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

Software development projects are highly complex in nature. 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 endeavor is different. Projects may deviate from the plan for a number of reasons. Inadequate decisions made on the planing phased, bad management or improper resource allocation, to cite a few, may lead to situations where many ad-hoc interventions are required. In turn, this leads to unpredicted costs, missing milestone deadlines, unforeseen maintenance effort, and unsatisfactory quality. Therefore, monitoring the software development process is of utmost importance to project managers who want to deliver quality software in time and within budget.

Software process monitoring helps managers control the various aspects related to the development. Especially, it is a scaffold for gaining transparency about the work being done at specific times. In particular, managers want to know the actual versus planned progress, what are the resources that actively participate in the work, whether resource-occupancy is high, what are actual tasks performed by the different organizational roles, what are typical patterns of work that lead to specific outcomes, etc. A starting point for analysis is the empirical evidence given from the historical evolution of artifacts (e.g., files, documentation, etc). Therefore, there is the need for approaches that are able to extract process knowledge from the artifacts produced and stored in software repositories.

"This second paragraph is full of errors and lacks clarity. before talking about potential solutions, you should first dedicate a paragraph to the problem. Here you jump to solutions. Solution is already in the second sentence." – Addressed?!

Literature has addressed several aspects about obtaining process insights from historical data. For instance, pieces of evidence stemming from information systems used by people to work on tasks have been used to obtain knowledge about actual time patterns (e.g., when and how long do tasks take?) [LWR14], case perspective (e.g., what are the tasks and decisions?) [vdA05], organizational structure (e.g., who works on which task?) [SCJM16], and flow perspective (e.g., how are tasks logically connected?) [vdARW+07]. These event data is the output of the different tools that are used by the project stakeholders. Examples are documentation, file changes in a Version Control System (VCS), user comments in blogs, emails, task management in a Issue Tracking System (ITS). Hence, approaches that can handle semi-structured data (e.g. commits of a VCS) are needed.

Nevertheless, different disciplines tackle different aspects of mining the software development process. Contributions from the Process Mining (PM) discipline focus mainly on obtaining process models from well structured event logs [vdA16]. Contributions from the Mining Software Repositories (MSR) area focus on obtaining results from a software engineering point of view, e.g., code quality, code complexity, user analysis, functional dependencies, software visualization, etc [PK16]. Lastly, contributions from Text Mining (TM) focus on dealing with unstructured data. Hence, they are fundamental for analyzing user comments in software repositories [AZ12]. In order to obtain a better understanding of the real software development process these approaches should be combined.

With this dissertation, I aim at raising the transparency and objectiveness of software development practices by devising new algorithms and techniques that draws from the aforementioned disciplines. My research is guided by the question **RQ:** "*How can we make use of project event data to gather insights from the software development process that are informative to managers?*". To answer this question, new approaches are devised such as algorithms and methods for handling semi- and un-structured data which are not generated by a business process engine, e.g., logs from VCS and ITS. This includes transforming real data taken from the railway domain or from other real project into structured logs and further analyze them. At current stage, this thesis already contains a number of contributions that tackle the problem from different aspects. Among other research results, prototypes have been developed for helping managers obtain high level cues on the projects. First, this thesis provides a tool 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. Second, this thesis devises a prototype to understand the *de facto* roles of software project participants based on their comments in the VCS. Third, this thesis provides a prototype to spot signs of work dependencies which may lead to inefficient practices. A fourth prototype is currently being developed in the context of ongoing work on mining pull requests from ITS. With this last prototype, this thesis addresses all four processes aspects, i.e., time, case, organization, and control-flow.

The rest of this research proposal is organized as follows. Section 2 describes the problem, the existing literature, and derives the solution requirements. Section 3 provides background knowledge on the related fields of process mining, text mining, and mining software repositories. Section 4 explains the research method, the dataset and preliminary results in addressing the research requirements. Section 6 presents the expected results and future steps towards the completion of this dissertation. Section 7 draws the implications for research and presents a dissemination plan.

instead of talking about which data you have you should talk about the general challenges and the general solution idea. PM is a means to address your problem, not an end by itself! – Addressed?!

2 Problem Definition

Software development processes are highly complex endeavors that require the coordination of multiple resources. Compared to standard business processes, they present the following characteristics. First, although there is a planning phase, they involve creativity in executing the tasks, i.e. there exists no strict process model that is followed by the developers to produce a new piece of code. Second, they follow guidelines and methodologies on software development project, such Rational Unified Process (RUP), Scrum[1], Waterfall, etc. These guidelines are recommendations that merely stem from experience and were not extracted from empirical evidence given by the history of the project. Third, they typically make use of software tools such as ITS, VCS, Integrated Development Environment (ide), and so on. These tools typically

mining the software process is not a research problem, but a practical problem which springs from a not yet clearly described research problem.

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

record work activities into log files. Fourth, there is in general no process engine to control the execution. Rather, a project manager or a Scrum master has to make sure that tasks are made available to the corresponding resources. Fifth, it is not trivial to obtain high level information on performance measures such as what is the ideal number of issues, which are resource bottlenecks, what are time requirements for certain tasks, what type of work is actually being done, and how productive are people working in certain tasks, etc. Hence, there is the need for algorithms and tools that help extracting this knowledge.

Software development processes fall into the category of *project-oriented business processes* [BCM+15]. That is, rather 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 straight 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. Every time each resource want to save their progress issue a so-called *commit* command which creates a new version of the modified files on the VCS. Likewise, every time a certain task (which may be worked carried out through by different commits) is completed, it is marked as done in the ITS. Both the commits and the tasks can be commented by the users, respectively to document the changes and raise a discussion for better understanding the task goals.

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. This perspective is particularly interesting to project managers. One of their goal is to know whether the planning phase was realistic with

respect to the development efforts. Existing software repositories allow for many ways to access their log files. These log files offer factual information about actions done by the process participants to change the repository state.

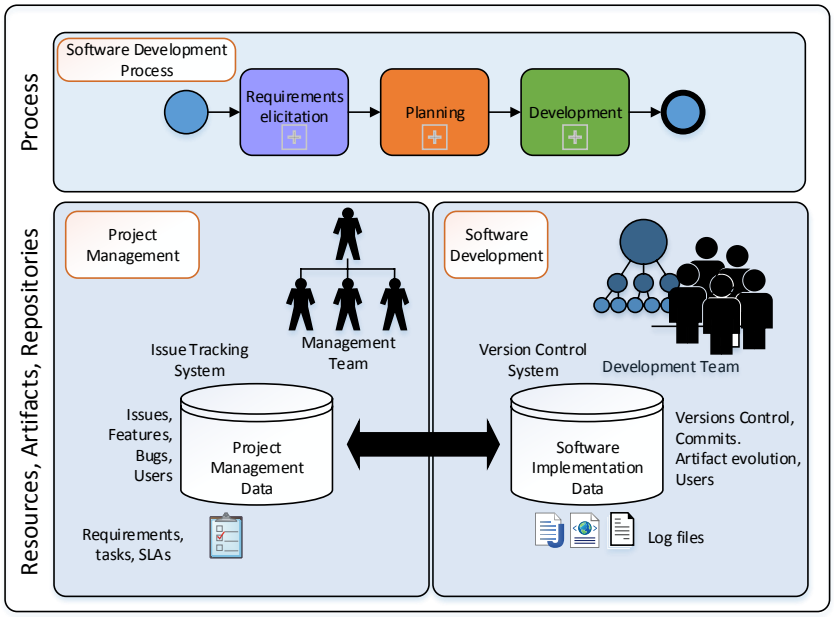

Figure 1: Software development scenario with two repositories used for project management and software development, respectively.

Figure 1 illustrates such problem scenario. Two software tools that are commonly used together in software development are ITS (e.g., JIRA, GitHub Issues) and VCS (e.g. Subversion, Git). One the one hand, ITS is used for tracking the project management aspect. This aspect offers plan related information, such as issues, features, bugs, text, planned and executed tasks, timestamps, stories, and effort estimation (e.g., story points in JIRA). On the other hand, VCS offers information about work traces, such a for the different versions of the artifacts, number and content of commits done by developers, artifacts evolution, users and their comments, and timestamps of each action.

Although different technologies exist in practice (e.g., Subversion, Mercurial, Git), the information contained in the logs can be roughly summarized by Table 1. The table displays an excerpt of a VCS log, i.e., a set of user commits, after having extracted and structured the data according to five attributes. The semantic of the attributes is a follows: *i) Id* is a unique identifier; *ii) Resource* is the resource that issued the commit; *iii) Date* is a timestamp associated to the time and date the commit was stored in the system; *iv) Comment* is a user comment on the changes they made; *v) Diff* is low-level information on the difference between the current version and the previous, for each file. Likewise, information about issues from ITS can be extracted presented in a tabular way. In this case, other attributes are more important. For example, we want to

Table 1: An excerpt of a VCS log data

Id	Resource	Date	Comment	Diff
1	John	2017-01-31 12:16:30	Create readme file	diff –git a/README.md b/README.md @@ -0,0 +1 @@ +# StoryMiningSoftwareRepositories
2	Mary	2017-02-01 10:13:51	Add a license	diff –git a/README b/README @@ -1,0 +2,3 @@ +The MIT License (MIT) + +Copyright (c) 2015 Mary+
3	Paul	2017-02-02 16:10:22	Updated the requirements.	diff –git a/README.md b/README.md @@ -1,4 +1,5 @@ + # string 1, string 2, string 3 diff –git a/requirements.txt b/requirements.txt @@ -0,0 +1 @@ +The software must solve the problems
4	Paul	2017-02-02 15:00:02	Implement new requirements	diff –git a/model.java b/model.java @@ -1,9 +1,10 @@ +public static methodA(){int newVal=0; @@ -21,10 +23,11 @@ + "1/0",,"0/0", diff –git a/test.java b/test.java @@ -0,0 +1,2 @@ +//test method A +testMethodA()

gather know about which the different status of the issues and how conversations around them take place.

Hence, studying the (co-)evolution of these two repositories can help extracting relevant knowledge about the software development process. Because the project managers can benefit from a process view to better analyze hidden aspects of the software development process, this thesis takes a process mining perspective on the problem. Thus, the following requirements are derived to address **RQ** according to the four aspects of a process [vdA16].

R1. Mining the time perspective. Given a software repository, how can we use project event data to 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, how can we use project event data to 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, how can we use project

event data to 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, how can we use project event data to 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.

3 State of the Art

When the research problem becomes section 2 and the fields are in section 3, then this part should be integrated with the part that is currently separate and before on each of the three fields. – Addressed.

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[2] 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 [RHL15] 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 Natural Language Processing (NLP). Works that use NLP can be found in both PM and MSR. In the Business Process Management (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 research combines ideas from the above mentioned areas to devise algorithms for mining the software development process. Table 2 lists the most relevant works in the literature and classifies reference sources and techniques that can support for addressing the four requirements.

Fill in table Literature vs. Requirements

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

Table 2: Classification of existing literature addressing the various aspects of mining the development process

	R1	R2	R3	R4
PM	Dotted Chart [SvdA07]	Decision mining [RvdA06]	Visualization techniques [BS13] Organizational mining [SvdA08] [SCJM15]	Bug fixing [PSvdB11], Workflow fragments [KRS06a, KRS06b]
TM	Information extraction [CL96]	Topic models [CTH16] Theory-generating case studies [LBGL16]	Social network [BGD+06] Survey [BKZ10] [dASB10]	Speech acts [DM13] [CRBS18] Natural language processing [FMP11]
MSR	Time series [RHL15] [HMCX14], Statistical analyses [OSGdS11]	Network analysis [DLL09], [ZN08], Language Models [AS13]	Email analysis [BGD+06], Role identification [YR07]	Exploratory studies [GPvD14]

3.1 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 2 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[3], Disco[4], minit[5], LANA Process Mining[6] 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

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

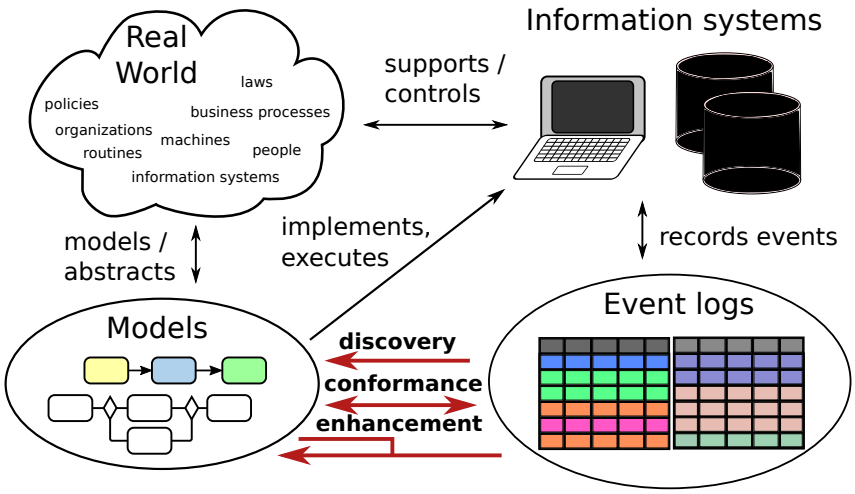

Figure 2: The process mining framework Adapted from [vdA16].

work with structured data [vDdMV+05, VBVV11]. 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 [VBVV11] required by process mining algorithms.

3.2 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 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 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[7].

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[8], Sumplify[9] and Online summarize tool[10].

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.

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

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 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[11], 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.

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

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.

3.3 Mining Software Repositories

Mining software repositories research has emerged in the last decade from the software engineering field. It focuses on analyzing patterns and trends in software. Main topics in MSR are about software quality and metrics, visual software analytics, bug analysis, communication among project members, defect tracking, predictions about software development, and software evolution.

Contributions in MSR use techniques from data mining, text mining and statistics. Typically, these methods focus on analyzing dependencies in the structure of the software artifacts. There are some existing works that can be taken into account as a starting point for understanding the development process. These works focus principally 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 [RHL15] and inter-dependencies between developers [LBGL16] are proposed. All these techniques allows to obtain several metrics and models from software repository data.

All in all, research on mining software repositories takes is mainly focused on metrics. As process mining research shows that analyzing the process gives relevant advantages over indicators and models that do not describe the process (cf. [vdA16]). Thus, the adoption of process mining techniques can help uncovering further information that is implicit in the data stored in software repositories.

4 Research Design and Methodology

4.1 Research Method

Information systems research is an interdisciplinary field of study that uses theories from social sciences, economics, and computer science. The field can be divided into two complementary paradigms: *behavioral science* and *design science*. Behavioral science aims at developing and justifying theories in order to explain or predict information systems phenomena [Gre06]. Design science focuses on the creation and evaluation of innovative design artifacts [HMPR04]. Figure 3 illustrates the design science approach as a process [PTRC08].

This doctoral thesis is employs both design and behavioral science. Following the approach suggested by [BSS18], real life data will be used for data-driven computationally-intensive theory development. This approach can be seen as a combination between behavioral science and design science. One the one hand there is a traditional theory development based on manual

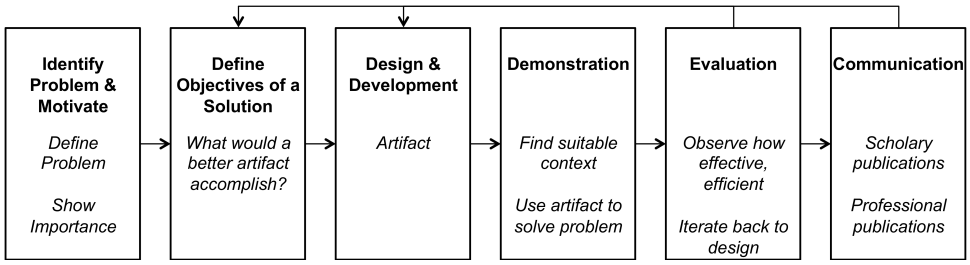

Figure 3: The Design Science Research process, adapted from [PTRC08]

coding, e.g. grounded theory methodology (GTM). One the other hand, trace data are used to automatically discover theory, e.g, computational theory discovery (CTD). In other words, this thesis uses real world data gathered from the SHAPE project[12] and Open-source Software (OSS) to develop novel artifacts which apply process mining methods to a new domain. This type of contribution is also referred to as exaptation [GH13].

With reference to Design Science Research (DSR) process in Figure 3, next sections show the design and development of the dataset and artifacts for addressing the requirements posed in Section 2.

i do not think that the guidelines work well here. better refer to the peffers process and to the recent article by Berente and Seidel.

4.2 Data Collection

This thesis includes the collection of and generation of datasets from real life software projects. These project data were collected both from industrial partners and from OSS. Logs from industrial partner are concerned with specific development activities (e.g., building railway interlocking-system software) and include a sufficient number of events that cover one software release. Logs from OSS were manually extracted by GitHub repositories. A tool has then been devised to parse these data and use them to populate a database. An original dataset is represented by Github pull requests. This dataset contains a list of manually annotated pull requests using the coding scheme by [MM16]. An additional output of this thesis will be the creation of a larger dataset using a machine learning approach that is able to automatically categorize pull requests in the classes given by [MM16].

4.3 Expected Outcome

In addition to an original dataset, this doctoral thesis is expected to produce outcomes that help the project manager analyze his software development processes. To this end new approaches in the form of artifacts [PTRC08] will be devised. These artifacts will serve as proofs-of-concept for the applicability of the research. In particular, the artifacts must address each of the four process aspects. Therefore, the outcome is categorized according to the requirements derived in Section 2 as follows.

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

Table 3: Available dataset

Log	Description
SHAPE	Industry VCS from software development in the railway domain
Github	Logs from real world OSS development
Github pull requests	Manually annotated pull requests from one real life open source project
Jira	Log data from ITS from industry partner
Asana	Generated dataset from project management tool of industrial partner
Kibana	Event traces from distributed process-agnostic enviroment used to handle communication among several processes

- **Mining the Time Perspective (R1)**. Artifact that allows for gaining transparency on the time perspective of software development work. For instance, and artifact that explicits what are the durations of tasks and when are actual tasks executed.
- **Mining the Case Perspective (R2)**. Artifact that allows for gaining transparency on the case perspective of software development work. For instance, an artifact that explicits how the different cases are handled and whether there is any dependency of work that might influence the case execution.
- **Mining the Organizational Perspective (R3)**. Artifact that allows for gaining transparency on the organizations aspect of software development work. For instance, an artifact that explicits which are the actual roles software developers are covering and which is the organizational network.
- **Mining the Control-Flow Perspective (R4)**. Artifact that allows for gaining transparency on the control-flow perspective of software development work. For instance, an artifact that allows for abstracting from data which is order of activities that are executed to accomplish a certain task or goal in software development.

5 Preliminary Results

Next, I present efforts done towards addressing the requirements presented in Section 2.

5.1 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.

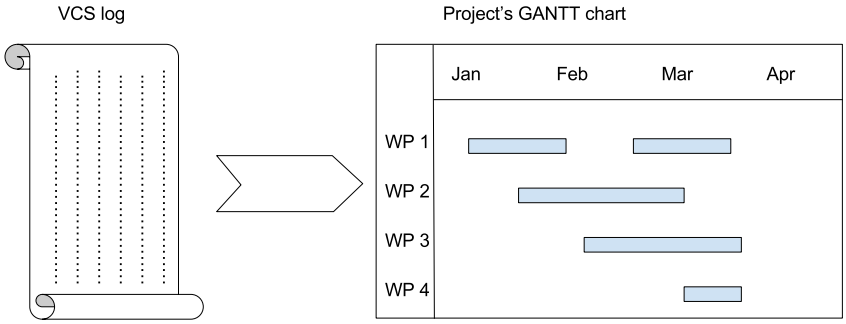

Figure 4: Mining the time perspective of software development into a Gantt chart

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].

5.2 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) 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.

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

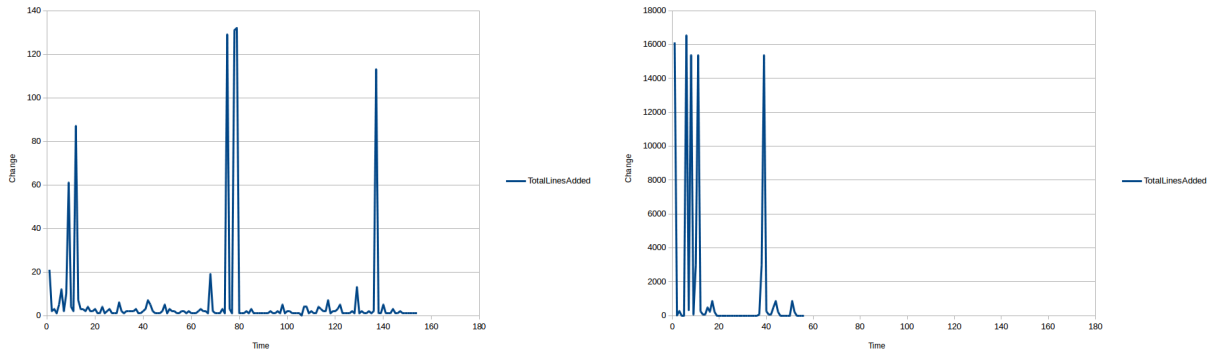

Figure 5: Evolution of files in a version control systems: LOC changes over time

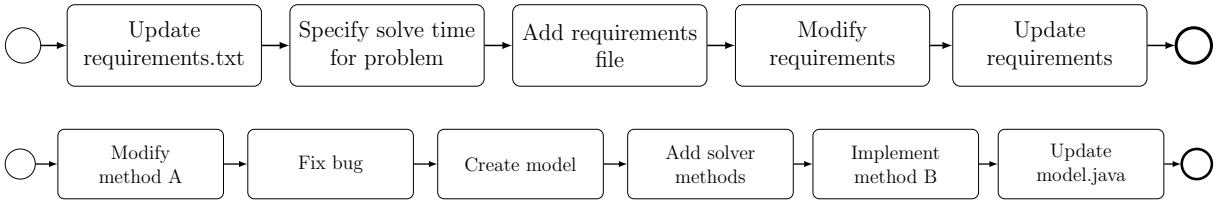

Figure 6: Processes of two work-dependent files. They may seem different but they co-evolve in time and space, i.e., they describe the same *case*.

have been combined together an mined as a story. Among other things, the technique allows to profile existing software development projects.

5.3 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

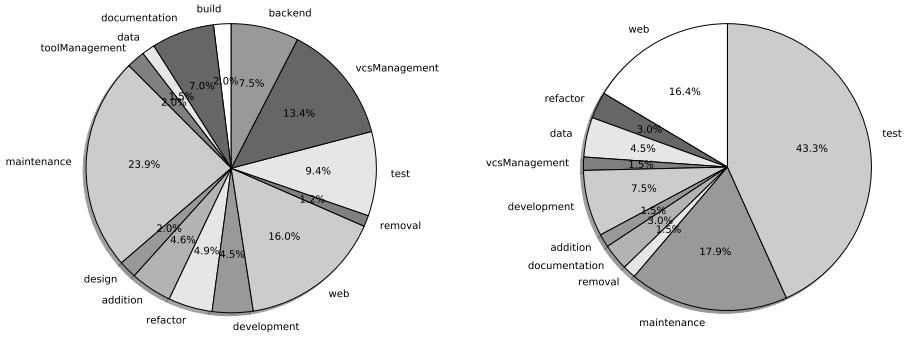

Figure 7: Profiles of a developer and a tester, from left to right.

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].

6 Next Steps and Finalization of the Thesis

6.1 Mining the Control-Flow Perspective (R4)

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 [Akb18]. The nature of software development process makes it difficult to recognize recurrent activities when only VCS are analyzed.

6.2 Finalization of the Thesis

The expected result for the above-mentioned work is an artifact that is able to explicitly show what are process patterns that lead to the generation of an idea. This will serve as a proof-of-concept that event data from projects can be used to increase understanding of process flow perspective. The overall thesis will be finalized by evaluating the results both against the real world dataset and project managers. There is ongoing work in which a questionnaire has been answered by industry partners. Questions pertains activities, tools and habits within a software development company. These information can be used to evaluate the artifacts developed by this thesis.

7 Dissertation Relevance and Publication Plan

7.1 Implications for Research

This proposal 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. Ongoing work aims 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.

Change?!

7.2 Work Plan and Dissemination

Target venues for communicating the results are BPM and software engineering journals and conferences. So far, the following publications have been achieved that directly address the requirements in Section 2.

Mining the Time Perspective (**R1**):

put all, and clarify to which part of the thesis they contribute

- **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]

Mining the Case Perspective (**R2**):

- **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). [BRd+17]

Mining the Organizational Perspective (**R3**):

- 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). [AAT+16]

Publications that address the requirements in Section 2 indirectly.

- **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). [BCH+15] – Related to **R1** & **R3**.
- **Bala, 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). [BHS+16] – Related to **R1** & **R3**.

Doctoral consortium and vision paper presenting the concepts of this research proposal.

- **Bala, S.**: *Mining projects from structured and unstructured data*. In: CEUR Workshop Proceedings (2017). [Bal17]
- **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 [BM18]

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. – Related to **R1**, **R3** and **R4**.

- 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. – Related to **R4**.
- 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. – Related to **R2** & **R4**.

Other work in the BPM area.

- 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). [WB18]

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.

[Akb18] Shirin Akbarinasaji. Prioritizing lingering bugs. *ACM SIGSOFT Softw. Eng. Notes*, 43(1):1–6, 2018.

[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.

[Bal17] S. Bala. Mining projects from structured and unstructured data. In *CEUR Workshop Proc.*, volume 1859, 2017.

[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.

[BHS+16] Saimir Bala, Giray Havur, Simon Sperl, Simon Steyskal, Alois Haselböck, Jan Mendling, and Axel Polleres. SHAPEworks: A BPMS extension for complex process management. In *CEUR Workshop Proc.*, volume 1789, pages 50–55, 2016.

[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.

[BSS18] Nicholas Berente, Stefan Seidel, and Hani Safadi. Data-Driven Computationally-Intensive Theory Development. *Inf. Syst. Res.*, 2018.

[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. In Ernest Teniente and Matthias Weidlich, editors, *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.

[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.

[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.

[GPvD14] Georgios Gousios, Martin Pinzger, and Arie van Deursen. An exploratory study of the pull-based software development model. In *Proc. 36th Int. Conf. Softw. Eng.*, pages 345–355. ACM, 2014.

[Gre06] Shirley Gregor. The nature of theory in information systems. *MIS Q.*, pages 611–642, 2006.

[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.

[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.

[LWR14] Andreas Lanz, Barbara Weber, and Manfred Reichert. Time patterns for process-aware information systems. *Requir. Eng.*, 19(2):113–141, 2014.

[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.

[MM16] Ann Majchrzak and Arvind Malhotra. Effect of Knowledge-Sharing Trajectories on Innovative Outcomes in Temporary Online Crowds. *Inf. Syst. Res.*, 27(4):685–703, nov 2016.

[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.

[RHL15] Jukka Ruohonen, Sami Hyrynsalmi, and Ville Leppänen. Time series trends in software evolution. *J. Softw. Evol. Process*, 27(12):990–1015, nov 2015.

[RvdA06] Anne Rozinat and Wil M P van der Aalst. Decision mining in ProM. In *Int. Conf. Bus. Process Manag.*, pages 420–425. Springer, 2006.

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

[SCJM16] Stefan Schönig, Cristina Cabanillas, Stefan Jablonski, and Jan Mendling. A framework for efficiently mining the organisational perspective of business processes. *Decis. Support Syst.*, 89:87–97, 2016.

[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.

[SvdA07] Minseok Song and Wil M P van der Aalst. Supporting process mining by showing events at a glance. In *Proc. 17th Annu. Work. Inf. Technol. Syst.*, pages 139–145, 2007.

[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.

[VBVV11] H M W Verbeek, Joos C A M Buijs, Boudewijn F Van Dongen, and Wil M P Van Der Aalst. Xes, xesame, and prom 6. In *Inf. Syst. Evol.*, pages 60–75. Springer, 2011.

[vdA05] W M P van der Aalst. Business alignment: using process mining as a tool for Delta analysis and conformance testing. *Requir. Eng.*, 10(3):198–211, 2005.

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

[vdARW+07] W 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 process mining: An industrial application. *Inf. Syst.*, 32(5):713–732, 2007.

[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.

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

[WB18] Bartosz Woliński and Saimir Bala. Comprehensive Business Process Management at Siemens: Implementing Business Process Excellence. In Jan vom Brocke and Jan Mendling, editors, *Business Process Management Cases: Digital Innovation and Business Transformation in Practice*, pages 111–124. Springer International Publishing, Cham, 2018.

[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.

[YR07] Liguo Yu and Srini Ramaswamy. Mining CVS repositories to understand open-source project developer roles. *Proc. - ICSE 2007 Work. Fourth Int. Work. Min. Softw. Repos. MSR 2007*, pages 7–10, 2007.

[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.