Data From Configuration Management Tools As Sources For Software Process Mining

Jana Šamalíková, Rob Kusters, Jos Trienekens, Ton Weijters,
IS, IE&IS
University of Technology Eindhoven
Eindhoven, The Netherlands
j.j.m.trienekens@tue.nl

Abstract—**Process mining has proven to be a valuable approach that provides new and objective insights into processes within organizations. Based on sets of well-structured data, the underlying 'actual' processes can be extracted and process models can be constructed automatically, i.e., the process model can be 'mined'. Successful process mining depends on the availability of well-structured and suitable data. This paper investigates the potential of software configuration management (SCM) and SCM- tools for software process mining. In a validation section, data collected by a SCM tool in practice are used to apply process-mining techniques on a particular software process, i.e., a Change Control Board (CCB) process in a large industrial company. Application of process mining techniques revealed that although people tend to believe that formally specified and well-documented processes are followed, the 'actual' process in practice is different. Control-flow discovery revealed that in the CCB process in most of the cases, i.e., 70%, an important CCB task 'Analysis' was skipped.**

Keywords-software configuration management; process mining; validation

I. INTRODUCTION

Software process improvement is a cyclic activity during which improvements are planned, applied and their impact is analyzed. Before planning improvements, the current state of the software process has to be assessed. Nowadays, process assessment is based on process descriptions obtained from quality manuals and process standards, as well as on information that is derived from interviews and brainstorm sessions with representative software developers [1]. A promising alternative way to obtain close-to-the-reality process descriptions is process mining. Process mining has proven to be a valuable approach that provides new and objective insights into the way processes are actually carried out within organizations [2], [3], [4]. In a number of case studies, event logs were created from data that were automatically recorded by process-enactment systems. Based on sets of well-prepared data, the underlying 'actual' processes can be extracted and process models can be constructed automatically, i.e., the process model can be 'mined'. Software developing organizations use various types of software tools to support and manage their software development processes, e.g., configuration management tools, inspection tools and testing tools. In this paper, we will focus on software configuration management (SCM) and SCM tools as a good example of process support tools. "Software Configuration Management (SCM) tools are 'the' real process-centered tools due to their ability to model, support and enact the processes by which all software developers are supposed to manipulate the product" [6]. In SCM tools, several types of data about the development processes are stored, such as data about the tasks or activities that are carried out by the developers, and data about the creation of and the changes on software components. Various SCM tools can provide so-called 'audit trails' of the collected data. However, data logged during software development are often not intended for process mining. It is necessary to investigate whether the data logged by SCM tools could in principle be used for process mining. For example regarding process mining, these data should have a notion of a process (e.g., time-stamp data), but they should also offer the possibility to identify particular objects with particular attributes that are handled by a process. In process mining, these objects are called 'cases'. The aim of this paper is to evaluate selected SCM tools regarding their potential to provide data for process mining. The paper is organized as follows. In Section 2, a structured overview is derived of the data types that are required for the application of process mining techniques. Section 3 identifies, from the viewpoint of process mining, the data that play a central role in the software processes, which are supported by SCM. Subsequently, Section 4 investigates the potential of selected SCM tools to provide the required data for software process mining. Section 5 finalizes the paper with conclusions and points to future work to be done.

II. PROCESS MINING TECHNIQUES AND THE DATA THAT THEY REQUIRE

Process mining techniques attempt to extract non-trivial and useful process information from so-called audit trails. In order to be useful for process mining an audit-trail has to be transformed into an event log. Currently a variety of process mining techniques is available and has been applied in practice [7]. These mining perspectives and mining techniques are described in more detail in the next subsections.

A. The control-flow perspective and the data required

From the control-flow perspective process model discovery techniques are applied to discover a process model that specifies the relations between the activities in an event log. The resulting mined process model is a 'real' *model* that depicts the possible flows that were followed by particular cases in an event log. Subsequently, conformance checking can be carried out in order to compare the 'official' process model with the 'real' process model as it is stored in the event log. The data requirements for process model discovery and conformance checking are: an event log containing ordered sequences of events (i.e., of type 'start' or 'end') where each event refers to a case and each event refers to an activity.

B. *The performance perspective and the data required*

The performance perspective focuses on the performance of processes. Mining from the performance perspective calculates the timeliness of cases, the execution times of tasks, and reveals the bottlenecks in processes by calculating waiting times (synchronization time). The throughput time calculation technique can be applied if a timestamp of an end event is recorded. This mining technique implements an event log replay and therefore a 'real' process model is required that captures the behavior in the event log. Applying performance sequence analysis techniques reveals sequence patterns that are present in an event log. Based on this application, it is possible to determine which sequence patterns are common and which patterns are less frequent. The data requirements are respectively: an event log containing ordered sequences of events where each event refers to a case and each event refers to an activity, and timestamps of the start and end event of activities.

C. *The organizational perspective and the data required*

The organizational perspective in process mining focuses on the analysis of the interrelations among persons (or groups of individuals) who are performing the activities in a process, i.e., their social network. Social network analysis techniques focus on the discovery and examination of the social interrelations. Several types of social network mining are possible with different types of results; such as a work transfer model, a subcontracting model, a collaboration model, and a similar activity model. Data requirements are respectively: an event log containing ordered sequences of activities, where each activity refers to a case, and each case refers to an activity and to its originator.

D. *The case perspective and the data required*

A case is an 'object' that is being handled by a process. Case attributes represent various case properties and together they specify a case. Two types of case attributes exist: the stable attributes (e.g., a defect id. or a phase in which a defect was detected) and the dynamic attributes, which become available during a process (e.g., a defect priority). Both types of case attributes enable process mining from the case perspective. The case perspective shows the process based on the case types (i.e., a set of cases with the same or similar attributes), discovering the data dependencies that affect the routing of a case. Within the case perspective decision analysis techniques are being applied. These techniques focus at the way case attributes influence the choices that are being made in the process, based on past process executions. The data requirements of decision analysis techniques are respectively: an event log containing ordered sequences of events where each event refers to a case and each event refers to an activity, and the case attributes that are modified during the execution of activities. Table I summarizes the event log data requirements per process mining technique.

TABLE I. EVENT LOG DATA REQUIREMENTS

Required data / Process mining techniques	Event log data					
	Case id	Activity	Event type	Timestamps	Originator	Additional attributes
Process model discovery	x	x	x			
Conformance checking	x	x	x			
Throughput time calculation, bottleneck analysis	x	x	x	x		
Performance sequence analysis	x	x	x	x		
Social network analysis	x	x	x		x	
Decision analysis	x	x	x			x

III. SOFTWARE PROCESSES (AND THEIR CASES) THAT ARE SUPPORTED BY SOFTWARE CONFIGURATION MANAGEMENT

A key aspect of SCM in software development is version management. A software component put under version control is called a configuration item (CI). Besides managing the versions of CIs in the software development process, SCM supports the change control process, the problem management, and the requirements management process. We selected these four processes as well-structured and formally described processes. In the following, we describe these processes in more detail and we will investigate whether the recorded data, i.e., their primary 'cases', can be used for process mining.

E. Software development process

The software development process contains the activities of developers that are performed during the software lifecycle. The process contains activities for requirements analysis, design, coding, integration, testing, and installation of software products. During these activities, various software components or CI's are produced [5]. Components are put under version control and are controlled by SCM. Therefore, we identify a software component as a case that is handled by the software development process.

F. Change control process

The change control process manages the change requests to a software product. Configuration management is in general under control of a CCB [1]. A CCB coordinates changes made to CI's. The CCB tracks and records the status of each change request, e.g., labeled as defects, from its entry until its exit from the CCB process. The CCB distributes tasks related to the required changes of CI's and evaluates the outcomes of the executed tasks with respect to the requests. A so-called audit trail is available, where each modification, the reason for the modification, and the authorization of the modification can be traced. Based on the foregoing we identify a change request as a case that is handled by the change control process.

G. Problem resolution process

The problem resolution process is established for handling problems detected in software products and activities. The process ensures that all detected problems are promptly reported and entered into the problem resolution process. A unique identification of a problem is assigned during this activity. A problem report is to be used as part of a close loop process, from detection of a problem through investigation, analysis, and resolution, and later for trend detection across problems. Status is tracked and reported, and records of problem reports are maintained. A problem report can be considered as a case that is handled by the problem resolution process.

H. Requirements management process

Requirements management is the process of eliciting, documenting, analyzing, tracing, prioritizing and agreeing on requirements with the relevant stakeholders. Requirements management includes the ensuring that requirements are well defined, agreed to by relevant parties and modified in accordance with defined procedures. The modification procedures must ensure that later changes are incorporated properly and that project plans are updated accordingly. In SCM, requirement specifications are considered as common CI's.

TABLE II. CASE IDENTIFICATION

Processes supported by SCM	Cases handled by the processes
Software development process	Component
Change control process	Change request
Problem resolution process	Problem report
Requirements management process	Requirement specification

Therefore, we consider a requirement specification as a case that is handled by the requirements management process. Table II summarizes the software development processes that are supported by SCM together with the cases that are being handled by these processes. In the next section, we will investigate the potential of four selected SCM tools with respect to the data that they can provide to software process mining.

IV. SCM TOOLS AS DATA SOURCES FOR PROCESS MINING

Nowadays, a variety of SCM tools exists. We can distinguish SCM tools that offer particular basic functionalities, such as version management, but also 'more rich' SCM tools that are called 'SCM suites'. The latter support a range of additional functionalities such as change request control, problem resolution control, build and release management, requirements management, project and task management, and even workflow management. SCM tools assist developers in their collaborative work, support the maintenance of software products, storing their history, providing a stable development environment and coordinating simultaneous product changes [8]. Data recording in SCM tools is reflected by so-called audit trails. We will investigate the content and the data structure of these audit trails, and to what extend these data can be used as event log data for software process mining. We will analyze selected SCM tools with respect to their ability to provide data for process mining techniques.

I. SCM tool selection and analysis

Based on a survey we selected the two most commonly used SCM tools in order to analyze their audit trails with respect to process mining. These tools, respectively Subversion and Microsoft Visual SourceSafe, are examples of basic version management tools. Additionally, we selected CM Synergy & Change Synergy and HP Quality Center tools as examples of the more 'richer' tools. These tools provide also support for change request control, problem resolution control and requirements management. Regarding the first mentioned SCM suite we identified the part that is called Rational Synergy as a basic version management tool. Consequently, we had three basis version management systems. In our analysis, we will focus on the one hand on particular software processes that are supported by these tools, respectively: the software development process, the change control process, the problem resolution process and the requirements management process (see section III, Table II). On the other hand, we will focus on particular data that are recorded by the selected SCM tools, respectively: case id., task, event type, time stamp, task originator, additional attributes of cases, as identified in Section II, Table I.

1) Basic version management tools

Basic version management tools offer basic functions of version management such as managing the evolution of configuration items, file sharing, controlling concurrent work, history tracking, and security and access control. We analyzed three basic version management tools more in depth, respectively Subversion, Microsoft Visual SourceSafe, and Rational Synergy (the basic version management part of the 'SCM suite' CM Synergy and Change Synergy). Because of similarities in the results we will describe in this paper only the analysis results of Subversion. Subversion is a basic open-source file-based version management tool, which tracks the changes to files and directories under version control. After a revision, i.e., a change of a file is made, the file is committed to a repository of files, and a log-entry is created in the tool log (the so-called 'change log'). Subversion supports the software development process. The software development process is reflected by sequences of document changes, with respect to particular components, in a change log. Table III shows the identified event log elements. Please note that there are no additional attributes of cases recorded.

TABLE III. EVENT LOG ELEMENTS IN THE SUBVERSION LOG

Event log element	Case id	Acti vity	Event type	Time stamp	Origi nator
software develop-ment process	com-ponent	docu-ment type	end	of check-in	user check-ing-in

In the following subsection, we will analyze two 'SCM suites', respectively CM Rational Synergy and HP Quality Center.

2) *Integrated SCM tools*

The so-called 'SCM suites' are in fact integrated tools that incorporate various functions of version management in combination with enhanced functionalities such as change control, build management, problem issue management, process and workflow management, baseline management and requirements management.

a) *CM Rational Synergy and Rational Change*

This is a SCM tool that provides integration and synergy between the different SCM functions. It is an integrated tool that supports distributed development on a unified change, configuration and release management platform. The 'suite' consists of a Rational Synergy part and a Rational Change part. The Rational Change part is a web-based, integrated, change management tool for tracking and reporting changes of cases. Rational Change supports in particular three processes, respectively the change control process, the problem resolution process and the requirements management process. The cases handled by these processes are respectively change requests, problem reports and requirements specification requests. The identified event log elements are shown in Table IV. Case id corresponds to a case, Activity corresponds to a status adjustment of a case, Event type: 'start' and 'end', Timestamp corresponds to a commit date and time, additional attributes of cases such as priority, severity. Regarding originator: an actual originator of an activity is not provided.

TABLE IV. EVENT LOG ELEMENTS IN THE RATIONAL CHANGE LOG

Event log element	Case id	A cti vit y	Eve nt Typ e	Time stamp	Add. attributes of cases
change control process problem resolu-tion process req. mngt. process	change control request pro-blem report req. spec. request	s t a t u s	start end	of activities e.g.,: submit analysis resolu-tion evalua-tion	e.g.,: priority, severity, phase detected, phase caused

b) *HP Quality Center*

HP Quality Center supports the change control process, the problem resolution process and the requirements management process. As shown in Table V, the event-log elements are identified as follows: for each of the cases a Case identification number is stored, Activity corresponds to the status adjustment of these cases (the status of a change request is adjusted as a result of executing an activity), Event-type: start or end, Timestamps: timestamp of the end event of a related activity. Additional attributes: for each supported process a rich variety of additional case attributes are recorded. These attributes describe the cases in more detail than the previously discussed Rational Change tool and as a consequence allow for more detailed analysis. Pleae note that although the originator is not shown in Table V, information on person/department executing the activities is recorded.

J. *SCM tools and the process mining techniques that can be supported by them*

Data needed for process mining are (partially) available in the so-called audit trails of the selected and analyzed SCM tools. These process data can be used as a basis for the construction of the event logs needed for process mining. We discovered that all the selected SCM tools, provide the minimal data requirements for process mining, i.e., the identification of a case, an activity, and the type of an event.

TABLE V. EVENT LOG ELEMENTS IN THE HP QUALITY CENTER LOG

Event log element	Case id	Activity	Event type	Time stamp	Add. Attributes of cases
Change Control Process	change request	status	start, end	history date, end event of a related activity	change category, root cause, priority, phase detected, estimated cost, responsible team, affected component
Prob. Resolution Process	problem report		end		problem category, related component, criticality, phase detected, responsible team
Req. mngt. Process	spec. request		end		priority, linked test, responsible team, related component

Furthermore, both the basic version management tools and the ‘SCM suite’ HP Quality Center provide originator information. The ‘SCM suites’ also provide various additional case attributes. Based on our findings in the foregoing sections, we summarize our SCM tool analysis results in Table VI, i.e., which shows the interrelations between the SCM tools (and the supported processes) and the process mining techniques. The mining techniques process model discovery, and conformance checking can be supported by each of the selected SCM tools. The process mining technique decision analysis is only possible with the SCM tools that provide additional case attributes, i.e., Rational Change and HP Quality Center. Rational Change is the only SCM tool that records both Start and End events of tasks, thereby enabling the process mining techniques Performance Sequence Analysis, Throughput Time Calculation and Bottleneck Analysis. The process mining technique social network analysis is possible with each of the SCM tools that record task originator (i.e., Subversion, VSS, Rational Synergy, and HP Quality Center).

TABLE VI. INTERRELATIONS BETWEEN PM TECHNIQUES AND SCM TOOLS

Process mining technique / SCM tool (and process supported)		Process model discovery	Conformance checking	Decision analysis	Throughput time calculation, bottleneck analysis	Performance sequence analysis	Social network analysis
Subversion	software development process	x	x		x		x
Visual SourceSafe	software development process	x	x				x
Rational Synergy	software development process	x	x				x
Rational Change - Change Synergy	change control, problem resolution, requirements management process	x	x	x	x	x	
HP Quality Center	change control, problem resolution, requirements management process	x	x	x			x

Summarizing we can state that the selected SCM tools meet important event log requirements for process mining. The basic version management tools, such as Subversion, provide software development process data. However, the integrated ‘SCM suites’, such as Rational Synergy, support particular software processes as well, respectively the change control process, the problem resolution and the requirements management process.

V. VALIDATION

In order to validate our approach, we used the data collected during ten middleware embedded-software projects of a large industrial company in The Netherlands. The detailed results of the validation are presented in [1]. In this section we address some main results of the case study. The industrial company develops software components for consumer electronic devices. The process under study is the CCB process. The CCB is an organizational unit that handles problem reports, change and implementation requests identified during software development. These are further referred to as defects. A CI’s defect is detected and submitted. The developer assigns attributes to the defect (e.g., priority, severity). After that, the defect is either processed by the main sequence of tasks Analysis, Resolution, Evaluation, Conclusion or it is evaluated by the CCB. If the CCB evaluates the defect, it sends the defect to a required task depending on the need, with the following possibilities: the defect is redirected to the Concluded task in case the defect is found duplicated, expected to be repaired in a next release, or out of the scope of the functionality required. The defect is redirected to tasks Analysis, Resolution or Evaluation depending on the need. When the task Analysis, Resolution or Evaluation is completed, one of the four possibilities is chosen: if the task's execution is successful, then an important defect is directed to the CCB and it waits to be redirected again to the next task; if the task's execution is successful, then a less important defect continues with the next task of the main sequence of tasks; if the task was not successfully executed, then an important defect is returned to the CCB for a re-evaluation. Once all the tasks of the CCB process have been successfully carried out, the defect is closed.

The software development team collects defect data and these are stored in the status database recorded by a Rational Change SCM tool. A quality assurance specialist creates copies (snapshots) of the status CCB database content on a weekly basis. The snapshots follow the evolution of the handling of the defects by the CCB. The snapshots include the following information: the date when the snapshot was taken, identification of a subsystem from which the snapshot was taken, identification of the defect, priority and severity of the defect, type of the defect, the actual status of the defect, identification of a team responsible for resolving the defect, timestamps of the start event of the tasks of the main sequence, timestamps of the end event of the tasks of the main sequence and a timestamp of the date of an update of the status database. We studied whether it is possible to use such data to discover and construct underlying process models. We identified the following event log elements. Case id (a defect is identified as a case), a Case id is retrieved from a Problem_nr data field that uniquely identified the defect. Activities are executed when they handle a case during a process. As a result of executing an activity, the status of the defect changes, therefore activities have been derived from the status field. Event-type: in the snapshots, only the event types start and end are used. Timestamp have been extracted from the fields in the snapshots that store time information. Originator is not available due to the fact that the snapshots only provide the information about the team responsible for resolving the defect. Additional attributes are derived from the priority, severity, request type, life-cycle phase during which a defect had been discovered. Based on the identified event log elements, we identified process mining techniques that are possible to be applied to the data. We applied respectively control-flow discovery and conformance checking. We applied the mentioned techniques to the event-log filtered on additional case attributes, namely priority, severity, request type and life-cycle phase. The control-flow discovery revealed that the official CCB process as described above was not followed. Furthermore, conformance checking of the CCB process revealed that in most of the cases (70%) the Analysis task is skipped and the cases are being directly resolved. Moreover, we compared the duration of tasks and the total throughput time during different lifecycle phases. The results showed that the duration of the validation tasks involving external stakeholders are longer than the verification tasks performed without the external involvement. Application of process mining techniques revealed that although people tend to believe that specified and well-documented processes are followed, the real practice is different., and that process mining techniques can provide useful insights into the software development process.

VI. Conclusions

This paper investigates the application of process-mining techniques in the software development domain. We addressed the suitability of particular software development support tools, i.e., SCM tools, to provide process data. Our research provides an original view at process mining literature from a data requirements perspective: 'different process mining techniques require different data', and can form a basis for the development of new functionalities, i.e., process mining support, to develop a future generation of SCM tools. We addressed some main results of a case study, to indicate which process mining techniques can make use of particular SCM tool data to get in-depth insights into 'actual' software development processes.

References

[1] J. Samalikova, R. Kusters, J. Trienekens, T. Weijters, and P. Siemons, "Towards objective software process information: experiences from a case study," Software Quality Journal, vol. 19, Jan. 2011, pp. 101-120.

[2] A. Weijters, W. van der Aalst and A.K. Alves de Medeiros, "Process Mining with the Heuristics Miner Algorithm," BETA Working Paper Series, WP 166, Eindhoven University of Technology, Eindhoven, 2006, pp. 1-18.

[3] W. van der Aalst et al, "Business process mining: An industrial application," Information Systems, 32(5), 2007, pp. 713-732.

[4] A. Rozinat., and W. M. P. van der Aalst, "Decision Mining in ProM", in S Dustdar and A Sheth eds., Business Process Management., Lecture Notes in Computer Science, Berlin, Springer, 2006, pp. 420-425.

[5] J. Samalikova, R Kusters, J Trienekens and T Weijters. "Information gathering in software process assessment," In Proceedings of the Information Systems IADIS Conference, Berlin, Germany, 2012, pp. 43-52.

[6] R. Conradi, A. Fuggetta and M. Jaccheri. "Six theses on software process research". Software Process Technology, 1998, pp. 100-104.

[7] B.F. van Dongen et al, "The ProM Framework: A New Era in Process Mining Tool Support", in Applications and Theory of Petri Nets Lecture Notes in Computer Science, Berlin, Springer, 2005, pp. 444-454.

[8] J. Estublier, "Software Configuration Management: a roadmap", in: Proceedings of the Conference on the Future of Software Engineering, Limerick, Ireland, 2000, pp. 279-289.