A Work Project, presented as part of the requirements for the Award of a Master's degree in Management from the Nova School of Business and Economics.

EVALUATION OF BUSINESS PROCESSES THROUGH PROCESS MINING TECHNIQUES

Hauke-Christian Uwe Sudmann

Work project carried out under the supervision of:

Professor Paulo Faroleiro

04-01-2021

Abstract

The aim of this thesis is to evaluate the existing sales process of the financial technology startup ABC to develop suggestions for process improvements that are intended to lead to a reduction in overall throughput time. For the evaluation, this thesis uses state-of-the-art process mining techniques, analyzes the common discovery algorithms, and determines the most suitable algorithm for ABC. The discovered process model is extended by performance data to identify bottlenecks and develop suggestions for their elimination.

This thesis concludes that chosen process mining techniques are well suited to evaluate business processes of ABC and develop proposals for process redesign.

Keywords

Business Process Management, Business Process Evaluation, Process Mining, Process Discovery, Process Performance Analysis

This work used infrastructure and resources funded by Fundação para a Ciência e a Tecnologia (UID/ECO/00124/2013, UID/ECO/00124/2019 and Social Sciences DataLab, Project 22209), POR Lisboa (LISBOA-01-0145-FEDER-007722 and Social Sciences DataLab, Project 22209) and POR Norte (Social Sciences DataLab, Project 222)

Contents

1 Introduction

In recent decades, ongoing digitalization and the resulting expansion of information systems have led to tremendous progress in the research field of business process management (BPM). However, the practical implementation of newly developed methods and techniques seems to lag behind. To address this problem, this thesis examines the application of state-of-the-art BPM methods, in particular with the subfield of process mining. Process mining uses historical log data of executed activities to discover processes, check for conformance, or conduct performance analysis.

By applying these methods to the sales process of the [redacted]-based startup ABC, the objective of this thesis is to demonstrate that process mining techniques are well suited to evaluate existing processes at ABC and to develop proposals for process improvements.

ABC finances merchandise orders for online retailers. [redacted] The sales process is crucial, especially for a young, growing company with no established reputation. This process is mainly responsible for the service quality and the turnover of ABC. Unfortunately, the current goal, delivering a contract within 24 hours after receipt of a loan request, has not yet been reached.

To achieve the objective of this thesis, first the current state of research in the field of BPM and process mining is presented, allowing for subsequent application of the gained knowledge to discover the existing process, identify bottlenecks, and develop suggestions for their elimination.

2 Literature Review

2.1 Business Process Management

The roots of BPM have their origin in the rising importance of workflow management (WFM) systems in the 1990s (van der Aalst 2016). BPM consists of methods and tools that combine knowledge from information technology, management sciences, and industrial engineering to improve business processes (van der Aalst, La Rosa and Santoro 2016). Weske (2012) describes BPM as an evolution of the concept of WFM. While the focus of WFM systems is on process automation, BPM considers a larger scope, ranging from “process automation and process analysis to operations management and the organization of work” (van der Aalst 2013, 1). BPM is based on the observation that each product provided to the market by a company is the outcome of a number of activities performed. Business processes are the key instrument used to organize these activities and to improve the understanding of their interrelationships (Weske 2012).

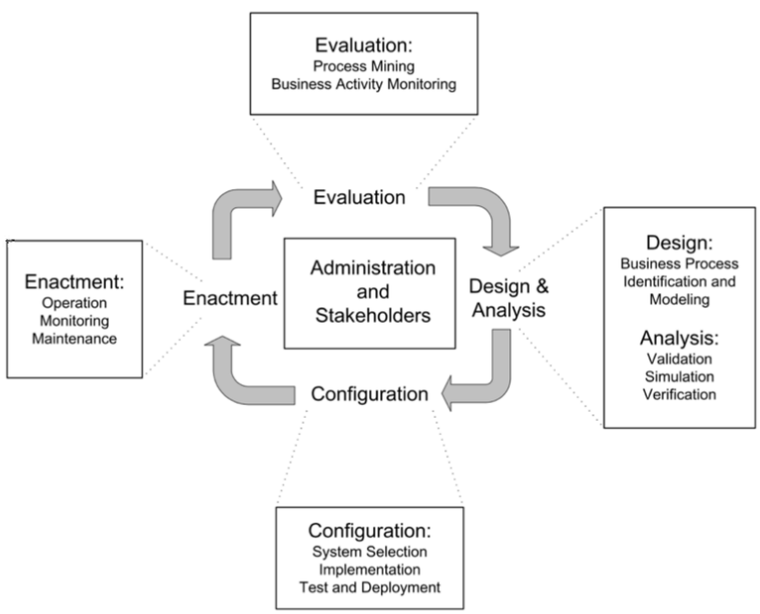
Van der Aalst et al. see a gap between state-of-the-art BPM technologies and actual BPM practices in use (2016). A fundamental problem is the method of collecting data, which are the basis for process analysis and evaluation. Data on process flows were typically collected manually, for example, through surveys or interviews (van der Aalst, La Rosa and Santoro 2016). To eliminate the inaccuracies that are common with this method, the exploration of event data from automatically generated log data can more appropriately link BPM with process improvements. The advantage is “that any insight extracted from this data is based on evidence, rather than on human confidence, and thus is a more accurate representation of reality” (van der Aalst, La Rosa and Santoro 2016). Such techniques to extract and analyze process data from event logs belong to the field of process mining, which is further discussed in Chapter 2.3.

To gain an overall understanding of the disciplines involved in BPM, the framework of the business process lifecycle (BPL) is used (Weske 2012). The BPL (Figure 1) consists of four interrelated phases: *design and analysis*, *configuration*, *enactment*, and *evaluation*. The lifecycle begins in the *design and analysis phase*, in which business processes are identified, reviewed, validated, and visualized through business process models. A core competence of this phase is the research area of business process modeling. On the basis of the models developed and, potentially, the simulations conducted, the stakeholders then decide which processes are to be taken to the next phase, the *configuration phase*, where the newly designed processes are then implemented. This is accomplished by introducing policies according to operations done by the key users and by adapting the technical systems to the new organizational environment and testing them.

Once the configuration is done, BPL moves to the *enactment phase.* Here, processes are executed, monitored, and maintained.

The obtained data can then be used in the *evaluation phase*[1] to assess the implemented process models and develop improvements. Process mining can play an important role in this phase. Based on the evaluation results, new process models are created, or existing models are adapted, and thus the lifecycle begins again with the design and analysis phase (Weske 2012).

FIGURE 1 - BUSINESS PROCESS LIFECYCLE (WESKE 2012)



[1] in other literature also named diagnosis/requirements phase (van der Aalst 2016, 31)

2.2 Business Process Evaluation

In the context of the BPL, this thesis focuses on the fields of evaluation and the application of process mining techniques. As already mentioned, business process evaluation uses the information collected to analyze and assess an existing process. The objective is to obtain information about the quality of the process to provide relevant input for the redesign. This input should support the four main dimensions of redesign measures: *time, cost, quality,* and *flexibility* (Reijers and Liman Mansar 2005). For example, information relevant to the redesign is the identification of bottlenecks, main cost drivers, or high error rates.

Van der Aalst (2016) stated that the evaluation phase "is not supported in a systematic and continuous manner […] and factual information about the current process is not actively used in redesign decisions" (31). As mentioned earlier, data used for the evaluation originally were based on surveys, interviews, and the manual measurement of times. Much more reliable data can be used today when evaluating a process, with such data recorded by operational tools in a so-called event log. Typical information about an event includes, for example, the start and completion time of an activity and the resource executing this activity. To bring this data into a relevant context for the redesign of business processes, the research area of process mining can be applied.

2.3 Process Mining

Process mining is a relatively young research area; it uses elements from process modeling and analysis and from data mining. Process mining begins by recording data on real processes in an event log to allow for various analyses of the data at a subsequent time. The goal of process mining is to discover, monitor, and improve processes (van der Aalst 2016).

Process mining can be divided into three types: *discovery, conformance,* and *enhancement. Discovery* techniques allow creation of a process model from existing log data without prior knowledge to visualize the actual processes (Wen, et al. 2007). *Conformance* compares an existing process model with the event log and vice versa. Deviations can be detected between the intended process model and the actually executed processes. Likewise, conformance to certain rules, such as review policies, can be investigated. The *enhancement* process mining type focuses on the improvement and extension of process models generated from a log. An extension can be the addition of performance data to the model to detect bottlenecks, service levels, and frequencies (van der Aalst 2016).

2.3.1 Event Log

An event log is usually generated in so-called process-aware information systems (PAIS), for instance, in WFM systems (Staffware, COSA), enterprise resource planning systems (SAP, Oracle), or customer relationship management systems (PeopleSoft, Baan) (van der Aalst, Weijters and Maruster 2002; van der Aalst 2016). Log data also can be generated by commercial web applications, or self-developed operational tools, which at least hold the possibility of enlarging them to accomplish this purpose. The main challenge is merging data from different tools and systems.

To apply process mining, an event log must meet some requirements (van der Aalst, Weijters and Maruster 2002). These are that:

i) Each event refers to a task (i.e., a well-defined step in the workflow);
ii) Each event refers to a case (i.e., a workflow instance); and
iii) Events are totally ordered.

For example, a case is initialized when a customer requests a product order. To service the customer's request, employees perform several tasks, which must be linked to the case (i.e., product order) and recorded in the order in which they are performed. Beyond that, event logs can contain additional data, for example, transaction types that specify whether an occurring event is *scheduled, started, completed*, *suspended*, or *resumed.* Additionally, timestamps or information about executing users can be recorded (van der Aalst, van Dongen, et al. 2003).

2.3.2 Process Discovery

Process discovery aims at deriving a process model from an event log without prior knowledge (Wen, et al. 2007) and can be generated in traditional modeling languages such as Petri nets or business process modeling language (BPML). Additionally, process trees, flexible nets, or yet another workflow language (YAWL) are languages to express process models. The essential elements of process languages are nodes and arcs. The arcs are the connections between the nodes and represent the process flow. In a Petri net, a node can either be a transition (activity) or a place. The latter determines the process behavior (*detailed description of Petri nets in Appendix A.5.1*). In a process tree, the nodes are divided into branch node (activity) and lead node (operator) (Buijs, van Dongen and van der Aalst 2012). An operator determines the process behavior and represents either an exclusive choice, sequential composition, interleaved parallel composition, or loops (Leemans, Fahland and von der Aalst 2013).

Over the past few years, significant progress has been made in the development of discovery algorithms. This chapter explains which criteria are used to evaluate discovery methods for discussion of the most prominent ones.

The challenge of process discovery is to identify the model that best describes the event log. In this context, a model created by a discovery algorithm must balance four quality criteria, namely: *fitness, precision, generalization and simplicity* (Weijters and Ribeiro 2011):

- Fitness: The discovered model should allow for the behavior seen in the event log.
- Precision: The discovered model should not allow for behavior completely unrelated to what was seen in the event log.
- Generalization: The discovered model should generalize the example behavior seen in the event log.
- Simplicity: The discovered model should be as simple as possible.

A reliable process discovery algorithm can create a representative process model from an event log, which reflects the behavior in the log. Ideally, a discovered model allows the replay of all traces, which is expressed by the term *fitness*. However, this can also lead to *imprecision* models, which allow for too much behavior, or models that are just not *simple* to read, since they exhibit too many transitions and arcs. Hence, these criteria are interdependent. Although the criteria are of qualitative nature, techniques have been developed to express *fitness*, *precision* and *generalization* quantitatively on a scale of 0–100% to compare different models. An approach of determining these values is given by van der Aalst, Adriansyah und van Dongen (2012).

Another important statement about a process model is *soundness*. A process model is *sound* if every activity can be executed, and the end marker (last node) can always be reached (Leemans, Fahland and von der Aalst 2013). Consequently, *soundness* is expressed as either *sound* or *not sound*.

One problem that process discovery must face is noise, which is not easy to properly detect. Noise refers to incorrect or unrepresentative instances that are present in the event log and thus bias the discovered model (Wen, et al. 2007). To address the noise problem, filters can be used to eliminate abnormal behavior (outliers). Some algorithms already make use of such filters to reduce noise.

2.3.3 Process Discovery Algorithms

In the following, four common algorithms are presented and classified according to the quality dimensions already mentioned.

Alpha Miner. The alpha miner (AM) was one of the first process mining algorithms that could handle concurrency (Wen, et al. 2007); in other words, it could detect whether events happened one after the other or in any order. In its original version, the AM has a relatively simple structure. To understand it in more detail, Appendix A.5.2 explains the basic approach (and common process patterns) of the AM, and, therefore, of process mining. In short, the AM analyzes the event log using "directly-follows" relations of events. The algorithm goes through the whole event log, case by case, and analyzes the sequence of events (activities) (van der Aalst 2016).

Although the AM can handle parallelism, it cannot handle short-loops and self-loops, in other words, self-repeating events or two events that are repeated together several times. This problem has been fixed with the alpha miner + (AM+) extension (de Medeiros, et al. 2004).

With the development of the alpha miner ++ (AM++), it has become possible to correctly visualize implicit dependencies that may result from non-free decisions. Implicit dependencies are dependencies between events that are not directly related, in other words, those that do not occur in direct sequence. An example of implicit dependencies caused by a non-free-choice can be found in Appendix A.5.3.

Nevertheless, the AM++ encounters its limitations in certain cases. For example, self-loops involved in implicit dependencies cannot be observed. Furthermore, the alpha miners (all) are not capable of handling noise and incompleteness. Therefore, the alpha miners (all) will reach their limits with more complex real-life logs and will no longer meet the quality criteria of generalization and simplicity. Other approaches, such as the heuristic or fuzzy miner, address these problems.

Heuristic Miner. The heuristic miner (HM) has similarities with the AM in its approach to discover an event log. A significant advantage of the HM is the ability of detecting long-distance dependency or implicit dependencies, that is done by analyzing all inputs of an activity (each activity that happened upstream) (Weijters, van der Aalst and de Medeiros 2006). The order of events is determined by a so-called dependency graph, which, in contrast to the directly follows relation of the alpha miner, also considers frequencies of events. This allows for excluding infrequently occurring paths (van der Aalst 2016) which leads to a more robust model that can better handle noise and, therefore, work more usefully with complex real-life logs. The filtering function, of course, comes with a slightly lower fitness level, because some paths of the event log are not represented by the model anymore. Nevertheless, a higher level of simplicity and generalization compensates for this effect overall. However, even the HM does not promise sound models.

Fuzzy Miner. The fuzzy miner is most similar to the heuristic miner, and also reduces the number of infrequently occurring paths. A further advantage is an adaptive simplification approach, which uses data clustering and graph clustering to simplify the model by grounding connected and infrequently occurring nodes into one. These clustered nodes can be understood as a subprocess in the final model (van der Aalst and Günther 2007; van der Aalst 2016).

The goal of the fuzzy miner is to create an understandable process model that contains high-level information. This approach is particularly beneficial for more complex processes, where conventional algorithms such as the AM or the HM create so-called spaghetti models (i.e., models with many arcs) that may be correct, but are far too detailed and difficult or impossible for the user to understand.

However, the simple and general fuzzy miner model is, of course, in conflict with the other two criteria: fitness and precision (van der Aalst and Günther 2007). Furthermore, the fuzzy miner uses its own processing language that is not convertible into common languages. Therefore, a model created by the fuzzy miner cannot be extended with additional plug-ins (e.g, conformance/performance plug-ins).

Inductive Miner. Inductive mining (IM) is currently one of the leading process discovery approaches due to its flexibility, formal guarantees, and scalability (van der Aalst 2016). The basic IM algorithm was first mentioned by Leemans et al. in 2013. Unlike the other algorithms, IM creates a process tree based on directly-follow relations (van der Aalst 2016). The log is subdivided into so-called sub-logs, which are treated as individual logs. Based on the process tree operators (see 2.3) within the sub-logs, further sub-logs are created until only single branch nodes (activities) remain. (Leemans, Fahland and von der Aalst 2013).

Afterward, the process trees can be converted into other modeling languages, such as BPMN or Petri nets. The advantage of using process trees is that sound processes models can be guaranteed due to their characteristics. Furthermore, when using IM in its basic form, perfect fitness can be assumed, meaning that any trace observed in the event log can be replayed in the model. Moreover, the basic IM can discover a much wider class of processes in situations where the alpha-algorithm and many other algorithms fail (van der Aalst 2016).

Additionally, the inductive miner infrequent (IMi) extension can filter out infrequent behavior. The various filtering methods used are more sophisticated than those of the heuristic miner, although in some respects they are similar (van der Aalst 2016). Once again, it should be noted that filtering reduces fitness but increases precision while maintaining the same generalization and simplicity, as Leemans et al. (2013) have observed, for the comparison between IM and IMi.

Furthermore, using the Pareto principle, the inductive miner can generate so-called 80/20 models where "typically, 80% of the observed behavior can be explained by a model that is only 20% of the model required to describe all behavior" (Leemans, Fahland and von der Aalst 2013, 11). Leemans et al. further challenge this model with an 80/20 model created by the HM and conclude that the HM model is less simple and not sound compared to a sound and more simple IMi model. Additionally, their testing concludes that the IMi always returns a sound 80% model quickly and scores well on all quality criteria except *precision*.

2.4 Performance Analysis

The majority of process mining techniques focus on process discovery and process conformance/compliance checking. However, the extraction of performance data from event logs is an equally crucial aspect, especially in the evaluation phase of the BPL. This enables a more detailed understanding of the process flows, which can be used to effectively improve processes (Li, van Zelst and van der Aalst 2019).

Flapper et al. define performance as “the way the organization carries its objectives into effect” and make organizational “success and continuity” dependent on it (1996, 1). An important factor, and typically a starting point to achieve this goal, is the identification of bottlenecks and frequent errors (Bentley and Davis 2009). The definition of a bottleneck varies in the literature. This thesis uses Mukherjee and Chatterjee's general definition: "A bottleneck constrains the performance of a system" (2006, 5). Consequently, when referring to bottlenecks, the activities that constrain the overall process the most are meant. Possible solutions to eliminate bottlenecks can be an increase in resources (e.g., more employees for an activity) or a simplification of the activity, which results in less processing time. Similarly, reducing variability in arrival and processing time can lessen the capacity utilization of an activity (Ivanov, Tsipoulanidis and Schöneberger 2019).

The prerequisite for this is to have performance data of the existing process. These data can be also found in an event log, if it provides timestamps of the events. With the obtained data performance metrics such as service level, the throughput time and frequencies can be calculated to identify bottlenecks (van der Aalst, Adriansyah and van Dongen 2012).

The performance analysis can be divided into a log-based (data-driven) approach and a model-based (process driven) approach (Song and van der Aalst 2007). The log-based approach considers only the event log without prior knowledge about a process model. This approach includes data visualizations such as dotted chart analyses, which enable an aggregated understanding of the process (Song and van der Aalst 2007).

The model-based approach evaluates performance data from an event log using a predefined or discovered process model. The advantage is that bottlenecks can be identified more easily and placed in the context of the overall process (Li, van Zelst and van der Aalst 2019). However, for this approach, a process model that accurately reflects the behavior in the event log is necessary; otherwise, important aspects of performance cannot be detected (Song and van der Aalst 2007).

3 Problem Statement

In recent years, ABC has constantly changed its processes. On the one hand, the process flow has evolved as a result of an increase in customer orders, enlargement of the team, and collaboration with partner companies. On the other hand, the steps within activities have also changed, mostly due to technological advances. New processes were introduced where the need was believed to be the most urgent. However, these decisions were usually not based on data, but rather on the subjective perception of the employees.

The goal of delivering a contract within 24 hours after the customer's loan request has not yet been achieved. According to internal statements, it is technically possible but it cannot yet be implemented to such an extent that almost every contract is sent within this time frame.

Some process charts are present, but most are outdated. Furthermore, a detailed analysis of process times and bottlenecks is missing, which would have been helpful for an appropriate prioritization of process improvements to come closer to the goal of 24-hour contract delivery.

4 Research Objective

The objective of this thesis is to evaluate the sales process of ABC by applying BPM approaches and process mining techniques in order to bring the company closer to the goal of 24-hour contract delivery. The chosen approaches aim to identify bottlenecks and develop optimization proposals.

5 Methodology

This chapter explains the methods used to achieve the research objective. First, an event log is created, which, in the next step, is discovered by process mining algorithms and visualized in a Petri net model. The reason for the use of Petri nets is that, in accordance with Leemans et al. (2013), it enables a comparison of the quality criteria of the algorithms. Subsequently, bottlenecks in the process and their sources are explored through a performance analysis and semi-structured interviews to develop and validate recommendations for process improvement.

5.1 Building an Event Log and Collecting Data

Since an event log did not exist and not all operational tools provide access to log data, the workflow automation tools Zapier (Appendix A.4.1) and n8n (Appendix A.4.2) were used to build mechanisms that record the execution of activities as so-called events. The *case id*, *event id*, *activity name*, *timestamp, transaction type*, and *resource* are recorded for each event. Transaction types can have either the value "*complete*" or "*start*" (Appendix A.4.3), whereas start times cannot be recorded for every activity.

5.2 ProM

ProM is a free and open-source process mining software that supports a wide variety of process mining techniques (ProM 2020). The main use of the software is of an academic nature, because it can be easily extended by plug-ins that allow for implementation and testing of new or modified algorithms (van Dongen, et al. 2005). In this thesis, existing filter, discovery, conformance, and performance plug-ins of the ProM framework are used.

5.3 Process Discovery – Algorithm Analyses

To find the best-fitting process model of the sales process, the algorithms described in Chapter 2.3.3 are challenged against the event log of ABC, except of the fuzzy miner. To accomplish this, the corresponding ProM mining plug-ins with their default settings are used. The previously introduced quality criteria (*fitness, simplicity, precision, and generalization*) and the *soundness status* are determined to compare the models. The calculations of *fitness, precision,* and *generalization* are done by the conformance plug-ins *PNetReplayer* (Verbeek 2010) and the *PNetAlignmentAnalysis* (Adriansyah 2012) along with the determination of simplicity through qualitative measures.

5.4 Performance Analysis Approach

The aim of the performance analysis is to identify weaknesses and bottlenecks in the sales process to allow for subsequent development of proposals for improvements. To analyze the performance of the sales process, enhancement techniques are used. First, the so-called model-based approach is applied by extending the previously discovered model with process times. Based on these, global statistics are calculated for the respective sub-processes, such as the *average throughput time*, as well as key figures for individual activities, for example, *average*, *median*, *quartiles*, *standard deviation*, *minimum value,* and *maximum value*. Using these statistics, bottlenecks are filtered out and prioritized according to their importance. Subsequently, the causes of the bottlenecks are identified. For this purpose, methods of the so-called log-based approach are used to visualize activity sequences directly from the log with the help of dotted charts, followed by semi-structured interviews to gain further insights.

6 Results

In this chapter, the results of the process evaluation are presented. The first part covers the discovery result and the second part the performance analysis. The underlying event log contains data over the observation period of two months, in which 1,779 events and 260 cases for 16 activities were recorded (see Appendix A.6 for activity description).

6.1 Process Modell – Discovery Algorithm

To compare the meaningfulness of process models, the event log is filtered once again. Only complete cases (i.e., cases that match predefined start and end points) and only events with the transaction type *complete* (since, for the moment, only the sequence of events is interesting) are taken into account. Since the sales process ends either with the activity *check signed contract* or *deal lost*, an artificial end event must be added to the cases, since not every miner does this automatically and the existence of a single start and end event is a prerequisite for the calculation of *fitness*, *precision,* and *generalization* (Leemans, Fahland and von der Aalst 2013). Consequently, the filtered log contains 51 cases and 676 events. On average, 13 events were recorded per case. The results of the analysis with the *PNetReplayer package* and the *PNetAlignmentAnalysis package* are indicated in TABLE 1. The corresponding Petri net process charts are indicated in Appendix A.10. First, it is noticeable that only the IM and IMi create a sound process model. Moreover, the model of IMi seems to be the easiest to understand, after that of the HM. In contrast, the alpha miners (all) are not so simple.

TABLE 1 - RESULTS MINER ANALYSIS

Algorithms[2]	*Fitness*	*Precision*	*Generalization*	*Soundness*
Alpha Miner (AM)	87.51%	15.88%	97.53%	No
Alpha# (A#)	85.47%	29.36%	98.80%	No
Inductive Miner (IM) [3]	100.00%			Yes
Inductive Miner infrequet (IMi)	90.55%	89.38%	99.42%	Yes
Heuristic Miner (HM)	93.80%	86.99%	99.71%	No

The basic AM and AM# did not generate sound models and, furthermore, have lower values for fitness, precision, and generalization compared to the HM and the IM. The reason for this is the existence of short loops in the event log, which can be seen in the fuzzy miner model, for example. In summary, the alpha miner and its extensions are not suitable for exploring the event log of ABC. As expected, due to its characteristics, the IM generates the best result in terms of fitnesss, but this is accompanied by a loss of simplicity, as seen in the process model in Appendix A.10.5.

As a result of the filter functions of the IMi and the HM, these models do not have 100% fitness. However, with more than 90%, they still have useful value (compared to results of Leemans, et al., 2013). Comparing the two algorithms reveals a striking correlation between fitness and precision. The value for fitness is higher for the HM by almost as much as the value for precision is lower. No significant difference can be observed for generalization.

[2] The PNet Paplayer plug-in was not able to calculate results for the AM+ and AM++

[3] PNetAlignmentAlaysis plug in was not able to get results for IM

On closer analysis, both the IMi and the HM demonstrate undesirable weaknesses (for details, see Appendix A.7). These include, in particular, the treatment of parallelism and loops, resulting in imprecise process models that allow too much behavior. As a result of the detailed analysis in Appendix A.7 and discussions with key users, the undesirable paths were identified. With the help of a parameter adjustment ("noise threshold" from 20% "default" to 25%), a new model was created with the IMi (FIGURE 23). This model is even simpler, more sound, and represents the most common and desired process behavior. Therefore, it is used for the performance analysis.

6.2 Performance Analysis

First, the overall process is divided into two sub-processes to exclusively consider the internal performance and exclude customer response time. The first process begins with the offer request (by the customer) and ends with the upload of the offer (output). Henceforth, this process is titled *offer process* (FIGURE 2). The second process ranges from the submission of the application (by the customer) to the upload of the contract (output) and is titled *contract process* (FIGURE 3).

FIGURE 2 - OFFER PROCESS

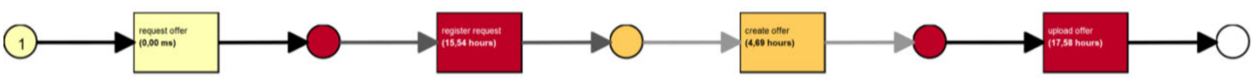
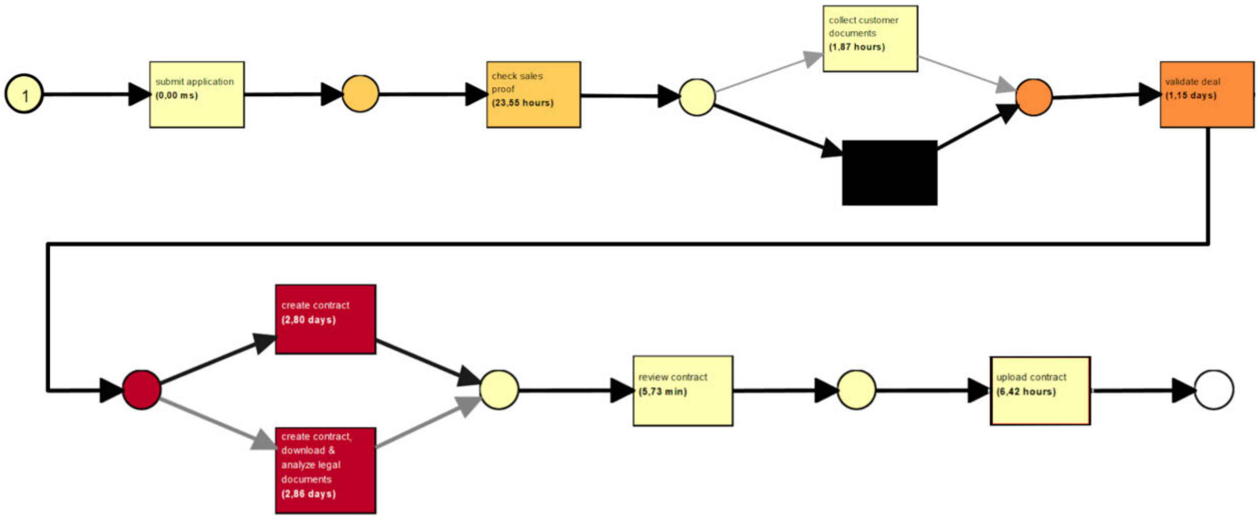


FIGURE 3 - CONTRACT PROCESS



The figures also include the average sojourn times calculated by the ProM plug-in *Replay a Log on Petri Net for Performance/Conformance Analysis*. The average case throughput time of the *offer process* is [redacted] hours and of the *contract process* [redacted] days (Appendix A.8), which is cumulatively far above the target of [redacted] hours. Since the event log does not contain schedule transaction types, it is assumed that the completion of the previous activity schedules the next one.

6.2.1 Identifying Bottlenecks

In the following, the bottlenecks of the process are determined using qualitative measures and statistical parameters. Analogously to the process discovery, the same filtering measures are applied to the event log, with the exception that first the event log is divided into the sub-processes and then the filters are applied. This approach has the advantage of considering more cases. Consequently, cases in the *offer process* that have not yet completed the *contract process* are considered. Likewise, cases that have completed the contract process – but data for the *offer process* have not yet been recorded because they took place before the observation period began – are also taken into account.

In addition to the average sojourn time from FIGURE 2 and FIGURE 3, TABLE 7 and TABLE 8 reveal further statistics for the sojourn time of the activities calculated with Google Spreadsheets.

Considering the tables, it is remarkable that there is an extremely high variance at each activity of the process. Therefore, the median and quartiles also provide important information about the dispersion of the sample. As the median is mostly significantly lower than the average, the latter is heavily biased by extreme outliers. Such outliers occur, for example, as a result of missing transaction types, such as suspended or resumed. Since a closer examination of all activity would exceed the extent of this work, the two most time-consuming activities are filtered out and prioritized according to qualitative and quantitative characteristics.

TABLE 2 - PRIORITIZING BOTTLENECKS FOR DEEPER ANALYSIS

Activity	#	*Reason*
create contract (+ dow. & ana. leg. doc.)	1	highest mean, high mean compared to low median, highest std. dev, extremely low quartile 1 compared to quartile 3, high frequency
validate deal	2	highest median, highest 1. quartile, quartile 3 lower than average

6.2.2 Analyzing Bottleneck Symptoms

Using the ProM plug-in dotted chart, the completion time of activities is projected on the observation period (FIGURE 14 & FIGURE 15). The linking between the dots can be interpreted as the sojourn time of the ending activity (the activity to the right). The case identifiers are displayed on the Y-axis and the observation period on the X-axis. The knowledge gained from the dotted chart is discussed with key users to identify the source of the bottlenecks. Furthermore, where possible, a division between waiting time and processing time is made.

Create Contract (+ Download & Analyze Legal Documents). Since the only difference between *create contract* and *create contract, download & analyze legal documents* is that an additional step is taken in the latter, they are examined together. For these activities, 66 and 13 observations are available, respectively. On average, [redacted] contracts are created for one case.

When looking at the dotted chart (FIGURE 14), one can see an accumulation of contract activity up to September 28, all of which was abruptly executed on that day. After this day, a significantly lower average sojourn time is noticeable. According to key users, the reason for this backlog of activities was a [redacted]

Additionally, the technical implementation of the new contract templates took a certain amount of time. Consequently, the data up to September 28 are not representative of the process times for these activities. In this context, TABLE 3 divides the sojourn statistics into two periods.

TABLE 3 - CONTRACT CREATION STATISTICS

	n	*Average*	*Median*	*Std. dev*	*min*	*max*	*Quartile 1*	*Quartile 3*
since 28.09.20	[redacted]							
before 28.09.20								

(Time in hours)

The differences are clearly visible in each value. Additionally, for the contract activities, start times are recorded in the event log that enable the sojourn time to be divided into waiting time and processing time (TABLE 4). It can be seen clearly that the largest part of the sojourn time consists of the waiting time, which is especially evident for the data before September 28. Nevertheless, even from September 28 onward, the average processing time of approximately [redacted] hours is still too high compared to the median processing time of [redacted] minutes.

In summary, the problems with this activity are the outliers in the average processing time and the number of repetitions.

TABLE 4 - CREATE CONTRACT TIMES

	#	*Average (hours)*	*Median (hours)*
Waiting time	[redacted]		
Processing time			
Sojourn time			

Before 09-28 in parentheses

Risk Validation. This activity is executed once per case. In 77 cases, it was scheduled by the activity *check sales proof* and in three cases by *collect customer documents*. No significant variation in sojourn times is discernible over the observation period (*see* FIGURE 15), which indicates a more structural problem of activity. Since no information is available on the start time, the processing times were obtained from the key users in interviews. For a new customer, the activity takes, on average, about [redacted] minutes and for an existing customer, about [redacted] minutes. Furthermore, the number of products to be financed has an influence on the processing time. For example, the key agent requires [redacted] base minutes for a new customer and [redacted] base minutes for an existing customer. For each product that is included in the project (case), an additional minute is spent. Therefore, a certain variance in the processing time is inevitable. Nevertheless, this activity with approximately [redacted] hours median sojourn time has a too high value, which is mostly caused by the lengthy waiting time.

6.2.2 Bottlenecks Summary and Solutions

In the following table, the findings of the bottleneck analysis are matched with possible solutions. The latter were discussed with key users and the information technology (IT) team to validate their effectiveness and efficiency. Additionally, the feasibility was classified into a time horizon using the terms *short-term*, *medium-term* and *long-term*.

TABLE 5 - BOTTLENECK SUMMARY

Bottleneck	Priority	Problem	Possible Solution	Problem Owner
Create contract	2	[redacted]	[redacted]	• IT • Product (customer education)
Download & analyze legal documents	3	• [redacted]	• [redacted]	• IT
Validate deal	1	[redacted]	[redacted]	• HR • Operations • IT

The discussion with key users and the IT department has revealed that if the proposed solutions are implemented, the current average throughput time ([redacted] hours[4]) of the *contract process* can be reduced by up to 42% in the long term. This corresponds to a reduction in the average throughput time of the entire sales process by approx. 27%. TABLE 6 outlines these three perspectives in comparison to the current process times. The calculation of these predictions is described in more detail in Appendix A.8.4.

[4] This value has been calculated with respect to the change in the sojourn time of the "create contract" activity. Thus, an average value of [redacted] hours was taken for the mentioned activity.

TABLE 6 - PROSPECTS ON PROCESS TIMES

	now	*Short-term*	*Medium-term*	*Long-term*
Validate deal	THIS SECTION WAS MARKED CONFIDENTIAL AND IS THEREFORE OMITTED [redacted]			
Create contract				
Contract sub-process				
Sales process				

(Time values in hours; In parentheses, percentage reduction compared with now)

7 Conclusion

This thesis first provided an overview of BPM areas to explain the research focus for this thesis, namely, business process evaluation. In this context, the research field of process mining was introduced. Subsequently, the acquired theoretical knowledge was applied to the practical case of ABC to pursue the research objective of using process mining techniques to improve the sales processes at this startup.

For this purpose, the common discovery mining algorithms were first analyzed on the event log of ABC to determine which algorithm is best suited to model the sales process. The analysis, following Leemann et al. (2013), concluded that the IMi yielded the best outcome. The resulting model was discussed with key users. Except for unnecessary loops due to errors or change requests by customers, the explored model represented the process exactly as it was intended to be.

It has turned out that an optimization of the sequence of activities is not possible due to the product structure. Therefore, to identify bottlenecks in this model, it was extended by adding process times. This made it possible to carry out a performance analysis, which identified the two most significant bottlenecks in the process, namely, the activities *create contract* and *validate deal*.

An analysis of the causes of the bottlenecks revealed that the main driver for the long sojourn times was the lengthy waiting period. To confront this problem, it was suggested that more resources be allocated to these activities to raise capacity. Furthermore, an automatic reminder system should be developed to ensure that specific time targets are better met.

In addition to the long waiting time, the *create contract* activity also had an extremely long average processing time, although this task is mainly automated. Through interviews with key users, it was possible to determine that the main reason is the high error rate of the automated function. This problem was assigned to the IT department for troubleshooting and can presumably be remedied within the short term.

Furthermore, it was found that, for some activities, too many operational tools were required. On the one hand, this causes unnecessary effort during processing, and on the other hand, it increases the complexity of the handover process. The latter can lead to a situation where executed activities are not marked as completed on the system side, and, therefore, the processing period in the log is not yet completed. Likewise, the subsequent activity is not scheduled. The solution for this is to reduce the operational tools in use; however, according to the IT department, this is more of a long-term project. The complete list of all identified problems and proposed solutions can be found in TABLE 5. Moreover, the table includes the validation of the proposed solutions in terms of efficiency, feasibility, and time. A rough estimate of time savings through the implementation is provided in TABLE 6. According to this, the current throughput time of the sales process could be reduced in the long term by approximately 27%, from [redacted] hours, on average, to [redacted] hours, if all improvement suggestions were implemented.

This result clearly indicates that the research objective has been achieved, and the application of process mining techniques can definitely help improve the sales process at ABC. Nevertheless, the proposed improvements are not enough to reach the goal of completing contracts within 24 hours, but they take ABC a significant step further toward this goal.

To ensure that the scope of this work was not exceeded, only the two most essential bottlenecks were investigated in detail. The recommendation is to continue this work by analyzing all activities in a similar pattern. Based on the collected findings, the next phase of the BPL – namely, the redesign phase – can be started, with the objective of implementing all suggestions into the current model to move closer to the 24-hour goal. Once again, at this point, process mining techniques can become quite useful to validate the quality of the redesign actions through pre- and post-analysis.

In summary, this work has demonstrated the importance of using process mining techniques for process evaluation and redesigns and should encourage the use of process mining in practice.

8 Limitations and Recommendations

One limitation of this research is the availability of transaction types. The lack of start times for most activities made it impossible to divide the sojourn time into waiting and processing times. Furthermore, transaction types, such as *suspend* and *resume*, could have prevented some outliers, since the suspension of an activity likely led to these outliers. Reworking the tracking and recording of activities should be considered to address the problem of outliers.

Another challenge was to find the sources of the bottlenecks in the event log. To overcome this problem interviews with the key users were necessary. Specifying the event log more precisely by adding sub-tasks within an activity could provide more information about the source of bottlenecks, thereby validating employees' statements or not even having to interview them.

A further limitation is the neglect of office hours in the performance analysis, with weekends or holidays especially causing extreme outliers (see Appendix A.9). To overcome this limitation, the consideration of office hours in performance analysis plug-ins could be added to ProM framework.

Due to the limit of this work, only a time-driven approach of performance was considered. The cost-driven approach mentioned in the literature chapter would be of interest for future research.

Finally, the performance plug-in of the ProM framework was limited in its statistical output. Adding further statistical parameters, such as medians and quartiles, could make the performance analysis more precise.

Bibliography

Adriansyah , Arya. 2012. *Replay a Log on Petri Net for Conformance Analysis Plug-in.* October 2012.

Bentley, William, and Peter T. Davis. 2009. *Lean Six Sigma Secrets for the CIO.* 1st Edition. CRC Press.

Buijs, J.C.A.M., B.F. van Dongen, and W.M.P. van der Aalst. 2012. "A Genetic Algorithm for Discovering Process Trees." *IEEE World Congress on Computational Intelligence* 1-8.

David, Rene, and Hassane Alla. 2005. *Discrete, continuous, and hybrid Petri nets.* Berlin Heidelberg: Springer.

de Medeiros, A.K.A., B.F. van Dongen, W.M.P. von der Aalst, and A.J.M.M Weijters. 2004. "Process Mining for Ubiquitous Mobile Systems: An Overview and a Contrete Algorithm." *Ubiquitous Mobile Information and Collaboration Systems* 156-170.

Flapper, Simme Douwe P. , Leonard Fortuin, and Paul P.M. Stoop. 1996. "Towards consistent performance management systems." *nternational Journal of Operations & Production Management.*

Ivanov, Dmitry, Alexander Tsipoulanidis, and Jörn Schöneberger. 2019. *Global Supply Chain and Operations Management.* Vol. 2. Springer.

Leemans, Sander J.J., Dirk Fahland, and W.M.P. von der Aalst. 2013. "Discovering Block-Structured Process Models From Event Logs Containing Infrequent Behaviour." *International conference on business process management* 66-78.

Li, Chiao-Yun, Sebastiaan J. van Zelst, and W.M.P. van der Aalst. 2019. "A Generic Approach for Process Performance Analysis using Bipartite Graph Matching." *International Conference on Business Process Management* (Springer) 199-211.

Mukherjee, S., and A.K. Chatterjee. 2006. *Unified concept of bottleneck.*

ABC. 2020. Accessed December 28, 2020. https://www.ABC.com/en/about-ABC.

n8n. 2020. August 24. Accessed November 1, 2020. https://docs.n8n.io/#what-is-n8n.

ProM. 2020. *http://www.promtools.org/doku.php.* Accessed November 20, 2020.

Reijers, H.A., and S. Liman Mansar. 2005. "Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics." *Omega* 33 (4): 283-306.

Song, M., and W.M.P. van der Aalst. 2007. "Supporting process mining by showing events at a glance." *Proceedings of the 17th Annual Workshop on Information Technologies and Systems (WITS)* 139-145.

van der Aalst, W.M.P. 2013. "Business Process Management: A Comprehensive Survey." *ISRN Software Engineering* 1-37.

van der Aalst, W.M.P. 2016. *Process Mining.* 2nd Edition. Berlin Heidelberg: Springer.

van der Aalst, W.M.P., A.J.M.M. Weijters, and L. Maruster. 2002. *Workflow mining: Which processes can be rediscovered.* Eindhoven: BETA Working Paper Series, WP 74, Eindhoven University of Technology.

van der Aalst, W.M.P., and C.W. Günther. 2007. "Fuzzy mining–adaptive process simplification based on multi-perspective metrics." *International conference on business process management* (Springer) 328-343.

van der Aalst, W.M.P., Arya Adriansyah, and Boudewijn van Dongen. 2012. "Replaying History on Process Models for Conformance Checking and Performance Analysis." *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 2 (2): 182-192.

van der Aalst, W.M.P., B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. 2003. "Workflow mining: A survey of issues and approaches." *Dara & Knowledge Engineering* 47: 237-267.

van der Aalst, W.M.P., Marcello La Rosa, and Flavia Maria Santoro. 2016. *Business Process Management: Don't Forget to Improve the Process!* Springer Fachmedien Wiesbaden.

van Dongen, B.F., A.K.A. de Medeiros, H.M.W. Verbeek, A.J.M.M. Weijters, and W.M.P. van der Aalst. 2005. "The ProM Framework: A New Era in Process Mining Tool Support." *International conference on application and theory of petri nets* (Springer).

Verbeek, H.M.W. 2010. *trac - integrated SCM & Project Management.* October 8. Accessed 10 10, 2020. https://svn.win.tue.nl/trac/prom/browser/Documentation/Package%20PetriNet Replayer.pdf.

Weijters, A.J.M.M., and J.T.S. Ribeiro. 2011. "Flexible heuristics miner (FHM)." *IEEE symposium on computational intelligence and data mining* 310-317.

Weijters, A.J.M.M., W.M.P. van der Aalst, and A.K. Alves de Medeiros. 2006. "Process Mining with the Heuristics Miner Algorithm." *Technische Universiteit Eindhoven, Tech. Rep WP* (166): 1-34.

Wen, Lijie, W.M.P. von der Aalst, Jianmin Wang, and Jiaguang Sun. 2007. "Mining process models with non-free-choice constructs." *Data Mining and Knowledge Discovery* 15 (2): 145-180.

Weske, Mathias. 2012. *Business Process Management.* 2. Berlin Heidelberg: Springer.

Zapier. 2020. Accessed November 12, 2020. https://zapier.com/learn/getting-started-guide/what-is-zapier/.

Appendix

A.1 List of Figures

A.2 List of Tables

A.3 List of Acronyms

BPM	Business Process Management
WFM	Workflow Management
BPL	Business Process Lifecycle
BPML	Business Process Modeling Language
YAWL	Yet Another Workflow Language
AM	Alpha Miner
AM+	Alpha Miner +
AM++	Alpha Miner ++
AM#	Alpha Miner #
HM	Heuristic Miner
IM	Inductive Miner
IMi	Inductive Miner infrequent
IT	Information Technology
CRM	Customer Relationship Management

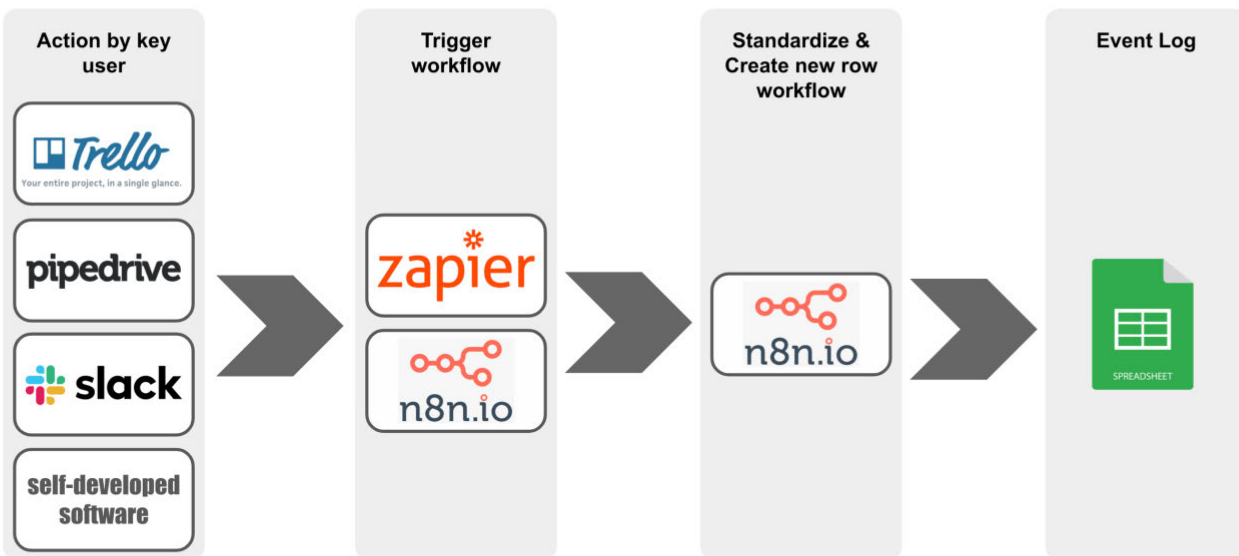
A.4 Building an event Log

To develop an event log that uses multiple data sources, workflows were built with the automation tools Zapier and n8n to record the completion of an activity and, in some cases, the start.

The recording works as follows:

1. A Zapier or n8n workflow is triggered automatically due to an action by the employee (e.g., ticking an activity done)
2. The triggering workflow sends the data to an n8n workflow, which standardizes the data via a Java Script
3. Next, the same n8n workflow creates a new row in a Google Spreadsheet recording the following data points:
    1. Event ID: Consecutive numbering of the events to enable ordering
    2. Case ID: Case identifier to assign the event to a case
    3. Timestamp: The time at which the event occurs
    4. Transaction type: can either be start or complete
    5. Resource: The user/team executing the activity

FIGURE 4 - EVENT RECORDING STREAM

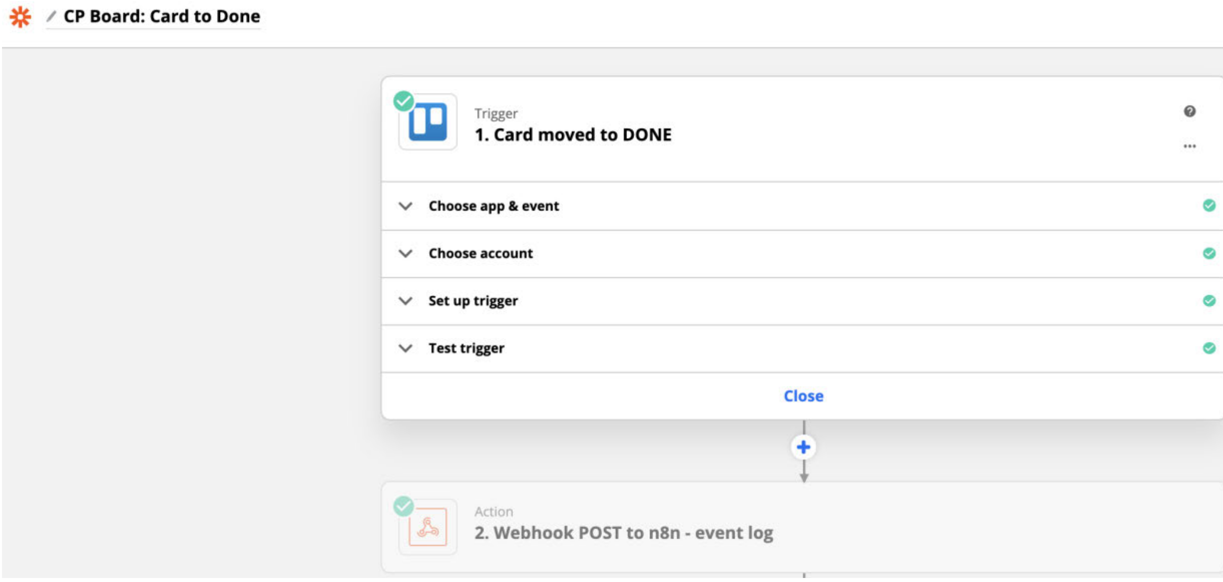


A.4.1 Zapier

Zapier is an online automation tool that can connect various online apps such as mail programs, customer relationship management (CRM) systems (Pipedrive, Salesmate), project tracking tools

(Trello, Jira), or all Google applications (Drive, Gmail, Spreadsheets) (Zapier 2020). One of Zapier's advantages is its user-friendly interface, making it possible to create these interconnections without any programming knowledge.

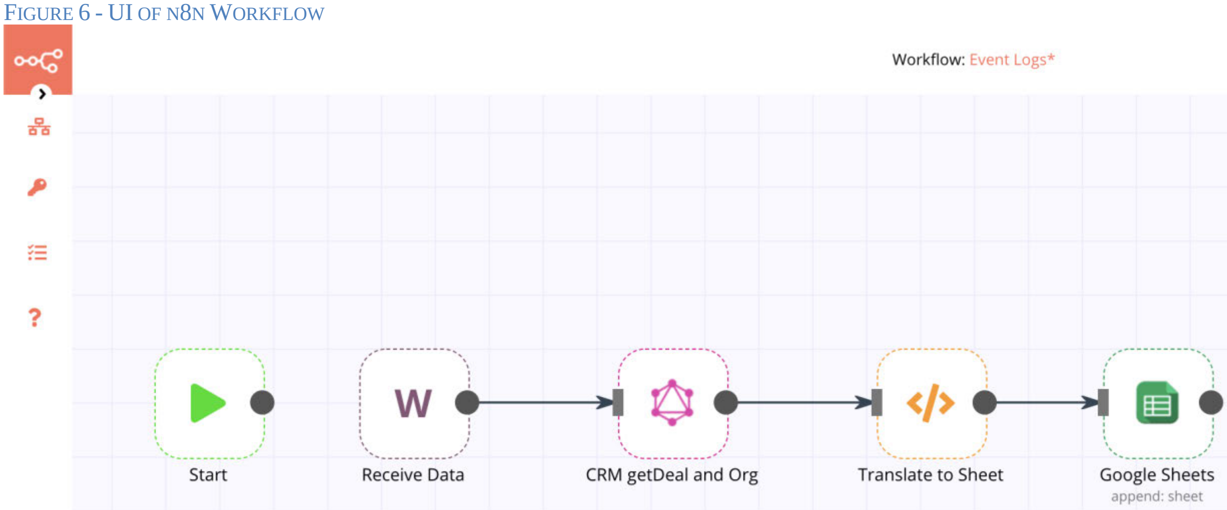
FIGURE 5 - UI OF ZAPIER WORKFLOW



A.4.2 n8n

n8n is also an online automation tool that can connect various online apps (n8n 2020). Compared to Zapier, it is less user-friendly, especially for people with little to no programming knowledge. For users with basic programming skills, it offers much more functionality than Zapier.

FIGURE 6 - UI OF N8N WORKFLOW



A.5 Process Discovery

A.5.1 Petri Net

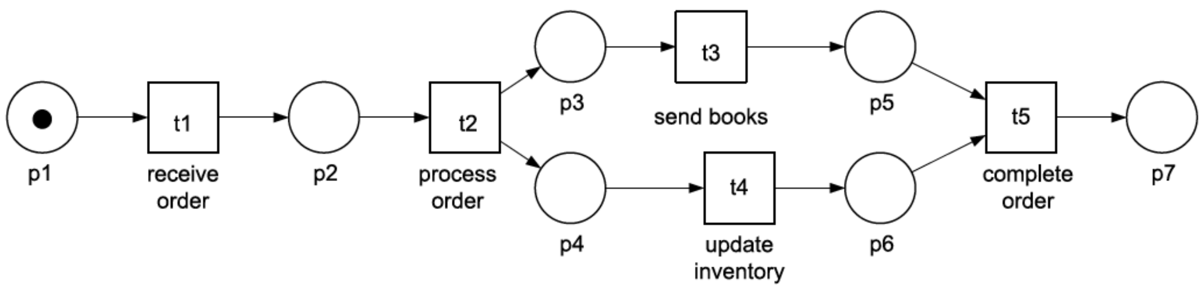
Petri net is a modeling language and according to Weske (2012)"one of the best known techniques for specifying business processes in a formal and abstract way" (149).

A model created with Petri nets can reflect a dynamic process by having one or more tokens running through the model from start to finish.

A Petri net model consists of the following elements (David and Alla 2005):

- Places: Represented by a circle; model starts with a place and ends with a place; can only have one incoming arc and one outcoming arc
- Transitions: Represented by a box; can have more than one incoming and outgoing arc but must have at least one in- and outgoing arc.
- Arcs: Connection between places and transitions

FIGURE 7 - SAMPLE PETRI NET REPRESENTING SINGLE PROCESS INSTANCE



(Weske 2012, 150)

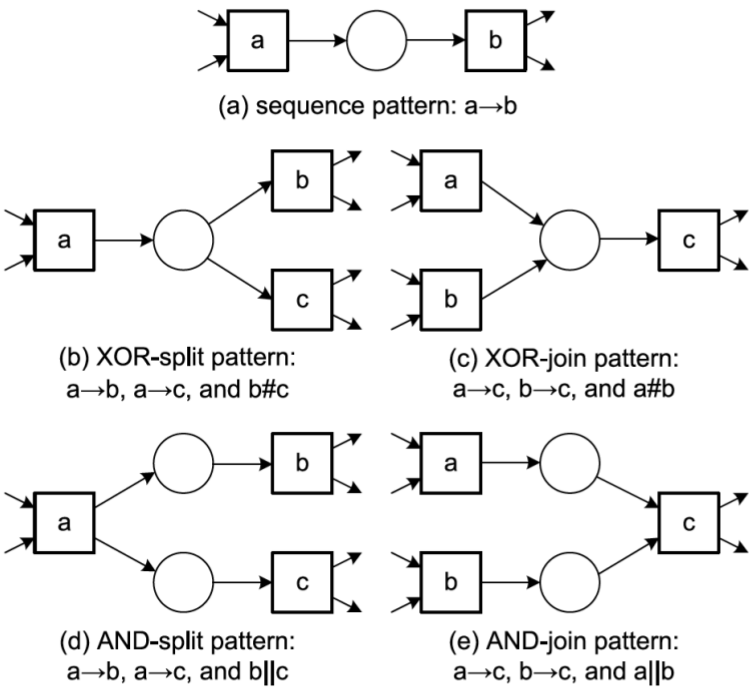
A token is represented as a dot, as shown in the place p1 in FIGURE 7. The transition t1 is fired, if the necessary token exists in p1. If several tokens are needed to fire a transition, as it is the case at t5, one token must exist in p5 and one in p6. Only then t5 is fired. In other words, both activities (transitions) *send books* (t3) and *update inventory* (t4) must be done first (no matter in which order) so that *complete order* (t5) can be fired.

The previous split into parallel workflows happens in t2. This transition needs only one token as input but gives two tokens as output.

A.5.2 Alpha Miner – Process Patterns

The event log is explored by examining the transactions from each case pairwise and grouping them into the following typical process patterns:

FIGURE 8 - TYPICAL PROCESS PATTERNS



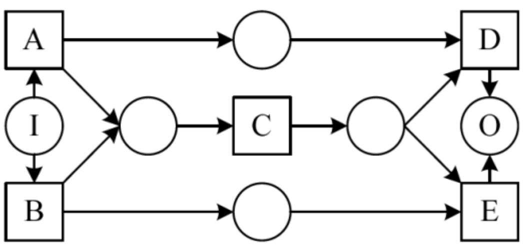
(van der Aalst 2016, 169)

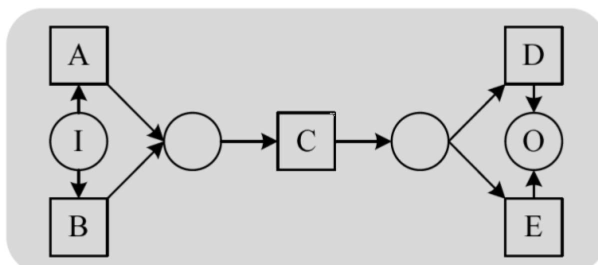
- Sequence pattern: A is always followed by B and never vice versa.
- XOR-split pattern: A is followed by either B or C, but not both. Furthermore, B and C are not directly connected.
- XOR-join pattern: A and B are both followed by C. Furthermore, A and B are not directly connected. This means that either A or B must pass before C can happen.
- AND-split pattern: A is followed by both B and C. Also, A is followed by C and vice versa. This means that A or B can happen in parallel.
- AND-join pattern: A and B are both followed by C. Also, A is followed by B and vice versa. This means that A or B can happen in parallel, and C can happen only after both A and B are completed.

A.5.3 Implicit Dependencies

FIGURE 9 reflects the problem of implicit dependencies caused by non-free choice. The corresponding log contains the traces {ACD} and {BCD}. Considering the gray-shaded model created by the alpha miner, D can be executed as a sequence of B and C. However, this trace is not reflected in the log and, therefore, a "non-free-choice" is present here. D can only be executed if A and C have been passed before, as the model on the left side, created by the alpha ++ miner, represents correctly (Wen, et al. 2007). Thus, the AM++ creates more precise models then the AM+. Moreover, Wen, et al. have proved that the fitness is significantly improved compared to the original alpha miner (2007).

FIGURE 9 - NON-FREE-CHOICE





A.6 Activity Description

Sales Process – Request (by Customer) to Contract Signed (by Customer)

The request to *contract process* is an end-to-end process where the actors are the Sales Department and the Risk Department. The process extends from the loan request of the customer to the creation and verification of an offer to the sending of a contract to the customer.

A loan request is submitted by the Amazon merchant (customer) via the ABC application (app). The customer submits data on the products he wishes to finance. These products must meet certain criteria, such as a sales history of at least two months on Amazon. The products are then automatically evaluated using data obtained from the Amazon listing. This evaluation and the number of products to be financed provide the basis for the interest and the principal of the loan.

Afterward, a member of the ABC risk department checks the loan calculation and approves it. After validation, the offer is sent to the customer and, in most cases, discussed on the telephone. If adjustments to the loan are necessary, the loan may have to be re-evaluated and approved.

Once the customer accepts the offer, a loan contract is created by a ABC employee and uploaded to the app. If it is an initial loan, in the same step a ABC employee downloads customer-related documents from a government website to obtain information about the organizational structure and the persons who must authorize use of an external service. Having gathered this information, the employee sends an e-mail to the customer to educate them on the next steps.

Activity	Resource	Description
Request offer	Customer	Start activity; [redacted]
Register request	Sales	[redacted].
Create offer	Risk/Finance	[redacted]
Upload offer	Sales	[redacted]
Submit application	Customer	[redacted]
Check sales proof	Sales	[redacted]
Collect customer documents	Sales	[redacted]
Validate deal	Risk/Finance	[redacted]
Create contract	Sales	[redacted]

Create contract, download & analyze legal documents	Sales	[redacted]
Review contract	Sales	[redacted]
Upload contract	Sales	[redacted]
Submit signed contract	Customer	[redacted]
Check signed contract	Sales	[redacted]
Deal lost	Sales/Customer	[redacted]

A.7 Detailed Analysis of the HM and the IMi

A closer look at the two models reveals three major differences. First, in the HM, the activity *deal lost* can only be executed after the activity *upload offer*, whereas in the IMi, this can only be executed after the *review contract* & *upload contract* activity. Secondly, the activity *update offer* occurs at two different positions.

The third noticeable aspect of the models are the different traces resulting from the activities *create contract, download & analyze legal documents*, *create contract*, *review contract*, and *upload contract*.

FIGURE 10 - IMI

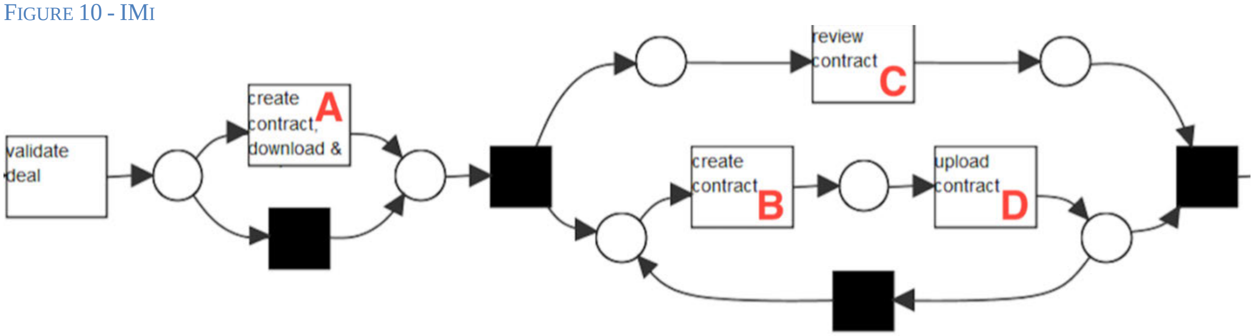
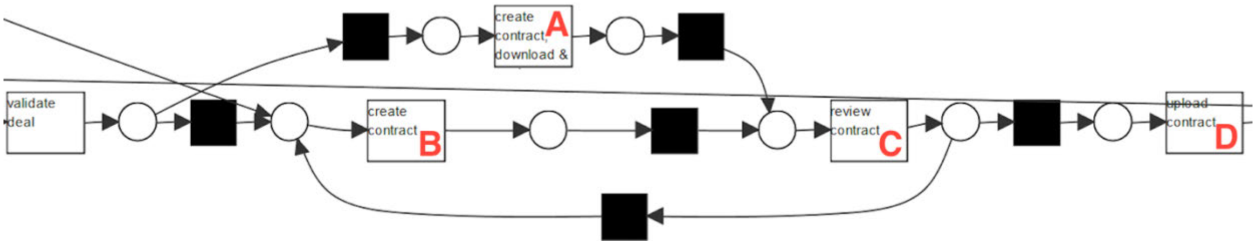


FIGURE 11 - HM



Neglecting the loops, the IMi allows the following traces: {ABCD, ACBD, ABDC, ABDC, BCD, CBD, BDC} and the HM allows the following:{ACD, BCD}.

Furthermore, the loops are different for the miners. In the IMi, {BD} can be executed in repetition. The HM has loops as follows: {BC, (AC)[5]BC}.

By filtering the event log, the following traces of the mentioned activities can be extracted.

- 26 * {BCD}; 3 * {BCDBCD}; 1 * {BCDBCDBCDBCD}; 1 * {BCDBC**DD**BCDBCD}; 1 * {**BB**CD}
- 6 * {ACD}; 1* {ACDACD}
- 4 * {BCBCD}; 1* {BCBCDBCD}
- 1 * {ACACD};

It is remarkable that the two traces of the HM correspond to the two most frequent traces in the event log.

Additionally, the Loop BC can be found in five traces. However, the possible Loop (AC)BC does not exist once in the event log. In summary, 36 of 45 cases can be replayed in the model. Therefore, three of 10 traces from the log are used. The {ACD, BCD} can be found in every trace of the event log.

In the IMi, only one of the seven possible traces can be found in the event log, namely, the most common case, BCD. Furthermore, the loop BD does not exist once in the log.

In summary, 26 of 45 cases can be replayed in the model using one out of 10 traces from the log. The sequence {BCD} is found in seven of 10 traces of the event log; this covers 37 of 45 cases.

[5] In brackets occurs only once

Interviews conducted with employees have indicated that only ACD and BCD are actually allowed. Only these two cases are desired, whereas loops are allowed after activities C and D. However, these are only carried out in case of errors or modification requests by the customer. In other words, a contract is created first, followed by a review, after which either a new contract is created, or it is uploaded to the web application. This entire procedure can also be conducted in loops, whereby a review is always necessary after each contract creation.

Consequently, in this case, the HM delivers a more precise result. The parallel activity *review contract* has a negative influence on the precision of the IMi model. Nevertheless, the HM overall has a lower precision. This is probably due to the fact that the activities *update offer*, *deal lost*, and *collect customer documents* may be performed in parallel with several other tasks. For example, *deal lost* can happen in parallel with 10 other activities. In fact, it happens four times after *upload offer*, one time after *register request*, and one time after *application received*. The HM covers two of these cases. The other eight possibilities never happen, driving the number of possible model traces upward and, therefore, reducing the precision result.

In summary, both the IMi and the HM demonstrate undesirable weaknesses, especially the handling of parallelism and loops. However, the models are well suited to demonstrate the correlation between simplicity and precision.

A prominent advantage of the IMi is its improved simplicity and the fact that the process model is sound. Therefore, the IMi is used for the subsequent performance analysis. Due to the lack of precision in contract activities, a new model is generated using the IMi with the parameter "noise-threshold" increased from 20% (default) to 25% (FIGURE 23). Thus, the contract activities are displayed in a sequence, in other words, as they happen most often and represent the optimal case.

A.8 Statistics

A.8.1 Global Statistics

FIGURE 12 - GLOBAL STATISTICS OFFER CREATION PROCESS

THIS SECTION WAS MARKED CONFIDENTIAL AND IS THEREFORE OMITTED

[redacted]

FIGURE 13 - GLOBAL STATISTICS CONTRACT CREATION PROCESS

THIS SECTION WAS MARKED CONFIDENTIAL AND IS THEREFORE OMITTED

[redacted]

A.8.2 Activity Statistics

TABLE 7 – OFFER PROCESS STATISTICS

Activity	*N*	*Average*	*Median*	*Std. dev*	*min*	*max*	*Quartile 1*	*Quartile 3*
register request	THIS SECTION WAS MARKED CONFIDENTIAL AND IS THEREFORE OMITTED [redacted]							
create offer								
upload offer								

(Time values in hours)

TABLE 8 - CONTRACT PROCESS STATISTICS

Activity	*n*	*Average*	*Median*	*Std. dev*	*min*	*max*	*Quartile 1*	*Quartile 3*
Check sales proof	THIS SECTION WAS MARKED CONFIDENTIAL AND IS THEREFORE OMITTED [redacted]							
Collect customer documents								
Validate deal								
Create contract								
Create Contract, dow. & ana. legal docs.								
Review contract								
Upload contract								

(Time values in hours)

The data in TABLE 7 and TABLE 8 and the time values indicated in the IMi model (FIGURE 2 and FIGURE 3) differ slightly because the data in the IMi model only considers cases that can run through the entire process from start to back. For example, if an activity is executed twice, then this case is no longer considered since this behavior is not reflected in the model. If an activity is skipped or the sequence is different, then these cases are also not taken into account.

In the calculations via Google Spreadsheet, this problem is circumvented. The only prerequisite for the calculation of the sojourn time is the existence of the upstream activity (based on the same IMi model). What happened before or after is not of interest in the spreadsheet calculation. However, to guarantee relative comparability, only complete cases were considered in the spreadsheet calculations.

TABLE 9 - CREATE CONTRACT, DOWNLOAD AND ANALYZE LEGAL DOCUMENTS STATISTICS

	N	*Average*	*Median*	*Std. dev*	*Min*	*Max*	*Quartile 1*	*Quartile 3*
Since 28.09.20	THIS SECTION WAS MARKED CONFIDENTIAL AND IS THEREFORE OMITTED [redacted]							
Before 28.09.20	[redacted]							

(Time values in hours)

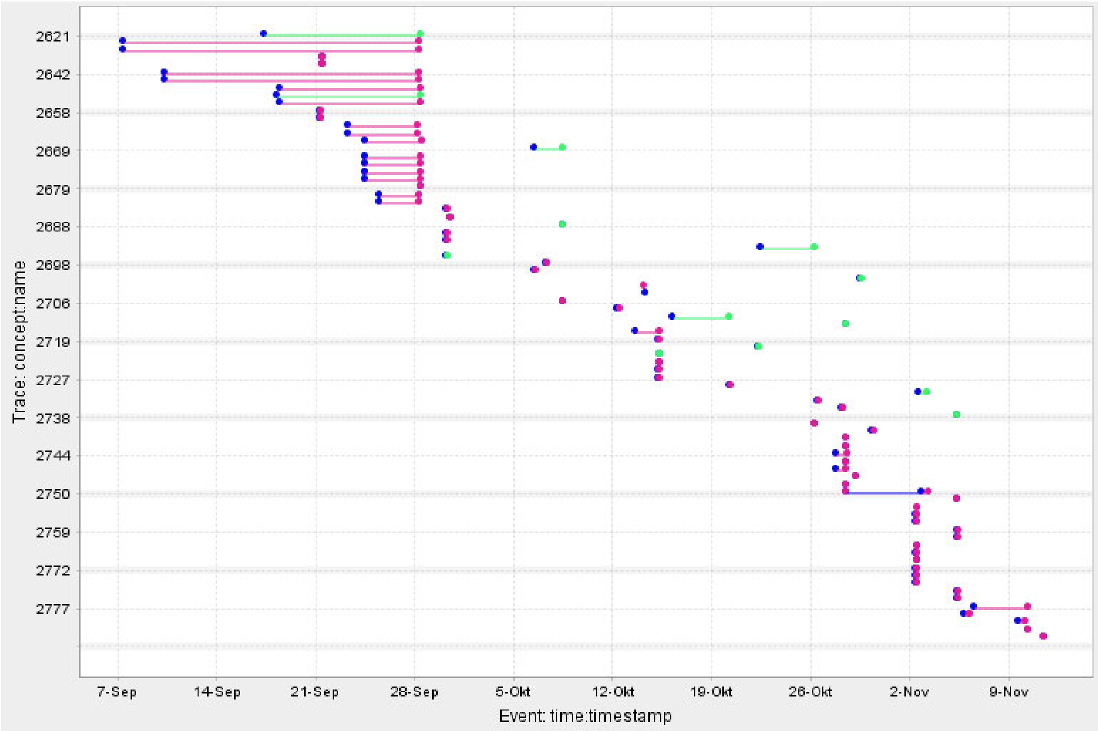
TABLE 10 - CREATE CONTRACT, DOWNLOAD AND ANALYZE LEGAL DOCUMENTS TIMES

	#	*Average*	*Median*
Waiting time	TH S SECT ON WAS MARKED CONF DENT AL AND S THEREFORE OM TTED [redacted]		
Processing time	[redacted]		
Sojourn time	[redacted]		

(Time values in hours)

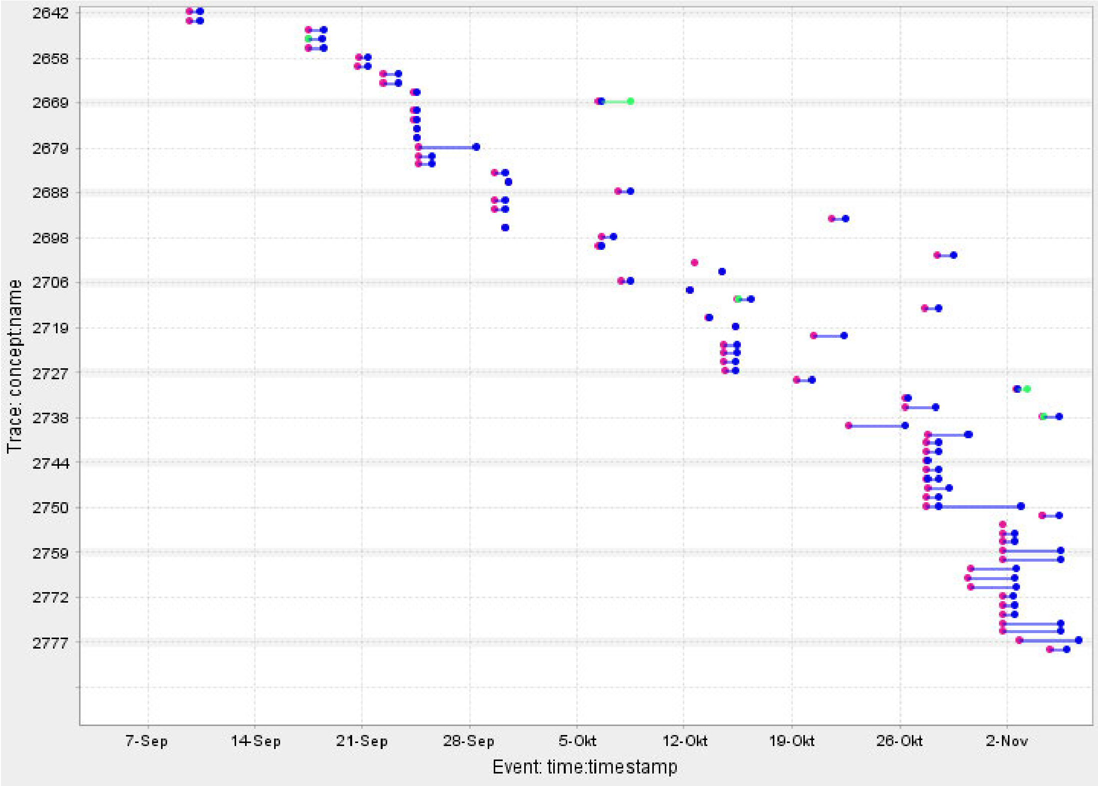
A.8.3 Dotted Charts

FIGURE 14 - CREATE CONTRACT DOTTED CHART



Green = Create contract, download and analyze legal documents; red = Create contract; blue = Validate deal

FIGURE 15 - VALIDATE DEAL DOTTED CHART



Green = collect customer documents; red = check sales proof; blue = validate deal

A.8.4 Impact Predictions

TABLE 11 - DETAILED CALCULATION ON PROCESS TIME IMPROVEMENTS

	now	*Short-term*	*Medium-term*	*Long-term*
register request	THIS SECTION WAS MARKED CONFIDENTIAL AND IS THEREFORE OMITTED [redacted]			
create offer				
upload offer				
offer process				
Check sales proof				
Validate deal				
Create contract				
Review contract				
Upload contract				
contract process				
% change to now				
Sales process (offer + contract)				
% change to now				

(Time values in hours; time values are stating the average sojourn time)

The calculation of the possible changes in process times are based on some assumptions:

- Only the process path via the activities listed above is taken into account. The reason is that this is the most frequently executed path.
- Since no activity of the offer sub-process was examined more closely in the work, no change is noted there.
- The estimation (in bold) of the new process times was made together with key users under previous knowledge of the current times.

A.9 Office Hour Issue

Not taking office hours into account can lead to distortions in execution data of activities, as weekends or nights are ignored. On the other hand, many companies, mostly young digital companies, work according to the best effort principle. Accordingly, tasks are done from home well after office hours. This is also the case at ABC. The question remains, what is the best way to analyze performance? If service time is of interest, then office hours should not be considered, as the customer is not very interested in this. However, if it is about internal performance, it is of great interest if an activity is scheduled just before the end of the week and completed right at the beginning of the next week. But how to deal with the problem if the activity was done during the weekend, because in this certain case an employee worked according to the best effort principle?

To address the office hours issue, it would be necessary to separate activities that are scheduled and completed during office hours from those that are either scheduled or completed outside of office hours. Once this is done, the performance values could be categorized.

A corresponding extension of the performance plug-ins in ProM would be desirable.

Considering the goal of this work - to reduce service time - it is acceptable that office hours are not considered. Nevertheless, it would be of interest to have performance data with office hours taken into account to analyze the sources of bottlenecks.

A.10 Process Models

FIGURE 16 - ALPHA MINER

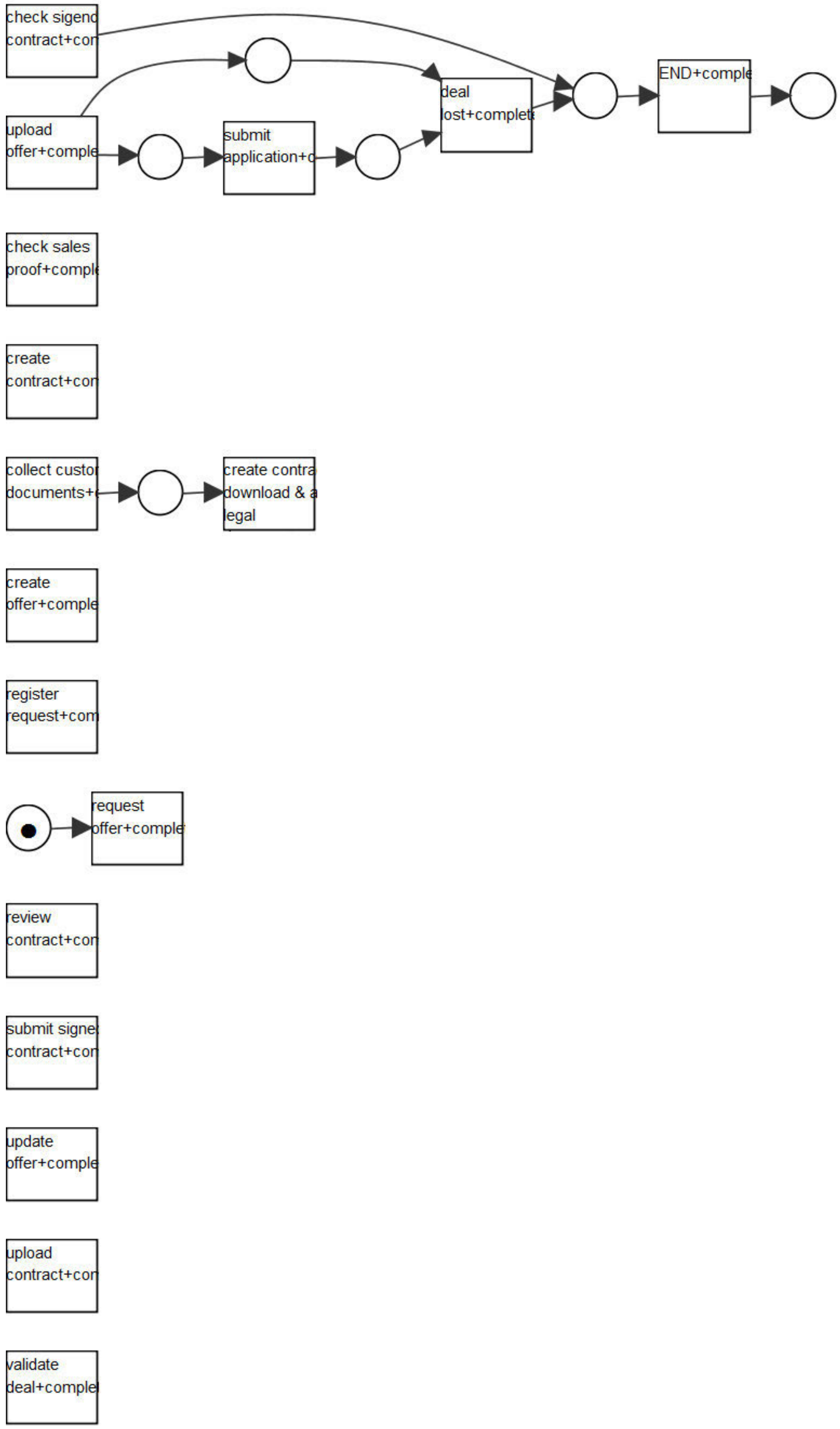


FIGURE 17 - ALPHA MINER +

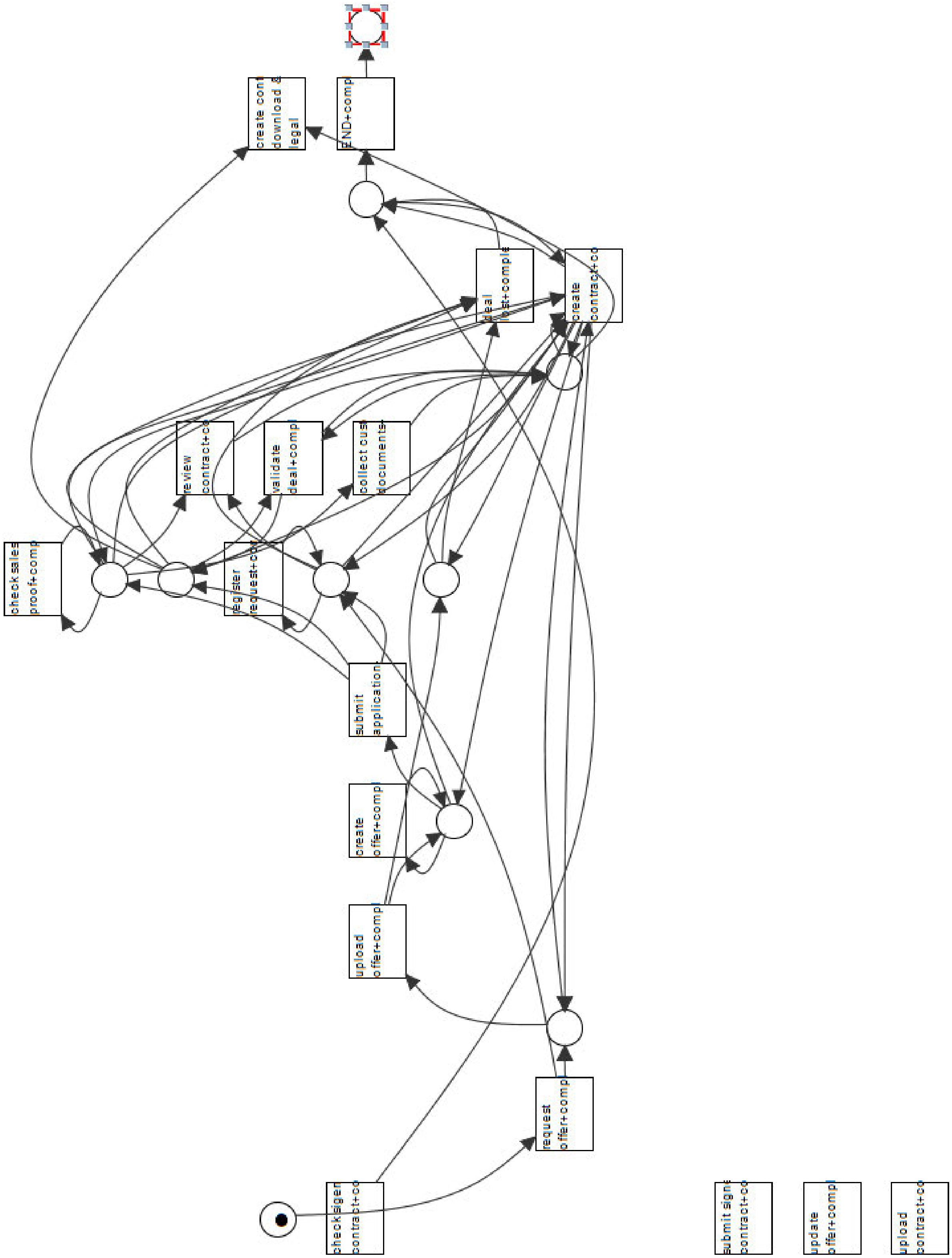


FIGURE 18 - ALPHA MINER ++

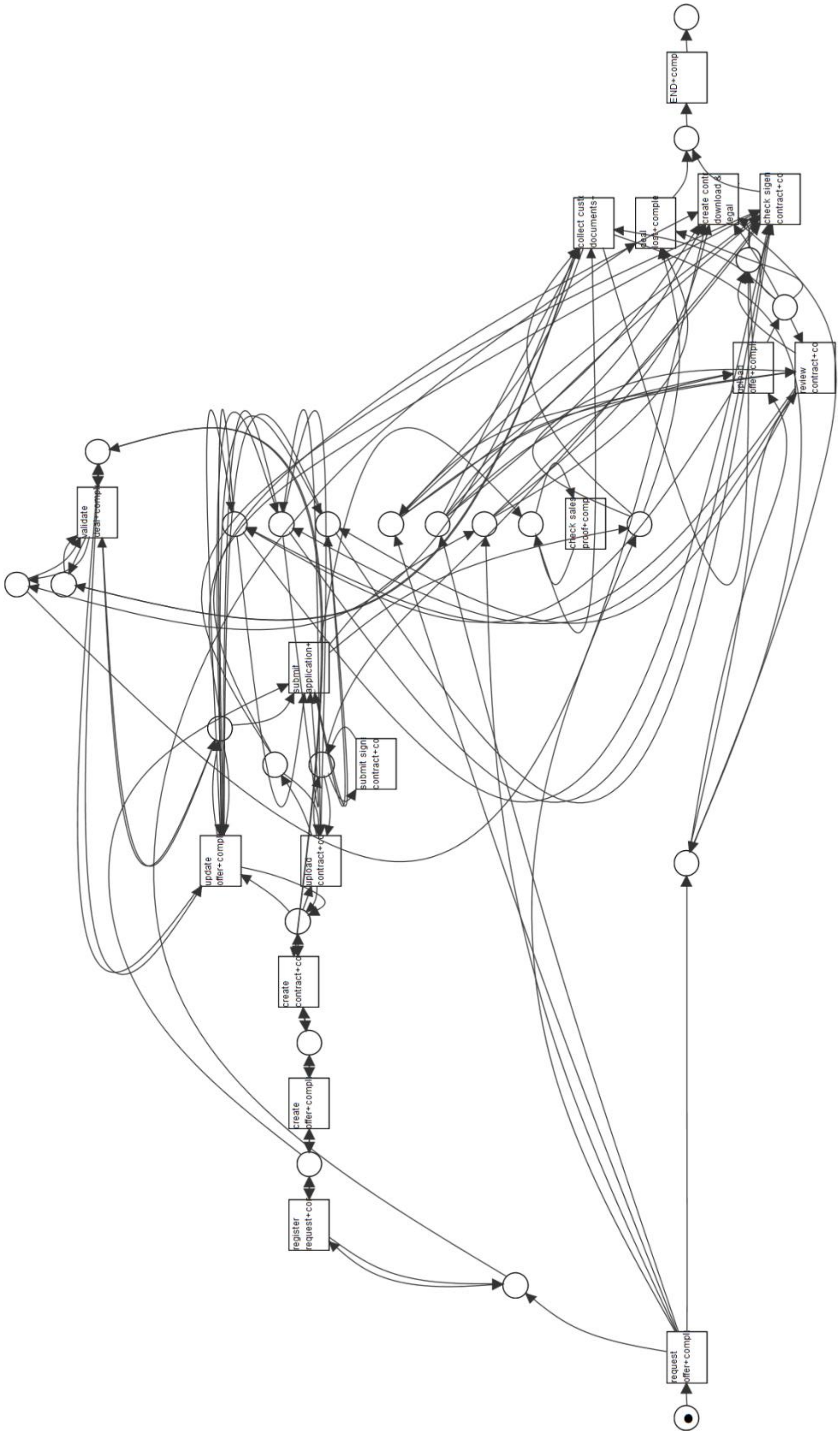


Figure 19 - Alpha Miner #

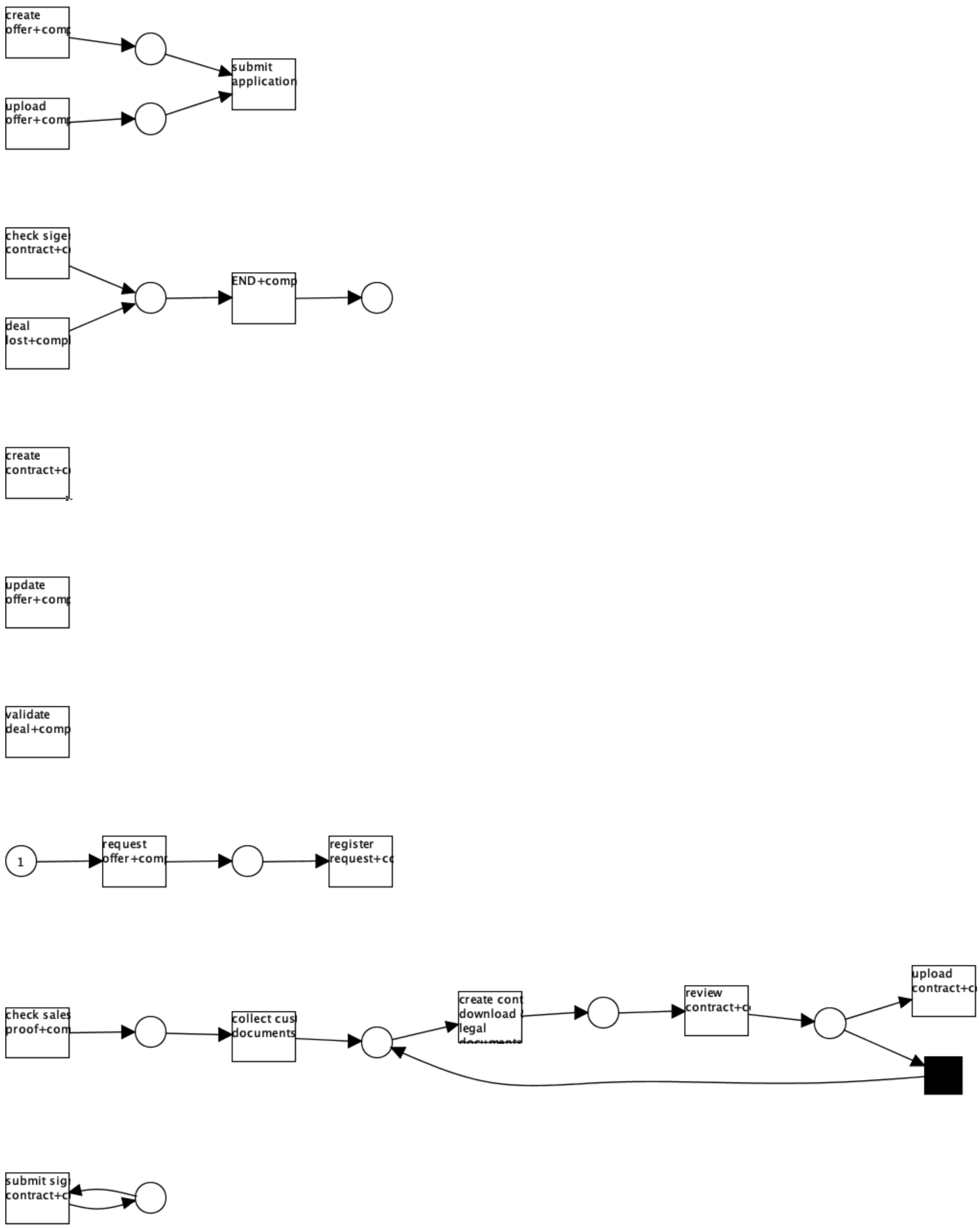


FIGURE 20 - INDUCTIVE MINER

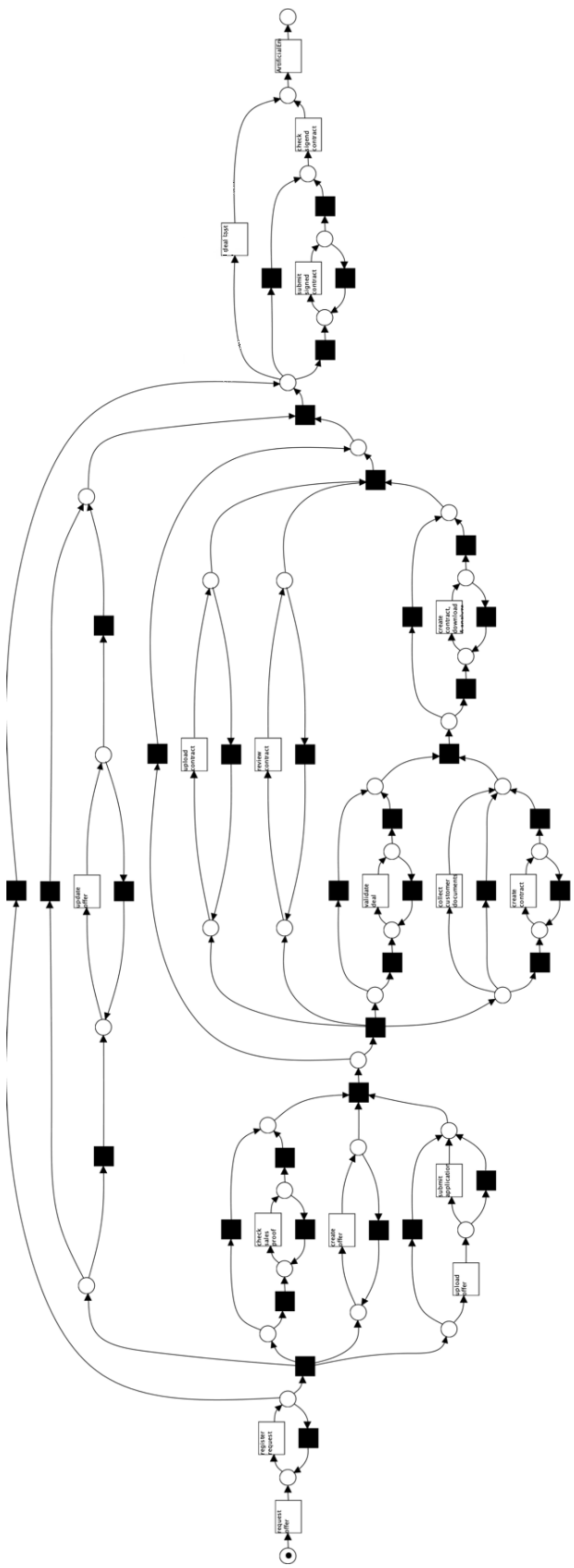


FIGURE 21 - HEURISTIC MINER

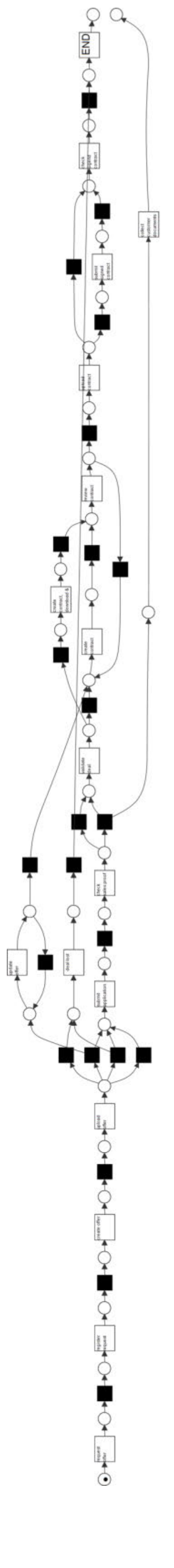


FIGURE 22 - INDUCTIVE MINER INFREQUENT

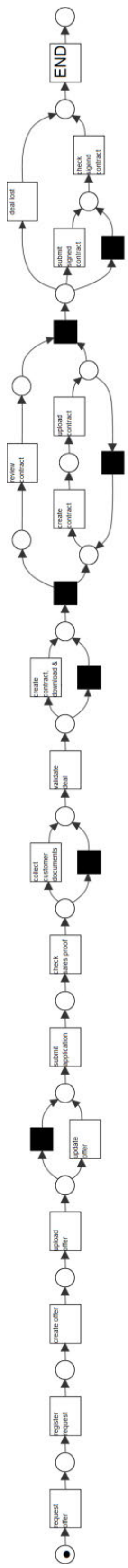


FIGURE 23 - INDUCTIVE MINER INFREQUENT (NOISE-THRESHOLD 25%)

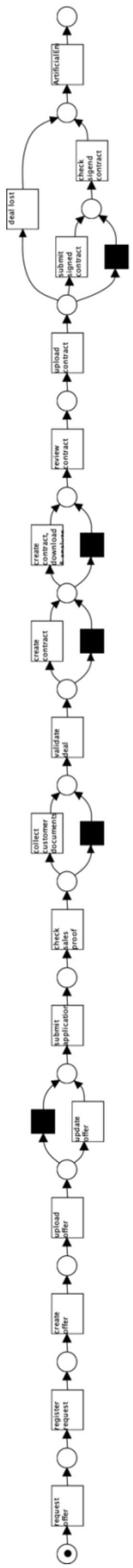
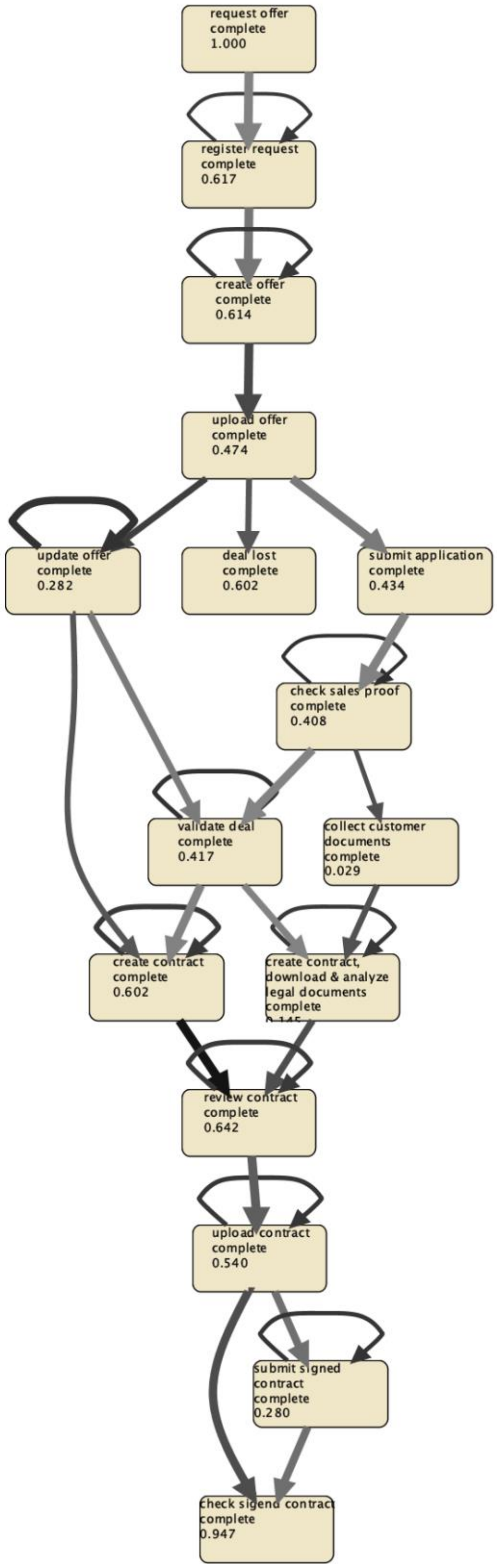


FIGURE 24 - FUZZY MINER



A.11 Interview Questions

The following questions were asked during the semi-structured interviews:

1. What is your area of responsibility in the sales process?
2. Can you please describe each step of your activities?
3. What conditions need to be met in order for your activities to be executed without any problems?
4. Can you please describe the most common sources of errors when performing your activities?
5. How does the handover to the next activity take place?
6. What do you think is the average duration of your activity in case of no problems?
7. What would need to be done to get your task done faster?