A data-driven approach for generating insights into software development

Rexhepi, N.

Award date:
2017

Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science
Department of Industrial Engineering and Innovation Sciences

A data-driven approach for generating insights into software development

Master's Thesis

Nderim Rexhepi

Supervisors:

prof.dr.ir. Uzay Kaymak
dr. Anna Wilbik
dr. Alexander Serebrenik
ir. Edwin Geritz
ir. Martijn Kabel

in partial fulfillment of the requirements for the degree of
Master of Science in Business Information Systems

Eindhoven, August 2017

Abstract

Data analysis has become an integral function for all organizations due to the multitude of generated data and benefits that it can provide. One important field of application is that of software development, where information is constantly generated and accumulated. In the scope of software production, there are various issues that arise due to functionality or incompatibility of applications with client needs. To manage the process of resolution, workflow management systems are in place. Though these systems provide the means to record the procedures and investigation to a certain extent, no systematic methodology for analysis of this data is in place.

The current literature lacks a well-structured approach for implementing a thorough data analysis on ticket-resolution systems while considering all important metrics for software development. In this master thesis, we address this research gap by investigating the theory and applying suitable practices to our case-study. We investigate three different data analysis approaches and assess the generated insights with regard to validity and relevance to improving software development performance. Our single most important metric in a fast-paced environment such as software engineering is time, followed by compliance and other important distribution elements. By applying the series of approaches, we comprehensively determine the optimal approach based on process characteristics and stakeholder preferences. The result of this project is a practical investigation that can be used for bench-marking and as reference in choosing the right data analysis method for similar future projects.

Keywords: Data analysis, Process Mining, Linguistic Summarization, Software Process Improvement

Acknowledgments

This thesis presents the results of my master graduation project in Business Information Systems at the Eindhoven University of Technology. The research falls under the Information Systems research group. The practical implementation of the research was carried out within a software development environment. It has been a challenging but rewarding journey with many unknowns and a lot of support from my closest ones.

First, I would like to thank my first supervisor, professor Uzay Kaymak for his guidance throughout the project. During our meetings it was possible to focus on the essentials of the research and establish the right study objectives. I would also like to thank my second supervisor prof. Anna Wilbik for her constructive help and ideas for the research. Third, I would like to thank prof. Alexander Serebrenik for accepting to be in my assessment committee.

During this graduation project, I had the opportunity to work among an amazing group of professionals. I would like to express my gratitude to Martijn Kabel and Edwin Geritz for their support during all stages of the project. I would also like to thank my fellow graduates, Shamil Mustafayev and Santiago Ruiz for their support.

Finally, but most importantly, I am eternally thankful to my family and girlfriend Mirela, for their years of support and motivation. You truly made this experience memorable.

Nderim Rexhepi August 2017

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This introduction chapter provides the context and setting of the research. The research context and motivation, problem, objective and questions are described. Lastly, the report outline concludes the introductory part.

1.1 Context and Motivation

As software is developed and rising issues are patched, developers, testers, engineers, managers and other stakeholders share continuous efforts to deliver updated state-of-the-art software to the market.

In practice, the development cycle of software engineering in this organization leans towards agile methodologies. A series of short-term sprints are organized to develop and address software issues only to be concluded with a software release. Due to the robustness and application, the methodology of agile software development is in full effect as the main catalyst of handling complex software projects and numerous client requests. As new features are developed and problem reports pile up, it is challenging to prioritize tasks and resources without concrete evidence on the potential impact. The organization utilizes an advanced ticket-handling platform to manage the submission and resolution process of issues. Though this platform serves the handling and resolution purpose, it is arguable whether the generated data is used to its full extent in terms of performance measurement.

Managers and other stakeholders at different levels in the organization are dependent on business decision support tools to make informed business decisions. This is enabled through the field of business intelligence and analytics commonly referred to as the techniques, technologies, systems, practices, methodologies, and applications that analyze critical business data to help an enterprise better understand its business and market and make timely business decisions [1]. However, as data congregates and information requirements change, business intelligence approaches become more time-consuming and expensive to deploy.

Traditional data analysis with statistical methods and visualization often retrieve information through pre-defined procedures which suffer [10] from such an environment with high variation. In multiple industries, the data is analyzed, visualized and described in various ways. This variety in interpretation and data types, coupled with differences among users makes it difficult to understand information gained with traditional methods. Even though a great deal of automation in Business Intelligence has been researched on to increase efficiency and standardization [34, 24], the information produced with such methods may not serve the requirements of all its users. A standardized approach that is easily comprehensible by the average user is necessary to provide consistent business-intelligent information.

One alternative approach for the generation of relevant insights for today's information systems is that of business process mining. As organizations make use of workflow management technology, systems that have a good description of the process in terms of a process model [45] can make great

use of such mining techniques. Business process mining make use of the event log information and provide insights in terms of the process, control, data, organizational, and social structures [3].

With the large amount of data, it is still a challenge to extract potentially useful information in an efficient way. Furthermore, the challenge of human interpretation is an important variable to consider when dealing with data summaries [6]. The insights extracted using either visualization approaches or process mining ones can be often challenging to understand for non-technical users. That is why we also consider a third and final approach, that of linguistic summarization.

Linguistic summarization allows for a high degree of automation and convenient summarization of data [23]. It is a technique that allows for analysis and insights of large amounts of data, with the main advantage being its ability to describe data in a human-language structure. It can be used to summarize large sets of data into simple sentences that provide key information on performance, quality or status of processes. This technique can potentially generate an additional layer of insights throughout software development processes.

In this research, we conduct a three-fold comprehensive investigation of the software defect-management data-set, including process event data. We first apply traditional data analysis approach to generate data visualizations, followed with process mining methods, and lastly with a linguistic summarization technique. Through the use of these methods, we ensure capturing of the important elements of the data-set and process, while also evaluating and comparing to determine which approach is most appropriate in this domain.

1.2 Problem Statement

Software development performance is a pervasive quality difficult to understand and measure, because it is affected by multiple aspects of design, code and environment. Performance is a significant issue in a large number of projects. It can cause delays, failures on deployments, and even abandonment of projects [53].

A study conducted with information technology executives in [53] shows that software projects face issues regarding performance in close to 20 percent of released products. The field of Software Performance Engineering (SPE) attempts to address these issues by investigating the key performance indicators (attributes) that best represent the performance.

However, there is no existing methodology or application that systematically shows what data analysis methods are appropriate in extracting such performance indicators in the software development domain. So far, only descriptive statistics — and to some extent inferential analysis — have been used to improve processes in terms of decision-support [46], and it is unknown whether that is the most valid approach. Despite the large amounts of data collected from software projects, many organizations struggle to extract, analyze and make use of this data for decision-support [23, 46]. To address this industry need and research gap, we propose a methodology that generates and evaluates insights within the domain of software development.

Problem statement: "In software development, there is no comprehensive methodology for generating insights in defect-management systems"

1.3 Research Goal

The goal and scope of this study is to develop a methodology for generation of insights into software development defect-management processes. The study will provide a series of data analysis approaches and validation of their usability in the context of software development. By providing a systematic approach for the extraction of software issue insights, we provide a means of direct investigation into the software development processes. For the scope of this study, the approaches are limited to three major data summarization techniques — visualization, process mining and linguistic summarization.

1.4 Research Questions

Based on the research goal of the study, the main research question is as follows:

- **How to generate insights into software development defect-management processes?**

The following sub-questions are defined and then investigated in order to answer the main research question:

- What qualifies as an insight in software development?

An "insight" is defined as *the capacity to gain an accurate and deep understanding of someone or something.* In our study, we use the common metrics elaborated in the field of software performance engineering as a benchmark to generate these insights. By measuring the available metrics and discussing them with software develpoment professionals, we reach the conclusion of what qualifies as useful insights.

Due to the focus on defect-management resolution process, our case study dataset limits the collectible metrics which can be turned into valuable insight for the involved stakeholders. Therefore the generated insights provide decision support in the terms of time, issue distribution and resolution process compliance.

- What data analysis methods can be used for generating such insights?

There are many data analysis methods that can be used to summarize data. In the context of software development, performance metrics regarding lead-time, process compliance and issue-specific attributes are the most significant figures [53]. A thorough study by Poncin, Serebrenik and van den Brand [38] provides with meaningful metrics that are necessary to analyze in a software development context. By investigating this data from a quantitative and qualitative point of view, it is possible to generate overviews and important information regarding the performance of the resolution process.

Our efforts focus on investigating and generating useful insights into the issue resolution process by applying data visualization approach, process mining and lastly linguistic summarization. Through the use of these methods, we ensure to capture important elements of the data-set and process, while also evaluating and comparing to determine the more valid and resourceful approach.

- What are the results on the case study?

After we have determined the methods of investigation, we apply the approaches in a case-study with real data from a software development environment. The data-set contains significant attributes captured by the issue resolution tracking system of which we generate insights based on the industry-preferred metrics in the field of software development.

- How to validate and evaluate insights with stakeholders?

After we have generated the insights from the data-set, we validate our findings with the organization stakeholders to determine their usefulness, relevance and validity. This step is crucial in determining whether the quality of our assessment meets the needs of the organization, as well as validation to check if our findings match with the perception of the stakeholders.

- How do the approaches compare?

Lastly, after the validation and evaluation steps, we investigate the differences of generated insights from the methods, and provide recommendations based on each approach. This comparison as well as conclusions from the previous questions help finalize a concrete answer to the research question and provide conclusive remarks.

1.5 Methodology overview

A systematic five-step approach was used to execute the study. The approach was motivated by the design-science paradigm in information systems as described in Hevner et al. [16]. The correlated activities and references to the chapters are described in 1.1.

Table 1.1: Project stages and main activities mapped to chapters

Stages	**Activities**	**Report Chapter**
Orientation	Case-study introduction, Research gap	Chapter 1 & 2
Analysis	Methodology approach, Data collection & evaluation	Chapter 3
Design	Data summaries, Process mining & simulation, Linguistic summaries	Chapter 4
Validation	Validate findings, Formal interviews, Compare approaches	Chapter 5
Implementation	Present & compare findings, Conclusion & future work	Chapter 6

1.5.1 Orientation

The first step of the research was to understand the project nature, and define clear objectives of the study. Meetings with the organization supervisor and relevant stakeholders were conducted to check what area of analysis can provide useful insights for the software development teams. During this initial stage, it was necessary to organize and discuss project objectives, define the necessary resources and privileges to conduct the study. Furthermore, it was important to establish the main contact resources to collect the static information regarding the issue resolution process.

1.5.2 Analysis

Analysis and conversational evaluation of data sources. Through meetings and discussions, it was determined which data resources are valid and can be used to reach our objectives. During this step, we also conducted exploratory literature review to deepen our knowledge into the topic, understand and acknowledge the state-of-the-art approaches in data analysis. Data collection from the workflow management system was also conducted, followed by descriptive analysis of the content. Due to numerous sources of information and noisy variables, we select the most valid ones in consultation with organization professionals.

1.5.3 Design

We proceed with the design and application of our data analysis methods - traditional, process mining and linguistic summarization. We use a combination of existing solutions for data analysis, process mining and develop a linguistic summarization environment in Matlab. During this stage it is possible to generate early results, analyze them and drill-down on produced insights.

1.5.4 Validation

We perform validation in two phases. First, we share our results on a continuous basis with the organization stakeholders, upon which the findings are validated. After multiple iterations and consultation with informed process owners, we establish the final results. Second, we evaluate and validate the findings of each analytical method through interviews in the later stages of the study.

1.5.5 Implementation

The generated insights, validation feedback and final remarks are summarized in written and presented to the organization. Furthermore, the tools and environment used to generate the insights are made available for use to the organization and future researchers. A thorough methodology of our approach is elaborated in 3.1.

1.6 Report Outline

This section describes the structure of the study. The report starts with the introduction chapter that provides the motivation and context on the topic, presents the problem of focus, the research objective and the research questions. In Chapter 2, the review of literature is presented along with the established research gap of the study. Chapter 3 provides the research methodology used in this report. Chapter 4 provides the case-study results for the application of data visualizations, process mining and linguistic summaries. In Chapter 5, we evaluate and validate the findings with the organization stakeholders. Chapter 6 presents our conclusions, recommendations, limitations and future research areas.

Chapter 2

Literature Review

Literature analysis enable the exploration of the topic of research and enable its application in our study. This review is aimed at exploring the methodologies for generating data summaries in software development, with emphasis on the topic of linguistic summarization in computer science literature specifically within the domain of Information Systems. We present the current status of software development data insights, and focus on major definitions and concepts in the field of data visualization, process mining and of linguistic summarization. We use the systematic literature review (SLR) guidelines by Kitchenham as described in detail in Appendix A.6.

In line with our main research question, we structure our literature based on the following key questions:

- What metrics are important in software development?
- What common approaches for data analysis/summarization exist?
- What are the common uses of visualization, process mining and linguistic summarization?
- How is linguistic summarization represented in the software development domain?

Through these questions, we seek to collect concepts and recent developments in the industry in order to motivate our study objectives.

2.1 Data analysis in Information Technologies

Rapid advancements in information technology have been trailed with large amounts of data across multiple domains. As the amount of data increases, it becomes challenging and time-consuming to understand data in its original (raw) format. To deal with this issue, various data mining approaches for summarization have been developed.

According to Wu et al. [54], there are two main classes of data-set summarization: numerical summarization and linguistic summarization. The traditional approach of summarization with descriptive statistics provides meaningful information on the measures of central location of data (mean, median, mode), variability (range, standard deviation, coefficient of variation), relative standing (percentiles, quartile), relationship (covariance, correlation) and more. Information is then processed and can be shown graphically using charts, bars, and lines. It is a common practice among many industries and provides valuable insights into numeric data sets and time series data [35].

The limitations of the technique surface when the data in question is stored in an unstructured format, or is simply non-numerical data. This scenario is a commonality in software development processes as various log files are generated; data is stored, but often at the cost of highly unstructured formats and storage environments. Analysis of such data with traditional descriptive statistics requires extensive efforts to conduct. Furthermore, descriptive statistics are efficient

at analyzing quantitative, highly structured data, which is less common in the field of software development.

2.1.1 Use of analytics in organizations

In all industries around the world, digital information has become an integral part of their business decisions [28]. However, the massive amount of information that is collected is not always being used in generating useful insights. As technologies for collecting and analyzing data are becoming more frequent, organization leaders are questioning what is the best approach in gaining value from such systems.

A study carried by MIT Sload Management Review and IBMs Institute for Business Value with nearly 3,000 executives, managers and analysts from multiple industries, shows that data analytics is crucial in the performance of organizations [28]. The most top-performing organizations use analytics five times more than lower performers. Furthermore, the same study reveals that improvement of information and analytics is a top priority in their organizations, emphasizing the market need to develop improved solutions and generate more insightful approaches for the future.

Data analytics as a concept revolves around the concept of analyzing a set of records (actions, events), to make inferences about the general dataset (population). The most frequent insights gained from such analysis are presented in the forms of visualized charts, graphs or images. The observed charts and graphs are referred to as functional data and statistical methods for analyzing them are described as functional data analysis (FDA) [40]. In essence, the goals of data analysis are:

- Represent and transform data in ways that aid analysis.
- Display data so as to highlight various characteristics.
- Study important sources of pattern and variation among the data.
- Explain variation in an outcome or dependent variable by using input or independent variable information.

However, data analysis can provide a limited view over a certain data-set, and a more exploratory approach is needed. That is where data mining techniques provide the additional prowess and aid in recognizing various patterns in data, through mathematical and computational algorithms. In essence, data mining is a pattern discovery approach, involving a combination of methods such as machine learning, statistics and database systems. [9] It is an approach frequently favored in business, science research and even government security [12].

Based on the type of stored data, alongside data mining techniques, there exist other mining approaches which deal specifically with text (text-mining), process (process-mining) and web (web-mining) etc. Much research has been done in this context, and the specific techniques are thoroughly review by Linoff et al. [30]. Their application in todays industries is an integral characteristic into making better and informed decisions.

2.2 Why data analysis, process mining and linguistic summarization?

Our case-study is a large organization that already has numerous analytical platforms in place for the improvement of processes and activities, although our analysis will deepen the context and provide future recommendations on the suitable approach. The combination of visualization of information, metrics on the performance of processes and finally linguistic summarization of the data will provide sufficient insights for any organization.

Significant literature exists with regard to the methodology of data visualization [11, 29, 25, 18, 43] and process mining [32, 4, 48, 13, 2]. This literature is aimed at concepts and methodologies

necessary to expand knowledge and use of analytics in multiple industries. Since we elaborated on these concepts briefly in sections 2.3 and 2.4, we shift our attention towards the area of linguistic summarization.

2.3 Descriptive Statistics

Analysis of data has been an important element in biology, economics, engineering and countless other disciplines [35]. The preferred approach to deriving insights from these fields has commonly been that of statistical methods, or more particularly descriptive statistics. Using this technique, organizations are able to extract information of key performance indicators, making use of large stored data sets. Most frequently, analysis is done on exploring the relationships between attributes, grouping the data, identifying patterns and trends, building regression/classification models and describing such findings statistically [33].

These descriptions are often presented in the form of graphs, tables or similar visualization techniques, only then to be interpreted in text. This poses a challenge regarding the gained insights, as the visualizations are often not interpreted uniformly by different users. The issue of interpretation makes chart visualizations an unreliable method, as users may perceive the findings different from the original intent [27].

With the increasing load of data in the industry, it becomes challenging to manually describe statistical findings in a textual format. Businesses are seeking automated methods that present information in meaningful formats in order to support their decision making in near-real-time.

2.4 Process Mining

The nature of different industries requires a specific approach. When dealing with process data, a common approach is that of process mining. In essence, process mining is built on two pillars (a) process modeling and analysis and (b) data mining [1]. The continuous growth of stored data is a key driver for popularity of data mining and process mining approaches. According to van Dongen [45], organizations have various data, and more frequently in the form of workflows/processes, a type of event log that stores information about when and what activities were carried out during a timeframe . This information is then analyzed with process mining or data summarization/visualization techniques to provide insights into the process or variables that are significant to the context.

The output of this analysis can be expressed in terms of rules, clusters, tree structures, graphs, equations, patterns, etc. [1]. The extent to which these features are useful, is dependent on the final user. Many times the context is business-oriented, and not all final users are tech-savvy. Therefore the presumption that these extracted insights will be understood is a common mistake. That is why research is continuously conducted in an effort to bridge the gap of technicality and provide insights into explicit forms.

One significant problem that visualization and process mining insights bring is that of interpretation. The challenge is to extract potentially useful knowledge not only in an efficient way but also in a way that could be understandable by humans [6]. An alternative to this issue is that of linguistic summarization, aim of which is to produce easy-to-understand summaries from data in a human-language structure (sentence). This method is receiving great amount of attention due to the ever-increasing amount of complexity in data systems and diversity of the data interpretation (final users). By addressing these issues, this approach can be applied in various industries with great success and with an extensive amount of automation.

In order to be suitable for process mining, our data-set should conform to the process mining event log meta model in Figure 2.1. An event log is comprised of a number of processes (usually one) [38]. Every process contains a series of instances that are unique. Lastly, every process has activities with information on when it was executed.

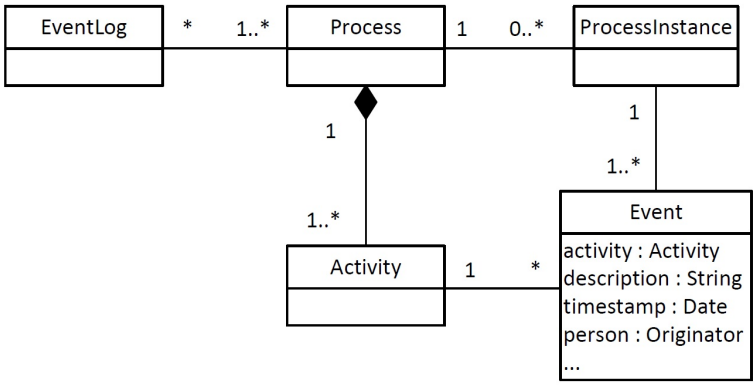


Figure 2.1: Process Mining meta-model.

For our purposes, we can use process mining to derive representations of process control flow, detect relations between activities and in event logs with resources (person), we could detect relations between them [38].

2.5 Linguistic Summarization

Linguistic summarization was enabled through the utilization of the concept of fuzzy sets (FS) by Zadeh [60], into implementable form by Yager [56, 57, 58, 59] and later developed into other domains by renowned scientists Kacprzyk [19, 20, 21, 22] and recently Wilbik and Kaymak [51, 50].

Wu et al. [54] describes linguistic summarization as a data mining or knowledge discovery approach to extract patterns from databases. This basic definition does not fully capture the extent of linguistic summarization, as the explicit characteristic of information generated with this methodology is crucial to this concept.

A more specific definition is presented by Mani et al. [31], according to whom summarization is the process of distilling the most important information from a source to produce an abridged version for a particular user and task. The current accepted definition of the term most commonly falls under the description of Wilbik and Kaymak, according to whom "linguistic summarization offers novel ways of gaining insight into large amounts of data by extracting their main properties and representing them linguistically" [51]. This depiction pertains to the current industry standard approaches that are deployed in weather forecasting, health-care, CRM systems and more.

The concept of linguistic summarization was introduced by Ronald Yager [56] in 1982, and then developed into an implementable form by Kacprzyk and Yager [21] in 2001. The basic idea of summaries is the interpretation in terms of the proportion of elements containing a certain property [54]. To illustrate, the summarization of data in a software development project could be presented in the form of "most bugs are time-consuming" or "increasing bugs most of the time are of a low variability". The produced output from language summarization techniques is considered as "less terse than the statistically produced data and permits various ways of data summarization [54].

2.5.1 Basic concept

The way humans process their thoughts and communicate is inherently fuzzy [20]. Due to a constantly changing environment, almost no events can be defined in only true or false categorizations. Decisions are made based on a preemptive though process logic whereby a person analyzes the scenario and decides based on the circumstances. The format for such a decision is defined as a generalized modus ponens, and is presented in the following format [60]:

If X then Y.

X.
Therefore, Y.

Due to the strictness of this format, many events in the real world may not be represented. By attributing fuzzy values to X and Y, it is now possible to interpret more scenarios in the real-world. A 'truth degree' between 0 and 1 is indicated, and a value can fall somewhere between (compared to only true or false crisp values).

If X, then Y.
mostly X.
Therefore, mostly Y.

The membership is expressed in object x of set A as:

$$\mathrm{X}_A(x) = \begin{cases} 1 \; if \; x \in A \\ 0 \; if \; x \notin A \end{cases}$$

For each object in universal set, a membership value is shown about the degree membership to which x belongs to A. Then the fuzzy set can be expressed:

$$A = \{(x, \mu_A(x)\ ) : x \in X\}$$

where $\mu_A(x)$ is the degree of membership of element x in A. Multiple approaches to summarization using fuzzy sets exist [36]. This basic concept is then extended into implementable formats. The most common one used frequently for the generation of fuzzy linguistic summaries is motivated on Yager's terminology [56] in the form of Q_y's are/have S where:

V = a quality (attribute) of interest, (e.g. types of issue in database)
$Y = \{y_1, .., y_n\}$ a set of records with quality (attribute) V for object y_1
$D = \{V(y_1), ..., V(y_n)\}$ a set of data ("database")

The *summary* of a dataset consists of:

a summarizer S (e.g. short, long)
a quantity Q (e.g. many, most)
a truth (degree) value T (e.g. 0.2 for low, 0.8 for high)

as, e.g., "T(*most of issues are long*)=0.8". The truth T is represented as a validity of the summary, and to a certain extent as a quality or goodness of the summary. For a given set of data D, we can explore various matching summarizer S and quantity Q, and the assumed measure of truth indicates the truth that Q data set entries satisfy the summarizer S.

Summarizer

The summarizer S is a linguistic expression semantically represented by a fuzzy set. A summarizer such as "short" is tagged as fuzzy set in the universe of discourse (e.g. length of issues in days $\{0, 1, 2, ..., 3000\}$), containing all the possible values for issue lengths in our data-set. To present an abstract view, let us assume cases can only be "short" or "long". With this assumption we can define the membership function as:

- issues that last up to 25 days are definitely "short", (membership = 1).
- issues that last more than 35 days are definitely "not short" (membership = 0).
- issues that fall between 25 to 35 days will have a membership value between 1 and 0. The shorter the issue, the higher its corresponding degree of membership.

Similarly, we can add the fuzzy set "average", "long" and "very long" to further define our membership function. It follows that a specific case of length (e.g. 40 days) can be a member of all fuzzy sets, but the functions $\mu_{short}(40)$, $\mu_{average}(40)$, $\mu_{long}(40)$ and $\mu_{verylong}(40)$ have very different membership values for same issue length.

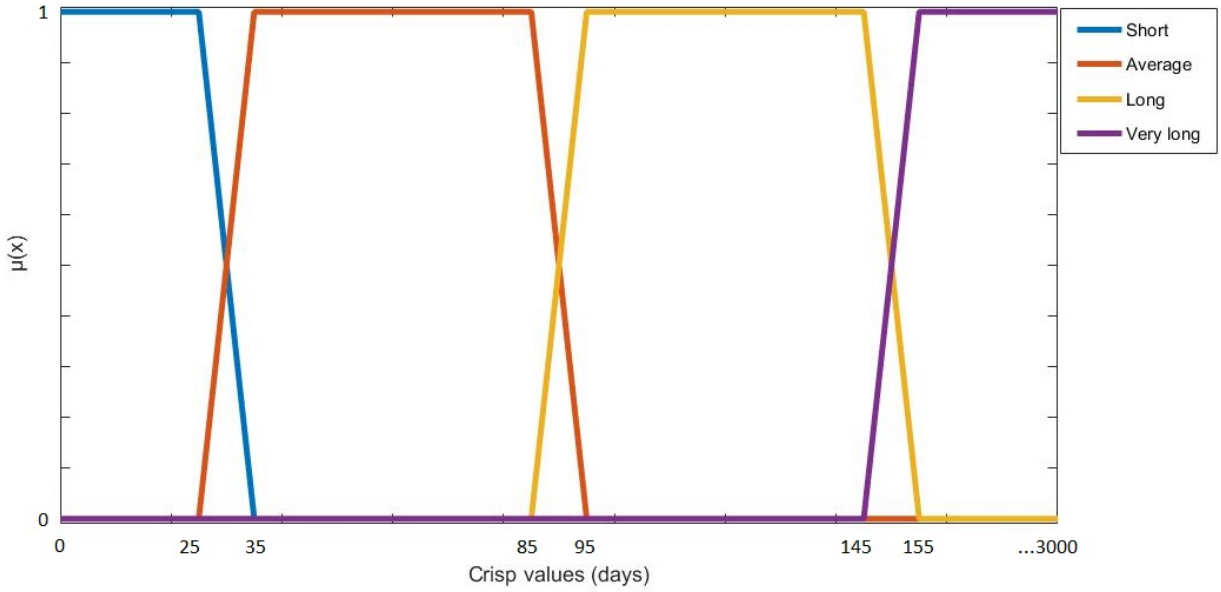


Figure 2.2: Trapezoid membership function for case length

Quantifier

There exist two types of quantifiers - the absolute quantifiers (e.g. nearly 20 issues), or relative quantifiers (e.g. many, most, almost all). The fuzzy property allows them to be defined as possibility distributions over non-negative integers and the real interval [0,1]. As the fuzzy summarizers and quantity in agreement are subjective, they can be predefined (as in Dijkman and Wilbik [15]) or defined (as we did for our main dataset).

Protoforms

Zadeh [61] defines a protoform as an abstract template for a linguistic summary. We can implement custom linguistic summaries by changing the format of the protoform. The basic summaries in section 2.5.1 can be represented as:

$$Q_y\text{'s are } S$$

e.g., "most (Q) of issues ($y's$) are short (S)", and the extended form:

$$QR_y\text{'s are } S$$

e.g., "most (Q) of Change Request (R) issues ($y's$) are short (S)", where Q is the quantifier, S is the summarizer, y's is the quality of a set of objects, and R is the qualifier.

Evaluation of linguistic summaries

The quality of summaries can be evaluated in multiple ways. Much research has been done to improve the estimation of linguistic summary validity, most notably by Wu et al. [54] and Delgado et al. [14]. The more common rules present in the field are: Degree of truth, Degree of sufficient coverage, Degree of usefulness, Degree of outlier and Degree of simplicity. For our purposes, the degree of truth or validity T serves the purposes of determining which summary is useful or not (from [0,1]).

For a given data-set D, the degree of truth T is a measure of how the quantity in agreement Q satisfies the summarizer S. A low degree of truth T *(most of issues are long) (0.2)* - stands for

a summarization that has very little support from the data. On the other hand, a high degree of truth T *(most of issues are average) (0.8)* - stands for a summarization that is highly true, where a large number of records support the conclusion.

Summarizer S and qualifier R are fuzzy sets in Y, and a linguistic quantifier Q is assumed to be a fuzzy set in [0,1] as:

$$\mu_Q(x) = \begin{cases} 1 & for \quad x \geq 0.8 \\ 2x - 0.6 & for \quad 0.3 < x < 0.8 \\ 0 & for \quad x \leq 0.3 \end{cases}$$

Based on this, Zadeh's [61] calculus of linguistically quantified propositions yields the following for Q_y's are S:

$$T = \mu_Q \left[\frac{1}{n} \sum_{i=1}^{n} \mu_S(y_i) \right]$$

and for QR_y's are S:

$$T = \mu_Q \left[\sum_{i=1}^{n} (\mu_R(y_i) \wedge \mu_S(y_i)) \Big/ \sum_{i=1}^{n} (\mu_R(y_i)) \right]$$

where μ_R is membership function representing the summarizer and $\mu_Q$representing the quantifier. The minimum operation is represented by $\wedge$ which can be generalized as a t-norm. Equations are for the basic and extended protoform, respectively.

2.5.2 Applications of linguistic summarization

The concept of linguistic summarization has been applied in different industries, for various types of data formats. There exists designated literature in Linguistic Summarization for summarizing databases [22], analysis of time-series data [8], and recently for process log data [50]. Investigating these dimensions is crucial to define the aptitudes and limitations of linguistic summarization. Additionally, there are summarization techniques and knowledge regarding video data, sensor data and even web logs [51].

Linguistic summarization in databases

Analyzing large data sets (databases) using linguistic summarization techniques can provide meaningful insights to organizations. Although the technique is not automated to a full degree, a semi-automated procedure where the user solely provides input is available using Kacprzyk and Zadroznys fuzzy querying add-on for Microsoft Access [22]. The technique can be expressed using mainly two types of linguistic quantity absolute e.g., about 3, more or less 50 and relative e.g., a lot, most. These types of expressions derive from the so-called fuzzy linguistic quantifiers discussed previously and elaborated by Zadeh [60].

In essence, there are two extensions of Yagers approach [56] to linguistic summarization of a set of data. The first is through the utilization of additional degrees of validity and their weighted average which is used to validate the quality, and the second through the embedded summarization procedure in a flexible fuzzy querying environment using Kacprzyk and Zadroznys (1994 1997) FQUERY approach for Access[22].

Linguistic summarization in event logs

With todays workflow management systems, software development organizations store much of their data in the form of event logs which resource accessed a file, who edited it, at what time, what action which are then logged on the database. Event logs show occurrence of events over time and each event refers to a process and case [45]. By this definition, linguistic summarization

must also provide insights into the timeliness and completeness of various events that occur in a given setting.

Analysis of event log data using linguistic summarization requires pre-processing and data understanding [50]. According to Wilbik and Kaymak [50], when we seek to calculate performance indicators such as lead time, we primarily need to combine the cases and calculate required values. However, this approach is not always applicable as data is not formatted properly. Based on the industry and scenario, event logs can vary in terms of storage, making event log analysis using linguistic summarization all the more challenging. In brief, producing meaningful linguistic summarization requires a certain process mining approach.

Traditionally, there are three perspectives in process mining [45]. The first perspective is that of processes, which focuses on control flow. The second is case perspective, where the focus is on case characteristics. Lastly, the organizational perspective focuses on the human resources (or other types) which are at some level involved in the process, and how. Using linguistic methodology, Wilbik and Kaymak [50] claim that process log summarization can provide insights by combining all the process mining perspectives . Various underlying processes of a setting can be improved, and detection of problems and bottlenecks could be feasible. Such improvements in an already existing system would be largely beneficial to any organization and industry.

A practical implementation of linguistic summarization in event logs has been recently (2017) investigated by Dijkman and Wilbik [15]. The approach for generation of insights is provided, alongside with results from a case-study in the Netherlands. This research was a key starting point for our study, motivation and evaluation of generated findings.

2.6 Software development metrics

Extensive research on software performance engineering exists, where the focus is mostly on the performance of the software (code) and hardware. Limited literature exists for the analysis of the software development issue resolution processes. Software performance as a field of study is generally concerned with capacity and timeliness [53]. Based on this approach, there exists a highly disciplined approach known as Software Performance Engineering (SPE) which aids in evaluating a systems performance and ways for improvement. The approaches of Software Process Engineering are described using early modeling [42]. Barber describes the process using specific measurements [5]. Similar approaches are further elaborated in [44] and [49].

Software Performance Engineering represents the software development procedures and analysis which is used through the software development cycle. Using the Software Performance Engineering definitions, our case-study falls into this category, with a different focus that of processes. By aligning with the metrics which are crucial in a software development context, we are also successfully providing relevant means of analytical approaches to draw insightful information from the software development context.

As software performance is a difficult quality to comprehend, it is affected by the code design, environment and resources. In many software development environments software problems cause delays, costly projects and failed deployment. It is therefore important to address why such performance problems arise, and tackle them.

Software Performance Engineering defines a series of activities (illustrated in Figure 2.3) significant in determining a systems performance. The first activity is that of identification of concerns (the important operations and resources). Afterwards the requirements such as operational profile, workload and throughput are necessary to describe and analyze behavior. Unified Modeling Language is commonly used to describe these processes. The next step is to predict scenarios by modeling the behavior of the resources using various industry-standards such as Business Process Management Notation (BPMN). The next step is to conduct performance testing on multiple segments of the system and understand the produced results. Then we need to conduct maintenance and evolution, by predicting the changes and necessary activities. It is possible to add features and determine what the impact on the process will be. As a final step, a total system analysis is necessary to extend the improvement activities of the process into the future [53].

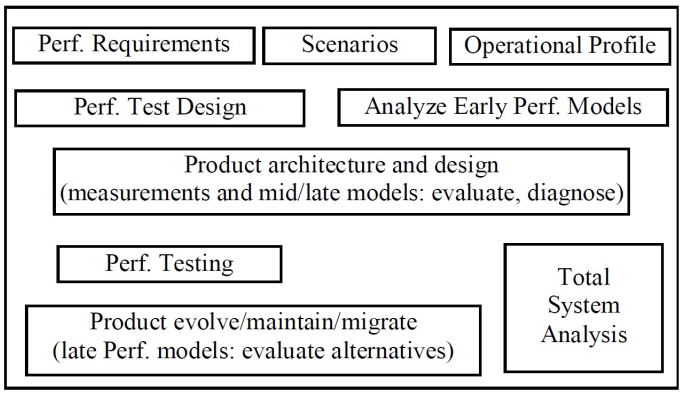


Figure 2.3: Steps of performance engineering
[42]

2.6.1 Why we measure time and process workflow?

Understanding the reasons behind performance bottlenecks in software development can be extracted from data most commonly through data visualization of patterns and relationships. Woodside et al.[53] argues that future areas must provide the means for better visualizations deep catalogs of performance patterns and algorithms for automated search and diagnosis. Challenges of future approaches is to visualize the causal interaction among behavior and resources and not only analyze one aspect of the system.

By combining metrics such as time (lead-time), diversity of issue types, it is possible to analyze and present visually (or in other forms) significant insights into the process of development. Furthermore, by modeling the information in terms of a process workflow, it is possible to analyze from a process compliance point of view, and determine bottlenecks and areas of improvement from a visual perspective. Franks et al. discusses a search approach that can detect resource pools and lead-time related bottlenecks. Then depending on the findings the process can be improved based on the designer expertise using the principles described in [55]. This research is instrumental in motivating our two key metric measurements time and process compliance.

2.7 Conclusion

Data has become more sizable and complex than ever before and thus the process of deriving insights has become more challenging. Descriptive statistics and traditional (manual) methods of textual interpretation have put barriers into the processing speed of analysis, and their accuracy. As data mining and process mining techniques continue to provide detailed information onto data and processes, an alternative approach of linguistic summarization has become available to combine insights and present it in simple, human language format.

Significant linguistic summarization research is being conducted recently, with Kacprzyk and Zadrozny [20] focused on generic data sets, on time-series by Castilla-Ortego et al. [8] and on event-log data by Wilbik and Kaymak [51]. A very recent practical implementation by Dijkman and Wilbik (2017) [15] served as a guide towards the implementation of the practices and motivation behind our study.

Nonetheless, linguistic summarization remains in isolation when it comes to software development and its potential in this domain. This review is partially motivated by the lack of this methodologys use in an industry where summarization of information is crucial to project leaders and developers alike, as well as the motivation to evaluate such summaries regarding their validity.

We believe that there is potential in studying the concept, algorithms and methodologies of linguistic summarization to generate insight into software defect management. By studying the recent research and applying it practically, it is possible to determine which methodology

suits software development environments best. This is done as an attempt to bridge the current knowledge gap for what analytical approaches are best in the field of software development.

With rapidly increasing software development projects, team-members, complex data structures, it is of particular significance to study and develop a methodology for performance measures in software development - especially with a practical experimentation. In an environment where the necessity for real-time information is crucial, it is of great business value to analyze and determine what methods are best for generating insights in the software development industry.

Chapter 3

Methodology

According to Peffers [37], design science creates and evaluates IT artefacts intended to solve identified organizational problems. Although our study is highly driven by data analysis, our research objective is to generate artefacts, which can provide knowledge into software defect management processes [47]. While these processes are crucial for the improvement of software performance, the proposed design science methodology provides the guidelines to resolve observed problems, to make research contributions, to evaluate the design and to communicate the results to appropriate audiences [16].

The most important element that design research produces is an artefact created to address a problem . The artefacts may include constructs, models, methods, and instantiations, which in our case translate into the analytical environments of process mining and linguistic summarization [16]. In our research, we collect, analyze and implement activities necessary to generate insights and validate their usefulness.

We choose design science research methodology (DSRM), in order to document our approach in a step-by-step mental model. A mental model is a small-scale model of reality that can be constructed from perception, imagination or the comprehension of discourse [37]. This model translates into our methodological approach, as we motivate our research based on the guidelines of DSRM principles. The desired outcome of DSRM outweighs traditional theory testing or interpretative research, since it provides a process model that can be used to resolve tangible organizational issues.

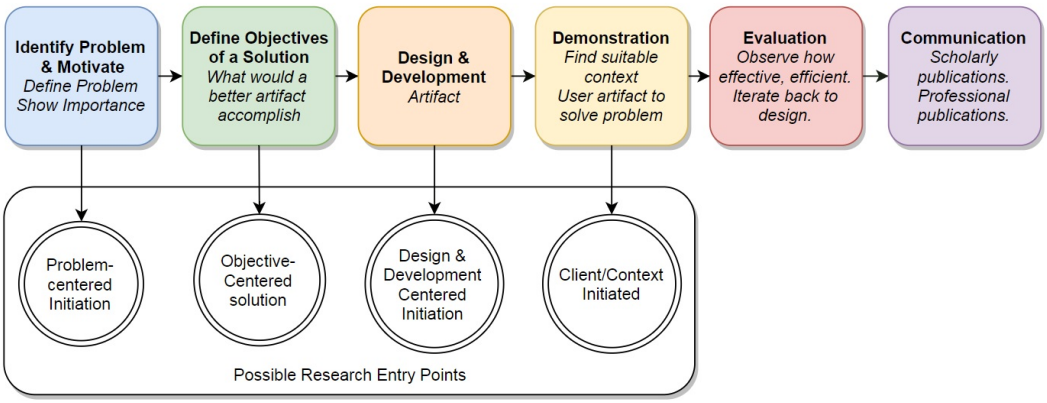


Figure 3.1: DSRM Process Model
[37]

3.1 Research Methodology

The detailed mental model with complementary activities is mapped in Figure 3.2. Using the DSRM principles, we have constructed our methodological approach. The complete list of matching activities is grouped accordingly, where our objective definition falls into the problem and motivation identification. Then we define our research question, which maps the objective definitions for a solution in the DSRM model. The next six activities are mapped to the design and development of our artifacts (data visualizations, process models, simulations, linguistic summary generation environment), where we select our analytical methods, determine the relevant case study, explore the data sources, collect, pre-process and analyze our data. We then validate our results in continous basis through demonstration and meetings. Afterwards, we evaluate the usefulness of our artifacts through direct interviews with the stakeholders. Lastly, we present our findings in written as a professional publication.

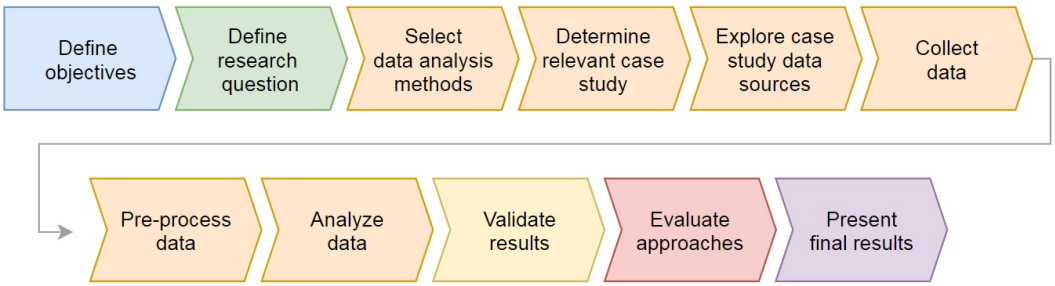


Figure 3.2: Methodology mental map

3.1.1 Define objectives

We begin our research by defining the overall objectives of the study. Our interest in determining suitable methods for generating insights in software development raises the need for a thorough analysis. We elaborate on the steps required to reach our objective by defining the research questions. To clarify the goal and necessary procedures, we organized multiple meetings with academic and business stakeholders. This was done to ensure that the objectives are in compliance with all relevant parties.

3.1.2 Define research questions

We arranged joint meetings with the relevant stakeholders during the early stages of the study. Based on these discussions, we identified the main research question that would have scientific and practical value to the organization, as well as to the field of software development.

We then break down the main research question into five sub-questions that can be translated into concrete study objectives. First we define what an insight means in the context of software development processes. After this is established, we determine what are the state-of-the-art methods that we can utilize to generate such insights. This was accomplished through extensive literature review and in consultation with field experts. Afterwards, we implement our analytical methods and generate final results. These results are reviewed continuously through discussions and presentation, in order to ensure their validity. Then, we conduct interviews to evaluate the usefulness of each method to the case study. Finally, we compare the approaches and provide recommendations.

3.1.3 Select data analysis methods

This study explores and evaluates three methods for generating insights into software development defect-management processes. These methods are selected after a thorough literature review and

academic supervision. The methods we implement in this study are:

- Traditional data analysis
- Process mining
- Linguistic summarization

Due to the common application of traditional data analysis in various industries, we begin our study using this method. We acknowledge that this approach can provide useful information in tabular forms and visualizations (charts, graph). This method can be extended by drill-down options for deeper analysis, making it suitable for different commercial users.

However, there are certain limitations to this method, e.g. visualization of workflow processes, or its crisp representation of variable relationships. To explore beyond these limitations, we first analyze the data using process mining practices.

With process mining we are able to extract useful information from data in the form of event logs. We use it to check the conformance of processes, detection of bottlenecks and predict future execution problems [1]. In software development, it can provide direct insights into the performance of issue resolution, time, compliance, and predict future system behavior.

Though visualizations are often used to increase understanding of data by emphasis or discovery of visual cues, there are other ways to generate insights into data. One of these methods is linguistic summarization, where the most characteristic aspects of data are presented linguistically in natural language [23]. Linguistic summarization techniques make it easy to gain insight into large amounts of data by describing its main properties linguistically [52]. Using this technique, we attempt to provide quick overviews on the main data characteristics, which can be extremely useful to high level management.

3.1.4 Determine relevant case study

In this step we analyze whether the case study environment provides the information necessary for the analysis we want to conduct. This is done in direct communication with company experts to ensure the data and human resources required to perform primary and secondary analysis are available. In compliance with the academic goals of this study, we have determined that our case-study environment provides a suitable environment.

Current process description

The AS-IS resolution process contains any of the following activities: *Initiate Register, Investigate, Assign, Fix, Verify, Close, and Reject.*

When an issue is reported by the client or from within, a resource submits this issue along with the description of the problem, and other characteristics to the system. The issue is then registered by a regulation board for investigation, upon which the problem is investigated and then returns to the board. This task is then assigned to a resource (could be the same person) who will carry the fixing step. After its completion, another resource verifies if the solution is valid. Lastly the issue can be closed, or it can be rejected based on a previous step — in the investigation phase — or immediately after initiation by the board. The original (intended) process workflow map is presented in Figure 4.18.

Organizational stage of data utilization

Before we initiate with the investigation of software performance metrics, it is crucial to determine the stage of data utilization within the organization. Since our entire analysis revolves around data, it is important to know at which stage of data utilization the organization stands.

The emerging stages of analytical adoption, as described by LaValle et al. are crucial in identification of which approach is the most valid [28]. There are three levels of analytic adoption aspirational, experienced and transformed [28], and detailed description is illustrated in Figure

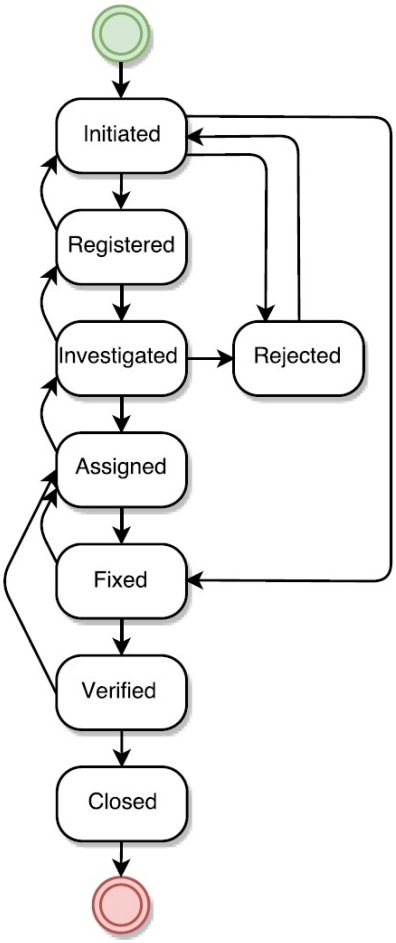


Figure 3.3: Original workflow map

3.4. The first stage can be found in organizations that are largely focusing on automation of existing processes and cutting spending. The experienced level is found within organizations that already have some analytic experience and practice in place, and frequently act on new initiatives to optimize their operational capability. Lastly, the transformed level of analytics is found among organizations that use data in multiple sectors. By using analytics they are able to create the competitive advantage and constantly analyze their way of working for the purpose of improvement. These organizations pay less attention to the implication of cost, as they have already automated many of their operations earlier in the process.

Our case-study is a mixture of a highly experienced and transformed organization, where data is a key driver in decision-making, although due to large amounts of data, many aspects of it are not analyzed.

3.1.5 Explore case study data sources

Gaining deep understanding of the case-study environment is crucial to defining the relevant data sources needed for the analysis. Practical experience was a pivotal point in gaining hands-on knowledge on how software development activities are distributed. This enabled us to identify process issues that we could address with our research.

In collaboration with company experts, we identified multiple data sources within software development, out of which we selected the data set of the defect management platform. This decision

	ASPIRATIONAL	EXPERIENCED	TRANSFORMED
Motive	•Use analytics to justify actions	•Use analytics to guide actions	•Use analytics to prescribe actions
Functional proficiency	•Financial management and budgeting •Operations and production •Sales and marketing	•All Aspirational functions •Strategy/business development •Customer service •Product research/development	•All Aspirational and Experienced functions •Risk management •Customer experience •Work force planning/allocation •General management •Brand and market management
Business challenges	•Competitive differentiation through innovation •Cost efficiency (primary) •Revenue growth (secondary)	•Competitive differentiation through innovation •Revenue growth (primary) •Cost efficiency (secondary)	•Competitive differentiation through innovation •Revenue growth (primary) •Profitability acquiring/retaining customers (targeted focus)
Key obstacles	•Lack of understanding how to leverage analytics for business value •Executive sponsorship •Culture does not encourage sharing information	•Lack of understanding how to leverage analytics for business value •Skills within line of business •Ownership of data is unclear or governance is ineffective	•Lack of understanding how to leverage analytics for business value •Management bandwidth due to competing priorities •Accessibility of the data
Data management	•Limited ability to capture, aggregate, analyze or share information and insights	•Moderate ability to capture, aggregate and analyze data •Limited ability to share information and insights	•Strong ability to capture, aggregate and analyze data •Effective at sharing information and insights
Analytics in action	•Rarely use rigorous approaches to make decisions •Limited use of insights to guide future strategies or day-to-day operations	•Some use of rigorous approaches to make decisions •Growing use of insights to guide future strategies, but still limited use of insights to guide day-to-day operations	•Most use rigorous approaches to make decisions •Almost all use insights to guide future strategies, and most use insights to guide day-to-day operations

Figure 3.4: Stages of data utilization among organizations [28]

was based on the quality of data contained and the overall significance of defect management to software performance.

Relevant stakeholders were also identified in order to collect additional information regarding the process and validate our findings. These experts were provided continuous input and support for the research purposes. Table 3.1 shows the data sources for the collected data during the study.

Table 3.1: Data sources

Data types	Sources
Issue data	Workflow management system
Code-size metrics	Information system
SW Development	Primary research
SW Life cycle	Primary and secondary research

3.1.6 Collect data

Primary and secondary data was collected to address the research question. The data derived provided key insights into the software defect management process. Using simple SQL queries we were able to extract basic information that is recorded by the workflow management system. We were able to investigate various characteristics, focusing on issue type, project type, lead time, completion rate, resolution type, process compliance, influx and outflux rates and others for the purposes of the study. Thorough analysis was performed to ensure we work with data that is relevant to the analysis methodologies outlined in this study. Therefore, we selected the

following attributes to be able to conduct comparative analysis throughout all analytical methods: state, project type, severity, issue type, submit and close date. Furthermore, we collect the defect management event log to analyze the behavior of the system from a workflow point of view.

Due to relevance of study at in relation to current processes, the analysis was focused only in the last 5 years. The extracted data contains 3,981 completed issues.

Dataset of defect-management system

The data-set being analyzed contains information on the process of issue resolution system, whereby information on issue type, project, priority, severity, description, and other variables are stored. Each issue is stored with a specific issue ID, and a complementary time stamp is recorded for each activity. A state of an issue shows at which stage the current issue exists in the current moment. The project type refers to project buckets among issues. Severity is provided as a means to issue resolution team to determine what the impact of the case has on the oveall system. In this way, the issues can be prioritized and resolved accordingly. The issue type refers to the different defects, which can be a problem report for a non-functional aspect of the software, or a change request, which is a request to modify or change a particular characteristic of the software in order to improve it. Lastly, the submit and close date are given for each record.

Table 3.2: Dataset of issues

Issue Id	State	Project type	Severity	Issue Type	Submit Date	Close Date
321913	Rejected	Field	Crash	Problem Report	1/5/2012	1/10/2012
321667	Rejected	MS windows	Major	Problem Report	1/4/2012	3/1/2017
322439	Closed	Field	Minor	Problem Report	1/10/2012	1/17/2012

Our secondary data resource — the event log — contains records of the workflow process for handling issues allows tracking the sequential activities that are undertaken by staff. It records the all state changes over the resolution process, by specifying the date of such a change with a timestamp and respective state change.

Table 3.3: Event log of issues

Issue Id	State	Timestamp
321913	Initiated	1/5/2012 16:44
321913	Registered	1/5/2012 16:44
321913	Investigated	1/9/2012 15:00
321913	Rejected	1/9/2012 17:00

Our second data collection source was the software development stakeholders. The experienced professionals played a crucial role during the gathering of additional insights as well as validating the relevance of our results. This information was collected in an informal manner during the exploration period, and then followed by formal interviews to collect their insights on final results.

3.1.7 Pre-process data

Pre-processing was done to ensure only values that contained insightful data were contained in the final data-set. More than 470 different variables were available for extraction. Due to the system being used for a variety of departments, many of the variables were not used at all, or to an extent that provides insightful information.

As the data was recorded using an efficient information system, there were very limited cases when data cleaning was necessary. This was only the case when generating the process maps from event logs, as there were erroneous activities that were manually recorded. Some issues

have unusual traces and produce noise in our process map. It was necessary to remove a small number of entries due to inconsistencies with regard to intended process workflow (incorrect traces, duplicates). The impact of this removal was minimal (6 percent).

Furthermore, additional filters were added to remove all cases which have taken more than 2 years to complete, as these issues are regarded as insignificant to the process owners and produced complicated process maps. The impact of this step was a removal of less than 3 percent of cases from the original dataset. Lastly, issues with Propose Reject were treated as Rejected due to the fact that they are always followed by this state. Time difference was taken into account for this change. The final dataset for the event log contains 3,485 issues which make up for 91 percent of the original entries.

3.1.8 Analyze data

Our investigation revolves around the analytical steps that are necessary to generate insights. This process is somewhat a direct answer to our research question, as we analyze the dataset and provide results. We also defined our crisp representation of issue lengths throughout all approaches earlier in the process. This definition is presented in section 3.1.8.

Traditional data analysis

We begin with the traditional data analysis, where we generate the basic descriptive statistics from the dataset, followed by visualizations of the findings. Then we describe the figures using tables, and drill-down on the distribution by length of issues, type of issues, whether issues are rejected or closed, etc. Furthermore, we ensure that we use the correct figure for average summarization, by checking the total distribution of issues based on their length. Based on the type of distribution (i.e. skewed vs. smooth), we determine whether the average or median is a good representation of our data. Using the dataset of issue resolution, we can generate a variety of other ad-hoc requests if necessary.

Using Pride and Ferrell's [39] exploratory and conclusive research design, it is possible to approach the collection and analysis stage of traditional data analysis in a meaningful and productive manner.

We also analyze the event log in order to view any frequent or unexpected behavior on the process traces. We need to pre-process the data presented in an unfavorable format (with each event having a repeat ID and timestamp) using Python environment further described in Figure 3.5. We aggregate the activities in sequential order, as determined by the timestamp for each issue. This step helps us modify the data in a meaningful format that we can then analyze. It is also useful for validation of findings drawn from Process Mining and Linguistic Summarization.

Process Mining

After we have conducted our basic analysis, we shift our attention to the area of process mining. Due to limitations of what we can achieve with trace information, we import the event log in the Fluxicon Disco environment to analyze it in detail. The parameters used for this process are the in-built Fluxicon discovery algorithm, mainly an updated version of Christian W. Gnthers Fuzzy Miner process discovery algorithm [41]. The output is a Fuzzy Model, ideal for use in unstructured log data. Using this discovery algorithm, we simplify the process model at our desired level of abstraction, thus leaving out some less important activities (hide in clusters) compared to Heuristic miner in ProM (which produced spaghetti-like process maps). One disadvantage of the fuzzy model is that it may not be directly converted to other process modeling languages.

To further determine the exact number of working hours spent on issues, we make use of the additional Timewarp function available in Fluxicon Disco. This ensures that the time-stamps are used as point of reference to calculate the actual working hours, acknowledging weekends and holidays. This helps to visualize the map with business hour metrics. The settings for the working hours were set from 08:00 until 17:00. Such a figure is crucial in determining the exact amount

Table 3.4: Components of exploratory and conclusive research design

Components	Exporatory research	Conclusive research
Research purpose	Generate insights about software development ticket-handling	Verify insights via linguistic summarization
Data needs	Challenge to retrieve only necessary information	Determine course of action with primary and secondary research
Data sources	Difficult to retrieve due to confidentiality and multiple-sources	Narrow scope supports conclusive remarks
Data collection form	Highly textual data, unstructured, followed with primary research (interviews)	Filtered, structured to good extent
Sample	Commenced with small sample analysis, followed by entire data-set since 2002.	Only entries since 2012, narrow study period to last 5 years
Data collection	Multiple data sources available, workflow management systems	More organized, though still multiple sources necessary
Data analysis	General statistics, quantitative and theoritical	Qualitative and quantitative insights
Reccomendations	Suggestive of trends based on data/text/process mining	Conclusive and exact

```
import sys

if (len(sys.argv) != 2) :
        print('Usage: {0} filename.csv'.format(sys.argv[0]))
        sys.exit(-1)

f = open(sys.argv[1])
lines = f.readlines()
f.close()

last_id = None
for l in lines :
    tok = l.strip().split(',')
    tmp_id = int(tok[0])
    tmp_event = tok[1]

    if tmp_id != last_id :
        last_id = tmp_id
        print("\n{0}, ".format(last_id)),
```

Figure 3.5: Algorithm for aggregation of issue states by ID

of time spent on issue, and translate these findings with the cost of resources responsible for the activities. Due to lack of this information, we have not extended our analysis but merely find it significant to mention the methodology.

For our simulation purposes, we recreate the process model using the derived insights from the process map in the Bizagi environment. This is useful to derive and simulate insights into
920 the AS-IS and TO-BE defect management process. We are able to reproduce the results of our dataset, and conclude that the environment is able to simulate the next coming years based on a changing arrival rate of issues (or other configurations). We first experiment with the environment variables so that our simulated environment is able to reproduce the same (or similar) results as

our real data. This is done through a series of testing with our data for particularly year 2016. The actual activities are abstracted in the next paragraph, whereas the final results are presented in Chapter 4.

We first we introduce the average metrics of the entire last 5 years. Then we introduce the metrics of year 2016 using average as input. Since our dataset is not well represented by the average, we use the median for the next simulation. Getting closer to the desired results, we finally determine the arrival rates of the issues as they occurred using normal distribution. This provides us with an ideal environment where we can conduct further analysis and simulation of year 2017 with potential changes to the process.

Linguistic summarization

Lastly, we also implement the approach of Linguistic summarization. Linguistic summarization is a contemporary approach in data mining, which can be of great use in modern organizations. It is used for extracting and summarizing information in an understandable, natural language (sentences) format. Moreover, this method is suitable for environments where there is high variation and constant information is necessary to improve processes. Software development environments can greatly benefit from this methodology.

What makes linguistic summarization even more powerful is that in our case-study, we can use this approach to extract information from the data-set, and also the event log. We apply the algorithm and develop our implementable format in MATLAB, where we generate basic and medium-complexity sentences for the first data-set, and basic sequence summaries from the event log data-set using a recent solution implemented by Dijkman and Wilbik [15]. Due to unreliability of generated results and advice from the original author, we only present the event-log findings for validation purposes. As our main emphasis revolves around the metrics of time and compliance, this approach provides sufficient knowledge contribution (insights).

For the generation of linguistic summaries of our data-set, the membership definition is expanded into categories as deemed feasible for the data. We use the trapezoid membership function to define our parameters. Following the investigation on membership functions by Bouchon-Meunier (1996) [7], we studied the use of other approaches such as piece-wise linear functions, triangular (three parameters) or trapezoidal (four parameters) and smooth-curved functions such as Gaussian function. We choose the trapezoid membership function for reasons of simplicity into defining our parameters.

In our case, we classify our cases based on length in four separate categories. These qualifications were conducted and agreed upon with the organization metric standards. The membership function for the dataset is detailed in Figure 2.2. Cases can be short, average, long and very long - and in a crisp binary 'definition', they are defined as:

Short - Issues that take up to 30 days
Average - Issues that between 30 to 90 days
Long - Issues that between 90 to 120 days
Very long - Issues that take more than 120 days

In terms of fuzzy, we define the case-length values by the lower limit a and upper limit d, and the lower and upper limits of the nucleus b and c respectively. As agreed with stakeholders of the organization, the defined values are illustrated in 3.6 (shown only until 300 for clarity) using the following values: (0 0 25 35) (25 35 85 95) (145 155 2190 2190).

The membership for our quantifier is defined using trapezoid membership for values $veryfew$, few, $many$, $most$, respectively (0 0.1 1 1) (0.3 0.5 1 1) (0.5 0.7 1 1) (0.7 0.9 1 1). These values are set in order to spot isolated behavior ($veryfew, few$) and recurrent behavior (many, most) and are represented in Figure 3.7.

For the generation of linguistic summaries from the event-log, we use the definitions from research case-study by Dijkman and Wilbik (*many, most, almost all*) [15]. The membership function for length of issues is illustrated in Figure 3.8.

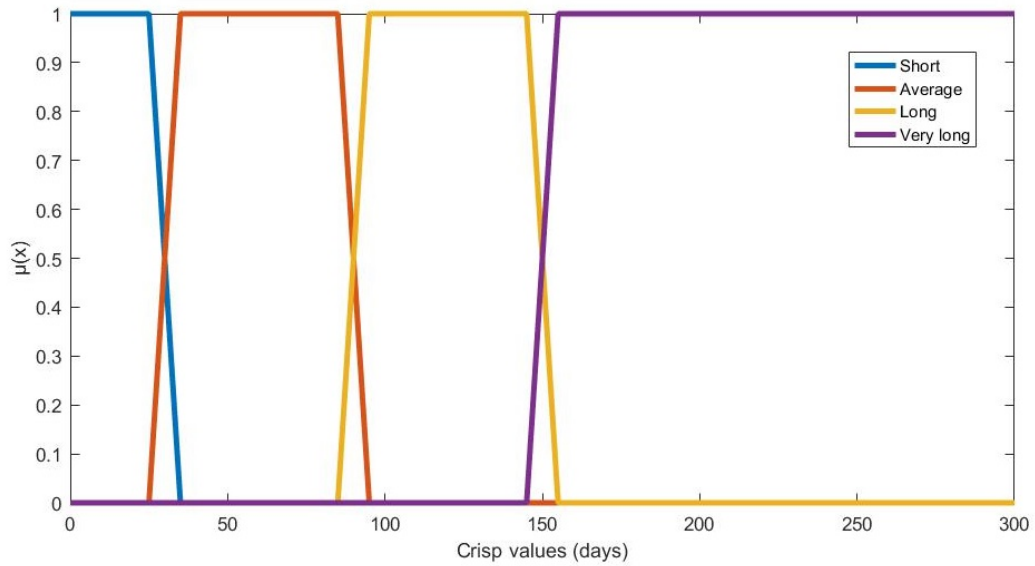


Figure 3.6: Membership function for issue length groups.

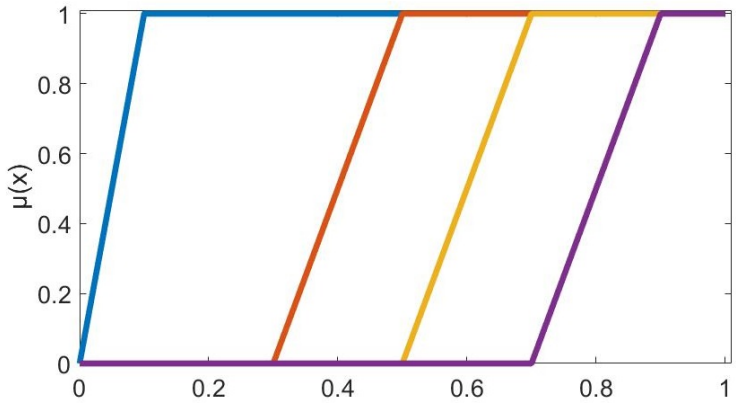


Figure 3.7: Fuzzy quantifiers of dataset.

3.1.9 Validate results

Results generated from descriptive statistics and process mining are validated continuously throughout the study. We present the data summaries and process maps after which we receive feedback on the accuracy of the findings from the company stakeholders. These experts help us in understanding particular behavior appearing in our analytical findings, as well as clarifying any inconsistencies.

As for validation of linguistic summaries, we discussed the results with key process owners, who were aware of the system behavior and could provide credible advice on our findings. We presented the results in their original context, while explaining the concept of linguistic summarization. When presenting the numerous summaries without any additional filters, the stakeholders had difficulty going through the list to find interesting summaries. That is why we deemed it necessary to evaluate these summaries through interviews.

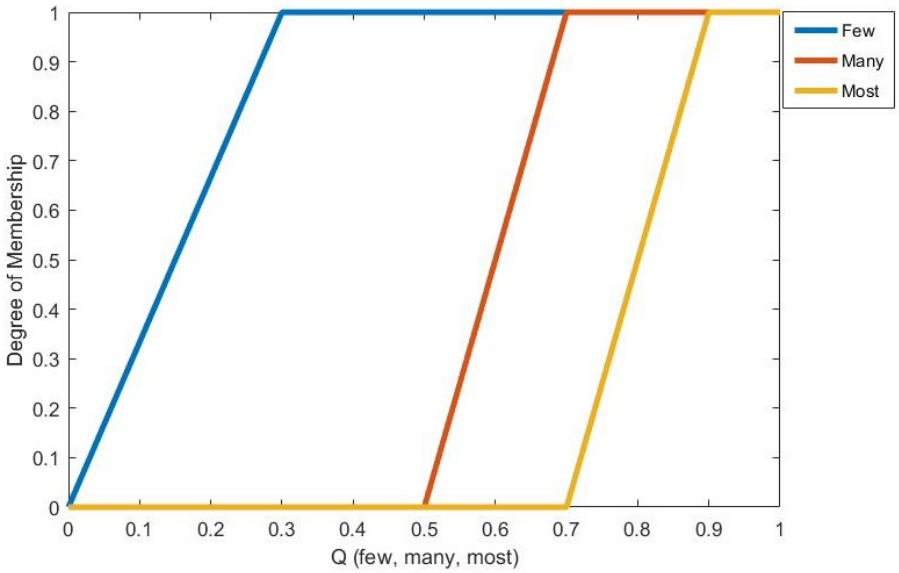


Figure 3.8: Fuzzy quantifiers of event log.

3.1.10 Evaluate approaches

We evaluate the preferred methods according to the opinions of the stakeholders in the organization. This is done both informally during the study duration — where we check what sequences, data summaries are possible — and formally later during the interview stage.

Besides evaluation of the preference of stakeholders regarding the summarization of data, we focus extensively on evaluating the extent to which the degree of truth summarizes the dataset. This evaluation is based on the perceived sense of company experts, who are informed about the process.

For the purpose of linguistic summary evaluation, we ask stakeholders at the organization in identifying sentences which are true (valid) and untrue (invalid) without showing them the degree of truth. We evaluate a series of summaries, which are categorized into four different groups with different truth values ranging from very low values - 0 to 0.25; low values 0.25 to 0.5; medium values 0.5 to 0.75; and very high values 0.75 to 1. Groups contain four randomly selected summary sentences and are presented in random fashion during the interviews. The gathered results are then summarized by the average of each group to determine whether there is a direct correlation between validity of sentences based on the value of truth, and the perceived validity based on company experts. Many stakeholders expressed interest in viewing the entire list of linguistic summaries at the end of the interview sessions.

The sentences used for this experiment are recalled in Tables 3.5, 3.6, 3.7 and 3.8. The order of sentences was distributed randomly to ensure the question was not a lead-type.

Table 3.5: Summaries with very low validity (0 to 0.25)

	Summary	**T**
ST281	Many rejected issues are short.	0.12
CM13	Most Change Requests which are Closed and of Critical severity are short.	0
CM765	Most Problem Reports which are Rejected and of Major severity are short.	0.11
IT45	Most Problem Reports are short.	0

Table 3.6: Summaries with low validity (0.25 to 0.5)

	Summary	**T**
SV309	Few major severity issues are short.	0.36
B16	Most issues without project type information are very long.	0.35
CM140	Many Change Requests in maintenance which are Closed and Minor are very long.	0.5
CM980	Very few Problem Reports in premium which are Closed and are Critical are very long.	0.41

Table 3.7: Summaries with average validity (0.5 to 0.75)

	Summary	**T**
TM57	Many issues in field are short.	0.58
CM176	Most Change Requests in maintenance which are Rejected and are Minor are very long.	0.7
CM1696	Most issues in interface which are rejected and of Minor severity are very long.	0.7
TM200	Few issues in transactions are very long.	0.57

Interviews

In order to collect additional information into the process of ticket resolution and quality of generated linguistic summaries, we conducted a series of interviews with relevant stakeholders. The main target groups were the managerial level and software developers, testers, architects. The interview process was necessary to understand the validity of the generated insights using linguistic summarization in particular, as well as investigating and confirming findings with other approaches.

A brief pilot-project was conducted with 1 software engineer and 1 research and development manager to ensure that the questionnaire was applicable and desired information retrievable. A total of 6 managerial level staff were then interviewed, followed by 5 developers and testers. The measured objectives were both qualitative and quantitative which further aided in the validation of findings and generation of new instances of relevant information.

The interviews were conducted face-to-face with the targeted groups and each interview lasted approximately 20 minutes. The main objectives of the interview process were:

- Usage and understanding of the workflow management system (WMS) (as perceived by managers/dev/testers)
- Development cycle of issue resolution (as perceived by dev/testers)
- Preferences for data summarization
- Evaluation of linguistic summaries

The **Usage and understanding of WMS** experiment was aimed at revealing the level of utilization of the defect-management system. This information is gathered to check whether stakeholders with more knowledge about the system have different opinions or evaluations regarding

Table 3.8: Summaries with high validity (0.75 to 1)

	Summary	**T**
B1	Very few issues are short.	1
B12	Many issues with no information on type are very long.	1
CM61	Most Change Requests which are Rejected and of Critical severity are short.	1
CM187	Many Change Requests in maintenance which are Closed and are Major are very long.	1

our analytical approaches. This was specifically useful in determining whether more informed process owners can correctly identify linguistic summaries with a high validity of truth T.

The **Development cycle of issue resolution** was focused at understanding the shared perceptions regarding a regular software cycle release. This information was intended to reveal the average time it takes for software developers and testers for a regular software engineering process.

The **Preferences of data summarization** section was focused on understanding what methodology was preferred within the software development context. We asked the participants questions regarding their preferred format of data summarization, and what kinds of insights do they consider useful in their line of work.

The **Evaluation of linguistic summaries** section was conducted to specifically check the extent to which the degree of truth T is able to represent valid summaries of the issue resolution data. The experiment was set up in a format that randomizes a set of linguistic summaries with different values of T, in order to check whether respondents can correctly identify them.

Complementary information was collected through these questionnaires which help validate the findings and reveal additional insights into the ticket resolution process. The questionnaires can be found in Tables A.1.1 and A.1.1, whereas the complete answers can be found in Table A.2.1.

3.1.11 Present final results

We lastly present our findings in written to the organization and relevant stakeholders. Furthermore, all primary and secondary research is archived and made available for use. All analytical environments are made available to the stakeholders and serve as a road-map for future researchers to recreate the environment and analysis.

Chapter 4

Results

Our secondary research and analysis is focused on analytical activities. The format of the datasets is described in section 3.1.6. Briefly, we analyze:

- classical multidimensional data-set in 3.2 - information on issues ID, State, Project, Severity, Type, Submit Date, Close Date.
- event log in 3.3 - information on activity type and time stamps based on issue ID.

The first dataset is useful in visualizing information from a classical perspective, and deriving insights into issue distributions, allowing for deeper extension of the analysis towards the area of performance measurements such as lead-time, influx-outflux, rates of arrival etc.

The second dataset is useful in reviewing the activities taken towards issue resolution, the compliance of the activities with the workflow, as well as time variations between different activities or entire issue groups.

4.1 Descriptive statistics and visualization

Using traditional approaches we conducted exploratory analysis with the intent to produce statistical and visual summaries of the data-set. We first focused on determining the distribution of the issues based on their length. Immediately the figure 4.1 shows us that the majority of issues are resolved within the first few months, and a large amount within the first year after they have been submitted.

We then focus our attention to generation of specific insights regarding lead-time based on issue, project type, and severity and on whether an issue was successfully resolved or rejected. We first present summaries with visualization (graphs), by focusing on the percentage distribution. Figure 4.2 shows that there are significantly more problem reports than change requests in the system.

Such figures are useful during performance reviews within the software development department. By summarizing with visualization in Figure 4.3, we determine that a slightly larger number of issues are closed successfully, raising questions to what type of issues are being rejected more than closed, or vice versa.

By cross tabulating the number of closed and rejected issues based on lead-time, it is possible to represent the findings in Figure 4.4. This shows that a majority (52%) of rejected issues are completed within a short time interval. On the other hand, a large (32%) percentage of closed issues are average. We categorize issues that require up to 30 days as short, those between 30 to 60 as average, 60 to 120 as long and finally those longer than 120 days as very long. This crisp categorization of our values, allows for a strict division of our data and a variety of cross tabulations with other data attributes.

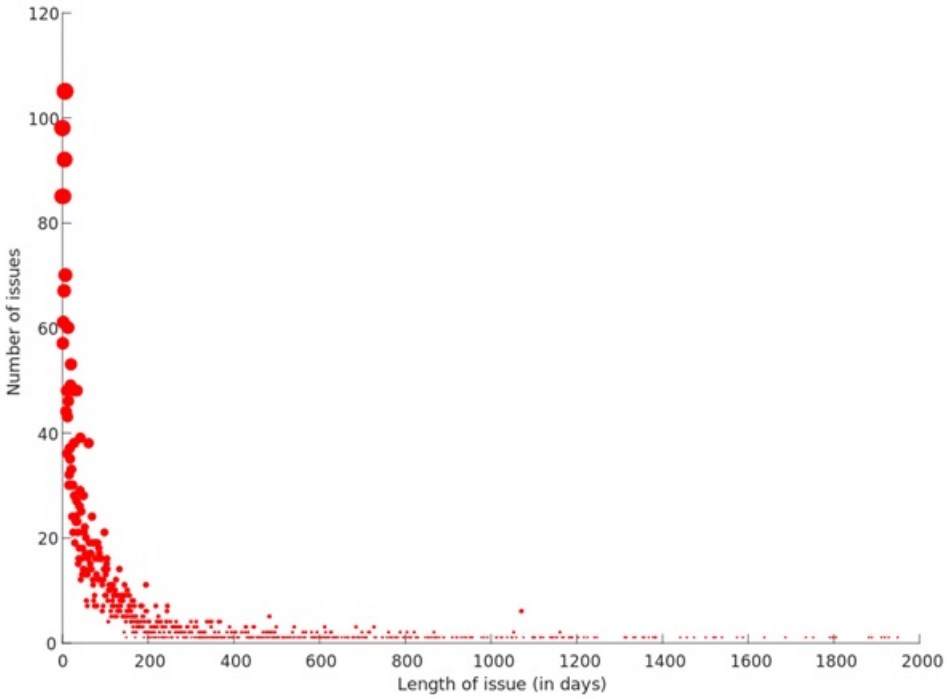


Figure 4.1: Distribution of issues by length (days)

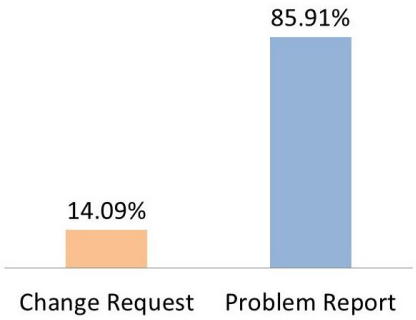


Figure 4.2: Distribution of issues by type.

Using the same approach, we can visualize the data-set and provide insights into the distribution based on severity among the issues. Figure 4.5 allows us to see that Critical severity issues are most commonly solved in a short-time interval. In fact, it shows that the severity of issues is widely compliant in terms of prioritization to resolve issues.

Another perspective is that of distribution by project types. This measure provides key information to the managerial level stakeholders and team-leads for their specific performance. From Figure 4.6, we can interpret that issues in project bucket *field* are most commonly solved within the short-term, whereas issues in *premium* and *transactions* require a long time to resolve.

It is important to also measure the standard deviation and variance of the issue distribution times, to understand whether the mean or median are good representations of the sample. In fact, judging by the distribution in Figure 4.1, we already are aware that our distribution is skewed. In such scenarios, using the mean may provide us with misleading information, and that is why the median is a better representation, of the average lead-time.

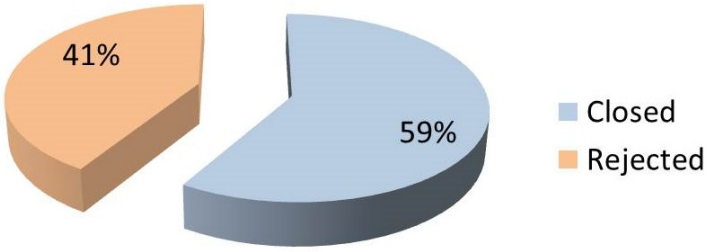
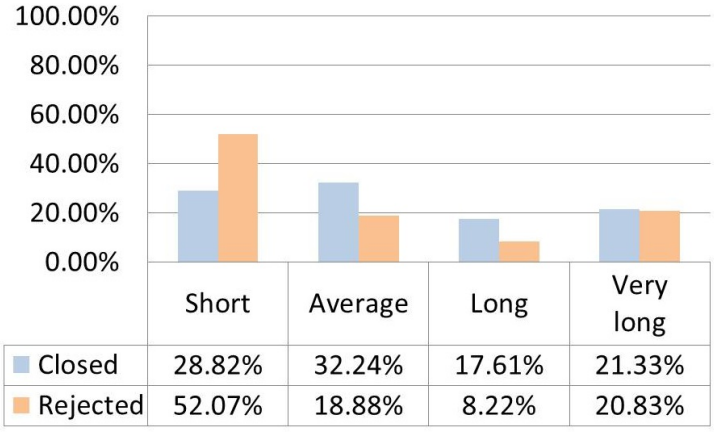
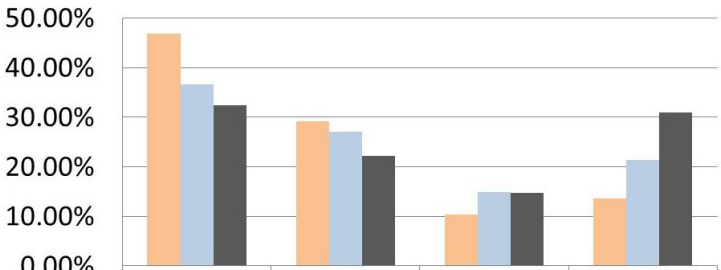


Figure 4.3: Distribution of issues by last state.



	Short	Average	Long	Very long
Closed	28.82%	32.24%	17.61%	21.33%
Rejected	52.07%	18.88%	8.22%	20.83%

Figure 4.4: Distribution of issues length based on last state.



	Short	Average	Long	Very long
Critical	46.88%	29.21%	10.32%	13.59%
Major	36.63%	27.04%	14.90%	21.43%
Minor	32.31%	22.13%	14.70%	30.86%

Figure 4.5: Distribution of issues length based on issue severity.

The standard deviation of lead-times is 230.8, a difference of 230 days among the distribution of issues. This figure is large enough to determine that the mean of our lead-time (128 days) is not a good representation of our data-set. Therefore any representation regarding the average time it

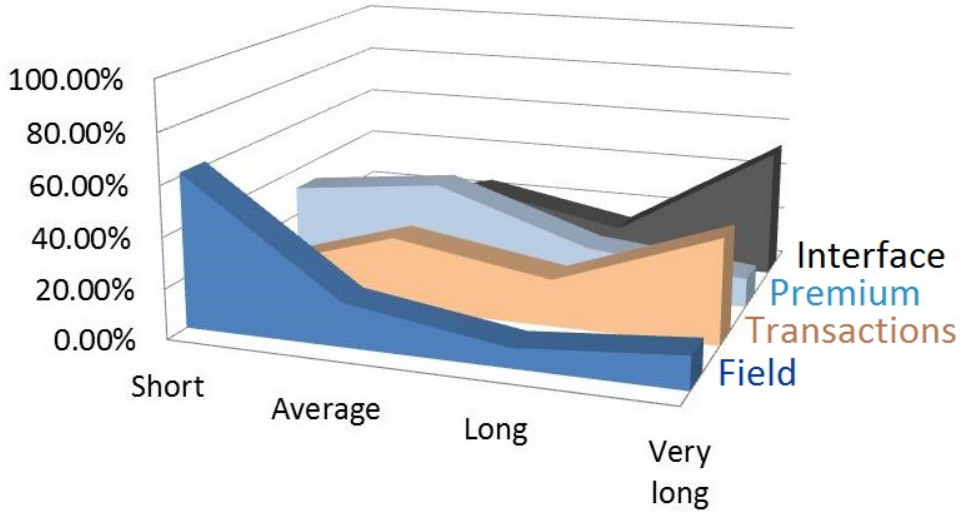
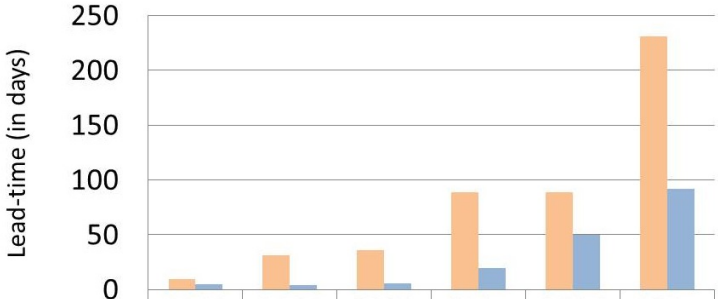


Figure 4.6: Distribution of issues length based on project team.

requires for a group of issues (i.e. based on year) to be completed is better summarized using the median in Figure 4.7. The median value for our data set stands at 50 days, which is the average amount of time it takes to complete an issue. As we can see, the median leadtime is increasing due to the fact that a number of issues are not being completed since their first submission date in 2012. As time passes this growth is more evident in the system, namely a median of 92 days in 2017.



	2012	2013	2014	2015	2016	2017
Average	9.2	31	36	89	89	231
Median	5	4	6	20	50	92

Figure 4.7: Yearly average and median values of lead time.

Using this dataset, we can do additional calculations and provide insights regarding the arrival rate and resolution rate of issues, provide visualizations on the influx-outflux of resolution for a specific time-frame (i.e. a sprint cycle, a release) and various cross tabulations with our data-set attributes. Although these are interesting measures for the process of software resolution but not for our specific scope of research, a simply approach to generate these insights is based on the same principles of the discussed variables.

Event log

When analyzing our event log data-set, using traditional analysis and visualization, we are limited in the extent to which we can generate insights. We can visit the state activities and generate which activity is most frequent among all, though this provides limited information from a process improvement perspective. We group all activities based on their ID and transpose it into a trace. This method generates all the possible traces that exist in our data-set.

Process traces that are most frequently are presented in Table 4.1. The most frequent sequence is of activities which go through a normal procedure of Initiation, Registration, Investigation, Assign, Fix, Verification and finally Closure. The next most present sequence is that of issues that are simply Rejected after Initiation. There are a total of 139 unique sequences, of which the shortest is comprised of 2 activities, and the longest is 21 activities. A full list of the sequences can be found in Appendix A.5.

Table 4.1: Frequent sequences in process event log

Nr.	Sequence	Freq.	% of total
1	Initiated Registered Investigated Assigned Fixed Verified Closed	1237	35.51%
2	Initiated Rejected	449	12.89%
3	Initiated Registered Initiated Rejected	380	10.91%
4	Initiated Registered Investigated Rejected	329	9.44%
5	Initiated Registered Investigated Registered Investigated Assigned Fixed Verified Closed	224	6.43%
6	Initiated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed	121	3.47%
7	Initiated Assigned Fixed Verified Closed	117	3.36%
8	Initiated Registered Investigated Registered Initiated Rejected	60	1.72%

We are limited in what we can do with trace information in this format. That is why traditional methods for visualizing and interpreting such a dataset are not suitable. Therefore, in order to analyze the process performance in more detail, we now switch our attention towards the area of process mining. Using this approach we will be able to visualize the process model and perform a business-oriented analysis.

4.2 Process Mining

Through the use of process mining we are able to generate various performance insights of the development process. It allows us to measure significant metrics in terms of performance — process compliance, time, distribution among process activities. We are able to visually represent the workflow process of resolving an issue and then simulate the system behavior with different arrival rates or process changes.

First we are able to determine the total number of events that occur in the event log, followed by the average and median case duration for data since 2012.

Total cases:	3,485
Total events:	21,480
Average case duration:	88.4 days
Median case duration:	43.4 days

4.2.1 Constructing process maps

We are then able to generate and visually represent this process using the event log data. Visualization 4.8 provides the distribution of all issues among the relevant activities, and all the possible traces during the last 5 years. The process models were constantly reviewed with stakeholders to validate their representation of process reality. In majority of cases, the models were perceived as the correct representation, with exceptions of behavior in cases when issues were rejected without an investigation phase. However, this is a common behavior that occurs (1,017 cases documented) and is further validated by the traces found in the dataset. This qualified as a surprising finding to many stakeholders during discussions.

The models reflect the reality of the process as expected and this was confirmed by multiple stakeholders of the organization. The only challenges that produce surprising behavior is of issues that were manually recorded in the system. Those issues do not make logical sense in the order they are presented (i.e. multiple sequential Initiations), and after drill-down analysis on the issue descriptions, they are commonly incorrect entries by personnel not fully acquainted with the information system.

Figure 4.8 shows the mined process map, along with information on time (median) spent between states (i.e. 10 hrs) and also the absolute case frequency (i.e. 1,345). It is useful in analyzing the process compliance and understanding whether any unwanted behavior is present (i.e. reiterations after rejections). It also provides us with the specific metrics of activity distribution among each state. We can analyze the main path emphasized by the thickness of the arrows, showing what is the most frequent process. From this visualization, it can be derived that there are two main activity variants that occur:

1. Initiated-Registered-Investigated-Assigned-Fixed-Verified-Closed *and*
2. Initiated-Rejected

An additional perspective that is significant in presenting insights into this process, is through the visualization of the average time spent in the activities. As discussed in the previous section, we use the median as a better representative of the average figure for the time-performance metric. In Figure 4.9, we can see the specific time it takes to resolve issues. An issue takes on average 51.5 hours to move from state Initiate to Registered which can be translated to nearly 2 working days. However, it takes 12 days to complete the investigation, and another 48 hours to be assigned to a resource for fixing. After an issue has been assigned, it takes a mere average of 67 minutes (1 hour) to fix it. After an issue has been fixed, it then takes close to 1 week (7.4 days) to verify, and another 28 hours to close it.

Process model in Figure 4.9 also allows us to review the performance metrics of issues that were rejected. Rejected issues are seemingly completed much quicker (68 hours), as in a large

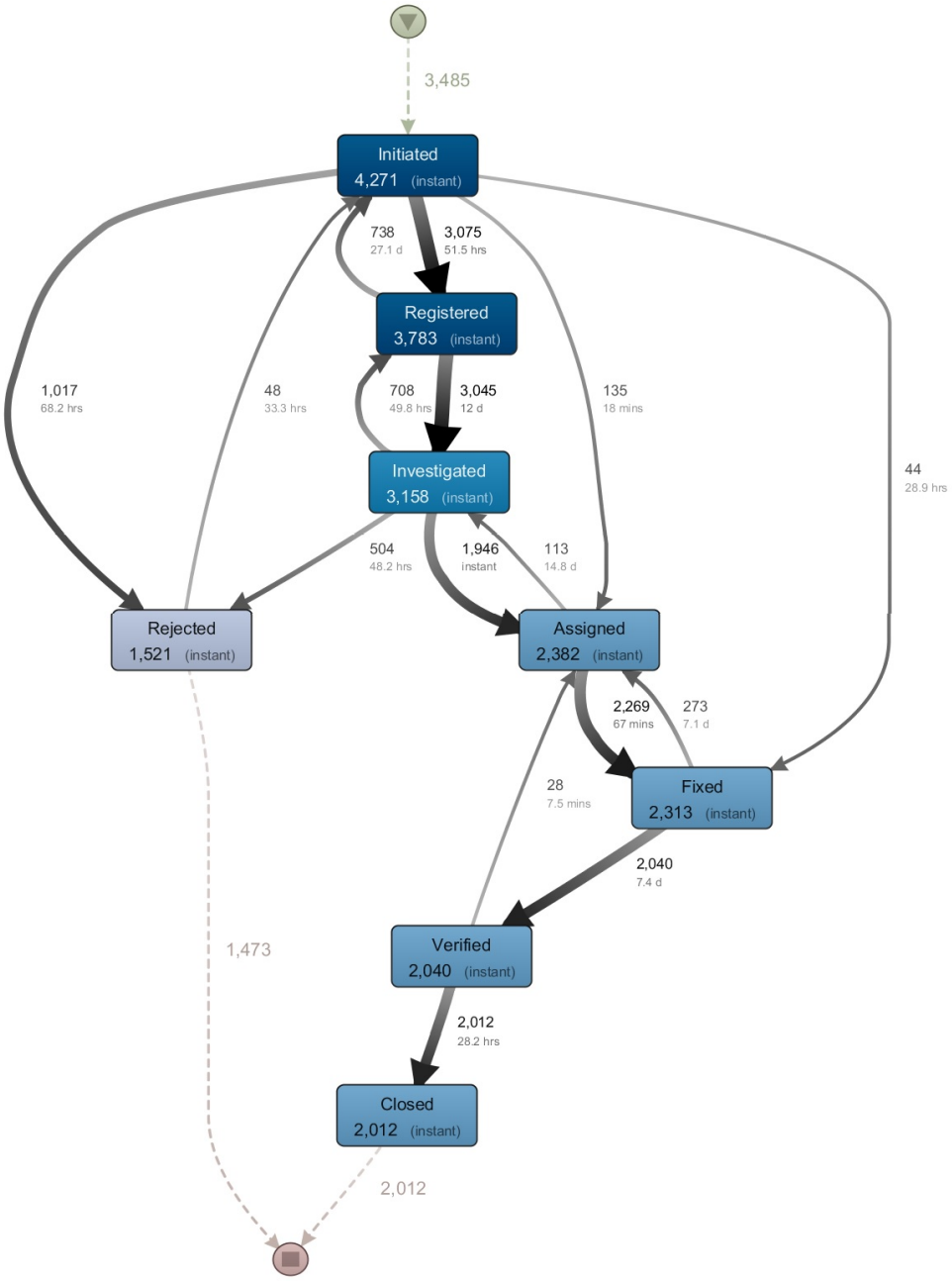


Figure 4.8: Process Map with activity frequency

number of cases they are immediately rejected. The map also allows for analysis of iterations of processes, or other behavior in the process. These behaviors are hard to capture without the approach of process mining, as it takes a tremendous effort to identify the sequence among more than 3900 unique issues in the system.

In fact, the analysis shows that activities which are most time-consuming are the ones that repeat certain process activities. Specifically the major bottleneck appears to occur in cases where a Registered issue is sent back for re-Initiation. This can occur when there is not sufficient information on the problem, and additional information is required by the original submitter before investigation. This occurrence is frequent, as it appears a total of 708 times in the event

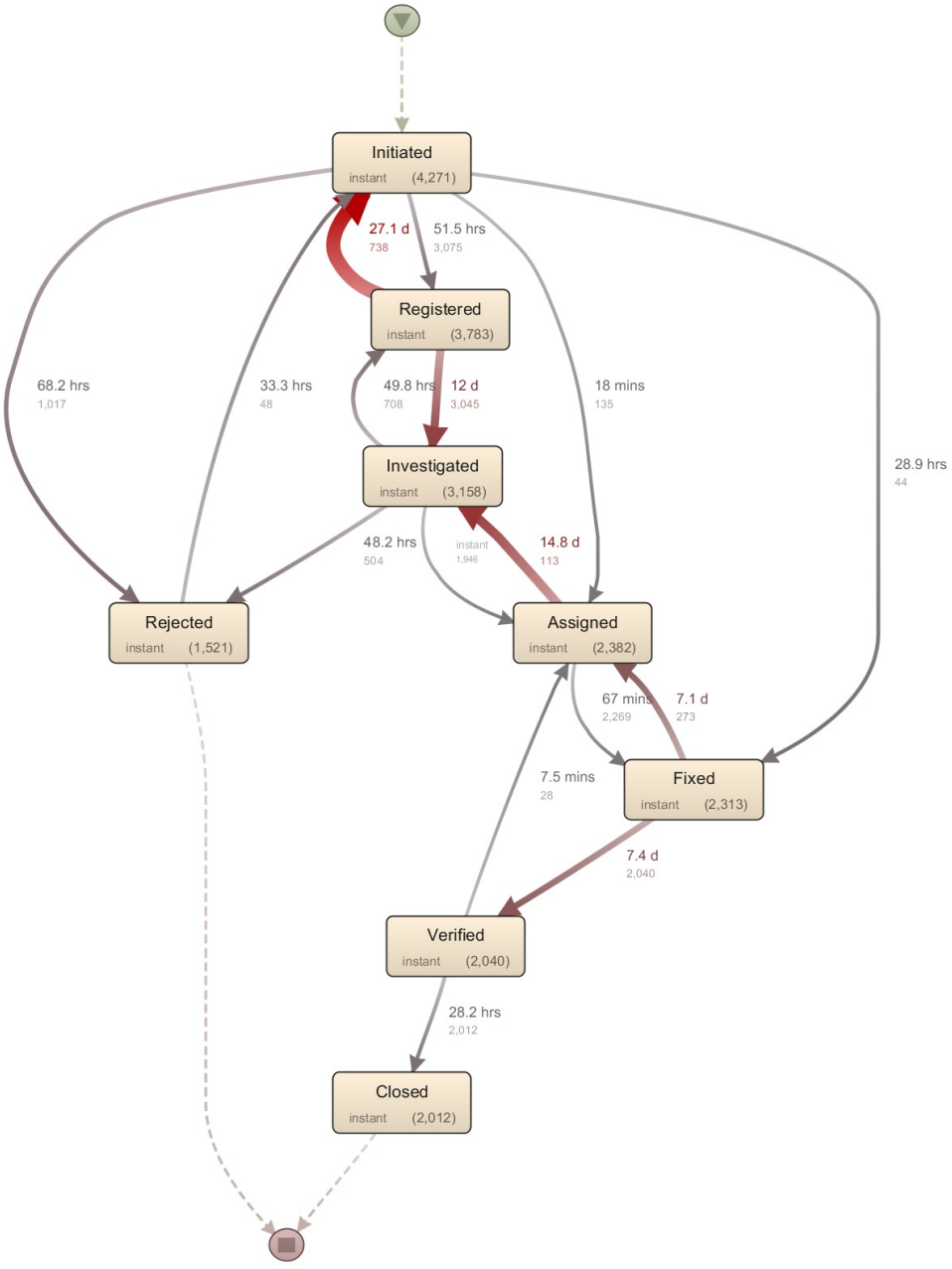


Figure 4.9: Process Map with performance metrics

log, showing that it is a serious hampering to the overall efficiency of the process. This process mining visualization (information) allows us to focus our improvement efforts in areas that require immediate attention. Another area of interest with regard to performance is that of repetitive activities after a case has been Assigned (back to Investigated 14.8 days) and Fixed (back to Assigned 7 days).

Although these iterations are part of the original intended workflow, by analyzing the process data we can present highly valuable insights for the managerial level on what specific areas require attention. Furthermore, we can deepen our analysis into specific variants that occur in the system. The analysis shows that in 35.52% of cases, the process flow is: Initiated-Registered-Investigated-Assigned-Fixed-Verified-Closed. The second most frequent variant is that of refused issues, at

12.88%. We can further enrich this knowledge by particularly filtering all activities with this specific trace, and determine its performance. This allows the stakeholders to view the performance and recurrence of activities within this variant. Other options using this tool include the metrics of the average time it takes to complete issues, the median, minimum and maximum and a plethora of other insights generation options.

Additionally, we can analyze cases specifically based on performance metrics such as short-running cases, or very long ones. This way, we can determine a common obstacle that is evident in the ones with long time-frames. An example is that of cases which take longer than 1 year to complete, it is possible to observe that the majority of time is spend in state Registered, after which a case is sent for re-Initiation after 18.6 months. This occurred 35 times in the filtered process map, and could be an important area of improvement.

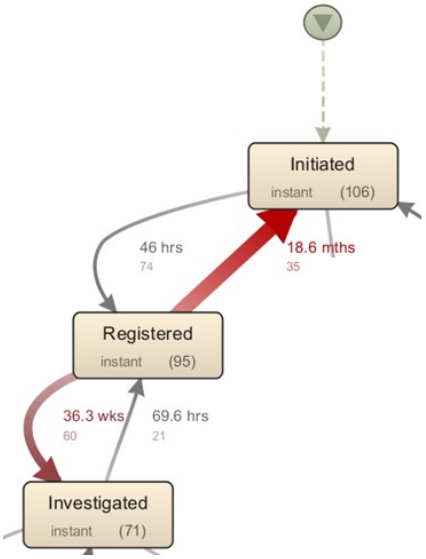


Figure 4.10: View of bottleneck Registered Initiated

For the process map illustrated based on the entire data-set, we can visually observe the sequence of activities and their frequency. The visualization is enriched by the performance metrics of time and distribution. This allows the process owner and team-leads to tackle the development process in the areas where it performs the lowest. It also provides them with the multitude of issues that do not follow the process protocol. One example that was not expected based on the original process 4.18 is shown in 4.11, where a number of cases are Initiated, Registered then re-Initiated only to be Rejected. This is of particular interest as the standard procedure requires that an issue is investigated before it may be rejected, or closed.

We also analyzed the exact amount of working hours spent on activities as summarized in Figure 4.12. This proceses map shows the exact amount of working hours neccesary to go through the different process activities. Based on this visualization the higher management is able to translate the direct costs of resources based on their hourly cost, and determine where to take measures if neccesary. We can conduct similar analysis with regard to performance, compliance and view how the process may have changed over the years by filtering out the dataset by year.

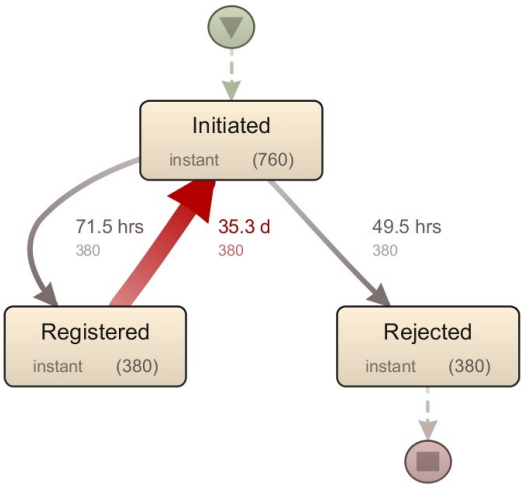


Figure 4.11: Process map with bottleneck cases only

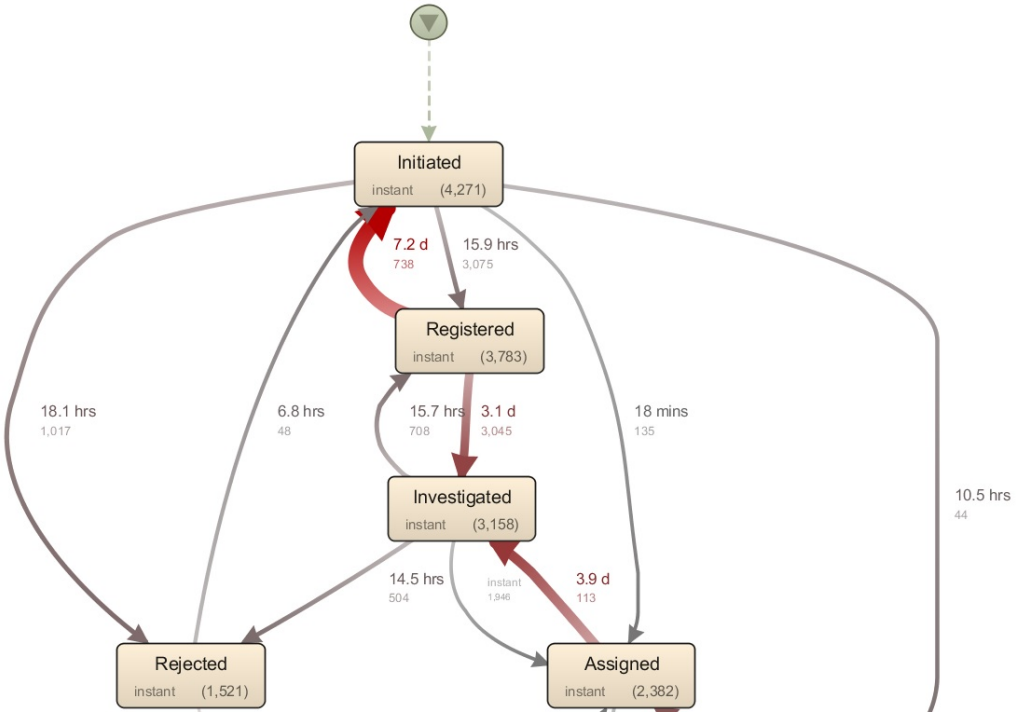


Figure 4.12: Process map considering only business hours

4.2.2 Simulating in Bizagi environment

Drawing from the insights gained through process mining in Figure 4.2 we can reconstruct our process model for further analysis and simulations.

Table 4.2: Performance metrics and distribution of processes

Start state	End State	Nr. of cases	Percent tot.	Avg. time to state
Initiated				
	Rejected	1,017	24%	7.2 days
	Registered	3,075	72%	3.9 days
	Assigned	135	3%	35 hours
	Fixed	44	1%	66 hours
Registered				
	Investigate	3045	80%	41.2 days
	Initiated	738	20%	84.9 days
Investigated				
	Assigned	1946	62%	18 hours
	Rejected	504	16%	5.4 days
	Registered	708	22%	3.5 days
Assigned				
	Fixed	2,269	95%	9.8 days
	Investigated	113	5%	48.7 days
Fixed				
	Assigned	273	12%	20.2 days
	Verified	2,040	88%	22.2 days
Verified				
	Assigned	28	1%	25 hours
	Closed	2,012	99%	6.8 days
Closed				
	End	2,012	100%	Instant
Rejected				
	End	1,473	97%	Instant
	Initiated	48	3%	30 days

We retrieve exact metrics of activity distribution and use them to draw our model within the Bizagi environment as illustrated in Figure 4.18.

Using this environment, we show the behavior of the AS-IS and TO-BE system along with addressing three key bottlenecks in the process. The performance metrics for the simulated model are set using the following information:

- The average time it takes from one state to another
- The frequency of occurrence from one state to another

From the first element we input the averages in order to ensure the model behaves similar to

the real scenario. Using the second element, we ensure that the distribution of cases among the transitions and states is representative of the original model.

The simulation allows for a analysis of the issue resolution. Primarily, we run a 365 days simulation (without specification on arrival times) on the model to test the validity of the model. As expected, the model produces results within the parameters of the data-set a proportion of 75% resolved cases out of the entire initiated ones. A total of 277 issues are submitted and a total of 277 are completed. Among the completed ones, 148 of them are closed and 129 are rejected.

Next, we apply a fixed arrival rate of issues as mined from the data-set. The predicted number of issues for year 2017 is at 2,699, suggesting that there will be more arrivals per day, more resolved issues in total and with some unresolved issues remaining at the end of the fiscal year. We can calculate the arrival rate using information on total arrivals from the data-set within the last 5 years. Furthermore, we use the same information to predict the increase in these figures for year 2017.

Table 4.3: Actual and predicted values for arrivals

	Actual values					Predicted value
	2012	2013	2014	2015	2016	2017
Total arrivals	159	208	516	1,634	2,329	2,699
Arrivals per day	0.4	0.6	1.4	4.5	6.4	7.4

By feeding this information to the model, we can simulate the behavior of the process for year 2017 with key attributes. Successive tests were conducted and the simulation generated valuable insights.

The simulated process model for year 2017 using normal distribution generates a total of 2,699 new issues started. Of these issues, a total of 2,469 are completed, and 230 remain uncompleted and carried towards the next fiscal year. Of the completed issues, 1,364 of them will be successfully closed and 1,105 will be rejected. These results are inconclusive as we need to further validate our model with real data as well as modify the metrics due to the yearly changes the process undergoes.

Thus, as a first step, we conduct a brief analysis on the previous year 2016 and ensure that the produced results are within the parameters of the real data-set for that year.

Testing model for year 2016

Table 4.4: Usage of different metrics for simulations vs. real outcome

Simulations using:	Submitted	Completed	Closed	Rejected
Last five year average metrics	2,329	2,114	1,160	954
2016 average metrics	2,329	1,504	785	719
2016 median metrics	2,329	1,835	1,046	789
2016 median & norm. distr of arrivals	2,329	1,658	1,119	539
Real 2016 outcome	2,329	1,673	1,095	578

For simulated year 2016, a total of 2,329 issues were submitted to the model, of which a total of 2,114 were completed. From the completed ones, a total of 1,160 cases were successfully closed, and 954 were rejected.

The real figures for year 2016 are somewhat different. Out of a total 2,329 total submitted issues, 1,673 were completed - 1,095 of them closed and 578 were rejected. The biggest difference between the simulated results and the real data is in the distribution of the total completed cases and rejected cases.

The reason behind this difference is mainly due to the improvement of the process over the years. The simulated environment uses the performance metrics of the combined 5 years in order

to make these estimations. To account for the continuous improvements of the process over the years, it is imperative that we use the improved metrics of year 2016 to first validate our model. This requires that we re-analyze the performance metrics of year 2016 in isolation and then reapply these metrics to our model.

Testing with 2016 performance metrics

The updated performance metrics had an impact on the overall performance of the process. Data inputs (further illustrated and explained in detail at end of document) ensured that in a span of 365 simulation days, a total of 2,329 cases were initiated. Of those cases, a total of 1,504 were completed. Of the completed ones, 785 were successfully closed, and 719 were rejected.

A careful statistical analysis revealed that the event distribution was not symmetrical, but rather skewed towards short-duration cases. Therefore we decided to use median as a more representative value for the sample set. Using median metrics for state-changes significantly improved the processing time of issues, as well as the accuracy of the model. Using 2016 median metrics the model generated a total of 2,329 issues, of which 1,635 completed 1,046 were closed and 589 were rejected. This data satisfies the real outcome of year 2016, and is thus a credible basis to proceed with future simulations.

Predicting outcome of year 2017

A 365-day simulation for year 2017 taking into account for the trend (information) rate of improvement in the process over the years is conducted on the model. The model assumes an arrival rate of 7.4 issues per day with normal distribution, and a total of 2,699 new arrivals in the system.

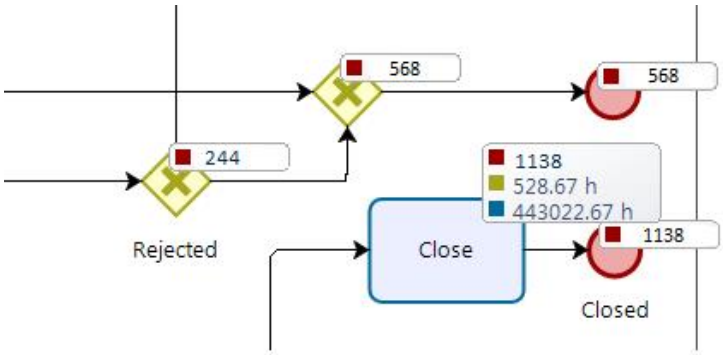


Figure 4.13: Simulation results

Out of a total 2,699 submitted issues, the simulation resulted in a total of 1,706 completed issues, with 1138 of them being closed and 568 being rejected. These results are a reliable estimate figure of the distribution of issues at the end of 2017, if the current trend of process resolution were to continue. This translates into a number of issues being left untreated and carried to the next year. Considering how issues have been piling up for the last 5 years, a careful review of the process and involved resources is necessary. Furthermore, if we assume that the number of resolved issues and the time effort spent is an indicator of the quality of software development, then a direct improvement would also result in better products and services.

Bottlenecks and challenges

Though the overall proportion of time spent in resolving an issue improves over the years, the worrying number of issues that are not closed within a calendar year is a subject that needs to be resolved. In essence, issues that spend more than 2 years within the system are logically no longer of significant value. These issues are frequently closed due to aging though also potentially due

to inadequate resources to resolve them. As these factors are statistically difficult to prove, we focus on simulating the environment by improving the current bottlenecks. We selected three key aspects that delay the resolution process, and then simulate an environment with these improved features.

1. Registered — back to — Initiated

When issues are in state registered, they can either continue for investigation or possible re-initiation. The lead to state investigation takes a total of 12 days, a rather rational amount of time for analyzing and collecting necessary resources to finding an issue resolution. However, when issues are re-initiated, this process takes close to one month.

Figure 4.14: First bottleneck

The frequency which cases go back to re-initiation is 18% of cases. This percentage adds a prolonged amount of time to the overall resolution. In fact, cases with such a trace take an average of 133 days to complete compared to the other cases which take 78 days on average, making it a large contributor to the process inefficiency.

Similarly, the same discrepancy is visible when comparing the median values 75 days in such cases and 40 days without such cases.

2. Assigned — back to — Investigated

After issues are assigned, they can either be fixed or sent back for investigation. The fixing step takes an average of 9.8 days to completed, whereas if sent back to state investigated it takes almost 49 days. The frequency of such cases is near 4% (100 cases) taking valuable time to complete. Such specific cases specifically take an average of 105 days to complete the whole issue, whereas if they did not occur, the average processing time would be 65 days. Comparing to the overall average of 78 days, it is clear that this procedure adds extra processing time.

Similarly the median figures reveal this bottleneck. It takes about an hour to proceed with the fixing step, whereas it takes almost 15 days to go back to state investigated.

3. Registered — to — Investigated

Although the investigation procedure is a crucial step to the resolution of an issue, it follows that a large share of time is spent in this phase. Nonetheless, a total of 41 days on average can be viewed as a somewhat long time in investigating procedure. The median figure for the same transition stands at 12 days. What makes this figure significant in terms of efficiency is that unlike challenges I and II described above, it occurs in a majority of cases 70% of them. Thus a potential improvement of this figure in the future may improve the overall process significantly.

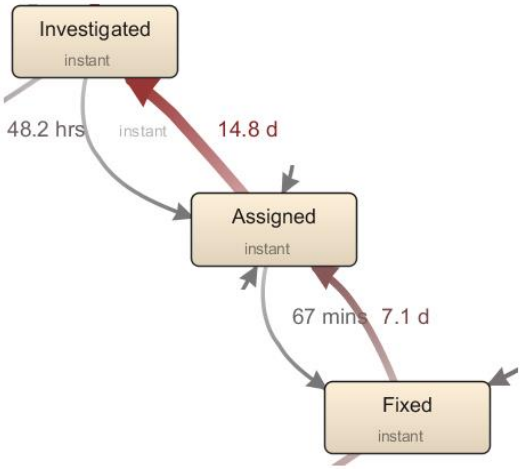


Figure 4.15: Second bottleneck

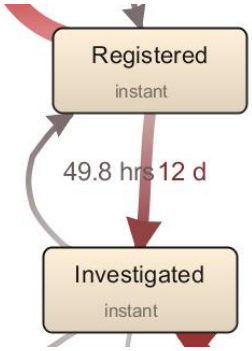


Figure 4.16: Third bottleneck

Addressing the challenges

In order to improve the process, we assume that any improvement in terms of resolution time is a direct improvement to the process itself. Therefore, we take hypothetical measures to remove or shorten the length of bottlenecks. We tackle the challenge in the appearing order, and discuss the impact in performance. Regarding challenge I, by removing traces of issues that are re-initiated, the overall process improves significantly. However, a realistic assumption is that such scenarios can not be removed altogether, as there simply will be cases that require such a procedure. Thus our approach for this challenge will be to shorten the length in simulation and evaluate the quality of this improvement.

From the average (median) of 27 days required for this process, we determine that this task is administrative in principle and can be improved with more structured supervision by the regulating board. The submitter files an issue, at which point the board is responsible for handing the task over to a resource for investigation. At this step, the responsible resource takes an unusual amount of time to attempt the investigation, where the problem is either irreproducible, the information is not satisfactory to investigate or due to some other challenges. Nonetheless, judging how this process takes about 12 days for a successful investigation, it is expected to take a similar amount of time to determine that the information supplement is not enough, or that the issue is registered to the wrong investigating resource. With this assumption, we apply a 12 day average estimate for transition Registered back to Initiated.

Next, we tackle challenge II, where we seek to improve the bottleneck of Assigned back to Investigated. This transition takes a total of 15 days (median), which translates into more than

2 weeks of work. The commonality of such cases is that an assigned resource refuses to carry the fixing step. A variety of reasons could occur here, with the most frequent being that of the resource attempting a fix but without success. As this step is necessary and similar scenarios are possible in the future, we decide not to remove this step from the process, but to improve its performance. Thus, by adding a process reminder to the necessary resource, and potentially to the board for increased awareness, we ensure that the resource tasked with the fixing step raises such an alarm earlier. This way, additional resources can be triggered to help in the resolution process and shorten this lengthy transition. We make the assumption that a reminder would shorten this procedure close to the amount it takes to successfully resolve an issue 7 days.

Tackling challenge III requires more insight into the investigation procedure. Since this is a normal and integral process that takes 12 days (median), any improvement in this figure would be significant to the overall process. Judging by the nature of the transition, it is difficult to assume any improvement of more than 15% within the first year. Through a combination of added resources (developers, testers etc.) and a reminder service to motivate the completion of investigation phase, a 15% improvement in time is feasible within the short-term. This improvement translates to a transition that takes 10.2 days instead of the current 12 days.

The improved model

After applying the aforementioned improvements, the simulated model was able to more efficiently address issues. Out of a total 2,699 submitted issues, the simulation resulted in a total of 2,278 completed issues, with 1,538 of them being closed and 740 being rejected. The improved transitions enabled the resolution of more issues in the long-term, and thus have a direct positive impact on the overall process of issue resolution.

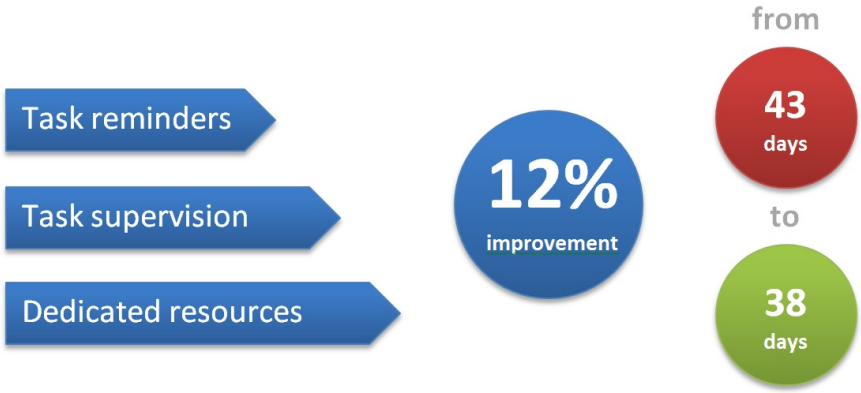


Figure 4.17: Reflection of improvements on process

Table 4.5: Simulations for 2017 environment

2017 simulations	Submitted	Completed	% Completed	Closed	Rejected
Predicted trend model	2699	1706	63%	1138	568
Improved model	2699	2278	84%	1538	740

The simulated environment shows that the enhancements in these particular areas improved the average resolution time from the original 43 days (median) to 38 days (median). This is a 12% improvement in terms of resolution performance, which is feasible by only following the aforementioned approaches.

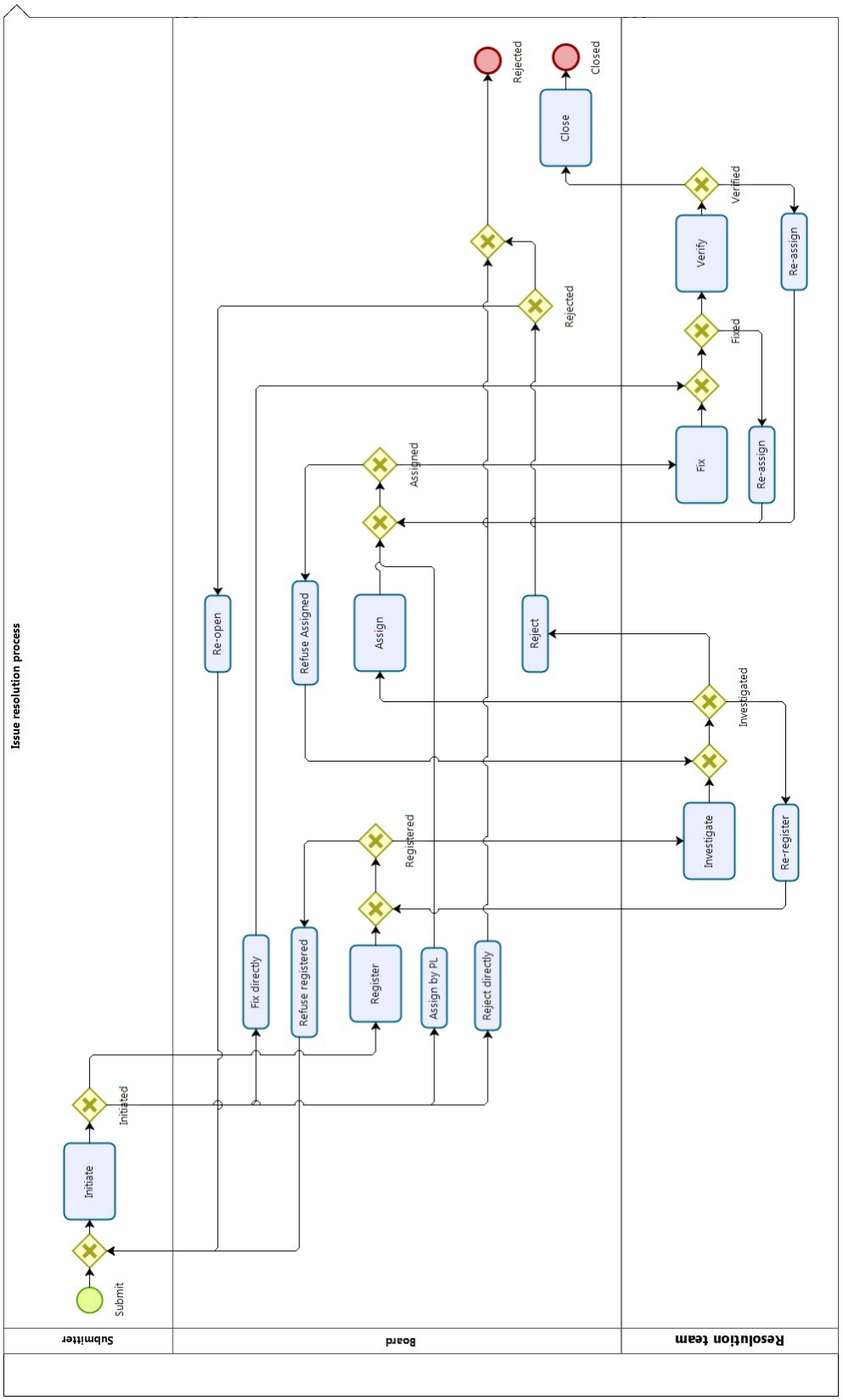


Figure 4.18: Process model using BPMN

4.3 Linguistic summarization

We apply no pruning to our generated summaries with regard to truthfulness, as the intent is to also check the validity of generated summaries with stakeholders in the organization. However, due to a large number of produced summaries, we present an excerpt of the results based on T value. Furthermore, we exclusively focus on the lead-time as our summary predicate. Results are presented and then validated in Chapter 5.1 to understand whether high truthfulness sentences are perceived as accurate and low truthfulness ones can be identified as inaccurate by the stakeholders.

In the first step, we generate summaries based on the issue type. In 4.6, we present summary sentences with a truth value larger than zero. A total of six basic summaries with validity exist, out of a total 337. The brute-force approach of our linguistic summary generation results in such a high initial number of results which are filtered using the degree of truth T.

Table 4.6: Basic (specific) summaries with T >0

	Summary sentence	T
B1	Very few issues of type: (None) are short.	1
B3	Very few issues of type: (None) are long.	0.3
B4	Very few issues of type: (None) are very long.	1
B8	Few issues of type: (None) are very long.	1
B12	Many issues of type: (None) are very long.	1
B16	Most issues of type: (None) are very long.	0.35

The basic summarizes provide limited insights into the process. Of the summaries with some or high degree of truthfulness (T >0), we can infer that a large number of issues that were not tagged with issue type information, the resolution process takes a very long time to complete (summary nr. B4). Sentences with the quantifier of events that do not occur frequently (very few) produces multiple summaries (B1, B3, B4) as it is more likely that certain properties of issues exists at low levels. Sentence B4 overrides B3 in that it is a subset of the main one. Basically, this sentence informs us that a large number of issues take a very long time to complete, which is a direct metric into the performance of the system. This is also confirmed by the counter-sentences generated with no truth value: *B13 most issues are short. T = 0* (a full list is present in Table A.4.1.

Next, we check whether there is any correlation of issue length based on whether an issue is a Change Request or a Problem Report. Summaries IT17, IT18, IT19, IT20, IT33, IT34, IT35, IT36 show that there is manifestation of all case-lengths among small groups of issues. Specifically, *very few* cases are present among *short, average, long, very long* groups with a high degree of truth T. Summaries IT17 — *Very few Change Requests are short* — or IT20 — *Very few Change Requests are very long* — do not provide substantial insight, since we already expect to have issues among all quantifiers (case-lengths). Thus, we shift our attention towards other summaries.

Table 4.7: Basic summaries exhibiting limited (very few) behavior

	Summary sentence	T
IT17	Very few Change Requests are short	1
IT18	Very few Change Requests are average	1
IT19	Very few Change Requests are long	1
IT20	Very few Change Requests are very long	1
IT33	Very few Problem Reports are short	1
IT34	Very few Problem Reports are average	1
IT35	Very few Problem Reports are long	1
IT36	Very few Problem Reports are very long	1

Similarly we generate summaries based on the project types, whether issues were finished by

rejection or proper closure and lastly by the severity of issues. The summaries generated do not show interesting insights, as there are no particular differences found among the attributes. Sentences with high degree of truth are present only in very few issues as presented in Appendix A.4.1. However, it is interesting to see that some issues exhibit a more frequent behavior. Summary TM53 shows that a few issues in $field$ are short, whereas summary TM231 shows that in *cloud* a few issues are long.

Table 4.8: Basic summaries exhibiting frequent (few) behavior

	Summary sentence	T
TM53	Few issues in field are short.	1
TM231	Few issues in cloud are long.	1
ST277	Few rejected issues are short.	1
SV293	Few critical issues are short.	0.8

The more complex summarizations involve the combination of all variables from our dataset. This combination results in the generation of a very large number of summaries: 1,905 unique summaries. Due to the large number of summaries, we can apply manual pruning to only view summaries that are insightful. The removal of all sentences with a truth degree of zero results in a remaining 610 summaries. To present more relevant summaries, we also select summaries with a degree of truth T higher than 0.7. This yields 464 summaries, still a high number to consider as a suitable summarization.

We are interested more frequent behavior in our system, namely in issues that occur in *many* or *most* situations. This yields a total of 25 sentences that summarize the dataset. We present this filtered list in two tables - Change Requests and Problem Reports - in Table 4.9 & Table 4.10, and the complete summary list in Appendix A.4.2.

These summaries provide insight into behaviors that are very complex to extract with different approaches. They show trends which are of particular interest to the stakeholders in terms of expectations, and information that reveals the development lead-time based on issue type, project, severity, and how if issues were closed or rejected.

4.3.1 Interesting findings

From the summaries, we can infer that Change Request issues of severity level *critical* are quickly investigated. Summary CM61 in Table 4.9 summarizes the behavior that such issues are $rejected$ in a *short* time. In fact, summaries CM73 and CM89 emphasize that even issues with severity level $major$ and *minor* are often times *short*.

Summaries CM172 and CM187 in Table 4.9 show that Change Requests in *maintenance* generally take a long time to complete, without regard to severity or their final state. Similar findings can be concluded for Change Requests in delivery (CM348, CM352, CM428).

The findings suggest that Problem Reports have somewhat different characteristics. Linguistic summaries CM745, CM761 and CM777 in Table 4.10 show that rejected issues in field mostly have short times. Similarly, issues in maintenance require a short amount of time to complete. This finding is contrary to the ones in Change Requests, as they take longer to complete there.

Lastly a large share of Problem Reports in projects transactions and interface disregarding their final status all take a very long amount of time to complete. Summary CM1600 shows that in fact almost all issues in transactions of Minor severity require a very long time to complete. A drill-down on this specific group, reveals a total of 7 issues that take an unusually long amount of time to complete — specifically an average of 961 days.

4.3.2 Surprising findings

When presented with a summary list, many of the stakeholders were surprised that a majority of issues are not completed within the short term (30 days). That is the case for almost all projects

Table 4.9: Complex CR summaries with T >0.7

	Summary sentence	T
CM61	Most Change Requests in field which are Rejected and are of Critical severity are short.	1
CM73	Most Change Requests in field which are Rejected and are of Major severity are short.	1
CM89	Many Change Requests in field which are Rejected and are of Minor severity are short.	1
CM172	Many Change Requests in maintenance which are Rejected and are of Minor severity are very long.	1
CM187	Many Change Requests in maintenance which are Closed and are of Major severity are long.	1
CM202	Many Change Requests in maintenance which are Rejected and are of Major Severity are average.	0.83
CM348	Many Change Requests in delivery which are Closed and are of undefined severity are very long.	1
CM352	Most Change Requests in delivery which are Closed and are of undefined severity are very long.	1
CM428	Many Change Requests in delivery which are Rejected and are of Minor severity are very long.	0.83
CM524	Many Change Requests in linux which are Closed and are of Minor severity are very long.	0.83
CM604	Many Change Requests in MS windows which are Rejected and are of Major severity are very long.	0.83
CM620	Many Change Requests in MS windows which are Rejected and are of Minor severity are very long.	1

Table 4.10: Complex PR summaries with T >0.7

	Summary	T
CM745	Many Problem Reports in field which are Rejected and are of Critical severity are short.	1
CM761	Many Problem Reports in field which are Rejected and are of Major severity are short.	1
CM777	Many Problem Reports in field which are Rejected and are of Minor severity are short.	1
CM937	Many Problem Reports in maintenance which are Rejected and are of Critical severity are short.	1
CM969	Many Problem Reports in maintenance which are Rejected and are of Minor severity are short.	1
CM1356	Many Problem Reports in storage which are Rejected and are of Minor severity are very long.	0.78
CM1548	Many Problem Reports in transactions which are Closed and are of Minor severity are very long.	0.83
CM1596	Many Problem Reports in interface which are Rejected and are of Minor severity are very long.	1
CM1600	Most Problem Reports in transactions which are Rejected and are of Minor severity are very long.	1
CM1644	Many Problem Reports in interface which are Closed and are of Minor severity are very long.	1
CM1660	Many Problem Reports in interface which are Rejected and are of Critical severity are very long.	1
CM1692	Many Problem Reports in interface which are Rejected and are of Minor severity are very long.	1

and issue types, resulting in some interesting discussions during the interview. An interesting finding was emphasized by multiple interviewees with regard to summary CM745, where Problem Reports that are rejected and are of critical severity are rejected very quickly. According to the interviewees, this can be considered as a positive metric in the sense that issues with critical severity are completed within a short amount of time. However, the same could not be claimed regarding problem reports in project *interface*, where regardless of the severity, many issues take a very long time to complete.

Summaries CM937 and CM969 were a positive surprise for the interviewees, and especially for the stakeholders directly related to the process management. The fact that issues in *maintenance* are rejected in a short notice is a positive behavior. According to them, the process is suffering from long-overdue cases, and not necessarily from the turnout of the resolution process. If issues are present in the system for a long time and minimal action is taken to complete them, this translates into a big cost for the organization.

The findings were relatively interesting to the stakeholders. Additional informal feedback received hinted towards a lot of interest in issues that require a long amount of time to complete, specifically in Change Requests in *maintenance* (CM172, CM187), Motion (CM348, CM352, CM428) and *MSwindows* (CM620).

4.3.3 Wrong or very surprising findings

It was difficult for stakeholders to answer this question with specifics regarding the summaries. The biggest concern about the findings were expressed with regard to summaries that generated information about issues with no type, and no project information. Multiple stakeholders were convinced that since the input format does not allow for such behavior, these summaries are impossible to generate. However, this is a misrepresentation, since the data-set does in fact contain such entries as proven by the traditional analysis in the early stages of the project. After drill-down, a total of 30 issues were found containing no information on severity or issue type, and were validated by re-querying the original database. The reason for such cases is that users manually changed the attribute after the issue had already been submitted, something that is possible only through editing of an issue (not through submission form).

4.3.4 Missed insights

None of the interviewed stakeholders had direct feedback regarding any missing information from the data-set. Their input was largely constructive and suggestive in extending this approach to other departments and data-set, to see trends in other segments. They claim that the current data-set under analysis provides a limited view of the resolution process. Combining data sources from other departments would possibly provide additional insights to the resolution process, and add more value to the analysis overall.

4.3.5 Event log summaries

We have also implemented an existing (recent) approach for summarization of the event log as developed by Dijkman and Wilbik [15]. Due to unreliability of results and consultation with expert (prof. Anna Wilbik), we briefly present the findings but make no claims or validation attempts.

In essence, the summarization produces clusters of sequences that occur more frequently in our data. The summaries are identical in comparison to our traditional brute-force sequences (using Python), although also produce common trace combinations without a start or end activity. These findings can help us in identifying a recurrent sequence that may be in-compliant with the originally intended workflow.

What we can infer from the produced summaries, is that a large share of the events share the initial sequence of being Initiated and Registered (EV23). This is not a significant insight, since we already are aware of the process context. However, the fact that this summary has a degree of

Table 4.11: Excerpt of event log summaries with T higher than 0.7

	Summary sentence	T
EV1	Many cases contain sequence: Initiated, Registered, Investigated, Investigated	1
EV2	Many cases contain sequence: Initiated, Registered, Investigated, Assigned, Fixed, Verified	1
EV3	Many cases contain sequence: Initiated, Registered, Investigated, Registered	1
EV4	Many cases contain sequence: Initiated, Registered, Investigated, Rejected	1
EV5	Many cases contain sequence: Initiated, Registered, Investigated, Verified	1
EV6	Many cases contain sequence: Investigated, Registered, Investigated, Assigned	0.75
EV7	Many cases contain sequence: Fixed, Assigned, Fixed, Verified, Closed	0.75
EV8	Many cases contain sequence: Initiated, Assigned, Fixed, Verified, Closed	0.78
EV9	Many cases contain sequence: Initiated, Registered, Investigated	1
EV10	Many cases contain sequence: Registered, Investigated, Assigned	0.72
EV11	Many cases contain sequence: Initiated, Registered, Investigated	1
EV12	Many cases contain sequence: Initiated, Registered, Initiated, Rejected	0.70
EV13	Many cases contain sequence: Initiated, Registered, Verified	0.99
EV14	Many cases contain sequence: Initiated, Registered, Rejected	0.98
EV15	Many cases contain sequence: Initiated, Registered, Registered	0.97
EV16	Many cases contain sequence: Initiated, Registered	0.97
EV17	Many cases contain sequence: Initiated, Registered, Investigated	1
EV18	Many cases contain sequence: Initiated, Registered, Fixed	1
EV19	Many cases contain sequence: Initiated, Registered, Delayed	0.97
EV20	Many cases contain sequence: Initiated, Registered, Closed	1
EV21	Many cases contain sequence: Initiated, Registered, Assigned	0.98
EV22	Many cases contain sequence: Initiated, Assigned, Investigated	0.81
EV23	Almost all cases contain sequence: Initiated, Registered	0.80

truth 0.8 shows that there are sequences which do not share these characteristics. So in turn, this does provide valid insight into what a potential problem with the process compliance could be.

Furthermore, when informally investigating these sequence results with stakeholders, it was very surprising to them how there are cases that can be rejected without an investigation phase. In fact, such a trace has a large presence in our log, as can be validated by summaries EV12 and EV14. Such event traces were also found using the previous process mining approach and the entire sequence list. The summaries generally show that the process data is compliant to the intended workflow, and the support for such traces is high (T values larger than 0.7 selected for preview).

4.4 Comparison of aproaches

The application of traditional data analysis, process mining and linguistic summarization was necessary to reach conclusions regarding the benefits of each method. According to theory and practice, as well as stakeholder preference based on their input during the interviews, it is determined that no approach is distinctly favored over the other in principle. According to informal discussions and during result presentation, stakeholders claimed that process mining and linguistic summarization provide deeper and sometimes surprising knowledge on the issue resolution process.

It is challenging to generate direct correlations using simple analysis, furthermore made difficult by the fact that our data-set contained mostly textual data. The traditional approach was useful in showing the exact metrics and distribution of data, and translating them to a visual representation using graphs or charts. However, the process of analysis and translation was manual and time-consuming. This is a major drawback in today's market as data congregates continually and such methods suffer from high variation.

By utilizing a process mining approach, it was possible to analyze the sequence of issue resolution by checking for conformance, modeling the process, and measuring all the exact metrics. This approach provided with a significant amount of interesting insights regarding the workflow, by pointing specific bottlenecks in the process, hinting towards the type of issues that cause such behavior, and allowing the user to view different perspectives over the years, releases, types of issues, based on length, process-flow etc. Though process mining brings valuable insights, it is also limited in what it can derive from the data.

That is why we felt it is crucial to apply a third and final approach to analyze our data-set, that of linguistic summarization using fuzzy logic. This method allows a deep understanding of the correlation between data fields, represented by fuzzy-set membership functions, allowing for more robustness and detailed analysis of the information. Results generated using this approach hint towards different metrics through different groups of data, with a high variation regarding lead-time performance of different projects, issue types, and other related variables. Most importantly, the method is able to correctly identify trends within specific projects, which is a significant indicator to team-leads and managerial level stakeholders. Through this information, recommendations and changes can be made to the process.

In conclusion, the three methodologies provide significant insights into the resolution process. Managerial level stakeholders preferred the provision of textual (linguistic summarizations) and visual data representations, whereas the software engineers heavily favored the visualizations for reasons of detail and ability to make individual interpretations. The linguistic summaries were favored among top-level staff due to the concise summaries that capture the important elements and provide the findings in a single explicit format.

Chapter 5

Validation and user evaluation

5.1 Validation

The stakeholders rated the insights positively throughout the study period. During presentations and informal discussions, they constructively reviewed the generated findings using traditional analysis and process mining models. Regarding basic data analysis, stakeholders were surprised that there is a large difference in terms of resolution time between problem reports and change requests.

When presented with process models, stakeholders often questioned the fact that there are possible traces in which issues are rejected without investigation. Upon drill-down, it was concluded that such behavior does in fact occur, and that these are issues immediately rejected by either the submitter or the control board (due to inaccuracy, inability to reproduce issue or aging). Further consideration was given to process changes — such as the introduction of a Propose Reject activity — which was short-lived and simply caused process noise. Upon discussion with process owners, these activities were clustered so that the process maps presented a clearer, more understandable format.

Regarding the validation of linguistic summaries, we informally reverse-check a number of highly valid and completely invalid summaries with out original dataset. This confirmed that the produced summary behavior is found in the dataset, and thus the sentences are accurate.

5.2 Evaluation

Evaluation of descriptive statistics and process mining was not done to a formal level. This is due to the fact that it was informally communicated multiple times during discussions with stakeholders and was proven satisfactory. Rather, we put our attention towards the evaluation of linguistic summaries.

Through our experiment we validate the extent to which the degree of truth is able to represent truthfulness of statements. We asked participants to review the summaries on a scale from 1 to 10, with 1 being *completelyuntrue* and 10 being *completelytrue*. Through our questionnaires, we combined the steps of gathering additional information for internal purposes and evaluation of our findings. The sample questionnaires for managers and testers/developers can be found in Appendices A.1.1 and A.1.1 respectively.

The results of our evaluation show that users with familiarity to the process are highly likely to rate high degree of truth summaries with a high score. This shows that in fact the degree of truth in summaries is a good metric for determining the validity of summaries. The distributed average score among managers and software engineers are summarized in Table 5.1 as averages for groups of sentences, and the individual score summaries are summarized in Table 5.2.

Overall, the results validate that linguistic summaries can be correctly identified by the stakeholders to a great degree. This confirms that stakeholders are aware of the process, and that the

findings are valid. There is however, room for improvement regarding the validation of findings with very low degree of truth. Overall, managers scores low validity sentences with an average score of 5.3, whereas software engineers with a score of 5.7. Although ideally we would expect such sentences to have a very low score (close to 0), the fact that they are scored as lowest among the groups is also a positive conclusion.

In fact, the average scores for groups of sentences increases proportionally with the degree of truth. This confirms that stakeholders are well-informed about the process, while also showing that linguistic summarization provides credible insights. Furthermore, we asked our stakeholders if they are informed about the process and data overall. From the stakeholders with great knowledge of the process (>7), the scoring is similar with a slight edge on the extremes. Briefly, they are able to tell low validity sentences slightly better than others (score them as 5.1 vs. 5.5 average), and similarly score the high degree of truth sentences with a higher value (7.8 vs 7.55 average).

However, the more unexpected result was that stakeholders would occasionally rate completely untrue sentences as valid (although the average was low). It is difficult to come to a conclusion regarding this behavior, as it could be simply due to a misunderstanding of the summary. When probed specifically about this issue, the stakeholders pointed towards the data-set itself. In their opinion, it is impossible to have summaries where there is no project type definition or no severity defined. However, the data-set itself suggests otherwise, even though these are mandatory fields in the input form of the ticket-resolution platform. This is due to users editing the issue variables after the initial submission, an action that is allowed by the current process.

Table 5.1: Average scores for summary groups

	Managers	**SW Engineers**
Summaries 0 to 0.25	5.3	5.7
Summaries 0.25 to 0.5	6	5.9
Summaries 0.5 to 0.75	6.4	6.7
Summaries 0.75 to 1	7.8	7.3

Table 5.2: Average scores for all summaries

	Managers	**SW Engineers**	**T**
ST281	5.5	4.8	0.12
CM13	4.7	5.2	0
CM765	5.7	6.2	0.11
IT45	5.3	6.6	0
SV309	6.0	5	0.36
B16	5.2	6	0.35
CM140	6.5	6.6	0.5
CM980	6.3	6.2	0.41
TM57	6.3	6.6	0.58
CM176	6.8	6.6	0.7
CM1696	5.7	6.6	0.7
TM200	6.7	7	0.57
B1	7.5	7	1
B12	7.8	7.6	1
CM61	8.2	7.2	1
CM187	7.8	7.4	1

Chapter 6

Conclusion

Our research resulted in the generation of numerous insights into the software setting. From the initial stages of data gathering, validation and down to implementation and application of our three methodologies, we were able to document the means of insight generation in software development defect-management systems. By investigating each approach individually, we generate insights that are relevant to the case study organization, and the software development industry as well.

The three methodologies provide significant insights into the resolution process. Managerial level stakeholders preferred the provision of textual (linguistic summarizations) and visual data representations, whereas the software engineers heavily favored the visualizations for reasons of detail and ability to make individual interpretations. The linguistic summaries were favored among top-level staff due to the concise summaries that capture the important elements and provide the findings in a single explicit format.

Using data analysis approaches, users can generate various insights into the software defect-management processes.

Our elaborate analysis on the potential of three methodologies provides a road-map for future data analysts of managers who are seeking to extract information from their data. By reviewing the insights generated in our methodology, it is possible to determine what each approach can reveal, and subsequently determine the feasibility of such a project. Furthermore, by following our implementations and approaches it also helps the scientific domain with practical applications through a eloquent case-study, with direct feedback from the domain stakeholders.

Another important element in our study was the evaluation of findings using the linguistic summarization methodology. By checking how the data summaries are validated by stakeholders without the provision of truthfulness degree, we could confirm that the degree of truth is a relatively correct estimator of data in Chapter 5.1. This is an interesting finding recalling that extensive research is conducted in extending the measures for quality control in the generation of linguistic summaries by Hirota et al. [17] and Wu et al. [54].

In brief, our study show the strengths and weaknesses of each methodology and approach, and provides a practical approach for generation of insights into software development defect-management. Using time as a key metric, we generate deep insights into the process, areas of improvement and a road map of how to reproduce such results. Our approach can be extended to measure other important metrics within software development such as code size growth, effective lines of code, text mining of issue descriptions and additional elements. Nevertheless, we believe that this document can serve as a point of reference for data analysts in selecting their data extraction and interpretation methodologies within the domain of software development.

6.1 Reccomendations

Due to the exploratory nature of the study, we are limited in the provision of direct recommendations regarding the process improvement. Our recommendations are rather presented as conclu-

sions regarding the suitable methods that can be used in the software development industry. In brief, the organization stakeholders should focus their efforts in utilizing our provided tools and environments to keep track of their process variables. Due to the ease-of-use of process mining tools, we recommend an of-the-shelf solution which can provide substantial amounts of insight to the managerial level. Concise recommendations regarding the software defect-management process were derived from process mining and the simulation process. These recommended improvements are presented in detail in section 4.2.2.

In addition, we recommend that the stakeholders revisit our methodology and conduct a similar analysis using extensive forms of linguistic summarization, with which interesting data summaries can be explored and analyzed. We encourage the stakeholders to make use of this environment for other data sources within the organization. A combination of insights from linguistic summarization and traditional data analysis can reveal future performance issues with regard to projects teams, types and more.

Overall, our direct recommendation is to primarily make much more use of the available information systems. The data records stored contain important information on the performance of the departments and overall software quality. By measuring these figures continuously using our proposed methodology, the organization can improve its decision making and translate these into improved software defect management.

Furthermore, efforts in cleaning the data are important to conduct ad-hoc and valid analysis. The system that contains the information is unfriendly to the average user, and provides unnecessary information during the extraction process. Close to 450 variables are available for extraction, and only 20 of them contain useful information. Users find it difficult in conducting valid analysis with the current set-up.

Forms of issue submission have loopholes regarding validity of entered data. The format still allows for unwanted behavior that leads to issues containing no project, severity and other variable information. This later translates into confusion during the analysis process, and can be addressed easily by the workflow management system administrator.

6.2 Limitations of the study

Although this research was carefully prepared, we are aware of its limitations and shortcomings. The main limitations of this study are methodological and researcher ones. Regarding methodology, we are limited in terms of measures used to collect primary data, sample size and the issue of self-reported data. In brief, the interviews would ideally be conducted earlier and at a slower pace, with more feedback from all relevant stakeholders. To address with this limitation we often had to re-design the questionnaire, and repeat interviews to collect the right information. Regarding the sample size, we interviewed a total of 11 stakeholders, a rather small number to make general claims regarding our quantitative results. The sample size was limited by the relevant stakeholders and interview timing, thus making it difficult to reach a larger number of respondents. However, as much of our research provided qualitative feedback as well, this issue was managed by asking more open questions to the relevant stakeholders. This qualitative feedback, although very useful, was at times difficult to document as the nature of conversations was often informal. We attempt to address this limitation by continuously requesting feedback and reflecting on it in our final results.

Regarding researcher limitations, we are limited in terms of access and longitudinal effects. It was challenging to gain access to the right information throughout the duration of the study. Due to hierarchy and access privileges, extensive amount of time was required to use particular systems — specifically in extracting event log data. This had an impact on the extent to which we could implement our process mining experiments. In terms of time, the 6-month duration was relatively short for developing the correct research methodology and analytical environment of linguistic summaries. To address this issue, we narrowed our scope of study towards more simplistic implementation of linguistic summaries and shorter experiments for its evaluation.

6.3 Future research

There are a number of gaps in our knowledge around software-defect management processes. Although our research attempts to bridge the methods and suitable analytical tools according to current literature and practices, there is a need for additional research and validation specifically in:

1. In-depth exploration of how linguistic summaries translate into concrete knowledge for the organization stakeholders. Further research might analyze and present various summaries to the stakeholders and then measure the extent that these insights improved the software defect-management process. This could be done via a before-and-after experiment during different release cycles, or between two different departments.

2. Additional methodological work is necessary to capture valid impact of the used analytical approaches in the domain. Potentially, this project could be extended by text-mining analysis of the issue descriptions and attempt to find correlation between the description and success rate of resolution. This way issues with high potential to be unresolved could be identified much earlier in the process, resulting in saved resources and costs.

3. It would be helpful to capture qualitative experiences with process mining and linguistic summaries of managerial level stakeholders. A series of focus groups could reveal what the methodologies provide and lack in the context of software defect-management. Although the basics of this research were covered in our study, a more valid extension is necessary to collect and validate information.

Bibliography

[1] Wil M.P. van der Aalst. "Process Mining: Discovery, Conformance and Enhancement of Business Processes". *Springer*, 3:352, 2011.

[2] Wil M.P. van der Aalst. "Process Mining Manifesto". *Business Process Management Workshops - Lectures Notes in Business Information Processing*, 99, 2012.

[3] Wil M.P. van der Aalst, H.A. Reijers, A.J.M.M Weijters, B.F. van Dongen, A.K. Alves de Medeiros, M. Song, and H.M.W. Verbeek. "Business Proess Mining: An industrial application". *ScienceDirect: Information Systems*, 32:713–732, 2007.

[4] Wil M.P. van der Aalst and A.J.M.M. Weijters. "Process mining: a research agenda". *Computers in Industry*, 53:231–244, 2004.

[5] Scott Barber. "Beyond performance testing". *IBM DeveloperWorks*, pages 1–18, 2004.

[6] Fatih Emre Boran, D. Akay, and R.R. Yager. "An overview of methods for linguistic summarization with fuzzy sets". *Expert Systems with Applications: An international Journal*, 61:356–377, 2016.

[7] Bernadette Bouchon-Meunier, M. Dotoli, and B. Maione. "On the choice of Membership Functions in a Mamdani-type Fuzzy Controller". 1996.

[8] Rita Castilla-Ortega, N. Marin, D. Sanchez, and A.G. Tettamanzi. "Linguistic Summarization of Time Series using Genetic Algorithms". *EUSLFAT Conference*, pages 1–8, 2011.

[9] Soumen Chakrabarti, M. Ester, U. Fayyad, J. Gehrke, J. Han, S. Morishita, G. Piatetsky-Shapiro, and W. Wang. "Data Mining Curriculum: A Proposal". *Intensive Working Group of ACM SIGKDD Curriculum Commmittee*, pages 23–44, 2006.

[10] Hsinchun Chen, R.H.L. Chiang, and C. Storey. "Business Intelligence and Analytics: From Big Data to Big Impact". *MIS Quarterly*, 36(4):1165–1188, 2012.

[11] Chen Chun-houh, W. Hrdle, and A. Unwin. "Handbook of Data Visualization". *Springer*, 2005.

[12] Christopher Clifton. "Data Mining". *Encyclopedia Britannica*, 2009.

[13] Jonathan E. Cook and A.L. Wolf. "Event-Based Detection of Concurrency". *Proceedings of the Sixth International Symposium of the Foundations of Software Engineering*, pages 35–45, 1998.

[14] Miguel Delgado, D. Snchez, and MA. Vila. "Fuzzy cardinality based evaluation of quantified sentences". *International Journal of Approximate Reasoning*, 23:23–66, 2000.

[15] Remco M. Dijkman and A.M. Wilbik. "Linguistic Summarziation of Event Logs - A Practical Approach". *Information Systems*, 67:114–125, 2017.

[16] Alan R. Hevner, S. March, J. Park, and S. Ram. "Design science in Information Systems Research". *MIS Quarterly*, 28(1):75–105, 2004.

[17] Kaoru Hirota and W. Pedrycz. "Fuzzy computing for data mining". *Proceedings of the IEEE*, 87(9):1575–1600, 1999.

[18] Peter J Huber. "Languages for Statistics and Data Analysis". *Journal of Computational and Graphical Statistics*, 9(3):600–620, 2000.

[19] Janusz Kacprzyk and P. Strykowski. "Linguistic summaries of sales data at a computer retailer - a case study". *Proceedings of IFSA*, 1(99):29–33, 1999.

[20] Janusz Kacprzyk, A. Wilbik, and S. Zadrozny. "Linguistic summarization of time series using a fuzzy quantifier driven aggregation". *Systems Research Institute: Polish Academy of Sciences*, 159:1–9, 2008.

[21] Janusz Kacprzyk and R.R. Yager. "Linguistic summaries of data using fuzzy logic". *Internat. J. General Systems*, 30:33–154, 2001.

[22] Janusz Kacprzyk, R.R. Yager, and S. Zadrozny. "Fuzzy linguistic summaries of databases for an efficient business data analysis and decision support". *Systems Research Institute: Polish Academy of Sciences, Machine Intelligence Insitute: Iona College*, 30:2–16, 2014.

[23] Janusz Kacprzyk and S. Zadrozny. "Supporting Decision Making via Verbalization of Data Analysis Results Using Linguistic Data Summaries". *Recent Advances in Decision Making*, pages 121–143, 2009.

[24] Daniel A. Keim. "Information Visualization and Visual Data Mining". *IEEE Transactions on Visualization and Computer Graphics*, 8(1):23–25, 2002.

[25] Andreas Kerren, J.T. Stasko, J. Fekete, and C. North. "Human-centered Issues and perspectives". *Springer* , 1998.

[26] Barbara Kitchenham. "Guidelines for performing Systematic Literature Reviews in Software Engineering". *Keele University, Dunham University*, 2:33–53, 2007.

[27] Anne Laurent. "A new approach for the generation of fuzzy summaries based on fuzzy multidimensional databases". *Intell. Data Analytics*, 7:155–177, 2003.

[28] Steve LaValle, E. Lesser, R. Shockley, M.S. Hopkins, and N. Kruschwitz. "Special Report: Analytics and the New Path to Value". *MIT Sloan Management Review*, 52:22–32, 2011.

[29] Jan De Leeuw and G. Michailidies. "Graph Layout Techniques and Multidimensional Data Analysis". *Institute of Mathematical Statistics*, 35:219–248, 2000.

[30] Gordon S. Linoff and M.J.A. Berry. "Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management". *Encyclopedia Britannica*, 3:60–100, 2011.

[31] Inderjeet Mani and M. Maybury. "Advances in Automatic Text Summarization". *MIT Press*, pages 233–244, 1999.

[32] Laura Maruster, A.J.M.M. Weijters, W.M.P. van der Aalst, and A. van den Bosch. "Process Mining: discovering direct successors in process logs". *Springer-Proceedings of the 5th International Conference on Discovery Science*, 2534:364–373, 2002.

[33] Tim Menzies. "Software Analytics: So what?". *IEEE Software*, pages 3–4, 2013.

[34] Florian Mlechert, R. Winter, and M. Klesse. "Aligning Process Automation and Business Intelligence to Support Corporate Performance Management". *Americas Conference on Information Systems (AMCIS)*, 36(4):1165–1188, 2012.

[35] Glenn J. Myatt and W.P. Johnson. "Making Sense of Data I: A Practical Guide to Exploratory Data Analysis and Data Mining". *John Wiley and Sons Inc.*, pages 13–30, 2014.

[36] Adam Niewiadomski. "A Type-2 Fuzzy Approach to Linguistic Summarization of Data". *IEEE Transactions on Fuzzy Systems*, 16(1):198–212, 2008.

[37] Ken Peffers, T. Tuunanen, M.A. Rothenberger, and S. Chatterjee. "A Design Science Research Methodology for Information Systems Research". *Journal of Management Information Systems*, 24(3):45–77, 2007.

[38] Wouter Poncin, A. Serebrenik, and M.v.d. Brand. "Process mining software repositories". *15th European Conference on Software Maintenance and Reengineering*, pages 5–14, 2011.

[39] William M. Pride and O.C. Ferrell. "Foundations of Marketing". *Barnes and Noble*, 6(1):36–40, 2007.

[40] Marcus E. Raichle. "A brief history of human functional brain mapping.". *Brain Mapping The Systems*, pages 33–75, 2000.

[41] Anne Rozinat. "Disco Tour". *Fluxicon*, pages 1–13, 2013.

[42] Connie U. Smith. "Software Performance Engineering". *Encyclopedia of Software Engineering*, 2002.

[43] Robert Spence. "Information Visualization". *Springer*, 2001.

[44] Mitschele A. Thiel and B. Muller-Clostermann. "Performance Engineering of SDL/measurements". *Journal on Computer Networks and ISDN Systems*, 31:1801–1815, 1999.

[45] Boudewijn F. van Dongen. "Process Mining and Verification". *Eindhoven University of Technology*, pages 6–8, 2007.

[46] Don Vilsack. "The Intersection of Big Data Analytics and Software Development: Why You Should Be There". *Recent Advances in Decision Making*, pages 121–143, 2009.

[47] G. Wedawatta, B. Ingirige, and D. Amaratunga. "Case Study as a Research Strategy: Investigating Extreme Weather Resilience of Construction SMEs in the UK". *Annual International Conference of International Institute for Infrastructure*, 7(1):1–9, 2011.

[48] A.J.M.M. Weijters and W.M.P. van der Aalst. "Process mining: Discovering Workflow Models from Event-Based Data". *Proceedings of the ECAI Workshop on Knowledge Discovery and Spatial Data*, pages 283–290, 2001.

[49] Elaine J. Weyuker and F.I. Vokolos. "Experience with Performance Testing of Software Systems: Issues, an Approach, and Case Study". *IEEE Trans. On software Engineering*, 26:1147–1156, 2000.

[50] Anna Wilbik and U. Kaymak. "Gradual linguistic summaries". *Information Processing and Management of Uncertainty in Knowledge-Based systems*, (15):2–3, 2014.

[51] Anna Wilbik and U. Kaymak. "Linguistic Summarization of Processes - a research agenda". *16th World Congress of the International Fuzzy Systems Association (IFSA), 9th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT)*, ():1636–1640, 2015.

[52] Anna M. Wilbik and R.M. Dijkman. "On the generation of useful linguistic summaries of sequences". *Information Systems - School of Industrial Engineering*, pages 555–562, 2016.

[53] Murray Woodside, G. Franks, and D.C. Petriu. "The Future of Software Performance Engineering. *2007 Future of Software Engineering*, 17:171–187, 2007.

[54] Dongrui Wu and M. Mendel. "Linguistic summarization using IF-THEN rules". *IEEE International Conference on Fuzzy Systems*, pages 2–8, 2010.

[55] Jing Xu and M. Woodside. "Performance Analysis of a Software Design using the UML Profile for Schedulability, Performance and Time". *International Conference Modeling Techniques and Tools for Computer Software Design using the UML Profile for Schedulability, Performance and Time*, 13, 2003.

[56] Ronald R. Yager. "A New Approach to the Summarization of Data". *Information Sciences*, 28:69–86, 1982.

[57] Ronald R. Yager. "On linguistic summaries of data". *Knowledge discovery in databases*, pages 347–363, 1991.

[58] Ronald R. Yager. "Linguistic summaries as a tool for database discovery". *Proceedings of FUZZ-IEEE'95/IFES'95, Workshop on Fuzzy Database Systems and Information Retrieval*, ():79–82, 1995.

[59] Ronald R. Yager. "Database discovery using fuzzy sets.". *International Journal of Intelligent Systems*, 11(9):691–712, 1996.

[60] Lotfi A. Zadeh. "Fuzzy Sets". *Information and Control*, pages 338–353, 1965.

[61] Lotfi A. Zadeh. "A note on prototype theory and fuzzy sets". *University of California*, 12, 1982.

Appendix A

Appendices

A.1 Questionnaires

A.1.1 Questionnaires for Managers

D1	Full name, Role/Position, Department
Q1	What is your role in the organization?
Q2	What is your team's role in the organization?
Q3	Do you use the WMS to track and manage the issue resolution process? [IF NO, why not?]
Q4	[IF YES] Have you ever submitted an issue through the WMS?
Q5	[IF YES] What steps do you usually take to ensure your submitted issue is resolved effectively and in time?
Q6	How informed are you about the progress of issue resolution? From 1-to-10, where 1 is "not informed at all" and 10 is "fully informed"
Q7	How satisfied are you with the in-built reporting tool in the WMS? From 1-to-10, where 1 is "not informed at all" and 10 is "fully informed"
Q8	What kind of insights would be important for you? (Influx-outflux, diversity of case handling, lead time, cycle time, team velocity etc.)
Q9	Would you rather see concise textual representations of such data or the traditional visualization approach (charts, graphs)?
Q10	Below are some sentences generated using linguistic summarization method. Please rank the following statements based on how true you perceive them from 1-to-10, where 1 is "completely not true" and 10 is "completely true".
a	Very few Problem Reports in premium which are Closed and are Critical are very long.
b	Many Change Requests in maintenance which are Closed and Minor are very long.
c	Most issues without project type information are very long.
d	Few major severity issues are short.
e	Few issues in transactions are very long
f	Most issues in interface which are rejected and of Minor severity are very long.
g	Most Change Requests in maintenance which are Rejected and are Minor are very long.
h	Many issues in field are short.
i	Many Change Requests in maintenance which are Closed and are Major are very long.
j	Most Change Requests which are Rejected and of Critical severity are short.
k	Many issues with no information on type are very long.

l	Very few issues are short.
m	Most Problem Reports are short.
n	Most Problem Reports which are Rejected and of Major severity are short.
o	Most Change Requests which are Closed and of Critical severity are short.
p	Many Rejected issues are short.
Q11	Judging by the summaries in tables 3 and 4, are there any surprising findings (interesting insights)?
Q12	Are there any impossible findings (wrong or very surprising findings) that you see in the statements?
Q13	Is there anything you would expect to see in the data summaries but you have not?
Q14	Additional comments/notes

A.1.2 Questionnaires for Developers/Testers

D1	Full name, Role/Position, Department/Team
Q1	What is your role in the organization?
Q2	Could you describe your duties in terms of steps and how much time each of them takes (on average)? (IN DAYS: Planning, Analysis, Design, Development, Testing, Release)
Q3	Do you use the WMS to track and manage the issue resolution process? [IF NO, why not?]
Q4	How informed are you about the progress of issue resolution? From 1-to-10, where 1 is "not informed at all" and 10 is "fully informed"
Q5	What is the main source of this information? (Team-leader, Sprint-meetings, WMS, Other)
Q6	Are you aware of WMS's in-built reporting/analytic tools?
Q7	Do you ever use WMS analytics tools to get additional information on issue resolution? [Yes, all the time - Yes, sometimes - Rarely - Never]
Q8	What kind of insights would be important for you? (Influx-outflux, diversity of case handling, throughput time etc.)
Q9	Would you rather see concise textual representations of such data or the traditional visualization approach (charts, graphs)?
Q10	Below are some sentences generated using linguistic summarization method. Please rank the following statements based on how true you perceive them from 1-to-10, where 1 is "completely not true" and 10 is "completely true".
a	Very few Problem Reports in premium which are Closed and are Critical are very long.
b	Many Change Requests in maintenance which are Closed and Minor are very long.
c	Most issues without project type information are very long.
d	Few major severity issues are short.
e	Few issues in transactions are very long
f	Most issues in interface which are rejected and of Minor severity are very long.
g	Most Change Requests in maintenance which are Rejected and are Minor are very long.
h	Many issues in field are short.
i	Many Change Requests in maintenance which are Closed and are Major are very long.
j	Most Change Requests which are Rejected and of Critical severity are short.
k	Many issues with no information on type are very long.
l	Very few issues are short.
m	Most Problem Reports are short.

n	Most Problem Reports which are Rejected and of Major severity are short.
o	Most Change Requests which are Closed and of Critical severity are short.
p	Many Rejected issues are short.
Q11	Judging by the summaries in tables 3 and 4, are there any surprising findings (interesting insights)?
Q12	Are there any impossible findings (wrong or very surprising findings) that you see in the statements?
Q13	Is there anything you would expect to see in the data summaries but you have not?
Q14	Additional comments/notes

A.2 Survey results - managers

A.2.1 Interview 1

- Q1. Management
- Q2. Deployment of software.
- Q3. Yes
- Q4. Yes
- Q5. Dispatch, prioritize and track
- Q6. 10
- Q7. 7
- Q8. Influx-outflux, lead-time
- Q9. Graph
- Q10.
  - 3.75 - Average perceived truth for sentences 0-to-0.25
  - 5.25 - Average perceived truth for sentences 0.25-to-0.5
  - 7 - Average perceived truth for sentences 0.5-to-0.75
  - 8 - Average perceived truth for sentences 0.75-to-1
- Q11. Interesting that issues with no information on their type are so long.
- Q12. No
- Q13. I am not familiar with the method, so I am not sure what else could be extracted.
- Q14. This approach is very interesting, reveals patterns which are very difficult to see otherwise.

A.2.2 Interview 2

- Q1. Management
- Q2.
- Q3. Yes
- Q4. Yes

- Q5. I receive the Change Requests that are difficult to resolve. The BIG items that are not resolved by others come to me. I communicate the information thoroughly with involved parties to ensure that the resolution process is carried to the end, or that the value created is justified.
- Q6. 1
- Q7. 5
- Q8. Influx-outflux, lead-time
- Q9. Influx-outflux, a potential link with source code
- Q10.
  - 6.5 - Average perceived truth for sentences 0-to-0.25
  - 6.25 - Average perceived truth for sentences 0.25-to-0.5
  - 7.5 - Average perceived truth for sentences 0.5-to-0.75
  - 7.5 - Average perceived truth for sentences 0.75-to-1
- Q11. Its an interesting approach to give the overall knowledge. Its interesting to see that there are only a few issues that are solved quickly.
- Q12. No
- Q13. Information on outflux-influx would be interesting to see if this method allows.
- Q14. Maybe focus on other data sources to enrich the summaries.

A.2.3 Interview 3

- Q1. Management
- Q2. Track and pursue long-resolutions with respective parties.
- Q3. Yes
- Q4. Yes
- Q5. Include as much detail as I can, so that if the resolving party needs data.
- Q6. 10
- Q7. 6
- Q8. Influx-outflux, info on time spent on issue, group issues based on features
- Q9. Graph
- Q10.
  - 6.5 - Average perceived truth for sentences 0-to-0.25
  - 5.75 - Average perceived truth for sentences 0.25-to-0.5
  - 6.25 - Average perceived truth for sentences 0.5-to-0.75
  - 7.75 - Average perceived truth for sentences 0.75-to-1
- Q11. That so many issues in maintenance take very long to close.
- Q12. I would disagree that Problem Reports take a short time (refers to low validity sentence).
- Q13. No.
- Q14. No.

A.2.4 Interview 4

- Q1. Management
- Q2.
- Q3. Yes
- Q4. Yes
- Q5. Keep list of submitted issues from our team, and directly contact the resolution party to keep track of progress.
- Q6. 5
- Q7. 5
- Q8. Issue types statistics, planned date and progress
- Q9. Graph
- Q10.
  - 4.5 - Average perceived truth for sentences 0-to-0.25
  - 5.5 - Average perceived truth for sentences 0.25-to-0.5
  - 7 - Average perceived truth for sentences 0.5-to-0.75
  - 7.75 - Average perceived truth for sentences 0.75-to-1
- Q11. They provide very detailed breakdown of issue behavior. It is interesting because you can then investigate these behaviors and fix them.
- Q12. Not that I can see.
- Q13. No.
- Q14. No.

A.2.5 Interview 5

- Q1. Management
- Q2. Review and lead the activities in software development, detail technical standards - coding, tools and improve overall software efficiency.
- Q3. Yes
- Q4. Yes
- Q5. Managing issue resolution, tracking problems and aiding in resolution.
- Q6. 7
- Q7. 6
- Q8. Influx-outflux, lead-time
- Q9. Graph
- Q10.
  - 6.25 - Average perceived truth for sentences 0-to-0.25
  - 7 - Average perceived truth for sentences 0.25-to-0.5

  - 5.5 - Average perceived truth for sentences 0.5-to-0.75
  - 8 - Average perceived truth for sentences 0.75-to-1

- Q11. That there are issues with no information on type. I would like to know if these are of recent cases, or previous years. Because the process is always improving and such events should not happen.
- Q12.
- Q13. I would expect that most issues fall into short resolution times. Apparently we take too much time to resolve issues, and some of them take very long which is not acceptable.
- Q14. This is a very interesting project. It would be very useful if it can be applied in other data.

A.2.6 Interview 6

- Q1. Management
- Q2. Managment of issues
- Q3. Yes
- Q4. Yes
- Q5.
- Q6.
- Q7.
- Q8. Influx-outflux, lead-time
- Q9. Graph
- Q10.
  - 4.25 - Average perceived truth for sentences 0-to-0.25
  - 4 - Average perceived truth for sentences 0.25-to-0.5
  - 5 - Average perceived truth for sentences 0.5-to-0.75
  - 8 - Average perceived truth for sentences 0.75-to-1
- Q11. A few of the attributes mentioned have mandatory fields, so it is very surprising that there are issues with no type, or project information.
- Q12. Again, mandatory fields data different from what I expected.
- Q13. Not at this moment, need time to consume the data and validate.
- Q14.

A.3 Survey results - software engineers/testers

A.3.1 Interview 1

- Q1. Software development
- Q2.
  - 1 - Planning
  - 2-3 - Analysis
  - 2-3 - Design
  - 5 - Development
  - 1 - Documentation
  - 2 - Testing
  - 2 - Release
- Q3. Yes
- Q4. 10
- Q5. Team-lead.
- Q6. No
- Q7. Never
- Q8. i.e. information missing, and sometimes deleted.
- Q9. Both graph and text.
- Q10.
  - 5 - Average perceived truth for sentences 0-to-0.25
  - 5.5 - Average perceived truth for sentences 0.25-to-0.5
  - 7.75 - Average perceived truth for sentences 0.5-to-0.75
  - 8 - Average perceived truth for sentences 0.75-to-1
- Q11. I am not familiar with the process details enough to provide an answer to this question.
- Q12. Don't know.
- Q13. Even though the concept is interesting, I feel that graphs are easier to understand. These summaries are rather complicated.
- Q14.

A.3.2 Interview 2

- Q1. Software development
- Q2.
  - 1 - Planning
  - 2-3 - Analysis
  - 2-3 - Design
  - 5 - Development

  - 1 - Documentation
  - 2 - Testing
  - 2 - Release
- Q3. Yes
- Q4. 10
- Q5. Team-lead, sprints.
- Q6. No
- Q7. Never
- Q8. Lead-time, details on issue.
- Q9. Both graph and text.
- Q10.
  - 6.5 - Average perceived truth for sentences 0-to-0.25
  - 6.25 - Average perceived truth for sentences 0.25-to-0.5
  - 7.25 - Average perceived truth for sentences 0.5-to-0.75
  - 7.5 - Average perceived truth for sentences 0.75-to-1
- Q11. Yes they are very interesting insights, very detailed.
- Q12. No.
- Q13. No.
- Q14.

A.3.3 Interview 3

- Q1. Software development
- Q2.
  - 1 - Planning
  - 3 - Analysis
  - 1 - Design
  - 3 - Development
  - 8 - Documentation
  - 1 - Testing
  - 3 - Release
- Q3. Yes
- Q4. 5
- Q5. WMS
- Q6. Yes
- Q7. Yes, sometimes.
- Q8. Current state of issue.

- Q9. Graph
- Q10.
  - 5 - Average perceived truth for sentences 0-to-0.25
  - 6.25 - Average perceived truth for sentences 0.25-to-0.5
  - 7.25 - Average perceived truth for sentences 0.5-to-0.75
  - 7.25 - Average perceived truth for sentences 0.75-to-1
- Q11. Yes, why are there issues that take so long? I'd like to know more about the reasons behind that.
- Q12. Not very well informed about the distribution among different teams. Regarding my issues, I expect some issues to take less time than what the summaries say.
- Q13. I would expect more information on the reasons why issues are rejected or closed.
- Q14. I like the approach and maybe more information could be revealed if you apply it in other data.

A.3.4 Interview 4

- Q1. Software development
- Q2.
  - 0 - Planning
  - 0 - Analysis
  - 2 - Design
  - 7 - Development
  - 1 - Documentation
  - 0 - Testing
  - 0 - Release
- Q3. No
- Q4. 3
- Q5. Email updates.
- Q6. No
- Q7. Never
- Q8. Time to resolution, current state of issue
- Q9. Both graph and text.
- Q10.
  - 5.75 - Average perceived truth for sentences 0-to-0.25
  - 6 - Average perceived truth for sentences 0.25-to-0.5
  - 5.75 - Average perceived truth for sentences 0.5-to-0.75
  - 6.25 - Average perceived truth for sentences 0.75-to-1

- Q11. The sentences definitely seem to provide some very detailed insights into projects and types of issues. These may be very useful in investigating and improving some aspects of our issue resolving.
- Q12. My job is not directly related to the process itself, so I can not say.
- Q13. No.
- Q14. There are issues that have not been solved for many years. Maybe that is also something to clear with the responsible submitters.

A.3.5 Interview 5

- Q1. Software development
- Q2.
  - 1 - Planning
  - 3 - Analysis
  - 2 - Design
  - 5 - Development
  - 3 - Documentation
  - 1 - Testing
  - 2 - Release
- Q3. No
- Q4. 1
- Q5. Team-lead, Sprint-meetings
- Q6. No
- Q7. Never
- Q8. Missing information, unable to re-create environment.
- Q9. Graph.
- Q10.
  - 6.25 - Average perceived truth for sentences 0-to-0.25
  - 5.75 - Average perceived truth for sentences 0.25-to-0.5
  - 5.5 - Average perceived truth for sentences 0.5-to-0.75
  - 7.5 - Average perceived truth for sentences 0.75-to-1
- Q11. We use another system.
- Q12. No
- Q13. Not well-informed about WMS.
- Q14.

A.4 Dataset linguistic summaries

A.4.1 Basic linguistic summaries

Due to large number of results, we only show summaries with T larger than 0.7.

	Summary Predicate	Summary Attribute	T
B1	very few are short	IssueType:(None)	1.0
B4	very few are very long	IssueType:(None)	1.0
B8	few are very long	IssueType:(None)	1.0
B12	many are very long	IssueType:(None)	1.0
IT17	very few are short	IssueType:Change Request	1.0
IT18	very few are average	IssueType:Change Request	1.0
IT19	very few are long	IssueType:Change Request	1.0
IT20	very few are very long	IssueType:Change Request	1.0
IT33	very few are short	IssueType:Problem Report	1.0
IT34	very few are average	IssueType:Problem Report	1.0
IT35	very few are long	IssueType:Problem Report	1.0
IT36	very few are very long	IssueType:Problem Report	1.0
TM49	very few are short	ProjectType: field	1.0
TM50	very few are average	ProjectType: field	1.0
TM51	very few are long	ProjectType: field	0.8
TM52	very few are very long	ProjectType: field	1.0
TM53	few are short	ProjectType: field	1.0
TM65	very few are short	ProjectType: maintenance	1.0
TM66	very few are average	ProjectType: maintenance	1.0
TM67	very few are long	ProjectType: maintenance	1.0
TM68	very few are very long	ProjectType: maintenance	1.0
TM81	very few are short	ProjectType: maintenance	1.0
TM82	very few are average	ProjectType: maintenance	1.0
TM83	very few are long	ProjectType: maintenance	1.0
TM84	very few are very long	ProjectType: maintenance	1.0
TM97	very few are short	ProjectType: premium	1.0
TM98	very few are average	ProjectType: premium	1.0
TM99	very few are long	ProjectType: premium	1.0
TM100	very few are very long	ProjectType: premium	1.0
TM113	very few are short	ProjectType: legacy	1.0
TM114	very few are average	ProjectType: legacy	1.0
TM115	very few are long	ProjectType: legacy	1.0
TM116	very few are very long	ProjectType: legacy	1.0
TM129	very few are short	ProjectType: delivery	1.0
TM130	very few are average	ProjectType: delivery	1.0
TM131	very few are long	ProjectType: delivery	1.0
TM132	very few are very long	ProjectType: delivery	1.0
TM145	very few are short	ProjectType: storage	1.0
TM146	very few are average	ProjectType: storage	1.0
TM147	very few are long	ProjectType: storage	1.0
TM148	very few are very long	ProjectType: storage	1.0
TM161	very few are short	ProjectType: linux	1.0
TM162	very few are average	ProjectType: linux	1.0
TM163	very few are long	ProjectType: linux	1.0
TM164	very few are very long	ProjectType: linux	1.0
TM177	very few are short	ProjectType: MS windows	1.0
TM178	very few are average	ProjectType: MS windows	1.0
TM179	very few are long	ProjectType: MS windows	1.0
TM180	very few are very long	ProjectType: MS windows	1.0
TM193	very few are short	ProjectType: transactions	1.0
TM194	very few are average	ProjectType: transactions	1.0

TM195	very few are long	ProjectType: transactions	1.0
TM196	very few are very long	ProjectType: transactions	1.0
TM209	very few are short	ProjectType: interface	1.0
TM210	very few are average	ProjectType: interface	1.0
TM211	very few are long	ProjectType: interface	1.0
TM212	very few are very long	ProjectType: interface	1.0
TM216	few are very long	ProjectType: interface	1.0
TM225	very few are short	ProjectType: cloud	1.0
TM227	very few are long	ProjectType: cloud	1.0
TM228	very few are very long	ProjectType: cloud	1.0
TM231	few are long	ProjectType: cloud	1.0
TM241	very few are short	ProjectType: nominal	1.0
TM242	very few are average	ProjectType: nominal	1.0
TM243	very few are long	ProjectType: nominal	1.0
TM244	very few are very long	ProjectType: nominal	1.0
ST257	very few are short	State: Closed	1.0
ST258	very few are average	State: Closed	1.0
ST259	very few are long	State: Closed	1.0
ST260	very few are very long	State: Closed	1.0
ST273	very few are short	State: Rejected	1.0
ST274	very few are average	State: Rejected	1.0
ST275	very few are long	State: Rejected	0.8
ST276	very few are very long	State: Rejected	1.0
ST277	few are short	State: Rejected	1.0
SV289	very few are short	Severity: Critical	1.0
SV290	very few are average	Severity: Critical	1.0
SV291	very few are long	Severity: Critical	1.0
SV292	very few are very long	Severity: Critical	1.0
SV293	few are short	Severity: Critical	0.8
SV305	very few are short	Severity: Major	1.0
SV306	very few are average	Severity: Major	1.0
SV307	very few are long	Severity: Major	1.0
SV308	very few are very long	Severity: Major	1.0
SV321	very few are short	Severity: Minor	1.0
SV322	very few are average	Severity: Minor	1.0
SV323	very few are long	Severity: Minor	1.0
SV324	very few are very long	Severity: Minor	1.0

A.4.2 Complex linguistic summaries

Due to large number of results, we only show summaries with T larger than 0.7 and with quantifiers few, many and most (not "very few").

	Summary Predicate	Summary Attribute	T
CM5	few are short	IssueType: Change Request and ProjectType: field and State: Closed and Severity: Critical	0.8
CM38	few are average	IssueType: Change Request and ProjectType: field and State: Closed and Severity: Minor	1.0
CM53	few are short	IssueType: Change Request and ProjectType: field and State: Rejected and Severity: Critical	1.0
CM57	many are short	IssueType: Change Request and ProjectType: field and State: Rejected and Severity: Critical	1.0
CM61	most are short	IssueType: Change Request and ProjectType: field and State: Rejected and Severity: Critical	1.0

CM69	few are short	IssueType: Change Request and ProjectType: field and State: Rejected and Severity: Major	1.0
CM73	many are short	IssueType: Change Request and ProjectType: field and State: Rejected and Severity: Major	1.0
CM85	few are short	IssueType: Change Request and ProjectType: field and State: Rejected and Severity: Minor	1.0
CM89	many are short	IssueType: Change Request and ProjectType: field and State: Rejected and Severity: Minor	1.0
CM103	few are long	IssueType: Change Request and ProjectType: maintenance and State: Closed and Severity: Critical	0.7
CM120	few are very long	IssueType: Change Request and ProjectType: maintenance and State: Closed and Severity: Major	0.9
CM136	few are very long	IssueType: Change Request and ProjectType: maintenance and State: Closed and Severity: Minor	1.0
CM149	few are short	IssueType: Change Request and ProjectType: maintenance and State: Rejected and Severity: Major	0.8
CM152	few are very long	IssueType: Change Request and ProjectType: maintenance and State: Rejected and Severity: Major	0.8
CM168	few are very long	IssueType: Change Request and ProjectType: maintenance and State: Rejected and Severity: Minor	1.0
CM172	many are very long	IssueType: Change Request and ProjectType: maintenance and State: Rejected and Severity: Minor	1.0
CM183	few are long	IssueType: Change Request and ProjectType: maintenance and State: Closed and Severity: Major	1.0
CM187	many are long	IssueType: Change Request and ProjectType: maintenance and State: Closed and Severity: Major	1.0
CM198	few are average	IssueType: Change Request and ProjectType: maintenance and State: Rejected and Severity: Major	1.0
CM202	many are average	IssueType: Change Request and ProjectType: maintenance and State: Rejected and Severity: Major	0.8
CM230	few are average	IssueType: Change Request and ProjectType: premium and State: Closed and Severity: Minor	1.0
CM261	few are short	IssueType: Change Request and ProjectType: premium and State: Rejected and Severity: Minor	1.0
CM312	few are very long	IssueType: Change Request and ProjectType: legacy and State: Closed and Severity: Minor	1.0
CM328	few are very long	IssueType: Change Request and ProjectType: legacy and State: Rejected and Severity: Major	1.0
CM344	few are very long	IssueType: Change Request and ProjectType: delivery and State: Closed and Severity: (None)	1.0
CM348	many are very long	IssueType: Change Request and ProjectType: delivery and State: Closed and Severity: (None)	1.0
CM352	most are very long	IssueType: Change Request and ProjectType: delivery and State: Closed and Severity: (None)	1.0
CM360	few are very long	IssueType: Change Request and ProjectType: delivery and State: Closed and Severity: Critical	1.0
CM391	few are long	IssueType: Change Request and ProjectType: delivery and State: Closed and Severity: Minor	1.0
CM424	few are very long	IssueType: Change Request and ProjectType: delivery and State: Rejected and Severity: Minor	1.0
CM428	many are very long	IssueType: Change Request and ProjectType: delivery and State: Rejected and Severity: Minor	0.8
CM485	few are short	IssueType: Change Request and ProjectType: storage and State: Rejected and Severity: Major	1.0

CM501	few are short	IssueType: Change Request and ProjectType: linux and State: Closed and Severity: Major	1.0
CM502	few are average	IssueType: Change Request and ProjectType: linux and State: Closed and Severity: Major	1.0
CM520	few are very long	IssueType: Change Request and ProjectType: linux and State: Closed and Severity: Minor	1.0
CM524	many are very long	IssueType: Change Request and ProjectType: linux and State: Closed and Severity: Minor	0.8
CM582	few are average	IssueType: Change Request and ProjectType: MS windows and State: Rejected and Severity: Critical	1.0
CM600	few are very long	IssueType: Change Request and ProjectType: MS windows and State: Rejected and Severity: Major	1.0
CM604	many are very long	IssueType: Change Request and ProjectType: MS windows and State: Rejected and Severity: Major	0.8
CM616	few are very long	IssueType: Change Request and ProjectType: MS windows and State: Rejected and Severity: Minor	1.0
CM620	many are very long	IssueType: Change Request and ProjectType: MS windows and State: Rejected and Severity: Minor	1.0
CM629	few are short	IssueType: Change Request and ProjectType: nominal and State: Closed and Severity: Critical	1.0
CM661	few are short	IssueType: Change Request and ProjectType: nominal and State: Closed and Severity: Minor	1.0
CM678	few are average	IssueType: Change Request and ProjectType: nominal and State: Rejected and Severity: Major	1.0
CM693	few are short	IssueType: Problem Report and ProjectType: field and State: Closed and Severity: Critical	0.8
CM741	few are short	IssueType: Problem Report and ProjectType: field and State: Rejected and Severity: Critical	1.0
CM745	many are short	IssueType: Problem Report and ProjectType: field and State: Rejected and Severity: Critical	1.0
CM757	few are short	IssueType: Problem Report and ProjectType: field and State: Rejected and Severity: Major	1.0
CM761	many are short	IssueType: Problem Report and ProjectType: field and State: Rejected and Severity: Major	1.0
CM773	few are short	IssueType: Problem Report and ProjectType: field and State: Rejected and Severity: Minor	1.0
CM777	many are short	IssueType: Problem Report and ProjectType: field and State: Rejected and Severity: Minor	1.0
CM837	few are short	IssueType: Problem Report and ProjectType: maintenance and State: Rejected and Severity: Critical	1.0
CM872	few are very long	IssueType: Problem Report and ProjectType: maintenance and State: Rejected and Severity: Minor	1.0
CM886	few are average	IssueType: Problem Report and ProjectType: maintenance and State: Closed and Severity: Critical	0.8
CM902	few are average	IssueType: Problem Report and ProjectType: maintenance and State: Closed and Severity: Major	1.0
CM933	few are short	IssueType: Problem Report and ProjectType: maintenance and State: Rejected and Severity: Critical	1.0
CM937	many are short	IssueType: Problem Report and ProjectType: maintenance and State: Rejected and Severity: Critical	1.0
CM949	few are short	IssueType: Problem Report and ProjectType: maintenance and State: Rejected and Severity: Major	1.0
CM965	few are short	IssueType: Problem Report and ProjectType: maintenance and State: Rejected and Severity: Minor	1.0

CM969	many are short	IssueType: Problem Report and ProjectType: maintenance and State: Rejected and Severity: Minor	1.0
CM981	few are short	IssueType: Problem Report and ProjectType: premium and State: Closed and Severity: Critical	1.0
CM998	few are average	IssueType: Problem Report and ProjectType: premium and State: Closed and Severity: Major	0.8
CM1015	few are long	IssueType: Problem Report and ProjectType: premium and State: Closed and Severity: Minor	1.0
CM1045	few are short	IssueType: Problem Report and ProjectType: premium and State: Rejected and Severity: Major	1.0
CM1063	few are long	IssueType: Problem Report and ProjectType: premium and State: Rejected and Severity: Minor	0.7
CM1125	few are short	IssueType: Problem Report and ProjectType: legacy and State: Rejected and Severity: Critical	1.0
CM1221	few are short	IssueType: Problem Report and ProjectType: delivery and State: Rejected and Severity: Critical	1.0
CM1237	few are short	IssueType: Problem Report and ProjectType: delivery and State: Rejected and Severity: Major	1.0
CM1253	few are short	IssueType: Problem Report and ProjectType: delivery and State: Rejected and Severity: Minor	0.8
CM1318	few are average	IssueType: Problem Report and ProjectType: storage and State: Rejected and Severity: Critical	0.8
CM1352	few are very long	IssueType: Problem Report and ProjectType: storage and State: Rejected and Severity: Minor	1.0
CM1356	many are very long	IssueType: Problem Report and ProjectType: storage and State: Rejected and Severity: Minor	0.8
CM1381	few are short	IssueType: Problem Report and ProjectType: linux and State: Rejected and Severity: Critical	1.0
CM1414	few are average	IssueType: Problem Report and ProjectType: MS windows and State: Closed and Severity: Critical	0.7
CM1461	few are short	IssueType: Problem Report and ProjectType: MS windows and State: Rejected and Severity: Critical	1.0
CM1509	few are short	IssueType: Problem Report and ProjectType: transactions and State: Closed and Severity: Critical	1.0
CM1526	few are average	IssueType: Problem Report and ProjectType: transactions and State: Closed and Severity: Major	1.0
CM1544	few are very long	IssueType: Problem Report and ProjectType: transactions and State: Closed and Severity: Minor	1.0
CM1548	many are very long	IssueType: Problem Report and ProjectType: transactions and State: Closed and Severity: Minor	0.8
CM1559	few are long	IssueType: Problem Report and ProjectType: transactions and State: Rejected and Severity: Critical	1.0
CM1576	few are very long	IssueType: Problem Report and ProjectType: transactions and State: Rejected and Severity: Major	0.9
CM1592	few are very long	IssueType: Problem Report and ProjectType: transactions and State: Rejected and Severity: Minor	1.0
CM1596	many are very long	IssueType: Problem Report and ProjectType: transactions and State: Rejected and Severity: Minor	1.0
CM1600	most are very long	IssueType: Problem Report and ProjectType: transactions and State: Rejected and Severity: Minor	1.0
CM1640	few are very long	IssueType: Problem Report and ProjectType: interface and State: Closed and Severity: Minor	1.0
CM1644	many are very long	IssueType: Problem Report and ProjectType: interface and State: Closed and Severity: Minor	1.0

CM1656	few are very long	IssueType: Problem Report and ProjectType: interface and State: Rejected and Severity: Critical	1.0
CM1660	many are very long	IssueType: Problem Report and ProjectType: interface and State: Rejected and Severity: Critical	1.0
CM1672	few are very long	IssueType: Problem Report and ProjectType: interface and State: Rejected and Severity: Major	1.0
CM1688	few are very long	IssueType: Problem Report and ProjectType: interface and State: Rejected and Severity: Minor	1.0
CM1692	many are very long	IssueType: Problem Report and ProjectType: interface and State: Rejected and Severity: Minor	1.0
CM1717	few are short	IssueType: Problem Report and ProjectType: nominal and State: Closed and Severity: Major	1.0
CM1736	few are very long	IssueType: Problem Report and ProjectType: nominal and State: Closed and Severity: Minor	0.8
CM1749	few are short	IssueType: Problem Report and ProjectType: nominal and State: Rejected and Severity: Critical	1.0

A.5 All event log sequences

Freq.	Sequence
1237	Initiated Registered Investigated Assigned Fixed Verified Closed
449	Initiated Rejected
380	Initiated Registered Initiated Rejected
329	Initiated Registered Investigated Rejected
224	Initiated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
121	Initiated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed
117	Initiated Assigned Fixed Verified Closed
60	Initiated Registered Investigated Registered Initiated Rejected
58	Initiated Registered Investigated Registered Investigated Rejected
43	Initiated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
40	Initiated Registered Initiated Registered Initiated Rejected
40	Initiated Registered Initiated Registered Investigated Assigned Fixed Verified Closed
31	Initiated Registered Investigated Assigned Investigated Rejected
27	Initiated Fixed Verified Closed
23	Initiated Registered Investigated Assigned Fixed Assigned Fixed Assigned Fixed Verified Closed
23	Initiated Registered Investigated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed
18	Initiated Registered Initiated Registered Investigated Rejected
15	Initiated Registered Investigated Assigned Fixed Verified Assigned Fixed Verified Closed
8	Initiated Registered Investigated Registered Investigated Registered Initiated Rejected
8	Initiated Registered Initiated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
8	Initiated Registered Investigated Registered Initiated Registered Investigated Assigned Fixed Verified Closed
8	Initiated Registered Investigated Registered Investigated Registered Investigated Rejected
8	Initiated Registered Investigated Assigned Investigated Registered Investigated Assigned Fixed Verified Closed
6	Initiated Registered Investigated Registered Investigated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Verified Closed

6	Initiated Registered Investigated Registered Investigated Assigned Fixed Assigned Fixed Assigned Fixed Verified Closed
5	Initiated Registered Initiated Fixed Verified Closed
5	Initiated Registered Initiated Rejected Initiated Rejected
5	Initiated Registered Investigated Rejected Initiated Registered Investigated Assigned Fixed Verified Closed
5	Initiated Registered Investigated Registered Initiated Registered Initiated Rejected
5	Initiated Rejected Initiated Rejected
4	Initiated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed
4	Initiated Registered Investigated Registered Initiated Registered Investigated Rejected
4	Initiated Registered Investigated Assigned Investigated Registered Investigated Rejected
4	Initiated Registered Investigated Assigned Fixed Assigned Investigated Assigned Fixed Verified Closed
4	Initiated Registered Investigated Registered Investigated Registered Initiated Registered Initiated Rejected
4	Initiated Registered Investigated Assigned Fixed Assigned Investigated Rejected
4	Initiated Registered Investigated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
4	Initiated Registered Investigated Rejected Initiated Rejected
4	Initiated Assigned Fixed Assigned Fixed Verified Closed
3	Initiated Registered Investigated Assigned Investigated Assigned Fixed Verified Closed
3	Initiated Registered Investigated Registered Investigated Assigned Fixed Verified Assigned Fixed Verified Closed
3	Initiated Registered Initiated Registered Investigated Registered Initiated Rejected
3	Initiated Registered Investigated Assigned Investigated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
3	Initiated Registered Initiated Assigned Fixed Verified Closed
3	Initiated Registered Initiated Registered Investigated Registered Investigated Rejected
3	Initiated Registered Investigated Assigned Investigated Registered Initiated Rejected
3	Initiated Registered Investigated Registered Initiated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
3	Initiated Registered Investigated Assigned Fixed Assigned Fixed Assigned Fixed Assigned Fixed Verified Closed
3	Initiated Assigned Investigated Rejected
3	Initiated Registered Initiated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed
2	Initiated Registered Investigated Registered Investigated Assigned Investigated Registered Investigated Assigned Fixed Verified Closed
2	Initiated Registered Investigated Registered Investigated Rejected Initiated Rejected
2	Initiated Registered Initiated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
2	Initiated Registered Initiated Rejected Initiated Assigned Fixed Verified Closed
2	Initiated Rejected Initiated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
2	Initiated Registered Initiated Rejected Initiated Registered Investigated Assigned Fixed Verified Closed
2	Initiated Registered Investigated Registered Investigated Assigned Investigated Rejected
2	Initiated Registered Investigated Assigned Investigated Registered Investigated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed
2	Initiated Registered Initiated Registered Initiated Registered Initiated Registered Investigated Rejected
2	Initiated Registered Investigated Registered Initiated Fixed Verified Closed

2	Initiated Registered Initiated Registered Initiated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
2	Initiated Registered Investigated Registered Investigated Assigned Fixed Assigned Investigated Registered Investigated Assigned Fixed Verified Closed
2	Initiated Registered Investigated Registered Investigated Assigned Investigated Registered Initiated Rejected
2	Initiated Registered Investigated Registered Initiated Rejected Initiated Rejected
1	Initiated Registered Investigated Assigned Fixed Assigned Fixed Assigned Investigated Rejected
1	Initiated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Assigned Investigated Rejected
1	Initiated Registered Investigated Registered Investigated Assigned Investigated Registered Investigated Registered Initiated Rejected
1	Initiated Registered Investigated Assigned Investigated Registered Initiated Registered Investigated Rejected
1	Initiated Registered Investigated Registered Investigated Registered Investigated Registered Investigated Rejected
1	Initiated Registered Initiated Rejected Initiated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Registered Investigated Assigned Investigated Assigned Investigated Registered Initiated Rejected
1	Initiated Registered Investigated Assigned Investigated Registered Investigated Assigned Fixed Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Investigated Registered Initiated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Assigned Investigated Registered Investigated Assigned Fixed Assigned Investigated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Assigned Fixed Assigned Fixed Assigned Investigated Registered Investigated Rejected
1	Initiated Registered Investigated Assigned Fixed Assigned Investigated Registered Initiated Fixed Verified Closed
1	Initiated Registered Initiated Registered Initiated Registered Investigated Registered Initiated Rejected
1	Initiated Registered Investigated Assigned Investigated Registered Investigated Assigned Investigated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Initiated Rejected Initiated Registered Initiated Rejected
1	Initiated Registered Investigated Assigned Investigated Registered Investigated Registered Initiated Rejected
1	Initiated Fixed Assigned Fixed Verified Closed
1	Initiated Registered Investigated Assigned Fixed Assigned Investigated Registered Initiated Registered Investigated Assigned Fixed Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Registered Investigated Assigned Fixed Verified Assigned Investigated Rejected
1	Initiated Registered Initiated Rejected Initiated Registered Investigated Registered Initiated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Registered Investigated Assigned Investigated Registered Investigated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Registered Initiated Registered Investigated Registered Investigated Registered Investigated Assigned Investigated Rejected
1	Initiated Registered Initiated Rejected Initiated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Initiated Registered Investigated Assigned Investigated Rejected

1	Initiated Registered Initiated Fixed Assigned Investigated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Initiated Registered Initiated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Initiated Fixed Assigned Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Rejected Initiated Registered Initiated Rejected
1	Initiated Registered Investigated Assigned Fixed Assigned Investigated Registered Initiated Rejected
1	Initiated Registered Investigated Rejected Initiated Registered Investigated Rejected
1	Initiated Rejected Initiated Registered Investigated Rejected
1	Initiated Rejected Initiated Registered Initiated Rejected
1	Initiated Rejected Initiated Rejected Initiated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Investigated Assigned Investigated Assigned Investigated Rejected
1	Initiated Assigned Fixed Verified Assigned Fixed Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Assigned Fixed Verified Assigned Fixed Verified Closed
1	Initiated Assigned Investigated Registered Initiated Rejected
1	Initiated Registered Investigated Registered Investigated Assigned Investigated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Initiated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Initiated Registered Investigated Assigned Fixed Assigned Investigated Rejected
1	Initiated Registered Initiated Registered Initiated Registered Initiated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Verified Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Initiated Registered Investigated Registered Initiated Fixed Verified Closed
1	Initiated Fixed Assigned Investigated Rejected
1	Initiated Registered Initiated Registered Investigated Assigned Fixed Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Verified Assigned Investigated Rejected
1	Initiated Registered Investigated Registered Initiated Registered Investigated Registered Investigated Assigned Investigated Rejected
1	Initiated Registered Investigated Registered Investigated Registered Investigated Assigned Investigated Rejected
1	Initiated Registered Initiated Rejected Initiated Rejected Initiated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Investigated Registered Investigated Registered Investigated Assigned Investigated Rejected
1	Initiated Registered Initiated Registered Initiated Fixed Assigned Investigated Rejected
1	Initiated Registered Investigated Registered Investigated Registered Investigated Registered Investigated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Initiated Rejected Initiated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Initiated Registered Initiated Registered Initiated Rejected
1	Initiated Registered Investigated Registered Initiated Rejected Initiated Registered Investigated Rejected

1	Initiated Registered Investigated Assigned Investigated Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Fixed Verified Assigned Investigated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Rejected Initiated Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Assigned Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Rejected Initiated Registered Investigated Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Registered Initiated Registered Initiated Fixed Verified Closed
1	Initiated Registered Investigated Rejected Initiated Registered Investigated Registered Investigated Rejected
1	Initiated Registered Investigated Assigned Fixed Assigned Investigated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Assigned Fixed Assigned Fixed Verified Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Investigated Assigned Fixed Verified Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Rejected Initiated Assigned Investigated Rejected
1	Initiated Registered Investigated Assigned Fixed Assigned Fixed Assigned Fixed Assigned Fixed Assigned Fixed Verified Closed
1	Initiated Rejected Initiated Registered Investigated Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Investigated Assigned Fixed Verified Assigned Investigated Registered Initiated Fixed Verified Closed
1	Initiated Registered Investigated Assigned Investigated Registered Initiated Registered Initiated Registered Initiated Rejected
1	Initiated Registered Initiated Registered Initiated Registered Initiated Registered Initiated Registered Initiated Rejected
1	Initiated Registered Investigated Registered Initiated Registered Initiated Registered Initiated Rejected
1	Initiated Registered Investigated Registered Initiated Registered Initiated Registered Investigated Assigned Fixed Verified Assigned Fixed Verified Closed
1	Initiated Registered Investigated Registered Initiated Registered Initiated Registered Investigated Assigned Fixed Verified Closed

A.6 Literature Study Methodology

The systematic literature review (SLR) guidelines by Kitchenham [26] are used to obtain and present available literature relevant to the research area. The aim is to present a substantial series of suitable data summarization approaches that help generate insights from data. The methods of focus for these study are visual summarization, process mining and linguistic summarization methodologies, trends and practices. All relevant material is collected with respect to the proposed research question and area of interest. A series of quality control measures are taken while accessing and collecting the relevant literature.

The following section describes the undertaken steps of the review process. Firstly, the delineation of the research topic is evaluated to ensure broadness of topic and originality. Next, the literature document assessment and collection process is described. The relevant pool of studies is introduced, analyzed and grouped to describe common approaches in data visualization, process mining and linguistic summarization. Lastly, the key concepts of the area are summarized along with the research gap, followed by a closing chapter to show future potential in business application.

In the preliminary stage of the research, a set of keywords was identified to ensure that successful search results could be obtained. These keywords were motivated by the current terminology in

popular literature on data analysis, process mining and linguistic summarization. Though the field of interest is broad, the practical business application focus of the study requires content-specific search strings, and advanced searches in specific credible domains. As per the research questions and area of interest, the following keyword queries were used to generate search content:

- (Data) AND (Analysis OR Mining OR Linguistic Summarization)
- (Information) AND (Analysis OR Mining OR Linguistic Summarization)
- (Process) AND (Analysis OR Mining OR Linguistic Summarization)

The search process is aimed at collecting credible information during two main stages. The first one is a broad-scope exploration process followed by a topic-specific examination applied in the following online glossaries:

A.6.1 Broad-scope exploration

- Scientific literature: digital access to TU/e library, full-text articles, electronic books and journals.
- JSTOR: digital library for more than 2,000 academic journals, full-text articles.
- Google Scholar: peer-reviewed online academic journals, books, and other scholarly literature.
- Scopus: abstracts of journal articles from 15,000 peer-reviewed journals from 4,000 publishers.

A.6.2 Topic-specific investigation

- CiteSeerX: scientific literature digital library with focus on computer and information systems.
- IEEEXplore: highly cited publications in electrical engineering, computer science and electronics.
- ScienceDirect: large database of scientific and medical research with over 12 million documents.
- ACM Digital Library: academic and scholarly knowledge in computer science.

A.6.3 Study selection

During the first stage — broad scope exploration — numerous search results were produced which did not fully pertain to the characteristics of the research area. Therefore, a primary skimming approach was used to ensure that relevant articles elaborate on the area of interest, followed by a thorough reading of the selected articles for analysis. Afterwards, during topic-specific investigation, concrete material was extracted locally and analyzed in detail.

From the analyzed documents in this stage, many entries were removed due to inconsistency with research area, or different study scope. Lastly, as per the SLR schema [26] articles were filtered using inclusion/exclusion criteria.

During the initial delineation stage, the area of research and criteria of content in data visualization, process mining and linguistic summarization was aligned with stakeholders in the organization and academia. Criteria were refined using SLR schema [26] and were also actively revised based on the study field content and research focus.