt from the discovered vaccination process model to give an impression of the representation in Celonis. The complete model is included in Appendix B. The discovered insurance process model is also available in the Appendix B.

In general, it is noticeable that the process model runs from top to bottom, and the process start and end are marked with an additional symbol with a label. The number of process instances is indicated both on the arrows and in the activities. The number can also be roughly recognised by the colour highlighting. Paths and activities that occur more frequently are shown in a darker shade of blue and with thicker arrows. Paths and activities that occur less frequently are shown in a lighter shade of blue. In the first scenario, the vaccination process shows that the two waiting activities are not distinguishable. Therefore, they are analysed as a loop. In the second scenario, the insurance process, it can be seen immediately after the start that two sequence flows begin here. The left sequence flow shows the standard process flow. The right one shows the sub-process that is started when an application is rejected.

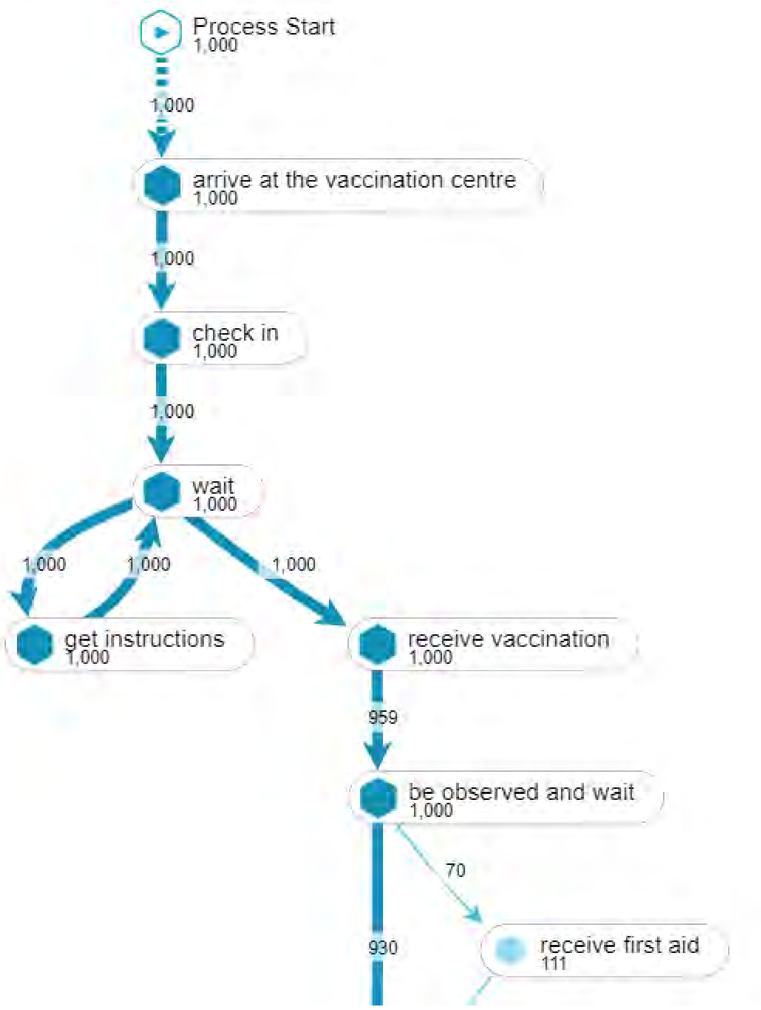

Figure 2.9: Excerpt of process model in Celonis

2.5.2 Disco

Disco is a process mining tool from Fluxicon for process discovery [23]. A licence is required to use Disco, but employees and students of partner universities can obtain it free of charge [18]. The tool specialises in process discovery and can deal with everything from reading in an event log to performing various analyses, and process automation [19]. Various formats can be read in an event log, including CSV, XES, MS Excel, and MXML files. The resulting process visualisation is shown as a process map. The algorithm used is based on the fuzzy miner but has been further developed. Further statistics can be created based on the process map.

Information about the number of instance variants or the frequency of activities can be retrieved. On the other hand, statements about performance can also be determined. With the help of filters, the focus can be set, for example, on a specific variant or the performance. By animating the process, the process flow can be visualised over time.

When uploading a CSV file, one has the option of assigning the columns to the types *Case* for the instance id, *Activity* for the activity name, *Timestamp* for the process time, *Resource* for indicating who processed the task and *Other* for all other columns. Analogous to Celonis Snap, only one column could be selected for the timestamp. Due to the same problem, the end timestamp is selected, and the start timestamp is set as Other.

Figure 2.10 shows an excerpt of the discovered vaccination process model to give an impression of the representation in Disco. The complete model can be seen in Appendix B. The discovered insurance process model is also included in Appendix B.

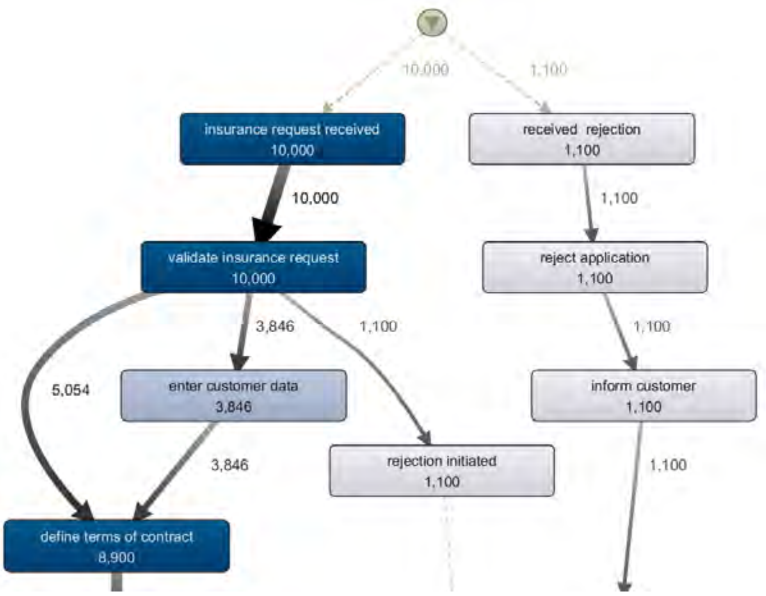

Figure 2.10: Excerpt of process model in Disco

In the process models in the scenarios, a certain similarity to the previous process visualisations of Celonis Snap can be seen. The process also runs from top to bottom, and the process start and end are marked separately. However, there is no textual representation here. Instead, there is a colour marking (start = green and end = red). The blue colour gradations and thickness of the paths for the frequency of the processed process instances can also be seen. The number of frequencies is also included in the arrows and the activities. The loop in the vaccination process described for Celonis Snap also occurs here. Furthermore, the direct splitting of the sequence flows in the insurance process can also be seen here. Thus, the results of Celonis Snap and Disco are very similar.

2.5.3 ProM

ProM is an open-source framework with a lot of different algorithms that could be used by installing corresponding plugins [39]. The ProM framework is based on packages that have been developed as plugins by various companies, and universities [49]. Event logs could be uploaded in various file formats such as CSV, XES and XML.
By using different plugins, different process visualisations can be generated. These include Petri nets, Heuristic nets and BPMN models. Several algorithms are available for this, including the α algorithm and HeuristicsMiner. The plugins can be used in various ways to make further analyses, such as conformance checking, possible. It also offers conversion options, for example, to transform a Process Tree into a Petri net.

If a CSV file is uploaded, it can be converted into the XES format. The first step is to set the column separation and the character encoding used. In the next step, the mapping to the standard is carried out. The column corresponding to the case column, the event column and the start and end time is specified. The converted file can now be used to execute various algorithms from the available plugins.

In summary, the tool is suitable in an academic environment to try out different algorithms and visualisations. However, the handling in other tools is more convenient for practice.
Earlier versions of ProM support the generation of C-nets. However, the current version of ProM uses the flexible HeuristicsMiner, which only supports Heuristic nets and dependency graphs instead of C-nets [51, 11]. ProM is not used to discover the process visualisations for the study because the visualisations are too unwieldy for this. Besides, enough visualisations have been created by the other tools.

2.5.4 Apromore

Apromore is an open-source online process mining platform [6]. For process discovery, event logs in standard file formats and many other formats can be used. Apromore supports exploratory data analysis by allowing the user to inspect, analyse and visualise event logs interactively. Process maps and BPMN models are supported as process visualisations. They allow users to analyse the direct following relationship of activities [6]. Apromore is using the Split Miner to discover BPMN models from event logs [7].

Not only can the "as-is" process be discovered, but other views can also be displayed to explore resources and roles. Further analyses can also be carried out, for example, to determine the frequency and duration of activities or to create filters for specific execution paths. In addition to discovery, animations of the various process flows can also be viewed. In the area of redesigning, models can be mixed, and similar models can be detected.

In Apromore, the event log can be uploaded in CSV or XES format. Besides the case ID, many other types can be set for the columns. The attributes are differentiated between case attribute and event attribute. The case attribute is the same for each case event, and the event attribute changes during the execution of a case. Because of these attributes, an ID and a name can be specified for the activity. In addition, it is possible to specify both the start time and the end time of the activity.

Figure 2.11 shows an excerpt of the discovered insurance process model to give an impression of the representation in Apromore. The complete model can be found in Appendix B. The discovered vaccination process model is also included in Appendix B.

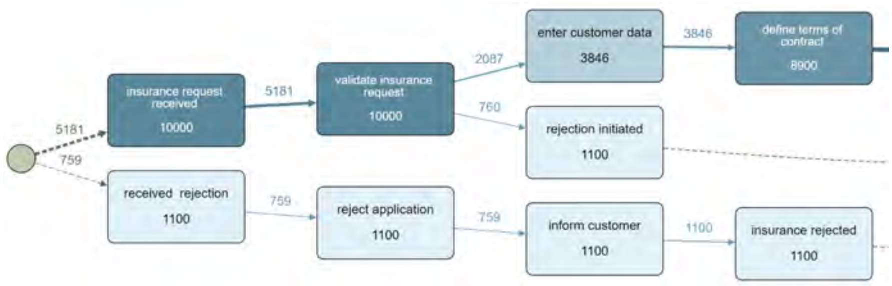

Figure 2.11: Excerpt of process map in Apromore

When displaying the discovered models for the defined scenarios, it is noticeable that the models' activities are displayed horizontally from left to right. The start event (green) and the end event (red) are highlighted in colour. The blue gradations for the frequency of activities and paths from dark to light are also used here. The frequency is additionally specified on the arrows and in the activities.

Interestingly, the BPMN process models are not represented in the block structure. Finally, it is also noticeable that the wait is shown as a loop for the vaccination process, and in the insurance process, the sub-process is shown directly parallel to the standard path.

2.5.5 PM4Py

PM4Py is a process mining framework initiated by the Process Mining Group of the Fraunhofer Institute for Applied Information Technology [21]. PM4Py chooses a new approach. Instead of a graphical user interface, PM4Py provides process mining libraries written in Python and, in addition, integrates state-of-the-art data science libraries [10]. In [10] it is shown that there are also limitations in dealing with large experimental settings through the use of user interfaces. One severe limitation with commercial tools is that one cannot freely choose the desired discovery algorithm.

For process discovery, files can be imported as XES and CSV files. Different process mining algorithms and process visualisations can be selected. For instance, the HeuristicsMiner may be used. It creates a Heuristics net that contains the activities and the relationships between them. The Heuristics net can be then converted into a Petri net [20]. PM4Py provides factory methods to use the following discovery algorithms: α, $\alpha+$, HeuristicsMiner, and Inductive Miner [20]. Petri nets, process trees and BPMN diagrams are possible as process visualisations. Since the models can only be generated directly as image files (e.g. PNG), direct process analysis is not possible, as it is in the previous tools. Therefore, analyses must be implemented in Python itself. Analyses can be carried out, for example, on the duration of cases, waiting times, and the elapsed time between different activities. PM4Py also supports conformance checking. In addition, a simulation of Petri nets is possible. Finally, both decision mining and social network mining are supported.

PM4py is also used in the study. The different process models for both scenarios that are discovered with the help of PM4Py can be seen in the Appendix B.

In the process models discovered, it is noticeable that the process steps shown run horizontally in Petri nets and BPMN models and from top to bottom in Heuristic nets. Start events are shown with a green circle and end events with an orange circle. The number of instances is only shown on the arrows and in the activities in the Heuristic net. Here a tiny colour gradation of the activities can be seen. The arrows remain the same. However, the number of instances is not displayed in Petri nets and BPMN models.
Process discovery has been tried with the α algorithm, the HeuristicsMiner and Inductive Miner. Due to the loop in the processes, the α algorithm could not be used. The models of the vaccination process obtained by the HeuristicsMiner and the inductive miner look identical. In contrast, more differences are observed in the vaccination process. Here, the Petri net created by the inductive miner is more general, allowing for many more different traces than initially intended. The algorithm's approach can explain this. The Petri net discovered by the HeuristicsMiner corresponds to the desired scenario.

2.6 Eye-Tracking with Pupil Core

Eye-tracking is a helpful tool to detect and analyse eye movements [9]. Both the points of gaze and the path of movement can be recorded [9]. The duration of the gaze on a point is also recognised. The recorded data can then be visualised with a suitable tool. The following sections explain the basic terms and how eye-tracking works in general. The tool used in the study is then presented.

2.6.1 Functionality

When recording eye movements, not only the movements but also the points of gaze can be recorded. The gaze points are called *fixations* and contain the points at which the eye looks "stably" at an object [43]. A person's eye is constantly moving, even if it appears stable [9]. A fixation typically lasts at least 100ms and can thus be identified based on the viewpoints in a given area [41]. The duration of fixation shows the time of paying attention to a specific object [9]. It may depend, among other things, on whether the object is interesting or challenging to understand. *Saccades* are the rapid jumps from one fixation to the next [59]. These are helping the eye to piece together the points it sees [9]. The directed sequence of gaze points is called *gaze path* when a directed sequence of gaze points is meant. The visualisation of this path enables a representation of the order a person looked at objects on a screen.

To be able to capture this data, an eye-tracker contains cameras that record the eyes [9]. The camera uses a reflection on the cornea to identify the pupil [9]. There are different technologies for how eye-tracking systems work [1]. A distinction is made between bright and dark pupil technology, which can be seen in Figure 2.12.

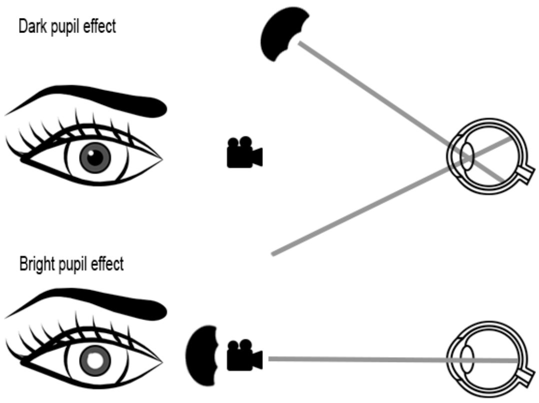

Figure 2.12: Eye-tracking techniques, based on [45]

Bright pupil technology uses an infrared source similar to the "red-eye" effect. The pupil is displayed as a white circle, which makes it detectable. The technique used in this thesis is based on dark pupils. An infrared source is used to illuminate everything except the pupil. This way, the image processing system knows that the darkest and round area is the pupil. What the dark pupil looks like can be seen in Figure 2.13. The advantage of this technique is the more robust behaviour in different light conditions.

Eye-tracking can be used to derive insights into comprehensibility [27]. A higher number of fixations indicates a poorer arrangement of elements, requiring more effort to explore. If the fixations are further apart, this indicates that a person is conducting a targeted search. Longer fixations mean more effort to understand, as more time and effort must be spent to understand the visual stimulus. Several metrics can be used to examine fixations, saccades, and scan paths [59]. Metrics performed based on fixations refer to either the number or duration of fixations. In [59] an overview of a variety of metrics that have been used in studies can be obtained.

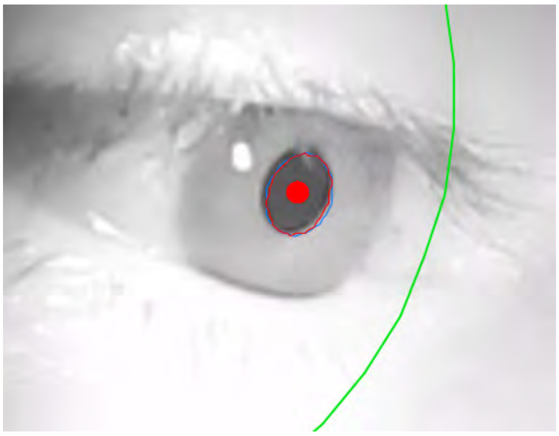

Figure 2.13: Eye-tracking: dark pupil

2.6.2 Pupil Core with Pupil Capture

The information about fixations, saccades, and gaze paths can be used to explore the user experience [9]. For this work, the eye-tracking glasses *Pupil Labs Core-Headset* have been used. Figure 2.14 shows the glasses. The glasses have two eye cameras that capture the pupils and a scene camera to capture the wearer's field of view [31]. The glasses can be easily worn and connected to the computer via a USB port.

An open-source software suite is required to use the glasses. It works on Windows 10 as well as on Mac OS and Linux. The software used is divided into two functional areas: capturing and visualising. With the help of *Pupil Capture*, the gaze data can be captured and processed in real-time [31]. The recordings are saved in folders as mp4 files. Further data (e.g. the individual positions of the fixations) are saved in CSV files. With the help of *Pupil Player*, this data can then be replayed and visualised [31].

Figure 2.14: Pupil Labs Core-Headset

"Pupil Core's fixation detectors implement a dispersion-based method." [40]. The different angles of the pupil can detect the dispersion and how many points are in one area.

3 Related Work

In this thesis, process models discovered by process mining will be analysed for comprehensibility. This Chapter looks at various works that have also dealt with the analysis of comprehensibility.

The basics of process mining have already been presented in Section 2.3. Process mining aims to identify and improve processes. The findings are based on the Manifesto, which presents the principles of process mining [48]. Various algorithms and tools have already been presented in sections 2.4 and 2.5.

Comprehensibility plays a significant role in computer science. Many types of text need to be understood. The first thing that comes to mind here is the source code. However, documents like specifications also need to be understood. These form the basis of the contract between a client and a contractor.
What is meant by comprehensibility is not easy to define [32]. Psychological research into comprehensibility goes back to the 1970s and considers four dimensions of text comprehensibility: linguistic simplicity, semantic redundancy, structure-order, and motivational stimulation [15]. For linguistic simplicity, a distinction is made between word difficulty and sentence difficulty. The empirical investigation of word difficulty results can be summarised: familiar words can be processed more quickly. In addition, a high technical word density leads to a slowing down of reading, and the text then becomes more difficult for the reader to understand. Empirical studies on the difficulty of sentences have identified several factors that affect comprehensibility. These include embedded overlong sentences with several sub-clauses, sentences with much information and syntactically ambiguous sentences. Less research has been done on the second dimension, semantic redundancy. According to [32], it can be assumed that a repetition of essential text aspects can increase comprehensibility. In the dimension of structure and order, the structuring of texts in terms of content is considered. It is proven that texts should follow a hierarchical-

sequential structure and that the level of abstraction shall gradually become more detailed. For text comprehension, a relationship must be established between sentences and text sections. There are signal words, such as "therefore" and "leads to", which establish content-related references between sentences and thus increase comprehension. Motivational stimulation looks at how curiosity and interest can be used to create a motivating text. One should proceed sparingly for a motivating text design since an enrichment of interesting but unimportant details does not improve comprehension and even harms overall retention. Thus, motivational stimulation has no direct effect on the comprehension of a text but can keep attention high.

Although a large number of empirical studies have already been carried out on the four dimensions, the research has reached its limits [15, 32]. There is no single optimum for all readers. Subjective factors such as prior knowledge, expectations, and interests influence text comprehension. However, a text can be adapted to a specific target group to increase comprehensibility for them.
In computer science, analysis and design models based on (semi-)formal languages like UML are often used because they are more precise than natural language. For UML modelling, some research has already been done on the comprehensibility of these models [38, 60].
In [60], features of UML class diagrams are identified by which a task is most effectively supported. These include features such as layout and colour. The participants of the study have to perform various tasks to understand the class diagram. An eye-tracker is used for analysis purposes, which helps make navigation observations in UML class diagrams. For example, novices explore the diagram from top to bottom and from left to right, while experts explore from the middle of the diagram outwards. They can also observe a positive effect when using additional information (such as the use of colour). However, if a participant can not answer the question, the answer is still close to the correct answer by using this additional information. Finally, it is noticeable that similar visually presented notations (such as aggregation relationships and generalisation) can reduce comprehension.
Porras examines the comprehensibility of UML diagrams and compares them with other representations [38]. An empirical study is conducted on the understanding of design patterns. Participants have to complete various tasks required for understanding design patterns while wearing an eye-tracker that records their eye movements. The experiment can determine which representations are most efficient and

most likely to confuse the participants. However, they find that specific notations are more suitable for tasks than others. Thus, they motivate those different notations to be supported in tools to accommodate all tasks.

Methodologically, eye-tracking is often used to investigate comprehensibility [27]. Eye movements can be analysed with the help of an eye-tracker, allowing interesting points or difficult-to-understand diagram elements to be identified. The basics of eye-tracking and metrics on how to use it have already been presented in Section 2.6.

Also, in the field of process visualisations, some studies on comprehensibility have already been conducted with the help of eye-tracking. One of the first eye-tracking studies to measure user satisfaction in business process modelling investigates in [29] among other things, the comprehensibility of two different ways of presenting EPCs (eEPC and oEPC) in an eye-tracking experiment. The participants are asked to evaluate the modelling languages subjectively and perform tasks such as modelling a process or detecting errors. Comprehension, completeness, and ease of use are identified as the essential requirements for process modelling languages. The study confirms that eye-tracking is a suitable method for measuring and assessing user-related requirements concerning user satisfaction in process modelling.

In [58] the understanding of business process models is looked at from a different perspective. They introduce a task perspective. In experiments on the comprehensibility of process models, they are often examined using comprehension questions. A study is conducted in which participants are shown a BPMN model and asked to answer true-or-false comprehension questions. The eye movements are recorded with an eye-tracker. Petrusel finds a correlation between how a respondent inspects the process model and the answers' performance. His analysis highlights the predictive power of the independent variables of the number of fixations and the time it takes a participant to fixate on a model element.

In [53] and [55] experiments are conducted in which the influence of expertise on process model understanding is investigated. In the first experiment, this is done by using process models modelled with BPMN [53]. In the second experiment, EPCs are used, and a comparison of the two papers is made [55]. The design of the studies is analogous. The participants are each shown three process models of different difficulty levels. In the model on the first level, only basic elements of the

modelling language are used, and as the difficulty increases, more elements and more tasks are added. The participants are divided into two groups depending on their experience with process models. The participants are asked to understand the models and then (when the model could no longer be seen) to answer true-or-false comprehension questions. During the experiment, eye movements are recorded using an eye-tracker.
In both experiments, it is noticeable that performance drops off as the difficulty of the process models increases. In [55] it is also noticeable that the performance of beginners and advanced learners converges with increasing difficulty. Overall, the performances of this second study are better than those of the first, which leads the authors to assume that EPCs are easier to understand than BPMN models. However, this still needs to be investigated through further research. The procedure of participants looking at process models while eye movements are recorded with the eye-tracker and true-or-false comprehension questions have to be answered adopted for this thesis.

In [54] a study is presented in which the comprehensibility of process models in four different modelling languages (BPMN, eGantt, EPC, and Petri net) is investigated with the help of an eye-tracker. The participants are presented with 12 different process models (three models for each modelling language). The models from one modelling language are divided into three levels of difficulty, which means that more elements are added to the basic elements of the modelling language with increasing difficulty. For each process model seen, the participants have to answer true-or-false comprehension questions without looking at the model again. Performance drops with increasing difficulty, and participants take longer to answer. In addition, Zimoch et al. can gain further insights, which are divided into nine lessons. First, they describe that the choice of the process scenario influences the cognitive load. Thus, the load is lower if the person is familiar with the scenario. Otherwise, the person first has to get an overview and understand the scenario. This thesis uses two scenarios (vaccination process and insurance process), paying attention to possible familiarity. In times of the Corona pandemic, the process in vaccination centres is familiar to many, and this scenario is not expected to be a significant burden. Taking out insurance is also commonplace but might not yet be so familiar depending on the participants' experience. The second lesson shows that process models with simple and clear information are easier to understand. In the experiment, models

are looked at element by element during the first viewing, and different strategies are used for further understanding. Here it is interesting to see if this is also observed in the study conducted in this paper. It is noted that further work needs to investigate the extent to which the structuring of a process model (e.g. block structure) influences its comprehensibility. In this thesis, process models are shown to be both horizontal and vertical. It will be interesting to see if there is a difference in performance in this thesis. In the next lesson, the results of the comprehension questions are compared using the different difficulty levels and modelling language. Here, for the modelling languages BPMN, EPC, and Petri nets, a decrease in the answer score with increasing difficulty is observed in each case. The difficulty of the insurance process is greater than that of the vaccination process. Therefore, in this thesis, it is expected that the response score for process models of the insurance process will be lower than for the vaccination process. It is described that participants initially search for the start of a model. Therefore, those process models with an explicit start and end symbol are easier to understand. Since the start events of the generated process models in this thesis are either on top (for vertical process models) or on the left (for horizontal process models), it is expected that the start is found quickly. In the next lesson, it can be seen that identifying, for example, XOR and AND gateways are very challenging. Only XOR gateways are used in this thesis, so participants are expected to recognise which kind is meant from the semantics.
In the study conducted, it is noticed that contrary to expectations, more experienced participants are not significantly more efficient than under-experienced participants. In summary, performance in both groups drops equally as the difficulty of the model increased. It remains to be seen whether an observation regarding performance differences between different modelling languages can be made in this thesis.

In Chapter 3 some work is presented that investigates the comprehensibility of process models. Different visualisation types (including EPC and BPMN) are compared, and both simple and more complex models are considered. The experiments are supported by the use of an eye-tracker. The structure of the investigation in this thesis is similar, as different process modelling types are also compared here with the help of an eye-tracker. However, in previous studies, the models are created by hand. In contrast, this thesis will examine models that have been generated. It will be interesting to see if this reveals any differences from the results obtained so far.

4 User Study

The previous chapters show that generated process models by process mining are essential for analysing and improving business processes. Therefore, these process models must be well understood. A user study is conducted to investigate the generated process models' comprehensibility, which is described in this Chapter. First, the context of the study is described. Then follows the experimental setting with the research question. Next, the hypotheses to be tested in this experiment are presented. These are followed by the experimental set-up, which describes the participants, the dependent and independent variables and the materials used. Finally, the mode of operation and data validation are presented.

4.1 Context of Experiment

In recent years, many research results concerning the comprehensibility of process models have been published [53, 29, 58]. Research to date has focused on different process modelling languages, models of varying complexity and the influence of user experience (see Chapter 3). Research methods based on eye-tracking are used to investigate the cognitive processes involved in understanding process models. Concepts from cognitive psychology are used to determine the cognitive load and the level of acceptability for different process visualisations. Research in the area of cognitive psychology suggests that our working memory is limited and that instructional methods should avoid overloading the memory to maximise learning [44]. This thesis uses these concepts from previous research and focuses on the new aspect of generated process models. The goal is to compare manually created and generated process models critically and different generated process visualisations.

The procedure of the experiment is based on the typical steps as presented in [52]. According to Wohlin et al., the first task is scoping to clarify what is being investigated. Then the experiment is planned and defined. The planning includes determining the environment in which the experiment is conducted, defining the hypotheses, and the dependent and independent variables. In the operation step, the experiment is prepared, commit the participants, and then the experiment is carried out. The raw data is then analysed and interpreted. Finally, the results can be presented in a report.

4.2 Experimental Setting

The user study investigates the comprehensibility and cognitive load of understanding process models generated by process mining. So far, it is unclear whether the type of process creation (whether the process model is modelled by hand or generated from an event log) influences the comprehension of the process model and which visualisation type is more comprehensible concerning generated process models. The following research question formulates this approach, which has not yet been investigated. More specific sub-questions complement the research question.

Research Question

How comprehensible are process models generated by process mining?

Sub-questions:

- Are the manually created process models preferred over the generated models?
- Which representation of the generated process models is considered as the most comprehensible?
- What factors influence the participants in naming the most and least comprehensible process model?

An eye-tracking experiment is conducted to test the research question. According to [52] an experiment is defined as "an empirical enquiry that manipulates one factor or variable of the studied setting". More specifically, it is a human-oriented experiment in which people randomly perform various treatments to objects in a laboratory setting [52]. Eye-tracking has proven to be a useful method for studying comprehensibility and is also used for this experiment. Eye-tracking can detect eye movements and thus show how participants view the process model to understand or learn it. From the collected eye-tracking data, it is possible to determine, among other things, the number of fixations, which forms the basis for the following analysis on the experiment. These measured values are then used to formulate hypotheses and test them.

4.3 Hypothesis Formulation

The following hypotheses can be derived based on the research questions. The hypotheses are defined according to the scenarios.

Hypothesis 1 (Vaccination Process)
H_1: *The manually created process model and the generated models are similarly understandable.*

> $H_{0,1}$: *Viewing the manually created process model takes the same amount of time as viewing the generated process models.*
> $H_{1,1}$: *Viewing the manually created process model takes significantly less or more time than viewing the generated process models.*
> $H_{0,2}$: *Viewing the manually created process model requires the same number of fixations as viewing the generated process models.*
> $H_{1,2}$: *When looking at the manually created process model, significantly fewer or more fixations are measured than when looking at the generated process models.*
> $H_{0,3}$: *The same number of questions are answered correctly in the manually created process model as in the generated process models.*
> $H_{1,3}$: *Significantly fewer or more questions are answered correctly in the manually created process model than in the generated process models.*

$H_{0,4}$: *Answering the questions takes the same amount of time with the manually created process model as with the generated process models.*
$H_{1,4}$: *Answering the questions takes significantly longer or shorter with the manually created process model than with the generated process models.*

Hypothesis 2 (Vaccination Process)
H_2: *The generated process models are similarly comprehensible.*

$H_{0,1}$: *Viewing each generated process model takes the same amount of time.*
$H_{1,1}$: *Viewing one generated process model takes significantly less time than viewing the other generated process models.*
$H_{0,2}$: *Viewing the generated process models requires the same number of fixations.*
$H_{1,2}$: *The number of fixations is significantly lower for one generated process model than for the other generated process models.*
$H_{0,3}$: *The number of correctly answered questions is the same for each process model.*
$H_{1,3}$: *The number of correctly answered questions is significantly higher for one process model than for the other generated process models.*
$H_{0,4}$: *The processing time is the same for all generated process models.*
$H_{1,4}$: *The processing time for one generated process model is significantly less than for the other generated process models.*

The two hypotheses above are made in the vaccination process because it is a simple and sequential process. Therefore, no significant differences are expected between the manually created process model and the generated process models or between the generated process models.

Hypothesis 3 (Insurance Process)
H_3: *The manually created process model is easier to understand than the generated process models.*

$H_{0,1}$: *Viewing the manually created process model takes less time than viewing the generated process models.*
$H_{1,1}$: *Viewing the manually created process model does not take significantly less time than viewing the generated process models.*
$H_{0,2}$: *Fewer fixations are measured when viewing the manually created process model than when viewing the generated process models.*

$H_{1,2}$: When looking at the manually created process model, significantly not fewer fixations are measured than when looking at the generated process models.

$H_{0,3}$: The number of correctly answered questions is higher for the manually created process model than for the generated process models.

$H_{1,3}$: The number of correctly answered questions is not significantly higher for the manually created process model than for the generated process models.

$H_{0,4}$: The time required to answer the questions is less for the manually created process model than for the generated process models.

$H_{1,4}$: The time needed to answer the questions is not significantly less for the manually created process model than for the generated process models.

Hypothesis 4 (Insurance Process)

H_2: The generated process models are similarly comprehensible.

$H_{0,1}$: Viewing each generated process model takes the same amount of time.

$H_{1,1}$: Viewing one generated process model takes significantly less time than viewing the other generated process models.

$H_{0,2}$: Viewing the generated process models requires the same number of fixations.

$H_{1,2}$: The number of fixations is significantly lower for one generated process model than for the other generated process models.

$H_{0,3}$: The number of correctly answered questions is the same for each process model.

$H_{1,3}$: The number of correctly answered questions is significantly higher for one process model than for the other generated process models.

$H_{0,4}$: The processing time is the same for all generated process models.

$H_{1,4}$: The processing time for one generated process model is significantly less than for the other generated process models.

The third hypothesis is based on the assumption that most of the participants are familiar with the notation of BPMN. A known notation could lead to a better understanding of the process. It is also expected that there will be problems in understanding the generated process models, as the branching for the sub-process might be unclear. The fourth hypothesis examines the different generated process models of the insurance process for possible differences.

4.4 Experimental Set-up

An eye-tracking experiment is conducted to test the various hypotheses. In this Section, the further planning of the experiment is discussed and the participants and objects, as well as the dependent and independent variables, are presented. Finally, the procedure is described, and the materials required are mentioned.

4.4.1 Participants

Due to the current Corona pandemic, the acquisition of participants is complex, as the study is conducted locally. Therefore, students, former students and doctoral students of the University of Ulm are invited as participants. Participation is voluntary, and participants are assured that anonymity is guaranteed based on data protection. When invited, participants are informed that this experiment uses an eye-tracker, and the comprehensibility of process models is examined. Therefore, it is pointed out that some experience with process models should be available. The participants should know what a process model is and how it looks like. However, a great deal of experience with process modelling is not required.

4.4.2 Objects

Each participant is shown a total of eight process models, four models from each scenario (see generated process models in Appendix B). Throughout the experiment, the participant wears an eye-tracker that records eye movements. The participant is asked to understand the process models semantically and take as much time as needed. Once the participant feels that the process model is understood, the screen is to be clicked with the mouse. The process model disappears, and the participant is asked to answer three true-or-false comprehension questions (see Appendix C). Based on the answers to these questions, it is to be determined whether the process model is interpreted correctly.

4.4.3 Independent and Dependent Variables

The experiment considers different performance indicators. The corresponding research model for the user study conducted is described in Figure 4.1.

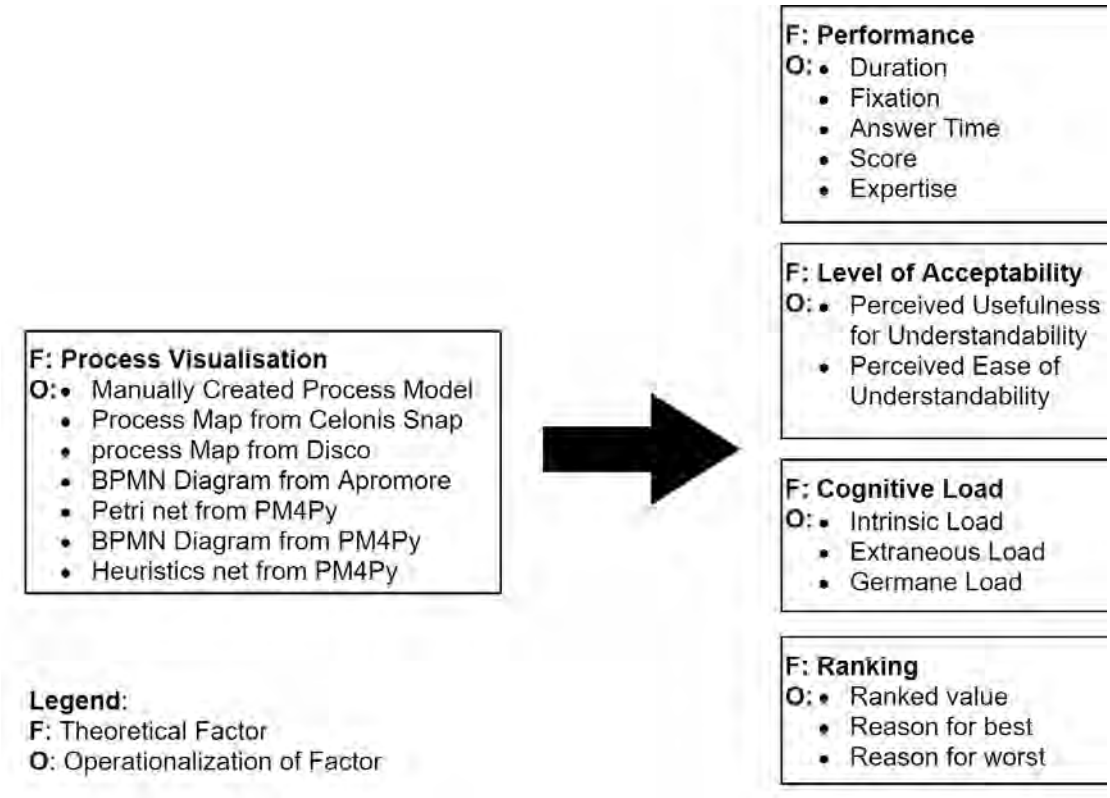

Figure 4.1: Research model for user study

The independent variable process visualisation is shown in the box on the left side. The study compares the process visualisations mentioned here and examines their comprehensibility. On the right side the boxes show the dependent variables that are examined for each process model. These are further described below.

Performance

While the participants are performing the study, a recording is made using the eye-tracker. Thus, the timestamps can be used to determine the duration that is needed by the participants for understanding the process model. The response time is recorded the same way. Fixations are gaze points the participant focus on. They are also collected with the help of the eye-tracker. For each process model, three true-or-false comprehension questions are asked (see Appendix C). The score is derived from the correctly answered questions.

Level of Acceptability

With the help of the level of acceptability, statements can be made about how well the participants accept a process model. For this purpose, a total of eight questions are asked on a 5-point Likert scale from strongly disagree (i.e., 0) to strongly agree (i.e., 5) (see Appendix C). The first four questions examine perceived usefulness for understandability (PUU), and the other four questions examine perceived ease of understandability (PEU).

Cognitive Load

Cognitive load describes the required capacities of working memory during a task. Here, a distinction is made between the three categories intrinsic (ICL), extraneous (ECL), and germane cognitive load (GCL), which are additive [36]. The intrinsic cognitive load is provided by the interactivity of the elements of the learning material. The required load can be influenced by the number of elements. By omitting some interactive elements, for example, the intrinsic cognitive load can be reduced. Extraneous cognitive load is the load imposed by the way the material is presented. Depending on how the information is shown to the learner, the required load can be reduced. Since the two categories are additive, it is essential to reduce the extraneous cognitive load when the intrinsic cognitive load is very high [36]. The last category is the germane cognitive load, which, like the extraneous load, depends on the mode of presentation. The difference, however, is that a germane cognitive load enhances learning. Among other things, it leads to schema acquisition and it is therefore crucial for learning.
There are seven questions which the participants have to answer with respect to cognitive load. The first two questions are used to analyse intrinsic cognitive load, the following three questions are used to determine extraneous cognitive load, and the last two questions are used to answer germane cognitive load. The answers

may be given according to a 5-point Likert scale from strongly disagree (i.e., 0) to strongly agree (i.e., 5) (see Appendix C).

Ranking

After answering the questions about a specific scenario, the participants are asked to sort the process models they have seen according to their subjective comprehensibility and then give a reason for the ranking (why the chosen process model is in first or last place).

4.4.4 Experimental Design

Despite the current corona situation, the experiment is carried out on-site at the University of Ulm. Special attention is paid to appropriate hygiene measures in order not to endanger the participants.

The participants are seated on a chair in front of a 24” monitor. Both the questionnaire and later the process models are displayed on this. Furthermore, a laptop with a keyboard and a mouse is used to answer the questions and navigate the questionnaires. The set-up can be seen in Figure 4.2.

In order to obtain comparable results, a fixed procedure is followed. Figure 4.3 shows the procedure of the study for one participant.

First, demographic data on the participant (age and gender) is collected. Then the participant is asked about their previous knowledge. Here, the participant is asked to list process modelling languages the participant knows and indicate the time spent with process modelling or looking at process models in days. Then five true-or-false knowledge questions (see Appendix C) are asked to query the level of experience. The eye-tracker is then calibrated with a five-point-screen-based calibration, in which the participant is shown five points in succession.
Then the participant is shown various process models in full-screen mode, which the participant is supposed to try to understand. The participant is asked to proceed as quickly as possible but also as accurately as possible. For each process model, three true-or-false questions then appear to determine whether the process model is interpreted correctly (see the questions in Appendix C).

Figure 4.2: Set-up for experiment

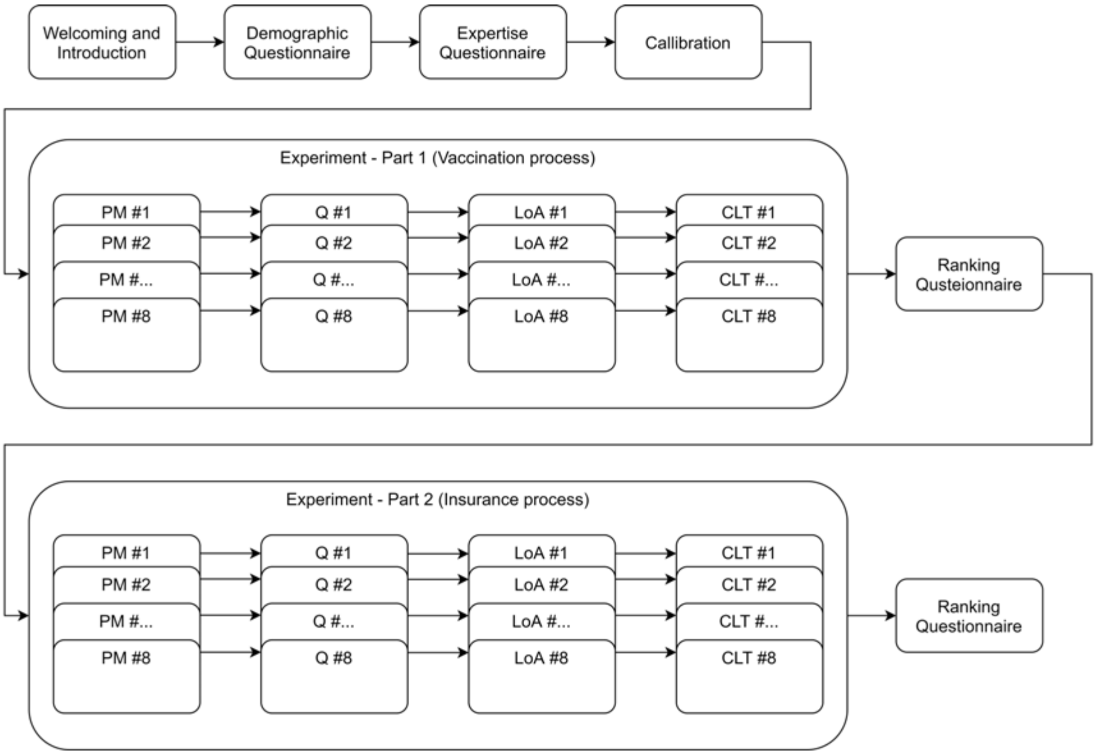

Figure 4.3: Procedure for each participant

In addition, the participant is asked to answer questions about the process model under consideration related to the level of acceptability and the cognitive load for each process model. The questionnaires used can be found in Appendix C. The participant is shown a total of eight process models. At first, process models are presented for the vaccination process and then for the insurance process. The manually created process model is presented first, followed by selecting three generated process models in random order. A permutation table is created to select which process models are shown to a subject (see Table 4.1). Finally, after each scenario, the participant is asked to rank the seen process models according to their comprehensibility and justify the choice.

	Permutation 1	**Permutation 2**	**Permutation 3**	**Permutation 4**
Apromore (BPMN)	x			x
Celonis Snap	x		x	
PM4Py (Petrinetz)	x		x	
PM4Py (BPMN)		x	x	
PM4Py (HeuNet)		x		x
Disco		x		x

Table 4.1: Permutations for process model selection

For the two scenarios, different permutations for selecting process models are used so that each participant views each type of process visualisation. The permutations 1 and 2 and permutations 3 and 4 are each used together.

4.4.5 Instruments

Various materials are needed for the experiment described. Firstly, the eye-tracker is needed to record the eye movements. For this purpose, the Pupil Labs Core headset is used in the experiment, which was already introduced in Section 2.6. With the help of the Pupil Player, the recordings can be visualised, exported and then used for analysis. The following Figure 4.4 shows a paused sequence from such an exported recording. This Figure shows the screen with one process model to be viewed. At the top left, the detected pupils are displayed. In this case, they are very well matched. In addition, the currently focused point of view is highlighted in colour.

In addition, questionnaires are needed to collect the demographic data and retrieve

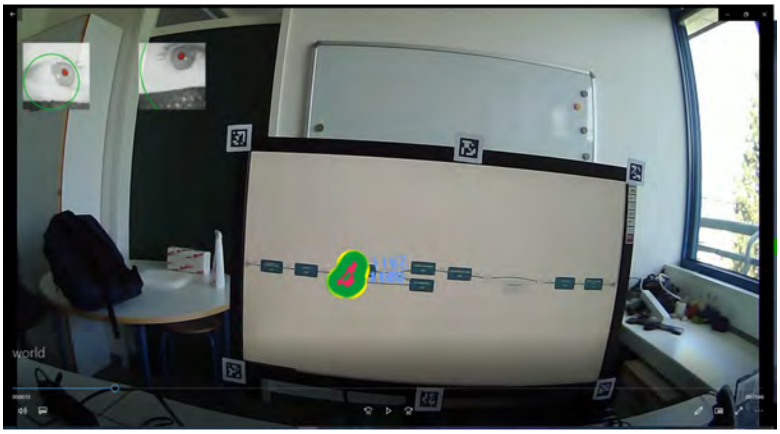

Figure 4.4: Eye-tracking recording

the comprehension questions, the questions on the level of acceptability and cognitive load.

All materials are presented digitally in a questionnaire using the EFS (Enterprise Feedback Suite) survey of questback [57] and thus presented to the participant in a coherent way. The process models, which represent the central materials, are integrated into the questionnaire. In this way, the participant is guided through the entire study by simply clicking on the questionnaire.

The process models have already been roughly presented in Section 2.5 and reference is made to Appendix B, which contains the generated process models. For the study, a selection of process models is made to reduce the effort for participants.

The selection of the process models can be justified as follows. Based on the research question of the hypotheses, the manually created process models must be included. The process maps of Celonis Snap and Disco are included in the selection. Since the process models created with ProM are not easy to display on a full screen, they are omitted. Participants should not have to scroll in a window of the screen to obtain comparable results. In addition, the created process visualisations can already be covered with other tools and would thus not have added any value to the study.
Regarding Apromore, only the BPMN diagram is considered and not also the process map. Since the process map is almost identical to the BPMN diagram, it is omitted. Due to the relevance of BPMN, BPMN diagrams should be included in the study.

Concerning PM4Py, all process models generated with the HeuristicsMiner are considered. The process models generated with the Inductive Miner are partly identical to those generated with the HeuristicsMiner, and other generated process models are identified as unsuitable. That is because the models are strongly generalised so that more types of execution are possible than are specified by the scenario. As this is not intended, these process models are not considered.

Finally, the IBM software SPSS Statistics (version 27) [30] is used for the statistical analyses. SPSS allows the results to be checked for significance.

4.5 Operation and Data Validation

A total of fifteen participants have been recruited for the study. Three of them are female, and twelve are male. The average age of the participants is 28, with the youngest born in 1996 and the oldest born in 1977.
The study is conducted on several days at the University of Ulm, always in the same room.

After providing their demographic data, the participants are asked to name all process modelling languages they already know. Fourteen participants name BPMN as the process modelling language they know so far, which is the most frequent. Six participants name EPC, and five participants name UML and Petri nets. In rare cases (up to a maximum of three participants), process modelling languages such as eGantt, Flowcharts, ADEPT and some process modelling languages from artefact-centred modelling are mentioned, to name a few. Only one participant can not think of any known process modelling languages.

The participants expertise vary greatly. When asked how much time they had spent on process modelling or looking at process models so far, the average is 105.5 days. However, the standard deviation is around 250 because a few participants only indicate a small experience of 4-5 days, while others indicate an extensive experience of 100 or 1000 days.

The demographic data, the details of the known process modelling languages and the experience in days can be found in the Appendix D in Table D.1.

In the answers to the knowledge questions, the differences in expertise are not so clearly visible. On average, 3.4 out of 5 questions are answered correctly, and the standard deviation is 0.95. How the participants answer the knowledge questions can be found in the Appendix D Table D.2.

All participants with whom the study has also been started completed it. No one has dropped out of the study. Questionnaire variants 1, 2 and 3 are completed four times and variant 4 three times. A study session took between 20 and 45 minutes.

All data sets are used to evaluate the comprehension questions (i.e., duration of answering and number of correct answers). For the eye-tracking data, one data set is to be partially excluded for the duration of the observation of the process models. In addition, one data set for the number of fixations is excluded. The exclusion is due to problems with the recording or a faulty calibration.

5 Study Evaluation

After presenting the study conducted in the previous Chapter, this Chapter focuses on the study results. First, a descriptive evaluation is carried out, followed by further analyses, in particular, to determine possible significances. Subsequently, the results are interpreted, and the limitations of the study are pointed out with a critical reflection of the study. Finally, the central results of the user study are summarised.

5.1 Descriptive Analysis

To be able to evaluate the experiments conducted, the various measured values are first needed. Average values are determined to be able to make comparisons more easily. In Table 5.2 for each process model (see column ID), the average duration for viewing the process model (see column Dur. in ms), the average number of fixations (see column Fix.), the average duration for answering comprehension questions (see column Resp. Time in ms), the average score for correctly answered comprehension questions (see column Score), perceived usefulness for understandability (see column PUU), and perceived ease of understandability (see column PEU), intrinsic (see column ICL), extraneous (see column ECL), and germane (see column GCL) cognitive load are depicted. For each process model, a unique process ID is assigned, which is listed in Table 5.1.

The duration for viewing the process models and the response times, as well as the number of fixations, are obtained from the eye-tracker recordings. All other data in the table are exported from the questionnaires.

The process models are viewed for an average of 52.89 seconds with a standard deviation of 29.70 seconds, and 226 fixations are required with a standard deviation of 122 fixations. The participants took an average of 22.40 seconds with a standard

ID	Process Model
1	manually created process model (vaccination process)
2	BPMN diagram generated by Apromore (vaccination process)
3	process model genereated by Celonis (vaccination process)
4	process model generated by Disco (vaccination process)
5	BPMN diagram generated by PM4Py (vaccination process)
6	Heuristic net generated by PM4Py (vaccination process)
7	Petri net generated by PM4Py (vaccination process)
8	manuelly created process model (insurence process)
9	process model generated by Apromore (insurence process)
10	process model generated by Celonis (insurence process)
11	process model generated by Disco (insurence process)
12	BPMN diagram generated by PM4Py (insurence process)
13	Heuristic net generated by PM4Py (insurence process)
14	Petri net generated by PM4Py (insurence process)

Table 5.1: Assignment of process model IDs

deviation of 10.80 seconds to answer the comprehension questions for the process models. They can answer 2.3 out of 3 questions correctly with a standard deviation of 0.74. This shows that the process models can be understood intuitively. The perceived usefulness for understandable is 13.95 (max. is 20) with a standard deviation of 3.887, and the perceived ease of understandable is 14.31 (max. is 20) with a standard deviation of 3.891. This indicates that the participants do not accept all visualisations. The best performing process models have IDs 1 and 8, which are the manually created process models. The worst performing process model is the process model with ID 14, which is the Petri net generated from PM4Py. As far as cognitive load is concerned, intrinsic cognitive load scores 2.88 (min. is 1, max. is 5) with a standard deviation of 1.117 and extraneous cognitive load scores 3.47 (min. is 1, max. is 5) with a standard deviation of 0.978. This shows that the working memory load is at a moderate level. The germane cognitive load value of 2.65 (min. is 1, max. is 5) with a standard deviation of 1.018 shows that there is not much learning in the tasks, as little new mental schemata have to be constructed. The data collected and summarised in Table 5.2 are detailed in the Appendix D in Table D.3 to Table D.9.

ID	Dur.	Fix.	Resp. Time	Score	PUU	PEU	ICL	ECL	GCL
1	32584,71	138	21848,67	2,33	17	18	2	3	2
2	42289,57	174	33114	3	15	15	3	3	2
3	62896,88	295	27066,5	2,38	14	14	3	4	3
4	30594,43	122	14451,14	1,86	14	16	3	3	2
5	37504,88	169	16525,5	2	15	15	2	3	2
6	28971,86	135	16349,29	2	12	14	3	3	3
7	34586,25	156	18227,38	2,75	14	14	2	3	2
8	82968,47	331	21270,93	2,6	16	16	3	4	2
9	82350,5	348	18363,25	2	12	13	3	4	3
10	56923,86	191	22899	1,71	13	13	4	4	3
11	64203,75	294	27914,63	2,25	13	13	3	4	3
12	61836,57	244	22311,14	2,57	12	13	3	4	3
13	57028,38	277	28385,5	2,38	12	12	3	4	3
14	48604,14	215	26556,71	2	10	10	3	4	3

Table 5.2: Measured results (average values) for each process model

In addition, after answering the questions about a specific scenario, the participants are asked to sort the four process models they have seen according to their comprehensibility and then give reasons for the ranking (why the chosen process model comes first or last).

The process model with ID 1 (the manually created process model) is ranked as the most comprehensible process model for the vaccination process. This is justified with the already known process modelling language BPMN, the horizontal orientation of the process model and the lack of numbers about the frequency. The participants disagree on which process model is the most difficult to understand. The process models with ID 1, ID 3 (process map generated by Celonis), ID 6 (heuristic net generated by PM4Py) and ID 7 (Petri net generated by PM4Py) are chosen most often (i.e. three times each). The reason given several times for model 1 is that the boundary event has to be known, and the process model is also perceived as complex. One participant finds the process model rather confusing. The vertical arrangement and the numbers are given several times as reasons for process model with ID 3. They also found the process model difficult to understand because much information is shown at once. In the case of process model with ID 6, the numbers for frequency are also mentioned, the choice of colours and the lack of gateways. In process model with ID 7, the black block, i.e. the element without obvious meaning, confused the participants. In addition, one participant stated that the numbers are missing to correctly represent the cycle between the briefing and the waiting (as this does not mean that only one briefing may take place, as it is presented in the other process models).

The process model with ID 8 (the manually created process model) is ranked as the most comprehensible process model for the insurance process. This is justified by the already known process modelling language BPMN, the lack of numbers about the frequency as well as the labelled paths and that the participants are represented. In addition, the choice of colours is mentioned and the fact that the rejection is shown more clearly as being executed later. The process model with the ID 14 (Petri net generated by PM4Py) is ranked as the most difficult to understand. Here it matters that it is not identified by all participants as a Petri net because the semantics of some elements is unclear (especially the kind of split elements and the meaningless black box). Finally, few elements and information are present in the process model, and it is not easy to follow the main path.

5.2 Data Analysis and Interpretation

In this Section, further analysis is presented to identify significant differences in the results. First, the vaccination process is compared with the insurance process. Then a variance analysis is carried out for both scenarios. Finally, the results of the correlation analysis are presented.

5.2.1 Analysis of Process Scenarios

The thesis has assumed that the vaccination process is a more understandable process than the insurance process. A two-independent Mann-Whitney U test is performed for each parameter to verify this statistically. The significance value is set to $p < .05$.

The results of the Mann-Whitney U test can be seen in Table 5.3. The U and z values are given for each parameter, and the significance value p is given in the next column. Some significant differences are found between the two groups based

Dur.:	U = 669.000	z = -5.852	p = .000	r = 0.536
Fix.:	U = 640.000	z = -5.400	p = .000	r = 0.510
Resp. Time:	U = 1363.000	z = -2.294	p = .022	r = 0.209
Score:	U = 1656.000	z = -.827	p = .408	r = 0,075
PUU:	U = 9.500	z = -1.951	p = .051	r = 0.521
PEU:	U = 5.500	z = -2.472	p = .013	r = 0.661
ICL:	U = 12.000	z = -2.015	p = .044	r = 0.538
ECL:	U = 3.500	z = -3.122	p = .002	r = 0.834
GCL:	U = 10.500	z = -2.082	p = .073	r = 0.556

Table 5.3: Results of Mann-Whitney U test for each parameter

on the different variables. Therefore, the r-value is also calculated. R is the correlation coefficient by which the effect size is calculated. For the number of correctly answered comprehension questions, small effect size and also no significance is found. The participants can answer the questions on both scenarios equally well. For the response time, significance can be determined, and medium effect size can be found. For the PUU and the GCL, no significance but a large effect size can be identified. There is a significance or even high significance and a large effect size for all other variables seen in the table. There are highly significant differences between the groups, especially concerning process observation time, the number of fixations, and ECL. The detailed reports of the analysis can be found in the Appendix D. Based on the results, it can be confirmed that the insurance process is more complicated to understand than the vaccination process.

5.2.2 Analysis of Vaccination Process

First, a detailed analysis of the vaccination process is to be performed. A one-way analysis of variance (ANOVA) is carried out to determine the extent to which the process models can be compared with each other. The significance value is set at $p < .05$. A function C is introduced, which acts like a compare function to specify which two process models are compared.

The manually created process model is compared with the different generated process models (see Table 5.4). No significance can be found here.

However, when comparing process models 1 and 3, it is noticeable that the value for fixations is only just above the defined significance value. A reduced value can also be found for the duration of how long the process models are viewed.

Compare	**Sig. Duration**	**Sig. Fixations**	**Sig. Resp. Time**	**Sig. Score**
C(1, 2)	p = 1.000	1.000	1.000	1.000
C(1, 3)	p = .537	.060	1.000	1.000
C(1, 4)	p = 1.000	1.000	1.000	1.000
C(1, 5)	p = 1.000	1.000	1.000	1.000
C(1, 6)	p = 1.000	1.000	1.000	1.000
C(1, 7)	p = 1.000	1.000	1.000	1.000

Table 5.4: Comparison between the manually created process model and the different generated process models

It can be seen from the values that there are no significant differences between the manually created process model and the generated process models. This means that the process models are equally easy to understand, and, above all, the hypothesis that the models are similarly understandable has been confirmed.

Looking at the comparison between the different generated process models, no significances can be found either (see Table 5.5).

However, there are reduced values for the response times and the number of correct answers when comparing process model 2 with process models 4, 5, 6 and 7. When looking at how long participants look at the process models and how many fixations are needed, decreased values can be found when comparing process model 3 with process models 4, 6 and 7.

The analysis shows that the comprehensibility of the process models is comparable, as no significant differences are found. Therefore, the second hypothesis can also be confirmed.

5.2.3 Analysis of Insurance Process

Next, a detailed analysis of the insurance process is to be performed. Also, here a one-way analysis of variance (ANOVA) is carried out to determine the extent to which the process models can be compared with each other. The significance value

Compare	Sig. Duration	Sig. Fixations	Sig. Resp. Time	Sig. Score
C(2, 3)	p = 1.000	1.000	1.000	1.000
C(2, 4)	p = 1.000	1.000	.078	.254
C(2, 5)	p = 1.000	1.000	.194	.607
C(2, 6)	p = 1.000	1.000	.239	.779
C(2, 7)	p = 1.000	1.000	.514	1.000
C(3, 4)	p = 1.000	.180	1.000	1.000
C(3, 5)	p = 1.000	1.000	1.000	1.000
C(3, 6)	p = .748	.376	1.000	1.000
C(3, 7)	p = 1.000	.607	1.000	1.000
C(4, 5)	p = 1.000	1.000	1.000	1.000
C(4, 6)	p = 1.000	1.000	1.000	1.000
C(4, 7)	p = 1.000	1.000	1.000	1.000
C(5, 6)	p = 1.000	1.000	1.000	1.000
C(5, 7)	p = 1.000	1.000	1.000	1.000
C(6, 7)	p = 1.000	1.000	1.000	1.000

Table 5.5: Comparison between the different generated process models

is set at $p < .05$. A function C is introduced, which acts like a compare function to specify which two process models are compared.

In Table 5.6 the manually created process model is compared with the generated process models.

Compare	Sig. Duration	Sig. Fixations	Sig. Resp. Time	Sig. Score
C(8 ,9)	p = 1.000	1.000	1.000	1.000
C(8, 10)	p = 1.000	.505	1.000	.601
C(8, 11)	p = 1.000	1.000	1.000	1.000
C(8, 12)	p = 1.000	1.000	1.000	1.000
C(8, 13)	p = 1.000	1.000	1.000	1.000
C(8, 14)	p = .237	1.000	1.000	1.000

Table 5.6: Comparison between the manually created process model and the different generated process models

As with the vaccination process, no significance can be found here. Reduced values can only be found in the comparison of process model 8 with models 10 and 14.

The analysis shows that, contrary to expectations, the generated process models are comparable to the manually created process model in terms of comprehensibility. The analysis, therefore, confirms the counter-hypothesis.

Finally, the generated process models are compared with each other (see Table 5.7).

Compare	**Sig. Duration**	**Sig. Fixations**	**Sig. Resp. Time**	**Sig. Score**
C(9, 10)	p = 1.000	.576	1.000	1.000
C(9, 11)	p = 1.000	1.000	1.000	1.000
C(9, 12)	p = 1.000	1.000	1.000	1.000
C(9, 13)	p = 1.000	1.000	1.000	1.000
C(9, 14)	p = .779	1.000	1.000	1.000
C(10, 11)	p = 1.000	1.000	1.000	1.000
C(10, 12)	p = 1.000	1.000	1.000	1.000
C(10, 13)	p = 1.000	1.000	1.000	1.000
C(10, 14)	p = 1.000	1.000	1.000	1.000
C(11, 12)	p = 1.000	1.000	1.000	1.000
C(11, 13)	p = 1.000	1.000	1.000	1.000
C(11, 14)	p = 1.000	1.000	1.000	1.000
C(12, 13)	p = 1.000	1.000	1.000	1.000
C(12, 14)	p = 1.000	1.000	1.000	1.000
C(13, 14)	p = 1.000	1.000	1.000	1.000

Table 5.7: Comparison of the different generated process models

Here, too, no significance and hardly any reduced values can be found. These only occur when comparing process models 9 with 10 and 9 with 14. Therefore, the fourth hypothesis can be confirmed.

5.2.4 Correlation Analysis

After conducting the variance analysis, a correlation test according to Bravais-Pearson [46] is carried out to test possible correlations between the different variables. Since all data are metric variables, this is possible.

The following Table 5.8 shows the correlations between the individual variables. The significance value is set at 0.05 (indicated with *), and the high significance value is set at 0.01 (indicated with **).

	Exp.	Dur.	Fix.	Resp. Time	Score	PUU	PEU	ICL	ECL	GCL
Exp.	1	-.332**	-.315**	-.304**	-.216*	-.168	-.160	.031	-.457**	-.003
Dur.	-.332**	1	.906**	.162	.201*	-.015	-.099	.196*	.451**	.213*
Fix.	-.315**	.906**	1	.162	.171	-.068	-.129	.203*	.390**	.208*
Resp. Time	-.304**	.162	.162	1	.042	.080	.057	.050	.325**	-.010
Score	-.216*	.201*	.171	.042	1	.055	.071	-.132	.035	-.017
PUU	-.168	-.015	-.068	.080	.055	1	.920**	-.358**	-.258**	-.670**
PEU	-.160	-.099	-.129	.057	.071	.920**	1	-.433**	-.312**	-.675**
ICL	.031	.196*	.203*	.050	-.132	-.358**	-.433**	1	.498**	.417**
ECL	-.457**	.451**	.390**	.325**	.035	-.258**	-.312**	.498**	1	.460**
GCL	-.003	.213*	.208*	-.010	-.017	-.670**	-.675**	.417**	.460**	1

Table 5.8: Results of the Pearson correlation for each parameter

Several significances can be identified. Firstly, the longer the participants look at the process model, the more fixations are needed. Furthermore, it is significant that the longer the participants look at the process model, the higher the cognitive load (both ICL, ECL and GCL). The cognitive load is also significantly higher when more fixations are needed, which can also be logically concluded by the first two statements.

The high significances between the intrinsic cognitive load, extraneous cognitive load and the germane cognitive load are due to the additivity of these three.

A more interesting result is that the longer the process models are viewed, the higher the number of correct comprehension questions. This means that it is worthwhile for the participants to look closely at the process model to be able to answer the questions. It also shows that the process models or the questions, in general, are too challenging to grasp all the information in a short viewing or to answer all the questions correctly.

The higher the intrinsic cognitive load, the significantly longer the participants look at the process models and also require more fixations.

The longer the response time is, the higher the extraneous cognitive load is measured to understand the process models. It can be concluded that the process model is more difficult to understand or that the comprehension questions are too complicated. This could be further investigated in future experiments.

A highly significant correlation between the level of acceptability and the cognitive load can also be measured. The higher the level of acceptability for the process model, the lower the cognitive load (both ICL, ECL and GCL). Conversely, it can be concluded that process models that are not considered useful or comprehensible due to the way of representation also require a higher cognitive load.

Some correlations can be found by looking at the experience of the participants. Other studies have already shown that experience has an influence on process understanding [53, 55]. The influence can also be confirmed here. Firstly, the higher the experience, the lower the time needed to look at the process models, the number of fixations and the reaction time required. This suggests that participants with more experience need less time than those with less experience. Conversely, participants with little experience require more time to understand the process models. Of particular interest is the highly significant correlation between experience and external cognitive load. The higher the experience of the participants, the lower is the external load. This suggests that the way the materials are presented can be improved for participants with low experience.

Finally, a significant relationship can be found between experience and the number of comprehension questions answered correctly. It is found that the higher the experience, the lower the number of correct answers. The reason could be that the more experienced participants overestimate themselves and therefore do not look closely enough at the individual parts of the process models. A longer time is necessary to read all the information from the process model that is relevant for the comprehension questions.

Finally, the analysis will now focus on the ranking. Here, too, a correlation test, according to Bravais-Pearson, is carried out. No significance is found concerning the variables for expertise, duration of observation, number of fixations, response time and number of correctly answered questions. In contrast, high significances are found for the level of acceptability and the cognitive load, which are shown in the following Table 5.9.

	PUU	PEU	ICL	ECL	GCL
Ranking	-.578**	-.568**	.271**	.282**	.515**

Table 5.9: Results of the Pearson correlation between ranking and level of accaptability and cognitive load

Apart from the correlations, the following is noticed to the given ranking with the number of correctly answered comprehension questions. The manually created process model for the vaccination process is rated as the best comprehensible process model. However, not more questions are answered correctly here. All participants who have seen the generated process model of Apromore answer all questions correctly. Nevertheless, they do not select it as the most comprehensible process model. The least correct answers are given to the generated process model of Disco, with an average of 1.86 correct answers. The ranking does not identify one process model as the worst comprehensible, but Disco's process model is not among them.

The manually created process model for the insurance process is chosen as the most understandable process model in the ranking, which is only slightly ahead with 2.6 out of 3 correct answers. Almost all process models achieve a value above 2 here. The only process model with fewer correct answers is the process model from Celonis, with an average of 1.7 correct answers. Nevertheless, this is not rated as the worst comprehensible process model, but the Petri net from PM4Py.

The analysis of the experiments conducted confirms the results of other studies but also reveals new approaches. In some cases, the number of participants is too small to obtain significant observations. Therefore, an important conclusion is that an experiment with a larger number of participants is required to examine these aspects again in more detail.

5.3 Limitations

In a critical review of the study, limiting factors that might have influenced the study results are detected. First of all, it can be said that the study conducted is an exploratory experiment. The objective has not been to verify previous results but to gain new insights. A limiting factor is the number of participants, which prevent drawing statistically significant conclusions for all hypotheses.
The study is based on the specification of two scenarios. The selection of the scenarios can be seen critically because it has not been checked beforehand whether the participants are familiar with them. However, care has been taken to choose scenarios that are as generally known as possible so that this aspect should not have greatly affected the study results.

Another limitation is that the first process model of the insurance scenario is created manually. The process model might not be representative, as insurance processes in the real world are more complex than the process model used here. If a new study is conducted in this area, attention should be paid to this. The process models should be adapted according to reality.
The following limitation relates to the comprehension questions. It should be noted that the process model cannot be reaccessed during the answer phase. This means that the participant has to remember everything. Thus, the questions not only test comprehensibility because the participant could also have forgotten the aspect. On the other hand, the questions are slightly different for each process model. Individual questions could be easier or more complicated than the others.
The fact that the process models have to be memorised additionally influences the cognitive load. Therefore, the measured cognitive load cannot be understood as a pure load of the process model.
The generated process models are exported from their corresponding tools and displayed only as an image. This is a substantial limitation, as no tool support could be used this way (for example, sliders, for abstracting the case frequency and other filters), which might have influenced the understanding. The use of eye-tracking glasses can also be seen as a limitation. Not all of the participants have an experience of wearing such glasses, which can negatively influence them.
Another aspect may be emerging fatigue, as the participants have to sit still with as little movement as possible during the entire experiment while concentrating on the different models and questions.
In addition, the aspect of experience can be identified as a limitation in the person-related characteristics. There is an imbalance that could have affected the statistics. During the study, there is no assistance to support the participants with less experience (for example, explanation of items or the numbers). In conclusion, the current hygiene measures due to the Corona pandemic (wearing a face mask) might have influenced the participant.

5.4 Results of User Study

Despite various limitations, some insights are gained from the study. It should be noted that this is an exploratory study and that further studies are needed to extend the findings. Various tests have been carried out to determine possible significances in the results of the study. The first step shows that there are significant differences between the vaccination process and the insurance process. As expected, the vaccination process is easier to understand, and the insurance process is a more challenging process concerning comprehensibility.
The next step is to look more closely at the vaccination process, where no significant differences can be found between the different process visualisations. Thus, null hypotheses 1 and 2 can not be rejected. Next, the insurance process is examined in more detail. Contrary to hypothesis 3, no significant difference between the manually created process model and the generated process models can be discovered. More significant differences between the process models are expected, as the information presented differs greatly from different generated process models. For example, the Petri net with low information content is rated poorly several times. However, some participants find the Celonis process model too overloaded with information. However, it should be noted that Celonis offers functions in its tool to better represent different aspects of the process model through the use of filters and sliders. This thesis, therefore, does not look at all the possibilities of Celonis and other tools. In the future, it could be investigated how these filters affect the comprehensibility of generated process models.
The results of the correlation test are interesting because some significant dependencies are identified. A correlation can be found between the experience of participants and the viewing time of process models, the number of fixations, response time and extraneous load. The correlation of experience with the understanding of process models has already been investigated in other studies and also plays a significant role in generated process models. Another correlation exists between the level of acceptability and the cognitive load. The higher the level of acceptability of the process model, the lower the cognitive load (both ICL, ECL and GCL). It can be concluded that when generating process models, attention must also be paid to the level of acceptability in order to reduce the cognitive load when looking at emerging processes. Thus, the view on this can be interesting for tool developers when choosing which process visualisation to use for process mining.

6 Conclusion and Future Work

In this concluding Chapter, the thesis is summarised, and an outlook for future work is given.

6.1 Conclusion

The objective of this thesis is to investigate the comprehensibility of process models generated by process mining. The quality of process models has a decisive influence on the analyses carried out by companies.
In order to investigate the comprehensibility of different generated process models, an exploratory eye-tracking study is conducted. With the help of an eye-tracker, gaze points can be measured, which can be an indicator of the comprehensibility of the models. The user study is carried out to answer the research question of how comprehensible the generated process models are.

First, a vaccination process and an insurance process are defined and created as manual process models. Event logs are then generated for these. A Python script is written to create an event log for the vaccination process. The insurance process is modelled in Camunda as an executable process. A Python script is written to extract the history in Camunda and create the event log. Both event logs can then be used for process mining, more specifically, process discovery. The two event logs are fed into various process mining tools, and the resulting process models are saved. A selection of these models is then tested for comprehensibility in the user study. The user study is conducted with fifteen participants at the University of Ulm. Each participant is shown a total of eight process models in succession, for each of which three true-or-false comprehension questions are then to be answered. In addition, questions regarding the level of acceptability and the cognitive load are answered for each process model. After each scenario, a ranking of the process models seen is also requested.

With the help of a Mann-Whitney U test, it can be shown that there are highly significant differences between the scenarios. Thus it can be justified that both scenarios are necessary for the study. No significant differences are identified in the analysis of variance of the process models of the vaccination process. Based on the analysis, a slight difference between the manually created process model and the generated process model from Celonis can be identified. In comparing the generated process models, minor differences are obtained when comparing the process model of Celonis with the process model of Disco, the heuristic net of PM4Py and the Petri net of PM4Py. No significances can be identified in the variance analysis for the insurance process either. When comparing the manually created process model with the generated process models, a slight difference to the process model of Celonis and the Petri net of PM4Py can be detected. When comparing the generated process models with each other, slight differences are found between the process models of Apromore and the process model of Celonis and the Petri net of PM4Py, respectively. These differences shall be investigated in a future study.

Based on the correlation analysis, some correlations can be found between the variables studied. The correlation between the time spent looking at the process models and the number of correctly answered comprehension questions is interesting. From this correlation, it can be concluded that understanding process models requires a certain amount of time and that not enough information can be gathered at a glance to answer all subsequent comprehension questions correctly. Some high significant correlations are identified concerning the reported experience of the participants. The higher the experience, the shorter the process models are viewed, and fewer fixations are needed. The response time also becomes shorter with increasing experience and the extraneous load decreases. An unexpected significance can be identified between experience and the number of correctly answered comprehension questions. The higher the experience, the fewer correct answers are given. It may be because a basic understanding of the process models is achieved very quickly by participants with much experience. The participants do not know the questions beforehand and do not memorise all the information accurately enough.

Using the Pearson correlation, it is finally shown that the subjective ranking of the participants is highly significant with the level of acceptability and the cognitive load. Therefore, in process modelling, care must be taken to ensure that the people who shall understand the model accept the way the process is visualised.

Despite interesting results, further studies are needed, as the study is confronted with some limitations (particularly the number of participants). The results can be used as a basis for future studies to further explore the field of research.

6.2 Future Work

As some significant differences could be identified between the different process visualisations, this should be investigated again with more participants. As the insurance process is simplified, an investigation with a larger real-world scenario would be fascinating. Also, it would be useful to conduct another study with business analysts and non-computer scientists who are confronted with the larger process models in the company.

Concerning the path frequencies, it could be investigated how the use of colour affects the understanding of the process models. The generated process models from this study, when colour used, rely on different shades of blue or turquoise though the numbers are still noted on each path. It would be interesting to investigate whether using the colour is sufficient for comprehension and whether the numbers only need to be faded if more detailed information is needed. Another aspect that could be further explored in future works is the influence of process orientation. Depending on the process visualisation language, the process models are displayed either vertically or horizontally.

During the research for this thesis, it is noticed that many different process modelling languages are used to represent generated models. BPMN is a standardised notation. Therefore, the question arises why BPMN is not always used to represent process models generated by process discovery. One could imagine that this could be due to the lack of support for responsibilities (pools and lanes) and the lack of differentiation between activities and events. However, this question should be investigated in future work. In addition, one could investigate to what extent the missing aspects can be read from the event logs, i.e. how this could be represented in the event logs and how this new log information can then be used to represent process models.

Bibliography

[1] A. Mark Mento - Director of Business Development Bitbrain North America. *This Is How Eye Tracking Technology Works*. 2018. URL: https://www.bitbrain.com/blog/eye-tracking-technology , last retrieved 08.09.2021.

[2] Wil MP Van der Aalst. “Business Process Management: A Comprehensive Survey”. In: *International Scholarly Research Notices* 2013 (2013). Article ID 507984. DOI: 10.1155/2013/507984.

[3] Wil MP Van der Aalst. “The Application of Petri Nets to Workflow Management”. In: *Journal of Circuits, Systems, and Computers* 8.01 (1998), pp. 21–66.

[4] Wil Van der Aalst. *Process Mining: Data Science in Action*. 2nd ed. 2016. Springer, 2016. DOI: 10.1007/978-3-662-49851-4.

[5] Wil Van der Aalst, Ton Weijters, and Laura Maruster. “Workflow Mining: Discovering Process Models from Event Logs”. In: *IEEE Transactions on Knowledge and Data Engineering* 16.9 (2004), pp. 1128–1142.

[6] Apromore Pty Ltd. *The Finest Process Mining Experience*. 2021. URL: https://apromore.org/ , last retrieved 08.09.2021.

[7] Adriano Augusto et al. “Automated Discovery of Process Models from Event Logs: Review and Benchmark”. In: *IEEE Transactions on Knowledge and Data Engineering* 31.4 (2018), pp. 686–705.

[8] Adriano Augusto et al. “Split Miner: Discovering Accurate and Simple Business Process Models from Event Logs”. In: *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2017, pp. 1–10.

[9] Jennifer Romano Bergstrom and Andrew Schall. *Eye Tracking in User Experience Design*. Elsevier, 2014.

[10] Alessandro Berti, Sebastiaan J van Zelst, and Wil van der Aalst. "Process Mining for Python (PM4Py): Bridging the Gap Between Process-And Data Science". In: *arXiv preprint arXiv:1905.06169* (2019).

[11] Marius Breitmayer. "Applying Process Mining Algorithms in the Context of Data Collection Scenarios". PhD thesis. Ulm University, 2018.

[12] Bundesministerium für Gesundheit and Robert Koch Institut. *Empfehlungen für die Organisation und Durchführung von Impfungen gegen SARS-CoV-2 in Impfzentren und mit mobilen Teams*. 2020. URL: `https://m.halle.de/push.aspx?s=downloads/de/Verwaltung/Gesundheit/Corona-Virus//Hinweise-zum-Impfen-10401/2020-12-08_empfehlungen_impfzentren.pdf` , last retrieved 08.09.2021.

[13] Camunda. *Camunda BPM Platform*. 2021. URL: `https://camunda.com/` , last retrieved 08.09.2021.

[14] Celonis. *Introducing the Celonis Snap Guide*. 2021. URL: `https://www.celonis.com/snap-guide` , last retrieved 08.09.2021.

[15] Ursula Christmann and Norbert Groeben. "Verständlichkeit: die psychologische Perspektive". In: *Handbuch Barrierefreie Kommunikation. Berlin: Frank & Timme* (2019), pp. 123–145.

[16] Marlon Dumas et al. *Fundamentals of Business Process Management*. Vol. 1. Springer, 2013.

[17] Yutika Amelia Effendi and Riyanarto Sarno. "Conformance Checking Evaluation of Process Discovery Using Modified Alpha++ Miner Algorithm". In: *2018 International Seminar on Application for Technology of Information and Communication*. IEEE. 2018, pp. 435–440.

[18] Fluxicon BV. *Academic Initiative For Process Mining Research and Education*. 2021. URL: `https://fluxicon.com/academic/` , last retrieved 08.09.2021.

[19] Fluxicon BV. *Discover Your Processes*. 2021. URL: `https://fluxicon.com/disco/` , last retrieved 08.09.2021.

[20] Fraunhofer Institute for Applied Information Technology. *Process Discovery*. 2021. URL: `https://pm4py.fit.fraunhofer.de/documentation\#discovery` , last retrieved 08.09.2021.

[21] Fraunhofer Institute for Applied Information Technology. *State-Of-The-Art-Process Mining in Python*. 2021. URL: `https://pm4py.fit.fraunhofer.de/`, last retrieved 08.09.2021.

[22] Jakob Freund and Bernd Rücker. *Praxishandbuch BPMN 2.0*. Carl Hanser Verlag GmbH Co KG, 2014.

[23] Christian W Günther and Anne Rozinat. “Disco: Discover Your Processes”. In: *BPM (Demos)* 940 (2012), pp. 40–44.

[24] Christian W Günther and Wil MP Van Der Aalst. “Fuzzy Mining–Adaptive Process Simplification Based on Multi-Perspective Metrics”. In: *International Conference on Business Process Management*. Springer. 2007, pp. 328–343.

[25] Esmita P Gupta. “Process Mining a Comparative Study”. In: *International Journal of Advanced Research in Computer and Communications Engineering* 3.11 (2014), p. 5.

[26] Michael Hammer. “Reengineering Work: Don’t Automate, Obliterate”. In: *Harvard business review* 68.4 (1990), pp. 104–112.

[27] Silvia Hansen-Schirra and Silke Gutermuth. “Empirische Überprüfung von Verständlichkeit”. In: *Eds. CHRISTIANE MAAß, and ISABEL RINK. Handbuch Barrierefreie Kommunikation. Berlin: Frank & Timme* (2019), pp. 163–182.

[28] Arthur H. M. Ter Hofstede and Mathias Weske. “Business Process Management: A Survey”. In: *Proceedings of the 1st International Conference on Business Process Management, Volume 2678 of Lncs*. Citeseer. 2003.

[29] Frank Hogrebe, Nick Gehrke, and Markus Nüttgens. “Eye Tracking Experiments in Business Process Modeling: Agenda Setting and Proof of Concept”. In: *Enterprise Modelling and Information Systems Architectures (EMISA 2011)* (2011).

[30] IBM. *IBM SPSS-Software*. 2021. URL: `https://www.ibm.com/de-de/analytics/spss-statistics-software`, last retrieved 08.09.2021.

[31] Moritz Kassner, William Patera, and Andreas Bulling. "Pupil: An Open Source Platform for Pervasive Eye Tracking and Mobile Gaze-Based Interaction". In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. 2014, pp. 1151–1160.

[32] Benedikt Lutz. "Verständlichkeit aus fachkommunikativer Sicht". In: *Handbuch Barrierefreie Kommunikation* (2018), p. 147.

[33] Jan Mendling, Mark Strembeck, and Jan Recker. "Factors of Process Model Comprehension—Findings from a Series of Experiments". In: *Decision Support Systems* 53.1 (2012), pp. 195–206.

[34] Stefan Obermeier et al. *Geschäftsprozesse realisieren: Ein praxisorientierter Leitfaden von der Strategie bis zur Implementierung*. Springer-Verlag, 2013.

[35] Object Management Group (OMG). *About the Business Process Model and Notation Specification Version 2.0*. 2010. URL: `https://www.omg.org/spec/BPMN/2.0/About-BPMN/` , last retrieved 08.09.2021.

[36] Fred Paas, Alexander Renkl, and John Sweller. "Cognitive load theory and instructional design: Recent developments". In: *Educational psychologist* 38.1 (2003), pp. 1–4.

[37] C.A. Petri. *Kommunikation mit Automaten*. Schriften des Rheinisch-Westfälischen Institutes für Instrumentelle Mathematik an der Universität Bonn. Rheinisch-Westfälisches Institut f. instrumentelle Mathematik an d. Univ., 1962.

[38] Gerardo Cepeda Porras and Yann-Gaël Guéhéneuc. "An Empirical Study on the Efficiency of Different Design Pattern Representations in UML Class Diagrams". In: *Empirical Software Engineering* 15.5 (2010), pp. 493–522.

[39] Process Mining Group, Math&CS department, Eindhoven University of Technology. *ProM*. 2016. URL: `https://www.promtools.org/doku.php?id=docs:start` , last retrieved 08.09.2021.

[40] Pupil Labs. *Fixations*. 2021. URL: `https://docs.pupil-labs.com/core/terminology/\#camera-intrinsics` , last retrieved 08.09.2021.

[41] Dario D Salvucci and Joseph H Goldberg. "Identifying Fixations and Saccades in Eye-Tracking Protocols". In: *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications*. 2000, pp. 71–78.

[42] Kamyar Sarshar and Peter Loos. “Comparing the Control-Flow of EPC and Petri Net from the End-User Perspective”. In: *International Conference on Business Process Management*. Springer. 2005, pp. 434–439.

[43] Bonita Sharif and Jonathan I Maletic. “An Eye Tracking Study on the Effects of Layout in Understanding the Role of Design Patterns”. In: *2010 IEEE International Conference on Software Maintenance*. IEEE. 2010, pp. 1–10.

[44] John Sweller. “Cognitive load during problem solving: Effects on learning”. In: *Cognitive science* 12.2 (1988), pp. 257–285.

[45] Tobii AB. *Dark and Bright Pupil Tracking*. 2021. URL: https://www.tobiipro.com/learn-and-support/learn/eye-tracking-essentials/what-is-dark-and-bright-pupil-tracking/ , last retrieved 08.09.2021.

[46] Universität Zürich. *Korrelation nach Bravais-Pearson*. 2020. URL: https://www.methodenberatung.uzh.ch/de/datenanalyse_spss/zusammenhaenge/korrelation.html , last retrieved 08.09.2021.

[47] Wil Van Der Aalst, Arya Adriansyah, and Boudewijn Van Dongen. “Causal Nets: A Modeling Language Tailored Towards Process Discovery”. In: *International Conference on Concurrency Theory*. Springer. 2011, pp. 28–42.

[48] Wil Van Der Aalst et al. “Process Mining Manifesto”. In: *International Conference on Business Process Management*. Springer. 2011, pp. 169–194.

[49] Boudewijn F Van Dongen et al. “The ProM Framework: A New Era in Process Mining Tool Support”. In: *International Conference on Application and Theory of Petri Nets*. Springer. 2005, pp. 444–454.

[50] AJMM Weijters, Wil MP van Der Aalst, and AK Alves De Medeiros. “Process Mining with the Heuristics Miner-Algorithm”. In: *Technische Universiteit Eindhoven, Tech. Rep. WP* 166 (2006), pp. 1–34.

[51] AJMM Weijters and Joel Tiago S Ribeiro. “Flexible Heuristics Miner (FHM)”. In: *IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*. IEEE. 2011, pp. 310–317.

[52] Claes Wohlin et al. *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[53] Michael Zimoch et al. "Cognitive Insights into Business Process Model Comprehension: Preliminary Results for Experienced and Inexperienced Individuals". In: *Enterprise, Business-Process and Information Systems Modeling*. Springer, 2017, pp. 137–152.

[54] Michael Zimoch et al. "Eye Tracking Experiments on Process Model Comprehension: Lessons Learned". In: *Enterprise, Business-Process and Information Systems Modeling*. Springer, 2017, pp. 153–168.

[55] Michael Zimoch et al. "Using Insights from Cognitive Neuroscience to Investigate the Effects of Event-Driven Process Chains on Process Model Comprehension". In: *International Conference on Business Process Management*. Springer. 2017, pp. 446–459.

[56] Michael Zimoch et al. "Utilizing the Capabilities Offered by Eye-Tracking to Foster Novices' Comprehension of Business Process Models". In: *International Conference on Cognitive Computing*. Springer. 2018, pp. 155–163.

[57] questback. *EFS (Enterprise Feedback Suite) Survey*. 2021. URL: https://community.questback.com/s/questback-efs-survey , last retrieved 08.09.2021.

[58] Razvan PetIn Proceedings of the rusel and Jan Mendling. "Eye-Tracking the Factors of Process Model Comprehension Tasks". In: *International Conference on Advanced Information Systems Engineering*. Springer. 2013, pp. 224–239.

[59] Zohreh SharafiProceedings of the et al. "Eye-Tracking Metrics in Software Engineering". In: *2015 Asia-Pacific Software Engineering Conference (APSEC)*. IEEE. 2015, pp. 96–103.

[60] Shehnaaz YusIn Proceedings of the uf, Huzefa Kagdi, and Jonathan I Maletic. "Assessing the Comprehension of UML Class Diagrams via Eye Tracking". In: *15th IEEE International Conference on Program Comprehension (ICPC'07)*. IEEE. 2007, pp. 113–122.

A Vaccination Process in Python

The Python script for generating the event log for the vaccination process is shown here.

```
from datetime import datetime
import random


# reformat the time and adds the random_time
def get_log_time(random_time):
    global current_time
    current_log_time = current_time
    # datetime to float
    sec = current_log_time.timestamp()
    # add random time
    new_sec = sec + random_time
    # convert float to datetime
    new_log_time = datetime.fromtimestamp(new_sec)
    # save new time base
    current_time = new_log_time
    # datetime to string with pattern
    dt_string = new_log_time.strftime("%d/%m/%Y %H:%M:%S
        ↪ ")
    # print("date and time =", dt_string)
    return dt_string


# split the time interval and returns a random value
    ↪ from it
```

```
def get_activity_time(line):
    activity_time = line.split(";")[1].split("-")
    # print(activity_time)
    # (random between 0 and 1 multiplied by max) + min
    return random.random() * float(activity_time[1]) + float(activity_time[0])


# set time for new case to allow overlapping
def set_time_for_new_case():
    global current_time
    current_log_time = current_time
    # datetime to float
    sec = current_log_time.timestamp()
    # minus 30 minutes
    new_sec = sec - 900
    # convert float to datetime
    new_log_time = datetime.fromtimestamp(new_sec)
    # save new time base
    current_time = new_log_time


# write event log
def write_event_log():
    output_object = open("event_log_vaccination.csv", "w")
    output_object.write('case_id;activity;start_timestamp;end_timestamp\n')

    for case_id in range(1000):
        input_object = open("vaccination_in.txt")
        for line in input_object:
            line = line.rstrip("\n")
            random_time = round(get_activity_time(line), 2)
```

```
            start_log_time = get_log_time(0)
            end_log_time = get_log_time(random_time)
            # line without the time (only activity name)
            activity = line.split(";")[0]
            # special case
            # generate random variable
            random_special_case = random.randint(0, 15)
            if activity == "be observed and wait" and random_special_case % 11 == 0:
                output_object.write(
                    str(case_id + 1) + ";" + activity + ";" + start_log_time + ";" + end_log_time + "\n")
                output_object.write(str(case_id + 1) + ";" + "receive first aid" + ";" + start_log_time + ";"
                                          + end_log_time + "\n")
                continue
            # write line in event log
            output_object.write(str(case_id+1) + ";" + activity + ";" + start_log_time + ";" +  end_log_time + "\n")
        input_object.close()
        set_time_for_new_case()
    output_object.close()


# main method and start
if __name__ == '__main__':
    current_time = datetime.now()
    write_event_log()
```

Listing A.1: Generation of the event log for the vaccination process

B Process Visualisations

B.1 Celonis Snap

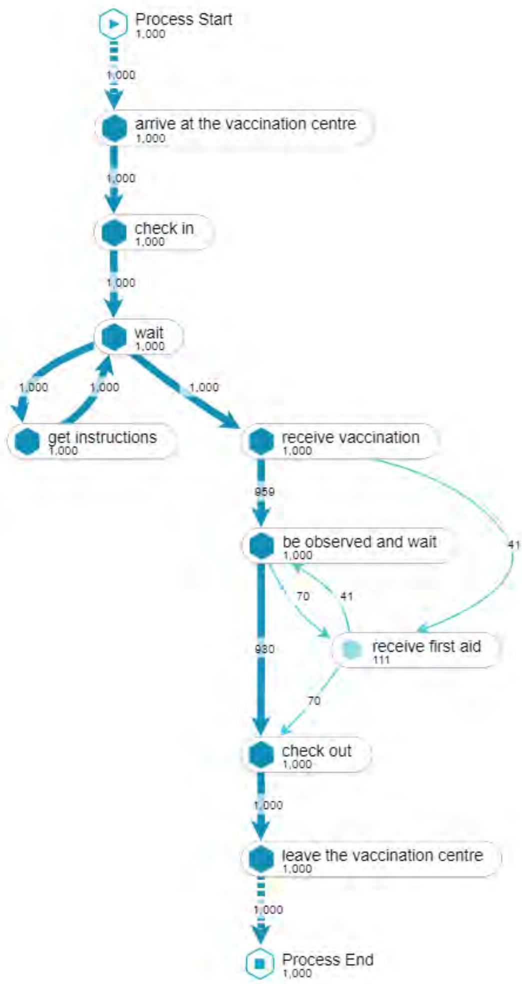

Figure B.1: Vaccination process (Celonis Snap)

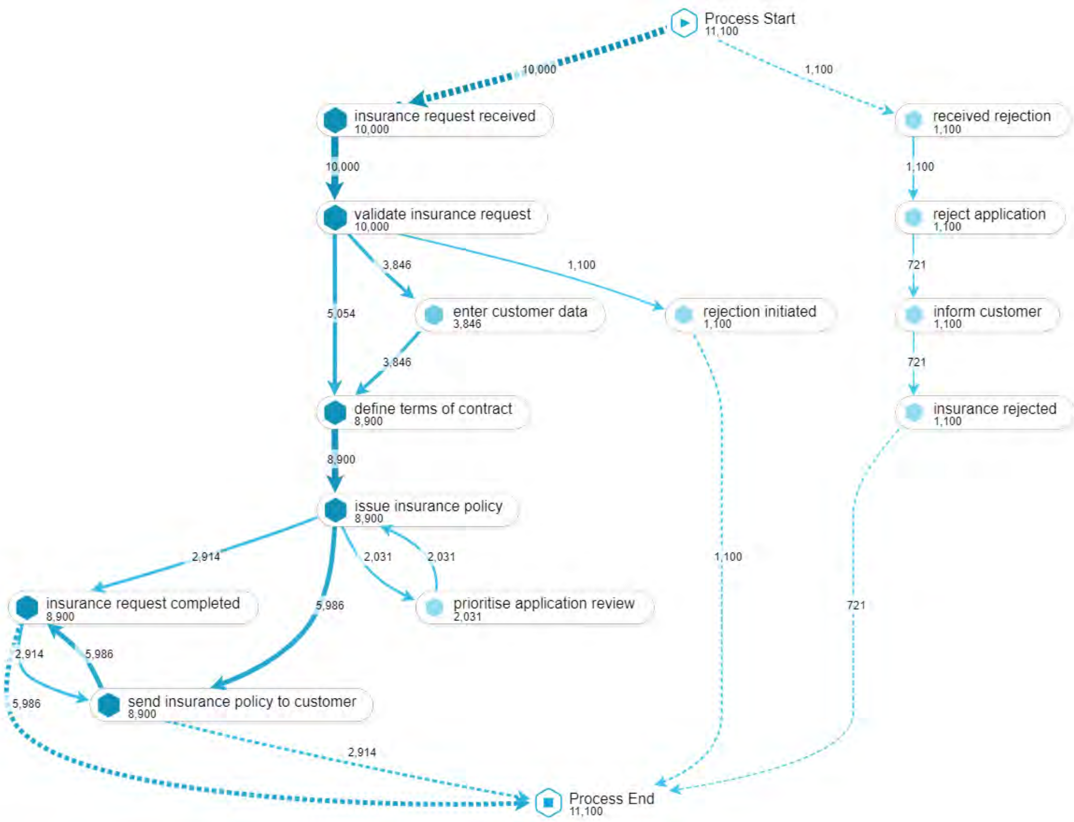

Figure B.2: Insurence process (Celonis Snap)

B.2 Disco

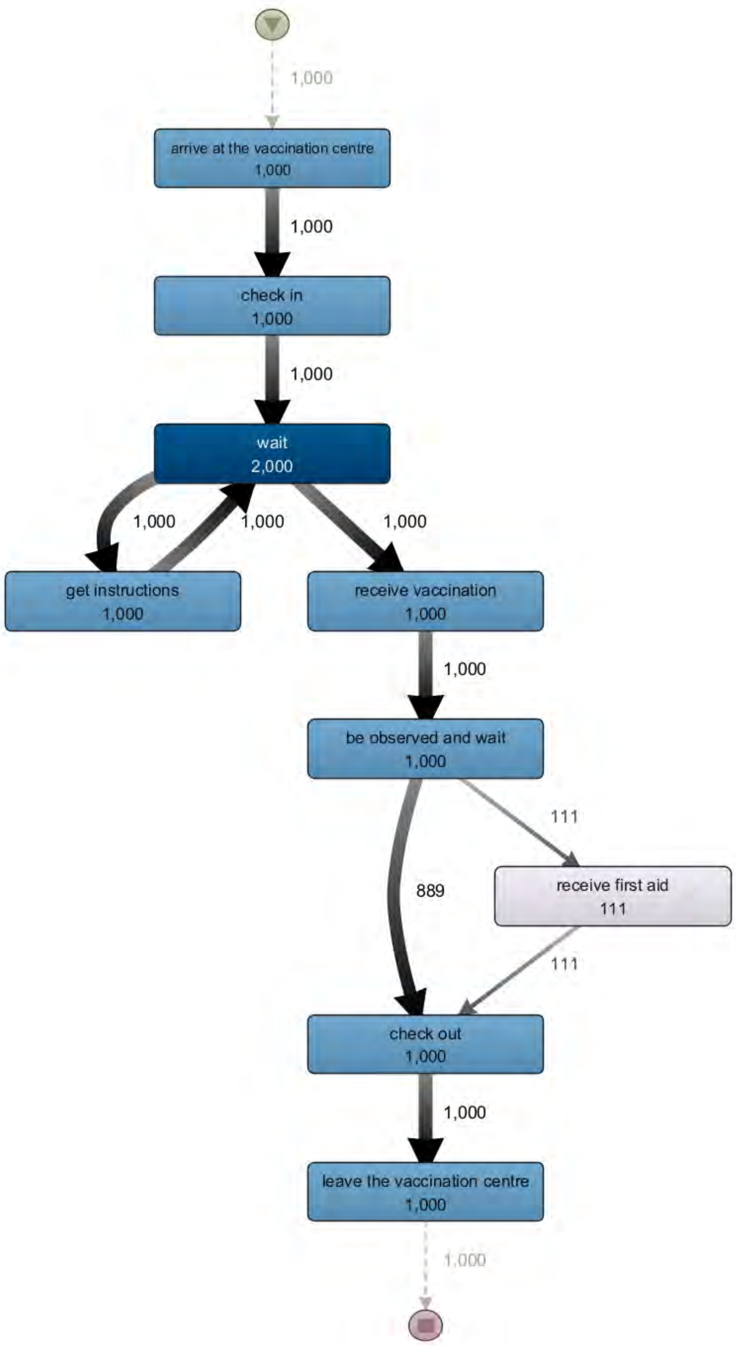

Figure B.3: Vaccination process (Disco)

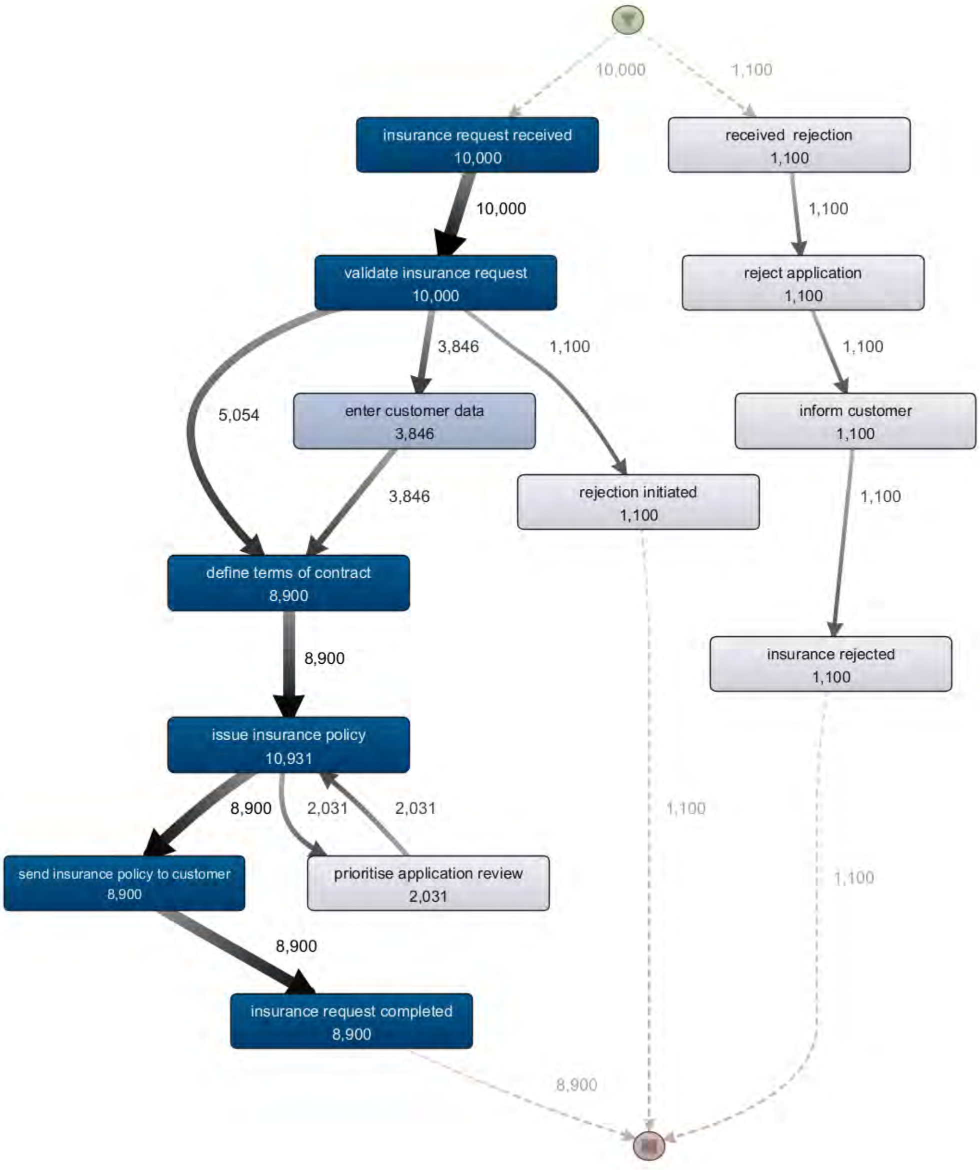

Figure B.4: Insurence process (Disco)

B.3 Apromore

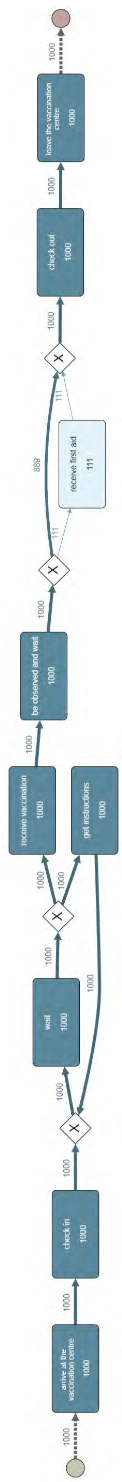

Figure B.5: Vaccination process (Apromore)

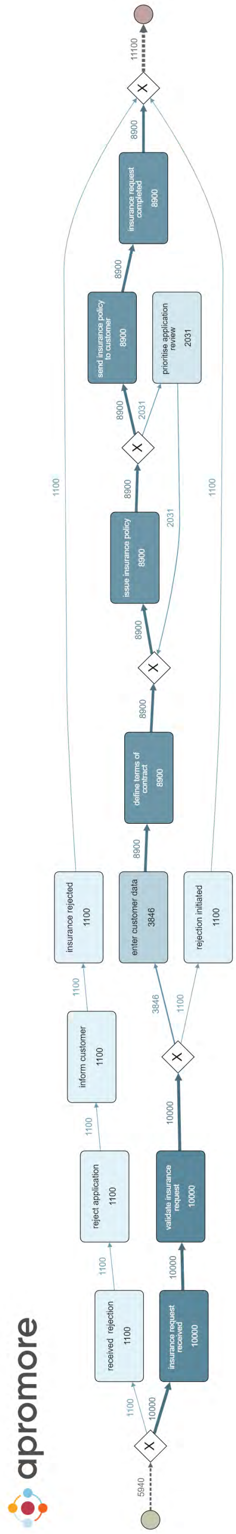

Figure B.6: Insurence process (Apromore)

B.4 PM4Py

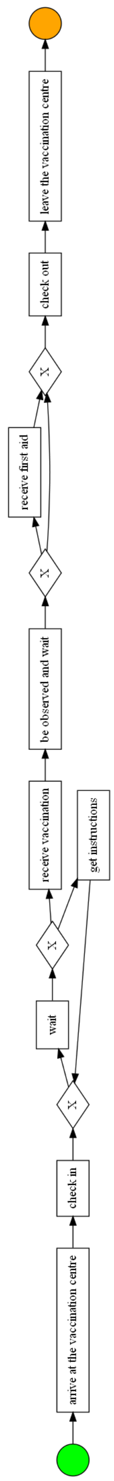

Figure B.7: Vaccination process as BPMN model (PM4Py)

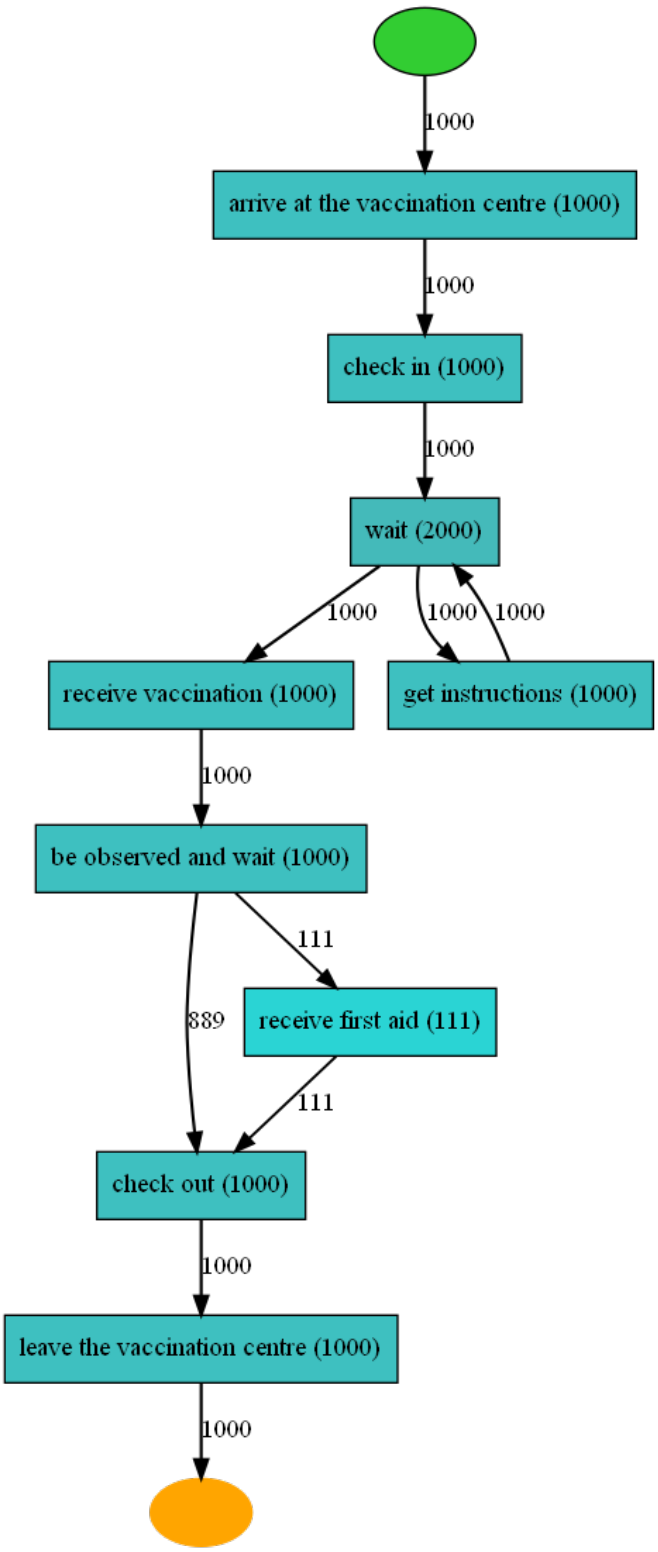

Figure B.8: Vaccination process as Heuristic net (PM4Py)

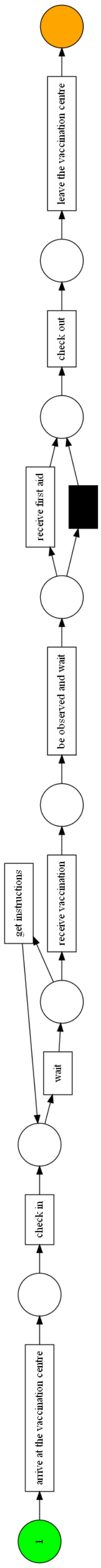

Figure B.9: Vaccination process as Petri net (PM4Py)

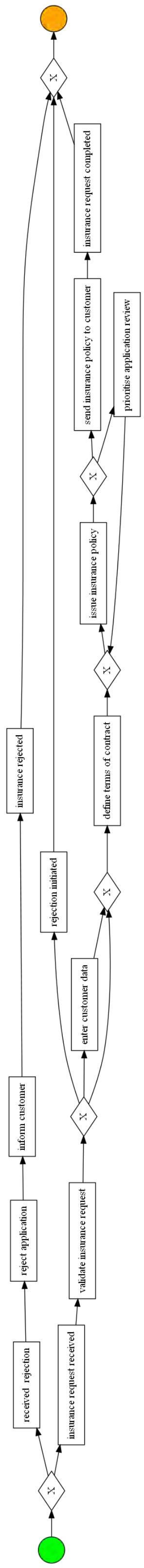

Figure B.10: Insurence process as BPMN model (PM4Py)

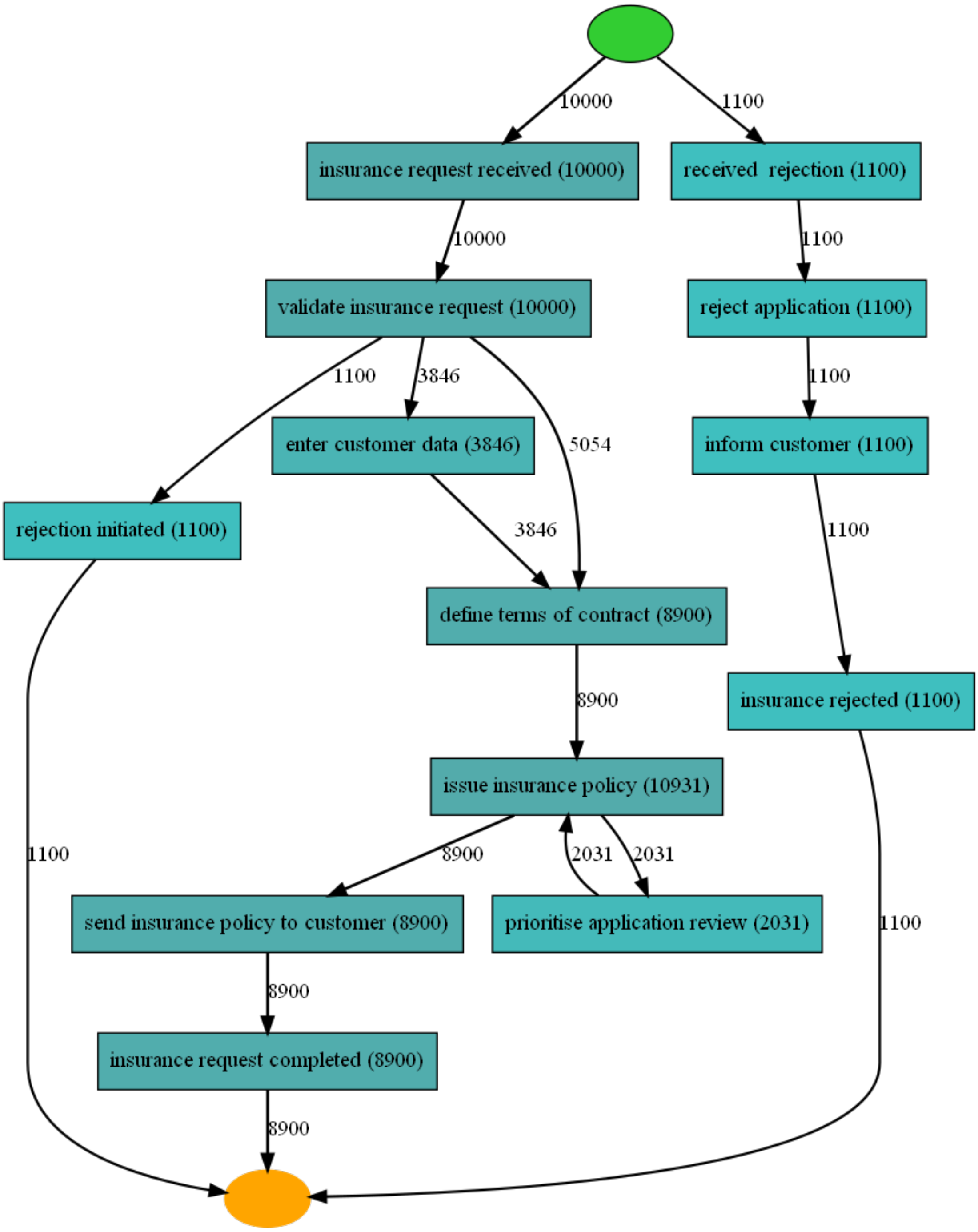

Figure B.11: Insurence process as Heuristic net (PM4Py)

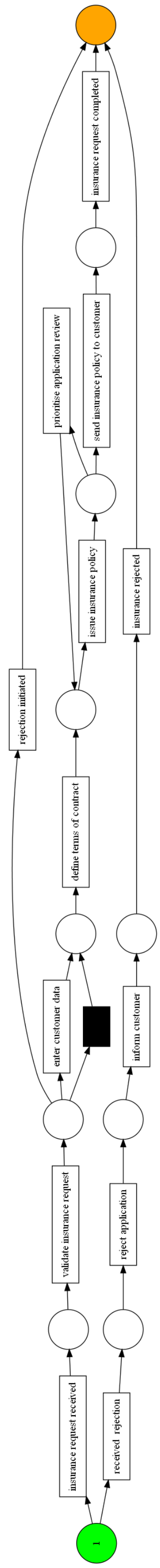

Figure B.12: Insurence process as Petri net (PM4Py)

C Questionnaires

C.1 Knowledge Questions

1. Nach einem XOR-Gateway kann höchstens nur ein Pfad ausgeführt werden.

   ○ wahr ○ falsch

2. XOR-Gateways können für die Modellierung von Schleifen verwendet werden.

   ○ wahr ○ falsch

3. Eine Synchronisation bedeutet, dass zwei Aktivitäten zur selben Zeit ausgeführt werden.

   ○ wahr ○ falsch

4. Zwei simultane Aktivitäten müssen zur selben Zeit ausgeführt werden.

   ○ wahr ○ falsch

5. Eine Aktivität in einer Schleife muss mindestens einmal ausgeführt werden.

   ○ wahr ○ falsch

Figure C.1: True-or-false knowledge questions

C.2 Comprehension Questions

- Fachliches Modell:
  1. Bevor ich die Impfung erhalte, muss ich warten. (w)
  2. Nach der Impfung kann ich das Impfzentrum sofort verlassen. (f)
  3. Direkt nach dem Check-in bekomme ich eine Einweisung. (f)
- BPMN aus Apromore:
  1. Nachdem ich das erste Mal gewartet habe, erhalte ich eine Einweisung. (w)
  2. Meistens kann ich nach der Beobachtung direkt zum Check-out weiter und erhalte keine Erste Hilfe. (w)
  3. Ich erhalte parallel die Einweisung und die Impfung. (f)
- Celonis:
  1. Nach dem Check-In muss ich warten. (w)
  2. Direkt vor der Impfung muss ich warten. (w)
  3. Nach dem Check-Out kann ich das Impfzentrum verlassen. (w)
- Disco:
  1. Als erstes kommt der Check-In. (f)
  2. Meistens ist keine Erste-Hilfe notwendig. (w)
  3. Ich erhalte gleich nach der Einweisung die Impfung. (f)
- BPMN aus PM4py:
  1. Nach dem Check-In muss ich gleichzeitig warten und erhalte die Einweisung. (f)
  2. Die Erste-Hilfe wird immer angewendet. (f)
  3. Als letzten Schritt kann ich das Impfzentrum verlassen. (w)
- C Netz aus PM4py:
  1. Nach dem Check-In muss ich warten. (w)
  2. Ich erhalte mehrfach eine Einweisung. (f)
  3. Nach der Impfung gelange ich direkt zum Check-Out. (f)
- Petrinetz aus PM4py:
  1. Ich erhalte die Einweisung, nachdem ich gewartet habe. (w)
  2. Ich erhalte immer eine Erste-Hilfe. (f)
  3. Nach dem Check-Out kann ich das Impfzentrum verlassen. (w)

Figure C.2: Comprehension questions for vaccination process

- Fachliches Modell:
  1. Der Versicherungsantrag wird vom Kunden gestellt. (w)
  2. Kundendaten müssen immer eingetragen werden. (f)
  3. Wenn der Versicherungsantrag abgelehnt wird, ist der Prozess sofort beendet. (f)
- BPMN aus Apromore:
  1. Der Versicherungsantrag kann entweder abgelehnt oder akzeptiert werden. (w)
  2. Nachdem die Kundendaten bekannt sind, werden die Vertragsbedingungen festgelegt. (w)
  3. Als letzten Schritt wird der Versicherungsvertrag an den Kunden gesendet. (f)
- Celonis:
  1. In den meisten Fällen wird der Versicherungsantrag akzeptiert. (w)
  2. Sobald der Versicherungsantrag fertig ist, wird der Prozess beendet. (w)
  3. Nach der Definition von Vertragsbedingungen wird der Versicherungsschein erstellt. (w)
- Disco:
  1. Kundendaten müssen öfter eingetragen werden, als dass der Prozess abgelehnt wird. (w)
  2. Der Prozess startet mit dem Überprüfen des Versicherungsantrags. (f)
  3. Der Schritt zum Festlegen der Vertragskonditionen kann übersprungen werden. (f)
- BPMN aus PM4py:
  1. Kundendaten müssen in jeder Prozessausführung eingetragen werden. (f)
  2. Der Prozess endet entweder mit einem abgelehnten Versicherungsantrag oder dem abgeschlossenen Versicherungsantrag. (w)
  3. Wenn der Versicherungsantrag abgelehnt wird, wird der Kunde informiert. (w)
- C Netz aus PM4py:
  1. Die meisten Versicherungsanträge werden abgelehnt. (f)
  2. Nachdem der Versicherungsantrag geprüft wurde, werden gleichzeitig die Kundendaten eingegeben und die Vertragskonditionen festgelegt. (f)
  3. In jedem Fall wurde der Versicherungsantrag überprüft. (w)
- Petrinetz aus PM4py:
  1. In manchen Fällen müssen keine Kundendaten eingegeben werden. (w)
  2. Bevor die Ablehnung des Versicherungsantrags angestoßen wird, wird der Versicherungsantrag geprüft. (w)
  3. Der Versicherungsantrag wird gleichzeitig priorisiert und an den Kunden zurück geschickt. (f)

Figure C.3: Comprehension questions for insurence process

C.3 Level of Acceptability

Bitte lesen Sie jede Aussage durch und geben Sie an, inwiefern Sie dieser zustimmen oder nicht.

	stimme gar nicht zu				stimme voll zu
Auf diese Art dargestellte Prozessmodelle sind einfach zu verstehen.	○	○	○	○	○
Der Darstellungsansatz ist eine adäquate Form für die Darstellung von Prozessen.	○	○	○	○	○
Die Verwendung von Prozessmodellen dieser Art würde die Kommunikation in Prozessen erleichtern.	○	○	○	○	○
Die Prozessmodelle in dieser Studie wurden als nützlich erachtet.	○	○	○	○	○
Es wäre einfach, die Art der Darstellung von Prozessen zu erlernen.	○	○	○	○	○
Die Darstellung der Prozessmodelle war klar und einfach verständlich.	○	○	○	○	○
Die Art der Darstellung würde Prozesse verständlicher machen.	○	○	○	○	○
Die Art der Darstellung der Prozesse ist nützlich.	○	○	○	○	○

Figure C.4: Questions for level of acceptability

C.4 Cognitive Load

Bitte lesen Sie jede Aussage durch und geben Sie an, inwiefern Sie dieser zustimmen oder nicht.

	stimme gar nicht zu				stimme voll zu
Bei diesem Prozessmodell musste man viele Dinge gleichzeitig im Kopf bearbeiten.	○	○	○	○	○
Dieses Prozessmodell war sehr komplex.	○	○	○	○	○
Ich habe mich angestrengt, mir nicht nur die einzelnen Dinge zu merken, sondern auch den Gesamtzusammenhang zu verstehen.	○	○	○	○	○
Es ging mir beim Bearbeiten der Aufgabe darum, alles richtig zu verstehen.	○	○	○	○	○
In diesem Prozessmodell war es mühsam, die wichtigsten Informationen zu erkennen.	○	○	○	○	○
Die Darstellung dieses Prozessmodells ist ungünstig, um wirklich etwas zu lernen.	○	○	○	○	○
Bei diesem Prozessmodell war es schwer, die zentralen Inhalte miteinander in Verbindung zu bringen.	○	○	○	○	○

Figure C.5: Questions for cognitive load

D Results of User Study

In this Appendix, all results are presented as tables. The first Table D.1 shows the demographic data given for each participant. Table D.2 shows the answers to the knowledge questions and the sum of the correctly answered knowledge questions for each participant. Table D.3 shows the results of how long each participant looked at each process model. The questionnaire variant (Variant), the ID for the participant (P), the scenario (S; 0 = vaccination process and 1 = insurance process), the ID of the process model (PM) and the duration are given. The meanings of the rows are identical for the following tables.
The next Table D.4 shows the number of fixations. Table D.5 shows how long each participant took to answer the comprehension questions for each process model and Table D.6 shows the number of comprehension questions answered correctly.
Table D.7 then shows the level of acceptability (V = Variant). Finally, Table D.8 and Table D.9 show the results of the cognitive load questionnaire.

ID	Variant	Gender	Age (year of birth)	Known process modelling languages	Estimated experience in days
1	0	0	1995	UML, BPMN	20
2	1	1	1991	bpmn 2.0	21
3	2	1	1994	BPMN	5
4	3	1	1996	BPMN 2.0, EPK, UML, Petri Netz, Gant	100
5	0	1	1991	BPMN, AristaFlow	25
6	1	1	1992	bpmn, petri netze, epks, artifacts, case handling, philharmonicflows	1000
7	2	1	1996	BPM, AristaFlow	60
8	3	1	1977	Den ganzen Zoo	>100
9	0	1	1996	BPMN, UML, EPK	60
10	1	1	1990	BPMN, EPKs, Flow Charts, UML, ADEPT	7
11	2	1	1997	BPMN	35
12	3	1	1994	petri netze, flussdiagramme, bpmn 2.0, gantt charts	30
13	0	1	1995	Keine	4
14	1	0	1993	BPMN 2.0, EPK, Petri-Netze, object-aware etc.	100
15	2	0	1995	BPMN, Flussmodelle	10

Table D.1: Results of demographic data

ID	Question 1	Question 2	Question 3	Question 4	Question 5	Sum of correctly answered questions
1	1	1	1	1	1	2
2	1	1	0	0	1	4
3	1	0	0	1	1	2
4	1	1	1	0	0	4
5	1	0	1	0	1	2
6	1	1	0	0	0	5
7	1	0	1	0	0	3
8	1	1	1	0	0	4
9	0	1	0	0	1	3
10	1	1	0	0	0	5
11	1	1	1	0	1	3
12	1	1	0	0	1	4
13	1	0	0	1	0	3
14	1	1	0	0	1	4
15	0	1	1	0	0	3

Table D.2: Results of knowledge questions for each participant

Variant	P	S	PM	Duration
0	0	0	1	27512
0	0	0	2	35552
0	0	0	3	58571
0	0	0	7	51354
0	0	1	8	91963
0	0	1	11	92130
0	0	1	12	102435
0	0	1	13	71332
1	1	0	4	38857
1	1	0	5	45460
1	1	0	6	30293
1	1	1	8	151900
1	1	1	9	149312
1	1	1	10	132369
1	1	1	14	72326
2	2	0	1	45536
2	2	0	3	98553
2	2	0	5	61612
2	2	0	7	49896
2	2	1	8	87980
2	2	1	9	104795
2	2	1	11	64238
2	2	1	13	72102
3	3	0	1	20089
3	3	0	2	40357
3	3	0	4	30011
3	3	0	6	28313
3	3	1	8	68721
3	3	1	10	46139
3	3	1	12	55858
3	3	1	14	59115
0	4	0	1	20600
0	4	0	2	28518
0	4	0	3	31967
0	4	0	7	26975
0	4	1	8	52430
0	4	1	11	58662
0	4	1	12	37486
0	4	1	13	48958
1	5	0	1	29046

Variant	P	S	PM	Duration
1	5	0	4	21284
1	5	0	5	28790
1	5	0	6	24134
1	5	1	8	41861
1	5	1	9	51784
1	5	1	10	43344
1	5	1	14	42098
2	6	0	1	41464
2	6	0	3	72355
2	6	0	5	25497
2	6	0	7	30436
2	6	1	8	104591
2	6	1	9	149221
2	6	1	11	77067
2	6	1	13	33052
3	7	0	1	10385
3	7	0	2	61345
3	7	0	4	19157
3	7	0	6	23296
3	7	1	8	19361
3	7	1	10	22723
3	7	1	12	19525
3	7	1	14	19887
0	8	0	1	34904
0	8	0	2	41184
0	8	0	3	49111
0	8	0	7	36225
0	8	1	8	65185
0	8	1	11	58754
0	8	1	12	66419
0	8	1	13	54942
1	9	0	1	21050
1	9	0	4	32135
1	9	0	5	21560
1	9	0	6	21292
1	9	1	8	59301
1	9	1	9	33778
1	9	1	10	40646
1	9	1	14	31653
2	10	0	1	40244

Variant	P	S	PM	Duration
2	10	0	3	66658
2	10	0	5	32848
2	10	0	7	21281
2	10	1	8	91411
2	10	1	9	42925
2	10	1	11	50456
2	10	1	13	76190
3	11	0	1	30580
3	11	0	2	40117
3	11	0	4	44919
3	11	0	6	31305
3	11	1	8	49503
3	11	1	10	49808
3	11	1	12	64263
3	11	1	14	62243
0	12	0	1	42859
0	12	0	2	48954
0	12	0	3	51208
0	12	0	7	35654
0	12	1	8	122461
0	12	1	11	74897
0	12	1	12	86865
0	12	1	13	42798
1	13	0	1	37828
1	13	0	4	27797
1	13	0	5	40887
1	13	0	6	44168
1	13	1	8	86123
1	13	1	9	49461
1	13	1	10	63436
1	13	1	14	52904
2	14	0	1	54089
2	14	0	3	74752
2	14	0	5	43385
2	14	0	7	24867
2	14	1	8	151735
2	14	1	9	77525
2	14	1	11	37424
2	14	1	13	56853

Table D.3: Results of time spent looking at the process models

Variant	P	S	PM	Fixations
0	0	0	1	123
0	0	0	2	159
0	0	0	3	196
0	0	0	7	201
0	0	1	8	171
0	0	1	11	267
0	0	1	12	265
0	0	1	13	333
2	2	0	1	210
2	2	0	3	464
2	2	0	5	271
2	2	0	7	238
2	2	1	8	411
2	2	1	9	489
2	2	1	11	334
2	2	1	13	309
3	3	0	1	93
3	3	0	2	210
3	3	0	4	158
3	3	0	6	146
3	3	1	8	306
3	3	1	10	209
3	3	1	12	259
3	3	1	14	277
0	4	0	1	106
0	4	0	2	133
0	4	0	3	153
0	4	0	7	122
0	4	1	8	237
0	4	1	11	285
0	4	1	12	146
0	4	1	13	250
1	5	0	1	137
1	5	0	4	106
1	5	0	5	145
1	5	0	6	124
1	5	1	8	214
1	5	1	9	260
1	5	1	10	186
1	5	1	14	220

Variant	P	S	PM	Fixations
2	6	0	1	133
2	6	0	3	363
2	6	0	5	115
2	6	0	7	123
2	6	1	8	403
2	6	1	9	627
2	6	1	11	325
2	6	1	13	142
3	7	0	1	53
3	7	0	2	138
3	7	0	4	109
3	7	0	6	120
3	7	1	8	86
3	7	1	10	99
3	7	1	12	84
3	7	1	14	87
0	8	0	1	140
0	8	0	2	188
0	8	0	3	247
0	8	0	7	180
0	8	1	8	314
0	8	1	11	279
0	8	1	12	326
0	8	1	13	280
1	9	0	1	121
1	9	0	4	175
1	9	0	5	122
1	9	0	6	122
1	9	1	8	317
1	9	1	9	184
1	9	1	10	227
1	9	1	14	171
2	10	0	1	207
2	10	0	3	364
2	10	0	5	175
2	10	0	7	111
2	10	1	8	480
2	10	1	9	245
2	10	1	11	287
2	10	1	13	411

Variant	P	S	PM	Fixations
3	11	0	1	34
3	11	0	2	176
3	11	0	4	76
3	11	0	6	163
3	11	1	8	42
3	11	1	10	192
3	11	1	12	199
3	11	1	14	298
0	12	0	1	207
0	12	0	2	215
0	12	0	3	213
0	12	0	7	153
0	12	1	8	611
0	12	1	11	377
0	12	1	12	428
0	12	1	13	209
1	13	0	1	103
1	13	0	4	109
1	13	0	5	148
1	13	0	6	137
1	13	1	8	303
1	13	1	9	241
1	13	1	10	235
1	13	1	14	238
2	14	0	1	266
2	14	0	3	362
2	14	0	5	207
2	14	0	7	116
2	14	1	8	737
2	14	1	9	389
2	14	1	11	194
2	14	1	13	284

Table D.4: Results of number of fixations

Variant	P	S	PM	Duration
0	0	0	1	35883
0	0	0	2	73645
0	0	0	3	8535
0	0	0	7	15560
0	0	1	8	18900
0	0	1	11	31685
0	0	1	12	37065
0	0	1	13	25354
1	1	0	1	32015
1	1	0	4	20521
1	1	0	5	20339
1	1	0	6	32708
1	1	1	8	20631
1	1	1	9	24795
1	1	1	10	32525
1	1	1	14	38521
2	2	0	1	19016
2	2	0	3	14480
2	2	0	5	15588
2	2	0	7	11081
2	2	1	8	45674
2	2	1	9	14427
2	2	1	11	20143
2	2	1	13	19014
3	3	0	1	18401
3	3	0	2	23124
3	3	0	4	21035
3	3	0	6	11364
3	3	1	8	18496
3	3	1	10	36275
3	3	1	12	25020
3	3	1	14	26262
0	4	0	1	20288
0	4	0	2	26472
0	4	0	3	21707
0	4	0	7	16175
0	4	1	8	21126
0	4	1	11	25698
0	4	1	12	20539
0	4	1	13	28621

Variant	P	S	PM	Duration
1	5	0	1	22950
1	5	0	4	8038
1	5	0	5	10105
1	5	0	6	7785
1	5	1	8	8505
1	5	1	9	10096
1	5	1	10	8738
1	5	1	14	22427
2	6	0	1	14522
2	6	0	3	23993
2	6	0	5	23135
2	6	0	7	27375
2	6	1	8	13854
2	6	1	9	21652
2	6	1	11	21711
2	6	1	13	29806
3	7	0	1	16294
3	7	0	2	20942
3	7	0	4	7649
3	7	0	6	17960
3	7	1	8	14406
3	7	1	10	18636
3	7	1	12	13106
3	7	1	14	16114
0	8	0	1	15898
0	8	0	2	29583
0	8	0	3	27001
0	8	0	7	19125
0	8	1	8	11403
0	8	1	11	36265
0	8	1	12	24664
0	8	1	13	29011
1	9	0	1	30518
1	9	0	4	14656
1	9	0	5	15888
1	9	0	6	12126
1	9	1	8	11539
1	9	1	9	14952
1	9	1	10	13126
1	9	1	14	24323

Variant	P	S	PM	Duration
2	10	0	1	23863
2	10	0	3	76635
2	10	0	5	15610
2	10	0	7	10990
2	10	1	8	35661
2	10	1	9	16005
2	10	1	11	18745
2	10	1	13	17324
3	11	0	1	21661
3	11	0	2	15491
3	11	0	4	12671
3	11	0	6	11529
3	11	1	8	30023
3	11	1	10	30617
3	11	1	12	15978
3	11	1	14	33343
0	12	0	1	22036
0	12	0	2	42541
0	12	0	3	19389
0	12	0	7	27913
0	12	1	8	15520
0	12	1	11	26001
0	12	1	12	19806
0	12	1	13	42010
1	13	0	1	21445
1	13	0	4	16588
1	13	0	5	11958
1	13	0	6	20973
1	13	1	8	26355
1	13	1	9	24305
1	13	1	10	20376
1	13	1	14	24907
2	14	0	1	12940
2	14	0	3	24792
2	14	0	5	19581
2	14	0	7	17600
2	14	1	8	26971
2	14	1	9	20674
2	14	1	11	43069
2	14	1	13	35944

Table D.5: Results of duration during answering the comprehension questions

Variant	P	S	PM	Score
0	0	0	1	2
0	0	0	2	3
0	0	0	3	2
0	0	0	7	3
0	0	1	8	3
0	0	1	11	2
0	0	1	12	2
0	0	1	13	2
1	1	0	1	2
1	1	0	4	2
1	1	0	5	1
1	1	0	6	3
1	1	1	8	2
1	1	1	9	2
1	1	1	10	3
1	1	1	14	3
2	2	0	1	2
2	2	0	3	3
2	2	0	5	3
2	2	0	7	2
2	2	1	8	2
2	2	1	9	2
2	2	1	11	3
2	2	1	13	3
3	3	0	1	2
3	3	0	2	3
3	3	0	4	2
3	3	0	6	1
3	3	1	8	3
3	3	1	10	2
3	3	1	12	2
3	3	1	14	2
0	4	0	1	2
0	4	0	2	3
0	4	0	3	1
0	4	0	7	3
0	4	1	8	2
0	4	1	11	2
0	4	1	12	2
0	4	1	13	3

Variant	P	S	PM	Score
1	5	0	1	3
1	5	0	4	3
1	5	0	5	1
1	5	0	6	2
1	5	1	8	3
1	5	1	9	2
1	5	1	10	2
1	5	1	14	2
2	6	0	1	3
2	6	0	3	3
2	6	0	5	3
2	6	0	7	2
2	6	1	8	3
2	6	1	9	3
2	6	1	11	1
2	6	1	13	2
3	7	0	1	0
3	7	0	2	3
3	7	0	4	2
3	7	0	6	1
3	7	1	8	2
3	7	1	10	0
3	7	1	12	3
3	7	1	14	2
0	8	0	1	3
0	8	0	2	3
0	8	0	3	3
0	8	0	7	3
0	8	1	8	3
0	8	1	11	2
0	8	1	12	3
0	8	1	13	2
1	9	0	1	3
1	9	0	4	2
1	9	0	5	1
1	9	0	6	3
1	9	1	8	3
1	9	1	9	2
1	9	1	10	2
1	9	1	14	3

Variant	P	S	PM	Score
2	10	0	1	2
2	10	0	3	2
2	10	0	5	2
2	10	0	7	3
2	10	1	8	2
2	10	1	9	2
2	10	1	11	3
2	10	1	13	2
3	11	0	1	3
3	11	0	2	3
3	11	0	4	1
3	11	0	6	1
3	11	1	8	3
3	11	1	10	1
3	11	1	12	3
3	11	1	14	1
0	12	0	1	2
0	12	0	2	3
0	12	0	3	2
0	12	0	7	3
0	12	1	8	3
0	12	1	11	2
0	12	1	12	3
0	12	1	13	2
1	13	0	1	3
1	13	0	4	1
1	13	0	5	2
1	13	0	6	3
1	13	1	8	2
1	13	1	9	1
1	13	1	10	2
1	13	1	14	1
2	14	0	1	3
2	14	0	3	3
2	14	0	5	3
2	14	0	7	3
2	14	1	8	3
2	14	1	9	2
2	14	1	11	3
2	14	1	13	3

Table D.6: Results of number of correctly answered comprehension questions

V	P	S	PM	PUU	PEOU
0	0	0	1	20	20
0	0	0	2	17	17
0	0	0	3	20	18
0	0	0	7	14	16
0	0	1	8	20	20
0	0	1	11	11	11
0	0	1	12	17	18
0	0	1	13	19	18
1	1	0	1	20	19
1	1	0	4	15	15
1	1	0	5	20	20
1	1	0	6	15	16
1	1	1	8	19	20
1	1	1	9	13	11
1	1	1	10	12	14
1	1	1	14	17	17
2	2	0	1	18	18
2	2	0	3	9	10
2	2	0	5	15	15
2	2	0	7	12	14
2	2	1	8	20	18
2	2	1	9	7	7
2	2	1	11	16	16
2	2	1	13	6	8
3	3	0	1	17	18
3	3	0	2	14	14
3	3	0	4	15	15
3	3	0	6	15	15
3	3	1	8	13	12
3	3	1	10	15	15
3	3	1	12	14	15
3	3	1	14	14	12
0	4	0	1	16	16
0	4	0	2	20	19
0	4	0	3	16	18
0	4	0	7	19	17
0	4	1	8	12	13
0	4	1	11	12	11
0	4	1	12	16	16
0	4	1	13	15	14

V	P	S	PM	PUU	PEOU
1	5	0	1	16	16
1	5	0	4	16	18
1	5	0	5	16	16
1	5	0	6	14	16
1	5	1	8	14	13
1	5	1	9	13	12
1	5	1	10	15	16
1	5	1	14	8	8
2	6	0	1	16	16
2	6	0	3	15	13
2	6	0	5	16	13
2	6	0	7	9	9
2	6	1	8	14	11
2	6	1	9	9	10
2	6	1	11	16	14
2	6	1	13	8	9
3	7	0	1	20	20
3	7	0	2	16	16
3	7	0	4	4	8
3	7	0	6	7	8
3	7	1	8	20	20
3	7	1	10	4	4
3	7	1	12	8	8
3	7	1	14	4	4
0	8	0	1	20	20
0	8	0	2	6	6
0	8	0	3	10	11
0	8	0	7	15	16
0	8	1	8	18	20
0	8	1	11	10	12
0	8	1	12	10	9
0	8	1	13	11	12
1	9	0	1	15	20
1	9	0	4	16	20
1	9	0	5	8	11
1	9	0	6	10	13
1	9	1	8	20	20
1	9	1	9	14	14
1	9	1	10	15	13
1	9	1	14	5	5

V	P	S	PM	PUU	PEOU
2	10	0	1	15	18
2	10	0	3	15	15
2	10	0	5	14	16
2	10	0	7	15	13
2	10	1	8	18	19
2	10	1	9	12	15
2	10	1	11	10	14
2	10	1	13	9	6
3	11	0	1	14	15
3	11	0	2	14	15
3	11	0	4	15	14
3	11	0	6	14	14
3	11	1	8	13	13
3	11	1	10	15	15
3	11	1	12	11	12
3	11	1	14	13	15
0	12	0	1	18	19
0	12	0	2	15	15
0	12	0	3	10	11
0	12	0	7	17	17
0	12	1	8	15	14
0	12	1	11	10	9
0	12	1	12	11	12
0	12	1	13	14	13
1	13	0	1	19	20
1	13	0	4	19	20
1	13	0	5	12	12
1	13	0	6	11	13
1	13	1	8	13	10
1	13	1	9	15	16
1	13	1	10	15	16
1	13	1	14	10	8
2	14	0	1	17	19
2	14	0	3	15	17
2	14	0	5	17	17
2	14	0	7	10	11
2	14	1	8	15	15
2	14	1	9	16	16
2	14	1	11	15	16
2	14	1	13	12	16

Table D.7: Results of level of acceptability

Variant	P	S	PM	ICL	ECL	GCL
0	0	0	1	1	4	1
0	0	0	2	2	4	2
0	0	0	3	2	4	2
0	0	0	7	1	4	3
0	0	1	8	2	4	3
0	0	1	11	2	5	4
0	0	1	12	1	4	3
0	0	1	13	2	4	3
1	1	0	1	1	2	1
1	1	0	4	3	4	2
1	1	0	5	1	2	1
1	1	0	6	2	4	3
1	1	1	8	4	4	1
1	1	1	9	4	5	4
1	1	1	10	4	5	3
1	1	1	14	1	4	2
2	2	0	1	2	2	1
2	2	0	3	5	4	4
2	2	0	5	2	2	1
2	2	0	7	3	3	3
2	2	1	8	4	4	2
2	2	1	9	5	5	5
2	2	1	11	2	3	1
2	2	1	13	5	5	4
3	3	0	1	2	2	1
3	3	0	2	3	2	2
3	3	0	4	3	2	1
3	3	0	6	3	2	2
3	3	1	8	3	4	3
3	3	1	10	4	4	2
3	3	1	12	3	3	3
3	3	1	14	3	3	3
0	4	0	1	4	4	3
0	4	0	2	3	3	2
0	4	0	3	2	4	3
0	4	0	7	2	3	1
0	4	1	8	5	5	4
0	4	1	11	4	4	4
0	4	1	12	3	4	3
0	4	1	13	3	4	3

Variant	P	S	PM	ICL	ECL	GCL
1	5	0	1	2	2	3
1	5	0	4	2	2	2
1	5	0	5	2	2	3
1	5	0	6	2	2	2
1	5	1	8	4	3	3
1	5	1	9	4	2	3
1	5	1	10	3	2	2
1	5	1	14	4	3	4
2	6	0	1	2	3	2
2	6	0	3	3	3	3
2	6	0	5	2	3	3
2	6	0	7	3	3	3
2	6	1	8	4	4	3
2	6	1	9	3	4	3
2	6	1	11	3	3	2
2	6	1	13	3	3	3
3	7	0	1	3	1	1
3	7	0	2	3	2	3
3	7	0	4	2	2	2
3	7	0	6	3	3	4
3	7	1	8	3	2	1
3	7	1	10	4	4	4
3	7	1	12	3	3	3
3	7	1	14	3	3	3
0	8	0	1	1	2	1
0	8	0	2	4	5	5
0	8	0	3	3	4	5
0	8	0	7	1	3	1
0	8	1	8	3	3	3
0	8	1	11	4	4	4
0	8	1	12	3	4	3
0	8	1	13	3	4	4
1	9	0	1	2	3	2
1	9	0	4	3	3	2
1	9	0	5	3	3	4
1	9	0	6	3	3	2
1	9	1	8	4	4	2
1	9	1	9	4	4	3
1	9	1	10	4	4	3
1	9	1	14	5	5	5

Table D.8: Results of cognitive load (Part 1)

Variant	P	S	PM	ICL	ECL	GCL
2	10	0	1	2	2	2
2	10	0	3	4	4	2
2	10	0	5	1	2	4
2	10	0	7	1	2	4
2	10	1	8	3	4	2
2	10	1	9	1	2	3
2	10	1	11	1	3	3
2	10	1	13	4	5	4
3	11	0	1	2	3	2
3	11	0	2	4	3	2
3	11	0	4	2	3	2
3	11	0	6	2	3	2
3	11	1	8	4	4	2
3	11	1	10	4	4	3
3	11	1	12	4	4	4
3	11	1	14	3	4	3
0	12	0	1	1	4	2
0	12	0	2	3	4	2
0	12	0	3	3	4	3
0	12	0	7	1	3	1
0	12	1	8	2	4	2
0	12	1	11	2	4	3
0	12	1	12	2	5	3
0	12	1	13	2	4	2
1	13	0	1	3	3	1
1	13	0	4	4	5	1
1	13	0	5	5	5	3
1	13	0	6	4	5	4
1	13	1	8	5	4	3
1	13	1	9	4	5	2
1	13	1	10	5	5	3
1	13	1	14	5	5	4
2	14	0	1	2	3	2
2	14	0	3	3	3	2
2	14	0	5	2	3	2
2	14	0	7	3	4	4
2	14	1	8	4	5	3
2	14	1	9	2	3	3
2	14	1	11	3	4	3
2	14	1	13	3	4	3

Table D.9: Results of cognitive load (Part 2)

Name: Jana Bühler　　　　　　　　Matrikelnummer: 871153

Erklärung

Ich erkläre, dass ich die Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den 09.09.2021 .... Jana Bühler

Jana Bühler