Flattening Event Data Extracted From SAP For Process Discovery

MSc Research Project
Master of Science in Data Analytics

James Cookland
Student ID: x20167814

School of Computing
National College of Ireland

Supervisor: Muhammad Zahid Iqbal

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	James Cookland
Student ID:	x20167814
Programme:	Master of Science in Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Muhammad Zahid Iqbal
Submission Due Date:	01/02/2023
Project Title:	Flattening Event Data Extracted From SAP For Process Discovery
Word Count:	7454
Page Count:	20

Signature:	[illegible]
Date:	1st February 2023

Flattening Event Data Extracted From SAP For Process Discovery

James Cookland
x20167814

Abstract

Extracting and mining event log data from the SAP database architecture is a hot topic in the field of process mining. The object-centric nature of the SAP database architecture often leads to challenging problems. The most prominent of which being convergence and divergence present in the event data. This research will explore the challenges associated with extracting, preparing and modeling event data from a technology distribution business, which uses SAP as its Enterprise Resource Planning software. Resulting in an end-to-end exploration of a process mining endeavor using event data extracted from SAP. The business case used for this research is that of a bespoke purchasing process that includes a crossover between the EINKBELEG and VERKBELEG object classes in the SAP change log tables. Additionally, this research will discuss methods of clustering, that add weight to specific activity notions by either increasing or decreasing activity notion granularity. This method effectively highlights the specific process activities that a researcher or business would like to see in the final process model.

1 Introduction

As modern businesses grow, the increase in process and data complexity is inevitable. Process mining is a relatively new data science field that started gaining traction in the late 1990's. More recently, the research released by Aalst et al. (2011) served as a defining moment for the field of process mining.

Process mining initially started with event logs following a predictable flat pattern. The field of process mining has evolved since 2012. Modern research in the field includes mining more complex and unconventional data sources. Such as Enterprise Resource Planning (ERP) systems, medical databases and even user interface logs present on websites or software applications. This led to the research conducted by Accorsi and Lebherz (2022) to define event log imperfections, their common causes and possible remedies. Process mining with event logs extracted from the SAP[1] ERP software, is now one of the most researched topics in the field. There are many complex and interesting challenges associated with both the extraction and preparation of event log data from SAP databases. One of the key challenges is the object-centric nature of the SAP data warehouse architecture.

Many solutions have been proposed in the relevant literature on the topic. A prominent technique proposed is the use of object-centric process mining. While this method

[1]https://www.sap.com/uk/index.html

is effective, van der Aalst (2019) pointed out that the use of object-centric event logs is mostly unsupported by both enterprise and open-source process mining tools. Making object-centric process mining a challenging endeavor to pursue for the uninitiated. Additionally, algorithmic clustering methods are often proposed to flatten the object-centric nature of ERP event logs. While they do indeed assist with the flattening of event logs extracted from SAP, expert knowledge is required to ensure critical information is not lost in the process. Resulting in algorithms that require an increased level of fine-tuning.

ROEST (2012) demonstrated how event logs could be extracted from SAP for process mining. While this shows that there is existing research on this specific topic, this article is from 2011. The research conducted in this thesis will aim at utilizing methods found in recent academic literature and on a newer version of SAP, SAP 770. Additionally, the utilisation of the Qlikview [2] application to extract tables from SAP was not discussed.

The event data was extracted from SAP databases for a real business. The methodology in this research includes steps to identify and isolate event activity notions of interest. Activity notions of interest can be defined as *activities that a business would like to highlight in a business process.* This research shows that activity notions of interest were isolated by reducing the granularity of activity notions of lesser interest. The methodology in this research demonstrates which methods to apply in order for an activity notion of interest to be present in the resulting process mining model, without increasing the complexity of the model. Moreover, the methods are able to isolate activity notions where the occurrence of the activity is low in the event log.

Therefore the research questions for this thesis are as follows.

Research Question: *What methods contribute to successfully flattening event log data extracted from the SAP 770 ERP system using real business data from a technology distribution company?*

Sub Question 1: *Can selective clustering and filtering of event data succeed in producing a process mining discovery model, that highlights specific activity notions of interest?*

To achieve this the following objectives are proposed:

1. Research methods of extracting and preparing event log data from SAP.
2. Perform extraction of relevant SAP tables using the Qlikview application.
3. Implement a single case notion in the event log to flatten the events.
4. Rename users in the event log with their respective teams to implement user anonymity while retaining the context of the change made.
5. Produce and analyze a list of activity notions to determine which activity notions should be isolated.
6. Perform clustering methods.
7. Perform Filtering on event log trace occurrences.

[2]https://www.qlik.com/us/

8. Verify that the clustering and filtering methods performed, successfully isolated the activity notions of interest, in the final process model.

9. Check final process model fitness and precision.

In the following 6 chapters, the key areas of this research will be presented. Section 2 discusses the literature relating to the topics of event log extraction and preparation, the majority of which relate to event log data extracted from ERP systems. Additionally, the relevance of previous and modern research in the field will be compared. In section 3, the research methodology discusses how tools such as Qlikview, Python and Prom were used to extract and prepare the data. This section also presents how and why the proposed methods were chosen. Section 4 discusses the design specification regarding the remote machine and clustering algorithms required to replicate the research. Section 5 presents the final implementation of the clustering method applied and how it affected the final process mining model. Section 6 evaluates the applied methodology to extract, prepare and model event data from SAP. Finally, section 7 discusses the benefits, limitations and possible future research related to the process mining methodology presented in section 4.

2 Related Work

According to Pajić and Bečejski-Vujaklija (2016), ERP systems are often ill-suited for extracting traditional process mining event logs. The author asserts that the data architecture of ERP systems limits the ability of data analysts to clearly identify the full cycle of a case due to their object-centric architecture. Making the extraction of traditional process mining event logs a difficult task for researchers. Instead, the author proposed the use of artifact-centric event logs from ERP systems, stating that this method would reduce the presence of divergence and convergence in the event logs.

In contrast, Berti et al. (2022) presented an approach for extracting event logs from the SAP ERP software to build Object-Centric Event Logs (OCEL). The suggested approach starts with the building of a Graph of Relations (GoR), which displays the connections between the various tables in SAP. According to the author, the creation of OCEL using this method, naturally reduces the risk of convergence and divergence issues that commonly arise while obtaining event logs from SAP. This method built the event logs from the SAP tables specific to records, details and transactions. For example, the author suggested using the SAP record table EKKO as a core table and investigating what SAP tables connect to it. From the resulting GoR, the change logs CDHDR and CDPOS can be filtered based on specific tables and fields present in the GoR. Using this method relies heavily on expert knowledge of the SAP ERP software. Finally, the author discusses setting the event concepts to a human-readable format from the field CDHDR.TCODE, CDPOS.TABNAME, CDPOS.FNAME and the old and new values also present in CDPOS.VALUE_NEW and CDPOS.VALUE_OLD to provide the context associated with the change. According to the author these would provide richer detail when the process mining model is produced.

As previously discussed, traditional event logs extracted from SAP are likely to result in significant divergence and convergence in the final process model. As discussed by van der Aalst (2019) this is likely to occur if event logs are flattened to produce a traditional style event log. Additionally, the author defined convergence as when multiple case identifiers in the event log, converge on one event. Divergence on the other hand relates to where a single case identifier diverges into multiple events at the same time or the same event multiple times. The author presented how OCEL can be used as a means to mitigate the presence of divergence and convergence. However, the author suggested that current process mining tools do not yet have the capacity to process OCEL event data sufficiently.

While producing a standardised framework for extracting and preparing event data extracted from SAP databases, Accorsi and Lebherz (2022) presented 13 best practices to consider. In best practices 1 to 4, the author asserted that the aims of the research should be clearly identified with stakeholders as well as explaining what quality of information the process mining project is likely to produce. The author then suggested extracting small samples of each of the relevant tables to determine which fields could be used to apply filters on. The proposed techniques are designed to reduce the volume of data extracted from SAP as well as refine which data will best represent the process being analyzed. Best practice 4 suggested that a development or sandbox server should be used for the data mining exercise. An important factor to consider is that any version of SAP that is being used to extract event logs must, at some point, have real-life data migrated from the production server. A caveat to this method is that the production, development or test environment may have additional events present, where users perform actions that would not be permissible in the production environment. Best practices 5 to 9 the author presented considerations when preparing the event logs post-extraction. It is suggested to not only include the name of the change but also its context. Such as whether the event relates to a price change or date change in the relevant transaction or table. Finally, best practices 10 to 13 discussed considerations to take into account with event log preparation relating to the process mining model phase. This included the preparation of additional attributes in the event logs as well as considering how the structure of the event logs may affect the modeling algorithms. These practices align with the suggestions of Berti et al. (2022) discussed previously.

In a study conducted by Suriadi et al. (2017), a systemic identification and cleansing of event logs for process mining was presented. The author identified 11 of the most common pattern imperfections that may affect event log data. The most relevant of these imperfections, with regard to event logs extracted from SAP change tables, are; Form-based event capture, scattered case, collateral events and homonymous labels. Of these 4 imperfections, form-based event capture is the most prevalent. The author suggested clustering these events into a single event. While the author discussed scattered case imperfections, the definition provided by Fischer et al. (2022) provides a more concise method of identifying and mitigating scattered or missing events from the event logs. The author presented additional event log imperfections which relate to timestamps, and how to quantify them.

ROEST (2012) presented methods for extracting, preparing and developing a process mining model using SAP event data. The author presented the use of SAP transaction SE16 to explore SAP tables. the author also suggested the use of Java connectors, Microsoft Excel VBA as well as the use of the Talend application.

The research conducted by Eck et al. (2015), defines two types of clustering. The first, "is-a" relates to clustering event activities essentially by their context. In this definition, activities may not be directly connected to one another but the general concept of these activities is similar. In contrast, the "part-of" clustering definition relates to activities that are subtasks of another. As De Weerdt and Wynn (2022) stated, the clustering of events relies traditionally on expert domain knowledge of the process activities. Clustering techniques are used to reduce granularity in the event logs, subsequently resulting in simplification in the final process model. In a systemic literature review on the topic of event log abstraction, van Zelst et al. (2021) suggested that a process model derived from event data that have not been clustered, will likely result in an overly complex process model. In concluding remarks, the author also stated that evaluation techniques such as conformance checking would not be applicable, due to the high level of granularity in the data. In addition, Folino et al. (2015) discussed methods for clustering events, using a high-level process model (HLPM) to detect actions that could possibly be clustered together. The HLPM model treats the clustering of events as a traditional machine learning clustering problem, clustering events based on a preset list of attributes. Alternatively, Rehse and Fettke (2019) proposed the use of an algorithm to evaluate the similarity scores between activity labels both in time proximity and activity label hierarchy. Based on the threshold set on the scores the algorithm would cluster these events together. The last two clustering techniques mentioned here are analytical in nature, in contrast to the use of domain knowledge discussed by De Weerdt and Wynn (2022).

An additional area of event data preparation is that of event filtering. As Berti and van der Aalst (2022) stated, the use of event log filters is critical for flattening object-centric event data extracted from ERP systems. The author also provided in-depth definitions for attributes related to event logs such as traditional event logs, directly follows graphs and object-centric event logs.

In relation to process mining models, Leemans et al. (2015) suggested the use of the Inductive visual Miner (IvM) due to its easy-to-use nature as well as its ability to export model information to be used with other plugins in the PROM software. The IvM also has a feature for displaying the trace deviations, especially relevant to event logs extracted from the SAP ERP. An analysis of alternative process mining algorithms was presented by Mishra et al. (2018). This includes α miner, heuristic miner, fuzzy miner and directly follows graphs. The author also indicated the PROM software as an all-in-one solution for process mining as it includes plugins to analyze process mining models such as conformance checking. In contrast, van der Aalst (2019) suggested the use of Directly Follows Miner (DFM) to reduce the risk of convergence and divergence when analysing object-centric event data.

Regarding the evaluation of process mining models, Muñoz-Gama and Carmona (2010) presents the use of the ETConformance metric to determine the precision of the resulting Petri-net generated by the process mining model. Additionally, Aalst et al. (2011) presents 4 metrics to assess the performance of the final process model. These are fitness, simplicity, precision and generalization of the process model.

The research conducted by Berti et al. (2022) proved to be the most influential of the academic literature in the research. The author presented a method to extract event log data from SAP in a clear and concise manner when compared to De Weerdt and Wynn (2022). The naming conventions for the activity notions presented by the author, are also clear and straightforward to implement.

In line with this, van der Aalst (2019) discussed that the flattening of event log data extracted from ERP systems such as SAP are likely to cause divergence and convergence imperfections. While the relevant literature provided many useful techniques for process mining with SAP event data, very few examples of implementations were found. Additionally, the methods implemented by ROEST (2012) are now over 10 years old. Raising an intriguing question as to whether there are new methods available for implementing a process mining project on event logs extracted from an SAP ERP system. Thus presenting the first gap in the relevant research.

While there are methods already presented for extracting event data from SAP. None of the research papers reviewed contained a method for using the Qlikview application to extract event data. This presents the second gap in the research. The third gap related to the fact that the state-of-the-art research on this topic is only conducted in either purchase-to-pay (P2P) or order-to-cash (O2C) processes or some other predefined business process. Making space for a method that demonstrates its ability to produce a process mining model for hybrid processes using a mix of modern state-of-the-art techniques. The findings from this literature review fulfilled the first objective of this research.

3 Methodology

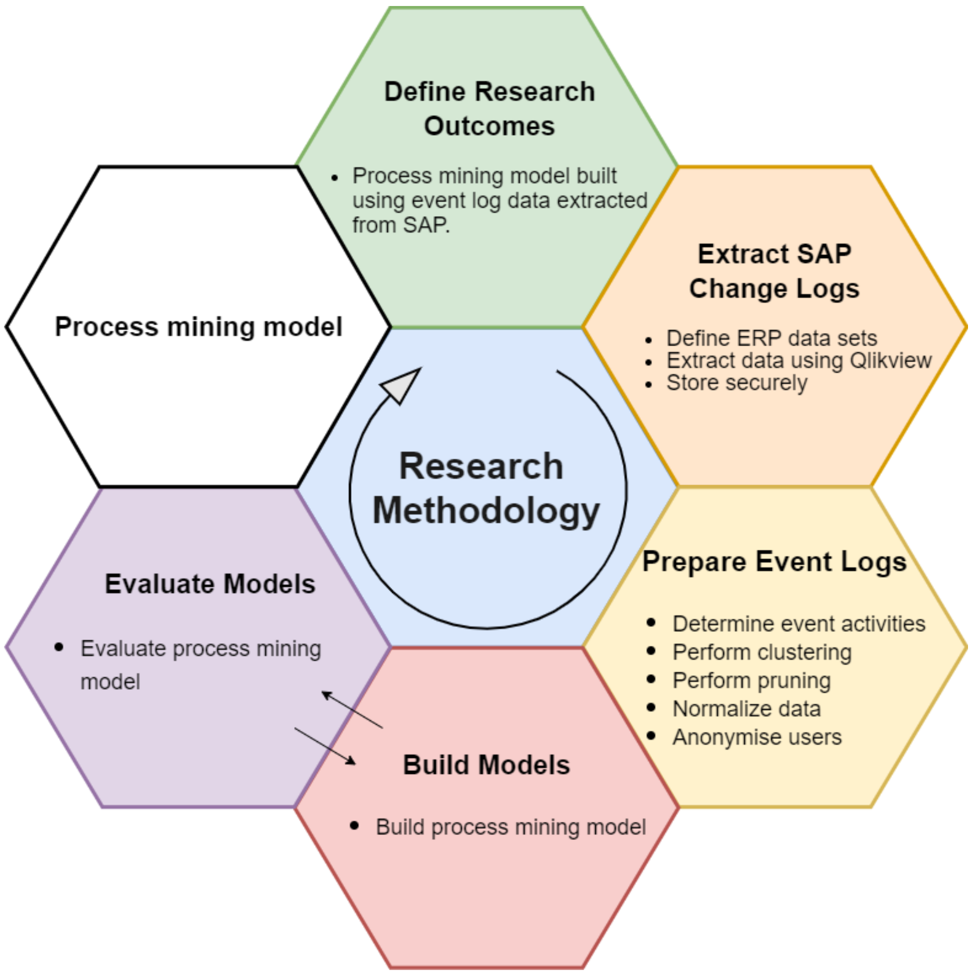

Figure 1: Research methodology for producing a process mining model from event data extracted from SAP.

The methodology for this research was created specifically for the creation of process mining models for bespoke processes, using event logs extracted from the SAP ERP software.

Figure 1 displays the methodology used to complete the research. This methodology was loosely based on the CRISP-DM data mining process model. As seen, the phases include Defining Research Outcomes, Extracting SAP Change Logs, Preparing Event Logs, Building Models, Evaluating Models and finally, producing a Process Mining Model. As with the CRISP-DM data mining process model, the process starts by defining the research outcomes that the researcher would like to achieve. The process follows a clockwise direction except between build models and evaluates models, where experimentation takes place to improve the process mining model using the parameters to fine-tune the results. Where the models do not achieve the desired results set out in the initial phase, the process can be restarted from the required phase in the methodology process model.

3.1 Define Research Outcomes

As mentioned in the introductory chapter, the expected outcome is a process mining model generated using event data extracted from SAP using the Qlikview application. The proposed methods should cluster activity notions that are already known to the researcher by abstracting less important activity notions to be more general. By retaining granularity on the activity notions of interest, the model should retain these granular activities while generalizing the rest. Additionally, the methodology includes the evaluation of the final process mining model both for its ability to isolate activites of interest and its fitness and precision metrics.

3.2 Extract Data From SAP

While the relevant literature, such as the author of Berti et al. (2022), supports the use of direct connections between process mining software and ERP systems[3]. The choice of using the Qlikview application, along with the respective SAP to Qlikview connector, was based on the direction of the business. Additionally, in line with the data governance policy of the business, a remote server was set up to host the research.

The remote server was installed with Windows 10, 16GB of RAM, an Intel(R) Xeon(R) Gold 6142 CPU @ 2.60GHz 2.59 GHz (2 processors) processor and 249GB of storage memory.

The Qlikview 12 app was installed with an SAP connector to a sandbox replica of the business's production SAP database (Accorsi and Lebherz; 2022). The sandbox database is periodically updated with data from the SAP production database, for the purpose of development and testing. This database was sufficient to mine SAP data from, as it contains all historical data from the production system. The open-source process mining software Prom 6.1 was then installed to produce the process mining models.

Resources in academic literature were used to discover the relevant SAP tables for the hybrid purchasing process, such as Berti et al. (2022). However, online resources present greater detail on what tables contain as well as the filters that could be applied to restrict the search. The primary resources used to discover the required data, include the list of input tables for the purchase-to-pay[4] and order-to-cash processes[5], provided

[3]https://github.com/Javert899/sap-extractor

[4]https://docs.uipath.com/process-mining/v2020.10/docs/input-tables-of-the-sap-p2p-connector

[5]https://docs.uipath.com/process-mining/v2020.10/docs/input-tables-of-the-sap-o2c-connector

by the process mining enterprise software company UiPath[6]. Additional information on SAP tables such as field names, primary keys and field descriptions can be found on both SAP community boards[7] and other third-party websites[8]. It is important to note however that due diligence is required when using these online resources for accurate data, cross referencing and experimentation was conducted often, to ensure the accuracy of the acquired information.

Table 1: SAP Tables and Filters Used in Qlikview For Extracting Data From SAP

SAP Table	**SAP Table**	**SAP Fields**	**Filter Used on Field**
		Sales Order Data	
VBAK	Sales Order Header	VBELN NETWR KUNNR ZZEXPORT_CO WAERK	ERDAT <= '20211231' AND ERDAT >= '20210601'
VBAP	Sales Order Item	VBELN MATNR MVGR1 PSTYV	ERDAT <= '20211231' AND ERDAT >= '20210601'
		Purchasing Data	
VBEP	Purchase Requisition Data	VBELN BANFN EDATU	EDATU <= '20211231' AND EDATU >= '20210601'
EKKN	Purchase Order Data	VBELN EBELN	AUGCP <= '20210401' AND AUGCP >= '20220331'
BSEG	Billing Header	BELNR VBEL2 AUGCP	VBELN <= Max Sales Order from VBAK Table AND VBELN >= Min Sales Order from VBAK Table
		Change Logs	
CDHDR	Change Log Header	OBJECTCLAS OBJECTID CHANGENR USERNAME UDATE UTIME TCODE	EDATU <= '20211231' AND EDATU >= '20210601'
CDPOS	Change Log Item	OBJECTCLAS OBJECTID CHANGENR TABNAME TABKEY FNAME CHNGIND VALUE_NEW VALUE_OLD	CHANGENR <= Max Change Number from CDHDR Table AND CHANGENR >= Min Change Number from CDHDR Table
		Master Data	
DD02T	SAP Table Names	All	DDLANGUAGE = "E"
TSTCT	SAP Transaction Names	All	SPRSL = "E"
DD04T	SAP Field Names	All	DDLANGUAGE = "E"
TCDOB	SAP Object Types	All	

Once the relevant fields required for the research were identified, they were extracted using the Qlikview 12 application. As seen in Table 1, the SAP tables were categorized

[6]https://www.uipath.com/
[7]https://answers.sap.com
[8]https://www.se80.co.uk

by their data content types. While the core tables for this research are the change logs CDHDR and CDPOS, these did not contain sufficient data to isolate the sales orders related to software. For this reason, the sales order data was extracted first. Using the tables VBAK for sales order header data and VBAP for sales order item data, Table 1 defines the fields and filters used for each in the SQL statements used (Accorsi and Lebherz; 2022). The key filter for the SAP data extraction was the document creation times. This was set to collect data from 01/06/2021 to 31/12/2021. This period was selected as the data was collected from the sandbox ERP system, as mentioned earlier. As such, data updates from the production system are only transferred periodically. The selected time frame was chosen due to the stability of the data in this period. The field PSTYV in the VBAP table was then used to filter out sales orders that related only to software orders, signified by the value "ZLI" being present in that field.

The second set of rows in Table 1, under purchasing data, relates where VBEP for purchase requisition documents, EKKN for purchase order documents and BSEG for billing header data for purchasing. These tables were extracted for the purpose of connecting the events related to purchasing, to their respective sales orders.

Since the research aims are to identify methods for process discovery only, additional attributes were excluded from the data extraction.

Finally, the change logs themselves are extracted, the required fields and filters used can be seen in Table 1 under the Change Logs section.

One benefit of using SAP data to generate event logs is that most of the table attributes and naming conventions are standardised. Meaning that if the event logs were generated from SAP standard tables, the same methods could be applied across multiple industries or businesses that use SAP as their ERP system.

3.3 Prepare Event Logs

Preparing event logs is a critical step in producing a process mining model. The following sections will discuss the preparation methods used in this research project along with the relevant literature to support such methods.

3.3.1 Combining SAP Object Types to Create Complete Event Case Notion

One of the biggest challenges associated with generating event logs from SAP tables is that both sales data and purchasing data are stored in separate object classes, namely VERKBELEG and EINKBELEG respectively. This difference in object class causes a disparity between the OBJECTID field in the change log tables. both of which contain the sales document number in the case of VERKBELEG and the purchasing document number in the case of EINKBELEG, as its primary event case notions. In contrast, traditional process mining tools require that a single event case notion variable is used to determine the full life cycle of a case.

To do this the SAP table EKKN, which contains purchase order data, was used to link the SAP sales order to its related purchase order. As such, case identifiers for event logs mined from the EINKBELEG object class where replaced with the sales order number. The same method was then applied to purchase requisitions contained in SAP table VBEP and billing information contained in SAP table BSEG.

3.3.2 Determining Event Activity Notions

As suggested by Berti et al. (2022), event activity notions were generated for the event logs using the tables DD02T, DD04T and TSTCT. These were used to add the names of tables, fields and transactions present in the SAP tables CDHDR and CDPOS.

Additionally, the user names which relate to each change in the event logs are also an important field to include, as this gives context to the change. However, including the names of users could result in considerable ethical considerations. For example, if an employee's name is observed to be correlated to the event traces that take the longest or show an increase in additional activity behavior, this may indicate to management that this employee is slower or less diligent. Therefore comparing the output of that person to others on the same team without considering what other factors may contribute to the observed behavior in the event logs.

So as not to use the process mining project in a potentially unethical manner, each name in the event logs was replaced by the business area in which they work. For example, when a user works in the software purchasing team, their name was replaced with "Purchasing". This method of name anonymisation removes user names from the event logs while retaining the context in which the change was made. One caveat of this approach is in the event that a team is made up of only one member. While not including the name of the user, it would be obvious to the business which user made the change.

In addition to the activity notions components mentioned, team names were added to the event notions. This decision was made so as to add context to each of the changes. For example, where the change "delivery block" was made by a member of the credit control team, this would indicate that the customer related to that order does not match the credit management criteria of the business. Thus initiating a credit review by the credit control team. In contrast, the same change made by the purchasing team would only indicate minor infractions to the credit management protocol. Minor infractions can be resolved by the purchasing team themselves, resulting in reduced additional work required to resolve the issue. The name of the team that made the change was added to the event description. The final format of the change description then was "Transaction Name - Field Name - Changed to Value - Team".

3.3.3 Event Log Clustering

Form-based event capture is one of the most prevalent imperfections found in the event logs extracted from the SAP change log tables. Since users may enter multiple points of data into a single SAP transaction at once, such as in the event of order creation, there will be multiple events with the same timestamp. Additionally, the event activity construction methods used in section 3.4.2 produced a high level of granularity for each event activity. This level of granularity produces the convergence and divergence problems mentioned by van der Aalst (2019). To address this, clustering methods were applied to the event activities based on activity label similarity. As discussed by De Weerdt and Wynn (2022), domain knowledge of the process was used to determine where granularity should be kept and where higher levels of abstraction could be applied to activity labels. While the suggested methods in the literature propose the use of algorithms to automatically detect where clustering can take place, the aim of this research was to display the effectiveness of manually selecting activity notions as activity notions of interest. It should be noted, however, that this method increases the risk of losing important event information. Therefore, selecting which case notions to keep as case notions of interest, must be considered carefully and should be conducted only by those that are familiar with the process.

Case notions of interest were chosen by analysing the value counts of each case notion present in the event logs. Two events were chosen to retain granularity.

3.3.4 Event Log Filtering

The filtering plugins native to the PROM[9] software were used for this research and are sufficient for applying multiple types of filters based on different attributes of the event logs. These filters were used to remove low-frequency traces. While filtering event logs increases the risk of losing important information, this research aims at process discovery. Including low-frequency events would only serve to increase complexity in the final process discovery model.

Filtering out low-frequency traces in the event logs in this methodology is based on a trial-and-error approach. The filter used in ProM was the "Event Filter in Event Log" found in the event log visualizer window. Once filtering has been applied, the resulting event log was used to generate a process mining model.

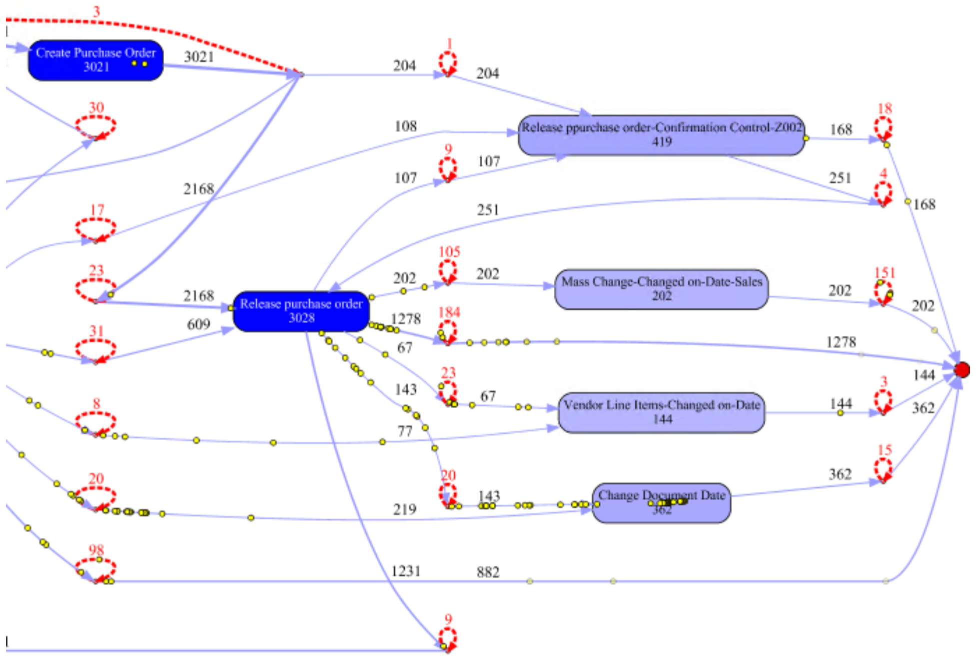
Create Purchase Order
3021
Release ppurchase order-Confirmation Control-Z002
419
Release purchase order
3028
Mass Change-Changed on-Date-Sales
202
Vendor Line Items-Changed on-Date
144
Change Document Date
362

Figure 2: Zoomed in image of DFM model displaying the deviations in the model represented by red dotted lines.

3.4 Build Process Mining Model

As per the suggestion by van der Aalst (2019), the PROM plugin selected to create the process mining models was DFM. Leemans et al. (2019) presented a guide on the use of this plugin. The DFM plugin for ProM also has the functionality to display the conformance of generated models. By selecting "paths and deviations" in the bottom

[9]https://www.promtools.org/doku.php

right of the ProM application, deviations from the generated process model can be seen by the red dotted lines, as in the zoomed-in image of the process model in Figure 2. Figure 2 shows a zoomed-in image of the DFM from the event logs before any preparation techniques were implemented.

4 Design Specification

The design specification diagram for this project can be seen in Figure 3. This diagram shows the end-to-end design required to collect event logs, flatten them and produce a process discovery model.

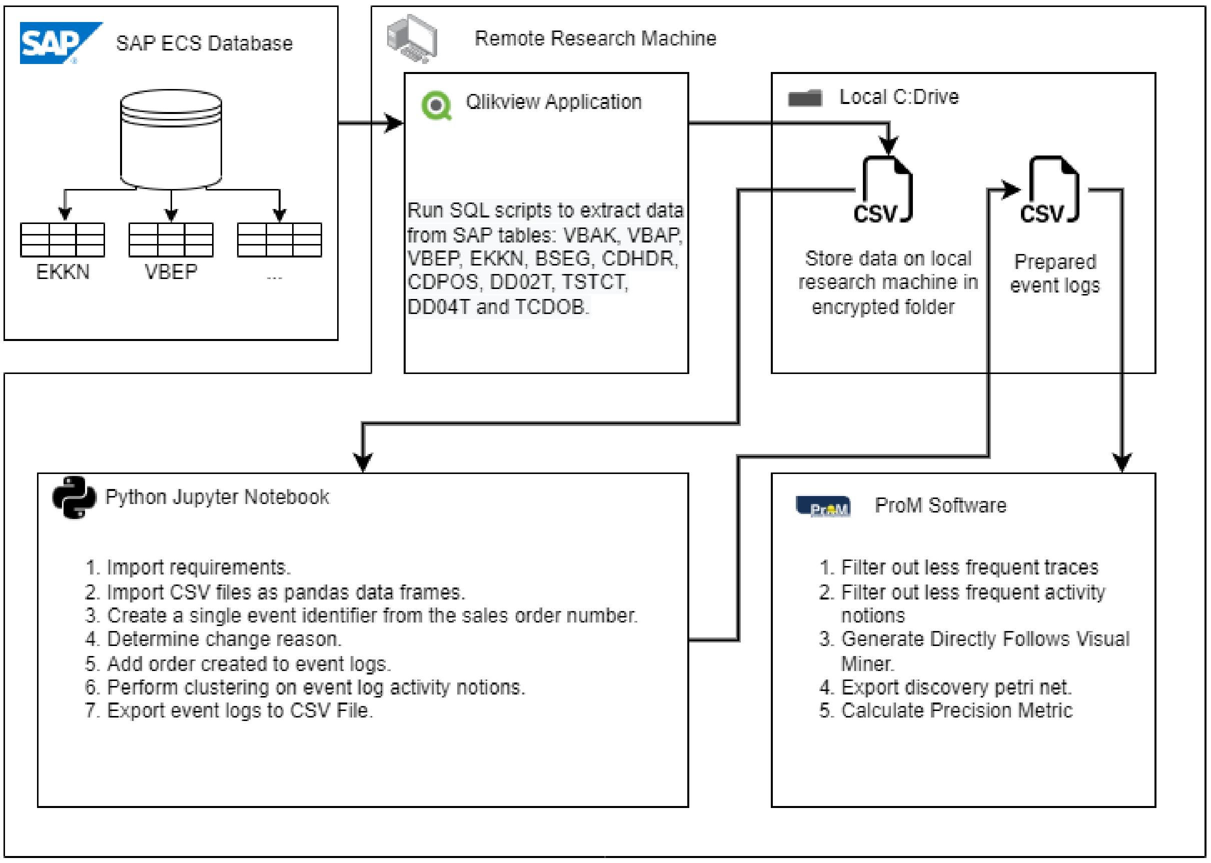
SAP ECS Database
EKKN
VBEP
...
Remote Research Machine
Qlikview Application
Run SQL scripts to extract data from SAP tables: VBAK, VBAP, VBEP, EKKN, BSEG, CDHDR, CDPOS, DD02T, TSTCT, DD04T and TCDOB.
Local C:Drive
CSV
Store data on local research machine in encrypted folder
CSV
Prepared event logs
Python Jupyter Notebook
1. Import requirements.
2. Import CSV files as pandas data frames.
3. Create a single event identifier from the sales order number.
4. Determine change reason.
5. Add order created to event logs.
6. Perform clustering on event log activity notions.
7. Export event logs to CSV File.
ProM Software
1. Filter out less frequent traces
2. Filter out less frequent activity notions
3. Generate Directly Follows Visual Miner.
4. Export discovery petri net.
5. Calculate Precision Metric

Figure 3: Design specification diagram for creating process mining model from event logs extracted from SAP.

4.1 Extract Data From SAP

As seen in Figure 3, the Qlikview 12 application was used to extract the relevant tables directly from SAP. This application relies on SQL queries to select data from the SAP database, via the SAP connection plugin. The SQL scripts are separated into tabs but follow an interpreted style when running the SQL statements. The Qlikview application runs each statement in the tabs from left to right. Each tab contained the SQL statement to extract data from a different SAP table, using the name of the SAP table it is extracting as the tab name. By programming Qlikview using this method, amending SQL statements and troubleshooting become much easier as specific table extractions are easier to identify. The tables, fields and filters applied can be seen in Table 1. All files are then exported as CSV files to an encrypted folder on the local drive of the research machine. The SQL scripts are constructed as "SELECT {SAP Fields} FROM {SAP Table} WHERE {Filter Used on Table}" for each row in Table 1.

4.2 Prepare Event Logs

Data preparation is then completed using a Python Jupyter notebook. Figure 3 displays each of the programming components that make up the Jupyter notebook in the bottom left of the image. This includes importing required methods, importing the CSV files as data frames, creating a single case identifier, determining activity notion, adding order creation to event logs, performing clustering and finally exporting the prepared event logs to a CSV file on the remote machines local directory.

Table 2: Regex patterns for clustering Event Logs with corresponding anchors

Event Activity Regex	**Transaction**	**Field**	**Changed-To**	**Team**
Blocked SD Documents(.*?)Sales	x_1			x_4
Blocked SD Documents(.*?)Credit Control	x_1			x_4
Call MIRO.*	x_1			
Change Document-Changed on-Date.*	x_1		x_3	
Change Document-Planning Date.*	x_1	x_2		
Change Document Date.*	x_1			
Change Purchase Order.*	x_1			
Change Sales Order-Address.*	x_1	x_2		
Change Sales Order-Billing.*	x_1	x_2		
Change Sales Order-Condition rate.*	x_1	x_2		
Change Sales Order-Delivery block-Z.*	x_1	x_2		
Change Sales Order-Delivery block-nan-Sales	x_1	x_2		
Change Sales Order-Delivery block-nan-Purchase Admin	x_1	x_2		
Change Sales Order-General-C.*	x_1	x_2	x_3	
Change Sales Order-Home address.*	x_1	x_2		
Change Sales Order-Next date.*	x_1	x_2		
Change Sales Order-PO number.*	x_1	x_2		
Change Sales Order-Pricing date.*	x_1	x_2		
Change Sales Order-Reason for rejection.*	x_1	x_2		
Change Sales Order-Synchronization key.*	x_1	x_2		
Create Purchase Order.*	x_1			
Display Purchase Order.*	x_1			
Release purchase order-Goods Receipt.*	x_1	x_2		
Release purchase order-Confirmation Control-Z002.*	x_1	x_2	x_3	
Release purchase order.*	x_1			
Vendor Line Items-Changed on-Date.*	x_1	x_2		

4.2.1 Event Log Clustering

The decision to choose event cases as an activity notion of interest was based on whether the case notion was not present in the known business process found in Figure 4 and whether the case notion occurred frequently.

Similar to the notation presented by Folino et al. (2015), an event activity for a single event can be represented by τ_i. τ_i then, represents the tuple set of activity anchors discussed in section 3.4.2. Each event attribute can then be represented by x_1 for transaction name, x_2 for field name, x_3 for changed to value and x_4 as the team that made the change. Therefore, each event activity label can be expanded as $\tau_i = (x_1,x_2,x_3,x_4)$. Where a new activity notion is generated using this method, the resulting new activity notion can be represented as $new\tau_i$. As mentioned, the key imperfections produced by flattening event logs are related to the loss of data granularity that may otherwise be important. To reduce the impact of flattening, the following method was implemented.

All event activities were isolated into a data frame with the corresponding value counts of their presence in the event logs. Event activities were then sorted by their name in alphabetical order. This produced a list of event activities in order of similarly named activities. This then provided an opportunity to identify which events should be considered as activity notions of interest based on the action type and the number of times it appeared in the event logs. The decision on which events to cluster was based on manually analysing each event, out of the 182 events present, to determine where granularity was not necessary.

For example, there were 11 event activity types that contained the anchor "Create Purchase Order" (x_1). The remaining anchors x_2, x_3 and x_4 contained additional information in relation to the creation of purchase orders. For the purpose of this research, only KPI events were necessary to retain granularity on. For this reason, all events that contained "Create Purchase Order" in anchor x_1, were converted to $new\tau_i = x_1$. Table 2 displays how each of these events where clustered. The first column displays the regex pattern that was used to identify the activity type and the preceding columns display how the new labels were constructed. Taking the first row in this table as an example, any events that start with "Blocked SD Documents" and end with "Sales" were converted to only contain x_1 and x_4 as the activity name.

Taking r_i as a regex pattern, R represents the full list of regular expressions where $r_i \in R$. Algorithm 1 describes the Python method applied to the event logs. Firstly, the regular expressions(r_i) and new activity labels($new\tau_i$) were converted into a dictionary using the values seen in Table 2. $Event\tau_i$ represents the activity label present for each event in the event logs. Therefore, for each $Event\tau_i$, the activity notion was replaced with the $new\tau_i$, only if $Event\tau_i$ corresponds a match to any regular expression present in R.

Algorithm 1 Event Clustering

```
1: Set R as {$r_1$:$new\tau_{i1}$,...,$r_n$:$new\tau_{in}$}
2: for $Event\tau_i$ in Event Log do
3:     if $Event\tau_i$ matches any $r \in R$ then
4:         SET $Event\tau_i$ as $new\tau_i$
5:
```

5 Implementation

To start the implementation a remote server machine was set up using the requirements discussed at the beginning of section 3.3. The required applications were then installed including Qlikview 12, Qlikview SAP Connector, Python3.9.13, Visual Studio Code and ProM 6.11.

Once all applications were installed the tables VBAK, VBAP, VBEP, EKKN, BSEG, CDHDR, CDPOS, DD02T, TSTCT, DD04T and TCDOB were extracted from SAP using

Qlikview 12 application. Only the fields seen in Table 1 were extracted from these tables with the accompanying filters applied.

A single event case notion was then applied to the event logs using the original sales order that the change related to. This fulfilled the third objective of this research.

The replacement of user names with their respective teams was then performed. A table was generated for each user in the event logs with the first column containing the name of the user and the second column containing the relevant team. To match users to teams the company organisational chart was referred to. This was completed by manually reviewing each of the names in the table and matching the team name in the table. A Python Pandas function was then implemented to replace each name in the event logs with the team name. This completed the fourth objective of this research.

The list of activity notions and their respective value counts present in the event log was then generated. The case notions containing "Release purchase order-Confirmation Control-Z002" was chosen first as this was not present in the initial process model found in Figure 4. Additionally, it occurred 437 times in the event logs meaning it occurred in more than 2.8% of traces. This activity notion relates to where master data of the items in the PO are incorrect. Software items that are set up as hardware items require a different confirmation process. When software orders with these items require releasing at the purchase order stage, they require additional actions to resolve. Thus resulting in a bottleneck in the process. This was included to demonstrate that if a case notion of interest with low frequency in the event logs can be included in the process mining model, without increasing the complexity of the model, this would demonstrate the effectiveness of the clustering methods at isolating a case notion of interest.

The second activity notion chosen was "Change Sales Order-Delivery block-nan-Credit Control". Delivery blocks that require releasing by a member of the credit control team require the purchasing team to wait for them to be released as access to this type of delivery release is restricted to the credit control team. This activity notion was present in 1304 cases, accounting for just over 8.4% of cases.

While there were more events that could have been chosen, the results of selecting a case notion of interest would be difficult to measure. These two events were chosen to display the efficacy of preparing event logs to highlight a case notion of interest regardless of their frequency.

Once activity notions of interest were identified, the activity abstraction technique discussed in section 4.2.1 was implemented.

The ProM application plugin called "Event Filter on Event Log (Filterd)" was used to filter event traces that occurred less than 5 times in the event log.

A process model was then generated using the Directly Follows Miner in the ProM application. This resulted in the process model seen in Figure 5.

Finally, the "ETConformance" plugin was utilised to produce the precision metric by replaying the unfiltered event log on Petri-net generated by the Directly Follows Miner. The unfiltered event logs were used rather than the filtered version to see how the model performed with additional traces. This completed the ninth objective set in the introductory chapter of this report.

6 Evaluation

This section will discuss the results of implementing the proposed methodology from section 3 in a hybrid purchase-to-pay process.

6.0.1 Case Study on Hybrid Purchase-to-Pay Process

The business process chosen to conduct the research on, was that of a hybrid software purchasing process. This includes parts of the standard order-to-cash (O2C) and purchase-to-pay (P2P) processes. The process starts with the sales order generated by the purchasing team, the purchase order is then created for the sales order items and finally, the purchase order is approved for payment once the software license is received.

Within this process, there are known exceptional cases that may arise. Such as orders that raise a delivery block, price differences due to special promotions or the rejection of orders.

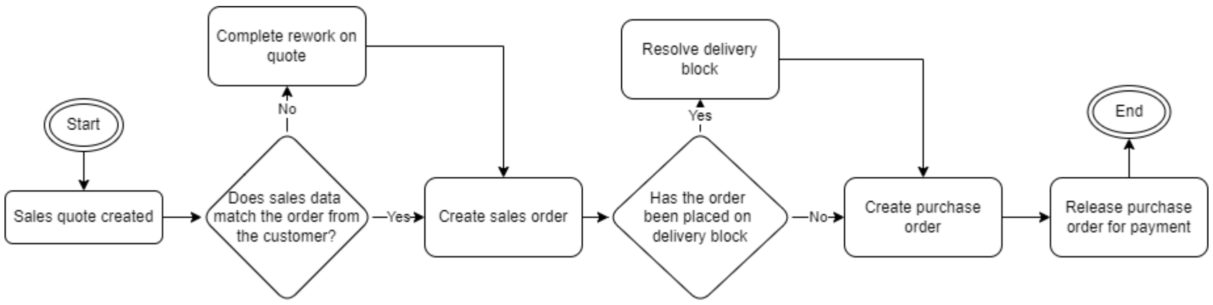
Start
Sales quote created
Does sales data match the order from the customer?
No
Complete rework on quote
Yes
Create sales order
Has the order been placed on delivery block
Yes
Resolve delivery block
No
Create purchase order
Release purchase order for payment
End

Figure 4: Process flow diagram of the software purchasing process.

Unlike the standard O2C process, the purchasing team does not process payments received by customers. Nor do they process the payment of vendors, as in a standard P2P process. Since this is the case, the receipt and sending of payments were excluded from the study to reflect the actual process conducted by the software purchasing team. A visual representation of this process can be seen in Figure 4.

By importing the unprepared event logs into Prom, there were 182 activity notions, 3036 cases and 20232 events detected. Once the clustering methods were applied the resulting event log contained 61 activity notions, 3036 cases and 15502 events.

Filtering events to remove event traces that occurred less than 5 times resulted in a new event log data set that contained 20 activity notions, 2556 cases and 12366 events. Even though this was a low threshold with which to reduce the event logs, the filtering method significantly affected the data set's size. The parameters used for this model, available on the right side of the ProM application, were set to include all activities present in the data set and retain paths that occur 80% of the time. The fitness of this model was 96.3% according to the ProM application. Additionally, there were 541 deviations present in the model.

The final model produced using the prepared event logs can be seen in Figure 5. In order of left to right in Figure 6, the activity notions included in this model where "Order Created", "Blocked SD Documents - Credit Control", "Create Purchase Order", "Release Purchase Order - Confirmation Control", "Release Purchase Order", "Change Sales Order - Condition Rate" and "Change Document Date".

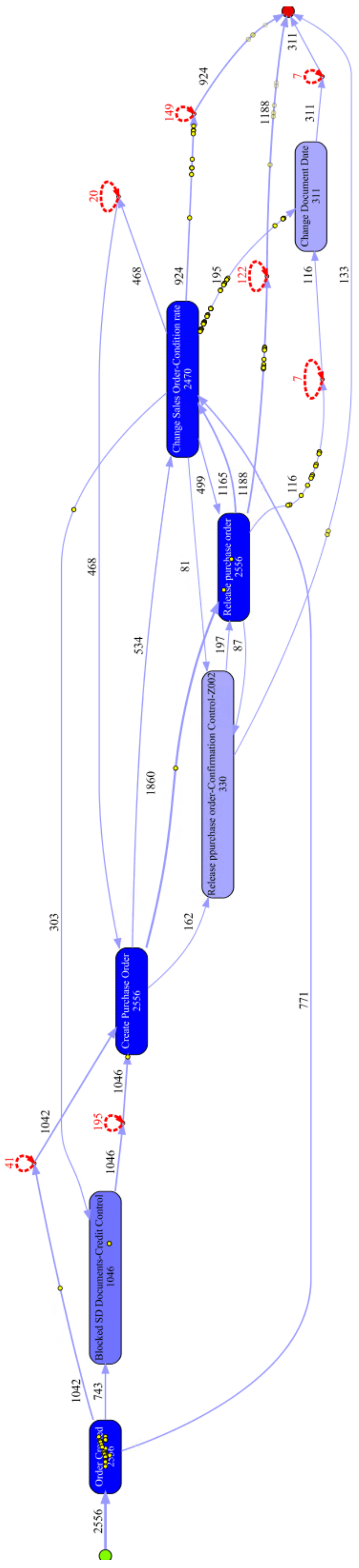

Figure 5: Resulting DFM process model from filtering out traces that occur less than 5 times.

Additionally, when importing the resulting Petri-net as well as the unfiltered event logs into the "Check Conformance using ETConfomance" plugin in ProM, the resulting precision of the model was 82%.

7 Conclusion and Future Work

Flattening event log data from ERP systems such as SAP is a challenging task. The SAP database architecture is object-centric in nature, resulting in significant data wrangling to flatten it. By selecting a single case notion, abstracting unnecessary granularity from activity notions and carefully filtering low-occurring event traces, the event data extracted from SAP were flattened.

As discussed, the Qlikview application was used to extract the data from SAP. The benefit of using this application to extract data from SAP is that the SAP tables can be queried at specific times of the day from the SAP production environment. These extractions can be set at times of the day that are non-critical and won't interrupt the day-to-day interactions of users on the application. The data extracted by Qlikview can then be stored in an accessible location for the process mining project to pick up, in order to perform conformance checking. A limitation to the use of Qlikview is that it would not be possible to implement event streaming in a process mining application.

The sales order numbers the events relate to was chosen as the event case notion. The methodology proposed in section 3.4.1 and design specification in section 4.2.1 show how the single case notion was identified and implemented in the event logs. While this was successfully implemented for the bespoke purchasing process seen in Figure 2, there are critical prerequisites that must be highlighted as limitations. Combining O2C and P2P processes, using SAP event data, is typically not advisable due to the resulting convergence and divergence that is likely to occur (van der Aalst; 2019). This is because as purchase orders are created and processed for sales orders, they often create multiple documents relating to the same original sales order. The business process used in this research was selected for one important reason. Unlike other purchasing processes in the business, the software purchasing process is atomic by nature. Only one purchase order is generated for each software sales order. In like manner, there would only be one goods receipt document or invoice document for each sales order. Therefore, using a single case notion for these events does not increase the risk of losing critical data in the later sections of the process. Additionally, convergence and divergence are naturally reduced in the process. While this is an ideal scenario for this research, it would serve as a major limitation to the methodology if applied to other purchasing processes where implementing a single case notion is not feasible.

The benefits of the clustering methods presented in sections 3.4.3 and 4.2.3 are that each type of event activity label can be analyzed for the value it brings to the final process model. As mentioned earlier, the form-based nature of event data from SAP results in multiple events being created for each field changed in a transaction. Each change is logged regardless of whether or not the changes were made at the same time. By clustering events in this way, multiple events with the same general concept can be clustered into one overarching event. Additionally, where granularity is required, this method allows for the inclusion of critical activity notions to differentiate between events that can be clustered and activity notions of interest. One of the limitations to this method is the time and expert business process knowledge required to detect where the granularity of activities is unnecessary and where events should be considered as activity notions of interest. In fact, this method would only be applicable where the researcher has expert business knowledge available.

As discussed in section 5, the filter implemented using the ProM application was used to remove traces that occurred less than 5 times in the event log. While this successfully "cleaned" the data, removing traces from event logs in this way should not be conducted without carefully considering its implications. For example, the event traces filtered out by ProM as discussed in section 5, resulted in 515 event traces being removed from the event logs. Effectively accounting for over 16% of the event traces. While the fitness and precision metrics of the final model were promising, more experimentation could be completed to optimise the results.

In conclusion, this research explored methods for flattening event data extracted from the SAP 770 ERP system. Thus answering the research question proposed in the introduction. Additionally, the clustering and filtering methods applied to the event logs successfully demonstrated that using these methods can highlight predefined activities of interest in the final process model.

An area of future research would be to see if Qlikview could be used to periodically test for model conformance using the scheduled report functionality. Another important area of future research would be to develop an algorithm that can cluster based on activity notions and identify activity notions of interest without the use of domain knowledge. While there are algorithms for this proposed in the relevant literature this focus more on similarities between activity notions and timestamps rather than conceptual attributes. Finally, additional attributes were excluded from the event logs in this report due to the complexity they would add to the research. While research has been conducted in this area in the relevant literature, it would be interesting to research how the recent state-of-the-art methods for the use of additional attributes could be implemented in a case study based on SAP event data.

8 Acknowledgements

I would like to thank Muhammad Zahid Iqbal for his assistance and patience throughout this research.

References

Aalst, W., Adriansyah, A., Medeiros, A., Arcieri, F., Baier, T., Blickle, T., R.P., J. C. B., Brand, P., Brandtjen, R., Buijs, J., Burattin, A., Carmona, J., Castellanos, M., Claes, J., Cook, J., Costantini, N., Curbera, F., Damiani, E., de Leoni, M. and Wynn, M. (2011). Process mining manifesto, Vol. 99, pp. 169–194.

Accorsi, R. and Lebherz, J. (2022). *A Practitioner's View on Process Mining Adoption, Event Log Engineering and Data Challenges*, Springer International Publishing, Cham, pp. 212–240.

Berti, A., Park, G., Rafiei, M. and van der Aalst, W. M. (2022). An event data extraction approach from sap erp for process mining, *International Conference on Process Mining*, Springer, Cham, pp. 255–267.

Berti, A. and van der Aalst, W. M. (2022). Oc-pm: analyzing object-centric event logs and process models, *International Journal on Software Tools for Technology Transfer* pp. 1–17.

De Weerdt, J. and Wynn, M. T. (2022). *Foundations of Process Event Data*, Springer International Publishing, Cham, pp. 193–211.

Eck, M., Lu, X., Leemans, S. and Aalst, W. (2015). Pm^2: A process mining project methodology, pp. 297–313.

Fischer, D., Goel, K., Andrews, R., van Dun, C., Wynn, M. and Röglinger, M. (2022). Towards interactive event log forensics: Detecting and quantifying timestamp imperfections, *Information Systems* **109**: 102039.

Folino, F., Guarascio, M. and Pontieri, L. (2015). Mining multi-variant process models from low-level logs, *in* W. Abramowicz (ed.), *Business Information Systems*, Springer International Publishing, Cham, pp. 165–177.

Leemans, S., Fahland, D. and Aalst, W. (2015). Exploring processes and deviations, Vol. 202, pp. 304–316.

Leemans, S. J., Poppe, E. and Wynn, M. T. (2019). Directly follows-based process mining: Exploration a case study, *2019 International Conference on Process Mining (ICPM)*, pp. 25–32.

Mishra, V. P., Dsouza, J. and Elizabeth, L. (2018). Analysis and comparison of process mining algorithms with application of process mining in intrusion detection system, *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 613–617.

Muñoz-Gama, J. and Carmona, J. (2010). A fresh look at precision in process conformance, *in* R. Hull, J. Mendling and S. Tai (eds), *Business Process Management*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 211–226.

Pajić, A. and Bečejski-Vujaklija, D. (2016). Metamodel of the artifact-centric approach to event log extraction from erp systems, *Int. J. Decis Support Syst. Technol.* **8**(2): 18–28.

Rehse, J.-R. and Fettke, P. (2019). Clustering business process activities for identifying reference model components, *in* F. Daniel, Q. Z. Sheng and H. Motahari (eds), *Business Process Management Workshops*, Springer International Publishing, Cham, pp. 5–17.

ROEST, A. H. (2012). Process mining on erp systems–the case of sap order to cash.

Suriadi, S., Andrews, R., ter Hofstede, A. and Wynn, M. (2017). Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs, *Information Systems* **64**: 132–150.

van der Aalst, W. M. P. (2019). Object-centric process mining: Dealing with divergence and convergence in event data, *in* P. C. Ölveczky and G. Salaün (eds), *Software Engineering and Formal Methods*, Springer International Publishing, Cham, pp. 3–25.

van Zelst, S. J., Mannhardt, F., de Leoni, M. and Koschmider, A. (2021). Event abstraction in process mining: literature review and taxonomy, *Granular Computing* **6**(3): 719–736.