UNIVERSITY OF TWENTE.

Faculty of Behavioural, Management and Social Sciences

Creating standardized process mining applications based on the open trip model

Written by

Thom Baas

Supervisors

Jean Paul Sebastian Piest

Rob Bemthuis

EMONS

officedog.ai

Preface

Dear Reader,

Before you lies my bachelor thesis, "Creating standardized process mining applications based on the open trip model," written for my bachelor's degree in Industrial Engineering and Management at the University of Twente. I worked on this thesis together with EMONS Group and Bullit Digital. They provided me insights into data on logistics and the platform Officedog which helped me execute the research.

I want to thank my first supervisor Jean Paul Sebastian Piest, and second supervisor Rob Bemthuis for the pleasant collaboration, feedback, and help during my work on this thesis.

Enjoy reading my bachelor thesis.

Thom Baas

July 2022

Management Summary

This bachelor thesis has been executed in collaboration with Bullit Digital and EMONS Group. Bullit Digital is the developer of Officedog, and EMONS Group is a logistics company that works with Bullit Digital on the development of Officedog.

Motivation

The research has started since EMONS Group and Bullit Digital have sought to develop standardized process mining applications for the platform Officedog based on the open trip model. The open trip model is a semantic standard that ensures that data will be formatted in a unified way. Since Officedog works based on the open trip model, developed process mining applications can be used by multiple clients and are called standardized process mining applications. Standardized process mining applications can be used to make a clear overview of processes and the detection of possible bottlenecks from data provided to Officedog by its partners. Bottlenecks are particularly interesting since they are assumed to cause inefficient logistics for EMONS Group. Therefore, identifying process flow and detecting bottlenecks are potent tools to indicate where processes might be improved inside businesses. Since there are currently no standardized process mining applications available, the research will be focused on creating these for Officedog

Research question

The main research question for this research was, "Which standardized process mining applications can be developed based on the open trip model to support logistics companies by detecting, analyzing, and resolving bottlenecks in their operations?" This question has been chosen as the main research question because information about all three aspects is needed to create suitable standardized process mining applications for Officedog. Therefore, the main research question has been split up into three sub-questions which are focused on either the aspect of detecting, analyzing, or resolving bottlenecks by the creation of standardized process mining applications.

Methodology

There has been chosen to use the Design Science Research Methodology (DSRM) in this research. There has first been done an identification of the problem of this research: bottlenecks assuming to create less efficiency in logistics for partners of Officedog. After this, the objective of a solution was defined as the creation of standardized process mining applications for Officedog. Then there has been started the actual design and development of standardized process mining applications, which were made with Python in a Jupyter Notebook connected to the Officedog database. In the Jupyter Notebook, standardized process mining applications were made and demonstrated on data from the EMONS Group.

Results

The results of this research are some standardized process mining applications that have been developed. The first standardized process mining application created is the mapping and statistical analysis of activities. This standardized process application can be helpful since it visualizes where bottlenecks are occurring based on activities, and a statistical analysis might reveal patterns about frequent days that particular bottlenecks occur. Based on this information, the management board can try to resolve the bottlenecks, e.g.waiting times or traffic jams. This resolve can be done by choosing a different route that is predicted to cause fewer waiting times for the truck drivers. The second standardized process mining application developed is truck drivers' driving and rest time analysis. The European Union has made guidelines for driving & rest times of truck drivers, which can be analyzed to see if these are in line with the European guidelines. This gives a powerful insight into the truck drivers' driving behavior and can be used to create better driving & rest times.

Table of contents

Preface 2
Management Summary 3
List of figures 7
List of tables 8
List of abbreviations 8
Chapter 1 - Introduction 9
1.1 Background information 9
1.2 Problem statement 9
1.2.1 Action problem 10
1.2.2 Core problem 10
1.2.3 Norm and reality 10
1.2.4 Research goal 10
1.3 Problem-solving approach 11
1.3.1 Methodology 11
1.3.2 Research design 12
1.4 Research questions 13
1.4.1 Main research question 13
1.4.2 Sub questions 14
1.5 Structure of the thesis 14
Chapter 2 – Current situation analysis 15
2.1 Open trip model 15
2.1.1 Events 15
2.1.2 Entities 15
2.1.3 Lifecycles 16
2.2 Officedog 17
2.2.1 Overview page 17
2.2.2 Logistics data 17
2.2.3 Event entity 17
2.2.4 Utilities 20
2.2.5 Integration 20
2.2.6 Jupyter Notebook template 21
2.3 Summary and conclusion 21
Chapter 3 – Literature review 22
3.1 Business process models (BPM) 22
3.1.1 Business process model and notation (BPMN) 22

3.1.2 Petri nets 24
3.1.3 Workflow nets 24
3.1.3 Yet Another Workflow Language (YAWL) 24
3.2 Process mining techniques 25
3.2.1 Process discovery 25
3.2.2 Conformance checking 26
3.2.3 Enhancement 29
3.3 Logistics & process mining 30
3.4 Summary and conclusion 31
Chapter 4 – Solution design 32
4.1 Possible process mining applications 32
4.1.1 Mapping of activities 32
4.1.2 Driving and Rest time analysis 32
4.2 The setup of data extraction 32
4.3 PM4PY 33
4.4 Kepler 34
4.5 Seaborn 34
4.6 Summary and conclusion 34
Chapter 5 – Standardized process mining applications 35
5.1 Connection to Officedog 35
5.1.1 Dependencies & database connection 35
5.1.2 Define and instantiate database models 36
5.2 Creation of a general object 37
5.3 Process discovery 39
5.3.1 Process discovery for one vehicle id 39
5.3.2 Process discovery for multiple vehicle id's 40
5.4 Mapping of all events 41
5.5 Mapping and analysis of waiting times 41
5.5.1 Creating a data frame for waiting times 41
5.5.2 Mapping of waiting times 42
5.5.3 Statistical Analysis of Waiting Times 43
5.6 Mapping and analysis of traffic jams 45
5.6.1 Creating a data frame for traffic jams 45
5.6.2 Mapping of traffic jams 45
5.6.3 Statistical analysis of traffic jams 46
5.7 Driving and rest time analysis 47

5.7.1 Daily driving time analysis 47
5.7.2 Weekly driving time analysis 47
5.8 Summary and conclusion 47
Chapter 6 – Validation of the research 48
6.1 EMONS Group 48
6.2 Bullit Digital 48
Chapter 7 – Conclusion and recommendations 49
7.1 Conclusion 49
7.2 Recommendations 49
References 50
Appendix A 52
Appendix B 54
Appendix C 55
Appendix D 56
Appendix E 57
Appendix F 58
Appendix G 59
Appendix H 60
Appendix I 62
Appendix J 67
Appendix K 68

List of figures

Figure 1: Problem cluster (draw.io, 2022) 9
Figure 2: Visualization of the DSRM (Peffers et al., 2007) 11
Figure 3: OTM data model (Open Trip Model, 2020) 15
Figure 4: Overview page of Officedog (Bullit Digital, 2022) 17
Figure 5: Logistics data in Officedog (Bullit Digital, 2022) 17
Figure 6: Event data in Officedog (Bullit Digital, 2022) 18
Figure 7: Singe event in Officedog (Bullit Digital, 2022) 19
Figure 8: Bulk upload feature in Officedog (Bullit Digital, 2022) 20
Figure 9: Excel template for bulk upload (Bullit Digital, 2022) 20
Figure 10: Integration section (Bullit Digital, 2022) 20
Figure 11: The four main elements of BPMN (Weske, 2020) 22
Figure 12: Types of gateways (Weske, 2020) 23
Figure 13: BPMN diagram (Weske, 2020) 23
Figure 14: Petri net of a process (Weske, 2020) 24
Figure 15: Example of a model created by YAWL(Weske, 2020) 24
Figure 16: Petri nets created based on the α-algorithm (van der Aalst, 2012) 25
Figure 17: A process visualized by N_1 and N_2 (van der Aalst, 2011) 27
Figure 18: Footprints of N_1 (Left) and N_2 (Right) (van der Aalst, 2011) 27
Figure 19: Differences between footprints of N1 and N2 (van der Aalst, 2011) 28
Figure 20: Formula for calculating fitness for a single trace (van der Aalst, 2011) 28
Figure 21: Formula for calculating fitness of a model (van der Aalst, 2011) 28
Figure 22: Total fitness of models N_1 and N_2 (van der Aalst, 2011) 29
Figure 23: Process Mining Studies in the six most common fields (dos Santos Garcia et al., 2019) 30
Figure 24: Data flow from Officedog to Jupyter Notebook 32
Figure 25: Data coming from a board computer of a vehicle by EMONS Group 33
Figure 26: Example for process discovery (Fraunhofer FIT, 2021) 33
Figure 27: Creation of a BPMN model with PM4PY 34
Figure 28: Creation of a Petri net with PM4PY 34
Figure 29: Code used to install and load dependencies 35
Figure 30: Code used for the database connection 35
Figure 31: Engine & class setup in Jupyter Notebook 36
Figure 32: The metadata visualized in Officedog for an event (Officedog, 2022) 36
Figure 33: Functions defined for the selection of location and day name 37
Figure 34: Object containing several functions to enable object-orientated programming 38
Figure 35: Code used to generate a Petri net for one vehicle id 39
Figure 36: Obtained Petri net for one vehicle id 39
Figure 37: Code used to generate a Petri net by using multiple vehicle id's 40
Figure 38: Discovered Petri net for multiple vehicle id's 40
Figure 39: Mapping of all events 41
Figure 40: Creation of data frame for waiting times 42
Figure 41: The creation of a map indicating the waiting times 42
Figure 42: Statistical plot of waiting activities durations 43
Figure 43: Statistical output for waiting activities from the duration stats function 44
Figure 44: Creation of data frame for traffic jam activities 45
Figure 45: Map of occurred traffic jams 45
Figure 46: Statistical plot of traffic jam duration 46

Figure 47: Statistical output for traffic jam from duration stats function 46
Figure 48: Code and result for daily driving hours 47
Figure 49: Code and results of weekly driving time analysis 47
Figure 50: Jupyter Notebook template provided by Bullit Digital 53
Figure 51: Example of a Workflow net containing multiple parties (Weske, 2020) 54
Figure 52: Notational elements of YAWL (Weske, 2020) 55
Figure 53: Event log containing traces (Van der Aalst, 2011) 56
Figure 54: Replay of N_1 (Van der Aalst, 2011) 57
Figure 55: Replay of N_2 (Van der Aalst, 2011) 58
Figure 56: Driving time and rest periods (European Commission, 2006) 59
Figure 57: Example notebooks created in the PM4PY library that can be used for process mining 61

List of tables

Table 1: Description of the OTM entities (Open Trip Model, 2022) 16
Table 2: Description of the OTM lifecycles (Open Trip Model, 2022) 16
Table 3: Variables from the formula for total fitness of a model (van der Aalst, 2011) 28
Table 4: Process enhancement studies that have been published (Yasmin et al., 2018) 29
Table 5: The topics from recent studies about process mining (dos Santos Garcia et al., 2019) 30
Table 6: Feedback form of EMONS Group 67
Table 7: Feedback form of Bullit Digital 68

List of abbreviations

OTM	Open Trip Model
DSRM	Design Science Research Methodology
PM4PY	Process Mining for Python
BPM	Business Process Models
BPMN	Business Process Modeling
SLR	Systematic Literature Review
YAWL	Yet Another Workflow Language

Chapter 1 - Introduction

This section discusses the context in which the research has been executed. **Section 1.1** contains information about Officedog and EMONS Group. **Section 1.2** explains the core and action problem by a problem cluster, the gap between norm and reality, and the research goal. **Section 1.3** is about the research methodology, which in this research was the DSRM. **Section 1.4** explains how the research plan has been designed. Finally, **Section 1.5** discusses the further structure of this thesis.

1.1 Background information

In today's world, data analysis has become an essential part of running a business. Companies have become part of Industry 4.0, where digital connectedness exists between all parts of the enterprise, and decision-making is built around intelligently analyzing data to seek optimization (Ghobakhloo, 2020). Industry 4.0 can be described as the fourth industrial revolution and the digital transformation of the business world (Ghobakhloo, 2020). Since companies' data are being stored due to the digitalization propelled by Industry 4.0 initiatives, novel approaches to solve problems that can be found by analyzing existing data are needed.

The platform Officedog is currently being constructed by Bullit Digital and is designed to play an essential role for businesses in logistics (Bullit Digital, 2022). The Officedog platform processes data using the Open Trip Model (OTM). The OTM is a semantic standard that ensures that data delivered by several companies will be formatted in a unified way (OTM, 2022). This unified way of formatting ensures that businesses can adequately communicate with each other because there is a form of agreed communication standard in the data. Therefore, businesses can track and see the logistic movements of their goods in real-time with correct information attached to them due to coherence to the OTM standard. **Chapter 2** gives a more detailed explanation of the OTM and its use within Officedog.

One of the companies involved in the development of Officedog concerns EMONS Group. EMONS Group is a logistics company that is active in multiple countries in the European Union and the UK (EMONS Group, 2022). They provide the shipments for goods of other businesses, and they have provided data to support the development of Officedog by this research.

1.2 Problem statement

This section is about the problem statement. **Subsection 1.2.1** explains the action problem. **Subsection 1.2.2** discusses the core problem. **Subsection 1.2.3** shows the difference between norm and reality. **Subsection 1.2.4** describes the research goal that has been formulated for this research.

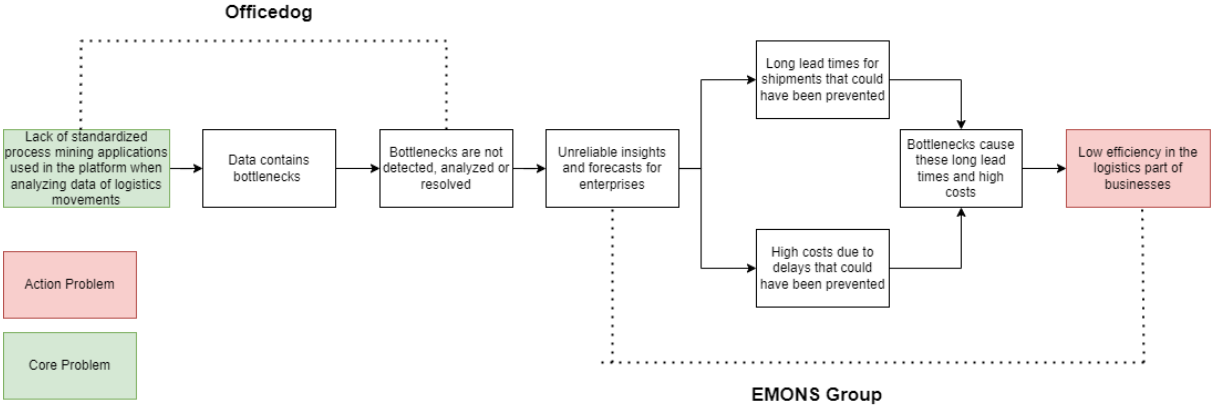

Figure 1: Problem cluster (draw.io, 2022)

Figure 1 shows the problem cluster created with the modeling tool draw.io (draw.io, 2022). There can be seen that the action problem is being put on the right part in red. The action problem is the problem that EMONS Group and Officedog are facing. There can be seen that a couple of causal relations are assumed, resulting in the action problem. Furthermore, there can also be seen the core problem of

this research on the left side in green and visual indicators to see which party owns which part of the problem cluster.

1.2.1 Action problem

Businesses that are part of the platform Officedog are facing efficiency problems in their logistics system. This inefficiency could be caused by bottlenecks which might be prevented if detected. Moreover, delays caused by bottlenecks result in longer lead times for shipments than would have been necessary and, eventually, higher costs. Therefore, the focus should be on identifying and mitigating bottlenecks to solve the action problem. EMONS Group is one of the companies working with Bullit Digital to develop the platform Officedog and is part of this research. They are facing this problem of efficiency problems in their logistics and want them to be solved by analyzing the core problem.

1.2.2 Core problem

The core problem which causes the action problem is that no standardized process mining applications are used in the Officedog when analyzing data of the shipments. Standardized process mining applications mean process mining techniques that can be applied to similar cases. In Officedog, standardized process mining applications can be developed based on the OTM since the data uploaded to Officedog is formatted uniformly. Therefore, data coming from different companies will be saved uniformly formatted in Officedog and can be analyzed similarly by standardized process mining applications based on the OTM. The low efficiency in logistics of the logistics partners of Officedog is assumed to be caused by the bottlenecks that could have been detected and resolved by the use of standardized process mining applications in Officedog. Standardized process mining applications created in Officedog might see that a particular bottleneck is causing problems for logistics efficiency. Once these bottlenecks are detected, a company could decide to resolve them by (manual) intervention. This prevents the action problem from occurring since the bottlenecks are tackled before causing a decrease in efficiency.

1.2.3 Norm and reality

As seen in the core problem of this research, standardized process mining applications for Officedog should be developed to improve the logistics efficiency of businesses working with the platform. Therefore, there is a gap between norm and reality. The gap between norm and reality would be that the norm is to detect bottlenecks by standardized process mining applications to create better insight into a company's operations. The reality is that Officedog currently does not use analysis on bottlenecks by standardized process mining applications to provide better insights to businesses that are using the platform. Therefore, these businesses are still facing inefficiency problems in logistics which may have been prevented if Officedog would use standardized process mining applications.

1.2.4 Research goal

Now that the action problem, core problem, and the difference between norm and reality have been explained, the research goal can be declared as "creating insights into bottlenecks to increase logistics efficiency by creating standardized process mining applications for Officedog based on the OTM." Concerning the problem statement, this research aims to deliver standardized process applications that can be used in Officedog to support EMONS Group and future partners to gain insight into found bottlenecks. It will help to create more efficient logistics by using process mining.

1.3 Problem-solving approach

This section is about the problem-solving approach. **Subsection 1.3.1** discusses the chosen research methodology, which is the DSRM. **Subsection 1.3.2** explains the research design of this research.

1.3.1 Methodology

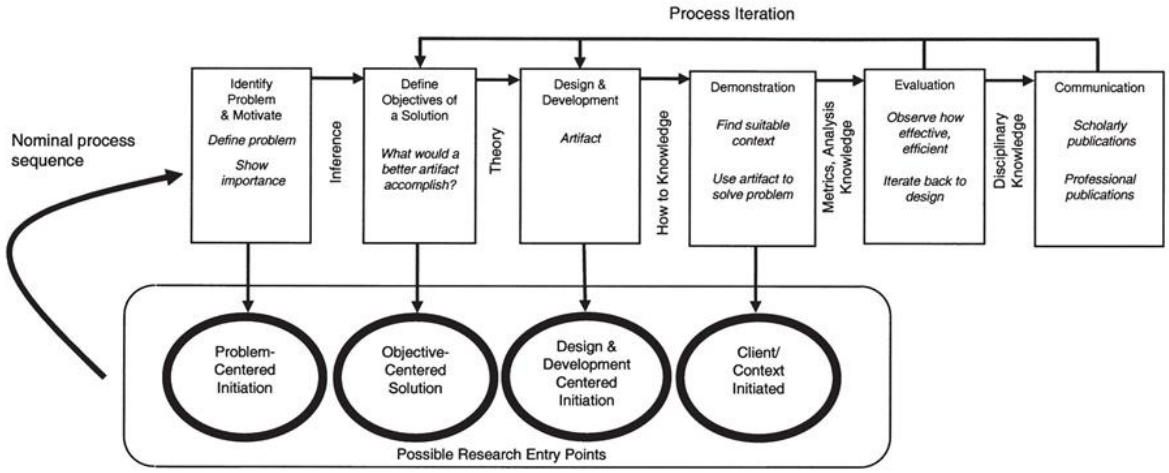

Figure 2: Visualization of the DSRM (Peffers et al., 2007)

This research uses the Design Science Research Methodology (DSRM) visualized in **Figure 2**. The DSRM contains several steps to create an artifact that can be used to solve a specific problem (Peffers et al., 2007). There has been chosen to use the DSRM since the development of standardized process mining applications for Officedog requires a structured methodology to be appropriately created. The DSRM gives this structure and ensures that the standardized process mining applications, the artifact, will be adequately created & evaluated to implement it into Officedog successfully. The stages of the DSRM and their relation to this research will now be discussed briefly.

Identify problem & motivate

The first step of the DSRM is to identify the problem and motivate why it is a problem. In the case of this research, the problem is that bottlenecks are assumed to create inefficiency in logistics for partners of Officedog. These bottlenecks are not desired for logistical partners of Officedog, and these should therefore be prevented or resolved.

Define the objective of a solution

The objective of the research is to propose a solution that increases the efficiency of logistics systems of partners from Officedog. To find out how this can be done there has been conducted a current situation analysis in **Chapter 2** and a literature review in **Chapter 3**. Furthermore, data of both Officedog and EMONS Group has been studied in **Chapter 4**.

Design and development

The design pieces of this research are the standardized process mining applications for Officedog based on the OTM. The standardized process mining applications should make it possible to detect, analyze, and resolve the bottlenecks for logistical partners using Officedog. **Chapter 4** gives an overview of what standardized process mining applications have been chosen to be created during this research

Demonstration

After there have been designed and developed standardized process mining applications based on the OTM, these have been validated using data sets provided by the EMONS Group. This resulted in a demonstration of how the developed standardized applications work in a real-world situation. **Chapter 5** show this demonstration and its results.

Evaluation

Here the results of the demonstrated standardized process mining applications have been analyzed. The selected standardized process mining applications are discussed to see if they have adequately detected, analyzed, and resolved bottlenecks. There has been looked if partners of Officedog can use the standardized process mining applications by validating this with EMONS Group and Bullit Digital in **Chapter 6**. Furthermore, the results, possible improvements, and topics for further research are discussed in **Chapter** 7.

Communication

In this stage, the proposed standardized process mining applications can be used by logistical partners of Officedog to increase logistical efficiency by detecting, analyzing, and resolving bottlenecks.

1.3.2 Research design

Type of research

Several types of research can be conducted, but this research is focused on descriptive and explorative research. Descriptive research means that there will be researched clearly defined hypotheses or investigative questions (Cooper et al., 2018). The main research question and several sub-questions for this research context have been clearly defined in **Section 1.4**. These sub-questions are straightforward investigative questions, which is why this research focuses on descriptive research. Explorative research means a question is being explained or described to understand the overall problem faced (Cooper et al., 2018). This research also contains questions that search for process mining information to better understand its relation to bottlenecks. Therefore, this research is also explorative.

The research subjects

The research contains a couple of research subjects. The first research subject is detecting, analyzing, and resolving bottlenecks. It is the leading research subject since this topic is used to develop standardized process mining applications based on the OTM for Officedog. This research subject also states the other research subjects, the OTM and process mining in logistics. Again, these research subjects are investigated to develop standardized process mining applications for Officedog.

Operationalization of key variables

As mentioned, the key variable will be time for this research. During this research, standardized process mining applications will be developed based on the OTM with the overall goal of increasing the efficiency of logistics companies. Furthermore, the developed standardized process mining applications will investigate timestamps of activities during shipments to search when and where bottlenecks occur, e.g., driving, waiting, and traffic jam activities.

Data gathering method

This research uses quantitative research as the data gathering method. There has been a search for bottlenecks in data from Officedog and EMONS Group, which required analyzing many shipments that have taken place. For this, the large data sets from Officedog and EMONS Group were used, containing the shipment information and thus potential bottlenecks. The assignment is based on the OTM with synthetic data that reflects operational transport scenarios; therefore, the research will be quantitative.

Data analysis method

Process mining will be used as the data analysis method since Officedog needed standardized process mining applications based on the OTM. Process mining will be used to detect, analyze and resolve bottlenecks in datasets to create standardized process mining applications based on the OTM. This will be done on data from Officedog and EMONS Group.

Scope

The research scope means the amount of breadth and depth a topic will be discussed (Cooper et al., 2018). This research looks at approaches to creating standardized process mining applications for Officedog based on data from Officedog and EMONS Group. There is searched for ways to detect, analyze and resolve bottlenecks based on this analysis. However, more data could be analyzed than

the data studied in this research. For example, one could think of data from other logistical partners of Officedog or even more data sets over a more significant period from EMONS Group. However, since there is only limited time available for this research, there will be focused on doing in-depth research on the provided data by Officedog and EMONS Group. Therefore, in the future, a researcher could be going into more breadth of this topic by analyzing data of other companies that might become part of Officedog or more data from EMONS Group to get an even better picture of possible standard process mining applications that could be created to prevent bottlenecks.

Limitations

The proposed research design has a few limitations. The first limitation of this research is that there cannot be checked whether the proposed standardized process applications will work in real-time. This is a limitation because Officedog is still under development and is not live yet. If Officedog is used in real-time in the future, there can be seen whether the proposed standardized applications will work. However, since the standardized process mining applications are analyzed whether they work, this does not mean they are not suitable. The second limitation is that the research only focuses on data from Officedog and the EMONS Group. Data from other companies might result in different results than the data from the EMONS Group. However, the data analysis on data from Officedog and EMONS Group will result in standardized process mining applications that can be implemented in Officedog. If future research is done, even better-standardized process mining applications can be made for Officedog based on data from several companies.

Intended deliverables

In this research, several deliverables will be reported. The main deliverable will be standardized process mining applications based on the OTM for bottlenecks of the logistics companies for Officedog. These are based on data analysis of data from Officedog and EMONS Group. The analysis will be done via a Jupyter Notebook containing several libraries related to process mining. Furthermore, a theoretical framework and methodology will be made to give the research more weight and body. Also, a detailed report will be handed in with a description of the research, conclusions, and recommendations.

Literature review methodology

The systematic literature review methodology (SLR) is used to find accurate information surrounding the research topic. The SLR methodology is helpful since it allows the user to compute search strings in a way that databases with a lower bias than traditional literature search in a search engine (Kitchenham et al., 2009). Furthermore, the SLR is helpful in the literature review section since it provides relevant scientific articles about process mining.

1.4 Research questions

This section is about the research question. **Subsection 1.4.1** discusses the main research question formulated to solve the core problem. **Subsection 1.4.2** shows the sub-questions that have been made to divide the main research question into several parts.

1.4.1 Main research question

The main research question of this research is "Which standardized process mining applications can be developed based on the OTM to support logistics companies by detecting, analyzing, and resolving bottlenecks in their operations?" This question must be solved since it connects the core problem with the action problem, as seen in **Figure 1**. The question is searching for a way to improve the efficiency of logistics (action problem) by executing standardized process mining applications (core problem). Therefore, it will also help to execute the "Communication" stage in the DSRM. Since the answer to the main research question will result in standardized process mining applications for Officedog that increase efficiency in logistics for logistical partners.

1.4.2 Sub questions

In order to solve the main research question, there have been created several sub-questions

Sub question 1

The first sub-question is "Which process mining techniques can be used to detect bottlenecks?". This sub-question needs to be answered to get a proper insight into detecting bottlenecks in data sets by process mining. The scientific literature on this topic will be studied to see how process mining is used on data sets to detect bottlenecks. Furthermore, the question fits in the "Design and development" stage of the DSRM since it helps design standardized process mining applications to detect bottlenecks.

Sub question 2

The second sub-question is "Which process mining techniques can be used to analyze the bottlenecks?" Once the bottlenecks have been detected, they should be analyzed using process mining techniques to get more information on why they are occurring. To get more information on how this can be done correctly, scientific articles based on this topic need to be studied which use some process mining techniques. This question is also necessary to properly execute the "Design and development" stage in the DSRM since it will show which process mining techniques can be used to analyze the bottlenecks and develop potential solutions.

Sub question 3

The third sub-question created to solve the main research question is "How to resolve bottlenecks by standardized process mining techniques?". This sub-question helps to find ways to prevent bottlenecks from occurring in the future. It could be said that it also supports the "Design and development" stage of the DSRM since it indicates ways of using process mining to resolve bottlenecks. So based on scientific literature, there could be seen how process mining applications help to prevent bottlenecks from occurring in the future.

1.5 Structure of the thesis

The remainder of this thesis is structured in the following way. **Chapter 2** describes the current situation analysis. **Chapter 3** takes a look at the literature surrounding the research questions. **Chapter 4** indicates which and how standardized process mining applications will be designed. **Chapter 5** presents the developed and demonstrated standardized process mining applications containing results. **Chapter 6** contains a validation of this research from EMONS Group and Bullit Digital. Finally, **Chapter 7** concludes the research and recommends possible topics for future research.

Chapter 2 – Current situation analysis

There will now be looked at the Open Trip Model (OTM) and the platform Officedog and how this platform currently handles the data process. **Section 2.1** discusses the concepts of the OTM and its use cases. **Section 2.2** is about the layout and essence of the development of Officedog. Finally, **Section 2.3** summarizes and concludes this chapter.

2.1 Open trip model

This section is about the OTM. **Subsection 2.1.1** explains the use of events within the OTM. **Subsection 2.1.2** describes the use of entities within the OTM. **Subsection 2.1.3** discusses the lifecycles that are used within the OTM.

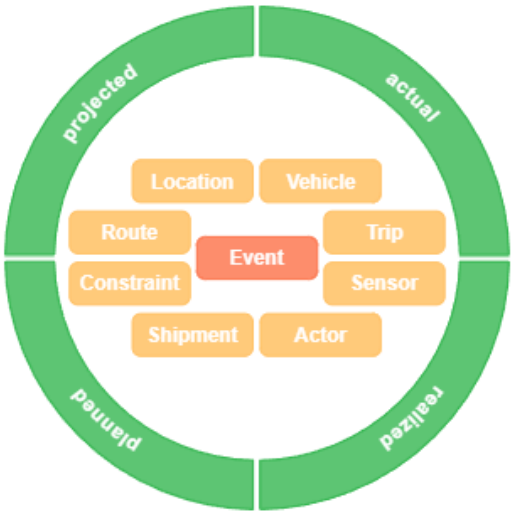

Figure 3: OTM data model (Open Trip Model, 2020)

The OTM is an open-source and easy-to-use data model which makes it possible to exchange real-time logistic trip data on the web and provides a standardized digital vocabulary to describe and exchange the information before, during, and after transport operations within a logistics supply chain (Open Trip Model, 2022). Therefore, the OTM enables multiple actors in a supply chain to see the actions happening in real-time, which is a big step forward for supply chains. The OTM is based on three concepts: lifecycles, entities, and events (Open Trip Model, 2022). These concepts have been visualized in **Figure 3** and will be discussed now.

2.1.1 Events

Events are central in the OTM since they model relationships between entities as events that are projected, actual, planned, or realized (Open Trip Model, 2022). An example of this is the relationship between the entities shipment and vehicle. Once a shipment is unloaded from a vehicle, the transportation has ended, which ends the relationship (Open Trip Model, 2022)

2.1.2 Entities

Entities are used in the OTM to divide metadata occurring in events in concrete sectors and are visualized in orange in **Figure 3**. A couple of different entities are used in the OTM, which are explained in **Table 1**.

Location	A Location entity models any sort of location. It can refer to an address (administrative reference) and a geographical point or area (geographic reference). This allows users to have an address reference that may differ from the geographical location as the address not always translates to a specific geographical location. Examples include a store, a warehouse, and a consumer. Other examples are buffer zones, dedicated loading spots, or environmental zones.

Vehicle	A Vehicle entity models any means in or by which someone travels or something is carried or conveyed. This means a vehicle can be a truck, a trailer, a train, an airplane, or even some means of transport that is not invented yet. Vehicles can be coupled, thus a combination of a truck and a trailer are modeled in the OTM as two coupled vehicles.
Route	A Route entity describes how a vehicle geographically moves between two points. Basically, it provides a geographical route between points.
Trip	A Trip entity models the concept that goods will be transported between two (or more) locations. A trip may be linked to a route. A Trip entity itself does not contain much data, all details of a trip are modeled as events that are published on a Trip entity.
Shipment	A Shipment entity models an arbitrary amount of goods that are transported. OTM does not say anything about minimum or maximum sizes or amounts. This means that a shipment might be as small as a letter delivered at an office, but also as large as a trailer full of goods delivered to e.g. a supermarket.
Actor	An Actor entity represents organizations or persons participating in a logistic process in OTM. Examples include persons that receive a parcel and stores that receive a truck full of goods

Table 1: Description of the OTM entities (Open Trip Model, 2022)

2.1.3 Lifecycles

Lifecycles are used in the OTM to express a particular phase in the transport process, which are visualized in the green circle in **Figure 3**. The four phases that are determined in the OTM are explained in **Table 2**.

Planned	This pre-trip phase is all about planning. Events in this phase represent planned events: planning of a logistic operation can be modeled as a series of "planned" Events in OTM. Planned events typically originate from a planning system. For organizations that do not rely on planning systems, this phase can be omitted.
Projected	This phase models projected (or estimated) times. Given a series of "planned" events and some associated "actual" events, projected events can be calculated. E.g. when an event is planned to happen at 8:00, but the associated "actual" event only happens at 8:30, all events that are planned after that delayed event can be projected 30 minutes later. Projections can also take into consideration other factors, such as traffic and weather conditions. It is not mandatory to implement a "projected" endpoint for an OTM server. The OTM standard also has nothing to say about the quality of the projections.
Actual	This on-trip phase models the reality that is happening at present time. As opposed to planned events, actual events typically originate from GPS tracking devices or traffic information systems.
Realized	This post-trip phase can be used to view and analyze a logistic operation in retrospect. Events in this phase are recorded and archived events from the actual phase

Table 2: Description of the OTM lifecycles (Open Trip Model, 2022)

2.2 Officedog

This section is about Officedog. **Subsection 2.2.1** shows the overview page of Officedog. **Subsection 2.2.2** explains the used logistic data in Officedog. **Subsection 2.2.3** discusses the event entity and the events page in Officedog. **Subsection 2.2.4** explains the use of several utilities in Officedog. **Subsection 2.2.5** shows the integration area for partners of Officedog. **Subsection 2.2.6** contains an overview of the Jupyter Notebook Template to set up a connection to Officedog.

2.2.1 Overview page

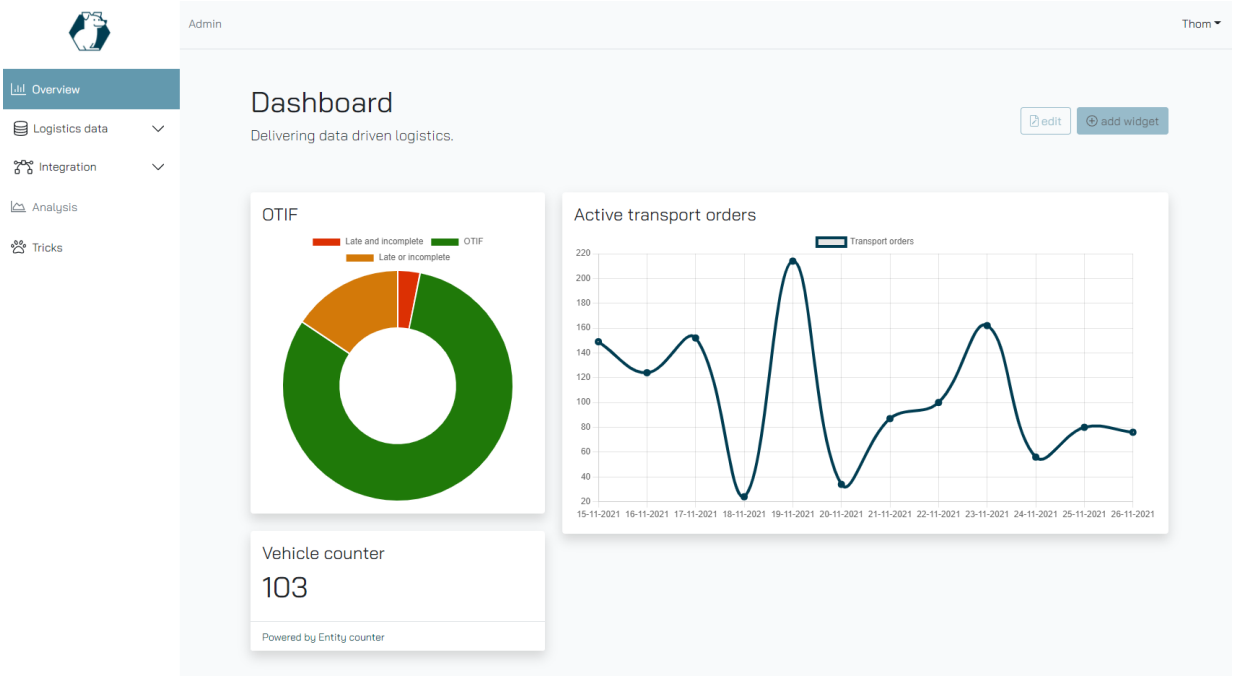

Figure 4: Overview page of Officedog (Bullit Digital, 2022)

The first thing that can be seen when looking at the platform Officedog is the overview page, which contains a dashboard, as visualized in **Figure 4**. The dashboard ensures the user can get a quick insight into a couple of visualized graphs and charts. Furthermore, there can be seen the sub-sections of Officedog, which are logistics data, integration, analysis, and tricks. These will be discussed now.

2.2.2 Logistics data

The logistic data section of Officedog stores all logistics data that has been collected on the platform. The logistics data has been divided into static and dynamic entities. Whereas static entities are entities that are fixed, and dynamic entities can change over time. All of the different entities used within the logistic data can be seen in **Figure 5**. These entities can contain metadata from users of Officedog, which is in line with the OTM. Furthermore, there can be seen that there has been added utilities and database access in this section. However, the user can only access the utility section, as will be discussed in **Subsection 2.2.4** later.

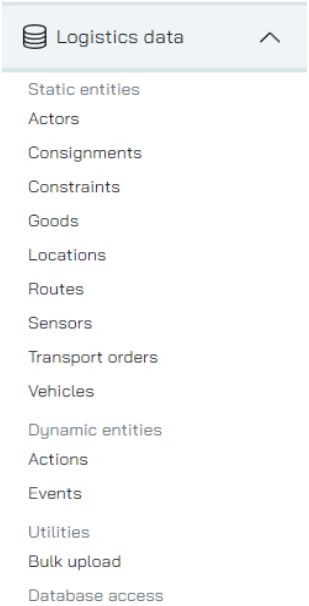

Figure 5: Logistics data in Officedog (Bullit Digital, 2022)

2.2.3 Event entity

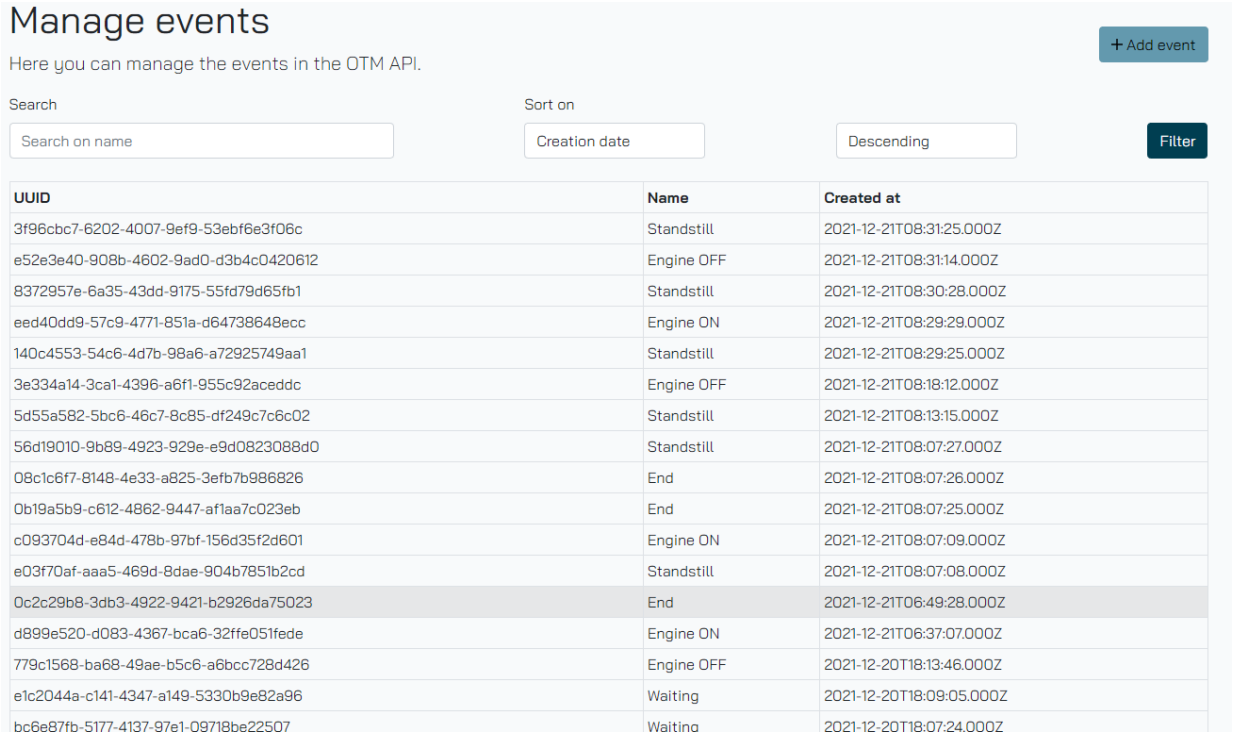

Figure 6: Event data in Officedog (Bullit Digital, 2022)

As explained in **Subsection 2.2.2**, a couple of different data entities are available in Officedog. However, since this research aims to design standardized process mining applications for Officedog, event data is essential since it forms an event log of trips necessary for process mining. **Figure 6** shows the event data section and several events imported from a trip of EMONS Group. This data has been formatted based on the OTM and can be used to make standardized process mining applications for Officedog. Furthermore, in the event section of Officedog, it is also possible to manually import events which the plus button that has been made in the upper right corner, as can be seen in **Figure 6**. Therefore, Officedog supports both manual and automatic imports to the platform.

Figure 7 shows an example of a single event that contains metadata about this event. These metadata contain information that can be used in this research to create standardized process mining applications. Furthermore, if a user decides to upload an event manually, the forms can be filled in, and there can be seen how the OTM transforms the data in the preview window.

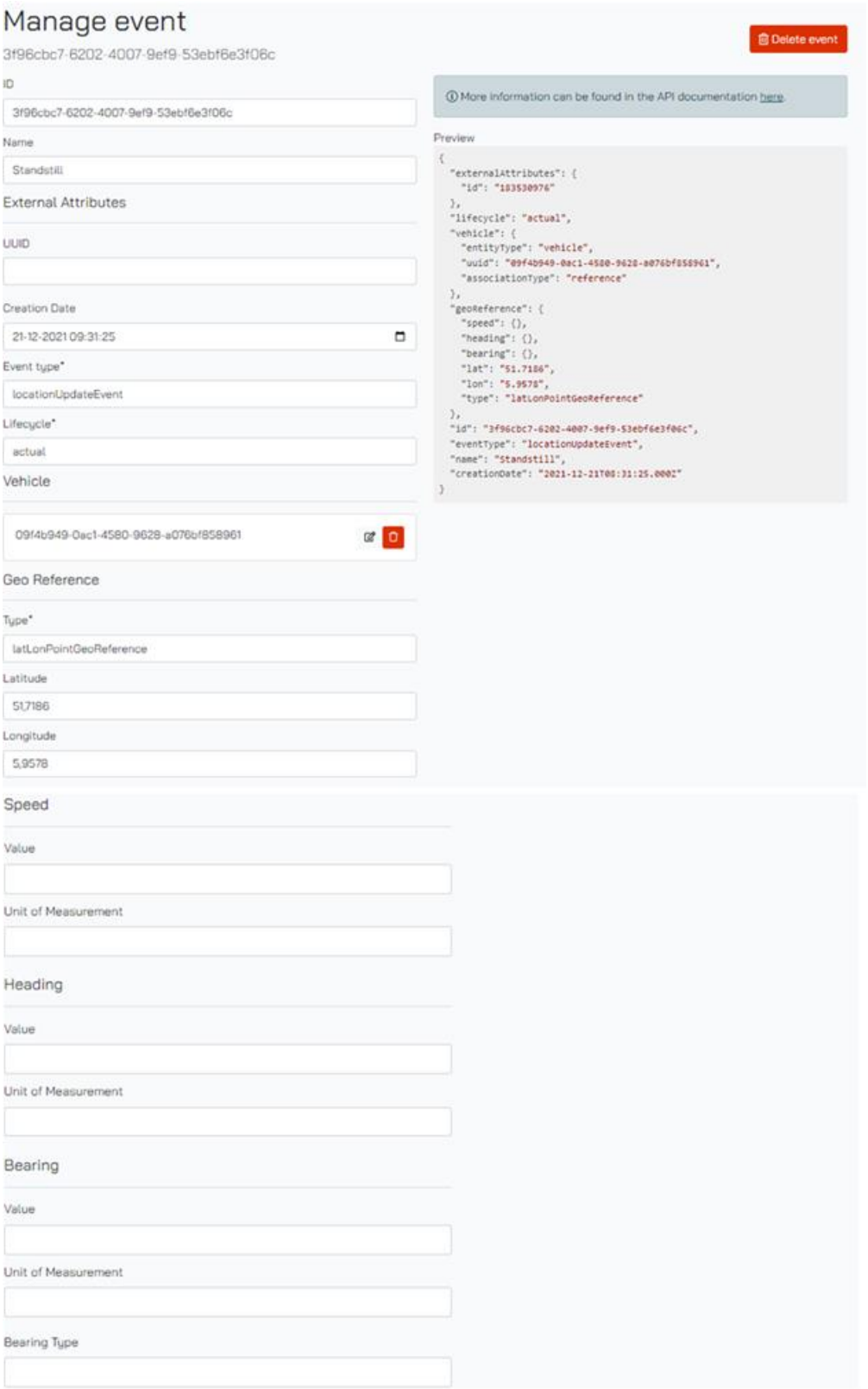

Figure 7: Singe event in Officedog (Bullit Digital, 2022)

2.2.4 Utilities

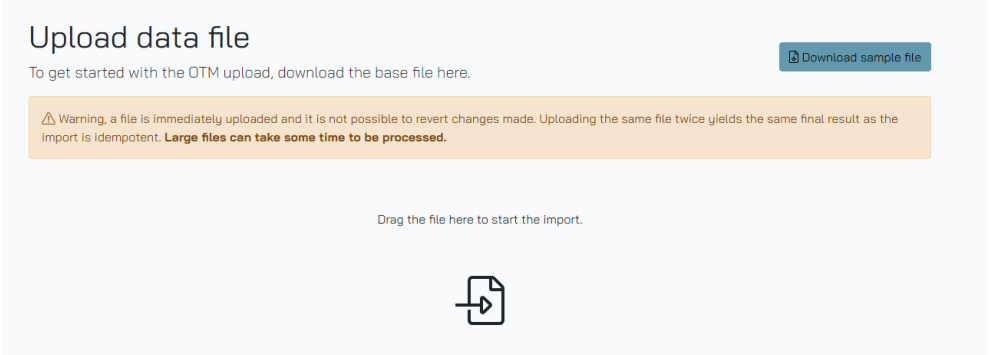

Figure 8: Bulk upload feature in Officedog (Bullit Digital, 2022)

Officedog also supports the option to upload a standardized Excel template, referred to as a bulk upload. The standardized Excel template ensures that data is uploaded in the correctly formatted way that is in line with the OTM. **Figure 9** shows the template where each tab in excel represents the data for one entity in Officedog.

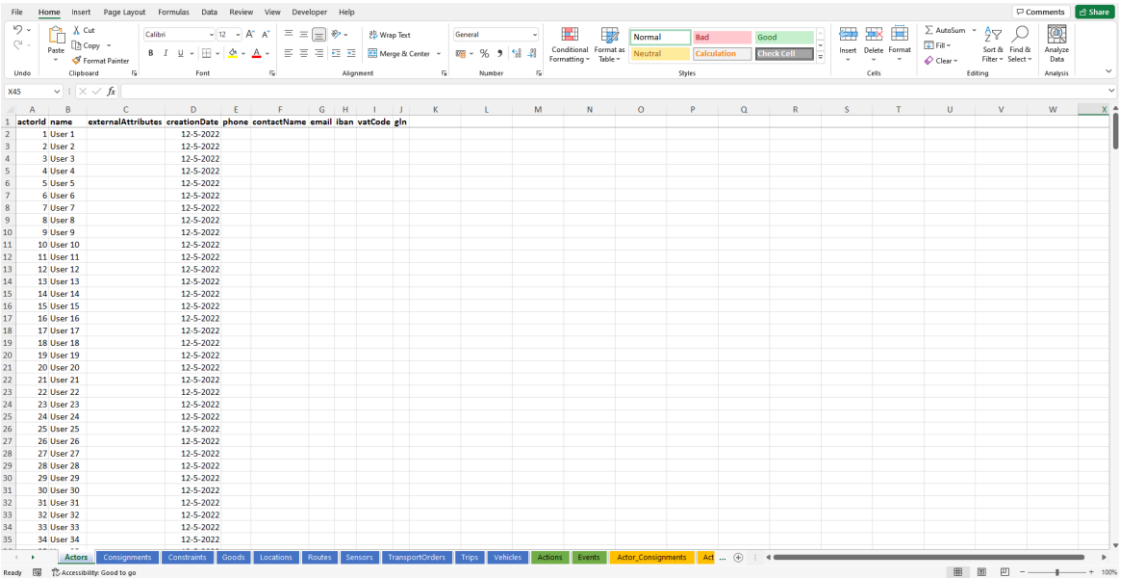

Figure 9: Excel template for bulk upload (Bullit Digital, 2022)

2.2.5 Integration

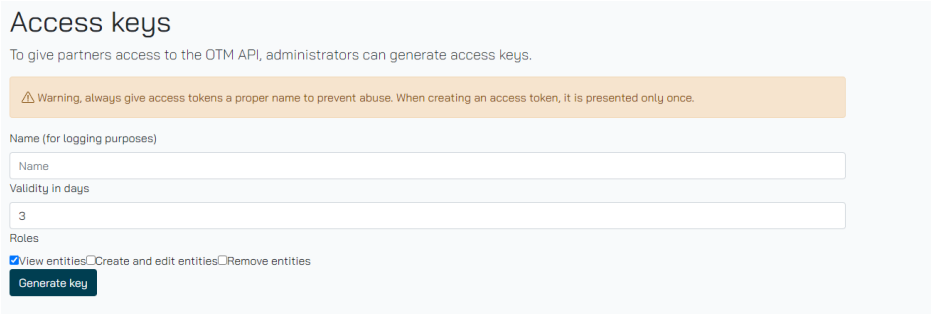

Figure 10: Integration section (Bullit Digital, 2022)

Bullit Digital has made an integration feature to give partners of Officedog access to the data collected in the entities. This feature can be seen in **Figure 10,** where a key will be generated for the partner that gives them access to the platform for the desired number of days.

2.2.6 Jupyter Notebook template

Bullit Digital has provided a Jupyter Notebook template that makes it easier to set up a connection to the Officedog database. This template is listed in **Appendix A** and later used as a starting point for developing standardized process mining applications in **Chapter 5**. The Jupyter Notebook template helps with connecting to the Officedog database from the Jupyter Notebook environment used in this research.

2.3 Summary and conclusion

In this chapter, there has been discussed important content surrounding the OTM and Officedog. The OTM is a semantic standard and formats data in a unified way. The core elements of the OTM are events, entities, and lifecycles. Due to the unified way of handling data, standardized process mining applications can be developed based on the OTM for Officedog. Officedog is a platform based on the OTM and makes it possible for partners to see logistical data in real-time. Therefore, the partners of Officedog can all use developed standardized process mining applications based on the OTM since all data is saved in the same way. Furthermore, the Jupyter Notebook template provided by Bullit Digital can be used to develop these standardized process mining applications. The Jupyter Notebook template ensures that the connection between Officedog and the Jupyter Notebook will be set up correctly.

Chapter 3 – Literature review

The literature review is being conducted to see how the research questions could be solved from a theoretical perspective to understand better how the problem can be solved. **Section 3.1** focuses on business process models (BPM) since these are important in executing process mining. **Section 3.2** discusses several process mining techniques that can be considered for creating standardized process mining applications for Officedog. **Section 3.3** elaborates on past use cases of process mining in logistics. Finally, **Section 3.4** summarizes and concludes this chapter.

3.1 Business process models (BPM)

This section is about BPM. **Subsection 3.1.1** discusses the BPMN method to model a business process. **Subsection 3.1.2** explains the use of a Petri net as a method to model business processes. **Subsection 3.1.3** describes the use of a Workflow net as a method to model a business process. Finally, **Subsection 3.1.4** shows the YAWL to model business processes.

Businesses use business process models to visualize processes that are going on inside a company. These models give an image of how companies think their processes occur and are helpful for insight into a company. By conducting process mining, there will also be made a business process model that indicates the process based on an event log. There are a couple of business process models that are used frequently by companies which will now be discussed briefly.

3.1.1 Business process model and notation (BPMN)

BPMN is a renowned method used as a notation of business processes. It was developed by an industry consortium BPMI.org and quickly became one of the standard notation methods since its official release in February 2006 (Recker, 2010). The four main elements of BPMN are flow objects, artefacts, connecting objects, and swimlanes (Weske, 2020). These elements have been visualized in **Figure 11** to give an idea of how these elements appear in BPMN diagrams.

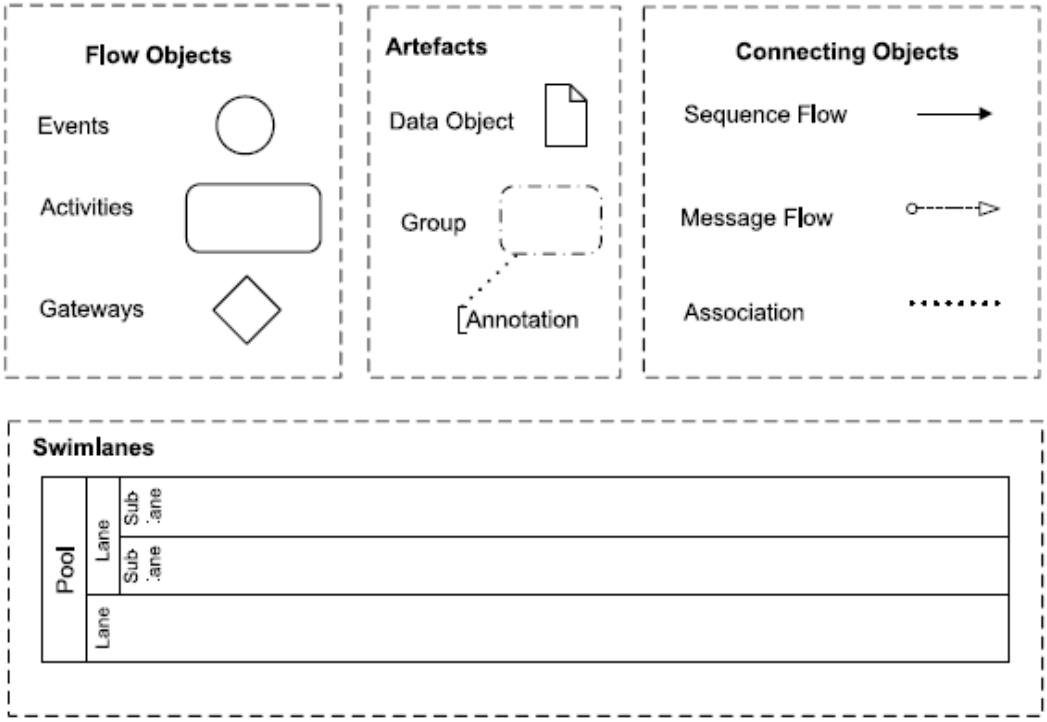

Figure 11: The four main elements of BPMN (Weske, 2020)

Flow objects

Flow objects are the core of BPMN and consist of events, activities, and gateways (Weske, 2020). Events can be seen as states that can occur in a business process from a company (Weske, 2020). Activities represent work units performed during business processes (Weske, 2020). Finally, gateways

are used in BPMN to clarify the split and join behavior of the events, activities, and gateways (Weske, 2020).

Gateways

There are several types of gateways that can be used in BPMN. **Figure 12** shows several examples of these gateways, which all serve a different purpose (Weske, 2020).

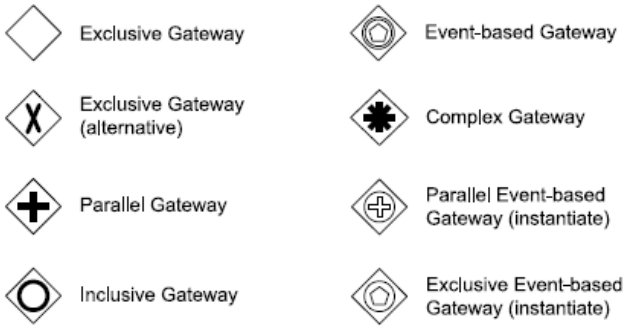

Figure 12: Types of gateways (Weske, 2020)

Artefacts

Artefacts are used in BPMN to address additional information that is not necessarily essential for the flow of the process (Weske, 2020). Data objects signify that data is used between several flow objects (Weske, 2020). Finally, annotations specify aspects of processes, and groups can be used to group elements of a process (Weske, 2020).

Connecting objects

Connecting objects connect flow objects, swimlanes, or artefacts (Weske, 2020). Sequence flow can show the order of flow objects, while message flow indicates the flow of messages between business partners represented by pools (Weske, 2020). Furthermore, association objects are used to link artefacts to elements in BPMN (Weske, 2020).

Swimlanes

A swimlane represents a part of an organization that is active in a process, which could be multiple parts, thus swimlanes.

BPMN Example

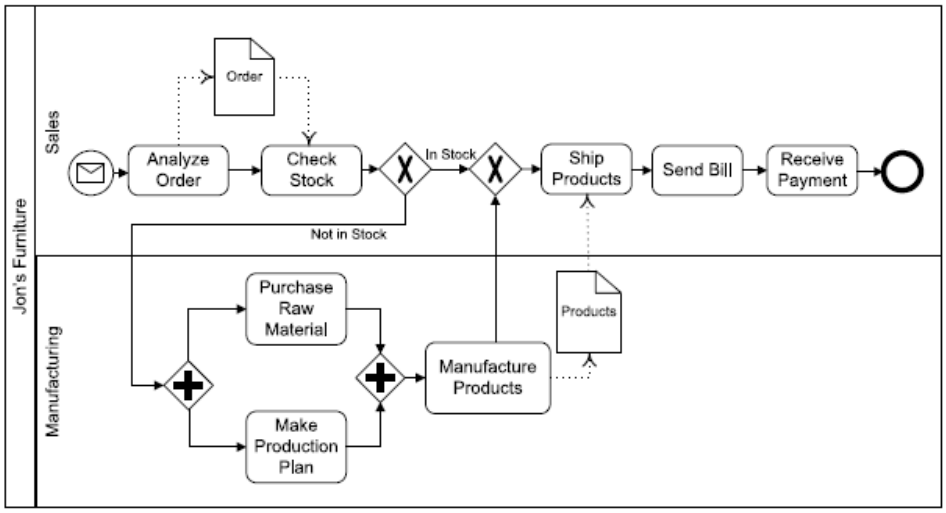

Figure 13: BPMN diagram (Weske, 2020)

Figure 13 shows an example of a BPMN diagram that includes all four main elements and a couple of different gateways.

3.1.2 Petri nets

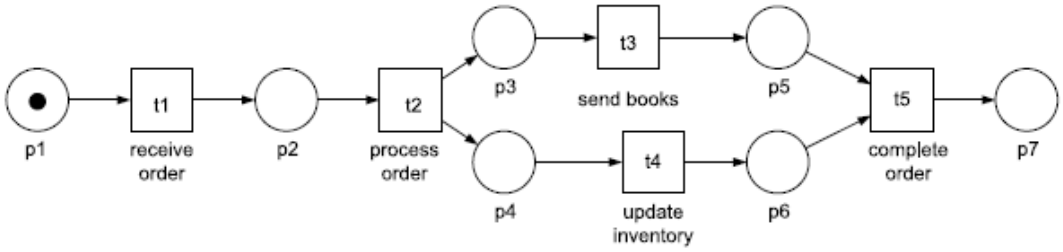

Figure 14: Petri net of a process (Weske, 2020)

Another widely adopted BPM type is a Petri net. A Petri net is a BPM notation developed by Adam Petri and is one of the best-known techniques for specifying business processes (Weske, 2020). A Petri net contains places, transitions, and arcs. It distributes tokens over places by firing them if they are enabled to do this. **Figure 16** shows a Petri net where places are visualized by the p's and the tokens by a black dot inside the places. Furthermore, transitions are visualized by t's and can be enabled if there are tokens in places in front of transitions. For example, in the case of t5, both p5 and p6 should contain a token to enable the transition.

3.1.3 Workflow nets

Workflow nets are a BPM type similar to Petri nets since they use the same notation required for modeling a process model as Petri nets (Weske, 2020). However, Petri nets are often only valid for simple modeling purposes, while Workflow nets are more complex and can often contain multiple parties (Weske, 2020). **Appendix B** contains an example of a Workflow net that consists of multiple parties (Weske, 2020)

3.1.3 Yet Another Workflow Language (YAWL)

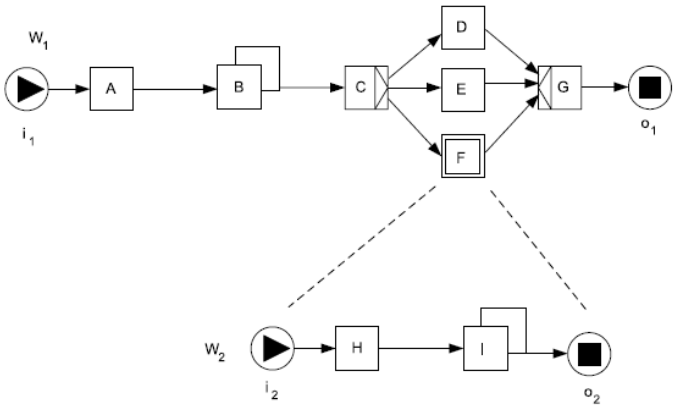

Figure 15: Example of a model created by YAWL(Weske, 2020)

The YAWL is another BPM type created to model business processes. YAWL is different from traditional workflow nets since it uses direct arcs between transitions, explicit split and join behavior that can be attached to transitions, nonlocal behavior (on the firing of a transition, parts of the YAWL net are cleansed of tokens), and the handling of multiple instances tasks (Weske, 2020). **Appendix C**, contains an overview of the notational elements used in YAWL. **Figure 15** shows an example of a process model created with YAWL (Weske, 2020)

3.2 Process mining techniques

This section is about process mining techniques. **Subsection 3.2.1** explains the use of process discovery. **Subsection 3.2.2** describes conformance checking and its use cases. **Subsection 3.2.3** shows how enhancement can be used as a process mining technique.

3.2.1 Process discovery

Process discovery is a well-known process mining technique that researchers often use. It is used to take an event log and produce a model without using prior information (van der Aalst, 2012). So the technique tries to model a process based on example behaviors stored in event logs (van der Aalst, 2012)

Event logs

Event logs are a fundamental principle for many process mining techniques. An event log contains information about an activity, often referred to as a case (van der Aalst, 2012). These event logs contain information about cases that multiple process mining techniques can analyze. These techniques are process discovery, conformance checking, and process enhancement. The information included in event logs is based on historical data generated in a company's process. Some of the most common information in event logs are persons involved in a case which is referred to as the "resource," starting/end times often referred to as the "timestamp," and "data elements," which could be the size of orders (van der Aalst, 2012).

Use of process discovery

Businesses have many procedures that can be used in certain situations. These could be made in an information system, but often these are not documented (van der Aalst, 2012). Process discovery is a process mining technique focused on visualizing and documenting a model representing a situation occurring frequently (van der Aalst, 2012). The situation can be modeled based on the company's collected event logs. Because the analysis of event logs is central to process mining, process mining is often referred to as "evidence-based" Business Process Management (BPM) which is a conventional technique for modeling business processes (van der Aalst, 2012). Process discovery uses the event data and can honestly represent business processes, whereas BPM models are often not based on scientific data (van der Aalst, 2012). Thus, process discovery can give insight into business processes using event data.

Algorithm for process discovery

#	trace
455	acdeh
191	abdeg
177	adceh
144	abdeh
111	acdeg
82	adceg
56	adbeh
47	acdefdbeh
38	adbeg
33	acdefbdeh
14	acdefbdeg
11	acdefdbeg
9	adcefcdeh
8	adcefdbeh
5	adcefbdeg
3	acdefbdefdbeg
2	adcefdbeg
2	adcefbdefbdeg
1	adcefdbefbdeh
1	adbefbdefdbeg
1	adcefdbefcdefdbeg
1391	

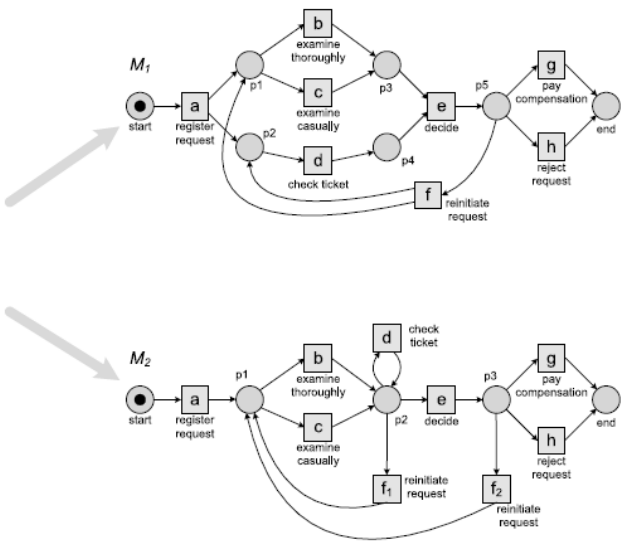

Figure 16: Petri nets created based on the α-algorithm (van der Aalst, 2012)

One of the most used algorithms is the α-algorithm. The α-algorithm is based on the idea that an event log can be investigated if specific patterns occur (van der Aalst, 2012). An example is when activity *a* is followed by *b*, but *b* is never followed by *a,* which shows a causal relation between *a* and *b* (van der Aalst, 2012). The search for patterns using the α-algorithm results in a collection of patterns that can be visualized using BPM notations (van der Aalst, 2012). **Figure 16** shows Petri nets M_1 and M_2, created by performing process discovery on an event log (van der Aalst, 2012).

3.2.2 Conformance checking

Conformance checking is a process mining technique used to compare an existing process model with an event log of the same process (van der Aalst, 2012). Conformance checking enables a company to see whether their created process model is in line with the created event logs of that process (van der Aalst, 2012).

Use of conformance checking

Conformance checking is a powerful process mining technique that shows how a business's current BPM model deviates from reality. However, there might be differences since conformance checking uses a model created based on the event log, which means it uses facts. In addition, businesses often use BPM models not based on historical data, whereas models based on process mining use the historical data from event logs (van der Aalst, 2012).

Dimensions for conformance checking

A couple of dimensions can be used to check for differences between the original and discovered models based on an event log. These dimensions are fitness, simplicity, precision, and generalization (van der Aalst, 2011).

Fitness

Fitness determines the amount of traces in the log that the model can recreate. A model has a perfect fit if all traces in an event log can be found back in the described model (van der Aalst, 2011). Similarly, a model has a low fitness if little traces from an event log can be found back in the model. Furthermore, it is common to describe fitness by a number between 0 and 1 (van der Aalst, 2011).

Simplicity

Simplicity is essential when building models by process mining. In a situation where multiple models can explain the traces of an event log, one should choose the simplest model to be the main model (van der Aalst, 2011). However, one should watch out that a simple model might not fit once another event log is used.

Precision

Precision has to deal with the amount a model behaves according to the design, where a model is precise if it does not allow for too much change (van der Aalst, 2011). It is underfitting if it has not been expressed with precision (van der Aalst, 2011). This underfitting causes behavior in a model different from the log, which is undesirable (van der Aalst, 2011).

Generalization

Generalization is another dimension of conformance checking. In the design process of a model based on event logs, there should be watched out for generalization of the findings of one event log. In contrast to the dimension of precision, the generalization dimension deals with the risk of overfitting a model (van der Aalst, 2011). If a model is overfitting, it is too focused on the behavior of one specific event log and does not pay enough attention to other possible traces that might be found in other event logs.

Algorithms for conformance checking

There will now be discussed some standard conformance checking approaches

Abstraction

The first approach for conformance checking is the abstraction method. First, one should take an abstraction of the behavior of an event log and initial the model (van der Aalst, 2012). These are referred to as footprints, and examples are given in **Figure 18**. Note that these models are based on the event log in **Appendix D**.

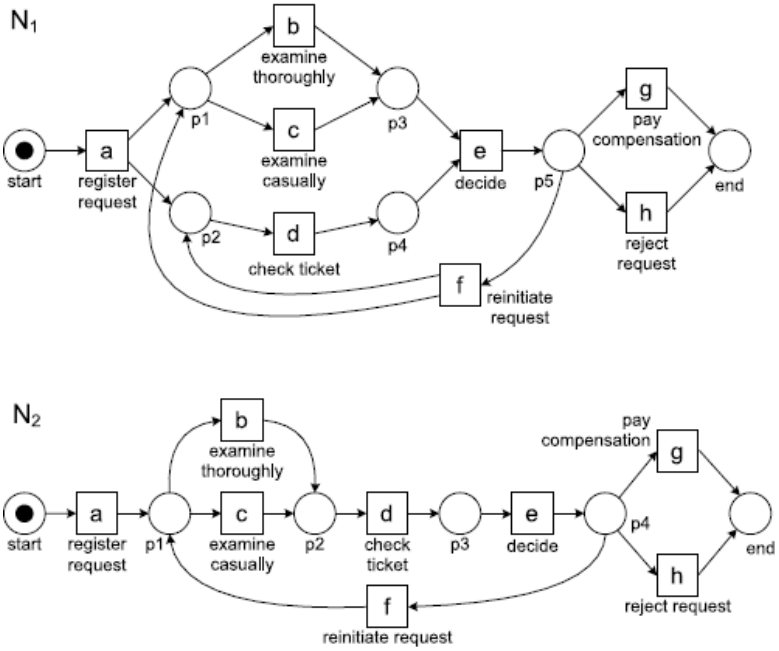

Figure 17: A process visualized by N_1 and N_2 (van der Aalst, 2011)

	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
a	#	→	→	→	#	#	#	#
b	←	#	#	‖	→	←	#	#
c	←	#	#	‖	→	←	#	#
d	←	‖	‖	#	→	←	#	#
e	#	←	←	←	#	→	→	→
f	#	→	→	→	←	#	#	#
g	#	#	#	#	←	#	#	#
h	#	#	#	#	←	#	#	#

	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
a	#	→	→	#	#	#	#	#
b	←	#	#	→	#	←	#	#
c	←	#	#	→	#	←	#	#
d	#	←	←	#	→	#	#	#
e	#	#	#	←	#	→	→	→
f	#	→	→	#	←	#	#	#
g	#	#	#	#	←	#	#	#
h	#	#	#	#	←	#	#	#

Figure 18: Footprints of N_1 (Left) and N_2 (Right) (van der Aalst, 2011)

Figure 17 shows the models N_1 and N_2, modeled after the same trace of an event log. N_1 is modeled after the α-algorithm, and N_2 is a sequential model. The footprints in **Figure 18** use a couple of signs to determine relations between places. The → states that a transition can go forward to another place, whereas the ← states that the previous transition could come from certain places. The # defines that a transition is not possible in these places. The abstraction method could compare these footprints to see where differences occur. For N_1 and N_2, this has been done in **Figure 19**, which shows that there are 12 differences in the cells between transitions N_1 and N_2. These differences show that the models disagree on how transitions occur. A decision-maker could look at these differences and choose to make an adapted model with the preferred transitions.

	a	b	c	d	e	f	g	h
a				→:#				
b				‖:→	→:#			
c				‖:→	→:#			
d	←:#	‖:←	‖:←			←:#		
e		←:#	←:#					
f				→:#				
g								
h								

Figure 19: Differences between footprints of N1 and N2 (van der Aalst, 2011)

Replays

Another approach to conduct conformance checking is replaying models. Replays look at traces from event logs and put these traces into the existing model and measure the model's fitness. **Appendix E** contains **Figure 54**, a replay of model N_1 with the trace σ_1 = (*a, c, d, e, h)* from **Figure 17** (van der Aalst, 2012). In replays, there is being made use of four counters which are: *p* (produced tokens), *c* (consumed tokens), *m* (missing tokens), and *r* (remaining tokens) (van der Aalst, 2012). **Figure 54** shows no missing tokens, and all produced tokens are consumed. Therefore the fitness can be calculated by the formula visualized in **Figure 20**

$$fitness(\sigma, N) = \frac{1}{2}\left(1 - \frac{m}{c}\right) + \frac{1}{2}\left(1 - \frac{r}{p}\right)$$

Figure 20: Formula for calculating fitness for a single trace (van der Aalst, 2011)

Now by filling in p=7, c=7, m=0, and r=0 (**Figure 54**), one could easily see that the fitness for trace σ_1 is 1. This is because there are no missing tokens found in the replay of the trace in N_1

Appendix F contains **Figure 55**, a replay of model N_2 with the trace σ_2 = (*a, b, d, e, g)*. There can be seen that there are two remaining and two missing tokens. Furthermore, five tokens are produced and consumed, resulting in a fitness of 0.6.

Calculating the fitness for all models using the found traces in the event log will give insight into which model is the most accurate if multiple models are available. **Figure 21** gives the formula that can be used to calculate, where ere L(σ) represents the frequency that a trace occurs in a model and L represents all traces as a whole. Furthermore, **Table 3** contains information about the variables in this formula.

$$fitness(L, N) = \frac{1}{2}\left(1 - \frac{\sum_{\sigma \in L} L(\sigma) \times m_{N,\sigma}}{\sum_{\sigma \in L} L(\sigma) \times c_{N,\sigma}}\right) + \frac{1}{2}\left(1 - \frac{\sum_{\sigma \in L} L(\sigma) \times r_{N,\sigma}}{\sum_{\sigma \in L} L(\sigma) \times p_{N,\sigma}}\right)$$

Figure 21: Formula for calculating fitness of a model (van der Aalst, 2011)

$m_{N,\sigma}$	missing tokens for trace σ at model N
$c_{N,\sigma}$	consumed tokens for trace σ at model N
$r_{N,\sigma}$	remaining tokens for trace σ at model N
$p_{N,\sigma}$	produced tokens for trace σ at model N

Table 3: Variables from the formula for total fitness of a model (van der Aalst, 2011)

Appendix D contains the complete event log used for this research, which can be used for all calculations of individual fitness scores (van der Aalst, 2011). After this, the total fitness of each model can be calculated, which for N_1 and N_2 are visualized in **Figure 22**.

$$fitness(L_{full}, N_1) = 1$$
$$fitness(L_{full}, N_2) = 0.9504$$

Figure 22: Total fitness of models N_1 and N_2 (van der Aalst, 2011)

3.2.3 Enhancement

Enhancement is a process mining technique that focuses on improving or extending an existing process model by using information about the actual process recorded in some event log (van der Aalst, 2012). In comparison to conformance checking, which focuses on the difference between model and reality, enhancement aims at changing or extending the prior model to show, e.g., bottlenecks, service levels, and throughput times (van der Aalst, 2012)

Use of enhancement

Enhancement is a valuable technique to improve already good functioning models, which might need minor improvements. Enhancement can be used in two ways: either repair or extension of a model (van der Aalst, 2011). For the extension of a model, three perspectives can be categorized that could be added to an existing process model. These categories are organization, time , and case (van der Aalst, 2011). The organization perspective aims to get information from recourses not yet discovered in event logs (van der Aalst, 2011). The time perspective looks at the data surrounding events' rate and time (van der Aalst, 2011). Finally, the case perspective looks at the properties of recent cases that happened to extend a model (van der Aalst, 2011).

Literature on process enhancement

Type	Perspective	Articles
Repair	Control-flow	[1] [4] [5] [7] [12] [22] [23] [24] [36]
Extension	Organisation	[2] [3] [8] [9] [10] [11] [13] [14] [15] [16] [17] [18] [19] [20] [21] [25] [26] [27] [28] [29] [30] [31] [32] [33] [35] [37] [38] [40] [41] [42] [43]
	Time	[6] [10] [15] [16] [18] [19] [26] [28] [29] [32]
	Case	[10] [15] [18] [19] [26] [28] [32] [34] [39]

Table 4: Process enhancement studies that have been published (Yasmin et al., 2018)

Table 4 shows the results collected in a recent literature review about process enhancement(Yasmin et al., 2018). The results show 46 studies related to process enhancement which all have been organized according to their type and perspective (Yasmin et al., 2018).

3.3 Logistics & process mining

This research focuses on finding standardized process mining applications for Officedog based on historical data from EMONS Group. Therefore, there has been taken a look at recent studies about process mining and logistics.

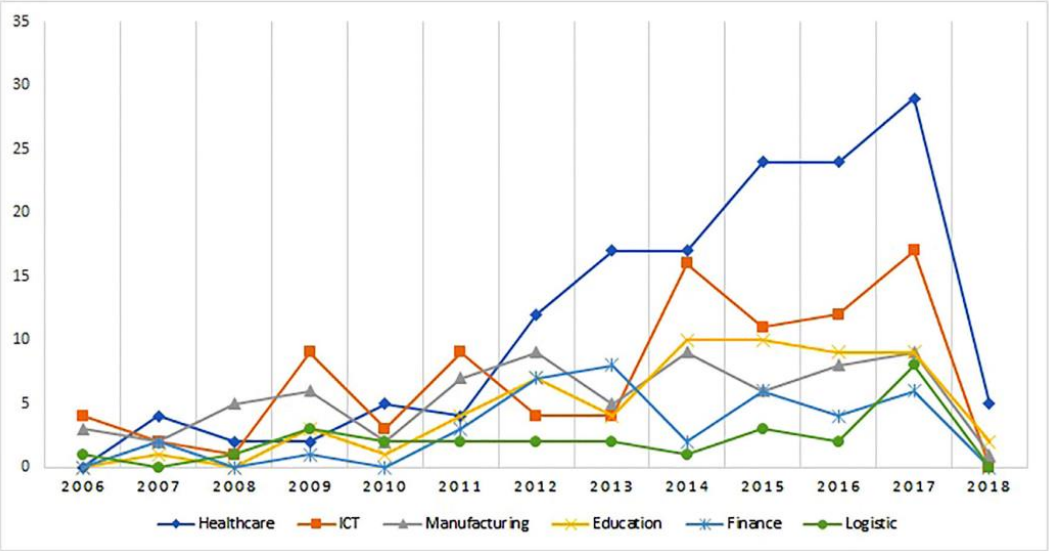

Figure 23: Process Mining Studies in the six most common fields (dos Santos Garcia et al., 2019)

Figure 23 shows the recent studies containing process mining in the six most common fields. There can be seen that 27 studies were conducted from 2006 until 2018, which corresponds to approximately 5% of the total studies for these six topics (dos Santos Garcia et al., 2019). The topics of these research studies were industrial shifts, warehouse layout improvements, bulk ports prediction, traffic control, detecting train rerouting, anomalies in freight, scheduling transport, and resource allocation in and resource allocation in an aircraft company (dos Santos Garcia et al., 2019). **Table 5** shows the process mining topics that these studies are about, where one could see that process discovery is the most coming process mining technique that has been used.

Main Contribution	Studies
Application of context awareness based on machine learning for improving process discovery	Becker and Intoyoad (2017)
Identification of design and operation problems for logistics process based on process mining techniques	Vasyutynskyy, Gellrich, Kabitzsch, and Wustmann (2010)
Multi-dimensional process mining approach	Sutrisnowati, Yahya, Bae, Pulshashi, and Adi (2017)
Prediction of event times based on a timed event graph with dynamic arc weights	Kecman and Goverde (2013)
Process discovery and social network analysis applied in a logistical context	Besri and Boulmakoul (2017), Li (2010)
Process discovery applied in a logistical context	Janssenswillen, Depaire, and Verboven (2017), Gou et al. (2016), Song, Jacobsen, Ye, and Ma (2016) Pulshashi, Bae, Sutrisnowati, Yahya, and Park (2015) Sutrisnowati et al. (2015), Sutrisnowati, Bae, Park, and Ha (2013), Krathu, Pichler, Zapletal, and Werthner (2012), Rozsnyai, Lakshmanan, Muthusamy, Khalaf, and Duftler (2012), Gellrich, Wagner, Vasyutynskyy, and Kabitzsch (2011), Haigh and Yaman (2011), Gerke et al. (2009), Gerke (2008), Gonzalez, Han, and Li (2006)
Process discovery, performance and conformance analyses for logistics processes	Wang, Caron, Vanthienen, Huang, and Guo (2014)
Process model discovery from event streams in a logistical context	Soffer et al. (2019), Repta and Stanescu (2017)
Remodeling logistic business processes based on process discovery	Wang, Zhu, Wang, and Huang (2016), Li and Deng (2009), Li (2009)
Resource allocation	Zhao, Zeng, Zheng, and Yang (2017)
Tools and techniques for analysing a logistics spaghetti process	Kumar, Thomas, and Annappa (2017)

Table 5: The topics from recent studies about process mining (dos Santos Garcia et al., 2019)

3.4 Summary and conclusion

This chapter has discussed literature on process mining and its relation to logistics. First, the topic of BPM and several BPM notations have been shown that are considered essential for generating a model based on an event log analyzed by process mining. BPMN and Petri nets are the most common BPM notations for research based on process mining. Furthermore, there have been studied typical process mining techniques. Process discovery, conformance checking, and process enhancement are the most common process mining techniques. However, most research on process mining in logistics is often focused on the process discovery technique. Now that a vast amount of relevant literature has been discussed, there can be started the design of standardized process mining applications for Officedog based on the historical data from EMONS Group.

Chapter 4 – Solution design

In this section, there have been designed standardized process mining applications. **Section 4.1** discusses possible process mining applications. **Section 4.2** covers how there has been extracted data from Officedog to enable process mining. **Section 4.3** explains the PM4PY library used during the development of standardized process mining applications. **Section 4.4** shows the Kepler library, which is used for geographical visualizations. **Section 4.5** explains the use of Seaborn for statistical analysis of bottlenecks. Finally, **Section 4.6** summarizes and concludes the chapter.

4.1 Possible process mining applications

This section is about possible process mining applications. **Subsection 4.1.1** explains the mapping of activities. **Subsection 4.1.2** shows the driving and rest time analysis for truck drivers.

Now that it is clear that the PM4PY library will be used in a Jupiter Notebook, which is connected to the database of Officedog, it has been decided to look at the possible process mining applications that can be developed for Officedog.

4.1.1 Mapping of activities

The mapping of activities is a powerful tool for a logistic partner of Officedog. It can give insight into areas where bottlenecks are occurring. Two examples can be investigated to see the potential of this standardized process mining application.

Waiting time analysis

The first problem that can be analyzed is the mapping of waiting times. The data contain an activity called "Waiting," meaning that a trucker cannot continue driving for unexpected reasons. Therefore it is interesting to see where this bottleneck is occurring since this can give insight into where this "Waiting" activity is occurring.

Traffic jam analysis

Traffic jams are inevitable when a logistics company conducts many trips. However, the analysis of traffic jams that have occurred could be vital information to have. For example, if there is created a visual mapping of where most traffic jams occur, the logistics company could decide to take a different route.

4.1.2 Driving and Rest time analysis

EMONS Group has provided data, as shown in **Figure 25,** which contains several activities from which the "Driving" and "Rest" time are interesting for process mining. The European Union has provided guidelines for truck drivers for their driving time and rest periods (European Commission, 2006). These have been developed to prevent truck drivers from not getting enough rest and can be used to develop a standardized process mining application. The standard process mining application can be based on the guidelines from the European Union and check if the truck driver gets enough rest by studying the "Driving" and "Rest" activities given in the events. The guidelines have been listed in **Appendix G.** However, the research will only investigate the first and second guidelines due to time constraints.

4.2 The setup of data extraction

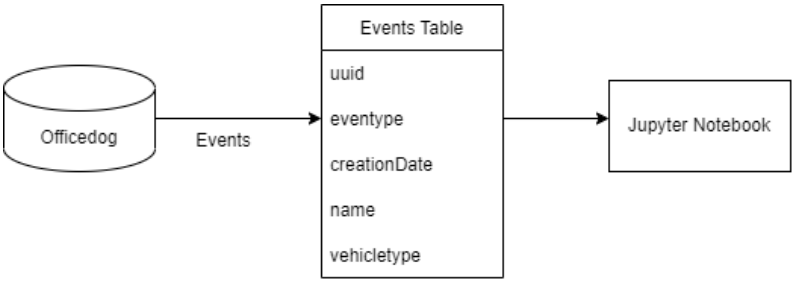

Figure 24: Data flow from Officedog to Jupyter Notebook

In **Figure 24**, there can be seen how Officedog is used to extract the data of events into a Jupyter Notebook. This research uses the programming language Python, which requires a Juptyer Notebook.

In this Jupyter Notebook, a BPM will be made using process discovery, and there will be a detection and analysis of several bottlenecks that could occur for logistics companies. In Officedog, there has been imported an Excel file based on data coming from a board computer of a vehicle from EMONS Group. This data has been visualized in **Figure 25** and is imported into Officedog based on the OTM model, as seen earlier in **Figure 6**.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	DateTime	Vehicle_ID	Longitude	Latitude	Heading	FuelLevel	Odo	Activity_Name	Activity_ActivityType	Location	CountryCode	LoadDate	PosId
2	30-9-2021 00:24	UAC7676	9,952	52,3843	10	90	505342	Rest	ACTIVITY	DEU - Europastraße (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437102
3	30-9-2021 00:52	UAC7676	9,952	52,3843	10	90	505342	Rest	ACTIVITY	DEU - Europastraße (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437103
4	30-9-2021 00:54	UAC7676	9,9519	52,3842	10	90	505342	Rest	ACTIVITY	DEU - Europastraße (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437104
5	30-9-2021 00:54	UAC7676	9,952	52,3842	10	90	505342	Rest	ACTIVITY	DEU - Europastraße (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437105
6	30-9-2021 00:54	UAC7676	9,952	52,3843	10	90	505342	Engine ON	REGISTRATION	DEU - Europastraße (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437106
7	30-9-2021 00:54	UAC7676	9,952	52,3843	10	90	505342	Contact Key On	REGISTRATION	DEU - Europastraße (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437107
8	30-9-2021 00:59	UAC7676	9,9519	52,3843	10	90	505342	Driving	ACTIVITY	DEU - Europastraße (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437108
9	30-9-2021 00:59	UAC7676	9,952	52,3843	10	90	505342	Driving	ACTIVITY	DEU - Europastraße (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437109
10	30-9-2021 01:01	UAC7676	9,9524	52,3862	290	91	505342	Driving	ACTIVITY	DEU - Westtangente (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437110
11	30-9-2021 01:03	UAC7676	9,9488	52,372	170	88	505344	Driving	ACTIVITY	DEU - Westtangente (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437111
12	30-9-2021 01:05	UAC7676	9,9245	52,3662	230	90	505346	Driving	ACTIVITY	DEU - Am Rehwinkel (-) - 31275 - Lehrte//Lehrte	D	30-9-2021 04:05	178437112
13	30-9-2021 01:07	UAC7676	9,9016	52,3533	230	90	505348	Driving	ACTIVITY	DEU - Gartenwinkel (24-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437113
14	30-9-2021 01:09	UAC7676	9,8846	52,3527	330	89	505349	Driving	ACTIVITY	DEU - Gretlade (4-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437114
15	30-9-2021 01:10	UAC7676	9,8845	52,3532	20	90	505350	Driving	ACTIVITY	DEU - Gretlade (4-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437115
16	30-9-2021 01:10	UAC7676	9,8835	52,3539	20	89	505350	Arrival loading	ACTIVITY	DEU - Rebhuhnweg (-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437116
17	30-9-2021 01:11	UAC7676	9,8831	52,3545	20	89	505350	Arrival loading	ACTIVITY	DEU - Gretlade (9-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437117
18	30-9-2021 01:11	UAC7676	9,8831	52,3545	20	89	505350	Contact Key Off	REGISTRATION	DEU - Gretlade (9-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437118
19	30-9-2021 01:11	UAC7676	9,8831	52,3545	20	89	505350	Engine OFF	REGISTRATION	DEU - Gretlade (9-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437119
20	30-9-2021 01:11	UAC7676	9,8832	52,3546	20	89	505350	Arrival loading	ACTIVITY	DEU - Gretlade (7-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437120
21	30-9-2021 01:11	UAC7676	9,8831	52,3546	20	89	505350	Arrival loading	ACTIVITY	DEU - Gretlade (7-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437121
22	30-9-2021 01:14	UAC7676	9,883	52,3546	20	89	505350	Contact Key On	REGISTRATION	DEU - Gretlade (7-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437122
23	30-9-2021 01:14	UAC7676	9,883	52,3546	20	89	505350	Engine ON	REGISTRATION	DEU - Gretlade (7-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437123
24	30-9-2021 01:14	UAC7676	9,8831	52,3547	20	90	505350	Arrival loading	ACTIVITY	DEU - Gretlade (7-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437124
25	30-9-2021 01:16	UAC7676	9,8826	52,3548	20	92	505350	Arrival loading	ACTIVITY	DEU - Rebhuhnweg (-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437125
26	30-9-2021 01:20	UAC7676	9,8823	52,3548	20	93	505350	Loading	ACTIVITY	DEU - Rebhuhnweg (-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437126
27	30-9-2021 01:20	UAC7676	9,8823	52,3548	20	93	505350	Loading	ACTIVITY	DEU - Rebhuhnweg (-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437127
28	30-9-2021 01:20	UAC7676	9,8823	52,3548	20	93	505350	Loading	ACTIVITY	DEU - Rebhuhnweg (-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437128
29	30-9-2021 01:20	UAC7676	9,8823	52,3548	20	92	505350	Loading	ACTIVITY	DEU - Rebhuhnweg (-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437129
30	30-9-2021 01:20	UAC7676	9,8823	52,3548	20	92	505350	Engine OFF	REGISTRATION	DEU - Rebhuhnweg (-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437130
31	30-9-2021 01:20	UAC7676	9,8823	52,3548	20	92	505350	Contact Key Off	REGISTRATION	DEU - Rebhuhnweg (-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437131
32	30-9-2021 01:21	UAC7676	9,8824	52,3549	20	92	505350	Contact Key On	REGISTRATION	DEU - A7/E45 (-) - 31319 - Sehnde/Höver/	D	30-9-2021 04:05	178437132
33	30-9-2021 01:21	UAC7676	9,8824	52,3549	20	92	505350	Engine ON	REGISTRATION	DEU - A7/E45 (-) - 31319 - Sehnde/Höver/	D	30-9-2021 04:05	178437133
34	30-9-2021 01:21	UAC7676	9,8826	52,3548	20	90	505350	Contact Key Off	REGISTRATION	DEU - Rebhuhnweg (-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437134
35	30-9-2021 01:21	UAC7676	9,8826	52,3548	20	90	505350	Engine OFF	REGISTRATION	DEU - Rebhuhnweg (-) - 31319 - Sehnde/Höver/Sehnde	D	30-9-2021 04:05	178437135
36	30-9-2021 01:22	UAC7676	9,8824	52,3549	20	90	505350	Contact Key On	REGISTRATION	DEU - A7/E45 (-) - 31319 - Sehnde/Höver/	D	30-9-2021 04:05	178437136
37	30-9-2021 01:22	UAC7676	9,8824	52,3549	20	90	505350	Engine ON	REGISTRATION	DEU - A7/E45 (-) - 31319 - Sehnde/Höver/	D	30-9-2021 04:05	178437137

Figure 25: Data coming from a board computer of a vehicle by EMONS Group

4.3 PM4PY

This research will use a process mining library for Python called PM4PY that contains practical algorithms for creating standardized process mining applications for Officedog (Fraunhofer FIT, 2021). For example, in **Figure 26,** there can be seen that the PM4PY contains several examples for process discovery and handling of event data.

Figure 26: Example for process discovery (Fraunhofer FIT, 2021)

The library is handy for process discovery since it contains some algorithms that can visualize several types of BPM models needed to visualize the discovered process. **Figure 27** and **Figure 28** show examples of a BPMN and a Petri net that the PM4PY library can create in Python. Furthermore, **Appendix H** contains a list of process mining techniques available in the PM4PY library.

```
import pandas as pd
import pm4py
df = pm4py.format_dataframe(pd.read_csv('data/running_example.csv', sep=';'), case_id='case_id',activity_key='activity',
                            timestamp_key='timestamp')
bpmn_model = pm4py.discover_bpmn_inductive(df)
pm4py.view_bpmn(bpmn_model)
```

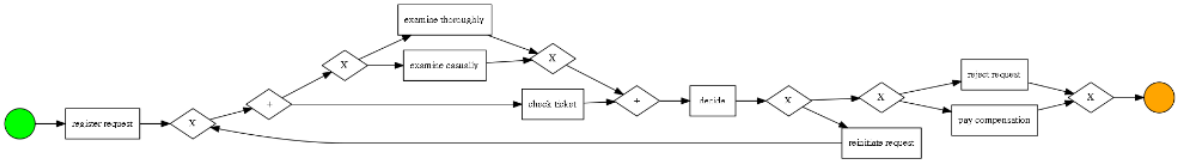

Figure 27: Creation of a BPMN model with PM4PY

```
net1, im1, fm1 = pm4py.convert_to_petri_net(process_tree)
pm4py.view_petri_net(net1,im1,fm1)
```

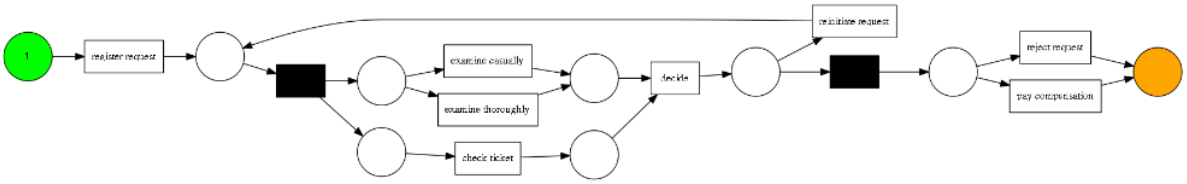

Figure 28: Creation of a Petri net with PM4PY

4.4 Kepler

Another library that is used during the development of standardized process mining applications is Kepler. The Kepler library is an open-source geographical analysis tool for large-scale data created by Uber (Kepler, 2022). The library can be used in multiple data analysis tools, including the Jupyter Notebook environment. Kepler is working based on geographical information provided in data sets and can therefore be indicated to get insight into bottlenecks coming from an event log since all events contain geographical information in Officedog. So, the latitude and longitude can be plotted based on this geographical information to visualize bottlenecks.

4.5 Seaborn

The Seaborn library is being used to make a statistical analysis of the bottlenecks that are occurring. Seaborn is a library for Python that provides a high-level interface for drawing attractive and informative statistical graphics (*Seaborn*, 2022). Furthermore, Seaborn offers a wide range of statistical tools that can be used to give insight into data in a visual way (*Seaborn*, 2022). Therefore, the Seaborn library is helpful for the analysis of found bottlenecks in the extracted events from Officedog.

4.6 Summary and conclusion

This chapter has discussed which standardized process mining applications will be developed. These are the mapping and statistical analysis of activities and an analysis of truck drivers' driving and rest times compared to the European Guidelines. Furthermore, it provided information on several tools that will be used within the Jupyter Notebook template by Bullit Digital. The first tool is the Kepler library which is used for geographical mapping. The second tool is the PM4PY library used for several process mining applications. The last tool used is the Seaborn library for the statistical analysis of activities. Now that it is clear what will be developed, the actual creation of these standardized process mining applications can be started.

Chapter 5 – Standardized process mining applications

This section shows how standardized process mining applications are developed. **Section 5.1** discusses the connection to Officedog. **Section 5.2** shows how a general object has been created. **Section 5.3** explains how process discovery has been used to create a process model. **Section 5.4** contains a visualization of a whole trip. **Section 5.5** shows the mapping and statistical analysis of waiting activities. **Section 5.6** contains the mapping and statistical analysis of traffic jam activities. **Section 5.7** shows the creation of the driving and rest times analysis. Finally, **Section 5.8** summarizes and concludes this chapter.

5.1 Connection to Officedog

The first step in creating the standardized process mining applications was setting up the connection from the Jupyter Notebook to Officedog. **Subsection 5.1.1** explains which dependencies are used for the database connection. **Subsection 5.1.2** shows how database models are defined and instantiated.

5.1.1 Dependencies & database connection

```
!pip install pydantic==1.8.2
!pip install psycopg2-binary==2.9.1
!pip install SQLAlchemy==1.4.26
!pip install pm4py

!pip install wheel
!pip install pipwin

!pipwin install numpy
!pipwin install pandas
!pipwin install shapely
!pipwin install gdal
!pipwin install fiona
!pipwin install pyproj
!pipwin install six
!pipwin install rtree
!pipwin install geopandas
!pip install matplotlib
!pip install keplergl
```

```
from sqlalchemy import create_engine, Table
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, Session, relation
from pydantic import BaseModel
from pydantic.types import UUID4
from sqlalchemy import Boolean, Column, ForeignKey, Integer, String
from sqlalchemy.orm import relationship
from typing import List, Optional

from datetime import datetime

import pandas as pd
import numpy  as np
import seaborn as sns
import pm4py
import keplergl
import geopandas as gpd
import matplotlib
import matplotlib.pyplot as plt
import os
os.environ["PATH"] += os.pathsep + 'C:/Program Files/Graphviz/bin/'
```

Figure 29: Code used to install and load dependencies

The data from this trip needs to be imported into the Jupyter Notebook. Therefore, Bullit Digital has made some code in the Jupyter Notebook template, which installs and loads dependencies, as shown in **Appendix A**. However, several extra libraries were installed and imported. These were required to develop standardized process mining applications, as seen in **Figure 29**. Furthermore, there has been made code that configures the database connection to Officedog, which is shown in **Figure 30**

```
DB_URL = "postgresql://otm-kpi-read@officedog-db-1:hftjhKqkdqugrEZHRzVad8PZ@officedog-db-1.postgres.database.azure.com:5432/otm-k
```

Figure 30: Code used for the database connection

5.1.2 Define and instantiate database models

```
# Import data types
from sqlalchemy.dialects.postgresql import JSONB, ARRAY, ENUM, TIMESTAMP

engine = create_engine(DB_URL)
SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)

Base = declarative_base()

# Define models
class Events(Base):
    __tablename__ = "events"

    id = Column(String, primary_key=True, index=True)
    name = Column(String)
    eventType = Column(String)
    creationDate = Column()
    vehicleId = Column(String, ForeignKey("vehicles.id"), primary_key=True, index=True)
    geoReference = Column(String)
```

Figure 31: Engine & class setup in Jupyter Notebook

After the database connection had been set up, the software package of sqlalchemy was used to extract the data from Officedog into the Jupyter Notebook. As seen in **Figure 31**, an engine has been created, and the class called "Events" is being made, which contains elements from the events entity in Officedog. **Appendix A** showed that multiple connections were already made available by Bullit Digital in the Jupyter Notebook template. However, this "Events" class has been added to the existing database models since it was needed to create standardized process mining applications that require event information. **Figure 32** shows that the linked events have similar names to the names in Officedog to make sure that they can be called.

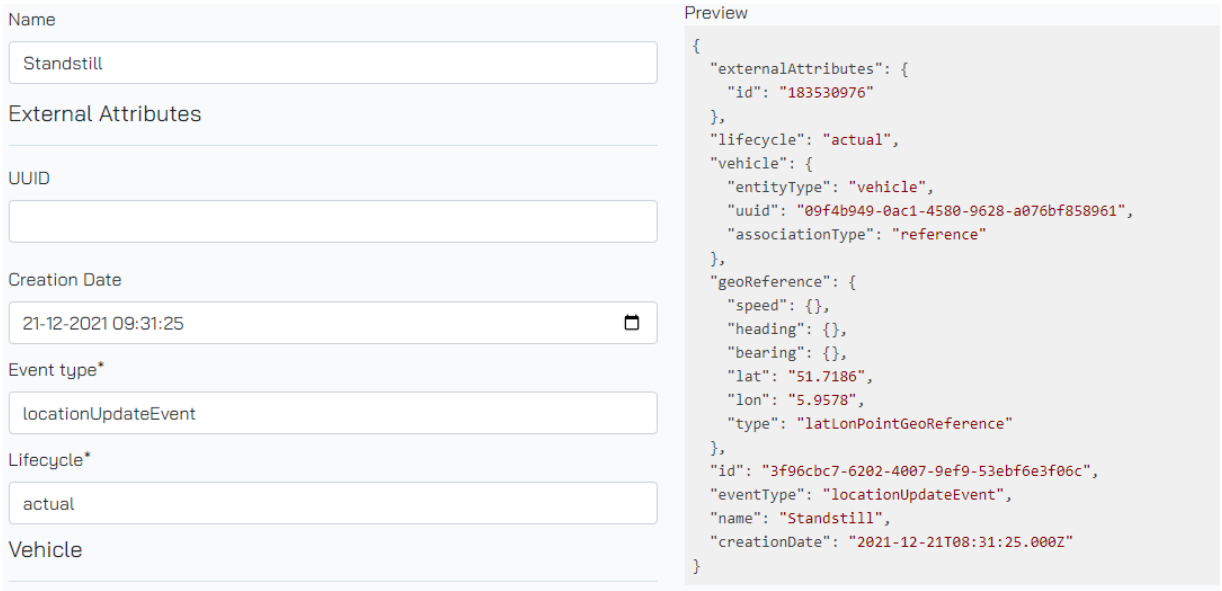

Figure 32: The metadata visualized in Officedog for an event (Officedog, 2022)

5.2 Creation of a general object

```
def select_location(row):       # function to return location based on latitude and longitude
    geolocator = Nominatim(user_agent="geoapiExercises")
    city = 'unknown'
    latitude  = row['lat']
    longitude  = row['lon']
    location = geolocator.reverse(str(round(latitude,2))+","+str(round(longitude,2)))
    address = location.raw['address']
    city = address.get('city', '')
    return city

def get_day_name(row):           # function to get the day name
    naame = 'unknown'
    date  = row['day']
    day_name= ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday','Sunday']
    day = datetime.strptime(str(date), '%Y-%m-%d').weekday()
    return day_name[day]
```

Figure 33: Functions defined for the selection of location and day name

A general object has been created that can be used for creating standardized process mining applications. The first step in this creation was the development of functions that select a location and day name, which can be seen in **Figure 33**. This has been done since mapping activities require a location coupled to an event. Furthermore, a statistical analysis can be supported by the day name since it can show differences between days.

```
class Solver:
  def __init__(self):
    self.event = ""
    self.vehicle_id = ""
    self.eventlog = pd.DataFrame()
    self.events = pd.DataFrame()
    self.events_red = pd.DataFrame()
    self.random = ""

  def loadtripseventlog(self):
    df = pd.read_sql(db.query(Events).statement, db.bind)    # create data frame
    df['RAND'] = [ random.randint(1,50)  for k in df.index]  # add a number 1 to 50 to each event as a new vehicle id
    df.columns = ['case_id', 'activity', 'event', 'timestamp','vehicleId', 'geoReference','RAND'] # identify columns
    df = pm4py.format_dataframe(df, case_id='RAND',activity_key='activity',
                                timestamp_key='timestamp')      # format based on the PM4PY requirements
    del df['case_id']                                           # remove useless columns
    del df['activity']
    del df['timestamp']
    self.eventlog = df

  def loadeventlog(self):
    df = pd.read_sql(db.query(Events).statement, db.bind)    # create data frame
    df.columns = ['case_id', 'activity', 'event', 'timestamp','vehicleId', 'geoReference'] # identify columns
    df = pm4py.format_dataframe(df, case_id='case_id',activity_key='activity',
                                timestamp_key='timestamp')      # format based on the PM4PY requirements
    del df['case_id']                                           # remove useless columns
    del df['activity']
    del df['timestamp']
    self.eventlog = df

  def loadallevents(self):
    df_events = self.eventlog.sort_values(by='time:timestamp')
    df_geo = dict(df_events.geoReference)                       # add latitude and longitude as column
    df2 = pd.DataFrame.from_dict(df_geo).T
    df2 = df2[["lat", "lon"]]

    df_events = pd.concat([df_events, df2], axis=1, join='inner') # add this to the case id in the events dataframe
    df_events = df_events.loc[:,~df_events.columns.duplicated()].copy()

    df_events['lat'] = pd.to_numeric(df_events['lat'])      # format type from string to numeric
    df_events['lon'] = pd.to_numeric(df_events['lon'])      # format type from string to numeric

    self.events = df_events

  def loadevents(self, event, vehicle_id):
    self.event = event
    self.vehicle_id = vehicle_id
    df_events = self.eventlog.sort_values(by='time:timestamp')    # sort on timestamp
    df_events['row_id'] = df_events.reset_index().index
    df_events = df_events.loc[df_events['concept:name'] == event] # only the waiting activities

    df_geo = dict(df_events.geoReference)                         # add latitdue and longtidue as column
    df2 = pd.DataFrame.from_dict(df_geo).T
    df2 = df2[["lat", "lon"]]

    df_events = pd.concat([df_events, df2], axis=1, join='inner') # add this to the case id in the events dataframe
    df_events = df_events.loc[:,~df_events.columns.duplicated()].copy() # remove duplicates

    df_events['lat'] = pd.to_numeric(df_events['lat'])            # format type from string to numeric
    df_events['lon'] = pd.to_numeric(df_events['lon'])            # format type from string to numeric

    self.events = df_events
```

```
def view_bpm(self):
  tree = pm4py.discover_process_tree_inductive(self.eventlog)   # create a process model with PM4PY
  net, initial_marking, final_marking = pm4py.convert_to_petri_net(tree) # convert to Petri net
  pm4py.view_petri_net(net, initial_marking, final_marking)

def calculateStateDuration(self):
  df_events = self.events
  df_events.loc[df_events['row_id']- df_events['row_id'].shift(1) != 1, 'start/end'] = 'start' # indicate start
  df_events.loc[df_events['row_id'].shift(-1)- df_events['row_id'] != 1, 'start/end'] = 'end'  # indicate end
  df_events_red = df_events[['row_id', 'time:timestamp','concept:name','start/end', 'lat', 'lon']].dropna()# select columns
  startend = df_events_red['time:timestamp'].diff()                                           # calucate activity duration
  df_events_red = pd.concat([df_events_red, startend], axis=1)                                # add this to the dataframe
  df_events_red.columns = ['row_id', 'time:timestamp', 'concept:name', 'start/end','lat', 'lon', 'duration']
  df_events_red = df_events_red[~df_events_red['start/end'].isin(['start'])]                  # only show end events

  avg_waiting_time = df_events_red["duration"].mean()           # calculate average waiting time
  print("avg_waiting_time is " + str(avg_waiting_time))

  df_events_red['duration'] = df_events_red.duration.apply(lambda x: x.seconds) # calculate duration of activity
  df_events_red['duration'] = df_events_red['duration'] / 3600                  # convert to hours

  self.events_red = df_events_red

def addLocationInfo(self):
  self.events_red['city'] = self.events_red.apply(lambda x: select_location(x), axis =1) # add location
  self.events_red.drop(['lat', 'lon'], axis=1, inplace=True)

def addDayInfo(self):
  self.events_red['day'] = self.events_red['time:timestamp'].dt.date                    # add day info
  self.events_red['day_name'] = self.events_red.apply(lambda x: get_day_name(x), axis =1)

def showdataframe(self):
  display (self.events_red)

def drivingtime(self):
  self.events_red = self.events_red.groupby('day')['duration'].agg(['sum','count']) # calculate driving time per day
  self.events_red = self.events_red.loc[self.events_red['sum'] > 9]             # filter out driving times higher than 9 hours

  display (self.events_red)

def drivingtimeperweek(self):
  self.events_red ['day'] = pd.to_datetime(self.events_red['day'], errors='coerce') # change to datetime
  self.events_red ['week_number_of_year'] = self.events_red['day'].dt.week       # add week number

  # Calculate weekly driving time
  self.events_red = self.events_red.groupby('week_number_of_year')['duration'].agg(['sum','count'])

  display (self.events_red)

def getEventLog(self):
  return self.eventlog

def getEvents(self):
  return self.events

def getEvents_red(self):
  return self.events_red

def displayEvents(self):
  display (self.events)

def displayEvents_red(self):
  display (self.events_red)

def showDurationPlots(self):
  sns.boxplot(data=self.events_red, x=self.events_red.duration) # duration in hours
  sns.displot(self.events_red, x="duration", binwidth=0.001, discrete=True)

def displayDurationStats(self):
  display(self.events_red.describe())
  display(self.events_red.groupby('day').duration.agg(['max', 'min', 'count', 'median', 'mean', 'sum']))
  display(self.events_red.groupby('day_name').duration.agg(['max', 'min', 'count', 'median', 'mean', 'sum']))
  display(self.events_red.groupby('city').duration.agg(['max', 'min', 'count', 'median', 'mean', 'sum']))
```

Figure 34: Object containing several functions to enable object-orientated programming

After the functions for selecting a location and day name have been created, the " Solver " object has been made, as seen in **Figure 34**. This object contains several functions that can be used for the geographical mapping of activities and statistical analysis. To see how this can be used for mapping activities and a statistical analysis, there has been a demonstration of the "Waiting" and "Traffic Jam" activities. Furthermore, the objective can be used to make a process model using the PM4PY library.

5.3 Process discovery

This section is about process discovery. **Subsection 5.3.1** discusses the obtained Petri net for one vehicle id. **Subsection 5.3.2** covers the Petri net discovered if multiple vehicle id's would be considered.

5.3.1 Process discovery for one vehicle id

```
p = Solver()
p.loadeventlog()
p.view_bpm()
```

Figure 35: Code used to generate a Petri net for one vehicle id

Figure 35 shows the code used to obtain the Petri net visualized in **Figure 36**. The PM4PY library was used to create the Petri net and resulted in a simple process model. The code in **Figure 35** uses the function "loadeventlog" written in the class "Solver" as seen in **Figure 34**. First, the events were loaded into a data frame formatted to be supported by the PM4PY library. This is because PM4PY works with standard names, e.g., " id" should always be "case_id. " After creating the data frame, a Petri net was made using the PM4PY library.

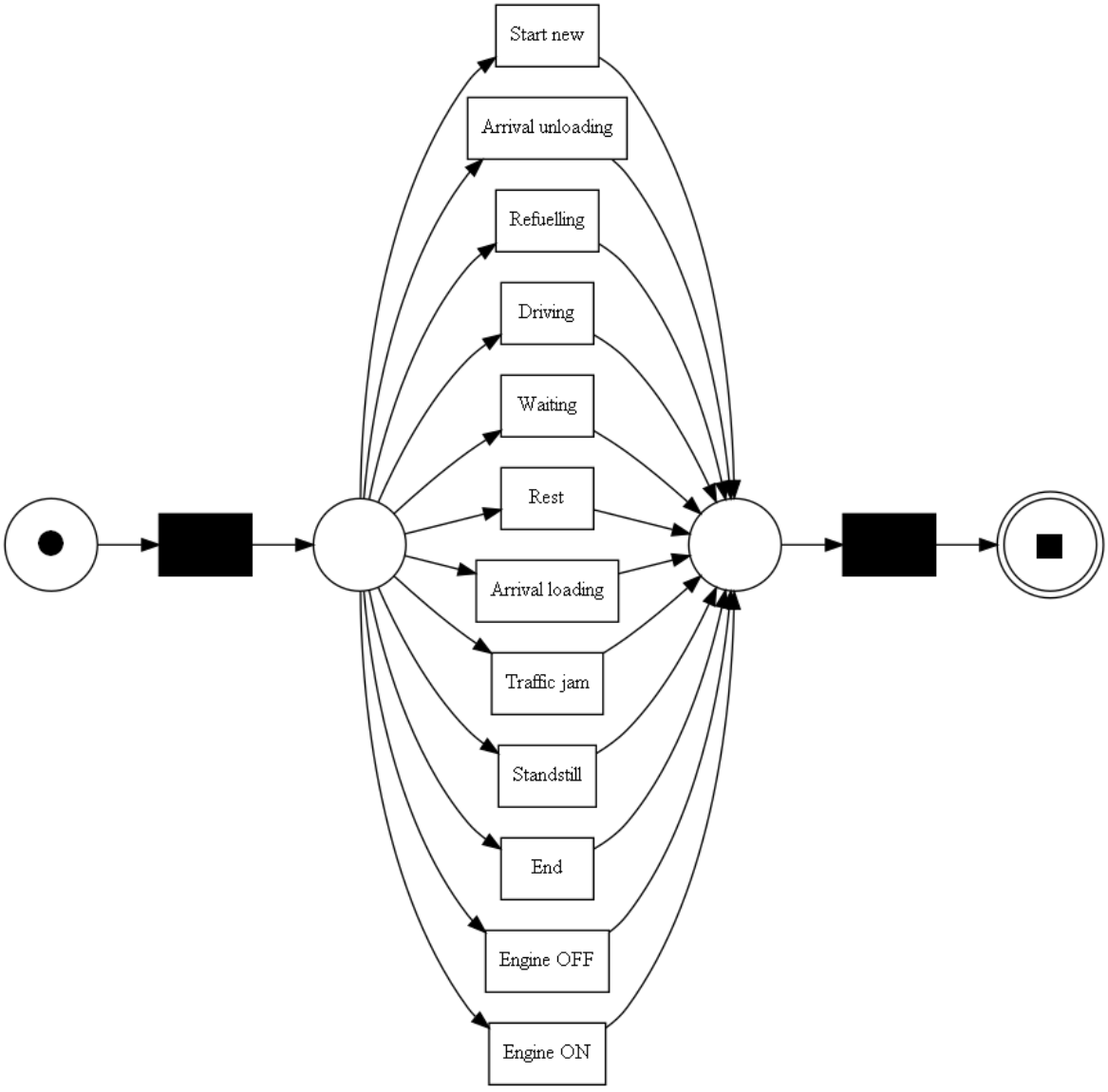

Figure 36: Obtained Petri net for one vehicle id

5.3.2 Process discovery for multiple vehicle id's

```
p = Solver()
p.loadtripseventlog()
p.view_bpm()
```

Figure 37: Code used to generate a Petri net by using multiple vehicle id's

Figure 37 shows the code used to generate a Petri net, which is visualized in **Figure 38**. The code in **Figure 37** calls the function "loadtripseventlog" written in **Figure 35**, which deviates from the function "loadeventlog". The earlier discovered process model in **Figure 36** was too simplistic and insufficient to be a proper process model based on activities. Therefore, the original trip has been split up over 50 vehicle id's to create a better process model reflecting a more accurate situation.

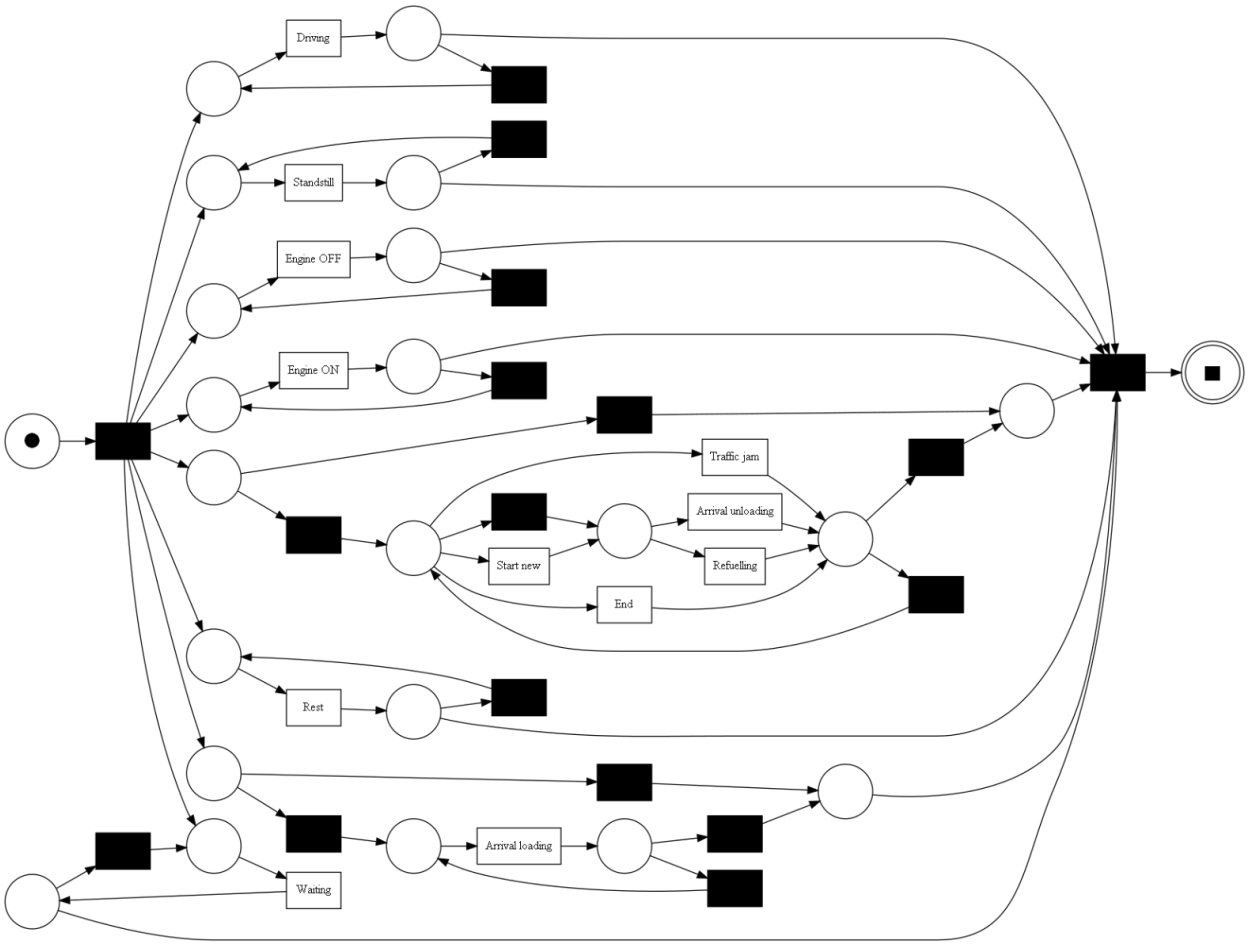

Figure 38: Discovered Petri net for multiple vehicle id's

5.4 Mapping of all events

```
p3 = Solver()
p3.loadeventlog()
p3.loadallevents()
```

```
from keplergl import KeplerGl

df_events = p3.getEvents()
df_latlon = df_events[["lat", "lon"]] # create a dataframe for latitude and longtitude

map= KeplerGl(height=500)
map.add_data(data=df_latlon, name='data_1')
map
```

```
User Guide: https://docs.kepler.gl/docs/keplergl-jupyter
```

Figure 39: Mapping of all events

Figure 39 shows the map containing all events available from the trip conducted by EMONS Group. There can be seen that the trip is made in Germany, Belgium, France, the Netherlands, and the UK. This map is helpful since it shows all places where a truck has been during the trip, whereas the mapping of specific activities only shows the location of these specific activities.

5.5 Mapping and analysis of waiting times

This section is about the mapping and analysis of waiting times. **Subsection 5.5.1** explains how there is created a data frame for waiting times. **Subsection 5.5.2** shows how there has been created the mapping of waiting times. **Subsection 5.5.3** covers how a statistical analysis for waiting times has been done

5.5.1 Creating a data frame for waiting times

Figure 40 shows several steps that have been taken to make a data frame required for the mapping and statistical analysis of the "Waiting" activities. The first step is loading only the waiting activity from a single-vehicle id which is done by "p1.loadevents ". In red, one can see that selecting an activity and vehicle id is possible, which filters away all other activities and vehicle id's. After this, the length of the activity has been calculated. This calculation is done by "p1.calculateStateDuration" and is necessary for the statical analysis. Furthermore, location and day info has been added to give insight into deviations between areas and days for the statistical analysis. Now that the data frame for only the waiting activities has been created, it can be used for mapping and a statistical analysis

```
p1 = Solver()
p1.loadeventlog()
p1.loadevents("Waiting", "09f4b949-0ac1-4580-9628-a076bf858961")
p1.calculateStateDuration()
p1.addLocationInfo()
p1.addDayInfo()
p1.showdataframe()
```

```
avg_waiting_time is 0 days 00:24:34.909090909
```

	row_id	time:timestamp	concept:name	start/end	duration	city	day	day_name
8554	355	2021-10-03 12:05:48	Waiting	end	0.712222		2021-10-03	Sunday
10571	383	2021-10-03 18:22:53	Waiting	end	5.258333		2021-10-03	Sunday
4058	706	2021-10-04 19:53:05	Waiting	end	0.009167		2021-10-04	Monday
7455	709	2021-10-04 20:20:24	Waiting	end	0.450000		2021-10-04	Monday
7260	712	2021-10-04 20:22:24	Waiting	end	0.033333		2021-10-04	Monday
...	...	...	...	...	...	...	...	...
4238	15469	2021-12-18 10:30:23	Waiting	end	0.033333	Folkestone and Hythe	2021-12-18	Saturday
10771	15471	2021-12-18 11:22:06	Waiting	end	0.861944	Folkestone and Hythe	2021-12-18	Saturday
11085	15759	2021-12-20 17:55:24	Waiting	end	0.015278		2021-12-20	Monday
15282	15761	2021-12-20 17:56:24	Waiting	end	0.016667		2021-12-20	Monday
13936	15764	2021-12-20 18:09:05	Waiting	end	0.028056		2021-12-20	Monday

Figure 40: Creation of data frame for waiting times

5.5.2 Mapping of waiting times

The creation of the map can be seen in **Figure 41,** where the pink points indicate places where waiting activities have occurred. The data frame from **Figure 40** and the latitude and longitude were used for this creation. The latitude and longitude have been created in the function "loadevents" from the class "Solver" in **Figure 34**. The mapping of these waiting times gives a good overview of where these are occurring, and the management board can use this information for decision-making in the future.

```
from keplergl import KeplerGl

df_events = p1.getEvents()
df_latlon = df_events[["lat", "lon"]] # create a dataframe for latitude and longitude

map= KeplerGl(height=500)
map.add_data(data=df_latlon, name='data_1')
map
```

```
User Guide: https://docs.kepler.gl/docs/keplergl-jupyter
```

Figure 41: The creation of a map indicating the waiting times

5.5.3 Statistical Analysis of Waiting Times

```
p1.showDurationPlots()
```

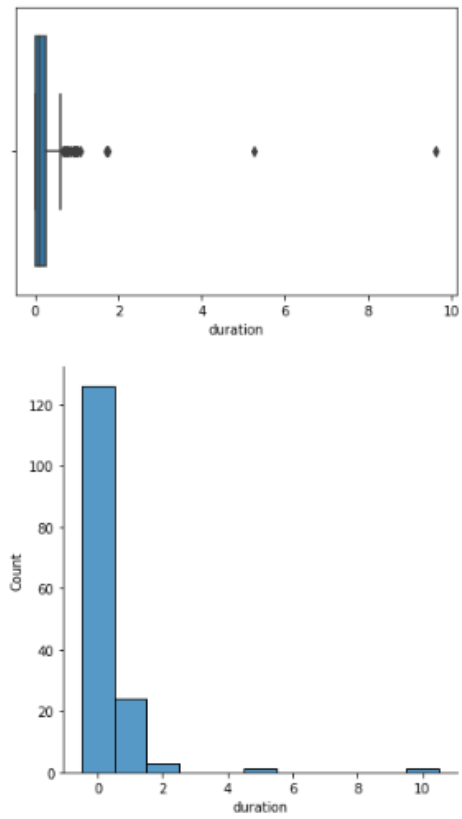

Figure 42: Statistical plot of waiting activities durations

The first statistical tools used are the boxplot and histogram of the duration of the waiting activities shown in **Figure 42**. These tools provide insight into the division of waiting times based on their duration, which can show outliers.

Figure 43 shows the statistical output created from the function "displayDurationStats" defined in the class "Solver" in **Figure 34**. First, there can be seen that the mean, standard variation, minimum, maximum, and division per quartile have been given. After this, a list of waiting times per date and a daily division has been made. This gives a good insight into which days the most waiting times occur. Lastly, a comparison has been made with the location where the most waiting times occur, and cities containing the most waiting times are listed.

```
p1.displayDurationStats()
```

	row_id	duration
count	155.000000	155.000000
mean	10032.212903	0.325910
std	3634.365664	0.920524
min	325.000000	0.000833
25%	9645.000000	0.016944
50%	10699.000000	0.083611
75%	12351.000000	0.270833
max	14158.000000	9.634722

day	max	min	count	median	mean	sum
2021-10-03	5.258333	0.712222	2	2.985278	2.985278	5.970556
2021-10-04	0.455278	0.006944	4	0.025000	0.128056	0.512222
2021-10-12	0.072778	0.061667	2	0.067222	0.067222	0.134444
2021-10-13	0.105556	0.012500	3	0.050000	0.056019	0.168056
2021-10-14	1.696944	1.696944	1	1.696944	1.696944	1.696944
2021-10-15	1.079722	0.537500	2	0.808611	0.808611	1.617222
2021-10-18	1.020000	0.000833	9	0.099167	0.265123	2.386111
2021-11-22	0.275000	0.016667	7	0.050000	0.095040	0.665278
2021-11-24	1.719167	0.001111	13	0.050000	0.238825	3.104722
2021-11-25	9.634722	0.000833	13	0.205556	0.949402	12.342222
2021-11-26	0.595556	0.016111	11	0.033056	0.100783	1.108611
2021-11-29	0.494722	0.016111	7	0.112500	0.169008	1.183056
2021-12-01	1.025556	0.001944	15	0.033611	0.183796	2.756944
2021-12-02	0.949167	0.006111	6	0.158056	0.285694	1.714167
2021-12-04	0.228611	0.228611	1	0.228611	0.228611	0.228611
2021-12-07	0.750000	0.001667	6	0.017361	0.153287	0.919722
2021-12-08	0.914722	0.003333	7	0.133333	0.308730	2.161111
2021-12-09	0.659444	0.083333	4	0.134722	0.253056	1.012222
2021-12-10	0.958333	0.002500	11	0.149444	0.329242	3.621667
2021-12-13	0.001389	0.000833	2	0.001111	0.001111	0.002222
2021-12-17	0.980278	0.001389	12	0.083472	0.220347	2.644167
2021-12-18	1.745278	0.010556	14	0.106250	0.321845	4.505833
2021-12-20	0.028056	0.015278	3	0.016667	0.020000	0.060000

day_name	max	min	count	median	mean	sum
Friday	1.079722	0.001389	36	0.075000	0.249769	8.991667
Monday	1.020000	0.000833	32	0.058333	0.150278	4.808889
Saturday	1.745278	0.010556	15	0.141944	0.315630	4.734444
Sunday	5.258333	0.712222	2	2.985278	2.985278	5.970556
Thursday	9.634722	0.000833	24	0.194444	0.698565	16.765556
Tuesday	0.750000	0.001667	8	0.039861	0.131771	1.054167
Wednesday	1.719167	0.001111	38	0.066806	0.215548	8.190833

city	max	min	count	median	mean	sum
	5.258333	0.000833	31	0.083611	0.472159	14.636944
Ashford	0.533056	0.001111	20	0.050000	0.149083	2.981667
Calais	9.634722	0.001389	21	0.083333	0.589828	12.386389
Coquelles	0.980278	0.001389	26	0.092083	0.188205	4.893333
Folkestone and Hythe	1.745278	0.000833	44	0.108472	0.296862	13.061944
Hann. Münden	1.079722	0.537500	2	0.808611	0.808611	1.617222
Lancaster	0.228611	0.228611	1	0.228611	0.228611	0.228611
Milsbeek	0.455278	0.006944	7	0.016667	0.081746	0.572222
Milton Keynes	0.072778	0.061667	2	0.067222	0.067222	0.134444
Tournai	0.003333	0.003333	1	0.003333	0.003333	0.003333

Figure 43: Statistical output for waiting activities from the duration stats function

5.6 Mapping and analysis of traffic jams

This section is about the mapping and analysis of traffic jams. **Subsection 5.6.1** explains how there is created a data frame for traffic jams. **Subsection 5.6.2** shows how there has been created the mapping of traffic jams. **Subsection 5.6.3** covers how a statistical analysis for traffic jams has been done.

5.6.1 Creating a data frame for traffic jams

```
p2 = Solver()
p2.loadeventlog()
p2.loadevents("Traffic jam", "09f4b949-0ac1-4580-9628-a076bf858961")
p2.calculateStateDuration()
p2.addLocationInfo()
p2.addDayInfo()
p2.showdataframe()
```

```
avg_waiting_time is 0 days 00:09:22.800000
```

	row_id	time:timestamp	concept:name	start/end	duration	city	day	day_name
13926	539	2021-10-04 13:53:28	Traffic jam	end	0.364167		2021-10-04	Monday
11123	543	2021-10-04 13:58:46	Traffic jam	end	0.000556		2021-10-04	Monday
15518	547	2021-10-04 14:05:36	Traffic jam	end	0.000833	Bremen	2021-10-04	Monday
10348	562	2021-10-04 14:35:27	Traffic jam	end	0.043889	Bremen	2021-10-04	Monday
8661	9534	2021-11-19 09:28:26	Traffic jam	end	0.820278	Magdeburg	2021-11-19	Friday
2770	9551	2021-11-19 09:47:25	Traffic jam	end	0.013056		2021-11-19	Friday
6163	9557	2021-11-19 09:53:25	Traffic jam	end	0.039722		2021-11-19	Friday
5252	11316	2021-11-26 15:04:27	Traffic jam	end	0.123611	Calais	2021-11-26	Friday
12497	12310	2021-12-03 09:08:17	Traffic jam	end	0.020278		2021-12-03	Friday
10832	12875	2021-12-06 08:36:26	Traffic jam	end	0.136944	East Staffordshire	2021-12-06	Monday

Figure 44: Creation of data frame for traffic jam activities

Similar to creating a data frame for "Waiting" activities, a data frame for the "Traffic jam" activities has been made using the general object, as seen in **Figure 34**. The only difference is that "p2.loadevents" now asks for "Traffic jam" instead of "Waiting" in **Figure 44**.

5.6.2 Mapping of traffic jams

```
from keplergl import KeplerGl

df_events = p2.getEvents()
df_latlon = df_events[["lat", "lon"]] # create a dataframe for latitude and longitude

map= KeplerGl(height=500)
map.add_data(data=df_latlon, name='data_1')
map
```

```
User Guide: https://docs.kepler.gl/docs/keplergl-jupyter
```

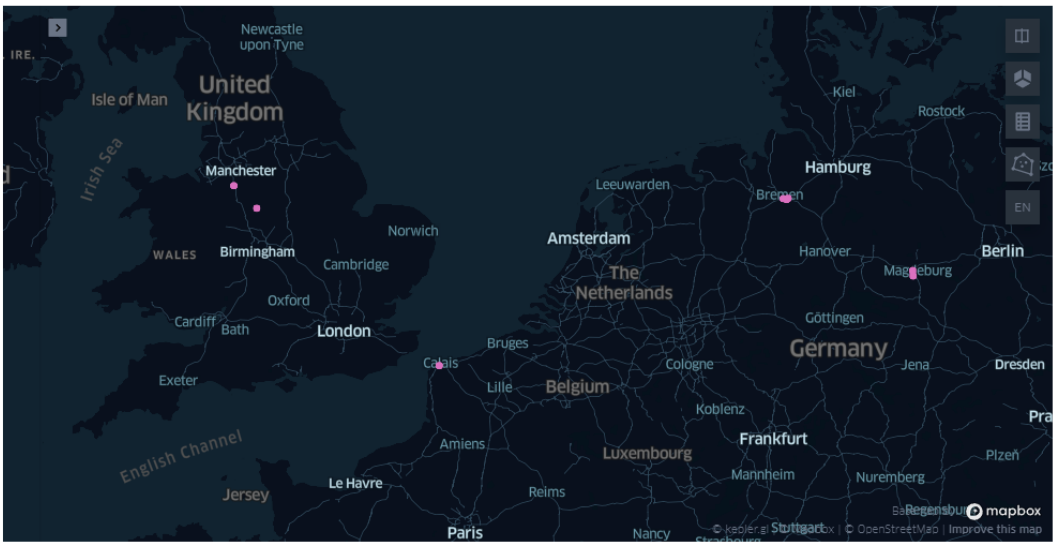

Figure 45: Map of occurred traffic jams

Figure 45 shows the map containing points where traffic jams have occurred. This map is created using the latitude and longitude coming from the data frame in **Figure 44**, created by the function "loadevents" in the class "Solver" from **Figure 34**.

5.6.3 Statistical analysis of traffic jams

```
p2.showDurationPlots()
```

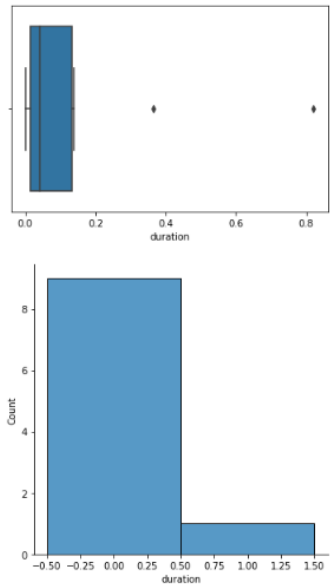

Figure 46: Statistical plot of traffic jam duration

Figure 46 shows a box plot and histogram created for the traffic jam duration. The function "showDurationPlots have again called these."

```
p2.displayDurationStats()
```

	row_id	duration
count	10.000000	10.000000
mean	6071.900000	0.156333
std	4916.005909	0.258025
min	474.000000	0.000556
25%	485.750000	0.014861
50%	8613.500000	0.041806
75%	9850.500000	0.133611
max	11582.000000	0.820278

day	max	min	count	median	mean	sum
2021-10-04	0.364167	0.000556	4	0.022361	0.102361	0.409444
2021-11-19	0.820278	0.013056	3	0.039722	0.291019	0.873056
2021-11-26	0.123611	0.123611	1	0.123611	0.123611	0.123611
2021-12-03	0.020278	0.020278	1	0.020278	0.020278	0.020278
2021-12-06	0.136944	0.136944	1	0.136944	0.136944	0.136944

day_name	max	min	count	median	mean	sum
Friday	0.820278	0.013056	5	0.039722	0.203389	1.016944
Monday	0.364167	0.000556	5	0.043889	0.109278	0.546389

city	max	min	count	median	mean	sum
	0.820278	0.000556	4	0.192222	0.301319	1.205278
Bremen	0.043889	0.000833	2	0.022361	0.022361	0.044722
Calais	0.123611	0.123611	1	0.123611	0.123611	0.123611
East Staffordshire	0.136944	0.136944	1	0.136944	0.136944	0.136944
Hohe Börde	0.039722	0.013056	2	0.026389	0.026389	0.052778

Figure 47: Statistical output for traffic jam from duration stats function

For the traffic jams, statistical output has been created that is visualized in **Figure 47**. This is also done by the "displayDurationStats" function, as shown in **Figure 34**.

5.7 Driving and rest time analysis

This section is about the driving and rest times analysis. **Subsection 5.7.1** discusses the daily driving time analysis during the conducted trip. **Subsection 5.7.2** shows the weekly driving time analysis for the conducted trip.

5.7.1 Daily driving time analysis

```
p4 = Solver()
p4.loadEventlog()
p4.loadEvents("Driving", "09f4b949-0ac1-4580-9628-a076bf858961")
p4.calculateStateDuration()
p4.addLocationInfo()
p4.addDayInfo()
p4.drivingtime()
```

```
avg_waiting_time is 0 days 01:05:47.654596100
```

day	sum	count
2021-10-06	9.988889	7
2021-11-09	19.468889	5
2021-12-04	9.681111	5

Figure 48: Code and result for daily driving hours

Figure 48 shows the results for driving time that exceeds nine hours. According to the regulations of the European Union listed in **Appendix G,** a truck driver cannot drive longer than nine hours a day, except for twice a week, when driving for 10 hours is allowed (European Commission, 2006). This constraint has been written in the "drivingtime" function that can be seen in **Figure 34.** There should be noted that this function assumes that one driver conducted this trip since, in Officedog, there was no provided information about the truck driver.

5.7.2 Weekly driving time analysis

```
p4 = Solver()
p4.loadeventlog()
p4.loadevents("Driving", "09f4b949-0ac1-4580-9628-a076bf858961")
p4.calculateStateDuration()
p4.addLocationInfo()
p4.addDayInfo()
p4.drivingtimeperweek()
```

```
avg_waiting_time is 0 days 01:05:47.654596100
```

week_number_of_year	sum	count
39	9.275000	8
40	34.348333	34
41	35.395833	37
42	36.356944	28
43	35.596667	31
44	33.712222	31
45	41.126389	25
46	38.424722	34
47	30.875278	32
48	31.038889	35
49	27.525000	27
50	34.916944	34
51	5.076667	3

Figure 49: Code and results of weekly driving time analysis

Figure 49 contains the code and results of the weekly driving time analysis that has been performed. The number of hours driven by the truck driver can be seen in the sum column calculated by the function "drivingtimeperweek," which can be seen in **Figure 34**.

5.8 Summary and conclusion

This chapter showed how there had been created standardized process mining applications for Officedog based on a Jupyter Notebook. A standardized process mining application has been created that can be used for mapping & statistical analysis of activities. Also, a standardized process mining application for truck drivers' driving and rest time analysis has been developed. Furthermore, the complete code used in the Jupyter Notebook has been listed in **Appendix I**. Now that the standardized process mining applications have been made, a validation is done with EMONS Group and Bullit Digital.

Chapter 6 – Validation of the research

This chapter provides a validation by EMONS Group and Bullit Digital on this research **Section 6.1** shows the validation of EMONS Group regarding this research. **Section 6.2** contains the validation of Bullit Digital regarding this research.

6.1 EMONS Group

There has been a discussion with EMONS Group about the created Jupyter Notebook and the standardized process mining applications to validate the research. To see EMONS Group's opinion about the standardized process mining applications, a demonstration of how the standardized process mining applications perform is conducted. Furthermore, they provided some feedback in a form that can be seen in **Appendix J**. The first standardized process mining application demonstrated is the mapping of activities. EMONS Group got an insight into the cases where "Waiting" and "Traffic Jam" activities have been mapped and statistically analyzed. They found this visual idea where a logistics company can see how a truck moves based on coordinates quite interesting. The ability to see where bottlenecks like waiting times are occurring on a map is a helpful tool to show customers and decision makers of EMONS Group where things are going wrong. The second standardized process mining application shown was the truck drivers' driving and rest time analysis. EMONS Group mentioned that this type of analysis gives insight into the amount of variance occurring based on the regulations of the European Union. All in all, they found the standardized process mining applications helpful and are satisfied with the abilities of these standardized process mining applications.

6.2 Bullit Digital

There has also been a validation with Bullit Digital to see their opinion about the created Jupyter Notebook and the standardized process mining applications. A demonstration of the standardized process mining applications has been done, including a technical explanation of the code. Similar to the validation done with EMONS Group, a feedback form has been filled in, as can be found in **Appendix K**. Again, the mapping of activities has first been shown. Bullit Digital found this a helpful tool for Officedog since it makes it easier to engage with stakeholders and helps them connect insights from data to the real world. According to them, it is preferred to have a quick visual insight into activities that are occurring, especially in the logistics world. After the demonstration of the mapping of activities, the truck drivers' driving and rest time analysis has been shown to Bullit Digital. Similar to EMONS Group, Bullit Digital found that any logistics company could profit from seeing the amount of variance occurring based on the guidelines provided by the European Union. Lastly, there has been a discussion about how the Jupyter Notebook was designed. Since Bullit Digital provided a template for the Jupyter Notebook, they wanted to see to what aspect this template was changed and if the code was correctly written. All in all, they found the delivered Jupyter Notebook a good product with adequately written code that could be used as a starting point for future implementation in Officedog and were satisfied with the outcome of this research.

Chapter 7 – Conclusion and recommendations

This section concludes the research and gives recommendations for future research. **Section 7.1** covers the conclusion of this research. **Section 7.2** gives recommendations for future research.

7.1 Conclusion

The main research question for this research was, "Which standardized process mining applications can be developed based on the open trip model to support logistics companies by detecting, analyzing, and resolving bottlenecks in their operations?". The thesis showed that several standardized process mining applications are developed for Officedog that can help detect and analyze bottlenecks for logistics companies. However, the research could not develop standardized process mining applications to resolve bottlenecks automatically. The developed standardized process mining applications are focused on giving insight and can be used the eventually resolve bottlenecks by the insight created.

The standardized process mining application of mapping and statistically analyzing activities has been created to satisfy the needs of the main research question. The mapping and statistical analysis of activities provides detection, analysis, and support for resolving bottlenecks discovered in activities. Therefore, it is a helpful tool since the events from Officedog can be filtered on individual activities to create a mapping and statistical analysis of these selected activities. The mapping & analysis can lead to different decision-making from partners of Officedog since they can use the collected information to change, e.g., the route based on historical events that have occurred for the activities if one selects the "Traffic Jam" activity.

Furthermore, there has been developed another standardized process mining application to analyze driving and rest times based on the European Union guidelines for truck drivers. This standardized process mining application can analyze the driving and rest times provided in Officedog to see if truck drivers are sticking to the mandatory driving and rest times. Therefore, this standardized process mining application is a helpful tool that can create less variance in driving and rest time based on these guidelines. Logistics partners of Officedog can profit from this standardized process mining application since it allows them to create better driving and rest times.

According to EMONS Group and Bullit Digital, the standardized process mining applications developed for Officedog are helpful. A good starting point for implementing standardized process mining tools in Officedog has been made, which can be expanded in the future.

7.2 Recommendations

There are some recommendations for future research that can be taken into account. First of all, there could be used more data in future research. The research was based on one vehicle id conducting multiple trips over approximately three months. The data from one vehicle id was assumed to come from multiple vehicle id's to make an advanced process model based on process discovery. The availability of more data containing multiple vehicle id's and trips could have prevented this split up of one vehicle id to create a good process model. Furthermore, if more data is available, there can be a better insight into bottlenecks. More significant differences could be seen in regions because the one-trip analysis only took place in a small part of Europe and the UK. Another recommendation for future research is to consider using the developed standardized process mining application as a web service that partners can use to get insight into their data. Currently, the standardized process mining application are only available in a Jupyter Notebook. However, this Jupyter Notebook could become an interactive tool in Officedog. This interactive tool would enable partners of Officedog to use the tool to eventually resolve bottlenecks based on the created standardized process mining applications. Lastly, the remaining European guidelines for truck drivers' driving and rest times could be implemented since only two of these are currently developed in the Jupyter Notebook.

References

Bullit Digital. (2022). *Officedog*. Officedog.Ai. https://otm-kpi.officedog.ai/admin

Caballero-Hernandez, J. A., Dodero, J. M., Ruiz-Rube, I., Palomo-Duarte, M., Argudo, J. F., & Dominguez-Jimenez, J. J. (2018). Discovering Bottlenecks in a Computer Science Degree through Process Mining techniques. *2018 International Symposium on Computers in Education (SIIE)*. https://doi.org/10.1109/siie.2018.8586680

Cooper, D. R., Schindler, P. S., & Sharma, J. K. (2018). *Business Research Methods, 12/e*. McGraw-Hill Education.

dos Santos Garcia, C., Meincheim, A., Faria Junior, E. R., Dallagassa, M. R., Sato, D. M. V., Carvalho, D. R., Santos, E. A. P., & Scalabrin, E. E. (2019). Process mining techniques and applications – A systematic mapping study. *Expert Systems with Applications*, *133*, 260–295. https://doi.org/10.1016/j.eswa.2019.05.003

draw.io. (2022). *draw.io*. https://draw.io

EMONS Group. (2022). *About EMONS Group*. https://www.emons.eu/about-us-emons-group/

European Commission. (2006). *Driving time and rest periods*. Retrieved June 27, 2022, from https://transport-ec-europa-eu.ezproxy2.utwente.nl/transport-modes/road/social-provisions/driving-time-and-rest-periods_en

Fraunhofer FIT. (2021). *PM4PY*. PM4PY. https://pm4py.fit.fraunhofer.de/

Ghobakhloo, M. (2020). Industry 4.0, digitization, and opportunities for sustainability. *Journal of Cleaner Production*, *252*, 119869. https://doi.org/10.1016/j.jclepro.2019.119869

Heerkens, H., & van Winden, A. (2021). *Solving Managerial Problems Systematically* (1st ed.). Wolters-Noordhoff.

Kepler. (2022). *Kepler*. Keplerg.Gl. https://kepler.gl/

Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering – A systematic literature review. *Information and Software Technology*, *51*(1), 7–15. https://doi.org/10.1016/j.infsof.2008.09.009

Leemans, M., van der Aalst, W. M. P., & van den Brand, M. G. J. (2018). Hierarchical performance analysis for process mining. *Proceedings of the 2018 International Conference on Software and System Process*. https://doi.org/10.1145/3202710.3203151

Open Trip Model. (2022). *The OTM5 Model*. Retrieved May 25, 2022, from https://www.opentripmodel.org/page/about

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45–77. https://doi.org/10.2753/mis0742-1222240302

Process Mining. (2022). Process Mining. Retrieved June 30, 2022, from http://processmining.org/home.html

Recker, J. (2010). Opportunities and constraints: the current struggle with BPMN. *Business Process Management Journal*, *16*(1), 181–201. https://doi.org/10.1108/14637151011018001

Royakkers, L., & van de Poel, I. (2011). *Ethics, Technology, and Engineering: An Introduction* (1st ed.). Wiley-Blackwell.

Rozinat, A., & van der Aalst, W. (2008). Conformance checking of processes based on monitoring real behavior. *Information Systems*, *33*(1), 64–95. https://doi.org/10.1016/j.is.2007.07.001

Seaborn. (2022). Seaborn. https://seaborn.pydata.org/

Toosinezhad, Z., Fahland, D., Koroglu, O., & van der Aalst, W. M. (2020). Detecting System-Level Behavior Leading To Dynamic Bottlenecks. *2020 2nd International Conference on Process Mining (ICPM)*. https://doi.org/10.1109/icpm49681.2020.00014

van der Aalst, W. (2012). Process Mining. *ACM Transactions on Management Information Systems*, *3*(2), 1–17. https://doi.org/10.1145/2229156.2229157

van der Aalst, W. M. (2014). Process Mining in the Large: A Tutorial. *Business Intelligence*, 33–76. https://doi.org/10.1007/978-3-319-05461-2_2

van der Aalst, W. M. P. (2011). *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer.

Weske, M. (2020). *Business Process Management: Concepts, Languages, Architectures* (3rd ed. 2019 ed.). Springer.

Yasmin, F. A., Bukhsh, F. A., & de Alencar Silva, P. (2018). Process enhancement in process mining: A literature review. *CEUR Workshop Proceedings*. https://ris.utwente.nl/ws/portalfiles/portal/84624357/proces_enhancement.pdf

Appendix A

officedog.ai Jupyter Template

This template is intended to provide the basic functionality for connecting to the officedog.ai database and extract its entities. This serves as a basis for further programming using Jupyter / Python to explore AI and ML workloads.

To this end, the template defines a few basic definitions of entities in the officedog.ai database, as well as example queries using the `SQLAlchemy` library. These can easily be extended with the required fields appropriate for the workload. An example of a relationship is also given by the `ActorVehicle` class. This models the custom many-to-many (in practice: many-to-one <--> one-to-many) relationship between `Actors` and `Vehicles`.

Please refer to the officedog.ai docs for information about the available entities and fields. Credentials for connecting to the database can be requested via your environment administrator.

Install and load dependencies

```
!pip install pydantic==1.8.2
!pip install psycopg2-binary==2.9.1
!pip install SQLAlchemy==1.4.26
```

```
...
```

```
from sqlalchemy import create_engine, Table
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, Session, relation
from pydantic import BaseModel
from pydantic.types import UUID4
from sqlalchemy import Boolean, Column, ForeignKey, Integer, String
from sqlalchemy.orm import relationship
from typing import List, Optional
import pandas as pd
```

Configure DB connection

```
DB_URL = "postgresql://otm-kpi-read@officedog-db-1:hftjhKqkdqugrEZHRzVad8PZ@officedog-db-1.postgres.database.azure.com:5432/otm-k
```

```
# Import data types
from sqlalchemy.dialects.postgresql import JSONB, ARRAY, ENUM

engine = create_engine(
    DB_URL
)
SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)

Base = declarative_base()
```

```
# Define models
class Vehicle(Base):

    __tablename__ = "vehicles"

    id = Column(String, primary_key=True, index=True)
    name = Column(String)
    externalAttributes = Column(JSONB)
    fuel = Column(String)
    licensePlate = Column(String)
    emptyWeight = Column(JSONB)

    # Relations
    actors = relationship("ActorVehicle", back_populates="vehicle")

class Actor(Base):

    __tablename__ = "actors"

    id = Column(String, primary_key=True, index=True)
    name = Column(String)

    # Relations
    vehicles = relationship("ActorVehicle", back_populates="actor")

class ActorVehicle(Base):

    __tablename__ = "actor_vehicle_associations"

    actorId = Column(String, ForeignKey("actors.id"), primary_key=True, index=True)
    actor = relationship("Actor", back_populates="vehicles")
    vehicleId = Column(String, ForeignKey("vehicles.id"), primary_key=True, index=True)
    vehicle = relationship("Vehicle", back_populates="actors")
    roles = Column(ARRAY(ENUM('shipper', 'carrier', 'consignee', 'consignor', 'receiver', name='ActorRoles')))
    description = Column(String)


class Consignment(Base):

    __tablename__ = "consignments"

    id = Column(String, primary_key=True, index=True)
    name = Column(String)

class TransportOrder(Base):

    __tablename__ = "transport_orders"

    id = Column(String, primary_key=True, index=True)
    name = Column(String)

# Instantiate DB
Base.metadata.create_all(bind=engine)

db = SessionLocal()
```

Figure 50: Jupyter Notebook template provided by Bullit Digital

Appendix B

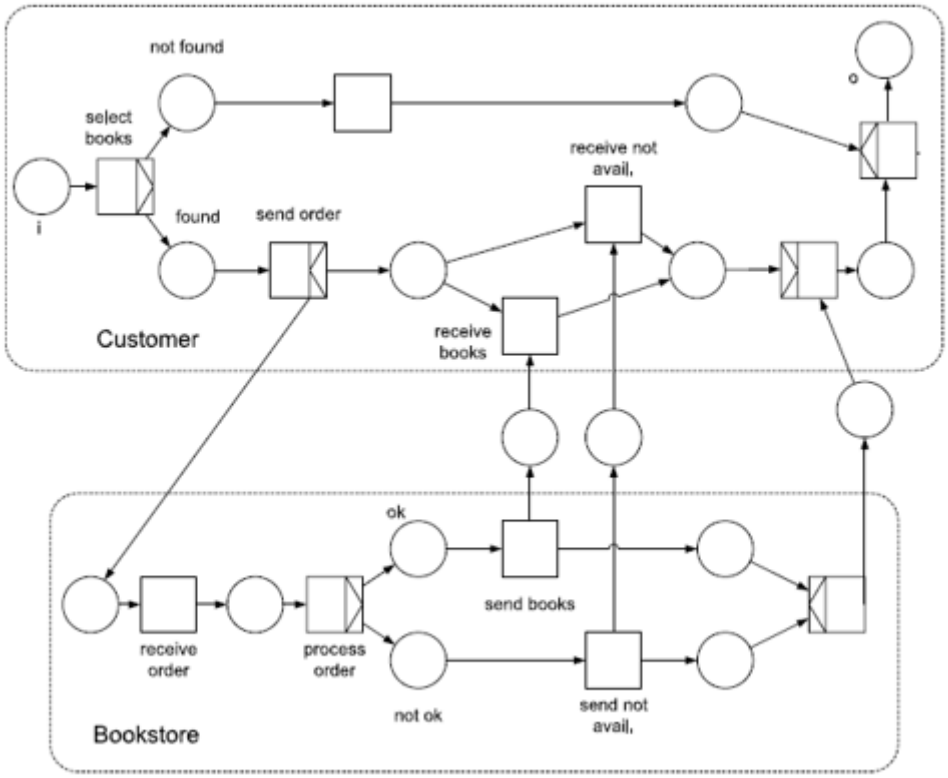

Figure 51: Example of a Workflow net containing multiple parties (Weske, 2020)

Appendix C

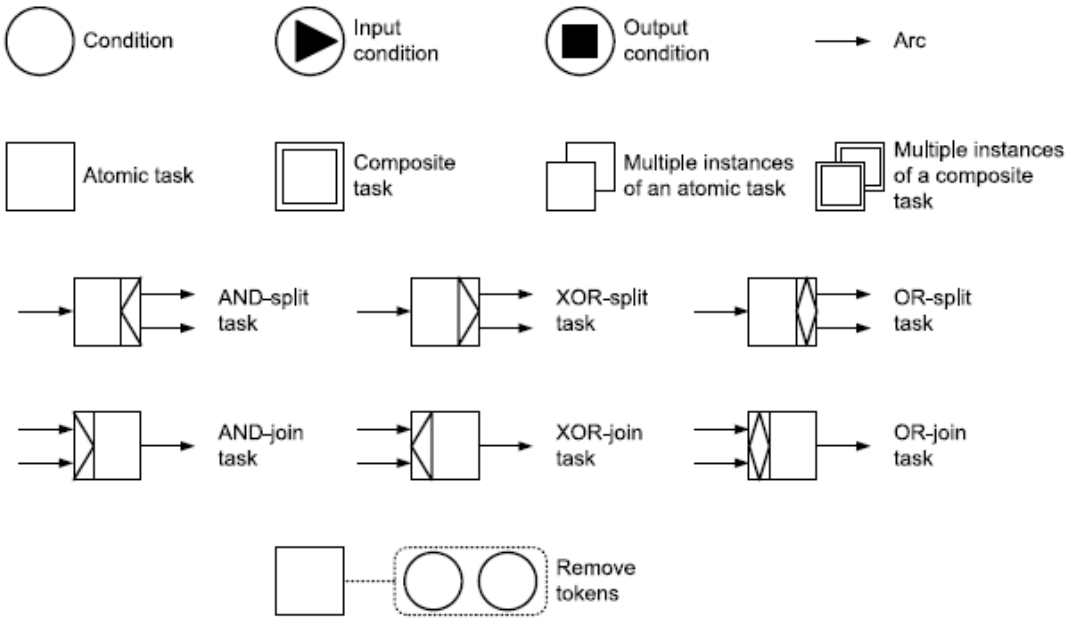

Figure 52: Notational elements of YAWL (Weske, 2020)

Appendix D

Frequency	Reference	Trace
455	σ_1	$\langle a, c, d, e, h \rangle$
191	σ_2	$\langle a, b, d, e, g \rangle$
177	σ_3	$\langle a, d, c, e, h \rangle$
144	σ_4	$\langle a, b, d, e, h \rangle$
111	σ_5	$\langle a, c, d, e, g \rangle$
82	σ_6	$\langle a, d, c, e, g \rangle$
56	σ_7	$\langle a, d, b, e, h \rangle$
47	σ_8	$\langle a, c, d, e, f, d, b, e, h \rangle$
38	σ_9	$\langle a, d, b, e, g \rangle$
33	σ_{10}	$\langle a, c, d, e, f, b, d, e, h \rangle$
14	σ_{11}	$\langle a, c, d, e, f, b, d, e, g \rangle$
11	σ_{12}	$\langle a, c, d, e, f, d, b, e, g \rangle$
9	σ_{13}	$\langle a, d, c, e, f, c, d, e, h \rangle$
8	σ_{14}	$\langle a, d, c, e, f, d, b, e, h \rangle$
5	σ_{15}	$\langle a, d, c, e, f, b, d, e, g \rangle$
3	σ_{16}	$\langle a, c, d, e, f, b, d, e, f, d, b, e, g \rangle$
2	σ_{17}	$\langle a, d, c, e, f, d, b, e, g \rangle$
2	σ_{18}	$\langle a, d, c, e, f, b, d, e, f, b, d, e, g \rangle$
1	σ_{19}	$\langle a, d, c, e, f, d, b, e, f, b, d, e, h \rangle$
1	σ_{20}	$\langle a, d, b, e, f, b, d, e, f, d, b, e, g \rangle$
1	σ_{21}	$\langle a, d, c, e, f, d, b, e, f, c, d, e, f, d, b, e, g \rangle$

Figure 53: Event log containing traces (Van der Aalst, 2011)

Appendix E

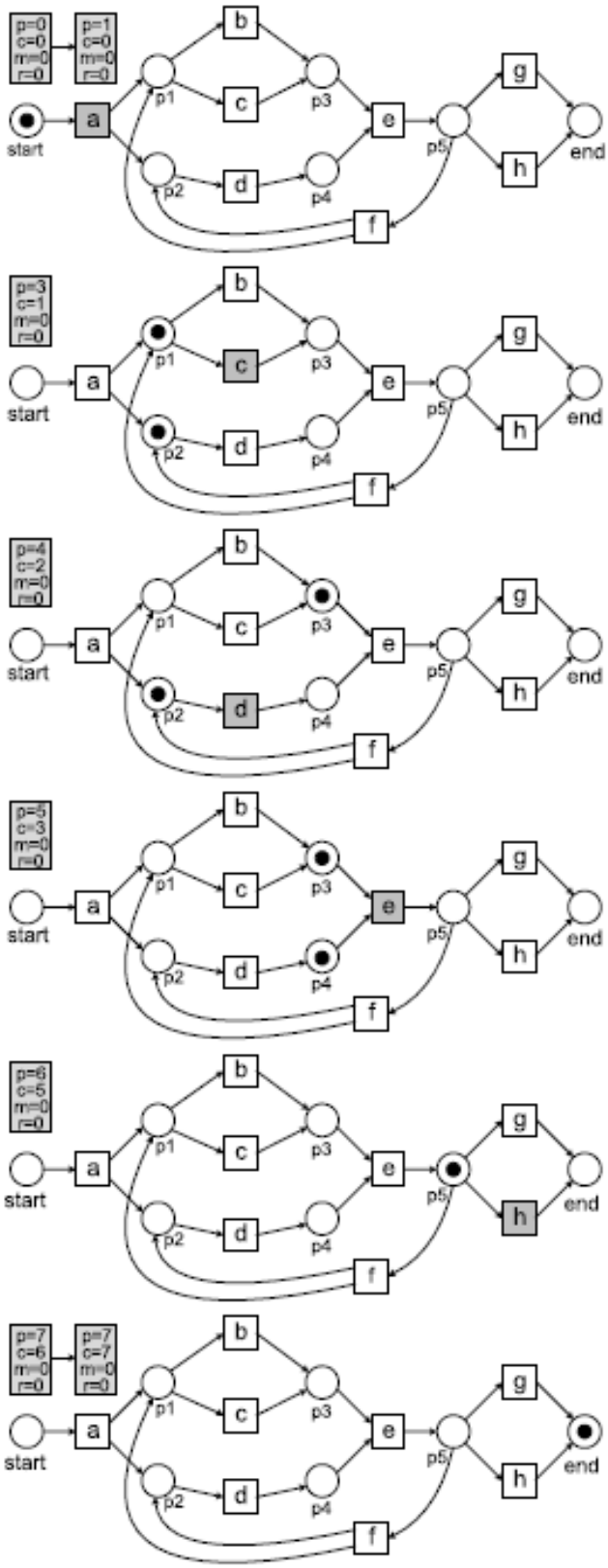

Figure 54: Replay of N_1 (Van der Aalst, 2011)

Appendix F

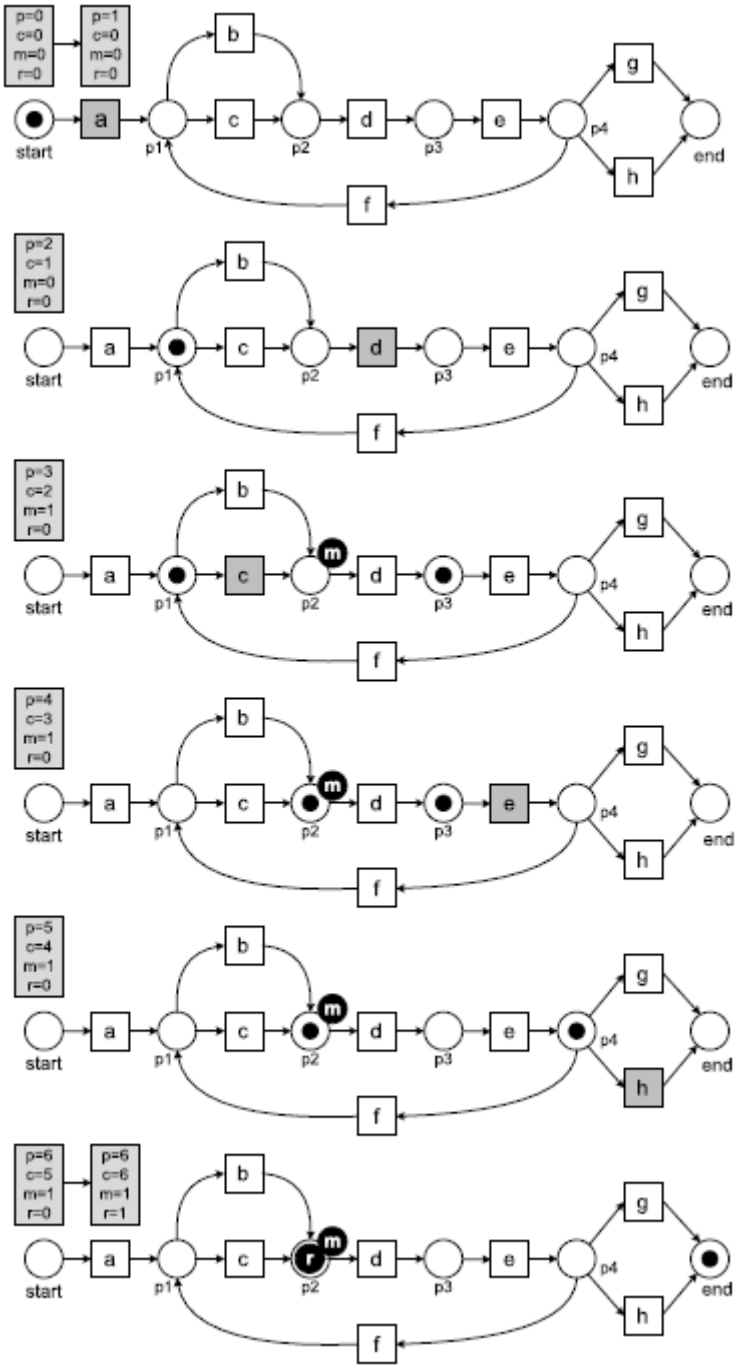

Figure 55: Replay of N_2 (Van der Aalst, 2011)

Appendix G

Regulation (EC) No 561/2006 EN provides a common set of EU rules for maximum daily and fortnightly driving times, as well as daily and weekly minimum rest periods for all drivers of road haulage and passenger transport vehicles, subject to specified exceptions and national derogations. The scope of operations regulated is tremendously diverse, it includes: passenger transport and road haulage operations, both international and national, long and short distance, drivers for own account and for hire and reward, employees and self-employed.

The aim of this set of rules is to avoid distortion of competition, improve road safety and ensure drivers' good working conditions within the European Union.

These rules establish notably that:

- Daily driving period shall not exceed 9 hours, with an exemption of twice a week when it can be extended to 10 hours.
- Total weekly driving time may not exceed 56 hours and the total fortnightly driving time may not exceed 90 hours.
- Daily rest period shall be at least 11 hours, with an exception of going down to 9 hours maximum three times a week. Daily rest can be split into 3 hours rest followed by 9 hour rest to make a total of 12 hours daily rest.
- Breaks of at least 45 minutes (separable into 15 minutes followed by 30 minutes) should be taken after 4 ½ hours at the latest.
- Weekly rest is 45 continuous hours, which can be reduced every second week to 24 hours. Compensation arrangements apply for reduced weekly rest period. Weekly rest is to be taken after six days of working, except for coach drivers engaged in a single occasional service of international transport of passengers who may postpone their weekly rest period after 12 days in order to facilitate coach holidays.
- Daily and/or weekly driving times may be exceeded in exceptional circumstances by up to one hour to enable the driver to reach his/her place of residence or the employer's operational centre in order to take a weekly rest period. Exceeding the daily and/or weekly driving times by up to two hours is also allowed to enable the driver to reach his/her place of residence or the employer's operational centre in order to take a regular weekly rest period.

The compliance with these provisions is subject to continuous monitoring and controls, which are carried out on national and international level via checking tachograph records at the road side and at the premises of undertakings.

Figure 56: Driving time and rest periods (European Commission, 2006)

Appendix H

jupyter

Quit Logout

0 / Downloads / PM4PY / examples

	Name	Last Modified	File size
..		een paar seconden geleden	
powerbi		11 dagen geleden	
activity_position.py		6 maanden geleden	508 B
align_approx_pt.py		6 maanden geleden	909 B
align_decomposition_ex_paper.py		6 maanden geleden	3.15 kB
align_decomposition_example.py		6 maanden geleden	1.19 kB
alignment_discounted_a_star.py		6 maanden geleden	2.07 kB
alignment_test.py		6 maanden geleden	2.08 kB
alpha_miner.py		6 maanden geleden	718 B
antialignments_and_precision.py		6 maanden geleden	1.02 kB
backwards_token_replay.py		6 maanden geleden	597 B
batch_detection.py		6 maanden geleden	579 B
bpmn_from_pt_conversion.py		6 maanden geleden	864 B
bpmn_import_and_to_petri_net.py		6 maanden geleden	808 B
case_overlap_stat.py		6 maanden geleden	488 B
corr_mining.py		6 maanden geleden	1.89 kB
cycle_time.py		6 maanden geleden	502 B
data_petri_nets.py		6 maanden geleden	1.71 kB
dataframe_prefix_and_fea_extraction.py		6 maanden geleden	826 B
dec_treplay_imdf.py		6 maanden geleden	1.25 kB
decisiontree_align_example.py		6 maanden geleden	1.02 kB
decisiontree_trivial_example.py		6 maanden geleden	1.82 kB
dfg_align_log.py		6 maanden geleden	2.13 kB
dfg_filt_act_paths_perc.py		6 maanden geleden	1.18 kB
dfg_min_ex_log.py		6 maanden geleden	2.62 kB
dfg_min_ex_pandas.py		6 maanden geleden	2.94 kB
dfg_playout.py		6 maanden geleden	1.76 kB
diagn_add_dataframe.py		6 maanden geleden	1.22 kB
discovery_data_petri_net.py		6 maanden geleden	500 B
dotted_chart.py		6 maanden geleden	594 B
emd_evaluation.py		6 maanden geleden	3.65 kB
enrich_log_with_align.py		6 maanden geleden	671 B
events_distribution.py		6 maanden geleden	2.62 kB
example_check_fitness.py		6 maanden geleden	891 B
execute_everything.py		3 maanden geleden	17.8 kB
extended_marking_equation.py		6 maanden geleden	1.69 kB
feature_extraction.py		6 maanden geleden	583 B
feature_extraction_ocel.py		5 maanden geleden	984 B
features_locally_linear_embedding.py		6 maanden geleden	719 B
footprints_petri_net.py		6 maanden geleden	2.39 kB
footprints_tree_conf.py		6 maanden geleden	3.42 kB
graphs_visualization.py		6 maanden geleden	385 B
heu_miner_test.py		6 maanden geleden	1.05 kB
heuminer_plusplus.py		6 maanden geleden	1.17 kB
im_example.py		6 maanden geleden	2.1 kB

inductive_miner.py	6 maanden geleden	1.9 kB
inhibitor_reset_arcs.py	2 maanden geleden	2.6 kB
interval_events_overlap.py	6 maanden geleden	680 B
link_analysis_vbfa.py	6 maanden geleden	1.43 kB
log_skeleton.py	6 maanden geleden	782 B
log_to_int_tree_open_paths.py	6 maanden geleden	588 B
logs_alignment.py	6 maanden geleden	2.1 kB
logs_petri_visual_comparison.py	6 maanden geleden	1.03 kB
manual_log_generation.py	6 maanden geleden	547 B
memory_profilation_alignments.py	6 maanden geleden	1.34 kB
memory_profilation_iterparse.py	6 maanden geleden	2.96 kB
merging_case_relations.py	6 maanden geleden	827 B
montecarlo_dfg.py	6 maanden geleden	1.86 kB
montecarlo_petri_net.py	6 maanden geleden	1.53 kB
multialignments.py	6 maanden geleden	918 B
network_analysis.py	6 maanden geleden	2.29 kB
object_centric_petri_net_discovery.py	6 maanden geleden	323 B
ocdfg_discovery.py	6 maanden geleden	503 B
ocel_streaming.py	6 maanden geleden	1.94 kB
ocel_validation.py	5 maanden geleden	730 B
orgmining_local_diagn.py	6 maanden geleden	1.89 kB
perf_spectrum_visualization.py	6 maanden geleden	753 B
performance_dfg_simulation.py	6 maanden geleden	610 B
petri_manual_generation.py	6 maanden geleden	1.94 kB
pn_to_pt.py	6 maanden geleden	2.58 kB
process_tree_frequency_annotation.py	5 maanden geleden	723 B
process_tree_reduction.py	6 maanden geleden	807 B
read_write_ocel.py	6 maanden geleden	1.95 kB
resource_profiles_log.py	6 maanden geleden	3.08 kB
resource_profiles_pandas.py	6 maanden geleden	3.13 kB
rework.py	6 maanden geleden	304 B
roles_detection.py	6 maanden geleden	513 B
simplified_interface.py	6 maanden geleden	9.92 kB
sna_log.py	6 maanden geleden	1.49 kB
streaming_conformance_footprints.py	6 maanden geleden	1.95 kB
streaming_conformance_tbr.py	6 maanden geleden	1.74 kB
streaming_conformance_temporal_profile.py	6 maanden geleden	806 B
streaming_csv_reader_event_stream.py	6 maanden geleden	686 B
streaming_discovery_dfg.py	6 maanden geleden	1.75 kB
streaming_xes_reader_event_stream.py	6 maanden geleden	789 B
streaming_xes_reader_trace_stream.py	6 maanden geleden	789 B
temporal_profile_dataframe.py	6 maanden geleden	733 B
temporal_profile_log.py	6 maanden geleden	583 B
test_evaluation.py	6 maanden geleden	561 B
timestamp_interleavings.py	6 maanden geleden	1.8 kB
token_replay_alpha.py	6 maanden geleden	1.22 kB
token_replay_imdf.py	6 maanden geleden	1.25 kB
trans_system_stochastic_vis.py	6 maanden geleden	682 B
transient_dfg.py	6 maanden geleden	989 B
transient_petri_net.py	6 maanden geleden	1.16 kB
transition_system_test.py	6 maanden geleden	1.02 kB
tree_manual_generation.py	5 maanden geleden	1.34 kB
tree_playout.py	6 maanden geleden	793 B
trie.py	6 maanden geleden	437 B
visualization_align_table.py	6 maanden geleden	729 B
visualization_processtree.py	6 maanden geleden	759 B
visualization_serialization.py	6 maanden geleden	2.3 kB
woflan.py	6 maanden geleden	592 B
workalendar_example.py	5 maanden geleden	736 B

Figure 57: Example notebooks created in the PM4PY library that can be used for process mining

Appendix I

officedog-typografie-naast.png

Install and load dependencies

```
!pip install pydantic==1.8.2
!pip install psycopg2-binary==2.9.1
!pip install SQLAlchemy==1.4.26
!pip install pm4py

!pip install wheel
!pip install pipwin

!pipwin install numpy
!pipwin install pandas
!pipwin install shapely
!pipwin install gdal
!pipwin install fiona
!pipwin install pyproj
!pipwin install six
!pipwin install rtree
!pipwin install geopandas
!pip install matplotlib
!pip install keplergl
!pip install geopy

from sqlalchemy import create_engine, Table
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, Session, relation
from pydantic import BaseModel
from pydantic.types import UUID4
from sqlalchemy import Boolean, Column, ForeignKey, Integer, String
from sqlalchemy.orm import relationship
from typing import List, Optional
from geopy.geocoders import Nominatim
from datetime import datetime

import pandas as pd
import numpy  as np
import seaborn as sns
import pm4py
import keplergl
import geopandas as gpd
import matplotlib
import matplotlib.pyplot as plt
import graphviz
import os
import random
os.environ["PATH"] += os.pathsep + 'C:/Program Files/Graphviz/bin/'
```

Configure DB connection

```
DB_URL = "postgresql://otm-kpi-read@officedog-db-1:hftjhKqkdqugrEZHRzVad8PZ@officedog-db-1.postgres.database.azure.com:5432/otm-kpi?sslmode=require" #@param {type:"string"}
```

Define and instantiate DB models

Clues are taken from this tutorial - great to 'learn-by-example' https://fastapi.tiangolo.com/tutorial/sql-databases/

```
# Import data types
from sqlalchemy.dialects.postgresql import JSONB, ARRAY, ENUM, TIMESTAMP

engine = create_engine(DB_URL)
SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)

Base = declarative_base()

# Define models
class Events(Base):
    __tablename__ = "events"
```

```
    id = Column(String, primary_key=True, index=True)
    name = Column(String)
    eventType = Column(String)
    creationDate = Column()
    vehicleId = Column(String, ForeignKey("vehicles.id"), primary_key=True, index=True)
    geoReference = Column(String)

class Vehicle(Base):
    __tablename__ = "vehicles"

    id = Column(String, primary_key=True, index=True)
    name = Column(String)
    externalAttributes = Column(JSONB)
    fuel = Column(String)
    licensePlate = Column(String)
    emptyWeight = Column(JSONB)

    # Relations
    actors = relationship("ActorVehicle", back_populates="vehicle")

class Actor(Base):
    __tablename__ = "actors"

    id = Column(String, primary_key=True, index=True)
    name = Column(String)

    # Relations
    vehicles = relationship("ActorVehicle", back_populates="actor")

class ActorVehicle(Base):
    __tablename__ = "actor_vehicle_associations"

    actorId = Column(String, ForeignKey("actors.id"), primary_key=True, index=True)
    actor = relationship("Actor", back_populates="vehicles")
    vehicleId = Column(String, ForeignKey("vehicles.id"), primary_key=True, index=True)
    vehicle = relationship("Vehicle", back_populates="actors")
    roles = Column(ARRAY(ENUM('shipper', 'carrier', 'consignee', 'consignor', 'receiver', name='ActorRoles')))
    description = Column(String)

class Consignment(Base):
    __tablename__ = "consignments"

    id = Column(String, primary_key=True, index=True)
    name = Column(String)

class TransportOrder(Base):
    __tablename__ = "transport_orders"

    id = Column(String, primary_key=True, index=True)
    name = Column(String)

# Instantiate DB
Base.metadata.create_all(bind=engine)

db = SessionLocal()
```

Generic solution

```
def select_location(row):       # function to return location based on latitude and longitude
    geolocator = Nominatim(user_agent="geoapiExercises")
    city = 'unknown'
    latitude  = row['lat']
    longitude  = row['lon']
    location = geolocator.reverse(str(round(latitude,2))+","+str(round(longitude,2)))
    address = location.raw['address']
    city = address.get('city', '')
    return city

def get_day_name(row):           # function to get the day name
    naame = 'unknown'
    date  = row['day']
    day_name= ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday','Sunday']
    day = datetime.strptime(str(date), '%Y-%m-%d').weekday()
    return day_name[day]

class Solver:
  def __init__(self):
    self.event = ""
    self.vehicle_id = ""
    self.eventlog = pd.DataFrame()
    self.events = pd.DataFrame()
    self.events_red = pd.DataFrame()
```

```
    self.random = ""

  def loadtripseventlog(self):
    df = pd.read_sql(db.query(Events).statement, db.bind)    # create data frame
    df['RAND'] = [ random.randint(1,50)  for k in df.index]  # add a number 1 to 50 to each event as a new vehicle id
    df.columns = ['case_id', 'activity', 'event', 'timestamp','vehicleId', 'geoReference','RAND'] # identify columns
    df = pm4py.format_dataframe(df, case_id='RAND',activity_key='activity',
                            timestamp_key='timestamp')      # format based on the PM4PY requirements
    del df['case_id']                                       # remove useless columns
    del df['activity']
    del df['timestamp']
    self.eventlog = df

  def loadeventlog(self):
    df = pd.read_sql(db.query(Events).statement, db.bind)    # create data frame
    df.columns = ['case_id', 'activity', 'event', 'timestamp','vehicleId', 'geoReference'] # identify columns
    df = pm4py.format_dataframe(df, case_id='case_id',activity_key='activity',
                            timestamp_key='timestamp')      # format based on the PM4PY requirements
    del df['case_id']                                       # remove useless columns
    del df['activity']
    del df['timestamp']
    self.eventlog = df

  def loadallevents(self):
    df_events = self.eventlog.sort_values(by='time:timestamp')
    df_geo = dict(df_events.geoReference)                   # add latitude and longitude as column
    df2 = pd.DataFrame.from_dict(df_geo).T
    df2 = df2[["lat", "lon"]]

    df_events = pd.concat([df_events, df2], axis=1, join='inner') # add this to the case id in the events dataframe
    df_events = df_events.loc[:,~df_events.columns.duplicated()].copy()

    df_events['lat'] = pd.to_numeric(df_events['lat'])      # format type from string to numeric
    df_events['lon'] = pd.to_numeric(df_events['lon'])      # format type from string to numeric

    self.events = df_events

  def loadevents(self, event, vehicle_id):
    self.event = event
    self.vehicle_id = vehicle_id
    df_events = self.eventlog.sort_values(by='time:timestamp')    # sort on timestamp
    df_events['row_id'] = df_events.reset_index().index
    df_events = df_events.loc[df_events['concept:name'] == event] # only the waiting activities

    df_geo = dict(df_events.geoReference)                         # add latitdue and longtidue as column
    df2 = pd.DataFrame.from_dict(df_geo).T
    df2 = df2[["lat", "lon"]]

    df_events = pd.concat([df_events, df2], axis=1, join='inner') # add this to the case id in the events dataframe
    df_events = df_events.loc[:,~df_events.columns.duplicated()].copy() # remove duplicates

    df_events['lat'] = pd.to_numeric(df_events['lat'])            # format type from string to numeric
    df_events['lon'] = pd.to_numeric(df_events['lon'])            # format type from string to numeric

    self.events = df_events

  def view_bpm(self):
    tree = pm4py.discover_process_tree_inductive(self.eventlog)   # create a process model with PM4PY
    net, initial_marking, final_marking = pm4py.convert_to_petri_net(tree) # convert to Petri net
    pm4py.view_petri_net(net, initial_marking, final_marking)

  def calculateStateDuration(self):
    df_events = self.events
    df_events.loc[df_events['row_id']- df_events['row_id'].shift(1) != 1, 'start/end'] = 'start' # indicate start
    df_events.loc[df_events['row_id'].shift(-1)- df_events['row_id'] != 1, 'start/end'] = 'end'  # indicate end
    df_events_red = df_events[['row_id', 'time:timestamp','concept:name','start/end', 'lat', 'lon']].dropna()# select
columns
    startend = df_events_red['time:timestamp'].diff()                                           # calucate activity
duration
    df_events_red = pd.concat([df_events_red, startend], axis=1)                                # add this to the da
taframe
    df_events_red.columns = ['row_id', 'time:timestamp', 'concept:name', 'start/end','lat', 'lon', 'duration']
    df_events_red = df_events_red[~df_events_red['start/end'].isin(['start'])]                  # only show end even
ts

    avg_waiting_time = df_events_red["duration"].mean()            # calculate average waiting time
    print("avg_waiting_time is " + str(avg_waiting_time))

    df_events_red['duration'] = df_events_red.duration.apply(lambda x: x.seconds) # calculate duration of activity
    df_events_red['duration'] = df_events_red['duration'] / 3600                  # convert to hours

    self.events_red = df_events_red
```

```
  def addLocationInfo(self):
    self.events_red['city'] = self.events_red.apply(lambda x: select_location(x), axis =1) # add location
    self.events_red.drop(['lat', 'lon'], axis=1, inplace=True)

  def addDayInfo(self):
    self.events_red['day'] = self.events_red['time:timestamp'].dt.date                    # add day info
    self.events_red['day_name'] = self.events_red.apply(lambda x: get_day_name(x), axis =1)

  def showdataframe(self):
    display (self.events_red)

  def drivingtime(self):
    self.events_red = self.events_red.groupby('day')['duration'].agg(['sum','count']) # calculate driving time per da
y
    self.events_red = self.events_red.loc[self.events_red['sum'] > 9]             # filter out driving times higher t
han 9 hours

    display (self.events_red)

  def drivingtimeperweek(self):
    self.events_red ['day'] = pd.to_datetime(self.events_red['day'], errors='coerce') # change to datetime
    self.events_red ['week_number_of_year'] = self.events_red['day'].dt.week       # add week number

    # Calculate weekly driving time
    self.events_red = self.events_red.groupby('week_number_of_year')['duration'].agg(['sum','count'])

    display (self.events_red)

  def getEventLog(self):
    return self.eventlog

  def getEvents(self):
    return self.events

  def getEvents_red(self):
    return self.events_red

  def displayEvents(self):
    display (self.events)

  def displayEvents_red(self):
    display (self.events_red)

  def showDurationPlots(self):
    sns.boxplot(data=self.events_red, x=self.events_red.duration) # duration in hours
    sns.displot(self.events_red, x="duration", binwidth=0.001, discrete=True)

  def displayDurationStats(self):
    display(self.events_red.describe())
    display(self.events_red.groupby('day').duration.agg(['max', 'min', 'count', 'median', 'mean', 'sum']))
    display(self.events_red.groupby('day_name').duration.agg(['max', 'min', 'count', 'median', 'mean', 'sum']))
    display(self.events_red.groupby('city').duration.agg(['max', 'min', 'count', 'median', 'mean', 'sum']))
```

Process Discovery

This standardized process mining application creates a process model based on vehicle id. The PM4PY library is used to perform process discovery and the creation of a Petri net.

```
p = Solver()
p.loadeventlog()
p.view_bpm()

p = Solver()
p.loadtripseventlog()
p.view_bpm()
```

Map All Events

```
p3 = Solver()
p3.loadeventlog()
p3.loadallevents()

from keplergl import KeplerGl

df_events = p3.getEvents()
df_latlon = df_events[["lat", "lon"]] # create a dataframe for latitude and longtitude

map= KeplerGl(height=500)
map.add_data(data=df_latlon, name='data_1')
map
```

Mapping & Statistical Analysis of Waiting Activities

This standardized process mining application can be used to create a map of waiting activities by vehicle id. Furthermore, there is generated statistical output based on the waiting activities.

Mapping

```
p1 = Solver()
p1.loadeventlog()
p1.loadevents("Waiting", "09f4b949-0ac1-4580-9628-a076bf858961")
p1.calculateStateDuration()
p1.addLocationInfo()
p1.addDayInfo()
p1.showdataframe()

from keplergl import KeplerGl

df_events = p1.getEvents()
df_latlon = df_events[["lat", "lon"]] # create a dataframe for latitude and longitude

map= KeplerGl(height=500)
map.add_data(data=df_latlon, name='data_1')
map
```

Statistical Analysis

```
p1.showDurationPlots()

p1.displayDurationStats()
```

Mapping & Statistical Analysis of Traffic Jams

This standardized process mining application can be used to create a map of traffic jam activities by a vehicle id. Furthermore, there is generated statistical output about the traffic jams.

Mapping

```
p2 = Solver()
p2.loadeventlog()
p2.loadevents("Traffic jam", "09f4b949-0ac1-4580-9628-a076bf858961")
p2.calculateStateDuration()
p2.addLocationInfo()
p2.addDayInfo()
p2.showdataframe()

from keplergl import KeplerGl

df_events = p2.getEvents()
df_latlon = df_events[["lat", "lon"]] # create a dataframe for latitude and longitude

map= KeplerGl(height=500)
map.add_data(data=df_latlon, name='data_1')
map
```

Statistical Analysis

```
p2.showDurationPlots()

p2.displayDurationStats()
```

Driving & Rest Time Analysis

This standardized process mining application can be used to see whether there occured daily drving times longer than 9 hours

```
p4 = Solver()
p4.loadeventlog()
p4.loadevents("Driving", "09f4b949-0ac1-4580-9628-a076bf858961")
p4.calculateStateDuration()
p4.addLocationInfo()
p4.addDayInfo()
p4.drivingtime()
```

This standardized process mining application can be used to calculate the weekly drving times

```
p4 = Solver()
p4.loadeventlog()
p4.loadevents("Driving", "09f4b949-0ac1-4580-9628-a076bf858961")
p4.calculateStateDuration()
p4.addLocationInfo()
p4.addDayInfo()
p4.drivingtimeperweek()
```

Appendix J

Do you see added value for EMONS Group in the mapping of activities that are occurring in a trip based on latitude and longitude?	The visualization of truck based upon the coordinates could be of help. The investigation of 'cumulative driver resting' times with relation to the guidelines or policies framed in accordance to Labor rules in Europe is an "Interesting" analysis.
Do you see added value for EMONS Group in the analysis of truck drivers' rest & driving times by comparing these to the European Guidelines for rest & driving time of truck drivers?	Yes. It gives us an insight about the "Percentage of variance".
What is your general opinion about standardized process mining applications in Officedog, do they seem useable for EMONS Group? If not, what could be changed to make them useful?	For now, we use applications pertaining to Microsoft. I saw use of Python which is not applied now. In future, it is going to be.
Are there other standardized process mining applications that can be used by EMONS Group that are not yet covered in this research?	N/A

Table 6: Feedback form of EMONS Group

Appendix K

Do you see added value for Officedog in the mapping of activities that are occurring in a trip based on latitude and longitude?	Option 1: In my opinion, displaying data and insights on a map can be a great tool for visualization. This way, it is easier to engage with stakeholders and help them connect insights from data to the real world. Specifically in the logistics industry, it can be really helpful to connect insights to location data. Option 2: Gathering insights from logistics companies' event data can be hard, as the types and definition of events differ between companies and their operational systems. By mapping these to the standardized OTM event types, a more uniform view of what these events mean can be generated. This helps in comparing data between organisations.
Do you see added value for Officedog in the analysis of truck drivers' rest & driving times by comparing these to the European Guidelines for rest & driving time of truck drivers?	Definitely. The results shown in the current implementation are promising. I believe that such analyses can be of great help to logistics companies for demonstrating compliance with regulatory guidelines.
What is your general opinion about how the standardized process mining applications have been developed? Do they seem useable for Officedog? If not, what could be changed to make them useful?	I believe that the developed Jupyter notebook is a good tool for further process mining applications. It is easy to get started with, and builds upon libraries that are widely used in Python data science applications. By standardizing the steps to use process mining, the barrier to entry for future research and development with similar data is substantially lowered.
Can you think of other standardized process mining applications that can be used, but are not yet covered in this research?	The most obvious alternatives that I can think of are the generic commercial ones from the likes of Celonis etc. However, these are not often tailored to the specifics of OTM event data. I must admit that I personally have not worked with process mining tools a lot, so my knowledge of what alternatives are available is limited.

Table 7: Feedback form of Bullit Digital