TÉCNICO LISBOA

Process Control-Flow Discovery

Hugo Martins Dias Simões Monteiro

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. José Luís Brinquete Borbinha
Eng. Nuno Dias

Examination Committee

Chairperson: Prof. Luís Manuel Antunes Veiga
Supervisor: Prof. José Luís Brinquete Borbinha
Member of the Committee: Prof. Diogo Manuel Ribeiro Ferreira

October 2021

Acknowledgments

First, I have to thank my research supervisor, Prof. José Luís Brinquete Borbinha. Without his assistance and dedicated involvement in every step throughout the process this dissertation would have never been accomplished.

A word of thanks to Nuno Dias from Border-Innovation for seeing this opportunity as a possible work for a thesis and for his help as being involved in the process and for supporting me through all this time.

Also thanks to Border-Innovation for enabling this data to be used and for enabling almost a year of Business Splunk Flow software license.

I should also thank all my friends, for making me believe that i could end this in time and for supporting me all this time.

Most importantly, none of this could have happened without the support of my parents. I must express my gratitude to my parents for providing me with continuous support and encouragement throughout my years of study and through the process of writing this thesis. This accomplishment would not have been possible without them. Thank you.

Abstract

Process mining focuses on the discovery and analysis of processes based on event logs that are recorded by the information systems in an organization. With more and more event data being recorded by such systems and with processes representing the core value of an organization, process mining becomes a fundamental and crucial task.

In this document first we start by present some of the tools used in Process Mining and how they use logs as input to apply discover and produce the respective process models. With focus on what transformations can be made to information.

We propose a solution using process mining to discover a specific process based on a ticketing system. For the implementation phase we used Splunk Business Flow a tool from Splunk big data platform capable of producing a process model and respective metrics from logs previously indexed. We used this tool since the project in case was implemented based on Splunk and we knew in this tool it would be possible to apply transformations to data used.

Keywords

Process; Process Mining; Process Discovery; Process Model; Event Log.

Resumo

Exploração de processo ("process mining" na literatura de língua inglesa) tem como foco a descoberta e análise de processos tendo como base registo de eventos que são registados pelos sistemas de informação de uma organização. Sendo que cada vez mais eventos são produzidos por estes sistemas e sendo que os processos representam o essencial de uma organização, Exploração de Processo torna-se uma tarefa fundamental e crucial.

Neste documento começamos por apresentar primeiro algumas das ferramentas usadas por Exploração de Processo e como usam o registo de eventos para aplicar descoberta e produzir os respectivos modelos dos processos. Com foco nas transformações que são possiveis fazer aos dados.

Propomos uma solução usando Exploração de Processo para descobrir um processo específico baseado num sistema de bilhetes. Para a fase de implementação foi usado Splunk Business Flow uma ferramenta da Splunk que é uma ferramenta de grandes dados capaz de produzir um modelo de processo e respetivas métricas de registo de eventos previamente colocados no Splunk. Usámos esta ferramenta porque o projeto em questão estava implementado em Splunk e sabiamos que com esta ferramenta seria possivel aplicar transformações aos dados usados.

Palavras Chave

Processo; Exploração de Processo; Descoberta de Processo; Modelo de Processo; Registo de Evento.

Contents

List of Figures

List of Tables

Listings

Acronyms

KPI	Key Performance Indicator
FFTH	Fiber to the Home
SLA	Service-level Agreement
SBF	Splunk Business Flow
PDO	Optical Distribution Point
SPL	Search Processing Language
BPD	Business Process Discovery
XML	Extensible Markup Language
CSV	Comma-Separated values

1

Introduction

Contents

Nowadays all organizations have their value services structured around processes with focus on different areas as for example manufacture products, purchase goods, handle applications and manage systems. This represents the core functionality of an organization and on this note organizations are getting more interested on studying and understanding the inner processes on which they rely for their business to get the most value out of it.

This services execute multiple times and generate millions of log events every day that can only be analyze with tools prepared to ingest large amounts of data and that can process them in real time or close to it. In order to bring value to an organization is necessary to use an efficient tool that enables extraction of relevant information as fast as possible and also offers different visualization methods to be possible to take conclusions from it.

In this challenge, the main focus was on processes known as sequence of steps that have an outcome and Process Mining which is a field of Computer Science that allows analysis like conformance checking (namely monitoring deviations between observed behavior and modeled behavior), discovery (learning a process model) and Extension of an organization's processes generated by existing information systems. Process Mining aims to help organizations navigate the business. That is, understand all possible routes, recommend the best path, calculate each journey time and other Key Performance Indicator (KPI) metrics. All this lead to continuous process improvement, provide insights, identify bottlenecks, anticipate problems, with direct adjusts on a service level meaning more productivity and less costs at the end. Process Mining has been successfully applied in a variety of domains like healthcare, electronic business to high-tech systems and auditing.

1.1 Problem and Motivation

The motivation was a project in a specific global organization with focus on measure and control Service-level Agreement (SLA) in the context of a Fiber to the Home (FFTH) project between two telecommunications organizations based on a ticketing system. This service works on a subscriber mode, meaning one organization hires fiber network from another organization in places where their customers fiber belongs to the other and vice-versa. Since this process is about a network installation, sometimes errors can occur related to the equipment and to solve this problem a fault ticket needs to be opened in order for the fault to be repaired. This faults can be of different types, which originates different paths from start to finish that we want to discover. Since the project was already implemented in Splunk Enterprise [1] the proposed solution is based on Business Flow tool from Splunk.

The main goal is to use process mining over real event logs by applying process discovery in order to get all different paths a ticket can follow. Once the respective process is known with all its steps, this

[1] https://www.splunk.com/

information can then be used as initial configuration for the engine and be updated automatically every time a change occurs. As off now this configuration is set and updated always by human intervention which is the main obstacle to overcome.

The main difficulties of this challenge are:

- Analyze logs efficiently and understanding which fields are the most important to use as input parameters for Splunk Business Flow (SBF) tool.
- Find differences that may exist and that represent possible discover errors from the normal flow of a ticket life-cycle.

1.2 Context of Work

In order to achieve the following goal and since the project was built in Splunk, we had to install Splunk Business Flow tool for the implementation phase in one of the older servers with a Splunk instance that had already the necessary data. This instance had two separate indexes one for raw data produced by each interaction between the two parts, as a result of each API call and another index with data produced by the project engine. In this server we could make any changes to the project configuration since it was a separate instance running Splunk from the one where the project was first implemented.

1.3 Results

During the implementation phase where Splunk Business Flow was used as the primary tool to process the logs and produce the corresponding process model, the software license expired during this period and we could not create a new one. When this happened the necessary results were already achieved and the graphs could be exported, however we could not go further in our analyzes and so it was not possible to execute a formal evaluation phase.

1.4 Structure of the Document

The remaining part of the document is organized as follows: in Chapter 2 Process Mining concepts and techniques will be described with some detail about process mining perspectives with focus on process discovery. In Chapter 3, we will describe related work carried out in the scope of Process Mining tools used to discover processes. In Chapter 4 we present the FTTH_SHARE project with details on different types of tickets and the dataset used as input for our work. In Chapter 5 we present our solution and how it was implemented. In Chapter 6 we present our results and experiments with data from FTTH_SHARE project by using Splunk with Business Flow tool. Finally, in Chapter 7 we include conclusions and a discussion about future work to do based on current limitations of the system used in our experiments.

2

Process Mining: Main concepts and techniques

Contents

In a world where data is growing exponentially, as a result of activities executed by people, machines and software, emerges a fundamental and very hard task, to analyze this data present in event logs and be able to get relevant benefits from it. Process mining emerges with its techniques using this logs as a way of discover, analyze and improve processes [6]. In this section the main concepts about process mining are described.

2.1 Event Log

The starting point for process mining is an event log, without data it is not possible to analyze processes and make improvements. Logs are made of events that represent activities, a well-defined step in some process as result of critical business actions by a certain resource. However this data comes in different formats as it depends on the system where it is produced from a database to a web service. This different formats can not be read by a process mining tool, so the first step is to transform this machine data that can come in any format to a format that each tool is able to read and to analyze.

For this to occur the event log as to follow a specific format containing the following fields:

- **Case ID**: ID that identifies the process instance, necessary to distinguish different executions of the same process. Can be seen as one run of the process [6].
- **Activity**: Each event is recorded as a result of an execution of human or automatic task, in the sense that it takes time. A group of related tasks form a process or a sub-process.
- **Timestamp**: Each task when executed has a timestamp, that helps keeping track of the order in which activities occur within a process. Can be used for bottleneck analysis.

events usually have additional fields that are captured based on the analysis to be made as event type, resource including human actors, organizations or software systems, that can be used for organizational mining and category as shown in Figure 2.1.

Once the fields are all captured, the next step is the transition from raw data to the events that represent the execution of activities in a process. Usually high variety of activities is the main source of problems, so the main goal is to be as precise as possible in order to take the most out of it. This transformations enable correct analysis for valuable insights regarding process improvement [10].

Case ID	Task Name	Event Type	Originator	Timestamp	Extra Data
1	File Fine	Completed	Anne	20-07-2004 14:00:00	...
2	File Fine	Completed	Anne	20-07-2004 15:00:00	...
1	Send Bill	Completed	system	20-07-2004 15:05:00	...
2	Send Bill	Completed	system	20-07-2004 15:07:00	...
3	File Fine	Completed	Anne	21-07-2004 10:00:00	...
3	Send Bill	Completed	system	21-07-2004 14:00:00	...
4	File Fine	Completed	Anne	22-07-2004 11:00:00	...
4	Send Bill	Completed	system	22-07-2004 11:10:00	...
1	Process Payment	Completed	system	24-07-2004 15:05:00	...
1	Close Case	Completed	system	24-07-2004 15:06:00	...
2	Send Reminder	Completed	Mary	20-08-2004 10:00:00	...
3	Send Reminder	Completed	John	21-08-2004 10:00:00	...
2	Process Payment	Completed	system	22-08-2004 09:05:00	...
2	Close case	Completed	system	22-08-2004 09:06:00	...
4	Send Reminder	Completed	John	22-08-2004 15:10:00	...
4	Send Reminder	Completed	Mary	22-08-2004 17:10:00	...
4	Process Payment	Completed	system	29-08-2004 14:01:00	...
4	Close Case	Completed	system	29-08-2004 17:30:00	...
3	Send Reminder	Completed	John	21-09-2004 10:00:00	...
3	Send Reminder	Completed	John	21-10-2004 10:00:00	...
3	Process Payment	Completed	system	25-10-2004 14:00:00	...
3	Close Case	Completed	system	25-10-2004 14:01:00	...

Figure 2.1: Event Log with the required attributes case ID, activity, timestamp for Process Mining (figure from [1])

2.2 Business Process Management System and Business Process

Enterprises can take even more benefits from this information if they use software systems for coordinating the activities involved in business processes, called business process management systems [4]. This type of system is driven by explicit process representations to coordinate the enactment of business processes to take advantage of process improvements including reducing costs with lower execution times resulting in lower error rates, but also gaining competitive advantage through innovation as described in [11].

A business process as defined in [4] consists of a set of activities that are performed in coordination in an organizational and technical environment. These activities jointly realize a business goal and bring value to the organization. Each process is enacted by a single organization, but it may interact with processes performed by other organizations. Examples of typical processes are:

- **Order-to-cash**. This is a type of process performed by a vendor, which starts when a customer submits an order to purchase a product or a service and ends when the product or service in question has been delivered to the customer and the customer has made the corresponding payment [11].
- **Quote-to-order**.This type of process typically precedes an order-to-cash process.It starts from the point when a supplier receives a Request for Quote (RFQ) from a customer and ends when the customer in question places a purchase order based on the received quote [11].

- **Procure-to-pay**.This type of process starts when someone in an organization determines that a given product or service needs to be purchased. It ends when the product or service has been delivered and paid for [11].

- **Issue-to-resolution**.This type of process starts when a customer raises a problem or issue, such as a complaint related to a defect in a product or an issue encountered when consuming a service. The process continues until the customer, the supplier, or preferably both of them agree that the issue has been resolved [11].

The desired behaviour for a process [5] can be further represented in a so called process model that illustrates the work done step by step by representing the flow from a starting point all the way to the end. This includes data, rules, documents, process participants, customers and other important factors. These models are typically used for communication within the organization, to facilitate process understanding and collaboration between people helping an organization reach its business goals [4]. In order for that to happen, models need to be well-defined to identify what tasks are most important, improving efficiency, preventing chaos from creeping into day-to-day operations and standardize a set of procedures to complete tasks that really matter[1].

To represent a process model there are very different techniques some of the most used are:

- **BPMN**: Provides an end-to-end graphical notation as shown in fig. 2.2 for processes with focus on control-flow by using graphical elements as flow objects, connecting objects, swimlanes and data [12]. The three basic elements are events, activities, and arcs.Events are represented by circles, activities by rounded rectangles, and arcs (called sequence flows in BPMN) are represented by arrows with a full arrow-head [11]. Each organization can run the same process multiple times a day making multiple instances run simultaneously. Tokens help keep track of each instance progress, by flowing through the all process. The main focus is to help process management with focus on representing the correct flow to enable all stakeholders an easy way to understand the process.

  At a more deeper level, it's targeted at the people who will implement the process, giving sufficient detail to enable precise implementation. It provides a standard, common language for all whether technical or non-technical: business analysts, process participants, managers and technical developers, as well as external teams and consultants. Ideally, it bridges the gap between process intention and implementation by providing sufficient detail and clarity into the sequence of business activities.

[1] https://kissflow.com/workflow/bpm/business-process/

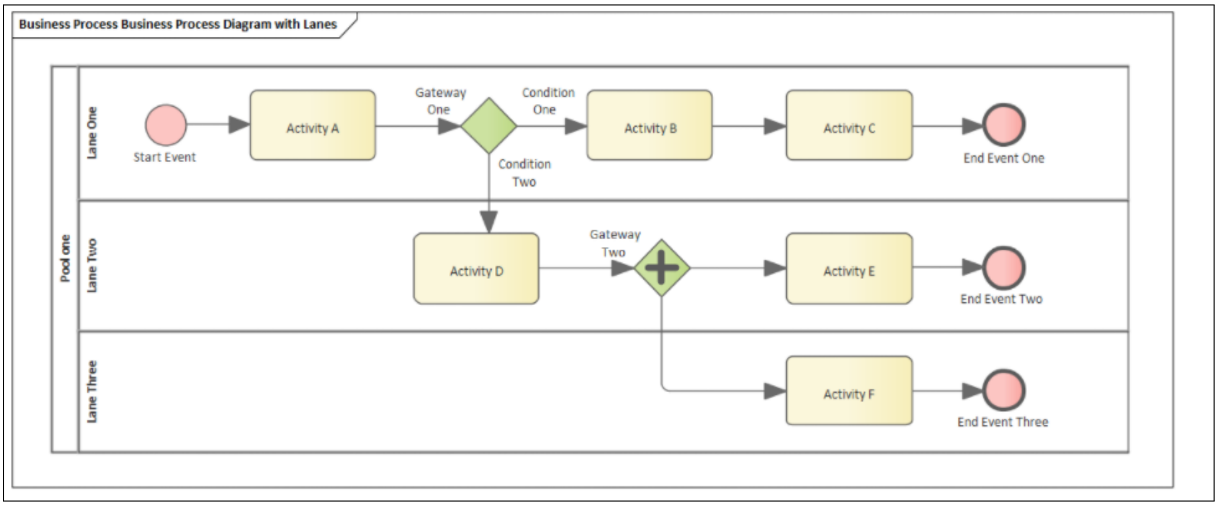


Figure 2.2: Business Process Model Notation example (figure from [2])

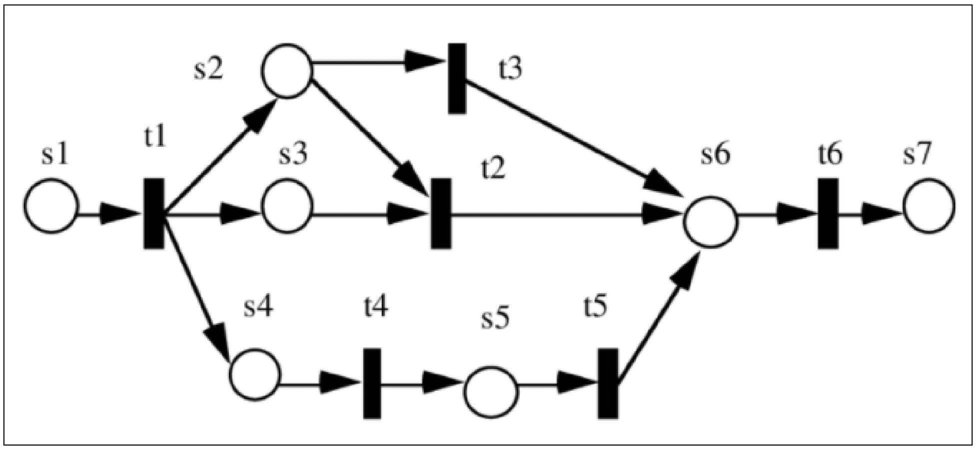


Figure 2.3: Petri Net Graph example (figure from [3])

- **Petri Net**: One of the first notations that was created in order to deal with concurrency and parallelism between activities [13] within a process. The basic elements of Petri Net graphs as shown in fig. 2.3 are called places and transitions, the nodes are connected via directed arcs.Places are represented by circles and transitions by bars. The number of tokens may change during the execution of the net.

As described before process models represent all the possible paths a process can take and each execution results in a sequence of tasks representing a process instance or case. This brings the possibility to analyze each execution isolated and track the most common paths that either result in errors or failures usually related to non-completed processes or successes. From this analysis is possible to compare how the process was suppose to be executed versus reality.

Process lifecycle shown in fig. 2.4 consists of phases related to each other, organized in a cyclical structure showing their logical dependencies.

Design and Analysis phase is the first step, in which surveys on the processes and their organizational and technical environment are conducted. From this surveys, processes are validated and represented by the corresponding models. Once this representation is finalized it needs to be validated through a workshop, during which the persons involved discuss the process. Also in this phase simulation techniques can be used to support validation and to check whether the process actually exposes the desired behaviour.

Once the process model is designed and validated, the process needs to be implemented. During the **Configuration phase** the dedicated software system needs to be defined according to the organizational environment on the enterprise and processes whose enactment it should control. Depending on the information infrastructure, the process configuration phase might also include implementation work, for instance integrating legacy software systems to the business process management system.

Once the configuration phase is completed, process instances can be enacted. The process **Enactment phase** covers the actual run time of the process. Instances are initiated to fulfil the business goals of an organization. Initiation of an instance usually is triggered by a defined event , for example the receipt of an order sent by a customer. Detailed information about the current state of process instances are available in the system as this type of information is valuable for instance, to respond to a customer request about the current status of his case. During this phase, execution data is collected in log files, indicating events that occurred during the process with information related to start and end of each one.

The **Evaluation phase** uses information available to evaluate and improve process models and their implementations. Execution logs are also evaluated in order to identify the quality of processes and the adequacy of the execution environment. For example, if a certain activity takes too long due to lack of resources required to conduct it.

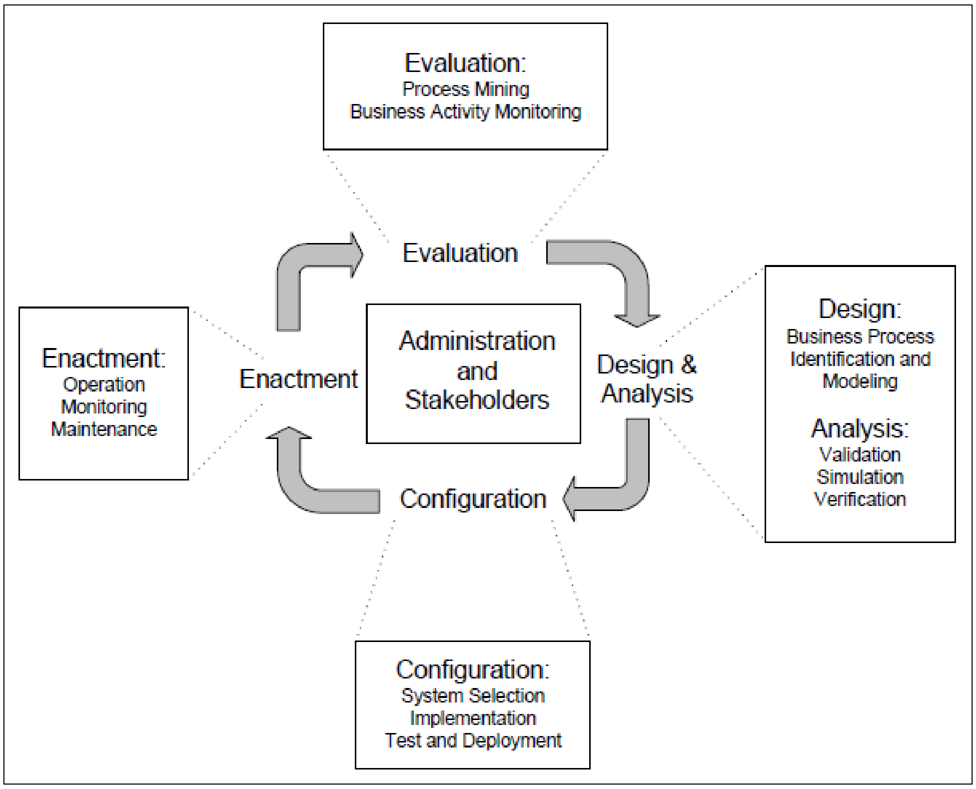


Figure 2.4: Process lifecycle (figure from [4])

2.3 Process Mining Purposes

As the number of events recorded increase day by day the need to analyse this critical information also increases in order to provide detailed insights about the most important processes. To improve and support processes in rapidly changing and competitive environments, process mining raises as a technique focused on extracting information from event logs and is the bridge between process analysis and data analysis.

There are three fundamental types of process mining as shown in fig. 2.5:

1. **Discovery**: This technique takes an event log and produces a process model without using any a-priori information. As an example the Alpha-algorithm takes an event log and produces a process model (Petri net) explaining the behavior recorded in the log. If the event log contains information about the resources then the produced model contains the interactions between the process stakeholders.

2. **Conformance**: An existing process model is compared with an event log of the same process. Conformance checking can be used to check the quality of documented processes, to identify deviating cases and understand what they have in common, to identify process fragments where most deviations occur, for auditing purposes, to judge the quality of a discovered process model, to guide evolutionary process discovery algorithms or as a starting point for model enhancement. There are three different approaches.

   The first is to create an abstraction of the behavior in the log and an abstraction of the behavior allowed by the model. An example of this is the notion of a footprint, a matrix showing causal dependencies between activities. The second approach replays the event log on the model many times until they are similar. The most advanced approach is to compute an optimal alignment between each trace in the log and the most similar behavior in the model. Conformance can be viewed in two ways, the model does not capture the real behavior or reality deviates from the desired model.

3. **Enhancement**: Aims at extending or improving an existing process model using information about the actual process recorded in the event log. In this case, the additional attributes can help with critical information used to analyze waiting times in-between activities, using resource information to discover roles or even construct social networks and analyze resource performance.

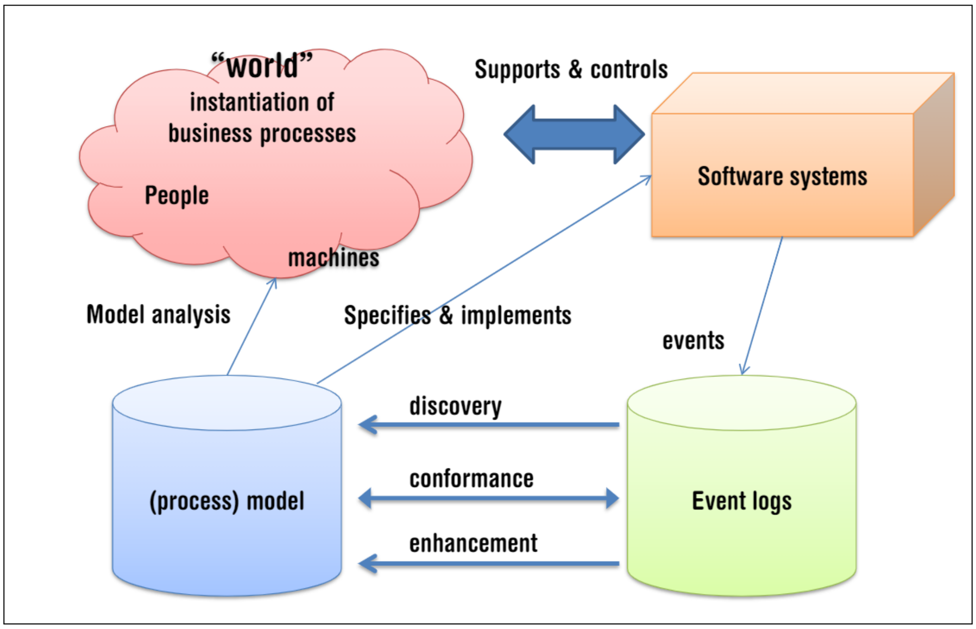


Figure 2.5: Process Mining perspectives in the context of Business Process Management life cycle (figure from [5])

Process mining is based on facts and so behavior recorded in event logs and the respective models. Using this information is possible to apply different types of analysis depending on what the goal is.

This way Process Mining can be related to four different perspectives:

1. **Control-Flow**: This perspective discovers the sequence of activities in a process and aims at finding a good characterization of all possible paths representing this information on one of the languages already described in section 2.2. Control-Flow algorithm can be used to discover the most frequent path of the process. The discovered process models can be used for discussing problems among stakeholders, for generating process improvement ideias, for model enhancement [6]. For this purpose there are different algorithms:

   (a) **α-Algorithm [14]**: Based on the causal relationships of tasks and their order, this algorithm scans the event log for particular patterns, each task that is followed by another originates one connection in the result Petri net as in fig. 2.6. The α-algorithm is simple and efficient, but has problems dealing with complicated routing constructs and noise.

   (b) **Heuristics-Miner [15]**: It is also based on the causal relationships of tasks, but it counts the number of times that each causal relationship is observed, in order to derive a dependency graph. This allows to correct some problems of the α-algorithm, such as dealing with noise (exceptional behavior) and low frequent behaviors, taking the frequency of those causal relationships into account.

   (c) **Genetic-Miner [16]**: Like the Heuristics-Miner, it can deal with noise and low frequent behavior, but it can also find non-local causal relationships (at the expense of some computational time). It is based on genetic algorithms.

   (d) **Fuzzy-Miner [17]**: It is a more recent process mining algorithm. It is able to deal with the 'spaghettilike' models by simplifying them, grouping less frequent and important tasks into clusters, based on some important and correlation metrics.

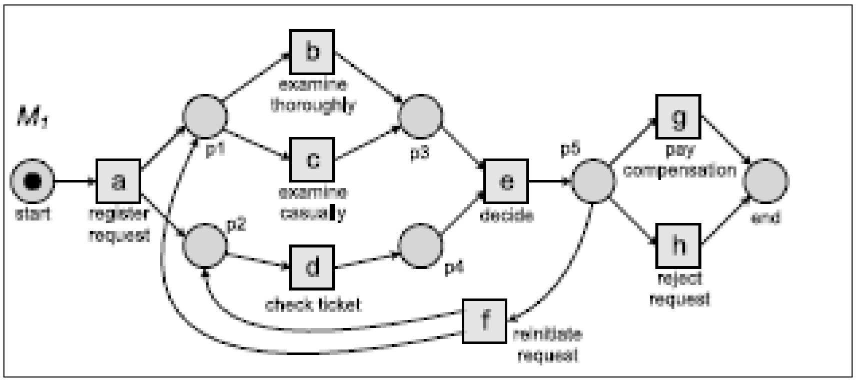


Figure 2.6: Graph produced using alpha algorithm (figure from [6])

2. **Organizational**:

This perspective focus on the participants involved in processes and the interactions between them. Looks to answer questions like what is the distribution of work, who does what, how do users interact and collaborate with each other, what is the performance of each user by activity and if any user is being overloaded. For this analysis the case ID and resource fields are the most important, in order to capture the participants within the same process instance. However, it is also possible to analyze the relation between resources and activities.

The main goal is to produce a graph showing interactions between resources [18] in different ways: **Handover of Work** if they perform consecutive tasks in a case, provides a view of the interactions between users showing the most recurring interactions as in fig. 2.7, workload on a user or user with a central role in the process. **Working Together** aims to analyze the shared cases between users, regardless of whether they have direct interactions or not being useful to discover group of users that often work together as a team. **Users and Activities** aims to analyze the number of shared activities, for each pair of users how many activities in common they are able to perform. **Work Distribution** aims to count the number of instances of each activity that was performed by each user.

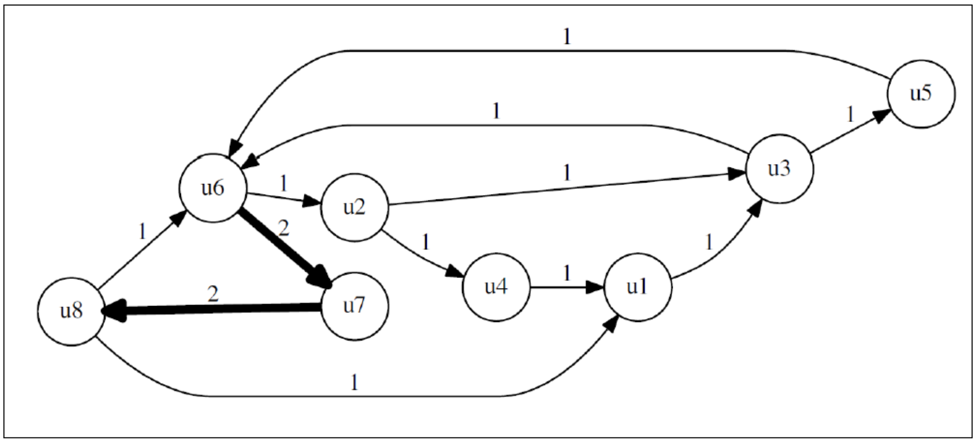


Figure 2.7: Graph produced using handover of work algorithm (figure from [7])

3. **Case**:

   This perspective focus on properties of a single process instance (case) [19]. A process instance can be characterized by its path, the users involved or the value of the data itself.

4. **Performance**:

   This perspective is mostly related with timing using the timestamp field to track duration of activities allowing to discover bottlenecks. The average time an activity takes to execute as represented in fig. 2.8 is a performance measure and makes it possible to compare the total time of different instances and predict times. Performance can also be related to financial reasons. It is possible to distinguish between **fixed costs**, related to use of infrastructure and the maintenance of software systems and **variable costs** correlated with some variable quantity. A cost notion which is closely related to productivity is **operational costs**, which can be directly related to the outputs of a process [11].

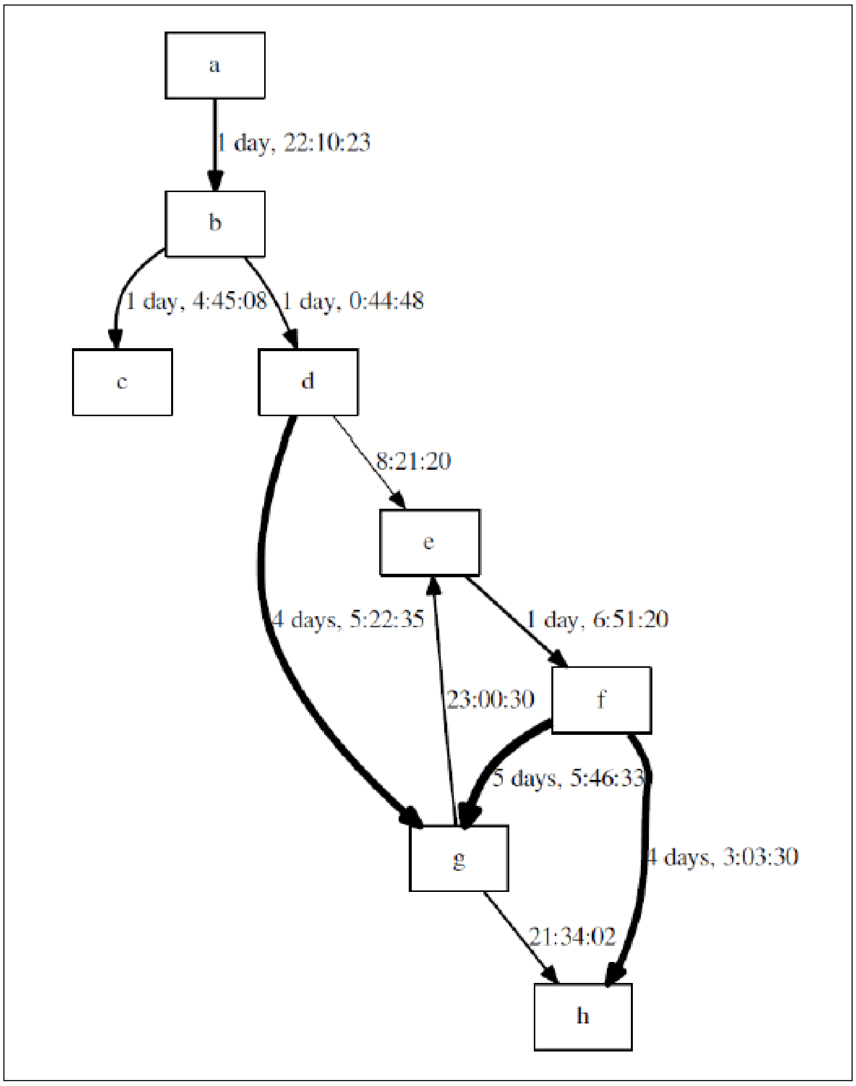


Figure 2.8: Graph showing timestamp difference between events (figure from [7])

2.4 Process Mining for Process Discovery

As previously described in section 2.3, process discovery is one of the most common process mining ways of producing a model from the information captured from different applications. Due to the extremely high volume of data that is available for each process, this activity is way more than process modelling. Using this information to identify bottlenecks, the basic causes of problems, deviations from the process, and the distribution of process events. In other words it captures what happens in reality and provides a meaningful simulation to enable process optimization. So the produced model is as accurate as the data linked to each event, containing in the best case starting time, ending time, resource information and data.

When it comes to process discovery different methods can be distinguished, **Evidence-Based Discovery** a method that focus on different evidence analysis, using document analysis for example process description, observation by following the execution of individual cases in order to understand how the process works or automated process discovery which is a method that uses event log to automatically discover a model of the process that is supported by software systems of an organization. **Interview-based discovery** a method that aims at interviewing domain experts to gather information about how a process is executed. The final method **Workshop-based discovery** also offers the opportunity to get a rich understanding of the process, with the advantage of resolving inconsistent views between domain experts more quickly than interviews.

Over the years many algorithms have been developed to address this problem, however the produced model is not 100% precise all the time. Is a trade-off between accuracy and complexity of the output [20]. Accuracy is measured in three different dimensions:

1. **Fitness**: How many paths of the input log, the output model is able to replay.
2. **Precision**: How many paths generated on the model match the input log paths.
3. **Generalization**: How many paths the model is able to reproduce despite not being present in the input log.

On the other hand, complexity is measured via size metrics, related to the number of total tasks represented in the model and the number of produced ramifications [21].

The main goal with process discovery is to help organizations discover which processes can be automated[2] bringing advantages:

- **Improve quality and performance** since it is much more accurate and ensures up-to-date process workflows, without any human intervention.

- **Visibility** for the entire process across the organization, finding new path ways and steps that were less relevant.

- **Less Risks** by checking if suggested changes should be done based on the process model.

- **Cost Efficiency** with less human resources needed, the automation costs go down significantly.

- **Improved Scalability** with fewer resources needed and less time consuming, by making intelligent decisions.

- **Competitive Advantage** versus the organizations that still use manual process discovery, with reduced costs and faster time to produce value.

During process discovery, many challenges can arrive[3]. The first challenge of process discovery relates to fragmented process knowledge [11], meaning different tasks are performed by different resources and a process analyst must collect all the information about a given process by talking with different domain experts who are responsible for each different task. This challenge is the reason why process discovery requires several iterations.

The next challenge relates to the fact that domain experts typically think of processes on a case level, they can describe their task in the context of a specific process instance but they do not understand how the process works in general. The Last challenge relates to the fact that domain experts usually are not familiar with process modeling languages. This challenges can bring difficulties when producing process models and can originate unstructured processes as in fig. 2.9 (also called spaghetti processes). One way to simplify this complex processes is filtering out the event log, either by selecting a group of resources or by selecting the most common activities.

[2]https://www.edgeverve.com/assistedge/process-discovery/
[3]https://research.aimultiple.com/process-discovery/

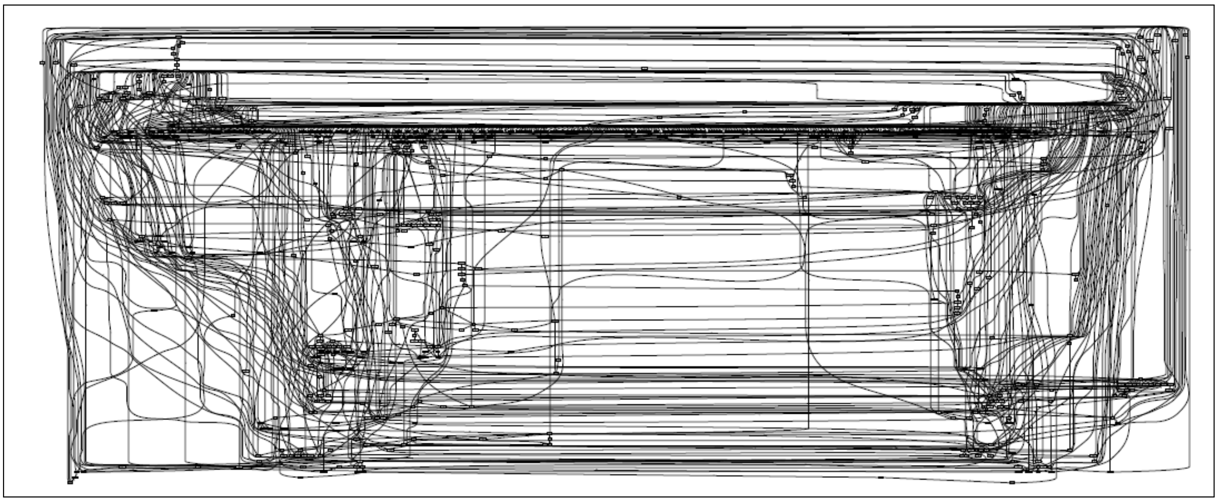


Figure 2.9: Spaghetti process model constructed based on an event log containing 114,592 events (figure from [6])

Using historical data might originate differences between reality and the process generated, find a balance between rare process instances and the important ones and making sure the input log always contains the fundamentals fields as previously mentioned.

3

Overview of Process Mining Tools

Contents

This section presents an overview of different process mining tools and how they transform event logs and normalize the data to apply process discovery algorithms. This tools were studied so that our proposed solution can address problems encountered in them and provide additional features. Based on this and after search in different sources, each tool can be classified depending on the level of expertise necessary in three main categories: open-source, academic or commercial. The differences are in terms of maturity and capabilities depending if they are for commercial purposes or developed by research groups [22].

3.1 Open-Source: ProM

ProM is an open-source framework that allows the interaction between several plug-ins. Each plug-in representing an implementation of an algorithm that can be added separately with no need to modify the framework by adding only the corresponding file [23].

This tool can read files in different formats like Extensible Markup Language (XML) through the log filter component that deals with large data sets and sorts the events within a case on their timestamps before processing the data. After this the mining plug-ins are applied and the result is stored in memory. This framework also allows to use multiple plug-ins one after the other, so the output of a plug-in can be used as input to another plug-in, fig. 3.1 showing configuration for filter plug-in. In the end, the mining result should be presented in a visualization which can be applied a transformation plug-in in case a different format is necessary. ProM is based on Petri nets, since is possible to prove whether a Petri net has deadlocks or non-executable paths and allowing precise analysis leaving no doubt or ambiguity in how the process will execute.

Figure 3.1: ProM - Configuring a filter plug-in in ProM

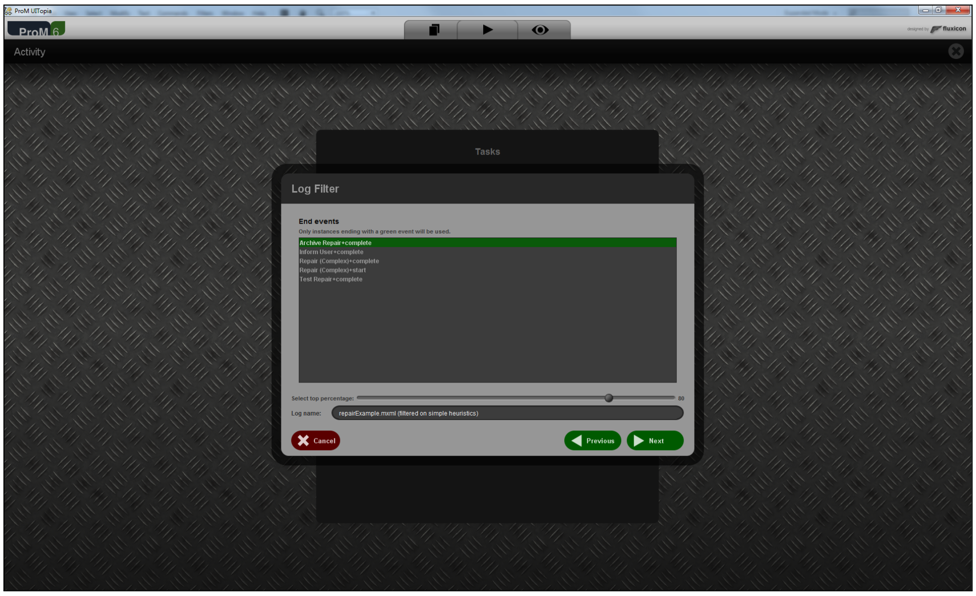


Figure 3.2: Log Filter - end event (figure from [8])

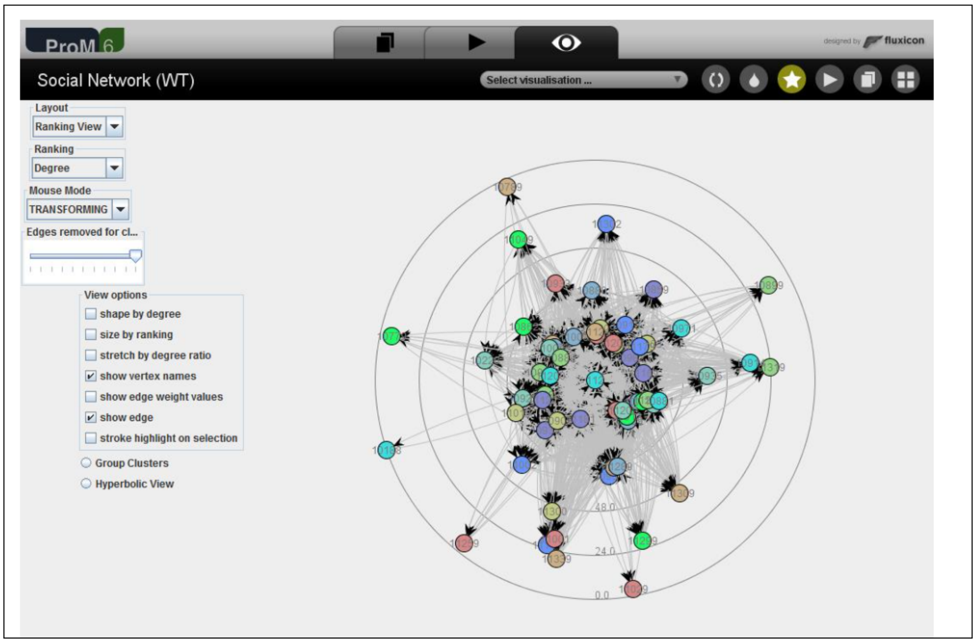


Figure 3.3: ProM - Organizational Perspective view

In the pre-processing phase is possible to choose the start and end activities to filter out the running processes as in fig. 3.2, with this we have a more clean look at the instances that completed and the path they took[1]. Also, if needed is possible to filter out specific tasks or even pick each event separately resulting in fewer process cases. This initial filters have some limitations since is not possible to rename tasks or events in a log.

All different Process Mining perspectives can be applied, as in fig. 3.3 social networks can be produced with the ability to group nodes by cluster and represent node size according to their weight.

3.2 Commercial

3.2.1 ARIS

ARIS is a product of Software AG[2] that focus mainly on monitoring performance through process instance analysis using different visualizations in dashboards.

This tool gives the possibility to see a potential model of each executed process instance, all done automatically and afterwards the option to save it in a repository[3]. The main focus is to replicate the reality as close as possible as we can see in fig. 3.4.

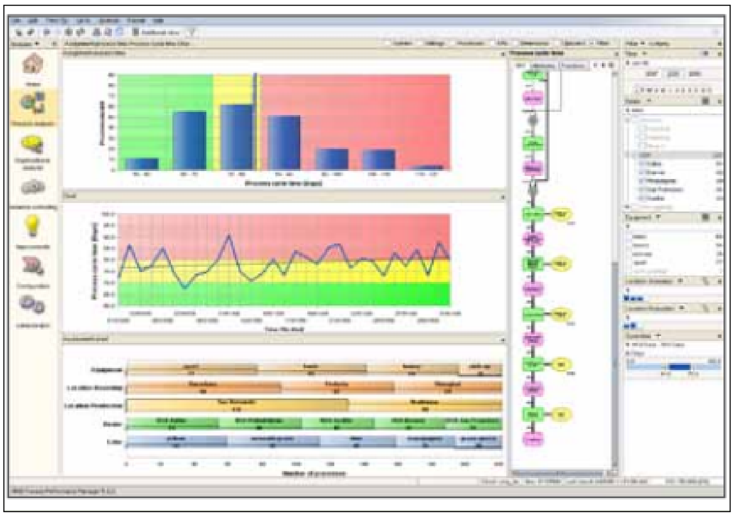
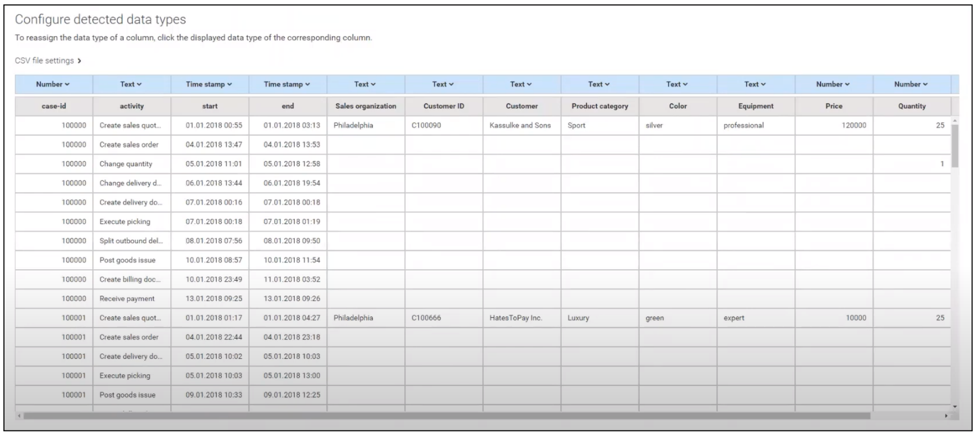


Figure 3.4: ARIS - Process discovery

[1] https://www.promtools.org/doku.php?id=tutorial:preprocessing
[2] https://www.softwareag.com/en_corporate.html
[3] https://www.ariscommunity.com/users/tobias-blickle/2009-06-10-automatic-process-discovery-aris-ppm-it-s-about-behavior



Configure detected data types

To reassign the data type of a column, click the displayed data type of the corresponding column.

CSV file settings ›

Number	Text	Time stamp	Time stamp	Text	Text	Text	Text	Text	Text	Number	Number
case-id	activity	start	end	Sales organization	Customer ID	Customer	Product category	Color	Equipment	Price	Quantity
100000	Create sales quot...	01.01.2018 00:55	01.01.2018 03:13	Philadelphia	C100090	Kassulke and Sons	Sport	silver	professional	120000	25
100000	Create sales order	04.01.2018 13:47	04.01.2018 13:53								
100000	Change quantity	05.01.2018 11:01	05.01.2018 12:58								1
100000	Change delivery d...	06.01.2018 13:44	06.01.2018 19:54								
100000	Create delivery do...	07.01.2018 00:16	07.01.2018 00:18								
100000	Execute picking	07.01.2018 00:18	07.01.2018 01:19								
100000	Split outbound del...	08.01.2018 07:56	08.01.2018 09:50								
100000	Post goods issue	10.01.2018 08:57	10.01.2018 11:54								
100000	Create billing doc...	10.01.2018 23:49	11.01.2018 03:52								
100000	Receive payment	13.01.2018 09:25	13.01.2018 09:26								
100001	Create sales quot...	01.01.2018 01:17	01.01.2018 04:27	Philadelphia	C100666	HatesToPay Inc.	Luxury	green	expert	10000	25
100001	Create sales order	04.01.2018 22:44	04.01.2018 23:18								
100001	Create delivery do...	05.01.2018 10:02	05.01.2018 10:03								
100001	Execute picking	05.01.2018 10:03	05.01.2018 13:00								
100001	Post goods issue	09.01.2018 10:33	09.01.2018 12:25								

Figure 3.5: ARIS - configuration step

Model analysis is possible through "slicing and dicing" at the process level [22]. This tool is capable of dynamically generating an aggregated process view for each query. The structure can also be visualized as a Gantt-chart to comprehend easily the sequence and overlap of the activities in the process. This is specially suited for the detection of waiting times within a process.

This tool prior to process discovery focus mainly on finding a way to group the events that belong to the same instance, by making rules to connect events and the corresponding fragments of a process. Some conditions must be met to link two fragments together. ARIS PPM supports the import of data in arbitrary order, as it keeps track of all events imported at any time [24].

To produce a model only need a file containing a process ID, activity name and timestamp. With this tool is only possible to import a file and preview the data by choosing each field data type[4] as in fig. 3.5, after that we need to chose the timestamp field and the case id to correlate the events and no other transformation is possible prior to mining step.

3.2.2 Disco

Disco is a tool from Fluxicon[5] made to speed up process mining by making data import really easy through automatically detecting timestamps, remembering configuration settings, and fast loading data sets [25]. Import step can be done by simply loading a csv or excel file and choosing which fields hold the caseID, activity, timestamp, resource and other as in fig. 3.6. This data sets are imported in read-only mode so the original files cannot be modified. This tool is fully compatible with ProM so analysis can be made in parallel. Supports control-flow shown in fig. 3.7 with the ability to filter by task or path absolute frequency and performance perspectives and has limited support for organizational perspective.

[4] https://resources.softwareag.com/youtube-aris-webinars/immediate-insights-into-your-processes-with-aris-process-mining-3

[5] https://fluxicon.com

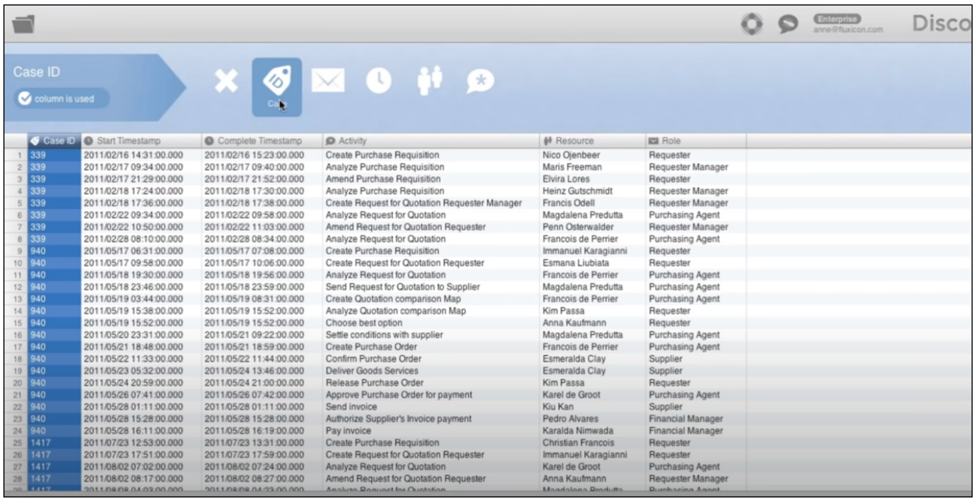


Figure 3.6: DISCO - Import configuration screen

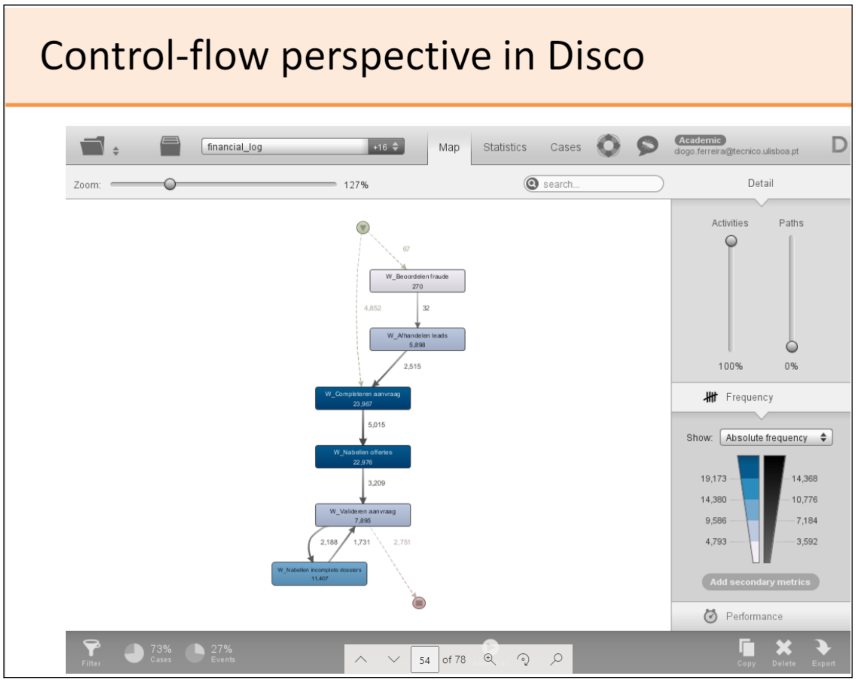


Figure 3.7: DISCO - Control-flow perspective applied in Disco

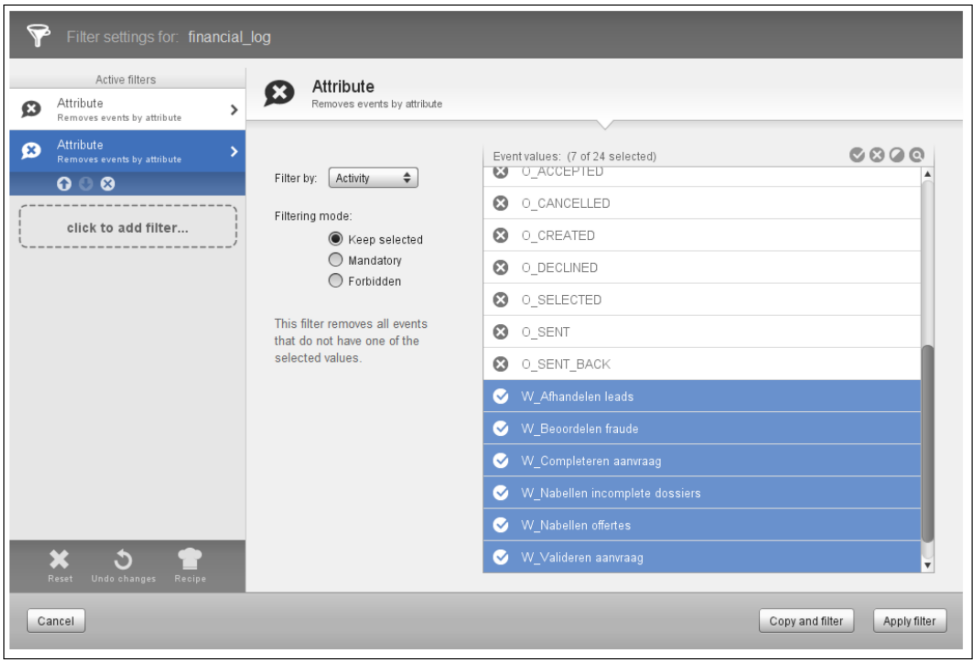


Figure 3.8: DISCO - Filter applied to task name

The input log file requires one event per line and at least one caseID as well as a timestamp field and events will be imported in the order in which they appear in the file[6]. Disco starts by guessing what each column might mean, but it is possible to adjust configuration before the import proceeds by using the preview mode. For each column is possible to ignore, set as the caseID, set as activity name, set as timestamp, set as resource or as an additional attribute[7]. Configuration also allows to choose between different timestamp patterns, to match the format in the file in case it is not standard. Multiple column attributes can be added as well as multiple timestamp columns. Disco allows also importing data sets in standard event log formats such as MXML and XES.

Different filters can be applied, for example activities can be filter to only show events that contain critical tasks for process analysis fig. 3.8.

3.2.3 Minit

Minit[8] is a Business Process Discovery (BPD) tool that works with data from different information systems as ERP, CRM, BPM, Marketing Automation Systems and Project Management.Allows import formats as CSV, XEX, MXML, SQL Server, Excel, Access and ODBC[9]. Also, if a process spans across

[6]https://fluxicon.com/disco/
[7]https://fluxicon.com/book/read/import/#importing-data-sets/
[8]https://www.minit.io
[9]https://www.minit.io/blog/how-to-prepare-your-data-for-a-process-mining-project

several different systems, the respective logs are combined to create a complete picture of a process.

Minit since version 4.5 allows to choose data type for each column as in fig. 3.9, detects and offers suggestions but is possible to change it. The available types are string, float, integer, duration and date[10]. Is also possible to choose decimal point and thousands separator. A great feature is the possibility to define a default to replace empty values in any column.

When importing a log file is possible to preview the data while choosing data encoding and delimiter in case of Comma-Separated values (CSV). Next step is to pick the attributes caseID, activity and timestamp necessary for process mining algorithms to work[11]. After this step, is possible to review the import settings as in fig. 3.10 with a new column called "attribute level" that shows if the corresponding field is event or case level. Finally, is possible to validate the settings using either full validation matching with the full log or without validation matching only the three fundamental attribute columns, pick a process name and after this if no validation errors occur is ready to use.

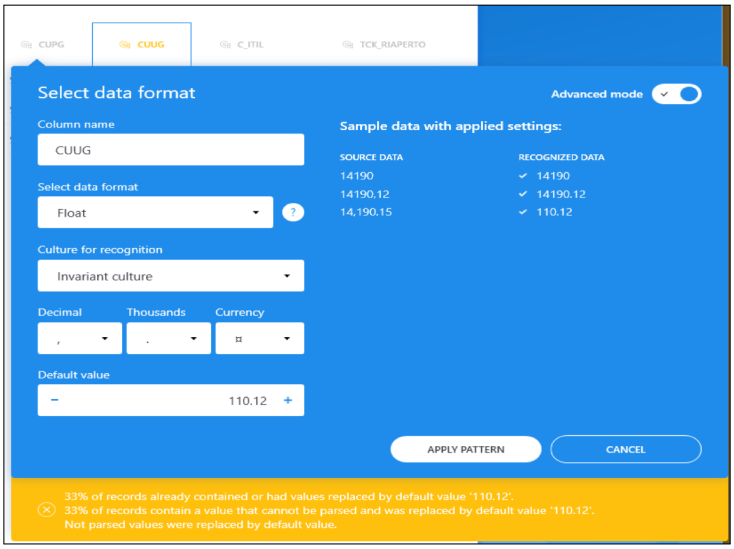


Figure 3.9: MINIT - Import configuration screen

[10]https://medium.com/minit-process-mining/minit-4-5-is-out-heres-a-rundown-of-what-s-new-5333314627fa
[11]https://www.youtube.com/watch?v=NfOuFtXi10A

Figure 3.10: MINIT - Import review settings

3.2.4 Celonis

Celonis[12] is one of the marketing leaders in Process Mining and has a process cloud based solution called "The Intelligence Business Cloud" and has released "Celonis Snap" the first free cloud process mining platform[13]. With this tool is possible to connect with new sources of data like Servicenow or Google Sheets platform or use the traditional format CSV.

This tool works with Data models as source of data, which are collections of tables used for the same purpose of analysis[14].When choosing a log file to import the data type for each attribute can be changed and the timestamp format can also be defined. The three main columns need to be defined as in any process mining tool as in fig. 3.11 and after uploading the data model with new data the tool is ready to use with no further configuration needed.

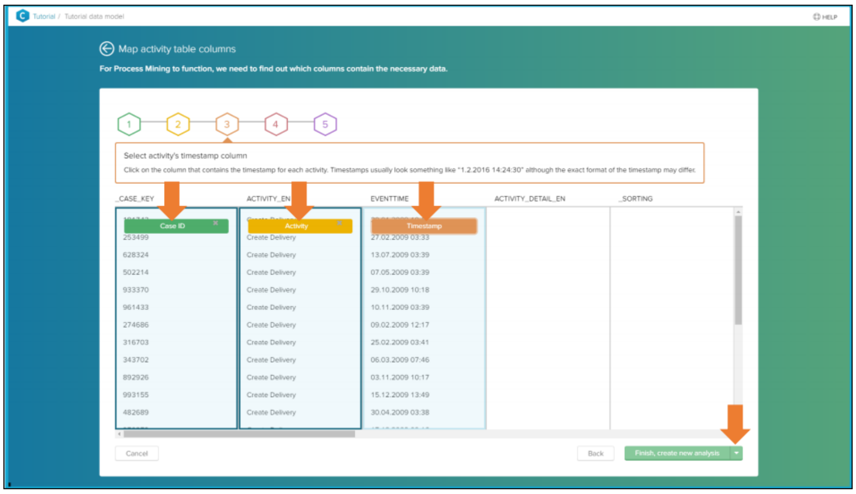


Figure 3.11: CELONIS - Import settings

[12] https://www.celonis.com
[13] https://www.youtube.com/watch?v=-EQcqxeYUMg
[14] https://support.celonis.de/display/CPM4E/Datamodels

3.2.5 bupaR

Another BPD tool is bupaR[15], an open-source integrated suite of R-packages for the handling and analysis of process data [26]. Here we focus on the main package "bupaR" for creating event log objects. Data transformations are allowed, as sometimes data will not come at the desired format[16]. The common issues are:

- **Lack of transitional lifecycle -** It happens when a single timestamp is recorded for each activity instance equivalent to an event. Normally it is the completion of a task that is recorded, in this case the other transition can be added manually.
- **Lack of resources -** Many of the functions in this tool require the resource attribute, in order to perform organizational and performance analysis. In this cases, the easiest solution they use is including an empty resource variable.
- **Activity log -** This final issue happens when instead of a list of events, there is a list of activity instances available like in fig. 3.12. When this is the case, first an unique id needs to define the activity instances, and subsequently by gathering the different timestamp columns.

```
## # A tibble: 5 x 5
##   patient  activity  schedule            start               complete
##   <chr>    <chr>     <dttm>              <dttm>              <dttm>
## 1 John Doe check-in  NA                  NA                  2017-05-10 08:33:26
## 2 John Doe check-out NA                  NA                  2017-05-11 14:52:36
## 3 John Doe surgery   2017-05-10 08:38:21 2017-05-10 08:53:16 2017-05-10 09:25:19
## 4 John Doe surgery   NA                  2017-05-10 10:41:35 2017-05-10 11:05:56
## 5 John Doe treatment NA                  2017-05-10 10:01:25 2017-05-10 10:35:18
```

Figure 3.12: BUPAR - log missing events

[15]https://www.bupar.net
[16]https://www.bupar.net/creating_eventlogs.html

4

Problem Analysis

Contents

The project in case is supporting a fault based ticketing system, based on full manual process configuration of its steps and corresponding transactions. The project and functionality will now be described.

4.1 Overview of Project: FTTH_SHARE

The goal of the project was to measure and control SLA in the context of the FTTH_SHARE project, which is a joint FFTH project between two telecommunications organizations. All types of interactions in the context of FFTH sharing process are captured by logging all service executions, including input and output parameters at the Middleware platform level to be collected in Splunk for further analyze.

This interactions when related enable the analyzes of the time it takes to resolve a technical failure called SLA. This calculations can become very hard when dependencies happen, like process steps that are not relevant for the final calculations or repetition of steps that represent process internal cycles.

This type of agreements between the two parts have to be respected because exceeding limits result in very expensive penalties per ticket. This solution will give a quick overview over data to gain vision on given penalties.

In this case, the project was already built in Splunk which is a big-data platform used mainly for processing logs, that enables data integration from different sources in real time and build fast searches to be used in reports,dashboards and generate threat detection alerts. In order to facilitate writing queries it has its own processing language so that a higher level of abstraction is provided to users. Splunk is composed by three main components: forwarder, which forwards the data to the remote instances; indexer, which stores the data and answers the searches; search head, which is the front-end usually accessed through a web interface. In this project the different knowledge objects (lookups, dashboards, saved searches) exist in the search head grouped under an app that includes an engine that categorizes all collected events previous to visualization. The main goal with this engine is to separate data processing from visualization.

The different dashboards available in the application allow for an analysis on points such as:

- The evolution of average SLA time.
- Evaluate most common types of technical failures.
- Understand most common resolution types.
- Search for open tickets per fiber provider.

4.2 Ticketing System and State Diagram

During the process life-cycle, when network sharing is contracted between two parties problems emerge and it is normal that technical faults occur. In order to communicate the problem, the beneficiary opens an incident and the provider after analysis proceeds with the resolution process. A common way to implement this kind of systems is based on tickets.

Tickets are created bidirectionally between a provider and a beneficiary to represent a flow of interactions based on request and response messages. In this case, there will be described only trouble tickets that represent among other processes, urgent and non-urgent fault reparations.

SLA measures the maximum time to repair for these types of faults. Different outcomes are possible like closed with success meaning the SLA was closed and the business operation was concluded, closed with error when the SLA was closed and the business operation was not concluded, open when the SLA is running and cancelled when the SLA is cancelled.

They represent the different API use cases and differ depending on the type of fault. In some cases exceptions can occur in case the SLA is not respected. In this case, an expected time of resolution is necessary and the reason for the respected violation, following updates every 12 hours are also necessary unless justified delay. Time of resolution will also depend if the fault is located on a solo Optical Distribution Point (PDO) or located on the network itself.

The implemented fault processes were:

- **"Avaria em Acesso de Cliente"**

  This fault process refers to an unique PDO or in point to point solutions to an unique fibre. Should be initialized by the beneficiary operator when a problem is detected in client's access, after performing remote testing. The beneficiary, when opening the ticket should provide information for the necessary controls to determine the nature of the problem.

  The flow shown in fig. 4.1 , starts with a create ticket operation by the beneficiary, with the necessary information as client access id, type and source of fault, urgency level and observation notes if existent. If accepted then ticket is pending for acceptance otherwise the response is failure and ticket is rejected.

  After notifyTicket is called and the beneficiary is informed about the resolution type, if success then the ticket can be marked as resolved and can reopen in the next few hours depending if it is urgent or not. The next update and notify interactions represent possible joint interactions as a result of a reopened ticket.

- **"Avaria em Serviço Cliente Detectado em Provisão"**

  This process should be used only when the beneficiary operator is doing his own client installation and can not be concluded because of a fault in one of the PDO ports.

  This type of fault can be created by the beneficiary when related to a problem found while client is in Provisioning. In this case, the urgency is higher and the priority is set as input parameter to "M_URG_2H". After creation, the provider has 15 minutes to accept or reject this request, sent through notifyTicket as in fig. 4.2. NotifyTicket can be sent with different parameters as "ACEITA_AVR_PROV" in this case the fault stays with the same priority and the respective SLA, when set to "NAO_ACEITA_AVR_PROV" here the respective priority and SLA go lower and the ticket life cycle continues with this new values. In case the 20 minutes are exceeded and the beneficiary gets no information about the state of the fault this gets immediately rejected.

- **"Avaria Comum: Detectada pelo Beneficiário"**

  The beneficiary notifies the owner by sending a generic create failure request as we can see in fig. 4.3 with information about the ticket, ports, respective PDO and extra information.

  In this type of fault, the parameter type identifies who detected the fault, in this case the fault type is "AV_COMUM_BENEF". After this, the owner sends the identifier of the fault in question through createFailureResponse method and then both parts can add comments. After the intervention, the owner sends a resolution notification with the root cause and information about all client fault's affected by that. When closed, means all other individual faults are closed too. The association between faults will only be possible if the owner part verifies that the fault in question is a generic one.

- **"Avaria Comum: Detectada pelo Proprietário"**

  The provider detects a fault in his own network and sends a create generic fault to the beneficiary as in fig. 4.4. In this request, the owner is responsible for sending information about the ticket, ports, respective PDO and the fault type will be "AV_COMUM_PROP". Like in the generic fault, both parts can do comments during the process. After resolution, the owner sends the notification with the root cause and the information about all client faults affected. After closure, all other unique faults are closed too.

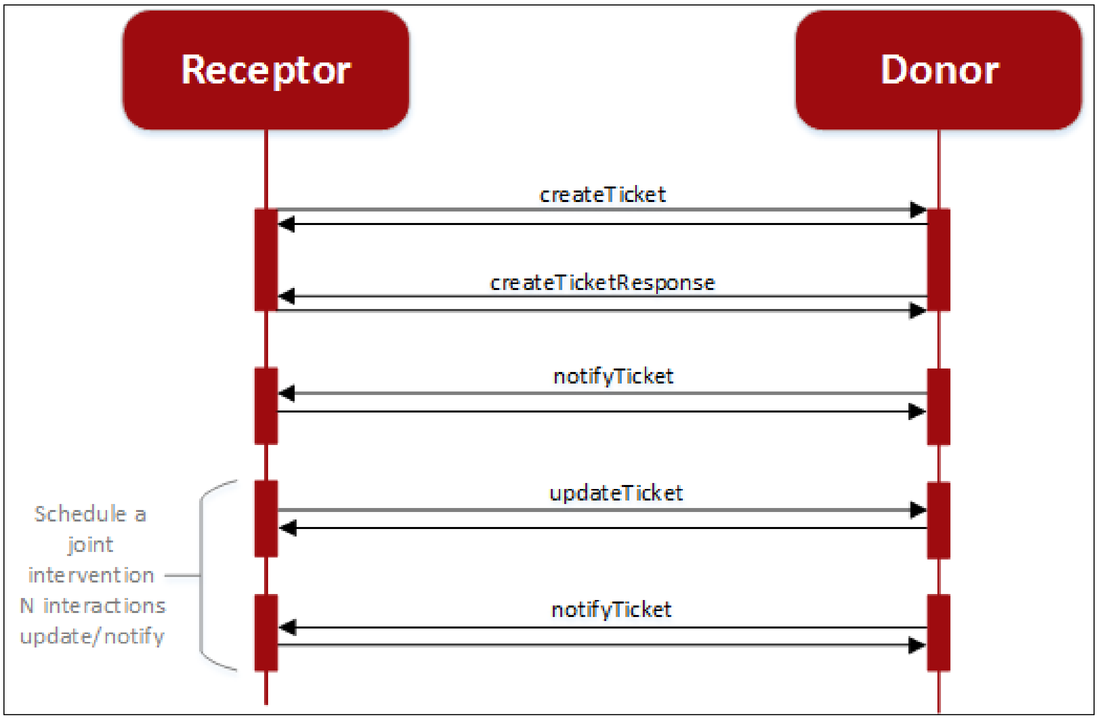


Figure 4.1: Flow diagram for "Avaria em Acesso de Cliente"

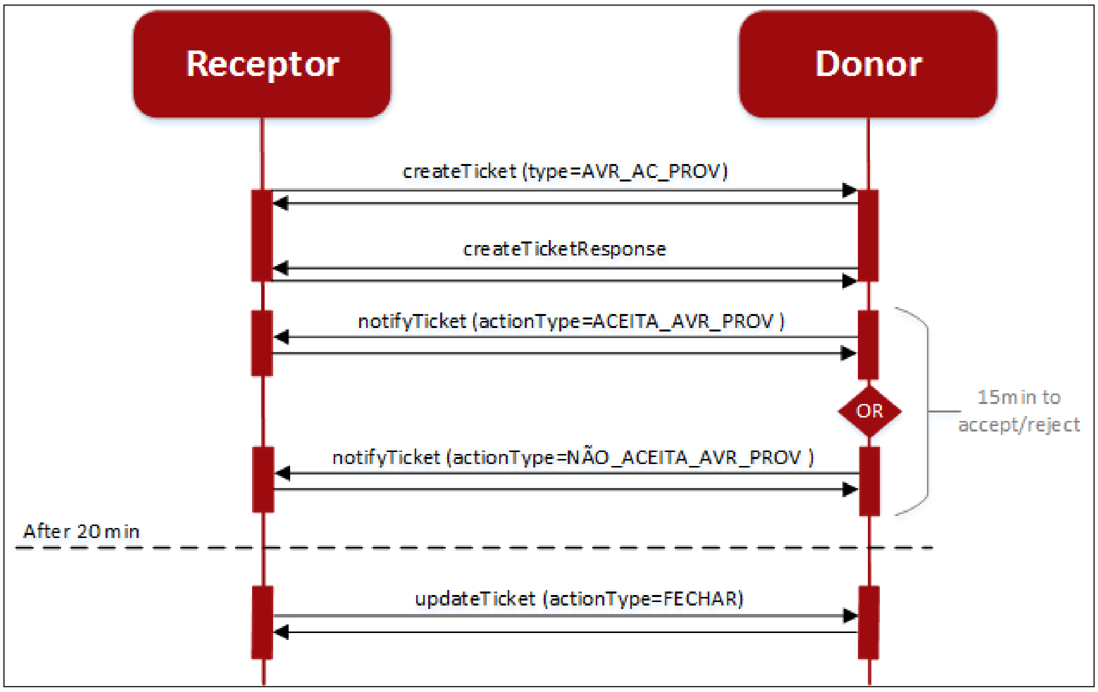


Figure 4.2: Flow diagram for "Avaria em Serviço Cliente Detectado em Provisão"

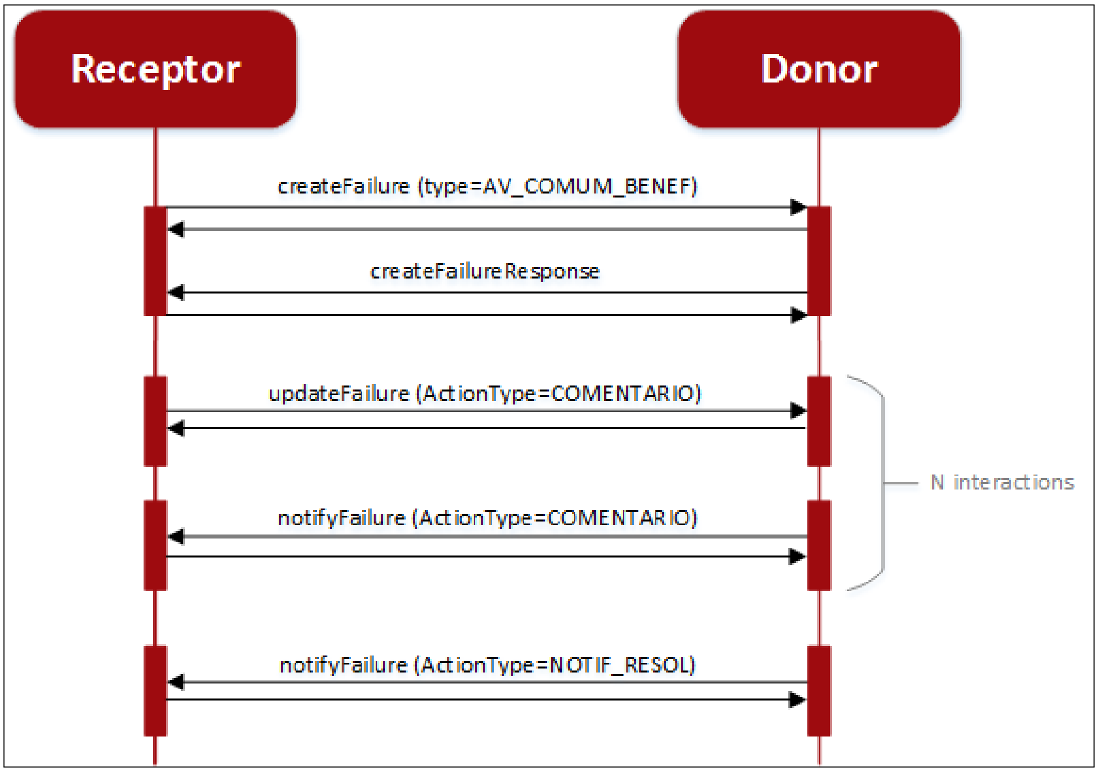


Figure 4.3: Flow diagram for "Avaria Comum Detectada pelo Beneficiário"

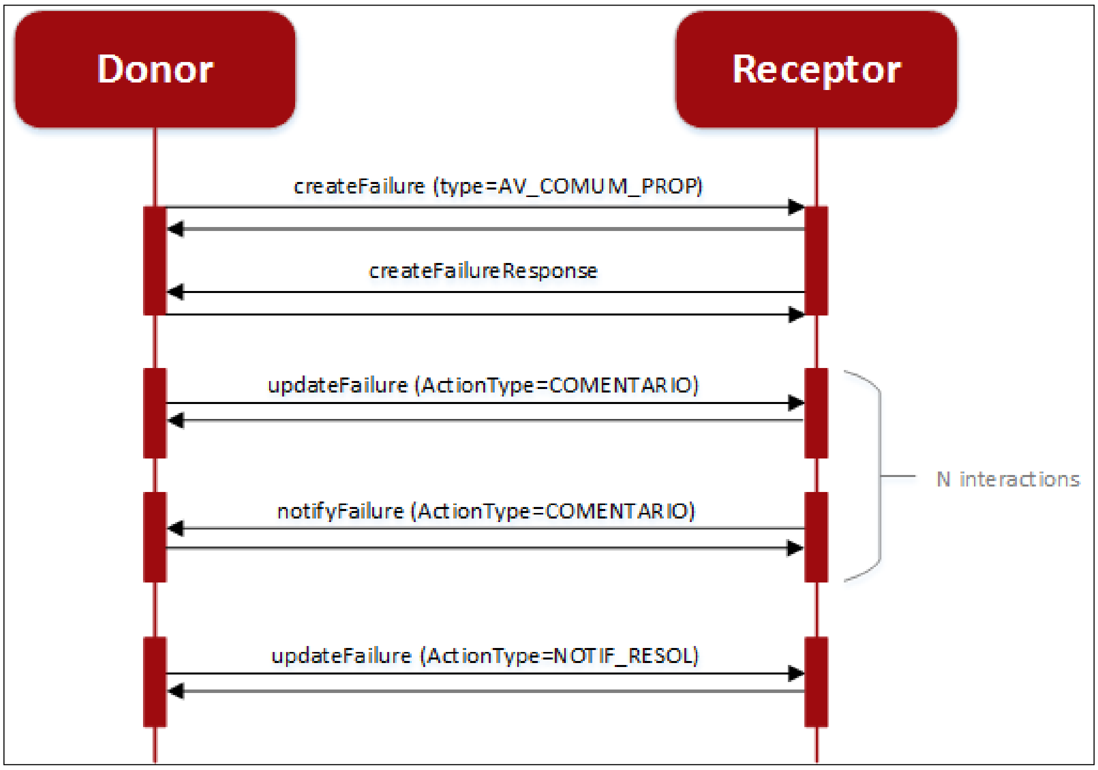


Figure 4.4: Flow diagram for "Avaria Comum Detectada pelo Proprietário"

The actions that can be taken during ticket's life-cycle are represented in a state diagram and each type of fault has its own flow. This represents the configuration used as input for the project. Transactions represent the actions taken associated with a specific operation while the states are the corresponding result. Both parts are shown, "B" for beneficiary" and "P" for owner. In fig. 4.5 is shown a generic diagram for "Avaria em acesso de cliente" representing all possible paths from start to finish including possible cancellations. This transactions are used as parameters in each API call and in case the process changes the necessary updates need to be made in order to avoid incorrect and inconsistencies with reality.

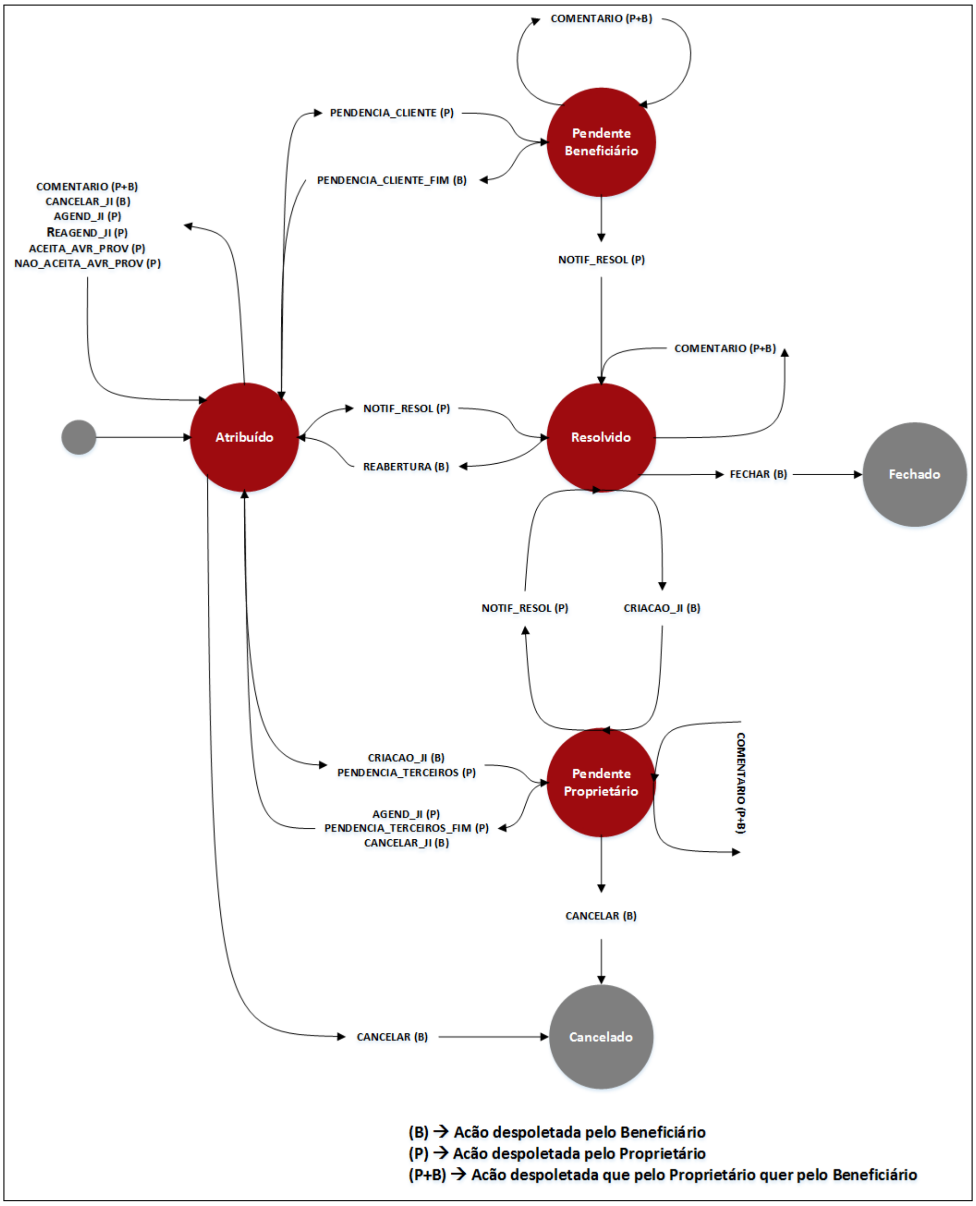


Figure 4.5: State diagram for "Avaria em acesso de cliente"

4.3 Dataset Description

Data retrieved from each API call, is based on a sonda that intercepts and captures meta-data for each ticket exchange. This meta-data consists of statistics about the service and the corresponding input and output parameters that will then be sent to Splunk for analysis.

4.3.1 Payload Data

This information is stored in a Splunk index called esi_payload, with around 195 thousand events for a month period. Each execution originates two events in JSON format one with the input and one with the corresponding output, related by **svc_name** the name of the service called, **rand_id** an unique id for each execution and **timestamp** fields.

In fig. 4.6 is an example of a service execution log for NotifyTicket operation. The service value is coded due to Splunk having limits for indexing quota, to avoid fields with long string values. In this case, the value "1653" corresponds to "PRFEO.GenericTicket notifyTicket" and the rand_id "314" is an unique id for that specific invocation alongside the timestamp. All events corresponding to synchronous responses are not analyzed in terms of process discovering.


```
{ [-]
   dt: { [-]
     in: { [-]
       ControlStructure: { [-]
         ProcessID: 1-1JA4KU8K_0
         ReceivingPartner: TELE1
         RequestID: 7E57D6FB-6EBD-399D-E053-0AB8A70A7385
         SendingPartner: TELE2
         Timestamp: 2019-12-31T19:58:42Z
         Version: 1
       }
       NotifyTicket: { [-]
         Ticket: { [-]
           actionSubType: REP_PROP
           actionType: NOTIF_RESOL
           id: TELE2_INC000029651835
           observations: foi feita fusão em PLC anterior. sinais -18.8 1490 e -1.7 1550
         }
         circuitID: VNF427.PLC-YJA14-009.1
       }
     }
   }
   port: 15555
   rand_id: 314
   svc_name: 1653
   ts: 1546286322810
}
```


Figure 4.6: Payload event for "PRFEO.GenericTicket:notifyTicket" service call

The dt{} structure contains payload information under an in{} structure for requests and out{} for the response.

In all API calls a common control structure (ControlStructure) is used with the following format:

Field	**Format**	**Description**
Version	String	API version starting at “1”.
Timestamp	DateTime	Date and time in format W3C. Example: “2019-12-28T16:00:14Z”.
ProcessID	String (100)	Persistent process identifier. The same ProcessID is used in all process requests.
RequestID	String (100)	Request unique ID.
SendingPartner	String (3)	Identifies the source operator for a request.
ReceivingPartner	String (3)	Identifies the destiny operator for a request.

Table 4.1: ControlStructure format description.

Depending on the type of operation a structure will follow with information about the request. In this case, NotifyTicket call contains the following fields:

Field	**Format**	**Mandatory**	**Description**
NotifyTicket/circuitID	String	Yes	Client circuit identifier.Example: TST51_PDO02.04.
NotifyTicket/Ticket/id	String	Yes	Ticket identifier.
NotifyTicket/Ticket/actionType	String	Yes	
NotifyTicket/Ticket/actionSubType	String	Yes	
NotifyTicket/Ticket/observations	String	No	Free text.

Table 4.2: NotifyTicket structure format description.

4.3.2 Process Final

Data from index esi_payload goes through a sequence of processing steps as a combination of Splunk macros and searches, using lookup tables for translation. Macros are essentially reusable chunks of Search Processing Language (SPL) that can be inserted into other searches. On the other hand, lookups are made to enrich event data by adding field-value combinations.

All this transformations occur in a main search called "esi_process_stream_finalized" with a goal of collecting tickets with a valid sequence of states and that are closed, to a final index called process_final as in fig. 4.7 produced by the engine.

```
12/31/2019 19:58:42 +0000, info_search_time=1553273544.095, t1=0, t3=0, tA=23674, tB=0, ts="1546286322810 1546286324270", tB1=0, tpc=0, tpt=0, origin=TELE2, svc_name=1653, ci
rcuitID="VNF427.PLC-YJA14-009.1", dt.in.ControlStructure.SendingPartner=TELE2, dt.in.NotifyTicket.Ticket.observations="foi feita fusão em PLC anterior. sinais -18.8 1490 e -
1.7 1550", esi_process_has_creation=0, esi_process_step_in_place=NOTOK, esi_process_step_raw_time="1546286322.81", esi_process_step_time="1546286322.81", dt.in.NotifyTicket.T
icket.id=TELE2_INC000029651835, dt.in.NotifyTicket.circuitID="VNF427.PLC-YJA14-009.1", esi_process_sequence=NOTOK, dt.in.NotifyTicket.Ticket.actionSubType=REP_PROP, sla_thres
hold=21600, dt.out.ResultsStructure.StatusCode=0, esi_process_id=FTTH_SHARE_ticket_ic_direta, esi_step_label=TKT_RESOLVED, previous_step_valid_destination_list=" ", actionTyp
e=NOTIF_RESOL, process_id="1-1JA4KU8K_0", dt.in.ControlStructure.ReceivingPartner=TELE1, process_has_creation=1, dt.in.UpdateTicket.Ticket.actionType="-", dt.in.ControlStruct
ure.Timestamp="2019-12-31T19:58:42Z", actionSubType=REP_PROP, esi_process_step=NOTIF_RESOL, dt.in.CreateTicket.Ticket.type="-", dt.in.NotifyTicket.Ticket.actionType=NOTIF_RES
OL, destination=TELE1, status_code=0, esi_process_status=CLOSED, dt.in.CreateTicket.Ticket.priority="-", dt.out.ResultsStructure.StatusMessage=""
```


Figure 4.7: Process final raw event

The first lookup used is process_step in combination with macro process_filter_payloads_fields_lookup and process_attrbute_fields_lookup, responsible for making sure that the engine only captures the correct events that respect all outlined features by applying a filter to all events and the corresponding fields that can be maintained before next processing steps start.

The second function process_step_tag is the most important by adding to each esi_payload event a couple of fields like esi_process_id, esi_step_label and esi_step_id that will allow the solution to know how to threat them in the context of stage in a ticket (process). Last transformation associates each ticket step with the corresponding timer flag: start, finish or cancel.

Field	**Description**
esi_process_id	Ticket type depending on combined features.
esi_step_label	Step in a ticket context.
esi_step_id	Used to help analyze a ticket over time.
svc_name	Service name.
payload_filters	Only events with the corresponding fields values will be used.
attribute_fields	Fields list to use in order to reduce disk space from ticket processing.
attribute_calculations	Transformations applied to source fields.
timer	Time flag.
timer_details	Details about time.

Table 4.3: Lookup esi_process_step fields description.

4.4 Problem Definition

As mentioned in section 1.1, processes are the core functionality of an organization and need to be as precise and well structured as possible to generate business value and make organizations more competitive. This way there is the need to constantly make adjustments and adapt to potential problems that can be faced on a daily basis. This adjustments need to be in the shortest time possible to minimize the impact on clients and to ensure the overall well function of services. The problem faced was related to the existing process configuration, based on a table with all possible steps and transitions with the need to use human intervention for every update made. Having this in mind, the challenge was to discover this configuration based on logs produced by the execution of the process, in order to turn all this mechanism automatic and to reduce resource allocation. This decision to automate process configuration looks to overcome the following problems:

- **Lack of Flexibility-** There can be times where the current process configuration is not aligned with the reality, generating wrong assumptions and blocking business growth.
- **Low Accuracy-** The current interface used to build this configuration is a Splunk Application called "Lookup Editor" that works as a simple excel spreadsheet. Is too complex to edit large amount of lines and can generate mistakes if one value is miswritten.
- **Low Cost-Effectiveness-** Since updating this table requires human intervention, a person from the project team with background and technical knowledge needs to be available every time a change is needed. Bringing more costs to the overall project and reducing the available members to do other functions.

5

Proposed Solution

Contents

In this section we will start by explaining the solution adopted and the way Splunk was used as a process mining tool. Then will be described how this solution was implemented to discover the configuration for the FTTH_SHARE project based on Splunk Business Flow Application. This solution is divided in two phases, first part using data collected by the FTTH_SHARE engine with the already build steps and transitions and then based on process execution logs.

5.1 Solution

Considering that the current process configuration with all possible steps and paths used as input for the FTTH_SHARE engine, was created and updated manually and having in mind that the main goal was to find a way to discover and turn this process fully automatic. Different process mining tools like ProM and ARIS were analyzed as shown in section 3, to understand how they use logs as input to further apply the different algorithms in order to discover processes. Considering that the logs in case were already in Splunk and the developed engine too, was made the decision to explore Splunk applications that could discover processes and enable the possibility to visualize them. Splunk Business Flow was chosen since is a process mining tool that besides the process models produces statistics and enables different interactions by applying all kind of filters like time and resource based.

5.2 Overview of Splunk Business Flow

Splunk Business Flow[1] is a process mining tool from Splunk to analyze and generate processes, by applying transformations using searches based on previously indexed data. To start using this tool a **Flow Model** must be created by defining a search to scan indexed data and extract events based on the specifications as well as fields for Correlation ID - an unique descriptor of events used to identify related events in the event log and group them into Journeys, Step - representing an action and Attributes - optional fields that are characteristics of an event that can be added to help further analysis. Is also possible to change sample size to view more events and so reducing search time and is also possible to change Max Duration parameter to group events with the same correlation ID into a journey within that time period. A **Journey** like fig. 5.1 represents a single process case path from start to finish.

[1] https://docs.splunk.com/Documentation/SBF/-Latest-/UserManual/WelcometoSplunkBusinessFlow

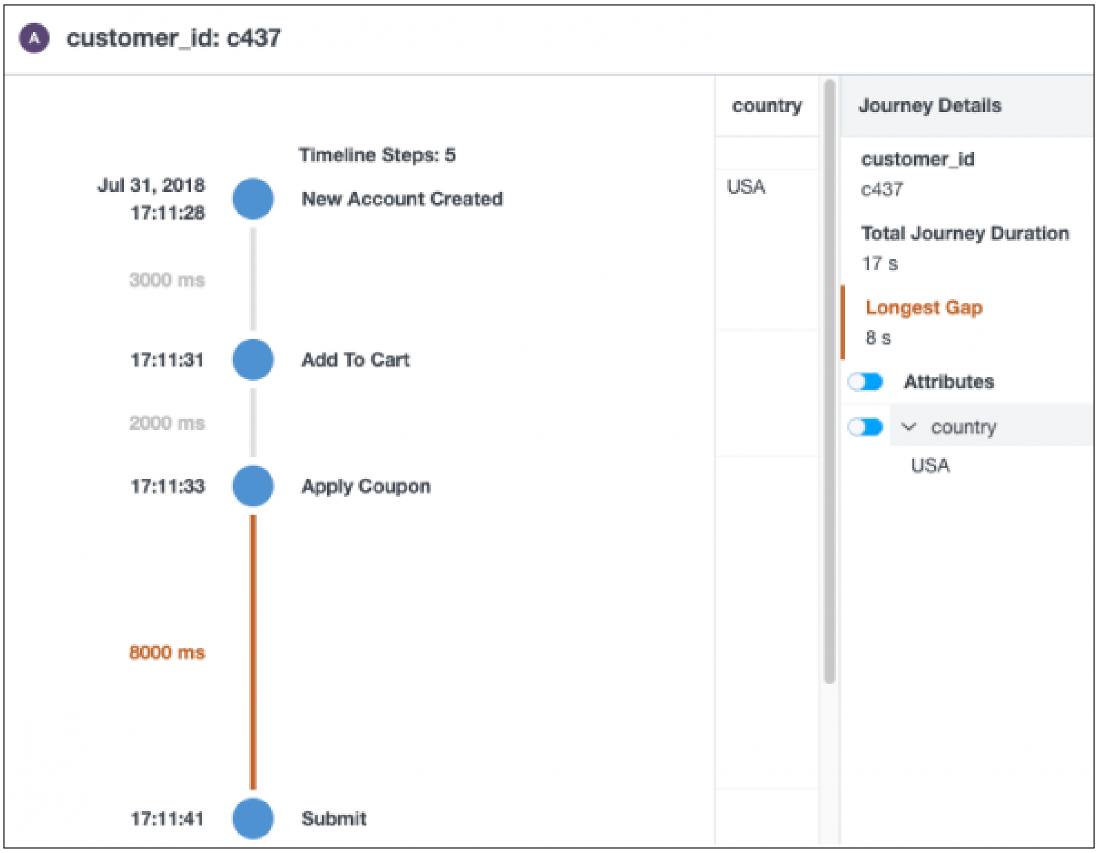


Figure 5.1: Single journey and respective details (figure from [9])

After Flow Model definition, a validation step is available to make sure all the steps we are interested in tracking are present in the model. Some adjustments can be made by changing the time range to increase the number of events, using filters like attributes, journey duration or start and end time. Additional proprieties can be applied to the flowchart to help analyze like step count or journey count. Also, different types of analyzes can be made by comparing two journeys side by side allowing drilldown by each Correlation ID as well as "List" view to see journeys grouped, "Attributes" view or "Metrics" view to analyze different perspectives. Before saving the flow model, a final step to configure settings is needed to change visibility between private or shared in case we want our flow model to be only for developing and testing purposes.

Flows work as a workspace in Splunk Business Flow and help analyze part of a process by choosing which steps to show as well as configure different lanes. In overview mode as in fig. 5.2 is shown a dashboard with different visualizations that help keeping track of total journeys, average duration by journey and distribution by step. Over this information can be defined alarms by creating notifications triggered when a number of journeys reach a threshold sending an email.

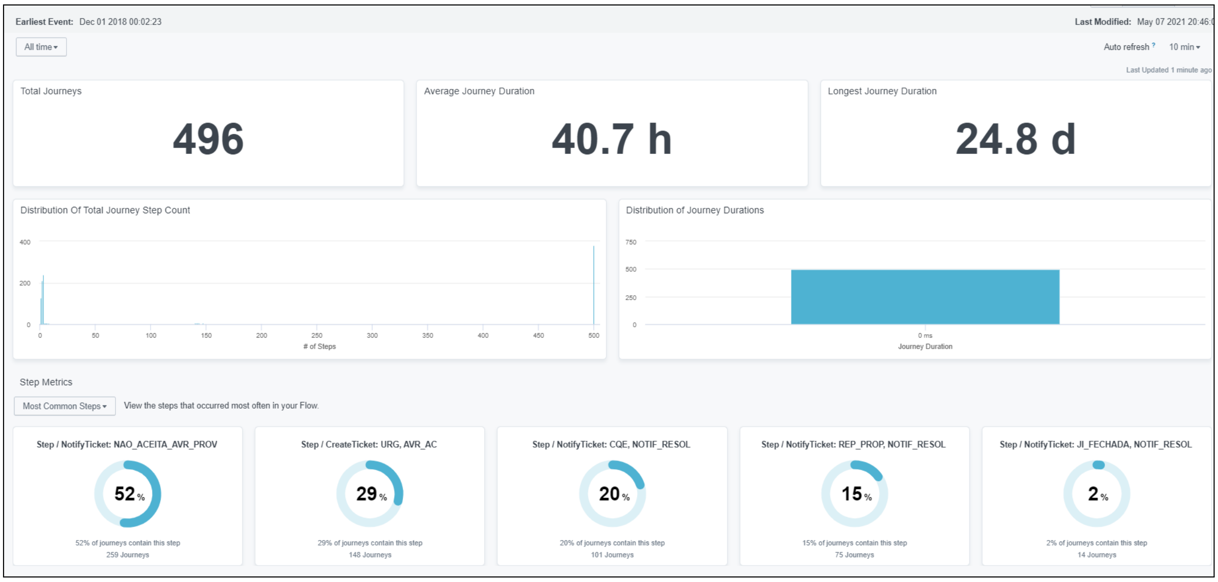


Figure 5.2: Flow overview dashboard example

5.3 Implementation

After studying how Splunk Business Flow uses logs and initial configuration as described in Section 5.1 to produce flow models, work was divided in two phases where each one uses different indexed data as input, to first discover the process model with the already known configuration steps and transitions and then discover the process from the original logs. The available data produced by the conversation between the two sides is used in phase 2 to add different complexity levels to the overall solution.

5.3.1 Phase 1: High-level process discovery

For this phase, the implementation was based on data from esi_process_final splunk index that contains steps and transitions produced by FTTH_SHARE project engine. This way, the search (listing 5.1) was written having in consideration only tickets with esi_process_sequence value "OK", meaning for each ticket every transition was checked for the all sequence to be considered valid. A transition is valid if the respective steps inside a ticket are in the same order as they occur in time. Since the events are collected in Splunk not as they occur in real time, their time column needs to change to the corresponding time in which they occur as in esi_process_step_time field.

Listing 5.1: Phase 1 - Splunk search used as input in Splunk Business Flow

```
index=esi_process_final esi_process_sequence="OK"
| eval _time=esi_process_step_time
```

In order to explore the corresponding flow model, the input parameters used by Splunk Business Flow need to be configured. The "Sample Size" was defined as 10 000 events and the "Max Duration" parameter as 365 days to cover tickets that have a duration of one year. The other input fields as we see in fig. 5.3 and fig. 5.4 and fig. 5.5, used to correlate events, were selected based on the definitions already explained before, for "Correlation ID" the selected field "process_id" since it is an unique identifier of each ticket, for "Step" the selected field "esi_step_label" that represents each step name and different "Attributes" were selected for further analysis as actionSubType, actionType and circuitID.

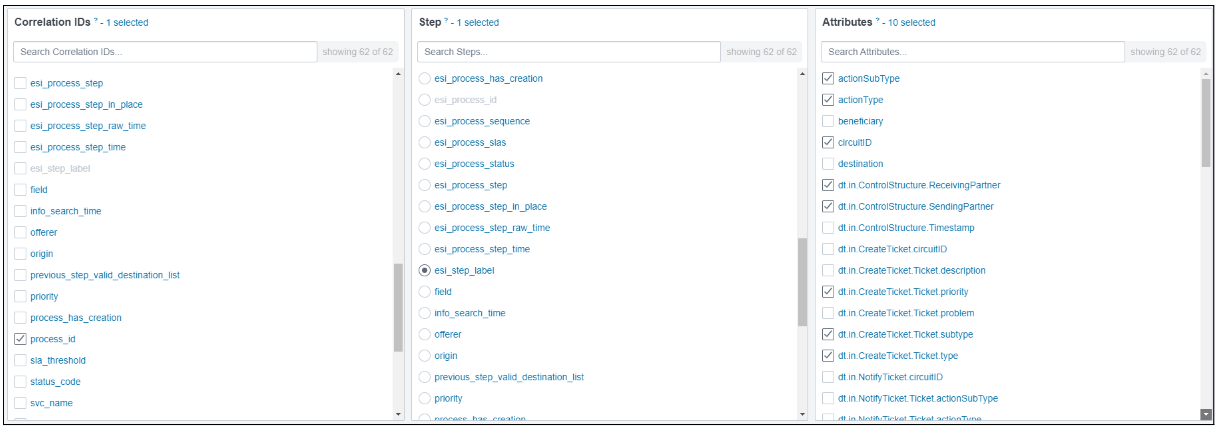


Figure 5.3: Phase 1 - Input parameters used in Splunk Business Flow

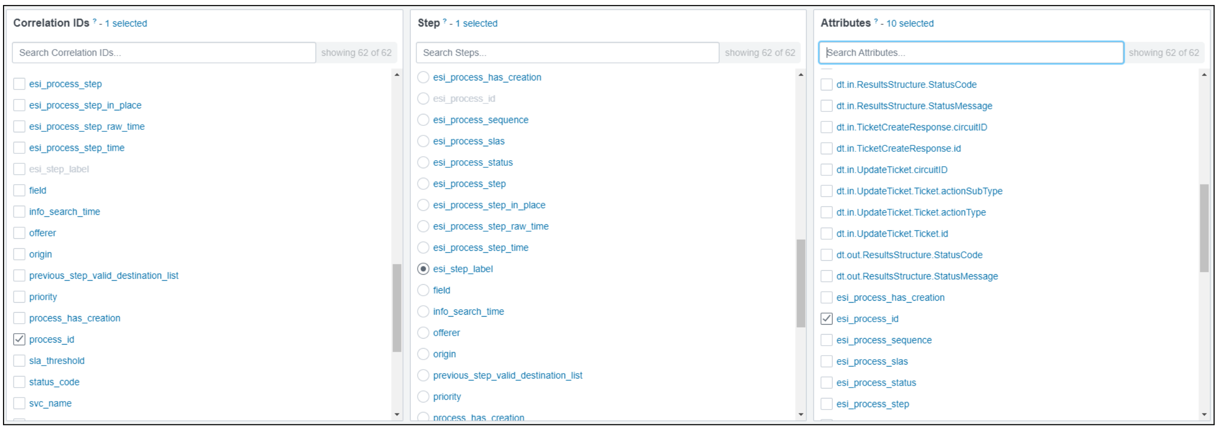


Figure 5.4: Phase 1 - Input parameters used in Splunk Business Flow

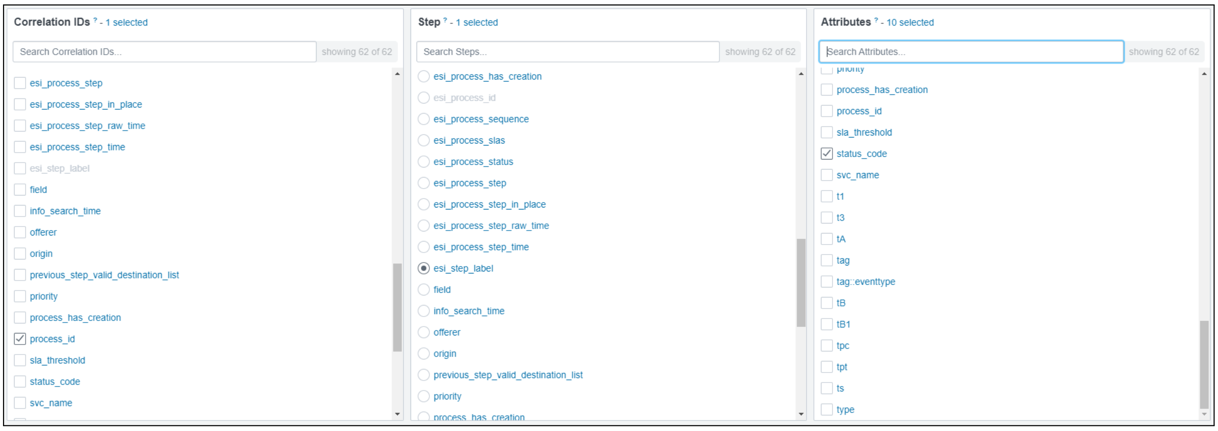


Figure 5.5: Phase 1 - Input parameters used in Splunk Business Flow

5.3.2 Phase 2: Process discovery from raw data

In this second phase, the goal was based on indexed raw data present in esi_payload as a result of each API call, discover the initial process configuration. In this case, was considered only data extracted from December 2018 since it was previously indexed in Splunk. In this search (listing 5.2), all calculations are defined under a macro component called ticket_process_discovery which is a knowledge object from Splunk that works as a function in a way that can be called with different parameter values.

This macro has three input parameters **steps**, all functions considered for a ticket,**filters** that represent splunk field names for this functions and an **ignore_filter** with all the field values to exclude from the search. Using a macro for this implementation makes it easier for future changes in case of process adjustments, with an easier way of adding new steps.

This macro (listing 5.3) first applies the ignore_filter to the already existing search string and using eval function creates a search field for each input parameter (step and filters). Since both fields have multiple values separated by commas, split function is applied to create a Splunk multi value followed by an eval to create a new field called "step_custom_name" that is further used by Splunk Business Flow tool to build the corresponding process model. This custom field is made of the function name applied to the ticket and the corresponding parameters used, for example "CreateTicket:URG,AVR_AC". In this first solution of phase 2 the parameter names are still coded without any translations being applied.

Listing 5.2: Phase 2 - Splunk search used as input in Splunk Business Flow

```
index=esi_payload source="esi_payload_ftth_share_20181201_20181231.txt"
`ticket_process_discovery("CreateTicket,NotifyTicket,UpdateTicket,TicketCreateResponse","dt.in.CreateTicket.Ticket.type,dt.
    in.CreateTicket.Ticket.priority,dt.in.NotifyTicket.Ticket.actionType,dt.in.NotifyTicket.Ticket.actionSubType,dt.in.
    UpdateTicket.Ticket.actionType","(NOT dt.out.ResultsStructure.StatusMessage=*) (NOT dt.in.NotifyTicket.Ticket.
```

```
actionType="COMENTARIO") (NOT dt.in.UpdateTicket.Ticket.actionType="COMENTARIO")")`
```

Listing 5.3: Phase 2 - Splunk macro used in search

```
$ignore_filter$
| eval step="$steps$",
        filters="$filters$",
        step_multi_field = split(step,","),
        filters_multi_field = split(filters,","),
        step_custom_name= mvmap(step_multi_field, if(like(_raw,"%".step_multi_field."%"), step_multi_field, null())).": "
| foreach *
    [eval step_custom_name = step_custom_name.if(isnotnull(mvmap(filters_multi_field, if("<<MATCHSTR>>" =
        filters_multi_field AND '<<FIELD>>'!="", '<<FIELD>>', null()))),mvmap(filters_multi_field, if("<<MATCHSTR>>" =
        filters_multi_field AND '<<FIELD>>'!="", '<<FIELD>>'.", ", null())),"")]
| eval step_custom_name =substr(step_custom_name,1,len(step_custom_name)-2)
```

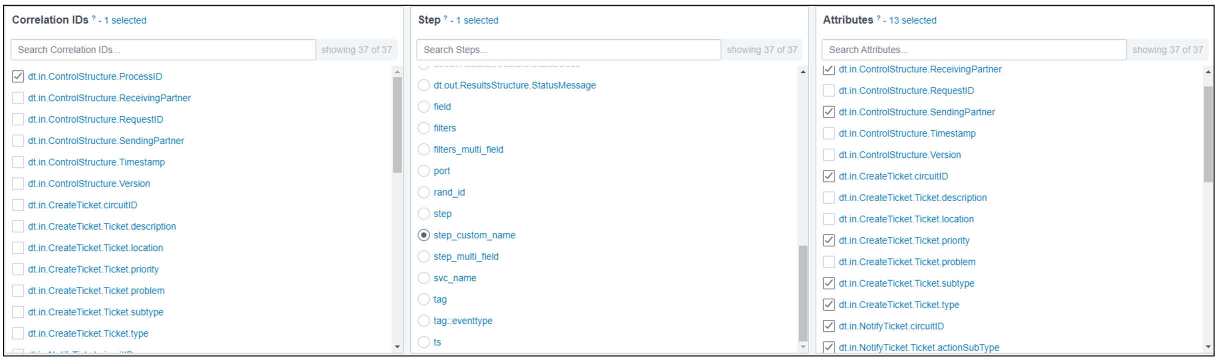


Figure 5.6: Phase 2 - Input parameters used in Splunk Business Flow

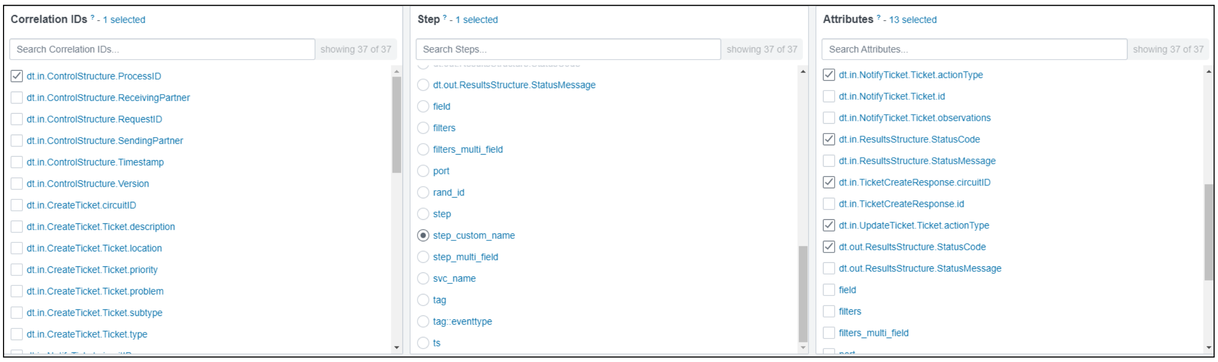


Figure 5.7: Phase 2 - Input parameters used in Splunk Business Flow

For this phase the input parameters defined in Splunk Business Flow (fig. 5.6 and fig. 5.7) were similar to phase 1 with the exception of "Step" parameter that was defined to use field "custom_step_name" as a result of the search previously described.

In this second phase since data used for input in Splunk was a result of each API call, the corresponding values were coded in a sense that were not understandable by anyone outside of the developers. To solve this problem was produced one different macro called ticket_process_discovery_decoded with the same parameters (listing 5.4) as the previous one but with the difference that translates each field value called "Motivo Resolução" (listing 5.5) to the corresponding text that anyone can look at the process and understand it.

Listing 5.4: Phase 2 - Splunk decoded search used as input in Splunk Business Flow

```
index=esi_payload source="esi_payload_ftth_share_20181201_20181231.txt"
`ticket_process_discovery_decoded("CreateTicket,NotifyTicket,UpdateTicket,TicketCreateResponse","dt.in.CreateTicket.Ticket.type,dt.in.CreateTicket.Ticket.priority,dt.in.NotifyTicket.Ticket.actionType,dt.in.NotifyTicket.Ticket.actionSubType,dt.in.UpdateTicket.Ticket.actionType","(NOT dt.out.ResultsStructure.StatusMessage=*) (NOT dt.in.NotifyTicket.Ticket.actionType="COMENTARIO") (NOT dt.in.UpdateTicket.Ticket.actionType="COMENTARIO")")`
```

Listing 5.5: Phase 2 - Splunk decoded macro used in search

```
$ignore_filter$
| lookup MapeamentoMotivosResolucao_ftth_share.csv "Motivo Resolucao (API)" as dt.in.NotifyTicket.Ticket.actionSubType OUTPUT "Motivo Resolucao (Texto)" as dt.in.NotifyTicket.Ticket.actionSubType
| eval step="$steps$",
        filters="$filters$",
        step_multi_field = split(step,","),
        filters_multi_field = split(filters,","),
        step_custom_name= mvmap(step_multi_field, if(like(_raw,"%".step_multi_field."%"), step_multi_field, null())).": "
| foreach *
    [eval step_custom_name = step_custom_name.if(isnotnull(mvmap(filters_multi_field, if("<<MATCHSTR>>" = filters_multi_field AND '<<FIELD>>'!="", '<<FIELD>>', null()))),mvmap(filters_multi_field, if("<<MATCHSTR>>" = filters_multi_field AND '<<FIELD>>'!="", '<<FIELD>>'.", ", null())),"")]
| eval step_custom_name =substr(step_custom_name,1,len(step_custom_name)-2)
```

5.3.3 Results

During the implementation phase not everything went smoothly as expected. The process model produced and the corresponding graphs created by different filters, could not be saved as images and exported from the application, this is a known issue from this tool. Every time Splunk Business Flow was opened the corresponding flow search had to run to produce the graphs and enable further analyzes. In this time period, the license ended and after trying different approaches to solve this, new license could not be added. With this limitation over application usage, the only access possible was flow's initial page with the search and filters.

All images used in section 6 were created before the license expired with focus on showing process models for the different ticket types and analyzes over the most frequent process cases, with focus on the most frequent case showing step details and count. In this analyzes, is shown an overview over the attributes where we can see the most frequent values for the most important parameters used in each API call and also Splunk Business Flow has a section "Metrics" that has statistics for the overall cases, with average duration and count.

6

Demonstration

Contents

In order to conduct our main experiments, we had to install Splunk Business Flow tool on a server where the logs were previously copied to. This data was exported from a month period corresponding to every API call as a result of every interaction between the two organizations. In preliminary experiments along with this information, the corresponding processed data from the engine was also collected as a starting point to our solution. As mentioned before, during this phase not everything went as expected and the software license ended earlier, restricting the results we could export. After trying different approaches to solve this, a new license could not be added and with this limitation the interaction with the tool got very restricted to only access flow's definition page with the corresponding search and filters.

Since this tool does not enable the option to export models as images we had to do it in the traditional way with prints each time a result was produced and this limitation is another reason why we could not proceed with a formal evaluation phase and all process images in this work are a result of it. All models used in section 6 are a result of applying process mining with focus on showing the corresponding processes for the different ticket types and analyzes over the most frequent process cases. In this analyzes, is shown an overview over the attributes where we can see the most frequent values for the most important parameters used in each interaction and also Splunk Business Flow has a section "Metrics" that has statistics for the overall cases, with values as average duration and count.

6.1 Process Model Produced

As a starting point to our solution we used logs with the already processed step names, as a result of the engine in the context of the FTTH_SHARE project. The goal was to use this logs to apply process mining using Business Flow tool in order to reach the configuration that was already known. This configuration is a large table with values for all possible steps and transitions for each ticket type and so it defines the overall process.

The first step was to find the model that could better describe the process by searching for only valid tickets where it could be possible to see from start to finish the different fault types related to each ticket since the moment they were created until they closed. As shown in fig. 6.1 for a ticket to be valid it has to start with create operation from one of the organizations followed by the corresponding response saying success or failure in creation. Following these operations is a sequence of steps and the process only finishes when we have a call over close or resolved operation, only these cases were considered as valid for mining the corresponding process.

In this phase, the valid sequences were already computed following these rules so we could apply mining directly from the log without the need to apply any transformations. After the process model was produced we could filter the results by the correlation_id attribute which was the field that was responsible for joining related events depending on each ticket type. In fig. 6.2, fig. 6.3 and fig. 6.4 we can see the different results produced and the respective ticket processes for "Avaria em Acesso de Cliente", "Intervenção Conjunta" and "Serviço Cliente Dectado em Provisão".

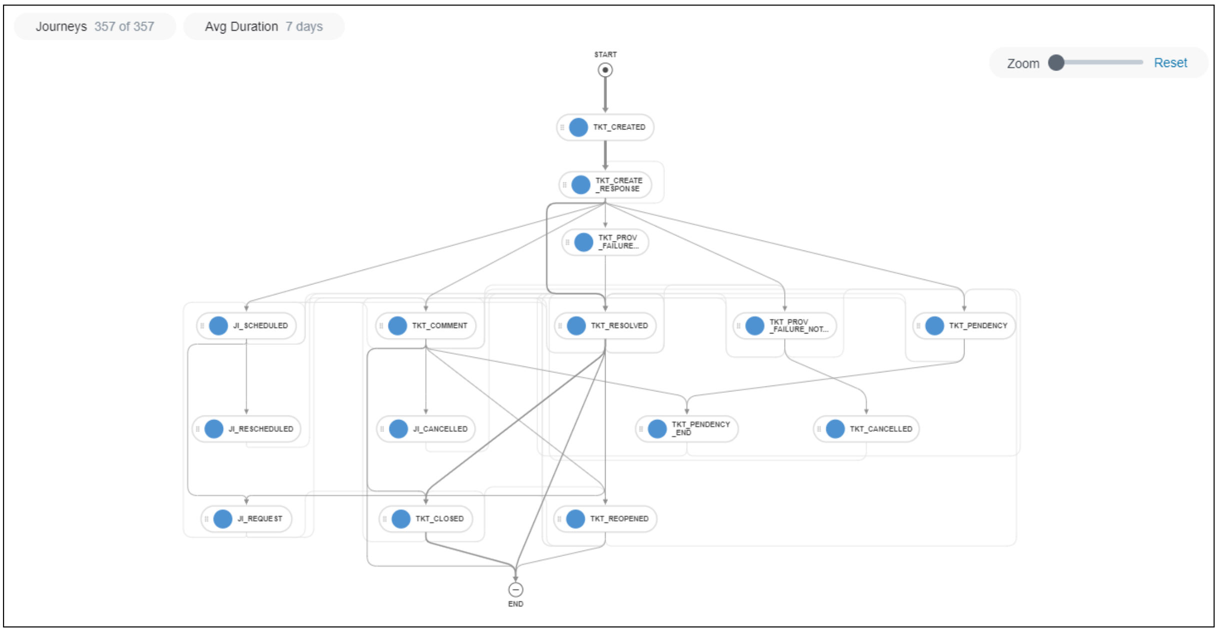


Figure 6.1: Phase 1 - Process model produced in Business Flow

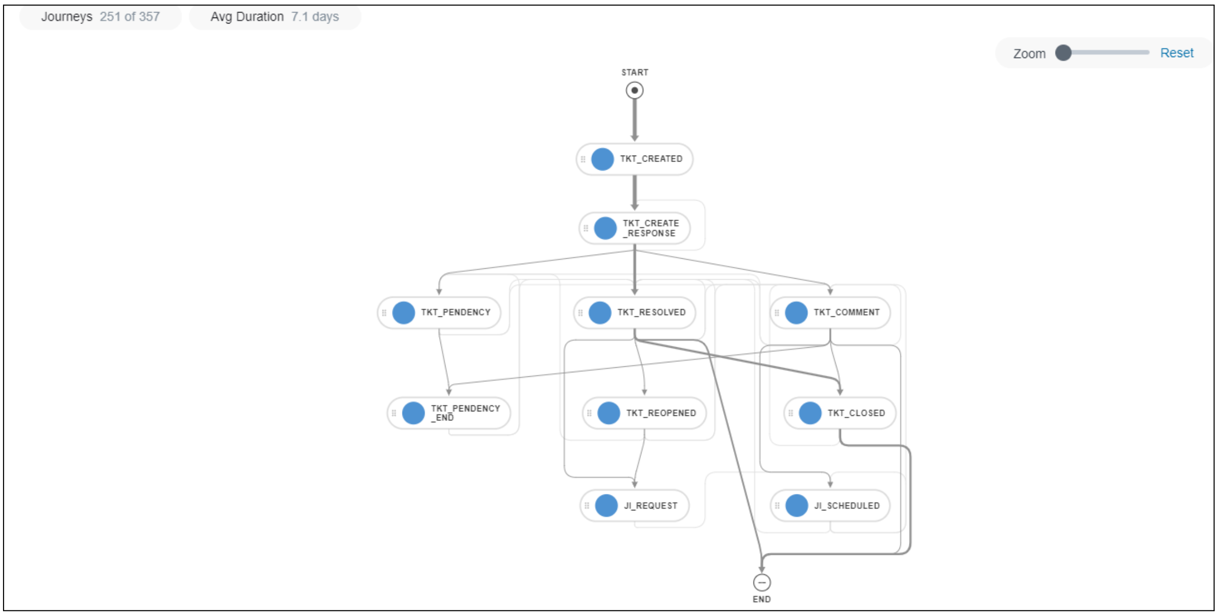


Figure 6.2: Phase 1 - Process model for "Avaria em Acesso de Cliente" produced in Business Flow

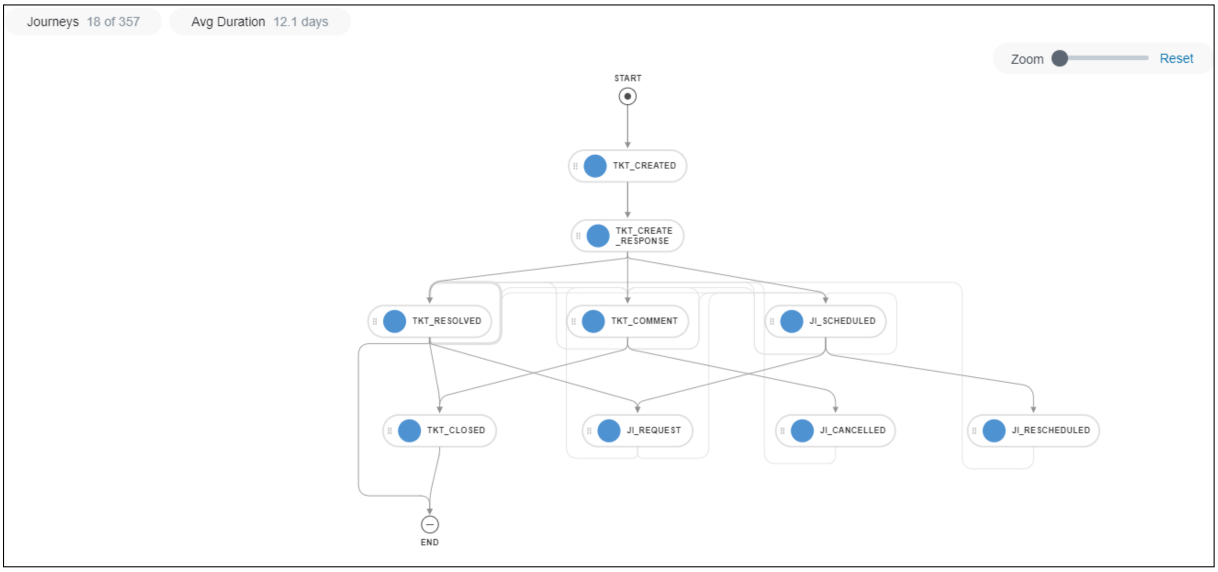


Figure 6.3: Phase 1 - Process model for "Intervenção Conjunta" produced in Business Flow

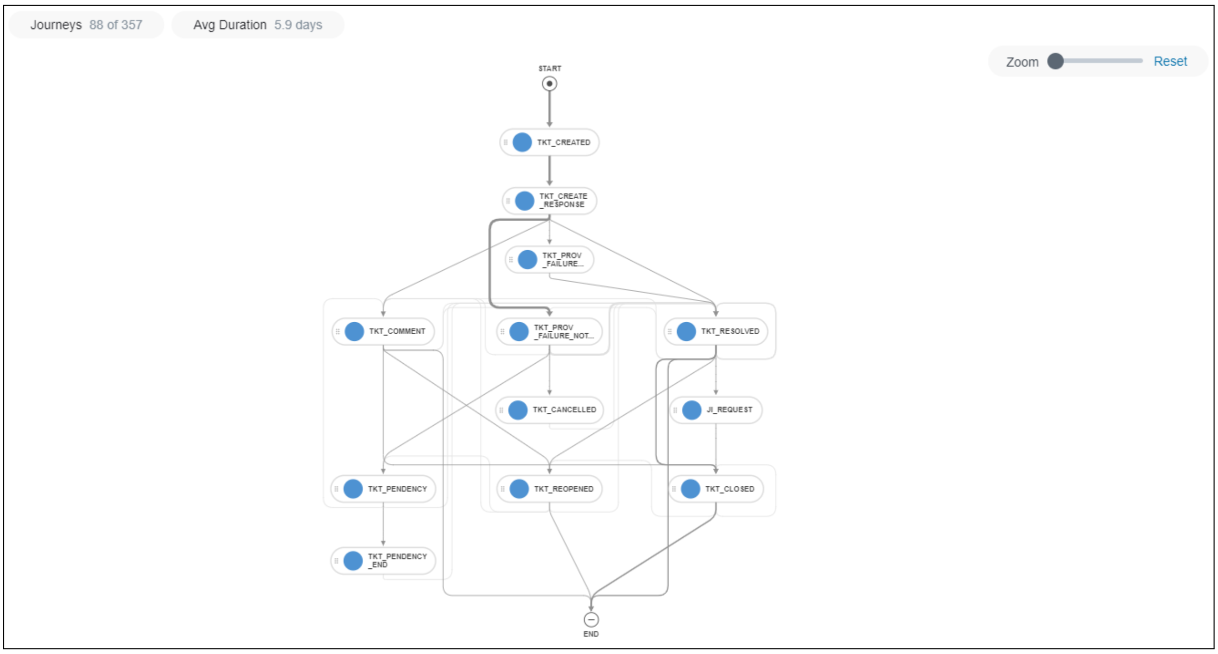


Figure 6.4: Phase 1 - Process model for "Avaria em Serviço Cliente Dectado em Provisão" produced in Business Flow

Once we got the process model for each ticket type, the next step to our solution was to produce this models but with the additional information present in the logs as a result of each interaction between the two parts. This second phase worked as a "drilldown" from the one presented, since here(fig. 6.5) we have access to the parameters that were used in each call resulting in much more steps and overall complexity for each process model. In this case since the information available did not receive any treatment and was directly from production log, we had to do an additional processing step since some cases reflected situations corresponding to processes that did not finish during that month period or to an API call that failed resulting in invalid processes. This processing step was were most of the time was spent during this phase because for each different correlation_ID we had to filter out every sequence that did not ended in one of the supposed steps (resolved or closed).

To separate the process model in each ticket sequence of steps, we used the parameter "type" that is used in the "CreateTicket" call. With this information is possible to divide the original process in one for each type of ticket as shown in fig. 6.6, fig. 6.7 and fig. 6.8. From "Avaria em Acesso de Cliente" process model we can see that the first step that was previously called "TKT_CREATED", now corresponds to three different steps depending on the parameters used in each call for example priority parameter can assume between the following values: URG, M_URG and NOR.

With this results we were looking to find similar results as the state diagram presented in Section 4.2 when it comes to tickets of "Avaria em acesso de cliente". This representation is used as a starting point for the project and represents all interactions between the owner of the service and the beneficiary. Here we translate this interactions to a sequence of actions that build a "conversation" between the two organizations.

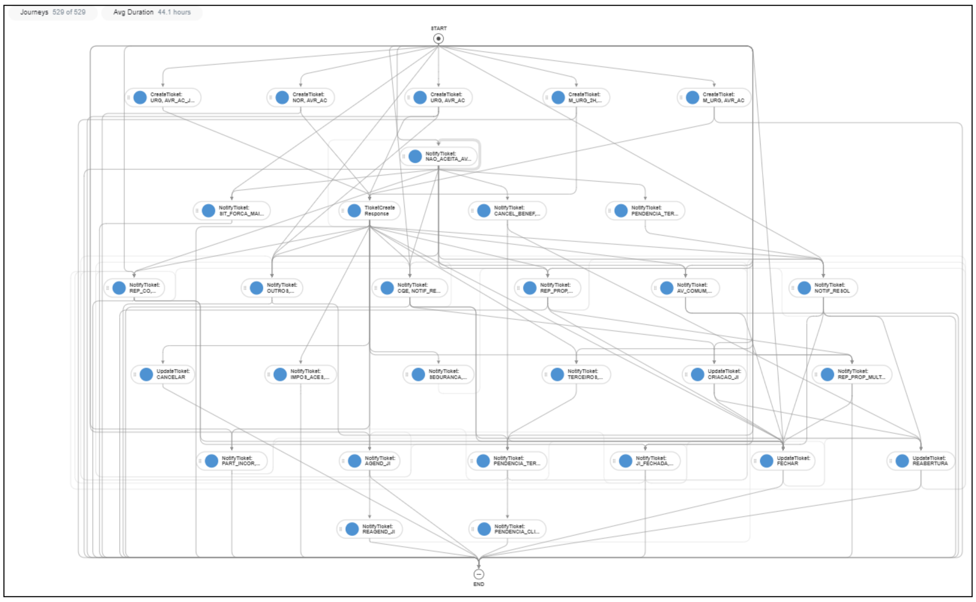


Figure 6.5: Phase 2 - Process model produced in Business Flow

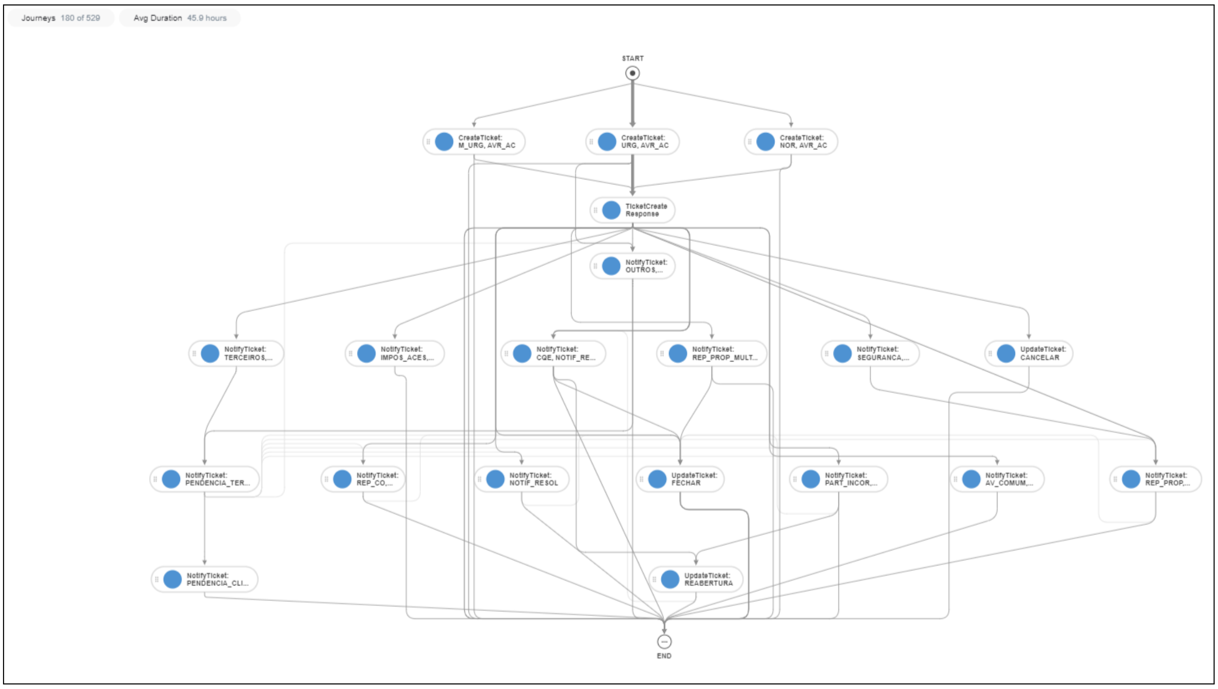


Figure 6.6: Phase 2 - Process model for "Avaria em Acesso de Cliente" produced in Business Flow

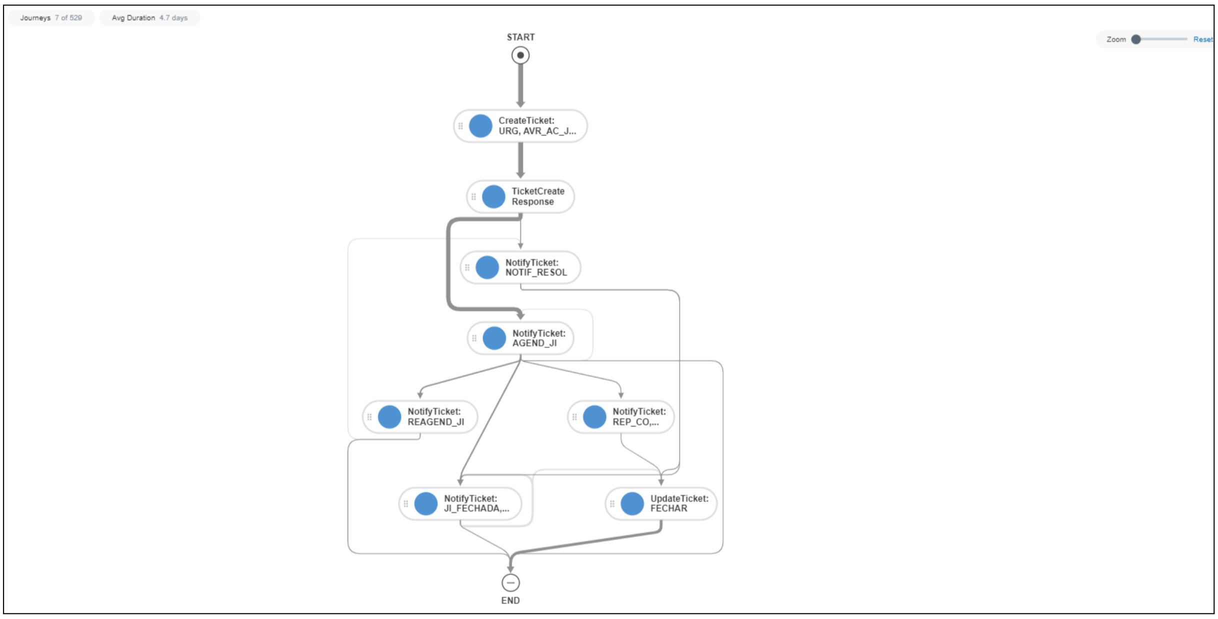


Figure 6.7: Phase 2 - Process model for "Intervenção Conjunta" produced in Business Flow

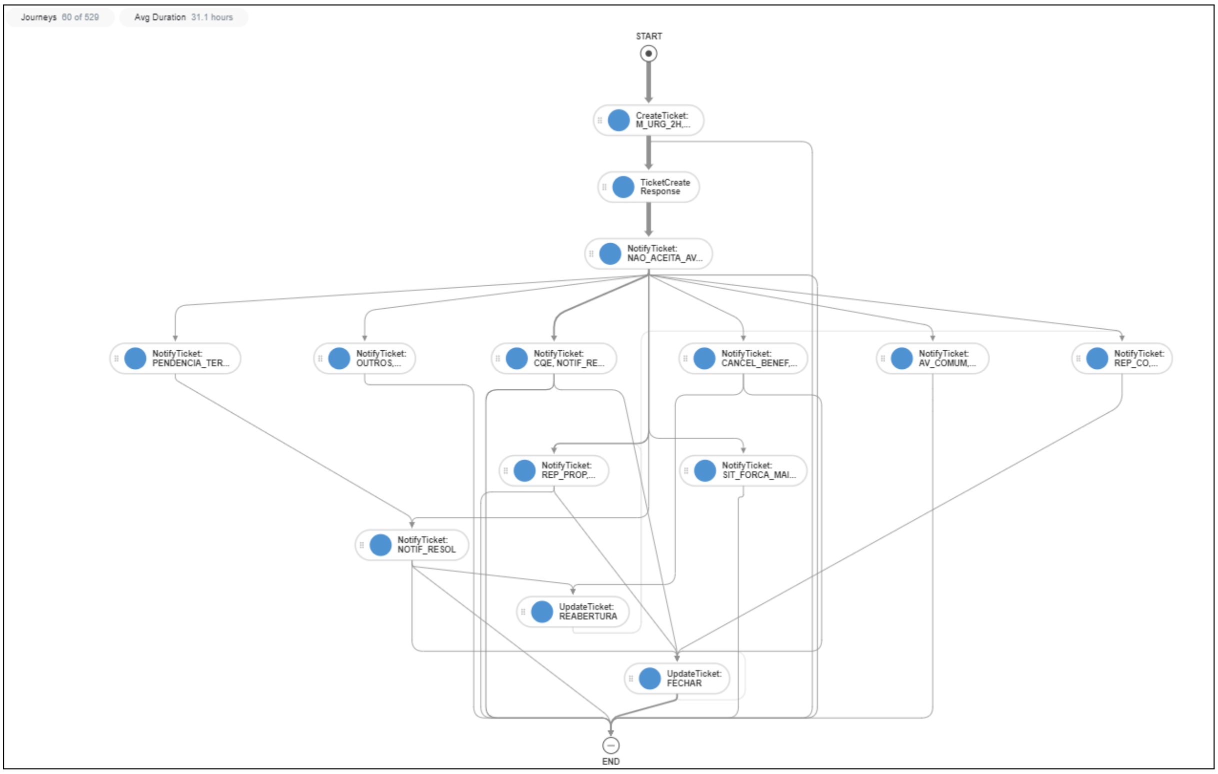


Figure 6.8: Phase 2 - Process model for "Avaria em Serviço Cliente Dectado em Provisão" produced in Business Flow

6.2 Attributes and Metrics Overview

In this section we present the main metrics and the analyzes produced in Splunk Business Flow, over the process models presented in Section 6.1 for each separate phase. Over on fig. 6.9 and fig. 6.10 we can see the "List" view where all process cases are listed for the results of phase 1 and also for phase 2 with the corresponding start and end times as well as the total duration and step count. This view enables a drilldown by clicking on a "process_id" value to see the sequence of steps for that particular case. Also, on this tab is possible through the filters side bar to see the most frequent case occurrences grouped by the respectively complexity (fig. 6.11).

Splunk Business Flow uses journey to call a process instance as described in Section 2, so in many images in this section we will see that term. In Conversion tab we can see a diagram (fig. 6.12) showing the distribution of process instances considering the four main stages. From the total of 357 process instances used for analyzes, all went through the TKT_CREATED operation and all got resolved but only 56.02% of this tickets could be closed. We can also see that one ticket got reopened, probably because the problem was not resolved completely on the first attempt.

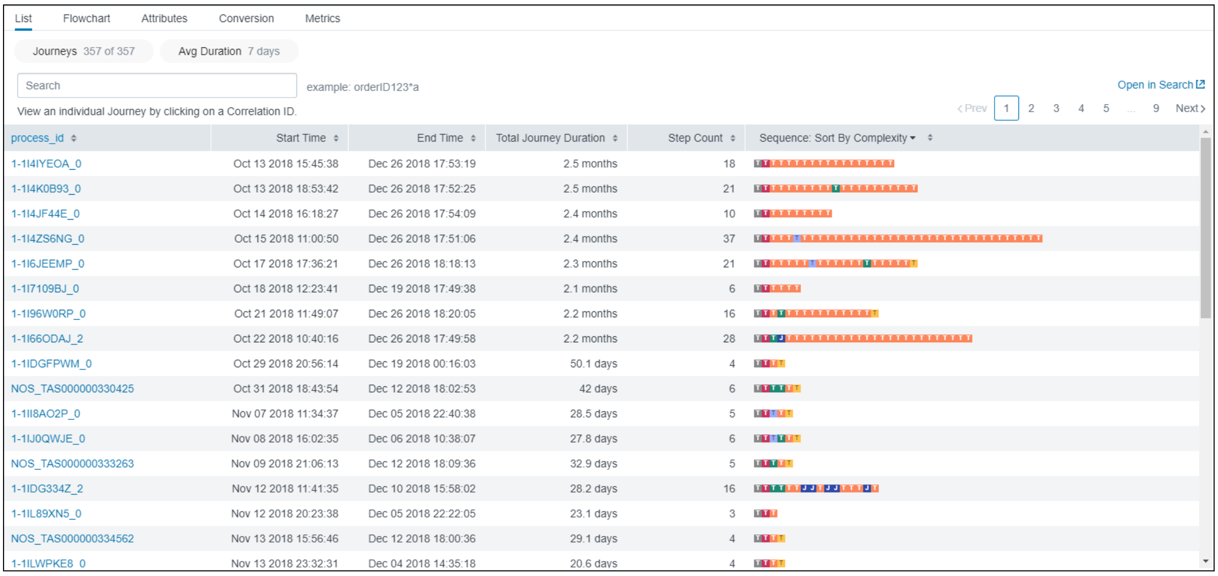


Figure 6.9: Phase 1 - Top cases in Business Flow

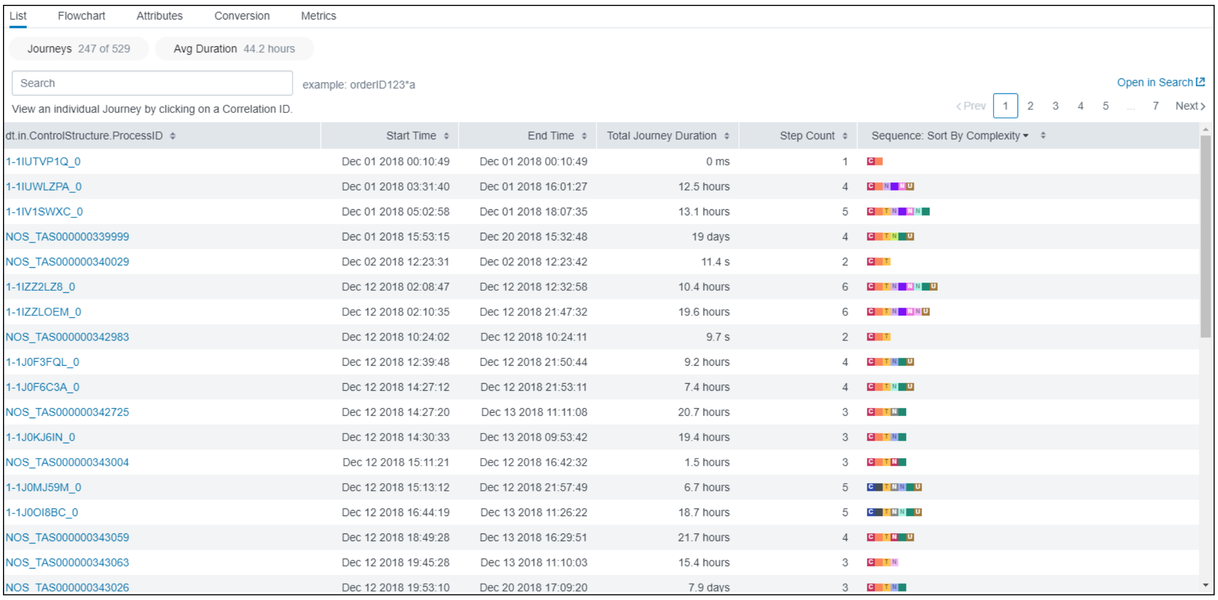


Figure 6.10: Phase 2 - Top cases in Business Flow

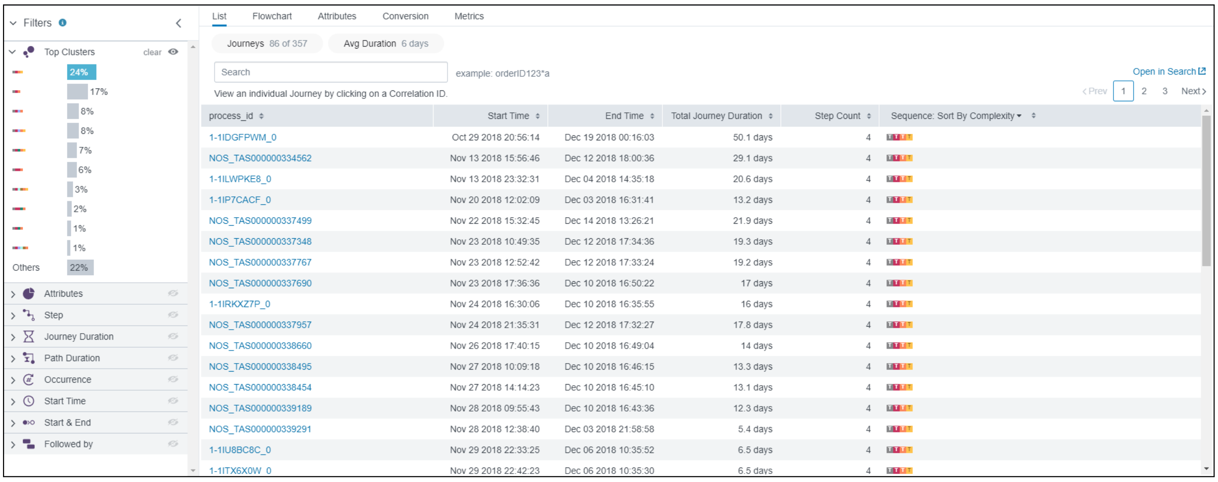


Figure 6.11: Phase 1 - Most frequent case occurrence in Business Flow

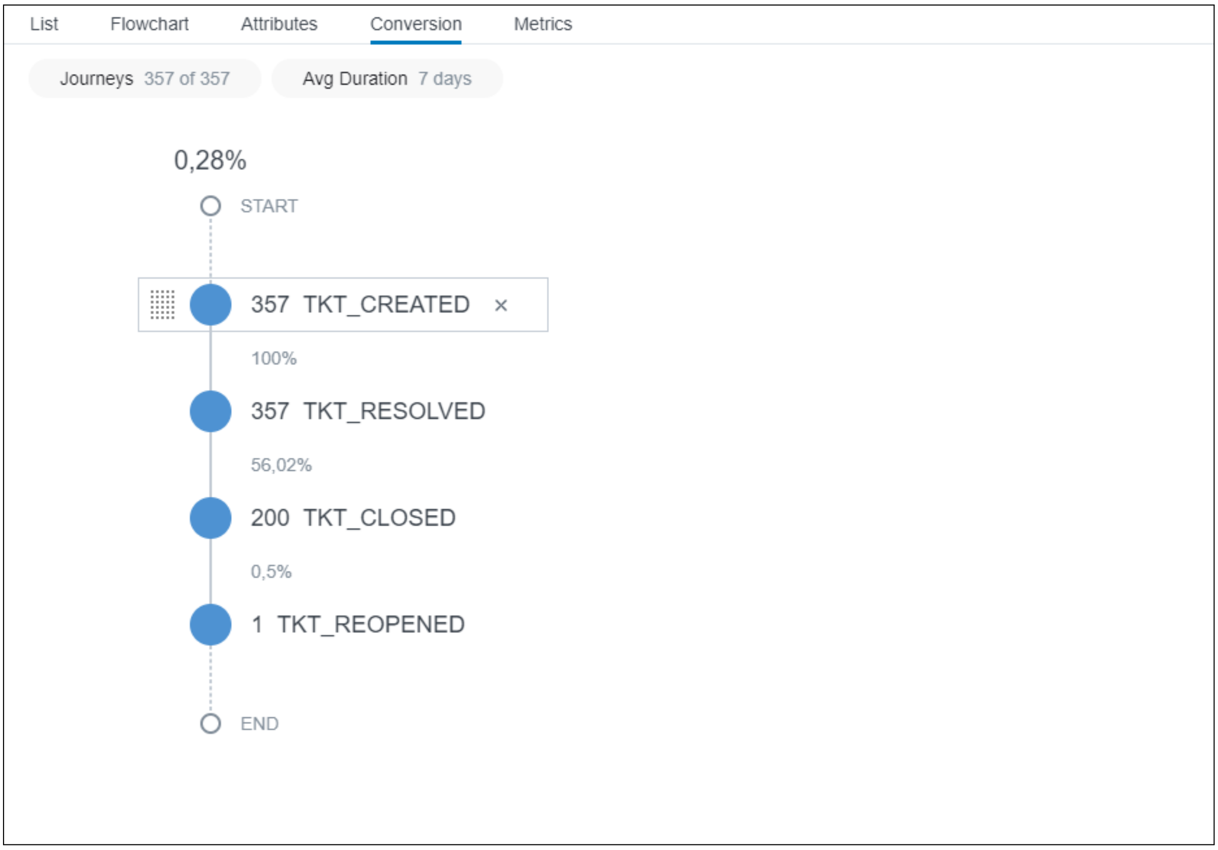


Figure 6.12: Phase 1 - Conversion path in Business Flow

In the Attributes page we can see the analyzes for the first phase showing the distribution of values for each defined attribute. The actionType and actionSubType are the two parameters used on notify and update actions (fig. 6.13). For the actionType we can see that all tickets have a NOTIF_RESOL meaning that all have a reason for their resolution as we have seen previously all tickets got resolved. For this resolution step the reason behind it is shown on actionSubType parameter that can have some of the following values OUTROS, CQE and REP_PROP_MULTIPLO. From this charts we can see that the actionType has the value "FECHAR" 200 times, which reflects the analyzes made in fig. 6.12 that shows 200 process instances got closed.

As we can see on fig. 6.14, when a ticket is created the type and priority of the fault are defined on parameters type and priority. Here we can see the three types of tickets previously presented with 250 occurrences for "Avaria em Acesso de Cliente" that represent faults where a client can not use their equipment at home. Alongside this fault, the most common priority is urgent for a total of 233 occurrences.

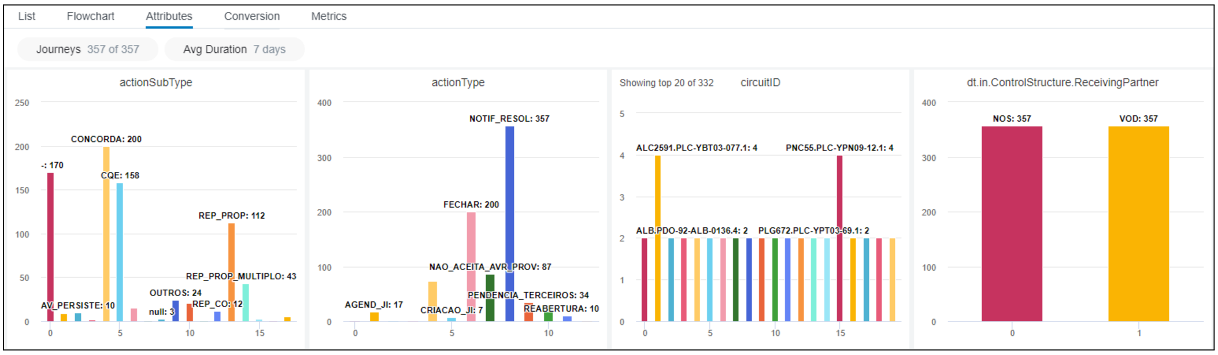


Figure 6.13: Phase 1 - Attributes overview in Business Flow

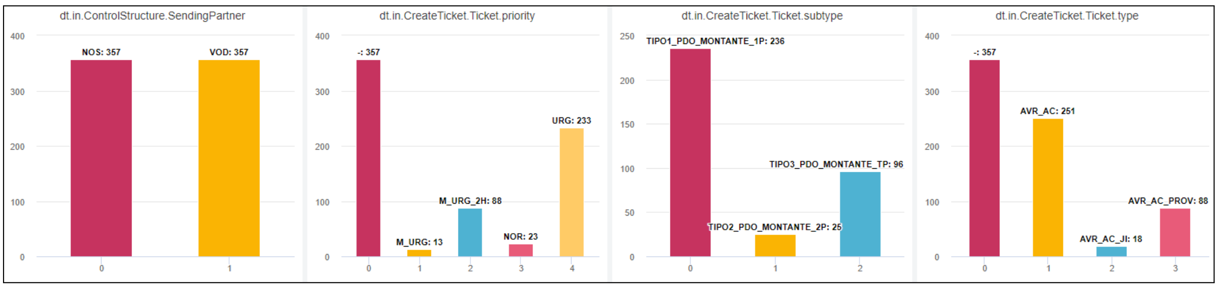


Figure 6.14: Phase 1 - Attributes overview in Business Flow

In the metrics page is shown overall statistics for the process(fig. 6.15 and fig. 6.16), with focus on average duration for a process instance around seven days meaning that a fault ticket on average gets a week to be resolved which is too much since it affects clients, restricting their access to equipment and limiting their life's. Also is shown the total of process instances and their distribution over that period. We can also see an analyzes over the conversion rate showing that only 0.28% of the processes end. This is not a good result, since every organization wants their service to be delivered as fast as possible with the best quality, specially if it has negative impact on peoples life's.

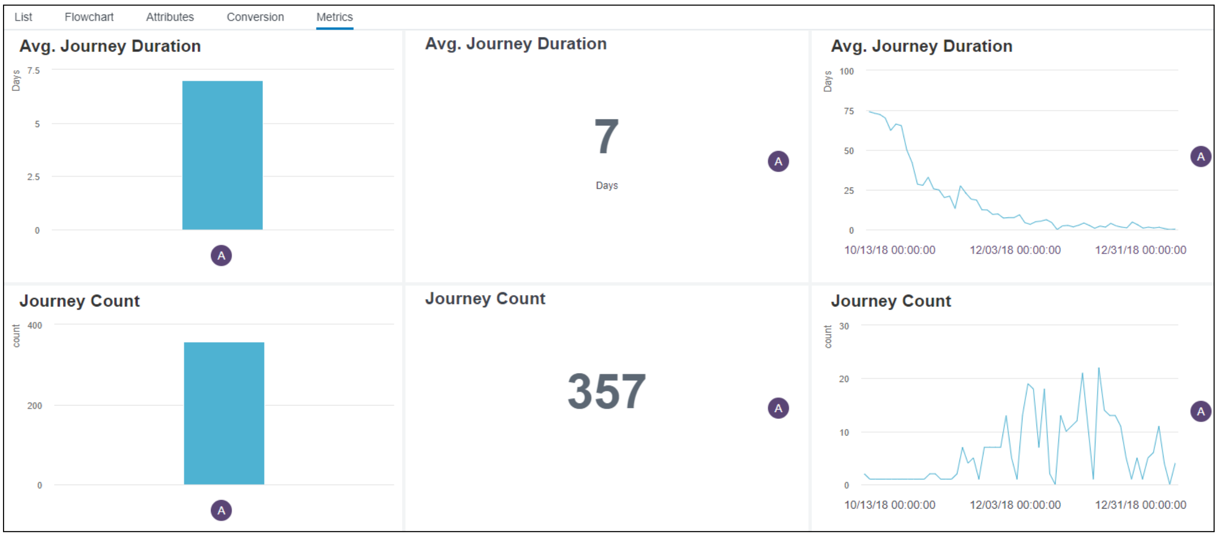


Figure 6.15: Phase 1 - Metrics analysis in Business Flow

Figure 6.16: Phase 1 - Metrics analysis in Business Flow

As shown previously the process model produced by Splunk Business Flow can be filtered by their attributes values and also on FlowChart view is possible to add the option to show step and path count to be shown on the produced model as in fig. 6.17. Here we can see that the most common path for a ticket is their creation, then waiting for the other organization to respond to the request and after this the ticket is resolved and then closed. From this analyzes is also possible to see that the most executed steps are ticket created since it is needed every time a fault occurs and the corresponding resolution to the problem even if sometimes the resolution applied does not solve the problem 100%.

Also on fig. 6.18 we can see the average duration and count for each transition showing that a ticket cancellation is the operation that takes more time related to the overall decision to be made. When a comment is made from one of the organizations usually is an operation that takes 113 hours to be done again, since is the time for the other part to answer on it.

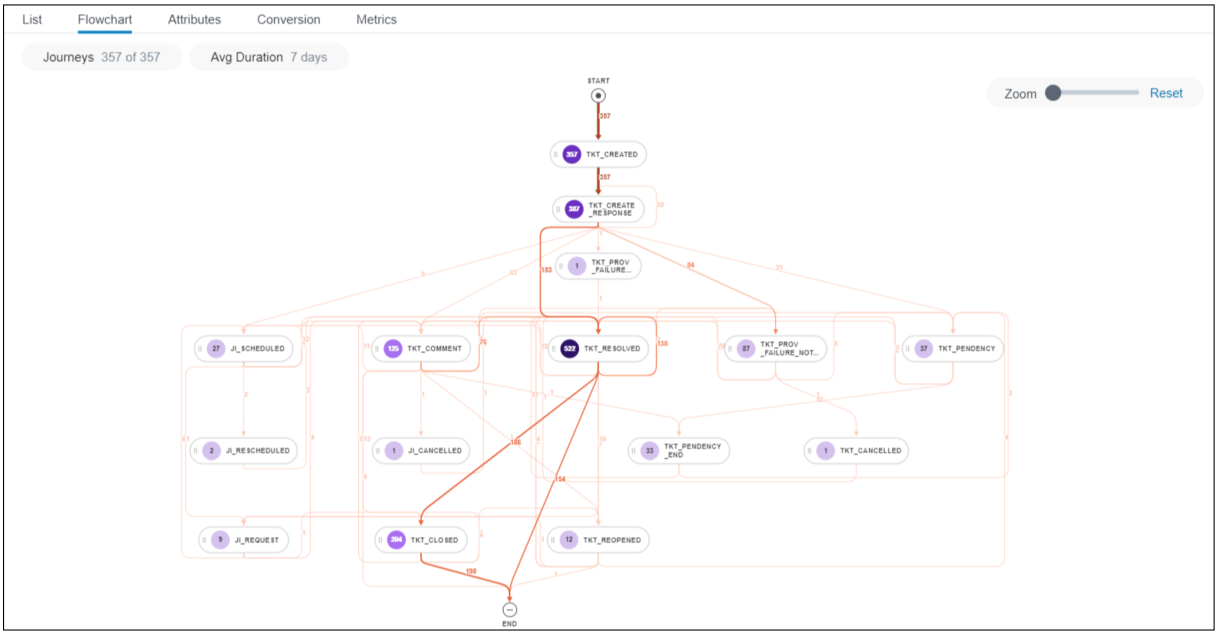


Figure 6.17: Phase 1 - Path and step count analysis in Business Flow

Now we present the statistics produced for the results of phase 2 with focus on the attribute analyzes shown on figures fig. 6.19, fig. 6.20 and fig. 6.21 where we can see similar results to phase 1 with the addiction of distribution of values shown for actionType and actionSubType parameters split by NotifyTicket and UpdateTicket operations.

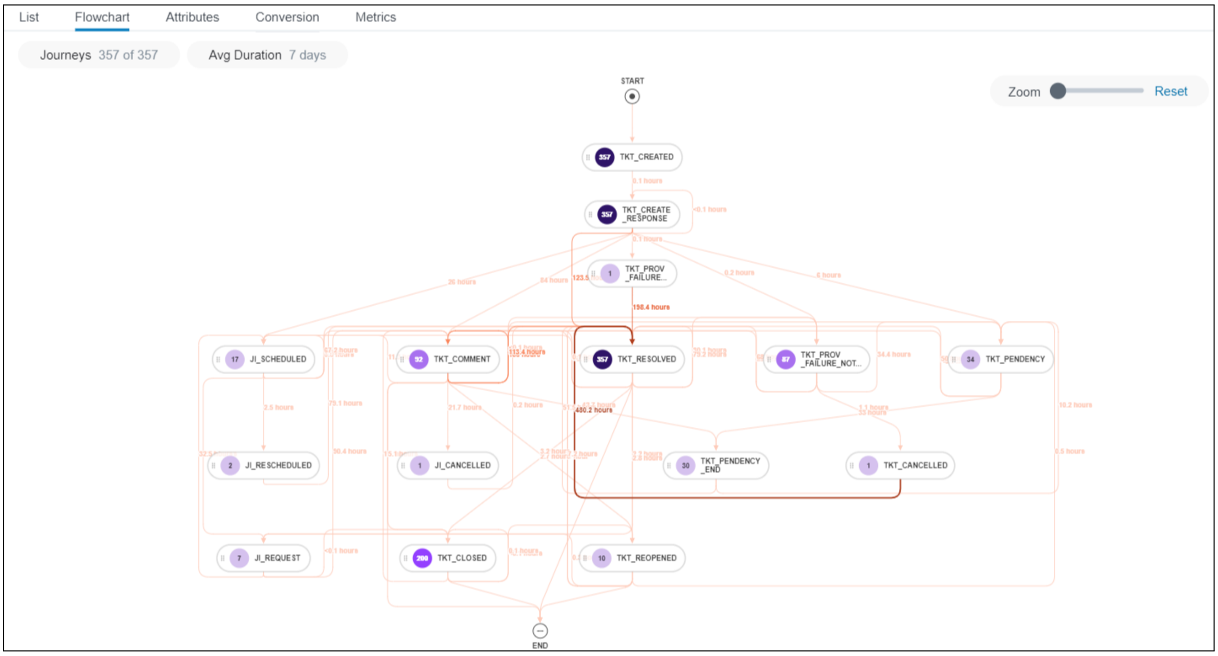


Figure 6.18: Phase 1 - Case and path analysis in Business Flow

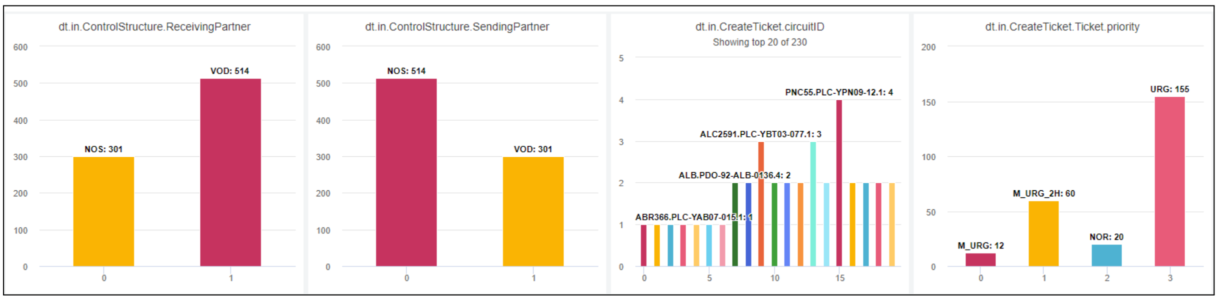


Figure 6.19: Phase 2 - Attributes overview in Business Flow

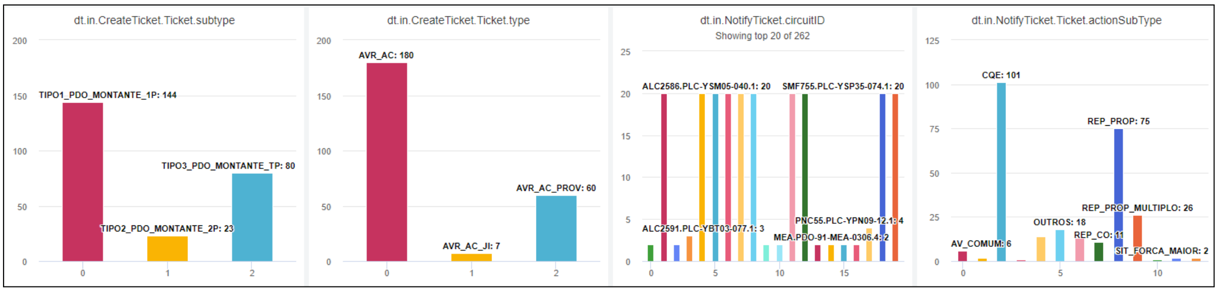


Figure 6.20: Phase 2 - Attributes overview in Business Flow

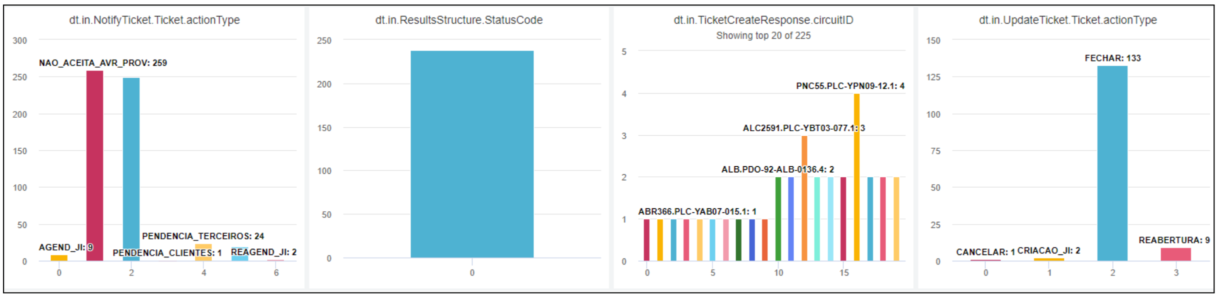


Figure 6.21: Phase 2 - Attributes overview in Business Flow

In this phase the metrics produced for average duration and journey count(fig. 6.22) were different as expected because the data that was used as input was different. In this case, we chose to exclude all NotifyTicket operations that had actionType parameter with value "COMENTARIO" since is not a relevant step to understand the process and how ticket interactions work resulting in a significant reduction on the overall average time for a process instance since it was the slower step to complete as explained before. In this phase we have more journeys because we were less restricted on the criteria to chose what a valid sequence of steps was, considering all process instances that ended with no other limitations.

For this phase we also made an analyzes over the process model(fig. 6.23 and fig. 6.24), showing that CreateTicket that we have seen before as the operation that was always executed as a starting point for the process is now divided in different steps with "Avaria em acesso de cliente" with priority "Urgent" as the most common type of fault. The rest of the steps have similar distribution with focus on NotifyTicket being called 58 times to report situations where fault requests are not accepted. Relative to the average time for the possible paths, we can see that the most time spent is on NotifyTicket steps since most of the time reflects a decision made by one of the organizations based on the received ticket information.

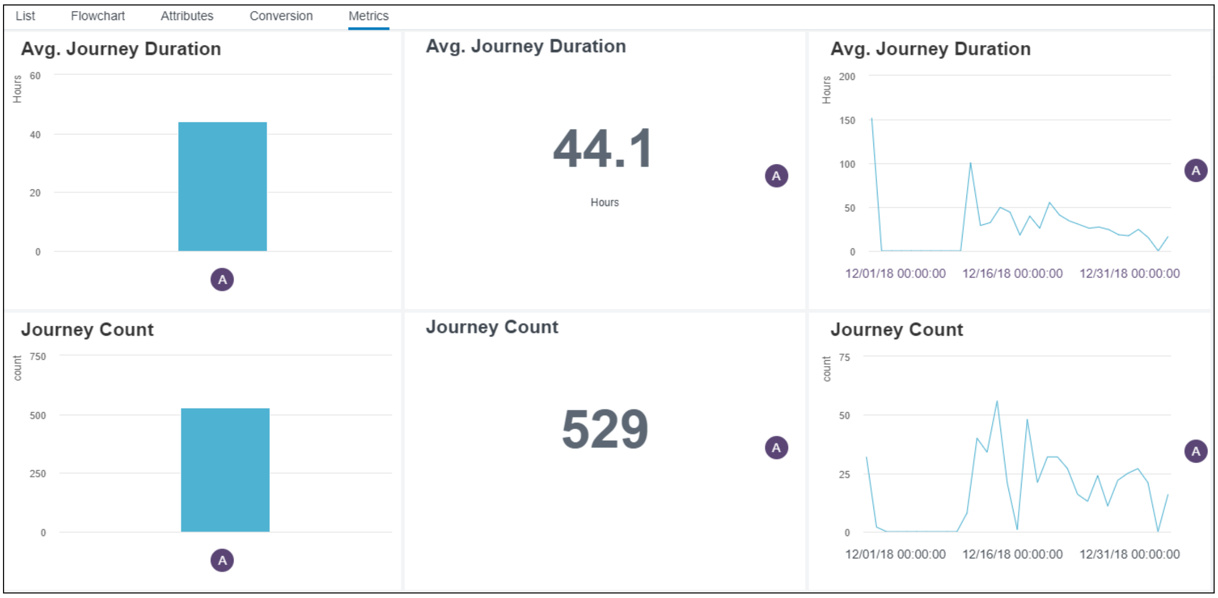


Figure 6.22: Phase 2 - Metrics analysis in Business Flow

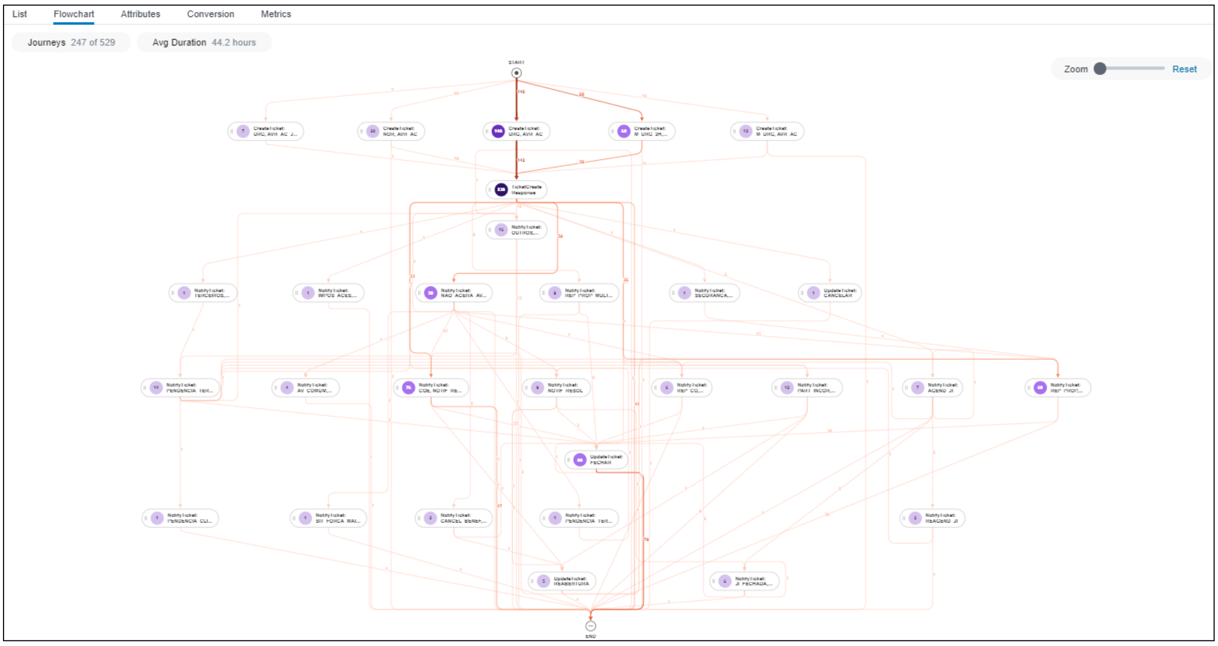


Figure 6.23: Phase 2 - Path and step count analysis in Business Flow

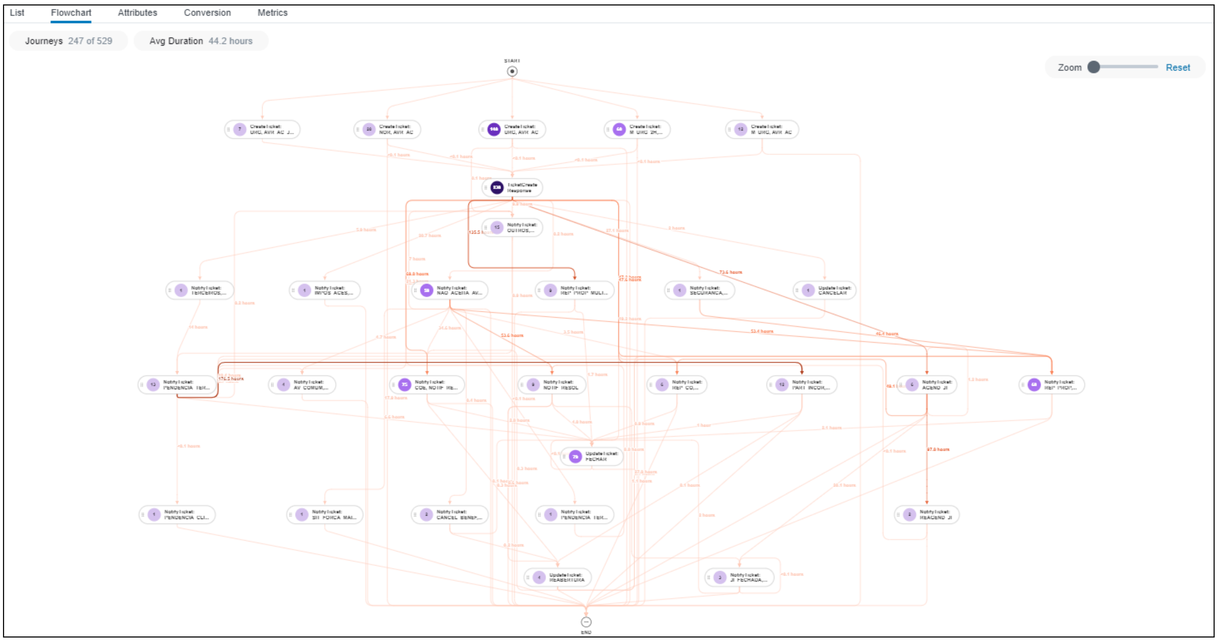


Figure 6.24: Phase 2 - Case and path analysis in Business Flow

6.3 Summary

In this chapter, we presented the results of applying Process Mining in the context of the FTTH_SHARE project to produce a process model using Business Flow Tool. The different ticket processes were presented alongside some of the most important metrics. After showing and analyzing the results of the process model produced, we compared metrics for both phases of the solution related to the overall count and duration of process instances. In the end, we presented a more detailed model with information about each step and path count as well as the respective duration.

7

Conclusions and Future Work

Contents

7.1 Conclusions

This document addressed a problem related to process discovery between two well known organizations related to Fibre to the Home share service. The process in case was about a fault management system based on tickets and the current solution implemented in Splunk was based on a configuration table that described the process, updated always by human intervention. We used Splunk Business Flow tool from Splunk which is a an application used for Process Mining that enables using logs previously indexed in Splunk to run searches to apply transformations on data before producing the respective process models and metrics over it. Since the logs were already in Splunk we used this information with the necessary transformations to produce the corresponding process model that better described the fault process and its different possibilities.

We proposed a solution for three different types of tickets "Avaria em Acesso de Cliente", "Intervenção Conjunta" and "Serviço Cliente Dectado em Provisão", by correlating events based on their process instance. The solution was divided in two phases, first using already processed logs as a result of an engine programmed on Splunk that categorizes the events and correlates them resulting on only valid processes. Then we opted for a more complex solution using the original logs that had all interactions between the two organizations and with this information we produced custom step names composed of the operation name plus the parameters used on each API call. In the end, we produced some analyzes over the process model with focus on average duration and count for the corresponding process instances and steps.

During this period where we had to use Splunk Business Flow not everything went as planned, the first steps before we could actually start using the tool and see some results were the most difficult and where the most time was spent because we had to find the correct period of data to use and apply the necessary adjustments to the corresponding events. Another obstacle during the implementation was that the software license ended before we could export the models with better quality or even go deeper on the analyze over the produced models and metrics. Since we lost access to Splunk Business Flow tool right after the implementation phase we could not proceed with the evaluation, where the plan was to compare the produced models with the configuration used for each ticket process.

7.2 Future Work

we believe the results we got would serve as a base for future work after going through evaluation phase for discovering the process with more detail. In this work, we focused only on three types of tickets and discovering the respective processes but with what was produced this solution could be extended to other fault types, meaning we could have multiple combinations only by adding the corresponding operation names and their field names to the macro parameters used in the search.

One feature that we see as a starting point for future work is to use the process models produced and transform it on the corresponding process definition that could be imported and used as initial configuration to the process this would reduce human intervention to almost zero, preventing possible errors when updating it and also would enable constant process adjustments. This could be done by investigating how the tool produces the model and if it saves the corresponding steps and transitions on an output file.

Since we had to use Splunk because the logs and the corresponding engine to process the events was already implemented there, it would be interesting to try one of the tools presented in Section 3 in specific Disco since it enables the possiblity to export in XML format and that would facilitate the transition from the process model to a definition that could be imported to the table that serves as initial configuration to the process.

Bibliography

[1] A. K. A. de Medeiros and W. V. der Aalst, “Process Mining towards Semantics,” pp. 35–80, January 1970. [Online]. Available: http://www.padsweb.rwth-aachen.de/wvdaalst/publications/p485.pdf

[2] “BPMN models,” https://sparxsystems.com/enterprise_architect_user_guide/14.0/model_domains/bpmn_1_4.html, accessed: 2021-06-11.

[3] K.-T. Chen and R. M. Lee, “Schematic evaluation of internal accounting control systems,” 1992. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.125.4644&rep=rep1&type=pdf

[4] M. Weske, *Business Process Management Concepts,Languages,Architectures.* Springer, 2007, 978-3-642-28616-2.

[5] S. Guerreiro, “Introduction to Business Process Automation,” 2017.

[6] W. V. der Aalst, “Process mining: Overview and opportunities,” *ACM Transactions on Management Information Systems*, vol. 3, pp. 7.1–7.17, 07 2012.

[7] D. R. Ferreira, *A Primer on Process Mining Practical Skills with Python and Graphviz.* Springer, 2020, 978-3-030-41819-9.

[8] “proM Log filter - end event,” https://www.promtools.org/doku.php?id=tutorial:preprocessing, accessed: 2021-06-29.

[9] “Important concepts in Splunk Business Flow,” https://docs.splunk.com/Documentation/SBF/-Latest-/Tutorial/concepts, accessed: 2021-05-19.

[10] E. Brzychczy, P. Gackowiec, and M. Liebetrau, “Data Analytic Approaches for Mining Process Improvement—Machinery Utilization Use Case,” *Resources*, vol. 9, p. 17, 02 2020.

[11] M. Dumas, M. L. Rosa, J. Mendling, and H. A. Reijers, *Fundamentals of Business Process Management.* Springer, 2018, 978-3-662-56508-7.

[12] S. White and D. Miers, *BPMN Modeling and Reference Guide: Understanding and Using BPMN*, ser. Business Process Management Process Modeling. Future Strategies Incorporated, 2008, 978-0-977-75272-0. [Online]. Available: https://books.google.pt/books?id=0Z2Td3bCYW8C

[13] W. V. der Aalst, "The application of petri nets to workflow management," *Journal of Circuits, Systems, and Computers*, vol. 8, pp. 21–66, 02 1998.

[14] A. Weijters, W. Wil Van der Aalst, and A. Alves De Medeiros, "Workflow mining: discovering process models from event logs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1128–1142, 2004.

[15] A. Weijters, W. Wil Van der Aalst, and A. Alves De Medeiros, *Process mining with the HeuristicsMiner algorithm*, ser. BETA publicatie : working papers. Technische Universiteit Eindhoven, 2006, 978-90-386-0813-6.

[16] A. K. A. de Medeiros, A. J. M. M. Weijters, and W. V. der Aalst, "Genetic process mining: an experimental evaluation," *Data Mining and Knowledge Discovery*, vol. 14, pp. 245–304, 2006.

[17] C. W. Gunther and W. V. der Aalst, "Fuzzy mining – adaptive process simplification based ¨ on multi-perspective metrics," pp. 328–343, 2007, business Process Management, 5th International Conference, BPM 2007, Brisbane, Australia, September 24-28, 2007, Proceedings. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.81.1207&rep=rep1&type=pdf

[18] W. V. der Aalst, H. A. Reijers, and M. Song, "Discovering social networks from event logs," *Information Systems*, vol. 14, no. 6, pp. 549–593, 2005.

[19] W. V. der Aalst, H. Reijers, A. W., and Others, "Business process mining: An industrial application," *Information Systems*, vol. 32, no. 5, pp. 713–732, 2007.

[20] J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens, "A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs," *Information Systems*, vol. 37, no. 7, pp. 654–676, 2012.

[21] jan Mendling, *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness.* Springer, 2008, 978-3-540-89224-3.

[22] W. V. der Aalst, *Process Mining - Discovery, Conformance and Enhancement of Business Processes.* Springer, 2011, 978-3-642-19344-6.

[23] B. van Dongen, A. de Medeiros, H. Verbeek, A. Weijters, and W. V. der Aalst, "The ProM Framework:A New Era in Process Mining Tool Support." [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.80.9950&rep=rep1&type=pdf

[24] D. T. Blickle and D. H. Hess, “Automatic Process Discovery with ARIS Process Performance Manager (ARIS PPM) “It’s about behavior!”,” October 2010. [Online]. Available: http://cdn.ariscommunity.com/documents/urelation/SAG-AEP_PPM-Automatic_Process_Discovery_with_PPM.pdf

[25] C. W. Gunther and A. Rozinat, “Disco: Discover your processes.” [Online]. Available: http://ceur-ws.org/Vol-940/paper8.pdf

[26] O. Anesini, “Using process discovery on business process management (BPM) through BPMN,” June 2020. [Online]. Available: https://www.bonitasoft.com/news/use-process-discovery-bpm