Research Proposal

Mining software development projects

Author:
Saimir Bala, M.Sc., h1454574
Darwingasse 9/7, 1020 Vienna, saimir.bala@wu.ac.at

Supervisor:
Prof. Dr. Jan Mendling, jan.mendling@wu.ac.at

Date of Submission:

Institute for Information Business, Department of Information Systems and Operations
Vienna University of Economics and Business, Welthandelsplatz 1, 1020 Vienna, Austria

DEPARTMENT FÜR INFORMATIONS-VERARBEITUNG UND PROZESS-MANAGEMENT DEPARTMENT OF INFORMATION SYSTEMS AND OPERATIONS

Abstract

This dissertation is about mining the software development process using data generated from software project endeavors.

Contents

1 Introduction

Monitoring the process of software development is critical for software project managers. In practice, various project management approaches exist that guide the managers during the development of software projects. Usually these guidelines and methodologies stem from experience. However, in practical scenarios, every software development project is rarely carried out as planned. Therefore, empirical evidence given from the historical evolution of generated artifacts may provide important cues for checking the adherence of work to existing projects plans and milestones.

Data science methods are a starting point for empirically addressing the problem of monitoring software development processes. Specifically, the solution in confined in the intersection of three scientific fields *i)* Process Mining (PM); *ii)* Text Mining (TM); and *iii)* Mining Software Repositories (MSR). PM is an extensively studied field within the Business Process Management (BPM) literature and allows for the discovery and conformance checking of processes from structured data logs [vdA16]. TM uses offers a plethora of algorithms for extracting information from text [AZ12]. Text mining has only recently been applied to understand business processes [FMP11, Leo13, **?**] from natural language text. MSR focuses specifically on data from software development and aims at uncovering interesting and actionable information about software systems and projects from a software engineering perspective [PK16]. Despite clear anchor points among these three disciplines when it comes to gaining transparency into the process of software development, only a few works [**?**, **?**] explicitly apply mature process mining techniques to analyze the software development data. Nevertheless, software project logs cannot readily be consumed by process mining techniques. Typically, preprocessing and text mining techniques are required to extract structured logs from software repositories. Due to this fact, process mining contributions in the BPM field have left behind these particular types of processes. Therefore, process mining techniques fall short in software projects.

With this dissertation, I aim at extending the process mining field by extending its applicability to software development. My research is guided by the question "*How can we adapt process mining to gather insight from the software development process that are informative to managers?*". To this end, I plan to extend the field of process mining to work with unstructured data that are not generated from a business process engine, e.g. logs from Version Control System (VCS) and Issue Tracking System (ITS). This includes transforming real data taken from the railway domain or from other real project into structured logs and further analyze them. A first contribution has been done towards a visual aid for managers that allows them to visualize the project history as a Gantt chart in order to check for anomalies of the amount of work that has been done on the different work-packages of the project.

The rest of this research proposal is organized as follows. Section 2 provides background knowledge on the field of process mining, text mining, and mining software repositories. Section 3 identifies the problem in context of existing literature, derives the research question and breaks it down into four requirements. Section 4 explains the research method, the dataset and preliminary results in addressing the research requirements. Section 5 presents the expected

results and future steps towards the completion of this dissertation. Section 6 draws the implications for research and presents a dissemination plan.

2 Background

What is the crucial literature about the topic? What are the open research questions? How does the re- search hypothesis relate to the existing literature on the topic? Are there other dissertations that cover the same topic?

2.1 Process Mining

Process mining is a discipline that emerged in the last decade. The goal of this discipline is to provide fact-based insights and support process improvement. On a broader context, process mining can be considered as the missing link between traditional model-based process analysis and data driven techniques such as data mining and machine learning [vdA16]. Compared to existing Business Intelligence (BI) technologies, process mining techniques go beyond the calculation of simple Key Performance Indicators (KPIs). Instead, by building on model-driven approaches, they provide means to gain transparency on several aspects of the end-to-end business process. More specifically process mining techniques can infer models from event logs, which inform about the diverse aspects of a business process. Main aspects or *perspectives* of a business process are the following.

- The *time perspective* aims at analyzing time and frequency of process events. The focus is to use time information for performance analysis, such as bottlenecks, resource utilization, prediction of workload, remaining running time of process instances, etc.
- The *case perspective* aims at identifying properties of process cases. A process case is one execution of a process from start to end, e.g. from the moment a customer order a product to its delivery. However, there can be other ways to determine cases, e.g., the evolution of a data artifact, such a for instance the fulfillment of monthly report of the orders.
- The *organizational perspective* aims at analyzing the event log to gain transparency on the resources involved in the process. The focus of this perspective is to help understanding which are the people and systems in the process and how are they related in terms of roles, hierarchy, handover of work, privileges, access-control, social network, etc.
- The *control-flow perspective* aims at analyzing the different variations of the process, i.e., in which order its constituting activities are carried out in real life. The focus of this perspective is on understanding the different process variants by the help of process models expressed as Petri nets, Business Process Model and Notation (BPMN), event-driven process chain (EPC), Unified Modeling Language (UML), etc.

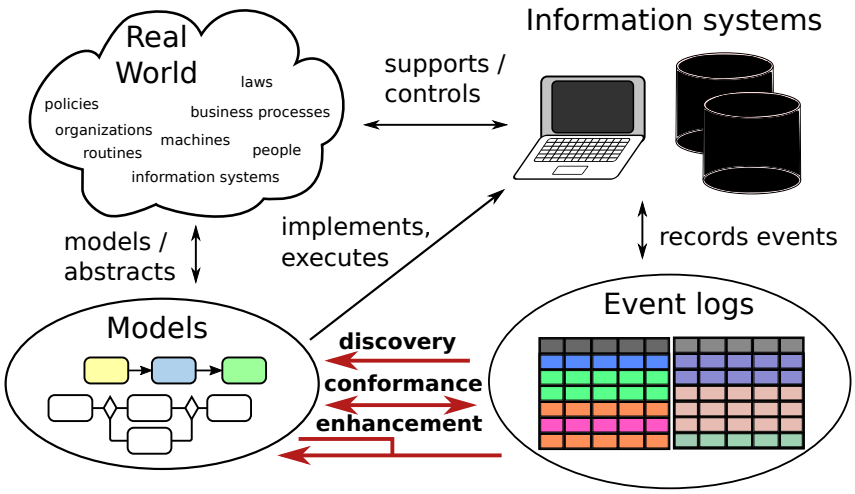

Figure 1: Process mining and data science. Adapted from [vdA16].

Figure 1 illustrates process mining research and how it relates event data from real world and business process models. There are three types of process mining, namely *i)* process discovery; *ii)* conformance checking; and *iii)* enhancement.

Process discovery. This type of process mining is concerned with the inference of process models from event logs. Process discovery algorithms are typically unsupervised techniques that produce models in various notations such as Petri nets, BPMN, EPC, etc. A example of a process discovery algorithm is the so-called α-algorithm [vdAWM04].

Conformance checking This type of process mining compares the model and the log of the same process. The goal is to verify if the reality as recorded in the event log corresponds to the plan as defined by the model. Possible deviations are quantified making it possible to obtain important cues about bad performance in case the model is prescriptive or noncompliance in case the model is normative.

Enhancement This type for process mining is focused on improving the existing process by using information from past executions recorded in the log. Differently from conformance, the goal is to go beyond measuring the misalignment but actually correcting the process model. Main types of corrections are *repair*, where a process model is repaired to better explain the data from the log, and *extension*, where a new information is added to the process model that is found in the log, e.g., labeling activities with the resource names, labeling sequence flows with durations, and so on.

Process mining is a mature field with considerable number of algorithms from academia and

a many industry tools such as Celonis[1], Disco[2], minit[3], LANA Process Mining[4] and many more. Mining algorithms are developed every year to deal to mine not only process workflow, but also other perspectives, i.e. organizational, data, etc. Nevertheless, process mining algorithms work with structured data [vDdMV+05, **?**]. In my research, I plan to draw from relevant techniques and algorithms from process mining and use the for analyzing data from VCSs. The challenge is that VCS log data is semi-structured and not always can be related to the standard XES [**?**] required by process mining algorithms.

2.2 Text Mining

Text mining refers to the process of deriving information from natural language text. It relates to data mining in that both strive to extract meaningful information from raw data. However, data mining is characterized as the extraction of implicit, previously unknown, and potentially useful information from data, whereas with text mining the information to be extracted is clearly and explicitly stated in the text [WBMT99]. Text mining can be regarded as going beyond information access to further help users analyze and digest information and facilitate decision making. There are also many applications of text mining where the primary goal is to analyze and discover any interesting patterns, including trends and outliers, in text data, and the notion of a query is not essential or even relevant. Although text mining is mostly about Natural Language Processing (NLP) [JM14], it embraces also applications that go beyond. For example, it analyzes linkage structures such as the citations in the academic literature and hyperlinks in the Web literature, both useful sources of information that lie outside the traditional domain of NLP. The main types of text main are the following.

Information extraction. Information extraction is used to refer to the task of filling in templates from natural language text. The goal is to extract from the documents (which may be in a variety of languages) salient facts about prespecified types of events, entities or relationships. These facts are then usually entered automatically into a database, which may then be used to analyze the data for trends, to give a natural language summary, or simply to serve for on-line access. Traditional information extraction techniques [CL96, Moo99] leverage on rule-based systems that match predefined linguistic patterns. More recently, work on named entity recognition uses statistical machine learning methods [SMR99]. A tool that uses unsupervised learning can be found in [BCS+07].

Topic detection and tracking. Topic Detection and Tracking (TDT) was a DARPA-sponsored initiative to investigate on finding and following new events in a stream of broadcast news stories. The TDT problem consists of three major tasks: (1) *segmenting* a stream of data, especially recognized speech, into distinct stories; (2) *identifying* those news stories that are

[1] https://www.celonis.com
[2] https://fluxicon.com/disco
[3] https://www.minit.io
[4] https://lana-labs.com/en

the first to discuss a new event occurring in the news; and (3) given a small number of sample news stories about an event, *finding* all *following* stories in the stream. The work of [All02] has formally defined this problem and proposed the initial set of algorithms for the task. Main subtasks of TDT, as identified in [Way00] are (i) finding topically homogeneous regions (segmentation); (ii) finding additional stories about a given topic (tracking); (iii) detecting and threading together new topics (detection); (iv) Detecting new topics (first story detection); and (v) Deciding whether stories are on the same topic (linking). An example of a real-world TDT system is Google Alerts[5].

Summarization. Summarization is the task of reducing the content obtained from text documents, still keeping a brief overview on a topic that they treat. Summarization techniques generally fall into two categories [AZ12]. In extractive summarization, a summary consists of information units extracted from the original text; in contrast, in abstractive summarization, a summary may contain "synthesized" information units that may not necessarily occur in the text document. An automatic summarization process can be divided into three steps [GL09]: (1) the *preprocessing* step where a structured representation of the original text is obtained; (2) the *processing* step where an algorithm must transform the text structure into a summary structure; and (3) the *generation* step where the final summary is obtained from the summary structure. A plethora of text summarization tools can be found online. A few examples are the open source libraries such Open Text Summarizer[6], Sumplify[7] and Online summarize tool[8].

Categorization. Categorization aims at identifying the main themes of a document. This translates into assigning natural language documents to predefined categories according to their content [Seb02]. Categorization often relies on a thesaurus for which topics are predefined, and relationships are identified by looking for broad terms, narrower terms, synonyms, and related terms. Support Vector Machines (SVMs) are used to automatically learn text classifiers from examples [Joa98]. Categorization tools usually rank documents according to how much of their content fits in a particular topic.

Clustering. Clustering is a technique used to group similar documents, but it differs from categorization in that documents are clustered on the fly instead of through predefined topics. Documents can also appear in multiple subtopics, ensuring that useful documents are not omitted from the search results. A basic clustering algorithm creates a vector of topics for each document and measures the weights of how the document fits into each cluster. A survey of clustering algorithms can be found in [AZ12, Chap. 4].

Concept Linkage. Concept-linkage tools connect related documents by identifying their shared concepts, helping users find information they perhaps would not have found through

[5] https://www.google.com/alerts
[6] https://www.splitbrain.org/services/ots
[7] http://sumplify.com/
[8] http://www.tools4noobs.com/summarize/

traditional search methods [GL09]. It promotes browsing for information rather than searching for it. For example, a text mining software solution may easily identify a transitive closure in a set of topics {X, Y, Z}, i.e., a link between X and Y, a link between Y and Z, and a link between X and Z. With large sets of data, such a relationship could be disregarded by humans.

Information Visualization. Information visualization aims at visualizing large textual sources in such a way that the content can be displayed in a hierarchy or map and provides browsing features, in addition to simple search. For instance, governments or police can identify terrorist networks in a map and identify crimes that were previously unconnected [GL09].

Question Answering. Another application area of developed text-mining technologies, along with the natural language processing, is natural language features Question Answering (QA). QA is concerned with building systems that automatically answer questions posed by humans in a natural language. One of the first Query Answering tools was START[9], developed in the work of [Kat97]. Question answering has been extensively used in Biomedical domain for aiding researchers and health care professionals in managing the continuous growth of information.

Association Rule Mining. The focus of association rules mining is to study the relationships and implications among topics, or descriptive concepts, that are used to characterize a corpus. The work of [ASO94] shows how it is possible to discover association rules from massive data from databases, referred to as *basket* data. The same approach can be followed by constructing a database of rules using information extraction methods and subsequently applying techniques, e.g. [HZY+10], to uncover hidden associations in the database. For instance, a rule might be that 98% of customers that purchase tires and auto accessories also get automotive services done. This suggests that association rules can be captured as if/then patterns. Criteria such as support and confidence [AIS93] are used to identify the most important relationships. Support is an indication of how frequently the items appear in the database. Confidence indicates the number of times the if/then statements has been found to be true.

My research involves unstructured data in the form of word processor documents, emails, and commit messages from VCS. Text mining algorithms can be combined to quantitative data to gather information about project work. For example, topics models can be combined to time-clustered email messages, in order to discover “hot topics" (e.g. deliverable, milestone, meeting, etc) during project development.

2.3 Mining Software Repositories

Mining software repositories research has emerged in the last decade from the software engineering field. It focueses on analyzing the software repositories from a software engineering

[9] http://start.csail.mit.edu/index.php

perspective. Main topics in MSR are about

Software quality and metrics. Find quality of software code in terms of engineering metrics.

Visualization. (Visual software analytics)

Bug analysis. Etc.

Contributions in MSR use techniques from data mining, text mining and statistics. All these techniques allows to obtain several metrics and models from software repository data. However, only a few works focus on the actual process. As process mining research shows that analyzing the process gives relevant advantages over analyzing indicators and models that do not describe the process (cf. [vdA16]), I argue that there is the need for analyzing software repositories from a process perspective.

The benefits of such approach would be

list some benefits

In general, methods from MSR have a strength in analyzing dependencies in the structure of the software artifact, but an explicit consideration of the type of work is missing. Contributions in this area focus on the users and the artifacts, mining co-evolution or co-change of project parts [ZRDvD08, DLL09] and network analysis of file dependency graph based on commit distance [ZN08, ABDZ09, YSX13]. Hidden work dependencies are mentioned as *logical dependencies* [OSGdS11]. Also techniques for trend analysis [RH15] and inter-dependencies between developers [LBGL16] are proposed. However, none of these works considers the type of work being done in the process.

Surprisingly enough, the behavioral perspective, usually the one that has received more attention by the process mining discipline, was not much considered by MSR contributions

3 Problem Definition

This section defines the problem of mining the software development process. It gives an overview on the context to which the problem belongs and formulates the research question. Consequently, challenges are drawn and a number of objectives are identified as requirements for a solution.

s?

how many

how many

3.1 Context and Research Question

Software development processes are characterized as follows. First, although there is a planning phase, they are conducted creatively, i.e. there exists no strict process model that is followed by the developers. Second, they follow guidelines and methodologies on software development project, such Rational Unified Process (RUP), Scrum[10], Waterfall, etc. These guidelines are recommendations that stem from experience. However, none of them is extracted from empirical evidence. Third, typically there is a plethora of tools used and process evidence is scattered among different logs. Fourth, there is typically no process engine to control the execution. Fifth,

[10] https://www.scrum.org

it is highly relevant to track the process because of project managers want to be able to efficiently detect performance measures such as number of issues, bottlenecks, timeliness, handover of work, productivity, etc.

Software development processes fall into the category of *project-oriented business processes* [BCM+15]. That is, ad-hoc plans performed with limited resources and time, but with a clear goal: develop a software artifact. Unlike classic business processes which are best captured with notations such as Petri nets and BPMN, software development processes fall into the category of *project-oriented business processes*, which are usually captured by models such as Gantt and PERT diagrams.

An example of software development is the following. A new version of a software needs to be developed by company X. In general, company X knows how to develop software. However, before running straightly into the development phase, the company first makes sure to gather and properly formulate all the requirements. In a Scrum scenario these requirements would be written down as user stories. At a later stage, the project manager needs to plan the time and resources allocated to the respond to project deadlines. He can go through the list of requirements that need to be implemented and assign to an effort estimation value to each of them. With the plan done, the development phase can commence. During the development phase, resources address the task in a creative way, choosing the order of the tasks according to their own knowledge and expertise, until eventually all the tasks are terminated.

Several tools are used by software project participants to support their work. Therefore, traces about the overall process are typically scattered among different repositories and different artifacts, e.g., spreadsheets, word processor documents, programming languages files, emails, etc. This makes it difficult to obtain full knowledge about the overall business process. Two major challenges are to *correlate* heterogeneous events from different sources of evidence pertaining the same process and to *extract* information from unstructured data such as user comments when working on tasks. Thus, obtaining full process knowledge is a complex task.

Although the data is scattered and often unstructured, there are still repositories that we can study in order to obtain process knowledge. An important dimension of the software development process is the work perspective. It is particularly interesting for project managers to know if planning phase was realistic with respect to the development efforts. Two software tools that are commonly used in software development are ITS (e.g., JIRA, GitHub Issues) and VCS (e.g. Subversion, Git). The evolution of these two repositories can be analyzed to extract relevant knowledge about the software development process. Figure 2 illustrates the problem context.

Existing software repositories allow for many ways to access their log files. However, the most relevant information that they provide are about actions done by the user to change the repository state. In the more specific case, VCS logs consist of an ordered set of *commits* bearing information about users, files, timestamp, comments and type of change that were stored at particular moment in time. ITS typically come with richer information, most importantly they inform about users, task, type of task (e.g., bug, new feature, requirement, etc), timestamps, related issues, etc. To extract knowledge about the software development process, these properties

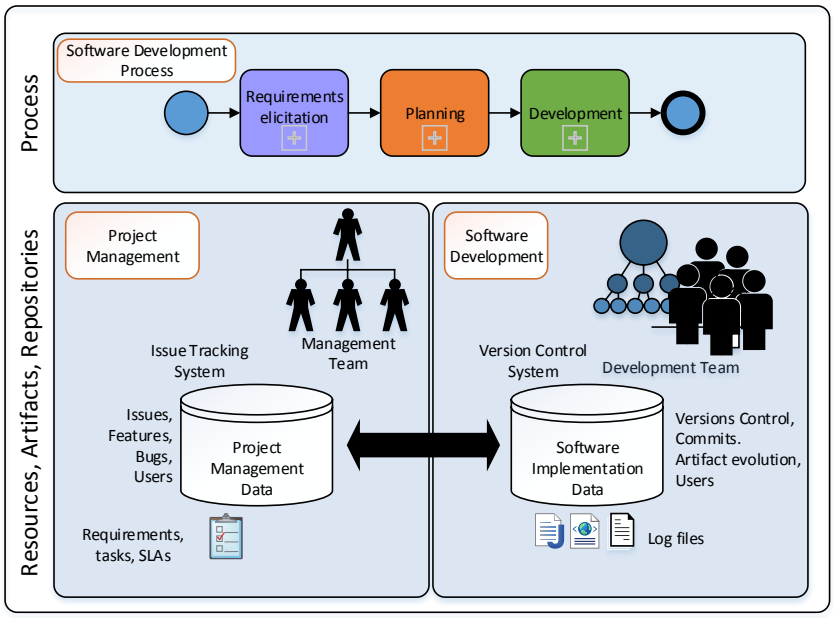

Figure 2: Software development scenario with two repositories used for project management and software development, respectively.

of repositories must be taken into account.

Because the project managers can benefit from a process view to better analyze hidden aspects (such as the behavioral one) of the software development process, we focus on adapting and transforming the data to fit process mining. Thus, we can better define our problem based on the four aspects of process mining [vdA16]: *i)* time perspective; *ii)* control-flow perspective; *iii)* organizational perspective; and *iv)* case perspective. In the following, we will define four requirements, each addressing one process mining perspective.

Big picture

On the one hand, project management data (GitHub Issues, JIRA, etc).

Issues, Features, Bugs, Text, Plan, Timestamps, Stories, Effort estimation (story points)

On the other hand, work traces from version control systems (e.g., Subversion, Git, Mercurial, etc).

Version control, Commits, Artifacts evolution, Users, Timestamps, etc.

Ends with Research Question: **RQ:** How does the specification of user stories influence software developn

3.2 Literature review

This problem is related to three areas: *i)* PM; *ii)* MSR; and *iii)* TM.

Process mining contributions have focused on transforming this problem into a process mining problem. These approaches enrich VCS log data with case and activity information and consequently use process mining to discover a model. In this category, Kindler et al. [KRS06a,

KRS06b] can discover a Petri net from a structured and enriched version control log. This approach was further improved by Rubin et al. [RGV+07] and a ProM[11] was provided. Poncin et al. [PSvdB11] provide the FRASR framework for preprocessing software repositories such that they can be used in ProM. While providing interesting insights, these contributions leave out many important aspects of software development projects, such as for instance trying to understand whether the process was done according to the plan.

MSR focuses on software engineering aspects, like code quality metrics, modularization, code complexity, user networks, and other important metric. In general, methods from MSR provide useful dependency analyses of the repository structure. Contributions in this area focus on the users, the artifacts and the repository evolution [ZRDvD08, DLL09], and network analysis of file dependency graph based on commit distance [ABDZ09, WSX13]. Also techniques for trend analysis [RH15] and inter-dependencies between developers [LBGL16] are proposed. However, none of these works aims at extracting knowledge about the business process.

TM focuses on obtaining structured information from unstructured textual data, and mainly uses NLP. Works that use NLP can be found in both PM and MSR. In the BPM area, NLP techniques have been used to understand process activities [Leo13, MLP14] and analyze software processes under a knowledge-intensive perspective [dSB11, RdBS17]. Likewise, in the MSR area, NLP has been used as an information extraction tool to obtain informative metrics from a software engineering perspective [THB14, CTH16].

This paper combines ideas from the above mentioned areas to devise algorithms for mining the software development process by exploiting techniques from MSR.

3.3 Solution Requirements

In the light of the above considerations, we derive the following requirements for extending process mining towards the analysis of software repositories.

R1. Mining the time perspective. Given a software repository, extract information about the temporal order of the activities. For example, the development activity takes 2 weeks on average, the average time of task creation is 15 minutes, etc.

R2. Mining the case perspective. Given a software repository, extract information about the case perspective. For example, all the bugs are solved in a 3 steps iteration, or a quality piece of code takes a conversation with 3 people and is successfully merged into the main branch after 1 week, etc.

R3. Mining the organizational perspective. Given a software repository, extract information about the organizational perspective. For example, the software development is carried out by a team of 4 people, the actual user roles of the company are developer and tester, etc.

R4. Mining the control-flow perspective. Given a software repository, extract information about the control-flow perspective. For example, the testing is always done before development, or while new features are worked on, also new requirements are created, etc.

Table 1 lists the most relevant works in the literature.

[11] http://www.promtools.org

Table 1: Classification of existing literature addressing the various aspects of mining the development process

	R1	R2	R3	R4	Output
PM			Visualization techniques [BS13], Organizational mining [SvdA08] [SCJM15]		Process model
TM	-	Topic models [CTH16], Theory-generating case studies [LBGL16]	Social network [BGD+06], Survey [BKZ10], [dASB10]	Speech acts [DM13], [CRBS18], Natural language processing [FMP11]	Structured data
MSR	Time series [RH15], [HMCX14], Statistical analyses [OSGdS11]	Network analyses [DLL09], [ZN08], Language Models [AS13]	Email analysis [BGD+06]		Software engineering metrics and artifacts

Fill in table Literature vs. Requirements

4 Research Design and Methodology

4.1 Research Method

What is the epistemological framework of the dissertation? For empirical studies it should be made clear: Why were the specific methods of data analysis chosen? How was the data acquired?

There are a number of research methods that can be used to understand and possibly improve the way people work on projects. This research is inspired by real world problems gathered from the SHAPE project[12]. This work intends to support project managers or auditors who must validate the compliance of the work against existing rules and regulations. This involves building new artifacts in order to both partially automate compliance checking and help by providing better overviews on the existing process.

[12] https://aic.ai.wu.ac.at/shape-project/

Design Science Research (DSR) is a research method which creates and evaluates IT artifacts intended to solve identified organizational problems. It was first introduced by Hevner et al. [HMPR04], who also give the following seven guidelines on doing design-science. Here, I use these guidelines to discuss my research.

Guideline 1 – Design as an Artifact. Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation In my research, I use state-of-the-art techniques and develop new artifacts that allow to capture information from both structured and unstructured types of data.

Guideline 2 – Problem Relevance. The objective of design-science research is to develop technology based solutions to important and relevant business problems. My research takes inspiration from real world needs which include human-centric safety-critical automated solutions in the railway domain.

Guideline 3 – Design Evaluation. The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods. The real world scenarios that I encounter in the SHAPE project require for novel solutions, which involve the design of new algorithms. Algorithms are converted to operational software. This operational software is an instantiated artifact [GH13], which is then tested against real data.

Guideline 4 – Research Contributions. Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies. The contribution of my research can be positioned as a set of *exapted* methods (cf. DSR knowledge contribution framework [GH13]) from the fields of process mining and text mining, which contribute to a better understanding of projects.

Guideline 5 – Research Rigor. Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact. My research builds upon existing work from natural language processing [DMMO06, CG13, KM03], a number of VCS mining works [BA15, AS13, VPR+15, KBDGAW15, GAH15], and process mining. I plan to adopt mature methods from the mentioned works and construct my artifacts upon existing ones.

Guideline 6 – Design as a Search Process. The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment. I plan to build my artifacts based on state-of-the-art methods and technology and follow both a rigorous process as the one described in [PTRC08] (cf. fig. 3). Given the nature of my data, an exploratory phase may be required in the initial activity of this process. This may involve exploratory data analysis, as described in the data science process [SO13, p. 41]

Guideline 7 – Communication of Research. Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences. I will use

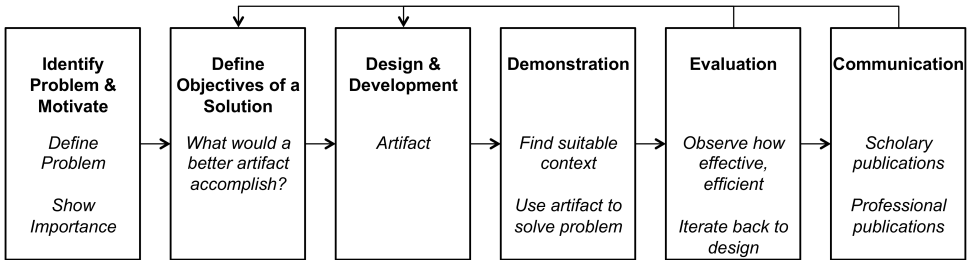

Figure 3: The Design Science Research process, adapted from [PTRC08]

guidelines [GH13, RM15] in order to properly position my work. The main target will be BPM conferences and journals.

4.2 Data Collection

Description of my dataset.

The data was collected from projects.

GitHub data.

Project extraction.

Dimensions of the data.

Coding of the data.

4.3 Preliminary Results

This section presents efforts done towards addressing the requirements presented in Section 3.

Rephrase the subsections below as solutions, not as challenges.

4.3.1 Mining the Time Perspective (R1)

Mining the time perspective from a software repository is defined as extracting temporal process knowledge. This problem relates to monitoring whether the process was executed respecting a predefined plan. Typically the process involves various actors which track their work through the use of VCS. Often, the plans are not represented in standard process notation. Rather, Gantt or PERT charts are used.

Challenges are *i) time approximation* (i.e., when the activity really started, compared to when it was registered in the log); *ii) granularity* (i.e., be able to switch from a detailed view of the single events and a coarse-grained view of the overall project); and *iii) coverage* (i.e., how much work effort was put within the duration of the activity). Figure 4 depicts the idea of mining a Gantt chart from VCS logs. The output is presented in a way that is informative to project managers. Details about the technique can be found in [BCM+15].

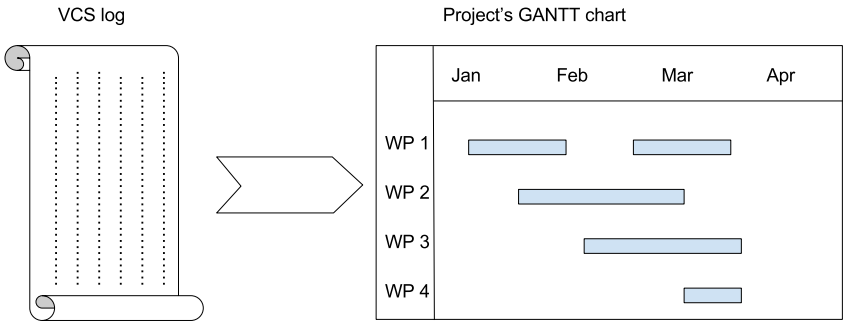

Figure 4: Mining the time perspective of software development into a Gantt chart

4.3.2 Mining the Case Perspective (R2)

The identification of process cases is not an easy task when it comes to software development data. However, a highly informative case candidate can be considered the data artifact itself. According to [vdA16], cases can also be characterized by the values of data elements. In line with this, it is possible to devise techniques that study the *artifact evolution*. Although it can be input to process mining techniques, this evolution can also be analyzed as a time series. An example of such evolution is given in Figure 5, which shows the lines of code (LOC) (LOC) changes over time.

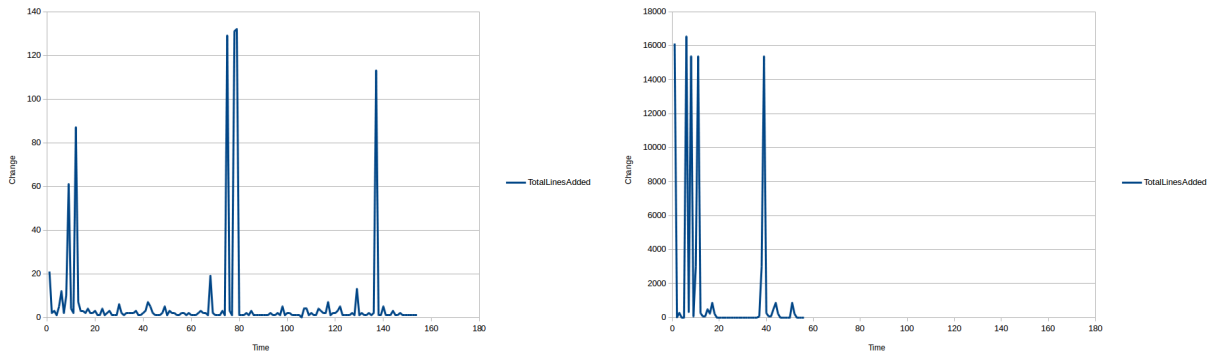

Figure 5: Evolution of files in a version control systems: LOC changes over time

Challenges related to the artifact evolution pertain *prediction* of plateaus and *pattern recognition*. The results reflect work patterns over the artifact. For example, when a data file is not being modified anymore, its time series has a plateau and it can be interpreted that the document is now ready for release. A relevant problem in software development is bad modularization. Especially, the dependency of two artifacts on one another is considered a bad practice. Two time series can be compared together and file dependencies can be found that reflect work coupling. This is equivalent to finding dependence between two processes which might have been designed for different purposes.

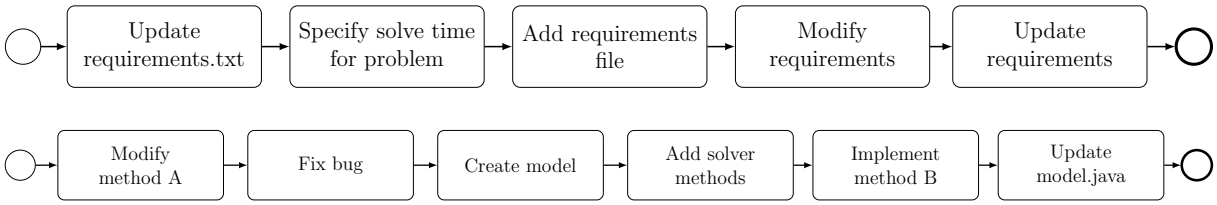

Figure 6: Processes of two work-dependent files. They may seem different but they co-evolve in time and space, i.e., they describe the same *case*.

We have devised a proof-of-concept in [BRd+17]. Figure 6 shows a possible outcome of the technique. The result is obtained by identifying couples of files with similar evolution and mining a business process from the comments. As the result is aggregated, several comments have been combined together an mined as a story. Among other things, the technique allows to profile existing software development projects.

4.3.3 Mining the Organizational Perspective (R3)

Software development projects are knowledge intensive, thus involve creativity and flexibility. Project participants are free to tackle development tasks according to their expertise and ideas to solve ad-hoc problems. De facto, members may behave according several roles in the organization. Therefore, the discovery of the roles of a software project may give important insights on the actual project. Role discovery can help in the monitoring existing projects in two ways. First, emerging roles can be analyzed in order to have a better skills profiling of the resources. Second, emerging roles can be used to check whether resources are breaking contractual agreements or norms.

We have developed an approach for tackling the problem of mining the organizational perspective in software development projects. The approach provides information about the actors involved in the business process and their relations. In substance, it provides automatic classification of resources into company roles (e.g., developer, tester, etc) based on the comments of project participants. It works on VCS logs and provides information about the people, their roles, other systems involved, the organization hierarchy, the social network, and resource profiling. Figure 7 shows an example of resource profiles automatically inferred from VCS comments.

Challenges of using NLP concern the *jargon* used by software developers. They often use technical terms, very short phrases, or a number of codes and hyperlinks. These makes NLP techniques score low results even on simple tasks like sentence parsing, or name-entity recognition. Therefore, approaches should take into account several factors when mining for the organizational perspective. The result allows the manager to have measurable information about the resources. For example, do two people work better together and how can we build the best team? Details of the approach to extract resource profiles can be found in [AAT+16].

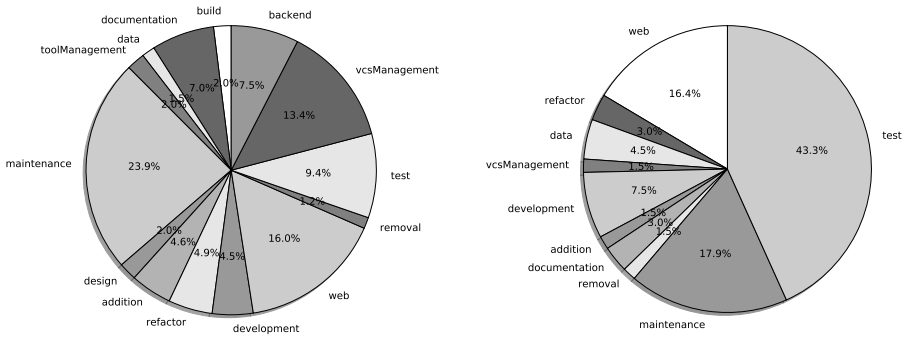

Figure 7: Profiles of a developer and a tester, from left to right.

5 Next Steps and Expected Results

Which results can be expected? What is new? Where lies the progress for science? In what way can scientific discussion proceed / be stimulated by the thesis?

Rephrase as solution, not as a challenge.

5.1 Mining the Control-Flow Perspective (R4)

Important knowledge of the control flow of a software development process can be extracted from VCS and ITS. One important mechanism widely used in software development for collaboration are *pull requests*. A pull request is issued every time a developer wants to contribute to the main source with a new piece of code. Pull requests trigger communication among different people and may help shedding light into problems, learn new things and obtain new ideas on the existing software. It is interesting for project managers and developers to better understand factors and behavior that makes a pull request successful.

In ongoing work, we are investigating the relation of the pull request process and surrounding factors such as the generation of new ideas or change of behavior. We have obtained a data set of pull requests spanning over two years from an real world repository. User conversations have been manually annotated with codes that classify them into categories of comments. We are current able to discriminate whether a comment is a new idea, an assumption, a merge that closes a pull request, etc. Considering the pull request identifiers as cases and the codes as activities, we are able extract the process of the pull requests. It is interesting to compare how the conversation (i.e., behavioral perspective) unfolds before and after a pull request is merged. We plan to explain these differences by also taking into account further information from text such as emotions.

The challenge of mining the control-flow usually pertains the completeness of data and the case and activity identification. While PM contribution have been focusing mostly on the control-flow perspective, techniques from MSR hardly go beyond mining the bug lifecycle [**?**].

The nature of software development process makes it difficult to recognize recurrent activities when only VCS are analyzed.

5.2 Expected Results

Expected results are the following.

1. Body of techniques which, drawing from neighboring disciplines such as text mining and mining software repositories, allow for the application of process mining approaches on software development data
2. A dataset of software repositories automatically classified using a human coded dataset as a training set

6 Dissertation Relevance and Publication Plan

6.1 Implications for Research

This position paper presents research on mining the software development process. Approaches towards discovering the perspectives of time, resources and cases have been developed and the results suggest that obtaining process knowledge from software repositories brings new insights for the project managers [4,6,7]. In future work, we plan to complete the research by tackling the problem of discovering the flow-perspective of the software development process. Usefulness will be evaluated through user studies. This research extends the process mining field towards its adoption on software repositories. Project managers would benefit from this research by having a process perspective on their ongoing projects through visual models and diagrams, thus overcoming existing flat approaches based on simple indicators such as burndown charts or change plots offered by existing tools.

6.2 Work Plan and Dissemination Timeline

These are all. Put only the relevant ones?

So far, the following publications have been achieved.

- **Bala, S.**, Cabanillas, C., Mendling, J., Rogge-Solti, A., and Polleres, A.: *Mining Project-Oriented Business Processes*. In Hamid Reza Motahari-Nezhad, Jan Recker, and Matthias Weidlich, editors, BPM 2015, Innsbruck, Austria, volume 9253 of Lecture Notes in Computer Science, pages 425–440. Springer, 2015. [BCM+15]
- **Bala, S.**, Cabanillas, C., Haselböck, A., Havur, G., Mendling, J., Polleres, A., Sperl, S., Steyskal, S.: *A Framework for Safety-Critical Process Management in Engineering Projects*. In: Ceravolo, P. and Rinderle-Ma, S. (eds.) SIMPDA (Revised Selected Papers). pp. 1–27. Springer (2015).

- Agrawal, K., Aschauer, M., Thonhofer, T., **Bala, S.**, Rogge-Solti, A., Tomsich, N.: *Resource Classification from Version Control System Logs*. In: Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOC Workshop. pp. 249–258 (2016).
- B**ala, S.**, Havur, G., Sperl, S., Steyskal, S., Haselböck, A., Mendling, J., Polleres, A.: SHAPEworks: A BPMS extension for complex process management. In: CEUR Workshop Proceedings. pp. 50–55 (2016).
- **Bala, S.**, Revoredo, K., de A.R. Gonçalves, J.C., Baião, F., Mendling, J., Santoro, F.: *Uncovering the Hidden Co-evolution in the Work History of Software Projects*. In: Carmona, J., Engels, G., and Kumar, A. (eds.) Business Process Management - 15th International Conference, BPM 2017, Barcelona, Spain, September 10-15, 2017, Proceedings. pp. 164–180. Springer (2017).
- **Bala, S.**: *Mining projects from structured and unstructured data*. In: CEUR Workshop Proceedings (2017).
- Woliński, B., **Bala, S.**: *Comprehensive Business Process Management at Siemens: Implementing Business Process Excellence*. In: vom Brocke, J. and Mendling, J. (eds.) Business Process Management Cases: Digital Innovation and Business Transformation in Practice. pp. 111–124. Springer International Publishing, Cham (2018).
- **Bala, S.**, Mendling, J.: *Monitoring the Software Development Process with Process Mining*. In Boris Shishkov, editor, Business Modeling and Software Design, volume 319 of Lecture Notes in Business Information Processing, 2018

From ongoing work the following publications are expected.

- Case study on tool productivity. Collaboration with researchers from the University of Ljubljana to be submitted to a class A journal on information systems.
- Process mining pull requests for identifying idea-creation patterns. Collaboration with researchers from Stevens Institute of Technology to be submitted to a class A journal on information systems.
- Mining knowledge intensive processes from pull requests. Collaboration with researchers from Federal University of the State of Rio de Janeiro (UNIRIO) to be submitted to a class A conference.

[BCH+15, AAT+16, BCM+15, BRd+17, BM18]

References

[AAT+16] Kushal Agrawal, Michael Aschauer, Thomas Thonhofer, Saimir Bala, Andreas Rogge-Solti, and Nico Tomsich. Resource Classification from Version Control

System Logs. In *Proc. - IEEE Int. Enterp. Distrib. Object Comput. Work. EDOCW*, volume 2016-September, pages 249–258, 2016.

[ABDZ09] Pietro Abate, Jaap Boender, Roberto Di Cosmo, and Stefano Zacchiroli. Strong dependencies between software components. *2009 3rd Int. Symp. Empir. Softw. Eng. Meas. ESEM 2009*, pages 89–99, 2009.

[AIS93] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD Rec.*, volume 22, pages 207–216. ACM, 1993.

[All02] James Allan. Introduction to topic detection and tracking. In *Top. Detect. Track.*, pages 1–16. Springer, 2002.

[AS13] Miltiadis Allamanis and Charles Sutton. Mining source code repositories at massive scale using language modeling. *IEEE Int. Work. Conf. Min. Softw. Repos.*, pages 207–216, 2013.

[ASO94] Rakesh Agrawal, Ramakrishnan Srikant, and Others. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499, 1994.

[AZ12] C C Aggarwal and C X Zhai. *Mining Text Data*. Springer, 2012.

[BA15] Shadi Banitaan and Mamdouh Alenezi. Software Evolution via Topic Modeling : An Analytic Study. *Int. J. Softw. Eng. Its Appl.*, 9(5):43–52, 2015.

[BCH+15] Saimir Bala, Cristina Cabanillas, Alois Haselböck, Giray Havur, Jan Mendling, Axel Polleres, Simon Sperl, and Simon Steyskal. A Framework for Safety-Critical Process Management in Engineering Projects. In Paolo Ceravolo and Stefanie Rinderle-Ma, editors, *SIMPDA (Revised Sel. Pap.*, volume 244 of *Lecture Notes in Business Information Processing*, pages 1–27. Springer, 2015.

[BCM+15] Saimir Bala, Cristina Cabanillas, Jan Mendling, Andreas Rogge-Solti, and Axel Polleres. Mining Project-Oriented Business Processes. In Hamid Reza Motahari-Nezhad, Jan Recker, and Matthias Weidlich, editors, *BPM 2015, Innsbruck, Austria*, volume 9253 of *Lecture Notes in Computer Science*, pages 425–440. Springer, 2015.

[BCS+07] Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction for the web. In *IJCAI*, volume 7, pages 2670–2676, 2007.

[BGD+06] Christian Bird, Alex Gourley, Prem Devanbu, Michael Gertz, and Anand Swaminathan. Mining email social networks. In *Proc. 2006 Int. Work. Min. Softw. Repos.*, pages 137–143. ACM, 2006.

[BKZ10] Andrew Begel, Yit Phang Khoo, and Thomas Zimmermann. Codebook: discovering and exploiting relationships in software repositories. In *Proc. 32nd ACM/IEEE Int. Conf. Softw. Eng. 1*, pages 125–134. ACM, 2010.

[BM18] Saimir Bala and Jan Mendling. Monitoring the software development process with process mining. In Boris Shishkov, editor, *Business Modeling and Software Design*, volume 319 of *Lecture Notes in Business Information Processing*, 2018.

[BRd+17] Saimir Bala, Kate Revoredo, João Carlos de A.R. Gonçalves, Fernanda Baião, Jan Mendling, and Flavia Santoro. Uncovering the Hidden Co-evolution in the Work History of Software Projects. In Josep Carmona, Gregor Engels, and Akhil Kumar, editors, *Bus. Process Manag. - 15th Int. Conf. {BPM} 2017, Barcelona, Spain, Sept. 10-15, 2017, Proc.*, volume 10445 of *Lecture Notes in Computer Science*, pages 164–180. Springer, 2017.

[BS13] Anne Baumgrass and Mark Strembeck. Bridging the gap between role mining and role engineering via migration guides. *Inf. Sec. Techn. Rep.*, 17(4):148–172, 2013.

[CG13] Luciano Del Corro and Rainer Gemulla. ClausIE : Clause-Based Open Information Extraction. *Proc. 22nd Int. Conf. World Wide Web*, pages 355–365, 2013.

[CL96] Jim Cowie and Wendy Lehnert. Information extraction. *Commun. ACM*, 39(1):80–91, 1996.

[CRBS18] Júlio Campos, Pedro Richetti, Fernanda Araújo Baião, and Flávia Maria Santoro. Discovering Business Rules in Knowledge-Intensive Processes Through Decision Mining: An Experimental Study BT - Business Process Management Workshops. pages 556–567, Cham, 2018. Springer International Publishing.

[CTH16] Tse-Hsun Chen, Stephen W Thomas, and Ahmed E Hassan. A survey on the use of topic models when mining software repositories. *Empir. Softw. Eng.*, 21(5):1843–1919, 2016.

[dASB10] Joao Carlos de A.R.Goncalves, F M Santoro, and F A Baiao. A case study on designing business processes based on collaborative and mining approaches. In *Comput. Support. Coop. Work Des. (CSCWD), 2010 14th Int. Conf.*, pages 611–616, apr 2010.

[DLL09] Marco D'Ambros, Michele Lanza, and Mircea Lungu. Visualizing co-change information with the evolution radar. *IEEE Trans. Softw. Eng.*, 35(5):720–735, 2009.

[DM13] Claudio Di Ciccio and Massimo Mecella. Mining Artful Processes from Knowledge Workers' Emails. *Internet Comput. IEEE*, 17(5):10–20, 2013.

[DMMO06] Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, and Others. Generating typed dependency parses from phrase structure parses. In *Proc. Lr.*, volume 6, pages 449–454, 2006.

[dSB11] João Carlos de A. R. Gonçalves, Flávia Maria Santoro, and Fernanda Araujo Baião. Let Me Tell You a Story - On How to Build Process Models. *J. {UCS}*, 17(2):276–295, 2011.

[FMP11] Fabian Friedrich, Jan Mendling, and Frank Puhlmann. Process Model Generation from Natural Language Text. In Haralambos Mouratidis and Colette Rolland, editors, *Adv. Inf. Syst. Eng. - 23rd Int. Conf. CAiSE 2011, London, UK, June 20-24, 2011. Proc.*, volume 6741 of *Lecture Notes in Computer Science*, pages 482–496. Springer, 2011.

[GAH15] DanielM. German, Bram Adams, and AhmedE. Hassan. Continuously mining distributed version control systems: an empirical study of how Linux uses Git. *Empir. Softw. Eng.*, pages 1–40, 2015.

[GH13] Shirley Gregor and Alan R Hevner. Positioning and Presenting Design Science Research for Maximum Impact. *MIS Q.*, 37(2):337–355, 2013.

[GL09] Vishal Gupta and Gurpreet S Lehal. A survey of text mining techniques and applications. *J. Emerg. Technol. web Intell.*, 1(1):60–76, 2009.

[HMCX14] Qiuju Hou, Yutao Ma, Jianxun Chen, and Youwei Xu. An Empirical Study on Inter-Commit Times in {SVN}. In Marek Reformat, editor, *26th Int. Conf. Softw. Eng. Knowl. Eng. Hyatt Regency, Vancouver, BC, Canada, July 1-3, 2013.*, pages 132–137. Knowledge Systems Institute Graduate School, 2014.

[HMPR04] Alan R Hevner, Salvatore T March, Jinsoo Park, and Sudha Ram. Design Science in Information Systems Research. *MIS Q.*, 28(1):75–105, 2004.

[HZY+10] Xiaohua Hu, Xiaodan Zhang, Illhoi Yoo, Xiaofeng Wang, and Jiali Feng. Mining hidden connections among biomedical concepts from disjoint biomedical literature sets through semantic-based association rule. *Int. J. Intell. Syst.*, 25(2):207–223, 2010.

[JM14] Dan Jurafsky and James H Martin. *Speech and language processing*. Pearson, 2014.

[Joa98] Thorsten Joachims. *Text categorization with support vector machines: Learning with many relevant features*. Springer, 1998.

[Kat97] Boris Katz. From sentence processing to information access on the world wide web. In *AAAI Spring Symp. Nat. Lang. Process. World Wide Web*, volume 1, page 997. Stanford University Stanford, CA, 1997.

[KBDGAW15] Zahra Karimi, Ahmad Baraani-Dastjerdi, Nasser Ghasem-Aghaee, and Stefan Wagner. Links between the personalities, styles and performance in computer programming. *J. Syst. Softw.*, 111:228–241, 2015.

[KM03] Dan Klein and Christopher D Manning. Accurate unlexicalized parsing. In *Proc. 41st Annu. Meet. Assoc. Comput. Linguist. 1*, pages 423–430. Association for Computational Linguistics, 2003.

[KRS06a] Ekkart Kindler, Vladimir Rubin, and Wilhelm Schäfer. Activity Mining for Discovering Software Process Models. *Softw. Eng.*, 79:175–180, 2006.

[KRS06b] Ekkart Kindler, Vladimir Rubin, and Wilhelm Schäfer. Incremental Workflow Mining Based on Document Versioning Information. In Mingshu Li, Barry Boehm, and LeonJ. Osterweil, editors, *Unifying Softw. Process Spectr.*, volume 3840 of *Lecture Notes in Computer Science*, pages 287–301. Springer Berlin Heidelberg, 2006.

[LBGL16] Aron Lindberg, Nicholas Berente, James Eric Gaskin, and Kalle Lyytinen. Coordinating Interdependencies in Online Communities: A Study of an Open Source Software Project. *Inf. Syst. Res.*, 27(4):751–772, 2016.

[Leo13] Henrik Leopold. *Natural Language in Business Process Models - Theoretical Foundations, Techniques, and Applications*, volume 168 of *Lecture Notes in Business Information Processing*. Springer, 2013.

[MLP14] Jan Mendling, Henrik Leopold, and Fabian Pittke. 25 Challenges of Semantic Process Modeling. *Int. J. Inf. Syst. Softw. Eng. Big Co.*, 1(1):78–94, 2014.

[Moo99] R Mooney. Relational learning of pattern-match rules for information extraction. In *Proc. Sixt. Natl. Conf. Artif. Intell.*, pages 328–334, 1999.

[OSGdS11] Gustavo A. Oliva, Francisco W.S. Santana, Marco A. Gerosa, and Cleidson R.B. de Souza. Towards a classification of logical dependencies origins. *Proc. 12th Int. Work. 7th Annu. ERCIM Work. Princ. Softw. Evol. Softw. Evol. - IWPSE-EVOL '11*, page 31, 2011.

[PK16] Martin Pinzger and Sunghun Kim. Guest editorial: mining software repositories. *Empir. Softw. Eng.*, 21(5):2033–2034, 2016.

[PSvdB11] Wouter Poncin, Alexander Serebrenik, and Mark van den Brand. Process Mining Software Repositories. *2011 15th Eur. Conf. Softw. Maint. Reengineering*, pages 5–14, 2011.

[PTRC08] Ken Peffers, Tuure Tuunanen, Marcus Rothenberger, and Samir Chatterjee. A Design Science Research Methodology for Information Systems Research. *J. Manag. Inf. Syst.*, 24(January):45–77, 2008.

[RdBS17] Pedro H Piccoli Richetti, João Carlos de A. R. Gonçalves, Fernanda Araujo Baião, and Flávia Maria Santoro. Analysis of Knowledge-Intensive Processes Focused on the Communication Perspective. In Josep Carmona, Gregor Engels, and Akhil Kumar, editors, *Bus. Process Manag. - 15th Int. Conf. {BPM} 2017, Barcelona, Spain, Sept. 10-15, 2017, Proc.*, volume 10445 of *Lecture Notes in Computer Science*, pages 269–285. Springer, 2017.

[RGV+07] Vladimir Rubin, Christian W Günther, Wil M P Van Der Aalst, Ekkart Kindler, Boudewijn F Van Dongen, and Wilhelm Schäfer. Process mining framework for software processes. In *Softw. Process Dyn. Agil.*, pages 169–181. Springer, 2007.

[RH15] Jukka Ruohonen and Sami Hyrynsalmi. Time series trends in software evolution. (November):990–1015, 2015.

[RM15] Jan Recker and Jan Mendling. The State of the Art of Business Process Management Research as Published in the BPM Conference. *Bus. Inf. Syst. Eng.*, 58(1):1–18, 2015.

[SCJM15] Stefan Schönig, Cristina Cabanillas, Stefan Jablonski, and Jan Mendling. Mining the Organisational Perspective in Agile Business Processes. In *BPMDS*, page In press., 2015.

[Seb02] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002.

[SMR99] Kristie Seymore, Andrew McCallum, and Roni Rosenfeld. Learning hidden Markov model structure for information extraction. In *AAAI-99 Work. Mach. Learn. Inf. Extr.*, pages 37–42, 1999.

[SO13] Rachel Schutt and Cathy O'Neil. *Doing data science: Straight talk from the frontline*. O'Reilly Media, Inc., 2013.

[SvdA08] Minseok Song and Wil van der Aalst. Towards comprehensive support for organizational mining. *Decis. Support Syst.*, 2008.

[THB14] Stephen W Thomas, Ahmed E Hassan, and Dorothea Blostein. Mining unstructured software repositories. In *Evol. Softw. Syst.*, pages 139–162. Springer, 2014.

[vdA16] Wil M P van der Aalst. *Process mining: data science in action*. Springer, 2016.

[vdAWM04] Wil M P van der Aalst, Ton Weijters, and Laura Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Trans. Knowl. Data Eng.*, 16(9):1128–1142, 2004.

[vDdMV+05] Boudewijn F van Dongen, Ana Karla A de Medeiros, H M W Verbeek, AJMM Weijters, and Wil M P Van Der Aalst. The ProM framework: A new era in process mining tool support. In *Appl. Theory Petri Nets 2005*, pages 444–454. Springer, 2005.

[VPR+15] Bogdan Vasilescu, Daryl Posnett, Baishakhi Ray, Mark G J van den Brand, Alexander Serebrenik, Premkumar Devanbu, and Vladimir Filkov. Gender and Tenure Diversity in GitHub Teams. *Proc. 33rd Annu. ACM Conf. Hum. Factors Comput. Syst.*, pages 3789–3798, 2015.

[Way00] Charles L Wayne. Multilingual Topic Detection and Tracking: Successful Research Enabled by Corpora and Evaluation. In *LREC*, 2000.

[WBMT99] Ian H Witten, Zane Bray, Malika Mahoui, and Bill Teahan. Text Mining: A New Frontier for Lossless Compression. In *Proc. Conf. Data Compression*, DCC '99, pages 198—-, Washington, DC, USA, 1999. IEEE Computer Society.

[WSX13] Yang Weicheng, Beijun Shen, and Ben Xu. Mining GitHub: Why Commit Stops - Exploring the Relationship between Developer's Commit Pattern and File Version Evolution. In Pornsiri Muenchaisri and Gregg Rothermel, editors, *APSEC 2013, Ratchathewi,Thailand, December 2-5, 2013 - Vol. 2*, pages 165–169. IEEE Computer Society, 2013.

[YSX13] Weicheng Yang, Beijun Shen, and Ben Xu. Mining GitHub: Why Commit Stops - Exploring the Relationship between Developer's Commit Pattern and File Version Evolution. In Pornsiri Muenchaisri and Gregg Rothermel, editors, *20th Asia-Pacific Softw. Eng. Conf. {APSEC} 2013, Ratchathewi, Bangkok, Thailand, December 2-5, 2013 - Vol. 2*, pages 165–169. {IEEE} Computer Society, 2013.

[ZN08] Thomas Zimmermann and Nachiappan Nagappan. Predicting defects using network analysis on dependency graphs. *Proc. 13th Int. Conf. Softw. Eng. - ICSE '08*, page 531, 2008.

[ZRDvD08] Andy Zaidman, Bart Van Rompaey, Serge Demeyer, and Arie van Deursen. Mining Software Repositories to Study Co-Evolution of Production & Test Code. In *ICST*, pages 220–229. {IEEE} Computer Society, 2008.